

Hitachi 16-Bit Single-Chip Microcomputer

H8S/2194 Series, H8S/2194C Series,
H8S/2194 F-ZTAT™, H8S/2194C F-ZTAT™

H8S/2194, HD6432194, HD64F2194,
H8S/2193, HD6432193
H8S/2192, HD6432192
H8S/2191, HD6432191
H8S/2194C, HD6432194C, HD64F2194C,
H8S/2194B, HD6432194B
H8S/2194A, HD6432194A

Hardware Manual

ADE-602-160A

Rev. 2.0

11/10/00

Hitachi, Ltd.

HITACHI



Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi's or any third party's patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party's rights, including intellectual property rights, in connection with use of the information contained in this document.
2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.
3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi's sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.
4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.
5. This product is not designed to be radiation resistant.
6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.
7. Contact Hitachi's sales office for any questions regarding this document or Hitachi semiconductor products.

Main Revisions and Additions in this Edition

| Page | Item | Revisions (See Manual for Details) |
|--------------------------|--|--|
| All pages of this manual | | Amendments due to introduction of the H8S/2194C series |
| 5 | 1.1 Overview | Table 1.1 Features Memory and Product lineup amended |
| 14 | 1.4 Differences between H8S/2194C Series and H8S/2194 Series | Added |
| All pages of section 2 | | Notes on TAS instruction added |
| 31 | 2.6.1 Overview | Table 2.1 Instruction Classification Notes 3 added |
| 65, 66 | 3.4 Address Map in Each Operating Mode | Address maps for the H8S/2194C series added |
| 68 | 4.1 Overview | Table 4.1 Internal Chip Status in Each Mode Timer L, PSU, 12-bit PWM added Sleep and Watch modes of I/O amended |
| 79 | 4.4.1 Sleep Mode | Description amended Other supporting modules (excluding the servo circuit and 12-bit PWM) do not stop. |
| 80 | 4.5.1 Module Stop Mode | Table 4.4 MSTP Bits and Corresponding On-Chip Supporting Modules Module corresponding to the MSTP1 bit amended |
| 119 | 6.4.5 Interrupt Response Times | Table 6.8 Interrupt Response Times Note 2 amended |
| 121 | 6.5.4 When NMI is Disabled | Added |
| 126 | 7.2.3 Flash Memory Operating Modes | Figure 7.3 Flash Memory Mode Transitions Amended |
| 127 | | Figure 7.4 Boot Mode Amended |
| 131 | 7.3.1 Flash Memory Control Register 1 (FLMCR1) | Description amended FLMCR1 is initialized by a reset, in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), or when a low level is input to the FWE pin. |

| Page | Item | Revisions (See Manual for Details) |
|------|--|--|
| 134 | 7.3.2 Flash Memory Control Register 2 (FLMCR2) | Description amended The ESU and PSU bits are cleared to 0 in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), hardware protect mode, and software protect mode. |
| 136 | 7.3.3 Erase Block Registers 1 and 2 (EBR1, EBR2) | Description amended EBR1 and EBR2 are each initialized to H'00 by a reset, in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), when a low level is input to the FWE pin, or when a high level is input to the FWE pin and the SWE bit in FLMCR1 is not set. Table 7.4 Flash Memory Erase Blocks EB3 address amended |
| 138 | 7.4 On-Board Programming Modes | Table 7.5 Setting On-Board Programming Modes MD0 pin level in use program mode amended |
| 140 | 7.4.1 Boot Mode | Figure 7.8 Boot Mode Execution Procedure Flow amended |
| 141 | | Table 7.6 System Clock Frequencies for which Automatic Adjustment of This LSI Bit Rate is Possible 2400-bps transfer bit rate deleted |
| 142 | | Figure 7.10 RAM Areas in Boot Mode Programming control program area amended |
| 145 | 7.5.1 Program Mode | Description amended (For details, see the flowchart in figure 7.12.) |
| 146 | 7.5.2 Program-Verify Mode | Description amended (For details, see the flowchart in figure 7.12.) |
| 148 | 7.5.3 Erase Mode | Description amended (For details, see the flowchart in figure 7.13.) |
| | 7.5.4 Erase-Verify Mode | Description amended (For details, see the flowchart in figure 7.13.) |
| 150 | 7.6.1 Hardware Protection | Table 7.7 Hardware Protection Reset/standby protection description amended |
| 152 | 7.6.3 Error Protection | FLER bit setting condition (3) amended Figure 7.14 Flash Memory State Transitions amended |
| 153 | 7.7 Interrupt Handling when Programming/Erasing Flash Memory | Note 1 amended |
| 154 | 7.8.2 Socket Adapters and Memory Map | Table 7.9 Socket Adapter Product Codes amended |

| Page | Item | Revisions (See Manual for Details) |
|-----------------|--|--|
| 166 | 7.8.9 Programmer Mode Transition Time | Figure 7.23 Oscillation Stabilization Time, Boot Program Transfer Time, and Power Supply Fall Sequence Vcc timing amended |
| 169 | 7.10 Note on Switching from F-ZTAT Version to Mask ROM Version | Added |
| — | Section 8 ROM (H8S/2194C Series) | Added |
| 221 | 9.1 Overview | Description amended due to introduction of the H8S/2194C series |
| 304 | 14.1.2 Block Diagram | Figure 14.1 Block Diagram of the Timer J $\phi/1024$ clock source (for H8S/2194C series) added |
| 308 | 14.2.1 Timer Mode Register J (TMJ) | Bits 3 and 2 Description amended |
| 311, 312 | 14.2.2 Timer J Control Register (TMJC) | Bit 0 description amended |
| 336 | 16.2.1 Timer R Mode Register 1 (TMRM1) | Bit 0 description amended |
| 416 | 20.2.1 12-Bit PWM Control Registers (CPWCR, DPWCR) | Initialization description amended |
| 419 | 20.2.2 12-Bit PWM Data Registers (CPWDR, DPWDR) | Initialization description amended |
| 420 | 20.2.3 Module stop Control Register (MSTPCR) | Added |
| 443 | 23.1.2 Block Diagram | Figure 23.1 Block Diagram of SCI1 Register names amended |
| 454 | 23.2.7 Serial Status Register (SSR1) | Bit 7: Clearing conditions amended |
| 520 | 25.1.4 Register Configuration | Table 25.2 Register Configuration Note 2 description amended |
| 522 | 25.2.1 I ² C Bus Data Register (ICDR) | Description amended |
| 531, 534 to 536 | 25.2.5 I ² C Bus Control Register (ICCR) | Bit 7 description amended Bit 1 description amended |
| 541 | 25.2.6 I ² C Bus Status Register (ICSR) | Bit 4 description amended |
| 544 | 25.2.7 Serial/Timer Control Register (STCR) | Bit 5 description amended |

| Page | Item | Revisions (See Manual for Details) |
|------------|---|--|
| 547 to 549 | 25.3.2 Master Transmit Operation | Description amended |
| 550 to 552 | 25.3.3 Master Receive Operation | |
| 556 | 25.3.5 Slave Transmit Operation | |
| 561 | 25.3.8 Sample Flowcharts | Figure 25.14 Flowchart for Master Transmit Mode (Example) amended |
| 562 | | Figure 25.15 Flowchart for Master Receive Mode (Example) amended |
| 565, 566 | 25.3.9 Initialization of Internal State | Added |
| 570 to 572 | 25.4 Usage Notes | Description (7) to (9) added |
| 604 | 27.3.7 RTS Instruction | Figure 27.15 RTS Instruction Description amended Stack storing |
| 613 | 28.1.2 Block Diagram | Figure 28.1 Block Diagram of Servo Circuits Amended |
| 624 | 28.2.5 Register Descriptions | (5) CTL Gain Control Register (CTLGR) Bits 3 to 0: CTL Amplifier Gain Setting Bits (CTLGR3 to 0) values of CTL output gain amended |
| 627 | 28.3.2 Block Diagram | Figure 28.6 REF30 Signal Generator amended |
| 634 | 28.3.4 Register Descriptions | (5) Reference Period Mode Register 2 (RFM2) Bit 7: TBC Selection Bit Description amended |
| 663, 664 | 28.4.5 Register Descriptions | (4) FIFO Output Pattern Register 1 (FPDRA) (5) FIFO Output Pattern Register 2 (FPDRB) Descriptions of bits in these registers added |
| 667, 668 | 28.4.5 Register Descriptions | (9) DFG Reference Register 2 (DFCRB) Descriptions of bits 4 to 0 added (11) DFG Reference Count Register (DFCTR) Initial value of bit 4 to 0 amended and descriptions added |
| 669, 670 | 28.4.6 Description of Operation | Completely Amended |
| 688 | 28.6.4 Register Descriptions | (5) Drum Speed Error Detection Control Register (DFVCR) Descriptions of bits 1 and 0 amended |

| Page | Item | Revisions (See Manual for Details) |
|--------------|---|--|
| 708 | 28.8.4 Register Descriptions | (5) Capstan Speed Error Detection Control Register (CFVCR) Descriptions of bits 1 and 0 amended |
| 745 | 28.12.5 Additional V Pulse Signal | Figure 28.46 Additional V Pulse Negative Polarity is Specified Value of POL amended |
| 761 | 28.13.5 Register Descriptions | (8) Duty I/O Register (DI/O) Descriptions of ASM Mark Detect Mode amended |
| 769 | 28.13.8 Duty Discriminator | Values of duty amended |
| 780 | 28.14.2 CTL Frequency Divider | Figure 28.63 CTL Frequency Divider amended |
| 817 | 28.17 Module Stop Control Register (MSTPCR) | Added |
| 819 to 860 | 29 Electrical Characteristics | Description amended due to introduction of the H8S/2194C series |
| 861 to 909 | Appendix A Instruction Set | Notes on TAS instruction added |
| 917 to 1017 | B.2 Function List | Following list of registers amended H'D097: RFM2 H'D0A4: CTLGR H'D13A: TMJ H'D13B: TMJC H'D148: SMR1 H'D14C: SR1 H'D158: ICCR H'D159: ICSR H'FFF8: FLMCR1 H'FFF9: FLMCR2 H'FFFA: EBR1 H'FFFB: EBR2 |
| 1019 to 1031 | C.1 Pin Circuit Diagrams | Table C.1 Pin Circuit Diagrams Completely amended |
| 1035 | E.3 Sample External Circuits | Figure E.3 Sample External Circuit for Servo Section Amended Figure E.4 Example of External Circuit for Sync Signal Detection Circuit Section Amended |
| 1036 | Appendix F List of Product Codes | Figure F.1 Product Codes List of H8S/2194 series and H8S/2194C series |

Contents

| | | |
|-----------|---|----|
| Section 1 | Overview..... | 1 |
| 1.1 | Overview..... | 1 |
| 1.2 | Internal Block Diagram..... | 6 |
| 1.3 | Pin Arrangement and Functions..... | 7 |
| 1.3.1 | Pin Arrangement | 7 |
| 1.3.2 | Pin Functions..... | 8 |
| 1.4 | Differences between H8S/2194C Series and H8S/2194 Series..... | 14 |
| Section 2 | CPU | 15 |
| 2.1 | Overview..... | 15 |
| 2.1.1 | Features..... | 15 |
| 2.1.2 | Differences between H8S/2600 CPU and H8S/2000 CPU..... | 16 |
| 2.1.3 | Differences from H8/300 CPU | 16 |
| 2.1.4 | Differences from H8/300H CPU | 17 |
| 2.2 | CPU Operating Modes | 18 |
| 2.3 | Address Space..... | 23 |
| 2.4 | Register Configuration | 24 |
| 2.4.1 | Overview..... | 24 |
| 2.4.2 | General Registers | 25 |
| 2.4.3 | Control Registers..... | 26 |
| 2.4.4 | Initial Register Values | 27 |
| 2.5 | Data Formats | 28 |
| 2.5.1 | General Register Data Formats | 28 |
| 2.5.2 | Memory Data Formats..... | 30 |
| 2.6 | Instruction Set..... | 31 |
| 2.6.1 | Overview..... | 31 |
| 2.6.2 | Instructions and Addressing Modes..... | 32 |
| 2.6.3 | Table of Instructions Classified by Function..... | 33 |
| 2.6.4 | Basic Instruction Formats | 43 |
| 2.6.5 | Notes on Use of Bit-Manipulation Instructions..... | 44 |
| 2.7 | Addressing Modes and Effective Address Calculation | 45 |
| 2.7.1 | Addressing Mode | 45 |
| 2.7.2 | Effective Address Calculation..... | 48 |
| 2.8 | Processing States..... | 52 |
| 2.8.1 | Overview..... | 52 |
| 2.8.2 | Reset State..... | 53 |
| 2.8.3 | Exception-Handling State | 54 |
| 2.8.4 | Program Execution State..... | 55 |
| 2.8.5 | Power-Down State..... | 56 |

| | | |
|-------------------------------------|---|----|
| 2.9 | Basic Timing | 57 |
| 2.9.1 | Overview | 57 |
| 2.9.2 | On-Chip Memory (ROM, RAM)..... | 57 |
| 2.9.3 | On-Chip Supporting Module Access Timing..... | 58 |
| 2.10 | Usage Note | 58 |
| Section 3 MCU Operating Modes | | 59 |
| 3.1 | Overview | 59 |
| 3.1.1 | Operating Mode Selection | 59 |
| 3.1.2 | Register Configuration..... | 59 |
| 3.2 | Register Descriptions | 60 |
| 3.2.1 | Mode Control Register (MDCR)..... | 60 |
| 3.2.2 | System Control Register (SYSCR)..... | 60 |
| 3.3 | Operating Mode Descriptions..... | 62 |
| 3.3.1 | Mode 1..... | 62 |
| 3.4 | Address Map..... | 63 |
| Section 4 Power-Down State | | 67 |
| 4.1 | Overview | 67 |
| 4.1.1 | Register Configuration..... | 71 |
| 4.2 | Register Descriptions | 72 |
| 4.2.1 | Standby Control Register (SBYCR)..... | 72 |
| 4.2.2 | Low-Power Control Register (LPWRCR) | 74 |
| 4.2.3 | Timer Register A (TMA)..... | 76 |
| 4.2.4 | Module Stop Control Register (MSTPCR) | 77 |
| 4.3 | Medium-Speed Mode..... | 78 |
| 4.4 | Sleep Mode..... | 79 |
| 4.4.1 | Sleep Mode | 79 |
| 4.4.2 | Clearing Sleep Mode | 79 |
| 4.5 | Module Stop Mode | 80 |
| 4.5.1 | Module Stop Mode..... | 80 |
| 4.6 | Standby Mode..... | 81 |
| 4.6.1 | Standby Mode | 81 |
| 4.6.2 | Clearing Standby Mode | 81 |
| 4.6.3 | Setting Oscillation Settling Time after Clearing Standby Mode | 81 |
| 4.7 | Watch Mode | 83 |
| 4.7.1 | Watch Mode..... | 83 |
| 4.7.2 | Clearing Watch Mode..... | 83 |
| 4.8 | Subsleep Mode | 84 |
| 4.8.1 | Subsleep Mode..... | 84 |
| 4.8.2 | Clearing Subsleep Mode | 84 |
| 4.9 | Subactive Mode | 85 |
| 4.9.1 | Subactive Mode..... | 85 |

| | | |
|-------------------------------------|---|-----|
| 4.9.2 | Clearing Subactive Mode..... | 85 |
| 4.10 | Direct Transition..... | 86 |
| 4.10.1 | Overview of Direct Transition | 86 |
| Section 5 Exception Handling | | 87 |
| 5.1 | Overview..... | 87 |
| 5.1.1 | Exception Handling Types and Priority | 87 |
| 5.1.2 | Exception Handling Operation | 88 |
| 5.1.3 | Exception Sources and Vector Table | 88 |
| 5.2 | Reset | 90 |
| 5.2.1 | Overview..... | 90 |
| 5.2.2 | Reset Sequence..... | 90 |
| 5.2.3 | Interrupts after Reset | 91 |
| 5.3 | Interrupts | 92 |
| 5.4 | Trap Instruction | 93 |
| 5.5 | Stack Status after Exception Handling | 94 |
| 5.6 | Notes on Use of the Stack | 95 |
| Section 6 Interrupt Controller..... | | 97 |
| 6.1 | Overview..... | 97 |
| 6.1.1 | Features..... | 97 |
| 6.1.2 | Block Diagram | 98 |
| 6.1.3 | Pin Configuration | 99 |
| 6.1.4 | Register Configuration..... | 99 |
| 6.2 | Register Descriptions | 100 |
| 6.2.1 | System Control Register (SYSCR)..... | 100 |
| 6.2.2 | Interrupt Control Registers A to D (ICRA to ICRD)..... | 101 |
| 6.2.3 | IRQ Enable Register (IENR)..... | 102 |
| 6.2.4 | IRQ Edge Select Registers (IEGR)..... | 103 |
| 6.2.5 | IRQ Status Register (IRQR)..... | 104 |
| 6.2.6 | Port Mode Register (PMR1) | 105 |
| 6.3 | Interrupt Sources..... | 106 |
| 6.3.1 | External Interrupts | 106 |
| 6.3.2 | Internal Interrupts | 108 |
| 6.3.3 | Interrupt Exception Vector Table | 108 |
| 6.4 | Interrupt Operation..... | 111 |
| 6.4.1 | Interrupt Control Modes and Interrupt Operation..... | 111 |
| 6.4.2 | Interrupt Control Mode 0 | 113 |
| 6.4.3 | Interrupt Control Mode 1 | 115 |
| 6.4.4 | Interrupt Exception Handling Sequence | 118 |
| 6.4.5 | Interrupt Response Times | 119 |
| 6.5 | Usage Notes..... | 120 |
| 6.5.1 | Contention between Interrupt Generation and Disabling | 120 |

| | | |
|------------------|--|------------|
| 6.5.2 | Instructions that Disable Interrupts..... | 121 |
| 6.5.3 | Interrupts during Execution of EEPMOV Instruction..... | 121 |
| 6.5.4 | When NMI is Disabled | 121 |
| Section 7 | ROM (H8S/2194 Series) | 123 |
| 7.1 | Overview | 123 |
| 7.1.1 | Block Diagram | 123 |
| 7.2 | Overview of Flash Memory..... | 124 |
| 7.2.1 | Features..... | 124 |
| 7.2.2 | Block Diagram | 125 |
| 7.2.3 | Flash Memory Operating Modes..... | 126 |
| 7.2.4 | Pin Configuration | 130 |
| 7.2.5 | Register Configuration..... | 130 |
| 7.3 | Flash Memory Register Descriptions | 131 |
| 7.3.1 | Flash Memory Control Register 1 (FLMCR1) | 131 |
| 7.3.2 | Flash Memory Control Register 2 (FLMCR2) | 134 |
| 7.3.3 | Erase Block Registers 1 and 2 (EBR1, EBR2)..... | 136 |
| 7.3.4 | Serial/Timer Control Register (STCR) | 137 |
| 7.4 | On-Board Programming Modes..... | 138 |
| 7.4.1 | Boot Mode | 139 |
| 7.4.2 | User Program Mode | 144 |
| 7.5 | Programming/Erasing Flash Memory | 145 |
| 7.5.1 | Program Mode..... | 145 |
| 7.5.2 | Program-Verify Mode | 146 |
| 7.5.3 | Erase Mode | 148 |
| 7.5.4 | Erase-Verify Mode | 148 |
| 7.6 | Flash Memory Protection | 150 |
| 7.6.1 | Hardware Protection..... | 150 |
| 7.6.2 | Software Protection | 151 |
| 7.6.3 | Error Protection..... | 152 |
| 7.7 | Interrupt Handling when Programming/Erasing Flash Memory | 153 |
| 7.8 | Flash Memory Programmer Mode | 154 |
| 7.8.1 | Programmer Mode Setting..... | 154 |
| 7.8.2 | Socket Adapters and Memory Map | 154 |
| 7.8.3 | Programmer Mode Operation..... | 155 |
| 7.8.4 | Memory Read Mode..... | 157 |
| 7.8.5 | Auto-Program Mode..... | 160 |
| 7.8.6 | Auto-Erase Mode | 162 |
| 7.8.7 | Status Read Mode..... | 163 |
| 7.8.8 | Status Polling | 165 |
| 7.8.9 | Programmer Mode Transition Time | 166 |
| 7.8.10 | Notes On Memory Programming | 166 |
| 7.9 | Flash Memory Programming and Erasing Precautions | 167 |

| | | |
|-----------|---|-----|
| Section 8 | ROM (H8S/2194C Series) | 171 |
| 8.1 | Overview | 171 |
| 8.1.1 | Block Diagram | 171 |
| 8.2 | Overview of Flash Memory..... | 172 |
| 8.2.1 | Features..... | 172 |
| 8.2.2 | Block Diagram | 173 |
| 8.2.3 | Flash Memory Operating Modes | 174 |
| 8.2.4 | Pin Configuration | 178 |
| 8.2.5 | Register Configuration..... | 178 |
| 8.3 | Flash Memory Register Descriptions | 179 |
| 8.3.1 | Flash Memory Control Register 1 (FLMCR1) | 179 |
| 8.3.2 | Flash Memory Control Register 2 (FLMCR2) | 182 |
| 8.3.3 | Erase Block Registers 1 (EBR1) | 185 |
| 8.3.4 | Erase Block Registers 2 (EBR2) | 185 |
| 8.3.5 | Serial/Timer Control Register (STCR) | 187 |
| 8.4 | On-Board Programming Modes..... | 188 |
| 8.4.1 | Boot Mode | 189 |
| 8.4.2 | User Program Mode..... | 194 |
| 8.5 | Programming/Erasing Flash Memory | 195 |
| 8.5.1 | Program Mode (n = 1 for addresses H'0000 to H'1FFFF and n= 2 for addresses H'20000 to H'3FFFF) | 195 |
| 8.5.2 | Program-Verify Mode (n =1 for addresses H'00000 to H'1FFFF and n = 2 for addresses H'20000 to H'3FFFF) | 196 |
| 8.5.3 | Erase Mode (n = 1 for addresses H'00000 to H'1FFFF and n = 2 for address H'20000 to H'3FFFF) | 198 |
| 8.5.4 | Erase-Verify Mode (n = 1 for addresses H'00000 to H'1FFFF and n = 2 for address H'20000 to H'3FFFF) | 198 |
| 8.6 | Flash Memory Protection | 200 |
| 8.6.1 | Hardware Protection | 200 |
| 8.6.2 | Software Protection | 201 |
| 8.6.3 | Error Protection | 202 |
| 8.7 | Interrupt Handling when Programming/Erasing Flash Memory | 203 |
| 8.8 | Flash Memory Programmer Mode | 204 |
| 8.8.1 | Programmer Mode Setting | 204 |
| 8.8.2 | Socket Adapters and Memory Map | 204 |
| 8.8.3 | Programmer Mode Operation..... | 205 |
| 8.8.4 | Memory Read Mode | 207 |
| 8.8.5 | Auto-Program Mode..... | 210 |
| 8.8.6 | Auto-Erase Mode | 212 |
| 8.8.7 | Status Read Mode..... | 213 |
| 8.8.8 | Status Polling | 215 |

| | | |
|---------------------------------------|---|-----|
| 8.8.9 | Programmer Mode Transition Time | 216 |
| 8.8.10 | Notes On Memory Programming | 216 |
| 8.9 | Flash Memory Programming and Erasing Precautions | 217 |
| 8.10 | Note on Switching from F-ZTAT Version to Mask ROM Version | 219 |
| Section 9 RAM..... | | 221 |
| 9.1 | Overview..... | 221 |
| 9.1.1 | Block Diagram | 221 |
| Section 10 Clock Pulse Generator..... | | 223 |
| 10.1 | Overview..... | 223 |
| 10.1.1 | Block Diagram | 223 |
| 10.1.2 | Register Configuration..... | 223 |
| 10.2 | Register Descriptions | 224 |
| 10.2.1 | Standby Control Register (SBYCR)..... | 224 |
| 10.2.2 | Low-Power Control Register (LPWRCR) | 225 |
| 10.3 | Oscillator..... | 226 |
| 10.3.1 | Connecting a Crystal Resonator | 226 |
| 10.3.2 | External Clock Input..... | 228 |
| 10.4 | Duty Adjustment Circuit | 231 |
| 10.5 | Medium-Speed Clock Divider | 231 |
| 10.6 | Bus Master Clock Selection Circuit | 231 |
| 10.7 | Subclock Oscillator Circuit | 232 |
| 10.7.1 | Connecting 32.768 kHz Crystal Resonator | 232 |
| 10.7.2 | External Clock Input..... | 233 |
| 10.7.3 | When Subclock is not Needed..... | 233 |
| 10.8 | Subclock Waveform Shaping Circuit..... | 234 |
| 10.9 | Notes on the Resonator | 234 |
| Section 11 I/O Port | | 235 |
| 11.1 | Overview..... | 235 |
| 11.1.1 | Port Functions | 235 |
| 11.1.2 | Port Input | 235 |
| 11.1.3 | MOS Pull-Up Transistors | 237 |
| 11.2 | Port 0..... | 238 |
| 11.2.1 | Overview | 238 |
| 11.2.2 | Register Configuration..... | 239 |
| 11.2.3 | Pin Functions | 240 |
| 11.2.4 | Pin States | 240 |
| 11.3 | Port 1..... | 241 |
| 11.3.1 | Overview | 241 |
| 11.3.2 | Register Configuration..... | 241 |
| 11.3.3 | Pin Functions | 245 |

| | | |
|------------|-----------------------------|-----|
| 11.3.4 | Pin States | 246 |
| 11.4 | Port 2..... | 247 |
| 11.4.1 | Overview..... | 247 |
| 11.4.2 | Register Configuration..... | 247 |
| 11.4.3 | Pin Functions..... | 251 |
| 11.4.4 | Pin States | 253 |
| 11.5 | Port 3..... | 254 |
| 11.5.1 | Overview..... | 254 |
| 11.5.2 | Register Configuration..... | 254 |
| 11.5.3 | Pin Functions..... | 258 |
| 11.5.4 | Pin States | 260 |
| 11.6 | Port 4..... | 261 |
| 11.6.1 | Overview..... | 261 |
| 11.6.2 | Register Configuration..... | 261 |
| 11.6.3 | Pin Functions..... | 264 |
| 11.6.4 | Pin States | 266 |
| 11.7 | Port 5..... | 267 |
| 11.7.1 | Overview..... | 267 |
| 11.7.2 | Register Configuration..... | 267 |
| 11.7.3 | Pin Functions..... | 271 |
| 11.7.4 | Pin States | 272 |
| 11.8 | Port 6..... | 273 |
| 11.8.1 | Overview..... | 273 |
| 11.8.2 | Register Configuration..... | 274 |
| 11.8.3 | Pin Functions..... | 277 |
| 11.8.4 | Operation | 278 |
| 11.8.5 | Pin States | 279 |
| 11.9 | Port 7..... | 280 |
| 11.9.1 | Overview..... | 280 |
| 11.9.2 | Register Configuration..... | 280 |
| 11.9.3 | Pin Functions..... | 282 |
| 11.9.4 | Pin States | 282 |
| 11.10 | Port 8..... | 283 |
| 11.10.1 | Overview..... | 283 |
| 11.10.2 | Register Configuration..... | 283 |
| 11.10.3 | Pin Functions..... | 286 |
| 11.10.4 | Pin States | 288 |
| Section 12 | Timer A..... | 289 |
| 12.1 | Overview | 289 |
| 12.1.1 | Features..... | 289 |
| 12.1.2 | Block Diagram | 290 |
| 12.1.3 | Register Configuration..... | 290 |

| | | |
|-------------------|---|------------|
| 12.2 | Descriptions of Respective Registers | 291 |
| 12.2.1 | Timer Mode Register A (TMA) | 291 |
| 12.2.2 | Timer Counter A (TCA) | 293 |
| 12.2.3 | Module Stop Control Register (MSTPCR) | 293 |
| 12.3 | Operation..... | 294 |
| 12.3.1 | Operation as the Interval Timer..... | 294 |
| 12.3.2 | Operation of the Timer for Clocks..... | 294 |
| 12.3.3 | Initializing the Counts..... | 294 |
| Section 13 | Timer B..... | 295 |
| 13.1 | Overview | 295 |
| 13.1.1 | Features..... | 295 |
| 13.1.2 | Block Diagram | 295 |
| 13.1.3 | Pin Configuration | 296 |
| 13.1.4 | Register Configuration..... | 296 |
| 13.2 | Descriptions of Respective Registers | 297 |
| 13.2.1 | Timer Mode Register B (TMB)..... | 297 |
| 13.2.2 | Timer Counter B (TCB)..... | 299 |
| 13.2.3 | Timer Load Register B (TLB)..... | 299 |
| 13.2.4 | Port Mode Register 5 (PMR5) | 300 |
| 13.2.5 | Module Stop Control Register (MSTPCR) | 300 |
| 13.3 | Operation..... | 302 |
| 13.3.1 | Operation as the Interval Timer..... | 302 |
| 13.3.2 | Operation as the Auto Reload Timer | 302 |
| 13.3.3 | Event Counter | 302 |
| Section 14 | Timer J..... | 303 |
| 14.1 | Overview | 303 |
| 14.1.1 | Features..... | 303 |
| 14.1.2 | Block Diagram | 303 |
| 14.1.3 | Pin Configuration | 305 |
| 14.1.4 | Register Configuration..... | 305 |
| 14.2 | Descriptions of Respective Registers | 306 |
| 14.2.1 | Timer Mode Register J (TMJ)..... | 306 |
| 14.2.2 | Timer J Control Register (TMJC) | 310 |
| 14.2.3 | Timer J Status Register (TMJS) | 312 |
| 14.2.4 | Timer Counter J (TCJ)..... | 313 |
| 14.2.5 | Timer Counter K (TCK) | 313 |
| 14.2.6 | Timer Load Register J (TLJ)..... | 314 |
| 14.2.7 | Timer Load Register K (TLK) | 314 |
| 14.2.8 | Module Stop Control Register (MSTPCR) | 315 |
| 14.3 | Operation..... | 316 |
| 14.3.1 | 8-bit Reload Timer (TMJ-1)..... | 316 |

| | | |
|--------------------------|---|-----|
| 14.3.2 | 8-bit Reload Timer (TMJ-2)..... | 316 |
| 14.3.3 | Remote Controlled Data Transmission | 317 |
| Section 15 Timer L | | 321 |
| 15.1 | Overview | 321 |
| 15.1.1 | Features..... | 321 |
| 15.1.2 | Block Diagram | 322 |
| 15.1.3 | Register Configuration..... | 323 |
| 15.2 | Descriptions of Respective Registers | 324 |
| 15.2.1 | Timer L Mode Register (LMR)..... | 324 |
| 15.2.2 | Linear Time Counter (LTC)..... | 326 |
| 15.2.3 | Reload/Compare Match Register (RCR)..... | 326 |
| 15.2.4 | Module Stop Control Register (MSTPCR) | 327 |
| 15.3 | Operation..... | 328 |
| 15.3.1 | Compare Match Clear Operation..... | 328 |
| Section 16 Timer R | | 331 |
| 16.1 | Overview | 331 |
| 16.1.1 | Features..... | 331 |
| 16.1.2 | Block Diagram | 331 |
| 16.1.3 | Pin Configuration | 333 |
| 16.1.4 | Register Configuration..... | 333 |
| 16.2 | Descriptions of Respective Registers | 334 |
| 16.2.1 | Timer R Mode Register 1 (TMRM1)..... | 334 |
| 16.2.2 | Timer R Mode Register 2 (TMRM2)..... | 336 |
| 16.2.3 | Timer R Control/Status Register (TMRCS) | 339 |
| 16.2.4 | Timer R Capture Register 1 (TMRCPI) | 341 |
| 16.2.5 | Timer R Capture Register 2 (TMRCPI2) | 342 |
| 16.2.6 | Timer R Load Register 1 (TMRL1)..... | 342 |
| 16.2.7 | Timer R Load Register 2 (TMRL2)..... | 343 |
| 16.2.8 | Timer R Load Register 3 (TMRL3)..... | 343 |
| 16.2.9 | Module Stop Control Register (MSTPCR) | 344 |
| 16.3 | Operation..... | 345 |
| 16.3.1 | Reload Timer Counter Equipped with Capturing Function TMRU-1..... | 345 |
| 16.3.2 | Reload Timer Counter Equipped with Capturing Function TMRU-2..... | 346 |
| 16.3.3 | Reload Counter Timer TMRU-3 | 346 |
| 16.3.4 | Mode Identification | 347 |
| 16.3.5 | Reeling Controls..... | 347 |
| 16.3.6 | Acceleration and Braking Processes of the Capstan Motor..... | 347 |
| 16.3.7 | Slow Tracking Mono-multi Function | 348 |
| 16.4 | Interrupt Cause | 350 |
| 16.5 | Exemplary Settings for Respective Functions | 351 |
| 16.5.1 | Mode Identification | 351 |

| | | |
|-------------------|--|------------|
| 16.5.2 | Reeling Controls..... | 352 |
| 16.5.3 | Slow Tracking Mono-multi Function | 352 |
| 16.5.4 | Acceleration and Braking Processes of the Capstan Motor..... | 353 |
| Section 17 | Timer X1..... | 355 |
| 17.1 | Overview..... | 355 |
| 17.1.1 | Features..... | 355 |
| 17.1.2 | Block Diagram | 356 |
| 17.1.3 | Pin Configuration | 357 |
| 17.1.4 | Register Configuration..... | 358 |
| 17.2 | Descriptions of Respective Registers | 359 |
| 17.2.1 | Free Running Counter (FRC)..... | 359 |
| 17.2.2 | Output Comparing Register A and B (OCRA and OCRB) | 360 |
| 17.2.3 | Input Capture Register A Through D (ICRA Through ICRD) | 361 |
| 17.2.4 | Timer Interrupt Enabling Register (TIER)..... | 363 |
| 17.2.5 | Timer Control/Status Register X (TCSRX) | 366 |
| 17.2.6 | Timer Control Register X (TCRX)..... | 370 |
| 17.2.7 | Timer Output Comparing Control Register (TOCR)..... | 372 |
| 17.2.8 | Module Stop Control Register (MSTPCR) | 375 |
| 17.3 | Operation..... | 376 |
| 17.3.1 | Operation of the Timer X1..... | 376 |
| 17.3.2 | Counting Timing of the FRC | 377 |
| 17.3.3 | Output Comparing Signal Outputting Timing..... | 378 |
| 17.3.4 | FRC Clearing Timing | 378 |
| 17.3.5 | Input Capture Signal Inputting Timing..... | 379 |
| 17.3.6 | Input Capture Flag (ICFA through ICFD) Setting Up Timing..... | 380 |
| 17.3.7 | Output Comparing Flag (OCFA and OCFB) Setting Up Timing | 381 |
| 17.3.8 | Overflow Flag (CVF) Setting Up Timing | 381 |
| 17.4 | Operation Mode of the Timer X1 | 382 |
| 17.5 | Interrupt Causes..... | 383 |
| 17.6 | Exemplary Uses of the Timer X1 | 384 |
| 17.7 | Precautions when Using the Timer X1..... | 385 |
| 17.7.1 | Competition between Writing and Clearing with the FRC | 385 |
| 17.7.2 | Competition between Writing and Counting Up with the FRC..... | 386 |
| 17.7.3 | Competition between Writing and Comparing Match with the OCR | 387 |
| 17.7.4 | Changing Over the Internal Clocks and Counter Operations | 388 |
| Section 18 | Watchdog Timer (WDT) | 391 |
| 18.1 | Overview..... | 391 |
| 18.1.1 | Features..... | 391 |
| 18.1.2 | Block Diagram | 392 |
| 18.1.3 | Register Configuration..... | 393 |
| 18.2 | Register Descriptions | 394 |

| | | |
|----------------------------|---|-----|
| 18.2.1 | Watchdog Timer Counter (WTCNT)..... | 394 |
| 18.2.2 | Watchdog Timer Control/Status Register (WTCSR)..... | 394 |
| 18.2.3 | System Control Register (SYSCR)..... | 397 |
| 18.2.4 | Notes on Register Access..... | 397 |
| 18.3 | Operation..... | 399 |
| 18.3.1 | Watchdog Timer Operation..... | 399 |
| 18.3.2 | Interval Timer Operation | 400 |
| 18.3.3 | Timing of Setting of Overflow Flag (OVF) | 401 |
| 18.4 | Interrupts | 402 |
| 18.5 | Usage Notes..... | 402 |
| 18.5.1 | Contention between Watchdog Timer Counter (WTCNT) Write and Increment..... | 402 |
| 18.5.2 | Changing Value of CKS2 to CKS0 | 403 |
| 18.5.3 | Switching between Watchdog Timer Mode and Interval Timer Mode..... | 403 |
| Section 19 8-Bit PWM..... | | 405 |
| 19.1 | Overview | 405 |
| 19.1.1 | Features..... | 405 |
| 19.1.2 | Block Diagram | 405 |
| 19.1.3 | Pin Configuration | 406 |
| 19.1.4 | Register Configuration..... | 406 |
| 19.2 | Register Descriptions | 407 |
| 19.2.1 | Bit PWM Data Registers 0, 1, 2 and 3 (PWR0, PWR1, PWR2, PWR3) | 407 |
| 19.2.2 | 8-bit PWM Control Register (PW8CR) | 408 |
| 19.2.3 | Port Mode Register 3 (PMR3)..... | 409 |
| 19.2.4 | Module Stop Control Register (MSTPCR) | 410 |
| 19.3 | 8-Bit PWM Operation | 411 |
| Section 20 12-Bit PWM..... | | 413 |
| 20.1 | Overview | 413 |
| 20.1.1 | Features..... | 413 |
| 20.1.2 | Block Diagram | 414 |
| 20.1.3 | Pin Configuration | 415 |
| 20.1.4 | Register Configuration..... | 415 |
| 20.2 | Register Descriptions | 416 |
| 20.2.1 | 12-Bit PWM Control Registers (CPWCR, DPWCR) | 416 |
| 20.2.2 | 12-Bit PWM Data Registers (CPWDR, DPWDR) | 419 |
| 20.2.3 | Module Stop Control Register (MSTPCR) | 420 |
| 20.3 | Operation..... | 421 |
| 20.3.1 | Output Waveform..... | 421 |
| Section 21 14-Bit PWM..... | | 423 |
| 21.1 | Overview | 423 |

| | | |
|---|--|-----|
| 21.1.1 | Features..... | 423 |
| 21.1.2 | Block Diagram | 424 |
| 21.1.3 | Pin Configuration | 424 |
| 21.1.4 | Register Configuration..... | 425 |
| 21.2 | Register Descriptions | 426 |
| 21.2.1 | PWM Control Register (PWCR) | 426 |
| 21.2.2 | PWM Data Registers U and L (PWDRU, PWDRL)..... | 427 |
| 21.2.3 | Module Stop Control Register (MSTPCR) | 428 |
| 21.3 | 14-Bit PWM Operation | 429 |
| Section 22 Prescaler Unit..... | | 431 |
| 22.1 | Overview | 431 |
| 22.1.1 | Features..... | 431 |
| 22.1.2 | Block Diagram | 432 |
| 22.1.3 | Pin Configuration | 433 |
| 22.1.4 | Register Configuration..... | 433 |
| 22.2 | Registers..... | 434 |
| 22.2.1 | Input Capture Register 1 (ICR1) | 434 |
| 22.2.2 | Prescaler Unit Control/Status Register (PCSR)..... | 434 |
| 22.2.3 | Port Mode Register 1 (PMR1) | 437 |
| 22.3 | Noise Cancel Circuit | 437 |
| 22.4 | Operation..... | 438 |
| 22.4.1 | Prescaler S (PSS)..... | 438 |
| 22.4.2 | Prescaler W (PSW)..... | 439 |
| 22.4.3 | Stable Oscillation Wait Time Count..... | 439 |
| 22.4.4 | 8-Bit PWM..... | 440 |
| 22.4.5 | 8-Bit Input Capture Using IC Pin | 440 |
| 22.4.6 | Frequency Division Clock Output | 440 |
| Section 23 Serial Communication Interface 1 (SCI1)..... | | 441 |
| 23.1 | Overview | 441 |
| 23.1.1 | Features..... | 441 |
| 23.1.2 | Block Diagram | 443 |
| 23.1.3 | Pin Configuration | 444 |
| 23.1.4 | Register Configuration..... | 444 |
| 23.2 | Register Descriptions | 445 |
| 23.2.1 | Receive Shift Register (RSR)..... | 445 |
| 23.2.2 | Receive Data Register (RDR1) | 445 |
| 23.2.3 | Transmit Shift Register (TSR) | 446 |
| 23.2.4 | Transmit Data Register (TDR1) | 446 |
| 23.2.5 | Serial Mode Register (SMR1)..... | 447 |
| 23.2.6 | Serial Control Register (SCR1)..... | 450 |
| 23.2.7 | Serial Status Register (SSR1)..... | 453 |

| | | |
|--|--|-----|
| 23.2.8 | Bit Rate Register (BRR1) | 457 |
| 23.2.9 | Serial Interface Mode Register (SCMR1) | 464 |
| 23.2.10 | Module Stop Control Register (MSTPCR) | 465 |
| 23.3 | Operation..... | 466 |
| 23.3.1 | Overview..... | 466 |
| 23.3.2 | Operation in Asynchronous Mode..... | 468 |
| 23.3.3 | Multiprocessor Communication Function..... | 478 |
| 23.3.4 | Operation in Clock Synchronous Mode..... | 486 |
| 23.4 | SCI1 Interrupts | 494 |
| 23.5 | Usage Notes..... | 495 |
| Section 24 Serial Communication Interface 2 (SCI2) | | 499 |
| 24.1 | Overview..... | 499 |
| 24.1.1 | Features..... | 499 |
| 24.1.2 | Block Diagram | 500 |
| 24.1.3 | Pin Configuration | 501 |
| 24.1.4 | Register Configuration..... | 501 |
| 24.2 | Register Descriptions | 502 |
| 24.2.1 | Starting Address Register (STAR) | 502 |
| 24.2.2 | Ending Address Register (EDAR)..... | 502 |
| 24.2.3 | Serial Control Register 2 (SCR2) | 503 |
| 24.2.4 | Serial Control Status Register 2 (SCSR2)..... | 504 |
| 24.2.5 | Module Stop Control Register (MSTPCR) | 507 |
| 24.3 | Operation..... | 508 |
| 24.3.1 | Clock | 508 |
| 24.3.2 | Data Transfer Format..... | 508 |
| 24.3.3 | Data Transfer Operations..... | 511 |
| 24.4 | Interrupt Sources..... | 515 |
| Section 25 I ² C Bus Interface (IIC)..... | | 517 |
| 25.1 | Overview..... | 517 |
| 25.1.1 | Features..... | 517 |
| 25.1.2 | Block Diagram | 518 |
| 25.1.3 | Pin Configuration | 519 |
| 25.1.4 | Register Configuration..... | 520 |
| 25.2 | Register Descriptions | 521 |
| 25.2.1 | I ² C Bus Data Register (ICDR)..... | 521 |
| 25.2.2 | Slave Address Register (SAR) | 524 |
| 25.2.3 | Second Slave Address Register (SARX) | 526 |
| 25.2.4 | I ² C Bus Mode Register (ICMR) | 527 |
| 25.2.5 | I ² C Bus Control Register (ICCR) | 531 |
| 25.2.6 | I ² C Bus Status Register (ICSR) | 538 |
| 25.2.7 | Serial/Timer Control Register (STCR) | 543 |

| | | |
|---|--|-----|
| 25.2.8 | Module Stop Control Register (MSTPCR) | 545 |
| 25.3 | Operation..... | 546 |
| 25.3.1 | I ² C Bus Data Format..... | 546 |
| 25.3.2 | Master Transmit Operation | 547 |
| 25.3.3 | Master Receive Operation..... | 550 |
| 25.3.4 | Slave Receive Operation..... | 553 |
| 25.3.5 | Slave Transmit Operation | 556 |
| 25.3.6 | IRIC Setting Timing and SCL Control | 558 |
| 25.3.7 | Noise Canceler | 560 |
| 25.3.8 | Sample Flowcharts | 560 |
| 25.3.9 | Initialization of Internal State..... | 565 |
| 25.4 | Usage Notes..... | 567 |
| Section 26 A/D Converter..... | | 573 |
| 26.1 | Overview | 573 |
| 26.1.1 | Features..... | 573 |
| 26.1.2 | Block Diagram | 574 |
| 26.1.3 | Pin Configuration | 575 |
| 26.1.4 | Register Configuration..... | 576 |
| 26.2 | Register Descriptions | 577 |
| 26.2.1 | Software-Triggered A/D Result Register (ADR) | 577 |
| 26.2.2 | Hardware-Triggered A/D Result Register (AHR)..... | 577 |
| 26.2.3 | A/D Control Register (ADCR)..... | 579 |
| 26.2.4 | A/D Control/Status Register (ADCSR) | 582 |
| 26.2.5 | Trigger Select Register (ADTSR) | 585 |
| 26.2.6 | Port Mode Register 0 (PMR0) | 585 |
| 26.2.7 | Module Stop Control Register (MSTPCR) | 586 |
| 26.3 | Interface to Bus Master | 587 |
| 26.4 | Operation..... | 588 |
| 26.4.1 | Software-Triggered A/D Conversion..... | 588 |
| 26.4.2 | Hardware- or External-Triggered A/D Conversion | 589 |
| 26.5 | Interrupt Sources..... | 590 |
| Section 27 Address Trap Controller (ATC)..... | | 591 |
| 27.1 | Overview | 591 |
| 27.1.1 | Features..... | 591 |
| 27.1.2 | Block Diagram | 591 |
| 27.1.3 | Register Configuration..... | 592 |
| 27.2 | Register Descriptions | 592 |
| 27.2.1 | Address Trap Control Register (ATCR) | 592 |
| 27.2.2 | Trap Address Register 2 to 0 (TAR2 to TAR0) | 593 |
| 27.3 | Precautions in Usage..... | 595 |
| 27.3.1 | Basic Operations | 595 |

| | | |
|---------------------------------|--|-----|
| 27.3.2 | Enable | 597 |
| 27.3.3 | Bcc Instruction | 597 |
| 27.3.4 | BSR Instruction | 601 |
| 27.3.5 | JSR Instruction | 602 |
| 27.3.6 | JMP Instruction | 603 |
| 27.3.7 | RTS Instruction | 604 |
| 27.3.8 | SLEEP Instruction | 604 |
| 27.3.9 | Competing Interrupt | 607 |
| Section 28 Servo Circuits | | 611 |
| 28.1 | Overview | 611 |
| 28.1.1 | Functions..... | 611 |
| 28.1.2 | Block Diagram | 612 |
| 28.2 | Servo Port..... | 614 |
| 28.2.1 | Overview..... | 614 |
| 28.2.2 | Block Diagram | 614 |
| 28.2.3 | Pin Configuration | 617 |
| 28.2.4 | Register Configuration..... | 618 |
| 28.2.5 | Register Descriptions..... | 618 |
| 28.2.6 | DFG/DPG Input Signals | 625 |
| 28.3 | Reference Signal Generators | 626 |
| 28.3.1 | Overview..... | 626 |
| 28.3.2 | Block Diagram | 626 |
| 28.3.3 | Register Configuration..... | 628 |
| 28.3.4 | Register Descriptions..... | 629 |
| 28.3.5 | Description of Operation..... | 635 |
| 28.4 | HSW (Head-switch) Timing Generator..... | 650 |
| 28.4.1 | Overview..... | 650 |
| 28.4.2 | Block Diagram | 650 |
| 28.4.3 | Composition..... | 652 |
| 28.4.4 | Register Configuration..... | 653 |
| 28.4.5 | Register Descriptions..... | 653 |
| 28.4.6 | Description of Operation..... | 669 |
| 28.4.7 | Interrupt | 675 |
| 28.4.8 | Cautions | 676 |
| 28.5 | Four-head High-speed Switching Circuit for Special Playback..... | 677 |
| 28.5.1 | Overview..... | 677 |
| 28.5.2 | Block Diagram | 677 |
| 28.5.3 | Pin Configuration | 678 |
| 28.5.4 | Register Description | 678 |
| 28.6 | Drum Speed Error Detector | 681 |
| 28.6.1 | Overview..... | 681 |
| 28.6.2 | Block Diagram | 681 |

| | | |
|---------|---|-----|
| 28.6.3 | Register Configuration..... | 683 |
| 28.6.4 | Register Descriptions..... | 684 |
| 28.6.5 | Description of Operation | 689 |
| 28.6.6 | f_H Correction in Trick Play Mode | 691 |
| 28.7 | Drum Phase Error Detector | 692 |
| 28.7.1 | Overview | 692 |
| 28.7.2 | Block Diagram | 692 |
| 28.7.3 | Register Configuration..... | 694 |
| 28.7.4 | Register Descriptions..... | 695 |
| 28.7.5 | Description of Operation | 698 |
| 28.7.6 | Phase Comparison | 700 |
| 28.8 | Capstan Speed Error Detector | 701 |
| 28.8.1 | Overview | 701 |
| 28.8.2 | Block Diagram | 701 |
| 28.8.3 | Register Configuration..... | 703 |
| 28.8.4 | Register Descriptions..... | 704 |
| 28.8.5 | Description of Operation | 708 |
| 28.9 | Capstan Phase Error Detector | 710 |
| 28.9.1 | Overview | 710 |
| 28.9.2 | Block Diagram | 710 |
| 28.9.3 | Register Configuration..... | 712 |
| 28.9.4 | Register Descriptions..... | 713 |
| 28.9.5 | Description of Operation | 716 |
| 28.10 | X-Value and Tracking Adjustment Circuit..... | 718 |
| 28.10.1 | Overview | 718 |
| 28.10.2 | Block Diagram | 718 |
| 28.10.3 | Register Descriptions..... | 720 |
| 28.11 | Digital Filters..... | 723 |
| 28.11.1 | Overview | 723 |
| 28.11.2 | Block Diagram | 724 |
| 28.11.3 | Arithmetic Buffer | 726 |
| 28.11.4 | Register Configuration..... | 727 |
| 28.11.5 | Register Descriptions..... | 728 |
| 28.11.6 | Filter Characteristics..... | 736 |
| 28.11.7 | Operations in Case of Transient Response..... | 738 |
| 28.11.8 | Initialization of Z^{-1} | 738 |
| 28.12 | Additional V Signal Generator | 740 |
| 28.12.1 | Overview | 740 |
| 28.12.2 | Pin Configuration | 741 |
| 28.12.3 | Register Configuration..... | 741 |
| 28.12.4 | Register Description | 741 |
| 28.12.5 | Additional V Pulse Signal..... | 743 |
| 28.13 | CTL Circuit | 746 |

| | |
|---|------------|
| 28.13.1 Overview..... | 746 |
| 28.13.2 Block Diagram | 747 |
| 28.13.3 Pin Configuration | 748 |
| 28.13.4 Register Configuration..... | 748 |
| 28.13.5 Register Descriptions..... | 749 |
| 28.13.6 Operation | 763 |
| 28.13.7 CTL Input Section | 766 |
| 28.13.8 Duty Discriminator..... | 769 |
| 28.13.9 CTL Output Section..... | 775 |
| 28.13.10 Trapezoid Waveform Circuit..... | 778 |
| 28.13.11 Note on CTL Interrupt | 779 |
| 28.14 Frequency Dividers..... | 780 |
| 28.14.1 Overview..... | 780 |
| 28.14.2 CTL Frequency Divider..... | 780 |
| 28.14.3 CFG Frequency Divider..... | 784 |
| 28.14.4 DFG Noise Removal Circuit..... | 793 |
| 28.15 Sync Signal Detector..... | 795 |
| 28.15.1 Overview..... | 795 |
| 28.15.2 Block Diagram | 796 |
| 28.15.3 Pin Configuration | 797 |
| 28.15.4 Register Configuration..... | 797 |
| 28.15.5 Register Descriptions..... | 798 |
| 28.15.6 Noise Detection..... | 806 |
| 28.15.7 Sync Signal Detector Activation | 809 |
| 28.16 Servo Interrupt..... | 810 |
| 28.16.1 Overview..... | 810 |
| 28.16.2 Register Configuration..... | 810 |
| 28.16.3 Register Description | 810 |
| 28.17 Module Stop Control Register (MSTPCR)..... | 817 |
| Section 29 Electrical Characteristics..... | 819 |
| 29.1 Absolute Maximum Ratings..... | 819 |
| 29.2 Electrical Characteristics of HD64F2194 | 820 |
| 29.2.1 DC Characteristics of HD64F2194..... | 820 |
| 29.2.2 Allowable Output Currents of HD64F2194, HD64F2194C..... | 826 |
| 29.2.3 AC Characteristics of HD64F2194, HD64F2194C | 827 |
| 29.2.4 Serial Interface Timing of HD64F2194, HD64F2194C..... | 830 |
| 29.2.5 A/D Converter Characteristics of HD64F2194, HD64F2194C | 835 |
| 29.2.6 Servo Section Electrical Characteristics of HD64F2194, HD64F2194C | 836 |
| 29.2.7 FLASH Memory Characteristics | 839 |
| 29.2.8 Usage Note..... | 840 |
| 29.3 Electrical Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A..... | 841 |

| | | |
|---|---|------|
| 29.3.1 | DC Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A | 841 |
| 29.3.2 | Allowable Output Currents of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A..... | 847 |
| 29.3.3 | AC Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A | 848 |
| 29.3.4 | Serial Interface Timing of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A..... | 852 |
| 29.3.5 | A/D Converter Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A..... | 857 |
| 29.3.6 | Servo Section Electrical Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A | 858 |
| Appendix A Instruction Set..... | | 861 |
| A.1 | Instructions | 861 |
| A.2 | Instruction Codes | 872 |
| A.3 | Operation Code Map..... | 882 |
| A.4 | Number of Execution States | 886 |
| A.5 | Bus Status During Instruction Execution | 896 |
| A.6 | Change of Condition Codes..... | 905 |
| Appendix B Internal I/O Registers | | 910 |
| B.1 | Addresses | 910 |
| B.2 | Function List..... | 917 |
| Appendix C Pin Circuit Diagrams..... | | 1018 |
| C.1 | Pin Circuit Diagrams..... | 1018 |
| Appendix D Port States in the Difference Processing States..... | | 1032 |
| D.1 | Pin Circuit Diagrams..... | 1032 |
| Appendix E Usage Notes | | 1033 |
| E.1 | Power Supply Rise and Fall Order..... | 1033 |
| E.2 | Pin Handling When the High-Speed Switching Circuit for Four-Head Special Playback Is Not Used..... | 1034 |
| E.3 | Sample External Circuits..... | 1035 |
| Appendix F List of Product Codes | | 1036 |
| Appendix G External Dimensions..... | | 1037 |

Section 1 Overview

1.1 Overview

The H8S/2194 Series, and H8S/2194C Series comprise microcomputers (MCUs) built around the H8S/2000 CPU, employing Hitachi's proprietary architecture, and equipped with supporting modules on-chip.

The H8S/2000 has an internal 32-bit architecture, is provided with sixteen 16-bit general registers and a concise, optimized instruction set designed for high-speed operation, and can address a 16-Mbyte linear address space. The instruction set is upward-compatible with H8/300 and H8/300H CPU instructions at the object-code level, facilitating migration from the H8/300, H8/300L, or H8/300H Series.

The H8S/2194 Series, and H8S/2194C Series are incorporated with digital servo circuit, ROM, RAM, seven types of timers, three types of PWM, two types of serial communication interface, I²C bus interface, A/D converter, and I/O port as on-chip supporting modules.

The on-chip ROM is either flash memory (F-ZTATTM*) or mask ROM, with a capacity of 256, 192, 160, 128, 112, 96, or 80 kbytes. ROM is connected to the CPU via a 16-bit data bus, enabling both byte and word data to be accessed in one state. Instruction fetching has been speeded up, and processing speed increased.

The features of the H8S/2194 Series, and H8S/2194C Series are shown in table 1.1.

Note: * F-ZTATTM is a trademark of Hitachi, Ltd.

Table 1.1 Features

| Item | Specifications |
|-------|--|
| CPU | <ul style="list-style-type: none">■ General-register architecture<ul style="list-style-type: none">• Sixteen 16-bit general registers (also usable as sixteen 8-bit registers or eight 32-bit registers)■ High-speed operation suitable for real-time control<ul style="list-style-type: none">• Maximum operating frequency: 10 MHz/4 to 5.5 V Operable by 32 kHz subclock• High-speed arithmetic operations 8/16/32-bit register-register add/subtract: 100 ns (10 MHz operation) 16 × 16-bit register-register multiply: 2000 ns (10 MHz operation) 32 ÷ 16-bit register-register divide: 2000 ns (10 MHz operation)■ Instruction set suitable for high-speed operation<ul style="list-style-type: none">• Sixty-five basic instructions• 8/16/32-bit transfer/arithmetic and logic instructions• Unsigned/signed multiply and divide instructions• Powerful bit-manipulation instructions■ CPU operating modes<ul style="list-style-type: none">• Advanced mode: 16-Mbyte address space |
| Timer | <ul style="list-style-type: none">■ Seven types of timer are incorporated<ul style="list-style-type: none">(1) Timer A<ul style="list-style-type: none">• 8-bit interval timer• Clock source can be selected among 8 types of internal clock of which frequencies are divided from the system clock (ϕ) and subclock (ϕSUB)• Functions as clock time base by subclock input(2) Timer B<ul style="list-style-type: none">• Functions as 8-bit interval timer or reload timer• Clock source can be selected among 7 types of internal clock or external event input(3) Timer J<ul style="list-style-type: none">• Functions as two 8-bit down counters or one 16-bit down counter (reload timer/event counter timer/timer output, etc., 5 types of operation modes)• Remote controlled transmit function• Take up/Supply Reel Pulse Frequency division |

| Item | Specifications |
|----------------|---|
| Timer | <p data-bbox="298 73 418 94">(4) Timer L</p> <ul data-bbox="298 113 1060 248" style="list-style-type: none"> <li data-bbox="298 113 570 134">• 8-bit up/down counter <li data-bbox="298 153 1060 212">• Clock source can be selected among 2 types of internal clock, CFG frequency division signal, and PB and REC-CTL (control pulse) <li data-bbox="298 225 913 245">• Compare-match clearing function/auto reload function <p data-bbox="298 264 422 285">(5) Timer R</p> <ul data-bbox="298 304 968 485" style="list-style-type: none"> <li data-bbox="298 304 546 325">• Three reload timers <li data-bbox="298 344 552 365">• Mode discrimination <li data-bbox="298 384 468 405">• Reel control <li data-bbox="298 424 968 445">• Capstan motor acceleration/deceleration detection function <li data-bbox="298 464 606 485">• Slow tracking mono-multi <p data-bbox="298 504 434 525">(6) Timer X1</p> <ul data-bbox="298 544 1132 679" style="list-style-type: none"> <li data-bbox="298 544 624 564">• 16-bit free-running counter <li data-bbox="298 584 1132 604">• Clock source can be selected among 3 types of internal clock and DVCFG <li data-bbox="298 624 642 644">• Two output compare outputs <li data-bbox="298 663 606 684">• Four input capture inputs <p data-bbox="298 703 506 724">(7) Watchdog timer</p> <ul data-bbox="298 743 871 804" style="list-style-type: none"> <li data-bbox="298 743 871 764">• Functions as watchdog timer or 8-bit interval timer <li data-bbox="298 783 787 804">• Generates reset signal or NMI at overflow |
| Prescaler unit | <ul data-bbox="298 823 1112 1070" style="list-style-type: none"> <li data-bbox="298 823 1100 882">• Divides system clock frequency and generates frequency division clock for supporting module functions <li data-bbox="298 895 1112 954">• Divides subclock frequency and generates input clock for Timer A (clock time base) <li data-bbox="298 967 859 987">• Generates 8-bit PWM frequency and duty period <li data-bbox="298 1007 781 1027">• 8-bit input capture at external signal edge <li data-bbox="298 1046 769 1067">• Frequency division clock output enabled |
| PWM | <p data-bbox="298 1090 745 1110">■ Three types of PWM are incorporated</p> <p data-bbox="298 1129 831 1150">(1) 14-bit PWM: Pulse resolution type x 1 channel</p> <p data-bbox="298 1169 787 1190">(2) 8-bit PWM: Duty control type x 4 channels</p> <p data-bbox="298 1209 871 1230">(3) 12-bit PWM: Pulse pitch control type x 2 channels</p> |

| Item | Specifications |
|--------------------------------------|---|
| Serial communication interface (SCI) | <ul style="list-style-type: none"> ■ Two types of serial communication interface is incorporated (1) SCI1 <ul style="list-style-type: none"> • Asynchronous mode or synchronous mode selectable • Desired bit rate selectable with built-in baud rate generator • Multiprocessor communication function (2) SCI2 <ul style="list-style-type: none"> • 32-byte data automatically transferrable • Transfer clock selectable among seven types of internal/external clock |
| I ² C bus interface | <ul style="list-style-type: none"> • Conforms to Phillips I²C bus interface standard • Single master mode/slave mode • Arbitration lost condition can be identified • Supports two slave addresses |
| A/D converter | <ul style="list-style-type: none"> • Resolution: 10 bits • Input: 12 channels • High-speed conversion: 13.4 μs minimum conversion time (10 MHz operation) • Sample-and-hold function • A/D conversion can be activated by software or external trigger |
| Address trap controller | <ul style="list-style-type: none"> • Interrupt occurs when the preset address is found during bus cycle • To-be-trapped addresses can be individually set at three different locations |
| I/O port | <ul style="list-style-type: none"> • 60 input/output pins • 8 input-only pins • Can be switched for each supporting module |
| Servo circuit | <ul style="list-style-type: none"> ■ Digital servo circuits on-chip • Input and output circuits • Error detection circuit • Phase and gain compensation |
| Sync signal detector | <ul style="list-style-type: none"> ■ On-chip sync signal detection circuit • Can separately detect horizontal and vertical sync signals • Noise detection function |

| Item | Specifications | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------------------|---|-----------------|-----------------|----------|-----------------|------------|-------------------|-----------------|------------|------------|------------|------------|----------|------------|------------|-----------|----------|------------|----------|-----------|-----------|----------|-----------|-----------|-----------|--------|-----------|---|-----------|--------|-----------|---|----------|--------|-----------|---|----------|--------|
| Memory | <ul style="list-style-type: none">Flash memory or mask ROMHigh-speed static RAM <table><thead><tr><th>Product Name</th><th>ROM</th><th>RAM</th></tr></thead><tbody><tr><td>H8S/2194C</td><td>256 kbytes</td><td>6 kbytes</td></tr><tr><td>H8S/2194B</td><td>192 kbytes</td><td>6 kbytes</td></tr><tr><td>H8S/2194A</td><td>160 kbytes</td><td>6 kbytes</td></tr><tr><td>H8S/2194</td><td>128 kbytes</td><td>3 kbytes</td></tr><tr><td>H8S/2193</td><td>112 kbytes</td><td>3 kbytes</td></tr><tr><td>H8S/2192</td><td>96 kbytes</td><td>3 kbytes</td></tr><tr><td>H8S/2191</td><td>80 kbytes</td><td>3 kbytes</td></tr></tbody></table> | Product Name | ROM | RAM | H8S/2194C | 256 kbytes | 6 kbytes | H8S/2194B | 192 kbytes | 6 kbytes | H8S/2194A | 160 kbytes | 6 kbytes | H8S/2194 | 128 kbytes | 3 kbytes | H8S/2193 | 112 kbytes | 3 kbytes | H8S/2192 | 96 kbytes | 3 kbytes | H8S/2191 | 80 kbytes | 3 kbytes | | | | | | | | | | | | | |
| Product Name | ROM | RAM | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| H8S/2194C | 256 kbytes | 6 kbytes | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| H8S/2194B | 192 kbytes | 6 kbytes | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| H8S/2194A | 160 kbytes | 6 kbytes | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| H8S/2194 | 128 kbytes | 3 kbytes | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| H8S/2193 | 112 kbytes | 3 kbytes | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| H8S/2192 | 96 kbytes | 3 kbytes | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| H8S/2191 | 80 kbytes | 3 kbytes | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Power-down state | <ul style="list-style-type: none">Medium-speed modeSleep modeModule stop modeStandby modeSubclock operationSubactive mode, watch mode, subsleep mode | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Interrupt controller | <ul style="list-style-type: none">Seven external interrupt pins (NMI, IRQ5 to IRQ0)38 internal interrupt sourcesThree priority levels settable | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Clock pulse generator | <ul style="list-style-type: none">■ Two types of clock pulse generator on-chipSystem clock pulse generator: 8 to 10 MHzSubclock pulse generator: 32.768 kHz | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Packages | <ul style="list-style-type: none">112-pin plastic QFP (FP-112) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Product lineup | <table><thead><tr><th rowspan="2">Series</th><th colspan="2">Product Code</th><th rowspan="2">ROM/RAM (bytes)</th><th rowspan="2">Packages</th></tr><tr><th>Mask ROM Versions</th><th>F-ZTAT Versions</th></tr></thead><tbody><tr><td rowspan="3">H8S/2194C</td><td>HD6432194C</td><td>HD64F2194C</td><td>256 k/6 k</td><td>FP-112</td></tr><tr><td>HD6432194B</td><td>—</td><td>192 k/6 k</td><td>FP-112</td></tr><tr><td>HD6432194A</td><td>—</td><td>160 k/6 k</td><td>FP-112</td></tr><tr><td rowspan="4">H8S/2194</td><td>HD6432194</td><td>HD64F2194</td><td>128 k/3 k</td><td>FP-112</td></tr><tr><td>HD6432193</td><td>—</td><td>112 k/3 k</td><td>FP-112</td></tr><tr><td>HD6432192</td><td>—</td><td>96 k/3 k</td><td>FP-112</td></tr><tr><td>HD6432191</td><td>—</td><td>80 k/3 k</td><td>FP-112</td></tr></tbody></table> | Series | Product Code | | ROM/RAM (bytes) | Packages | Mask ROM Versions | F-ZTAT Versions | H8S/2194C | HD6432194C | HD64F2194C | 256 k/6 k | FP-112 | HD6432194B | — | 192 k/6 k | FP-112 | HD6432194A | — | 160 k/6 k | FP-112 | H8S/2194 | HD6432194 | HD64F2194 | 128 k/3 k | FP-112 | HD6432193 | — | 112 k/3 k | FP-112 | HD6432192 | — | 96 k/3 k | FP-112 | HD6432191 | — | 80 k/3 k | FP-112 |
| Series | Product Code | | ROM/RAM (bytes) | Packages | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | Mask ROM Versions | F-ZTAT Versions | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| H8S/2194C | HD6432194C | HD64F2194C | 256 k/6 k | FP-112 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | HD6432194B | — | 192 k/6 k | FP-112 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | HD6432194A | — | 160 k/6 k | FP-112 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| H8S/2194 | HD6432194 | HD64F2194 | 128 k/3 k | FP-112 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | HD6432193 | — | 112 k/3 k | FP-112 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | HD6432192 | — | 96 k/3 k | FP-112 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | HD6432191 | — | 80 k/3 k | FP-112 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

1.2 Internal Block Diagram

An internal block diagram of the chip is shown in figure 1.1.

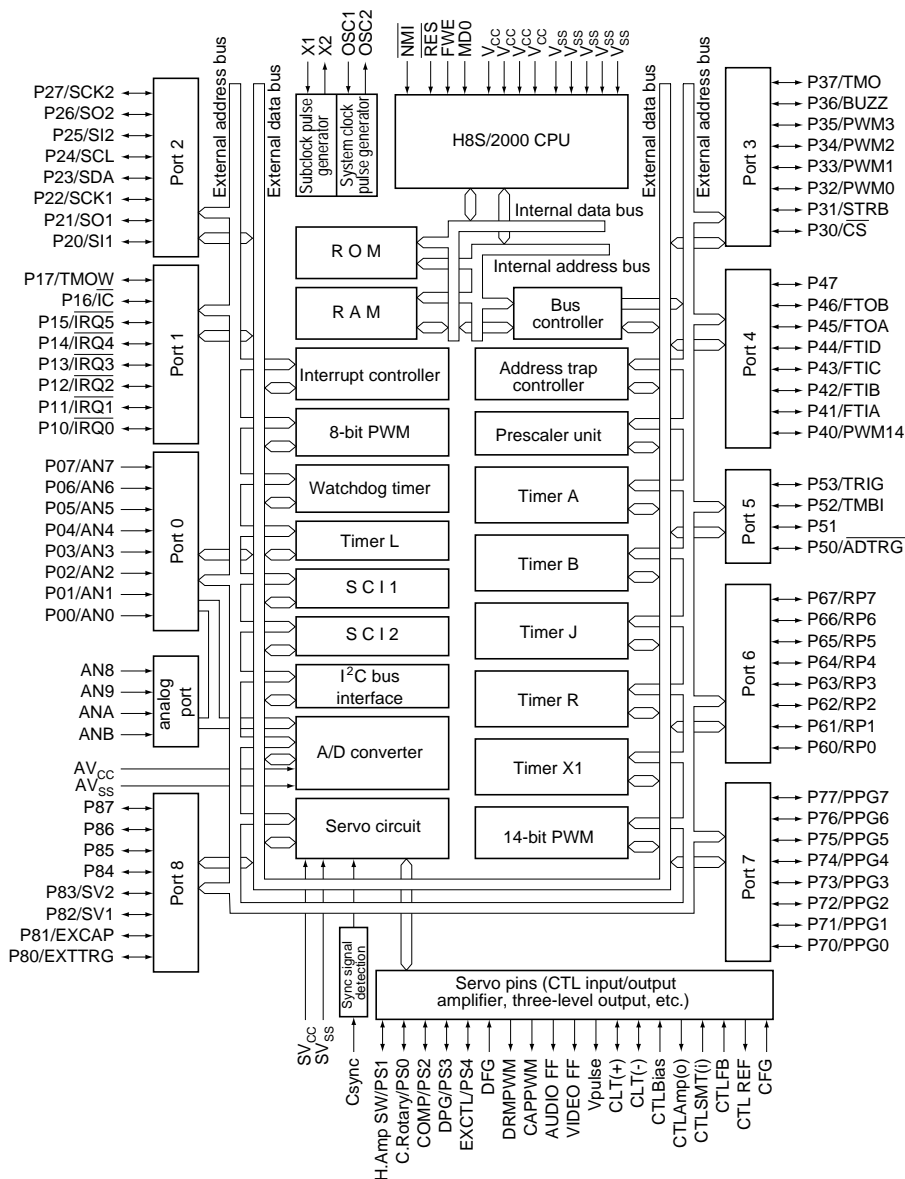


Figure 1.1 Internal Block Diagram of H8S/2194 Series

1.3 Pin Arrangement and Functions

1.3.1 Pin Arrangement

The pin arrangement of the chip is shown in figure 1.2.

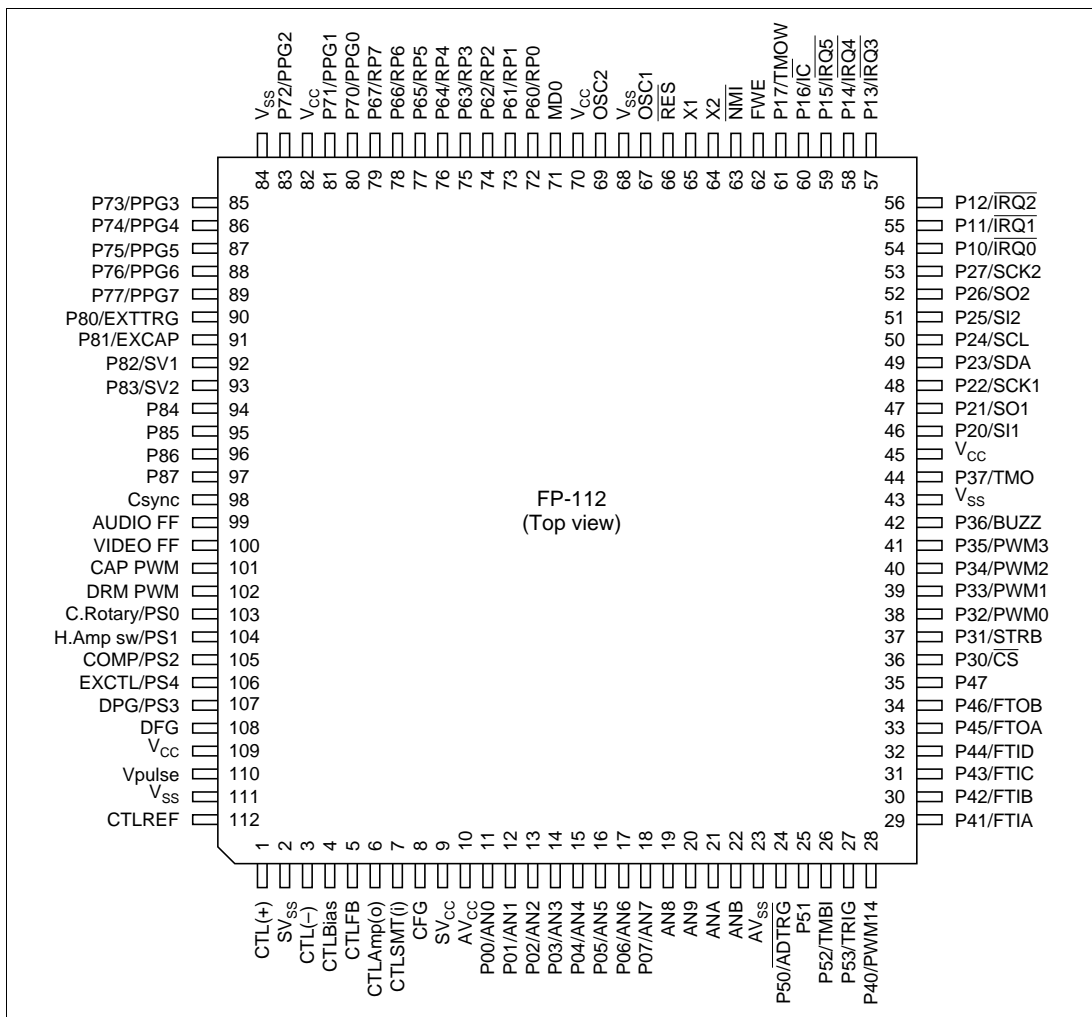


Figure 1.2 Pin Arrangement of H8S/2194 Series

1.3.2 Pin Functions

Table 1.2 summarizes the functions of the chip's pins.

Table 1.2 Pin Functions

| Type | Symbol | Pin No. | I/O | Name and Function |
|------------------------|--------|-----------------|--------|--|
| Power supply | Vcc | 45, 70, 82, 109 | Input | <u>Power supply:</u> All Vcc pins should be connected to the system power supply (+5V) |
| | Vss | 43, 68, 84, 111 | Input | <u>Ground:</u> All Vss pins should be connected to the system power supply (0V) |
| | SVcc | 9 | Input | <u>Servo power supply:</u> SVcc pin should be connected to the servo analog power supply (+5V) |
| | SVss | 2 | Input | <u>Servo ground:</u> SVss pin should be connected to the servo analog power supply (0V) |
| | AVcc | 10 | Input | <u>Analog power supply:</u> Power supply pin for A/D converter. It should be connected to the system power supply (+5V) when the A/D converter is not used |
| | AVss | 23 | Input | <u>Analog ground:</u> Ground pin for A/D converter. It should be connected to the system power supply (0V) |
| Clock | OSC1 | 67 | Input | Connected to a crystal oscillator. It can also input an external clock. See section 10, Clock Pulse Generator, for typical connection diagrams for a crystal oscillator and external clock input |
| | OSC2 | 69 | Output | |
| | X1 | 65 | Input | Connected to a 32.768 kHz crystal oscillator. See section 10, Clock Pulse Generator, for typical connection diagrams |
| | X2 | 64 | Output | |
| Operating mode control | MD0 | 71 | Input | <u>Mode pins:</u> These pins set the operating mode. These pins should not be changed while the MCU is in operation |

| Type | Symbol | Pin No. | I/O | Name and Function |
|----------------|--------------------------|---------|--------|---|
| System control | RES | 66 | Input | <u>Reset input:</u> When this pin is driven low, the chip is reset |
| | FWE | 62 | Input | <u>Flash memory enable:</u> Enables/disables flash memory programming. This pin is available only with MCU with flash memory on-chip. For mask ROM type, do not connect anything to this pin |
| Interrupts | $\overline{\text{IRQ0}}$ | 54 | Input | <u>External interrupt request 0:</u> External interrupt input pin for which rising edge sense, falling edge sense or both edges sense are selectable |
| | $\overline{\text{IRQ1}}$ | 55 | Input | <u>External interrupt requests 1 to 5:</u> External interrupt input pins for which rising or falling edge sense are selectable |
| | $\overline{\text{IRQ2}}$ | 56 | | |
| | $\overline{\text{IRQ3}}$ | 57 | | |
| | $\overline{\text{IRQ4}}$ | 58 | | |
| | $\overline{\text{IRQ5}}$ | 59 | | |
| Prescaler unit | NMI | 63 | Input | <u>Nonmaskable interrupt:</u> Nonmaskable interrupt input pin for which rising edge sense, falling edge sense or both edges sense are selectable |
| | $\overline{\text{IC}}$ | 60 | Input | <u>Input capture input:</u> Input capture input pin for prescaler unit |
| | TMOW | 61 | Output | <u>Frequency division clock output:</u> Output pin for clock of which frequency is divided by prescaler |
| Timers | TMBI | 26 | Input | <u>Timer B event input:</u> Input pin for events to be input to Timer B counter |
| | $\overline{\text{IRQ1}}$ | 55 | Input | <u>Timer J event input:</u> Input pin for events to be input to Timer J RDT-1 or RDT-2 counter |
| | $\overline{\text{IRQ2}}$ | 56 | | |
| | TMO | 44 | Output | <u>Timer J timer output:</u> Output pin for toggle at underflow of RDT-1 of Timer J, or remote controlled transmit data |
| | BUZZ | 42 | Output | <u>Timer J buzzer output:</u> Output pin for toggle which is selectable among fixed frequency, 1Hz frequency divided from subclock (32 kHz), and frequency division CTL signal |

| Type | Symbol | Pin No. | I/O | Name and Function |
|--------------------------------------|--------|---------|---------------|--|
| Timers | IRQ3 | 57 | Input | <u>Timer R input capture:</u> Input pin for input capture of Timer R TMRU-1 or TMRU-2 |
| | FTOA | 33 | Output | <u>Timer X1 output compare A and B output:</u> |
| | FTOB | 34 | | Output pin for output compare A and B of Timer X1 |
| | FTIA | 29 | Input | <u>Timer X1 input capture A, B, C and D input:</u> |
| | FTIB | 30 | | Input pin for input capture A, B, C and D of |
| | FTIC | 31 | | Timer X1 |
| | FTID | 32 | | |
| PWM | PWM0 | 38 | Output | <u>8-bit PWM square waveform output:</u> |
| | PWM1 | 39 | | Output pin for waveform generated by 8-bit |
| | PWM2 | 40 | | PWM 0, 1, 2 and 3 |
| | PWM3 | 41 | | |
| Serial communication interface (SCI) | PWM14 | 28 | Output | <u>14-bit PWM square waveform output:</u> Output pin for waveform generated by 14-bit PWM |
| | SCK1 | 48 | Input | <u>SCI clock input/output:</u> |
| | SCK2 | 53 | /output | Clock input pins for SCI 1 and 2 |
| | SI1 | 46 | Input | <u>SCI receive data input:</u> |
| | SI2 | 51 | | Receive data input pins for SCI 1 and 2 |
| | SO1 | 47 | Output | <u>SCI transmit data output:</u> |
| | SO2 | 52 | | Transmit data output pins for SCI 1 and 2 |
| | STRB | 37 | Output | <u>SCI2 strobe output:</u> This pin outputs strobe pulse for each byte transmit by SCI2 |
| | CS | 36 | Input | <u>SCI2 chip select input:</u> This pin controls the transfer start of SCI2 |
| I ² C bus interface | SCL | 50 | Input /output | <u>I²C bus interface clock input/output:</u> Clock input/output pin for I ² C bus interface |
| | SDA | 49 | Input /output | <u>I²C bus interface data input/output:</u> Data input/output pin for I ² C bus interface |

| Type | Symbol | Pin No. | I/O | Name and Function |
|----------------|--------------------------|----------------------|----------------------|--|
| A/D converter | AN7 to AN0 | 18 to 11 | Input | <u>Analog input channels 7 to 0:</u> Analog data input pins. A/D conversion is started by a software triggering |
| | AN8 AN9 ANA ANB | 19 20 21 22 | Input | <u>Analog input channels 8, 9, A and B:</u> Analog data input pins. A/D conversion is started by an external, hardware, or software triggering |
| | ADTRG | 24 | Input | <u>A/D conversion external trigger input:</u> Pin for input of an external trigger to start A/D conversion |
| | | | | |
| Servo circuits | AUDIO FF | 99 | Output | <u>Audio FF:</u> Output pin for audio head switching signal |
| | VIDEO FF | 100 | Output | <u>Video FF:</u> Output pin for video head switching signal |
| | CAPPWM | 101 | Output | <u>Capstan mix:</u> 12-bit PWM output pin giving result of capstan speed error and phase error after filtering |
| | DRMPWM | 102 | Output | <u>Drum mix:</u> 12-bit PWM output pin giving result of drum speed error and phase error after filtering |
| | Vpulse | 110 | Output | <u>Additional V pulse:</u> Three-level output pin for additional V signal synchronized to the Video FF signal |
| | C.Rotary /PS0 | 103 | Output, input/output | <u>Color rotary signal:</u> Output pin for color signal processing control signal in four-head special-effects playback |
| | H.AmpSW /PS1 | 104 | Output, input/output | <u>Head-amp switch:</u> Output pin for preamplifier output select signal in four-head special-effects playback. This pin can also be used as a general port when not used |
| | COMP /PS2 | 105 | Input, input/output | <u>Compare input:</u> Input pin for signal giving the result of preamplifier output comparison in four-head special-effects playback. This pin can also be used as a general port when not used |
| | CTL (+) CTL (-) | 1 3 | Input /output | <u>CTL head (+) and (-) pins:</u> I/O pins for CTL signals |
| | CTL Bias | 4 | Input | <u>CTL primary amp bias supply:</u> Bias supply pin for CTL primary amp |

| Type | Symbol | Pin No. | I/O | Name and Function |
|----------------|--------------|----------------------|---------------------|---|
| Servo circuits | CTL Amp (o) | 6 | Output | <u>CTL amp output:</u> Output pin for CTL amp |
| | CTL SMT (I) | 7 | Input | <u>CTL Schmitt amp input:</u> Input pin for CTL Schmitt amp |
| | CTLFB | 5 | Input | <u>CLT feedback input:</u> Input pin for CTL amp high-range characteristics control |
| | CTLREF | 112 | Output | <u>CTL amp reference voltage output:</u> Output pin for 1/2Vcc (SV) |
| | CFG | 8 | Input | <u>Capstan FG input:</u> Schmitt comparator input pin for CFG signal |
| | DFG | 108 | Input | <u>Drum FG input:</u> Schmitt input pin for DFG signal |
| | DPG/PS3 | 107 | Input, input/output | <u>Drum PG input:</u> Schmitt input pin for DPG signal. This pin can also be used as a general port when not used |
| | EXCTL /PS4 | 106 | Input, input/output | <u>External CTL input:</u> Input pin for external CTL signal. This pin can also be used as a general port when not used |
| | Csync | 98 | Input | <u>Mixed sync signal input:</u> Input pin for mixed sync signal |
| | EXCAP | 91 | Input | <u>Capstan external sync signal input:</u> Signal input pin for external synchronization of capstan phase control |
| | EXTTRG | 90 | Input | <u>External trigger signal input:</u> Signal input pin for synchronization with reference signal generator |
| | SV1 | 92 | Output | <u>Servo monitor output pin 1:</u> Output pin for servo module internal signal |
| | SV2 | 93 | Output | <u>Servo monitor output pin 2:</u> Output pin for servo module internal signal |
| | PPG7 to PPG0 | 89 to 85, 83, 81, 80 | Output | <u>PPG:</u> Output pin for HSW timing generator. To be used when head switching is required as well as Audio FF and Video FF |

| Type | Symbol | Pin No. | I/O | Name and Function |
|----------|------------|----------------------------|------------------|--|
| I/O port | P07 to P00 | 11 to 18 | Input | <u>Port 0:</u> 8-bit input pins |
| | P17 to P10 | 61 to 54 | Input /output | <u>Port 1:</u> 8-bit I/O pins |
| | P27 to P20 | 53 to 46 | Input /output | <u>Port 2:</u> 8-bit I/O pins |
| | P37 to P30 | 44, 42 to 36 | Input /output | <u>Port 3:</u> 8-bit I/O pins |
| | P47 to P40 | 35 to 28 | Input /output | <u>Port 4:</u> 8-bit I/O pins |
| | P53 to P50 | 27 to 24 | Input /output | <u>Port 5:</u> 4-bit I/O pins |
| | P67 to P60 | 79 to 72 | Input /output | <u>Port 6:</u> 8-bit I/O pins |
| | P77 to P70 | 89 to 85, 83, 81, 80 | Input /output | <u>Port 7:</u> 8-bit I/O pins |
| | P87 to P80 | 97 to 90 | Input /output | <u>Port 8:</u> 8-bit I/O pins |
| | RP7 to RP0 | 79 to 72 | Output | <u>Realtime output port:</u> 8-bit realtime output pins |
| | TRIG | 27 | Input | <u>Realtime output port trigger input:</u> Input pin for realtime output port trigger |

1.4 Differences between H8S/2194C Series and H8S/2194 Series

Though the H8S/2194C series is compatible with the H8S/2194 series and their supporting modules are almost identical, there are some differences between them as shown below. For details, see the following sections.

Table 1.3 Differences between H8S/2194C series and H8S/2194 series

| | H8S/2194C Series | H8S/2194 Series |
|---------------|--|--|
| ROM | H8S/2194C: 256 kbytes H8S/2194B: 192 kbytes H8S/2194A: 160 kbytes | H8S/2194: 128 kbytes H8S/2193: 112 kbytes H8S/2192: 96 kbytes H8S/2191: 80 kbytes |
| RAM | H8S/2194C: 6 kbytes H8S/2194B: 6 kbytes H8S/2194A: 6 kbytes | H8S/2194: 3 kbytes H8S/2193: 3 kbytes H8S/2192: 3 kbytes H8S/2191: 3 kbytes |
| Timer J | Five operating modes: TMJ-2 input clock sources: $PSS=\phi/16384$, $\phi/2048$, or $\phi/1024$; underflow of TMJ-1, external clock (IRQ2) | Four operating modes: TMJ-2 input clock sources: $PSS=\phi/16384$ or $\phi/2048$; underflow of TMJ-1, external clock (IRQ2) |
| Servo circuit | In the reference signal generator, the servo circuit selects whether reference signals are generated with VD when it is in PB mode, or in free-run | In the reference signal generator, when the servo circuit is in PB mode, reference signals are generated in free-run |
| Flash ROM | 256 kbytes When the flash ROM control flag is set, use the E (erase) bit and the P (program) bit in flash memory control register 1 (FMLCR1). | 128 kbytes When the flash ROM control flag is set, use the E (erase) bit and the P (program) bit in flash memory control register 2 (FMLCR2). |

2.1 Overview

The H8S/2000 CPU is a high-speed central processing unit with an internal 32-bit architecture that is upward-compatible with the H8/300 and H8/300H CPUs. The H8S/2000 CPU has sixteen 16-bit general registers, can address a 16-Mbyte (architecturally 4-Gbyte) linear address space, and is ideal for realtime control.

2.1.1 Features

The H8S/2000 CPU has the following features.

- Upward-compatible with H8/300 and H8/300H CPUs
Can execute H8/300 and H8/300H object programs
- General-register architecture
Sixteen 16-bit general registers (also usable as sixteen 8-bit registers or eight 32-bit registers)
- Sixty-five basic instructions
8/16/32-bit arithmetic and logic instructions
Multiply and divide instructions
Powerful bit-manipulation instructions
- Eight addressing modes
Register direct [Rn]
Register indirect [@ERn]
Register indirect with displacement [@(d:16,ERn) or @(d:32,ERn)]
Register indirect with post-increment or pre-decrement [@ERn+ or @-ERn]
Absolute address [@aa:8, @aa:16, @aa:24, or @aa:32]
Immediate [#xx:8, #xx:16, or #xx:32]
Program-counter relative [@(d:8,PC) or @(d:16,PC)]
Memory indirect [@@aa:8]
- 16-Mbyte address space
Program: 16 Mbytes
Data: 16 Mbytes (4 Gbytes architecturally)
- High-speed operation
All frequently-used instructions execute in one or two states
Maximum clock rate: 10 MHz
8/16/32-bit register-register add/subtract: 100 ns
8 × 8-bit register-register multiply: 1200 ns

16 ÷ 8-bit register-register divide: 1200 ns
 16 × 16-bit register-register multiply: 2000 ns
 32 ÷ 16-bit register-register divide: 2000 ns

- Two CPU operating modes
Normal mode*/Advanced mode
- Power-down state
Transition to power-down state by SLEEP instruction
CPU clock speed selection

Note: * Normal mode is not available for this LSI.

2.1.2 Differences between H8S/2600 CPU and H8S/2000 CPU

The differences between the H8S/2600 CPU and the H8S/2000 CPU are shown below.

- Register configuration
The MAC register is supported only by the H8S/2600 CPU.
- Basic instructions
The four instructions MAC, CLRMAC, LDMAC, and STMAC are supported only by the H8S/2600 CPU.
- Number of execution states
The number of execution states of the MULXU and MULXS instructions differ as follows.

| Instruction | Mnemonic | Number of Execution States | |
|-------------|-----------------|----------------------------|----------|
| | | H8S/2600 | H8S/2000 |
| MULXU | MULXU.B Rs, Rd | 3 | 12 |
| | MULXU.W Rs, Erd | 4 | 20 |
| MULXS | MULXS.B Rs, Rd | 4 | 13 |
| | MULXS.W Rs, Erd | 5 | 21 |

There are also differences in the address space, EXR register functions, power-down state, etc., depending on the product.

2.1.3 Differences from H8/300 CPU

In comparison to the H8/300 CPU, the H8S/2000 CPU has the following enhancements.

- More general registers and control registers
Eight 16-bit extended registers, and one 8-bit control register, have been added.

- Expanded address space
Normal mode supports the same 64-kbyte address space as the H8/300 CPU.
Advanced mode supports a maximum 16-Mbyte address space.
- Enhanced addressing mode
The addressing modes have been enhanced to make effective use of the 16-Mbyte address space.
- Enhanced instructions
Addressing modes of bit-manipulation instructions have been enhanced.
Signed multiply and divide instructions have been added.
Two-bit shift instructions have been added.
Instructions for saving and restoring multiple registers have been added.
A test and set instruction has been added.
- Higher speed
Basic instructions execute twice as fast.

2.1.4 Differences from H8/300H CPU

In comparison to the H8/300H CPU, the H8S/2000 CPU has the following enhancements.

- Additional control register
One 8-bit control register has been added.
- Enhanced instructions
Addressing modes of bit-manipulation instructions have been enhanced.
Two-bit shift instructions have been added.
Instructions for saving and restoring multiple registers have been added.
A test and set instruction has been added.
- Higher speed
Basic instructions execute twice as fast.

2.2 CPU Operating Modes

The H8S/2000 CPU has two operating modes: normal and advanced. Normal mode supports a maximum 64-kbyte address space. Advanced mode supports a maximum 16-Mbyte total address space (architecturally the maximum total address space is 4 Gbytes, with a maximum of 16 Mbytes for the program area and a maximum of 4 Gbytes for the data area).

The mode is selected by the mode pins of the microcontroller.

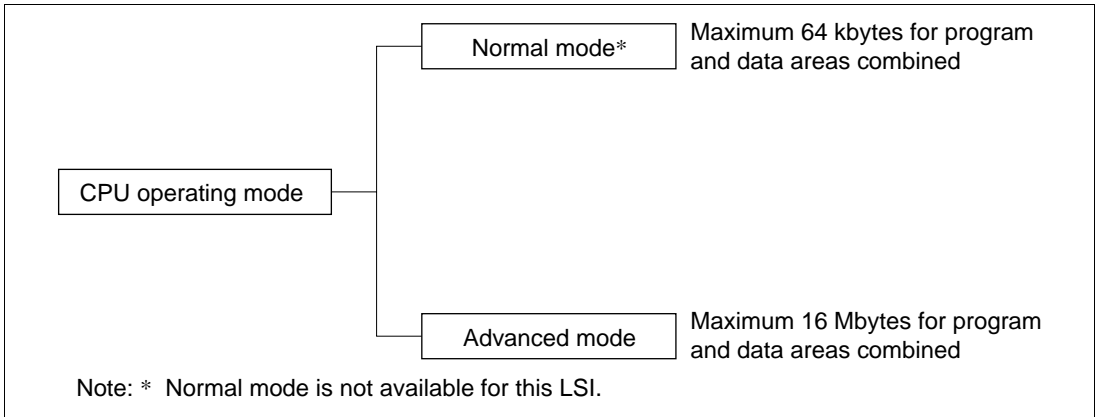


Figure 2.1 CPU Operating Modes

(1) Normal Mode

The exception vector table and stack have the same structure as in the H8/300 CPU.

(a) Address Space

A maximum address space of 64 kbytes can be accessed.

(b) Extended Registers (En)

The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers. When En is used as a 16-bit register it can contain any value, even when the corresponding general register (Rn) is used as an address register. If the general register is referenced in the register indirect addressing mode with pre-decrement (@-Rn) or post-increment (@Rn+) and a carry or borrow occurs, however, the value in the corresponding extended register (En) will be affected.

(c) Instruction Set

All instructions and addressing modes can be used. Only the lower 16 bits of effective addresses (EA) are valid.

(d) Exception Vector Table and Memory Indirect Branch Addresses

In normal mode the top area starting at H'0000 is allocated to the exception vector table. One branch address is stored per 16 bits. The configuration of the exception vector table in normal mode is shown in figure 2.2. For details of the exception vector table, see section 5, Exception Handling.

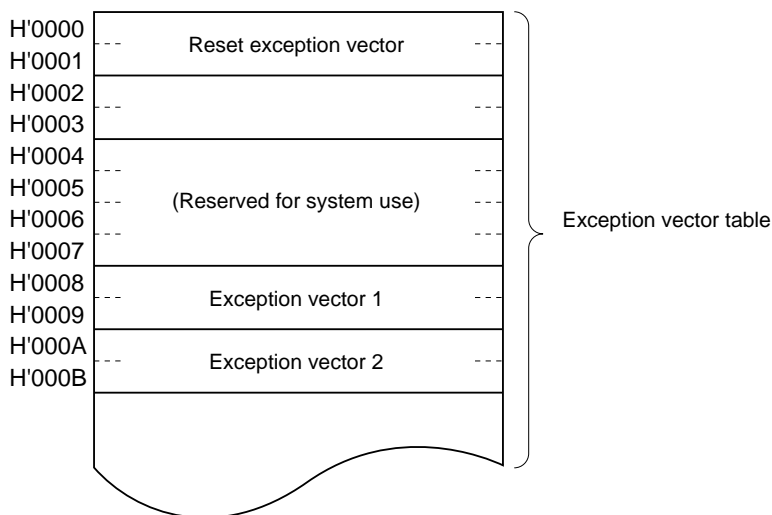


Figure 2.2 Exception Vector Table (Normal Mode)

The memory indirect addressing mode (@@aa:8) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In normal mode the operand is a 16-bit word operand, providing a 16-bit branch address. Branch addresses can be stored in the top area from H'0000 to H'00FF. Note that this area is also used for the exception vector table.

(e) Stack Structure

When the program counter (PC) is pushed onto the stack in a subroutine call, and the PC and condition-code register (CCR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.3. The extended control register (EXR) is not pushed onto the stack. For details, see section 5, Exception Handling.

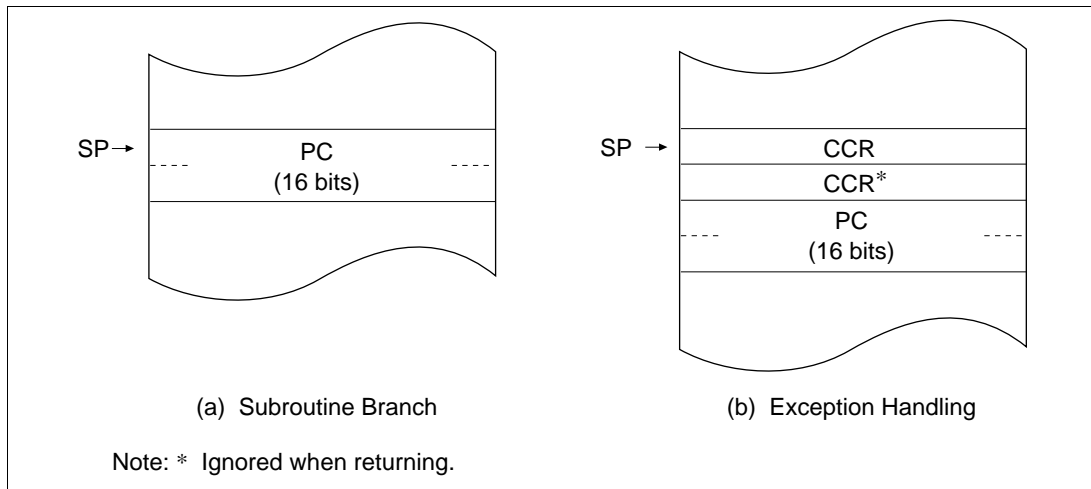


Figure 2.3 Stack Structure in Normal Mode

(2) Advanced Mode

(a) Address Space

Linear access is provided to a 16-Mbyte maximum address space (architecturally a maximum 16-Mbyte program area and a maximum 4-Gbyte data area, with a maximum of 4 Gbytes for program and data areas combined).

(b) Extended Registers (En)

The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers or address registers.

(c) Instruction Set

All instructions and addressing modes can be used.

(d) Exception Vector Table and Memory Indirect Branch Addresses

In advanced mode the top area starting at H'00000000 is allocated to the exception vector table in units of 32 bits. In each 32 bits, the upper 8 bits are ignored and a branch address is stored in the lower 24 bits (figure 2.4). For details of the exception vector table, see section 5, Exception Handling.

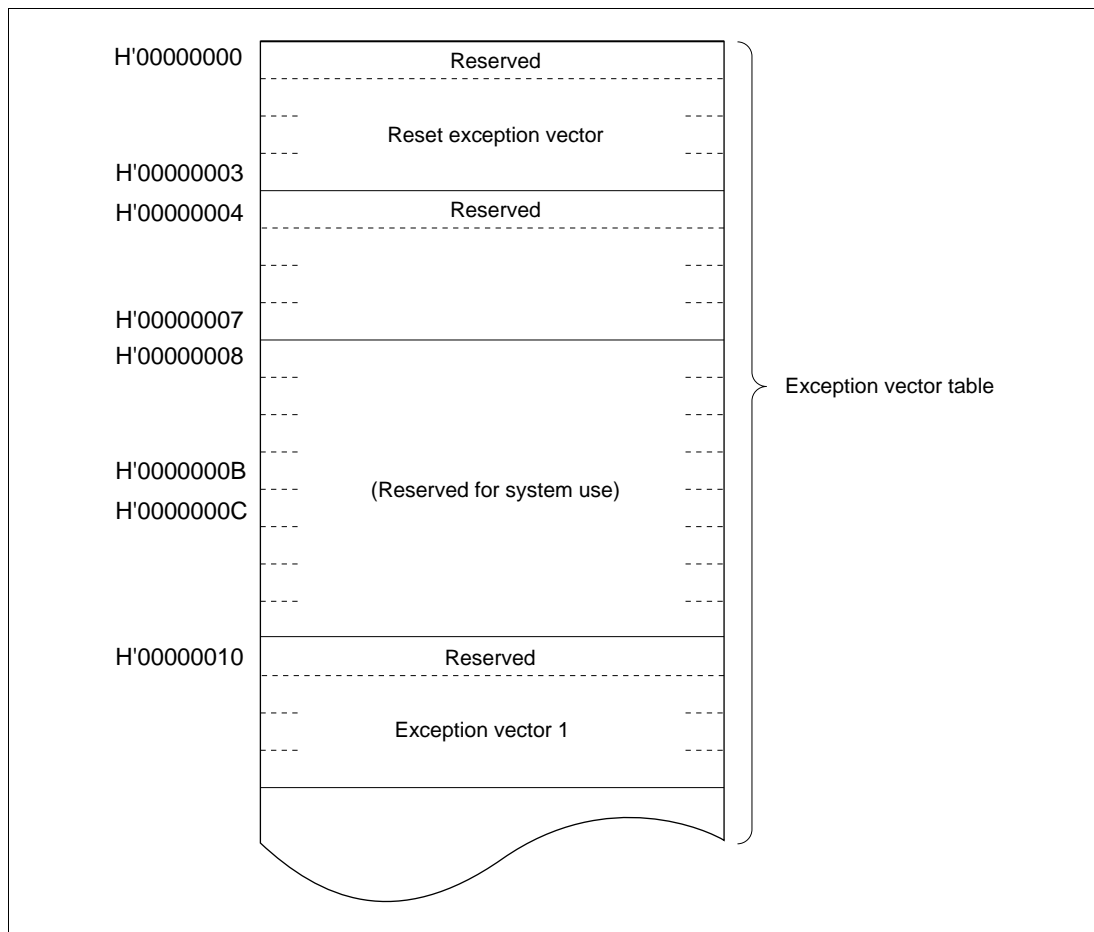


Figure 2.4 Exception Vector Table (Advanced Mode)

The memory indirect addressing mode (@@aa:8) employed in the JMP and JSR instructions uses an 8-bit absolute address included in the instruction code to specify a memory operand that contains a branch address. In advanced mode the operand is a 32-bit longword operand, providing a 32-bit branch address. The upper 8 bits of these 32 bits are a reserved area that is regarded as H'00. Branch addresses can be stored in the area from H'00000000 to H'000000FF. Note that the first part of this range is also the exception vector table.

(e) Stack Structure

In advanced mode, when the program counter (PC) is pushed onto the stack in a subroutine call, and the PC and condition-code register (CCR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.5. The extended control register (EXR) is not pushed onto the stack. For details, see section 5, Exception Handling.

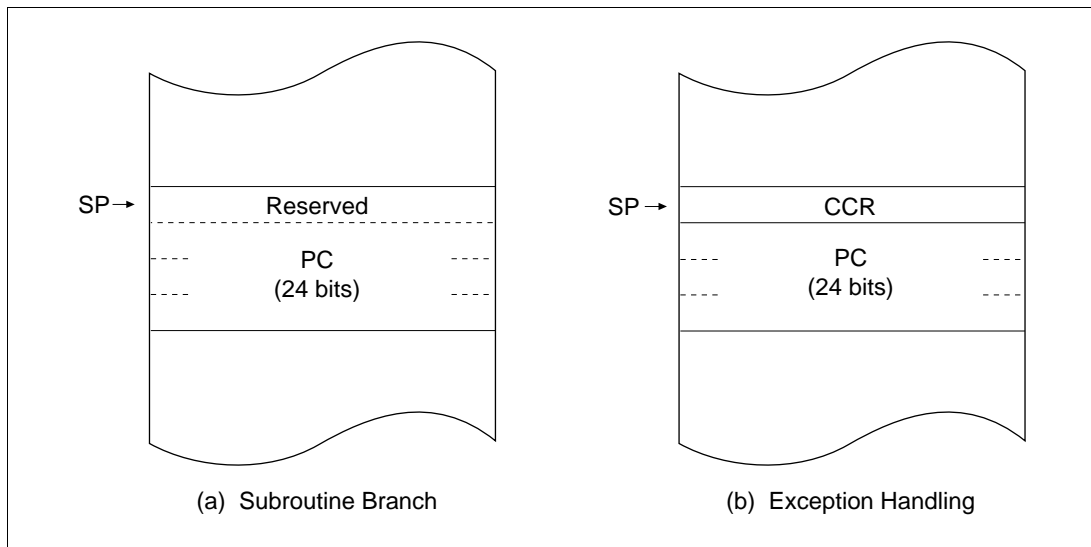


Figure 2.5 Stack Structure in Advanced Mode

2.3 Address Space

Figure 2.6 shows a memory map of the H8S/2000 CPU. The H8S/2000 CPU provides linear access to a maximum 64-kbyte address space in normal mode, and a maximum 16-Mbyte (architecturally 4-Gbyte) address space in advanced mode.

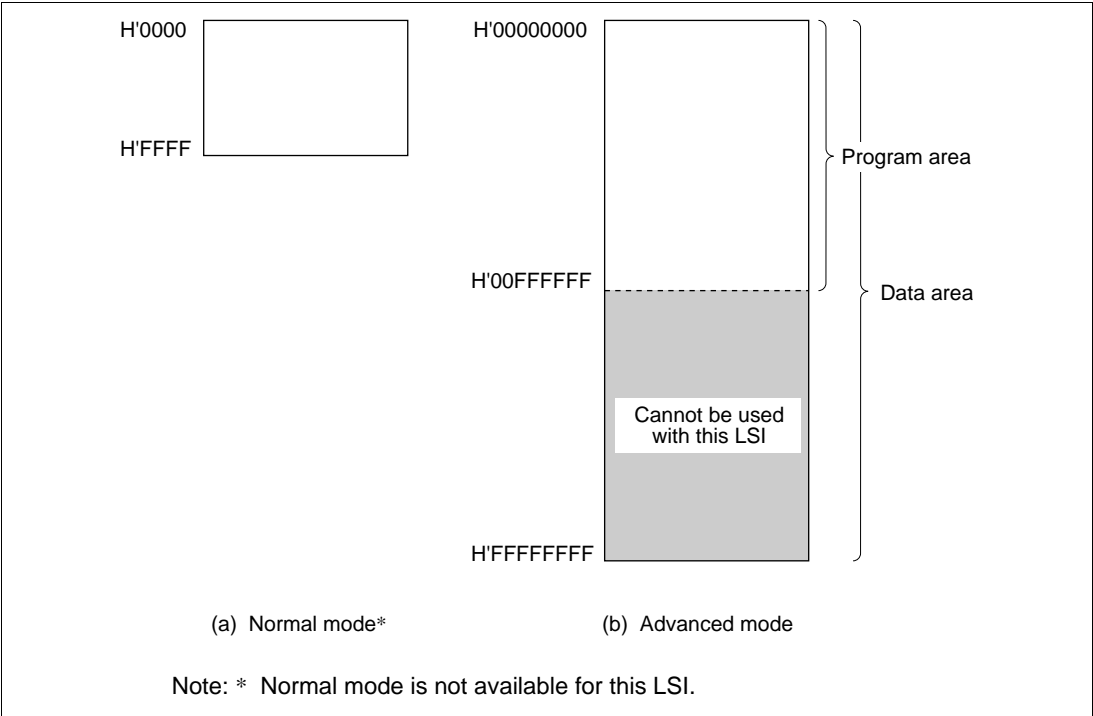


Figure 2.6 Memory Map

2.4 Register Configuration

2.4.1 Overview

The CPU has the internal registers shown in figure 2.7. There are two types of registers: general registers and control registers.

General Registers (Rn) and Extended Registers (En)

| | 15 | 0 7 | 0 7 | 0 |
|----------|----|-----|-----|---|
| ER0 | E0 | R0H | R0L | |
| ER1 | E1 | R1H | R1L | |
| ER2 | E2 | R2H | R2L | |
| ER3 | E3 | R3H | R3L | |
| ER4 | E4 | R4H | R4L | |
| ER5 | E5 | R5H | R5L | |
| ER6 | E6 | R6H | R6L | |
| ER7 (SP) | E7 | R7H | R7L | |

Control Registers (CR)

| | |
|----|---|
| 23 | 0 |
| PC | |

| | | | | | | | | |
|------|---|---|---|---|---|----|----|----|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EXR* | T | - | - | - | - | I2 | I1 | I0 |

| | | | | | | | | |
|-----|---|----|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CCR | I | UI | H | U | N | Z | V | C |

[Legend]

SP : Stack pointer
PC : Program counter
EXR : Extended control register
T : Trace bit
I2 to I0 : Interrupt mask bits
CCR : Condition-code register
I : Interrupt mask bit
UI : User bit or interrupt mask bit

H : Half-carry flag
U : User bit
N : Negative flag
Z : Zero flag
V : Overflow flag
C : Carry flag

Note: * Does not affect operation in this LSI.

Figure 2.7 CPU Registers

2.4.2 General Registers

The CPU has eight 32-bit general registers. These general registers are all functionally alike and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

The ER registers divide into 16-bit general registers designated by the letters E (E0 to E7) and R (R0 to R7). These registers are functionally equivalent, providing a maximum of sixteen 16-bit registers. The E registers (E0 to E7) are also referred to as extended registers.

The R registers divide into 8-bit general registers designated by the letters RH (R0H to R7H) and RL (R0L to R7L). These registers are functionally equivalent, providing a maximum of sixteen 8-bit registers.

Figure 2.8 illustrates the usage of the general registers. The usage of each register can be selected independently.

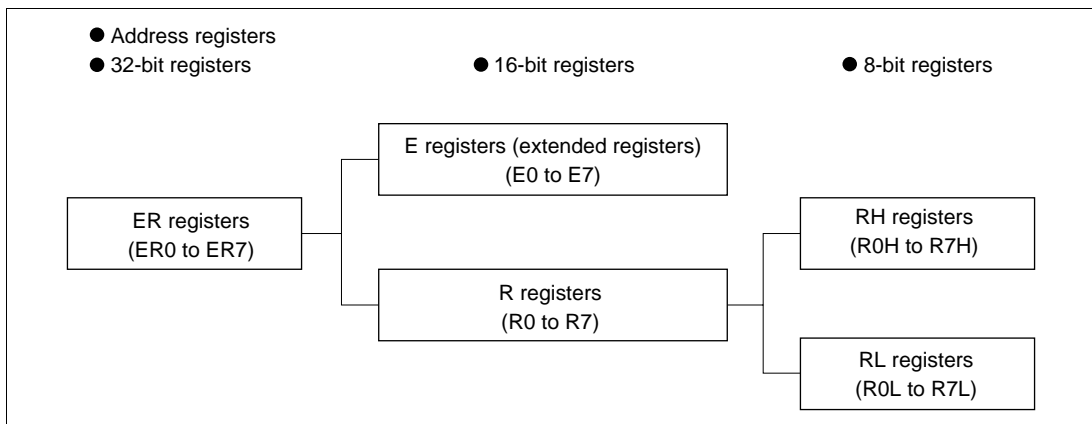


Figure 2.8 Usage of General Registers

General register ER7 has the function of stack pointer (SP) in addition to its general-register function, and is used implicitly in exception handling and subroutine calls. Figure 2.9 shows the stack.

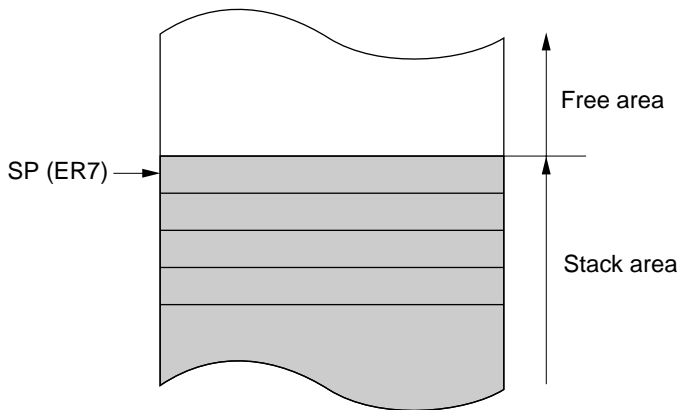


Figure 2.9 Stack

2.4.3 Control Registers

The control registers are the 24-bit program counter (PC), 8-bit extended control register (EXR), and 8-bit condition-code register (CCR).

(1) Program Counter (PC)

This 24-bit counter indicates the address of the next instruction the CPU will execute. The length of all CPU instructions is 2 bytes (one word), so the least significant PC bit is ignored. (When an instruction is fetched, the least significant PC bit is regarded as 0.)

(2) Extended Control Register (EXR)

An 8-bit register. In this LSI, this register does not affect operation.

Bit 7: Trace Bit (T)

This bit is reserved. In this LSI, this bit does not affect operation.

Bits 6 to 3: Reserved

These bits are reserved. They are always read as 1.

Bits 2 to 0: Interrupt Mask Bits (I2 to I0)

These bits are reserved. In this LSI, these bits do not affect operation.

(3) Condition: Code Register (CCR)

This 8-bit register contains internal CPU status information, including an interrupt mask bit (I) and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags.

Bit 7: Interrupt Mask Bit (I)

Masks interrupts other than NMI when set to 1. (NMI is accepted regardless of the I bit setting.) The I bit is set to 1 by hardware at the start of an exception-handling sequence. For details, see section 6, Interrupt Controller.

Bit 6: User Bit or Interrupt Mask Bit (UI)

Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions. This bit can also be used as an interrupt mask bit. For details, see section 6, Interrupt Controller.

Bit 5: Half-Carry Flag (H)

When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B, or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.

Bit 4: User Bit (U)

Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions.

Bit 3: Negative Flag (N)

Stores the value of the most significant bit (sign bit) of data.

Bit 2: Zero Flag (Z)

Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.

Bit 1: Overflow Flag (V)

Set to 1 when an arithmetic overflow occurs, and cleared to 0 otherwise.

Bit 0: Carry Flag (C)

Set to 1 when a carry occurs, and cleared to 0 otherwise. Used by:

- (a) Add instructions, to indicate a carry
- (b) Subtract instructions, to indicate a borrow
- (c) Shift and rotate instructions, to store the carry

The carry flag is also used as a bit accumulator by bit-manipulation instructions.

Some instructions leave some or all of the flag bits unchanged. For the action of each instruction on the flag bits, see section 29, Appendix A.1, List of Instructions.

Operations can be performed on the CCR bits by the LDC, STC, ANDC, ORC, and XORC instructions. The N, Z, V, and C flags are used as branching conditions for conditional branch (Bcc) instructions.

2.4.4 Initial Register Values

Reset exception handling loads the CPU's program counter (PC) from the vector table, clears the trace bit in EXR to 0, and sets the interrupt mask bits in CCR and EXR to 1. The other CCR bits and the general registers are not initialized. In particular, the stack pointer (ER7) is not initialized. The stack pointer should therefore be initialized by an MOV.L instruction executed immediately after a reset.

2.5 Data Formats

The CPU can process 1-bit, 4-bit (BCD), 8-bit (byte), 16-bit (word), and 32-bit (longword) data. Bit-manipulation instructions operate on 1-bit data by accessing bit n (n = 0, 1, 2, ..., 7) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

2.5.1 General Register Data Formats

Figure 2.10 shows the data formats in general registers.

| Data type | General Register | Data Format |
|----------------|------------------|--|
| 1-bit data | RnH | <div><div>70</div><div><div>7</div><div>6</div><div>5</div><div>4</div><div>3</div><div>2</div><div>1</div><div>0</div></div><div>Don't care</div></div> |
| 1-bit data | RnL | <div><div>70</div><div>Don't care</div><div><div>7</div><div>6</div><div>5</div><div>4</div><div>3</div><div>2</div><div>1</div><div>0</div></div></div> |
| 4-bit BCD data | RnH | <div><div>7430</div><div><div>Upper digit</div><div>Lower digit</div></div><div>Don't care</div></div> |
| 4-bit BCD data | RnL | <div><div>7430</div><div>Don't care</div><div><div>Upper digit</div><div>Lower digit</div></div></div> |
| Byte data | RnH | <div><div>70</div><div><div>MSB</div><div>LSB</div></div><div>Don't care</div></div> |
| Byte data | RnL | <div><div>70</div><div>Don't care</div><div><div>MSB</div><div>LSB</div></div></div> |

Figure 2.10 General Register Data Formats (1)

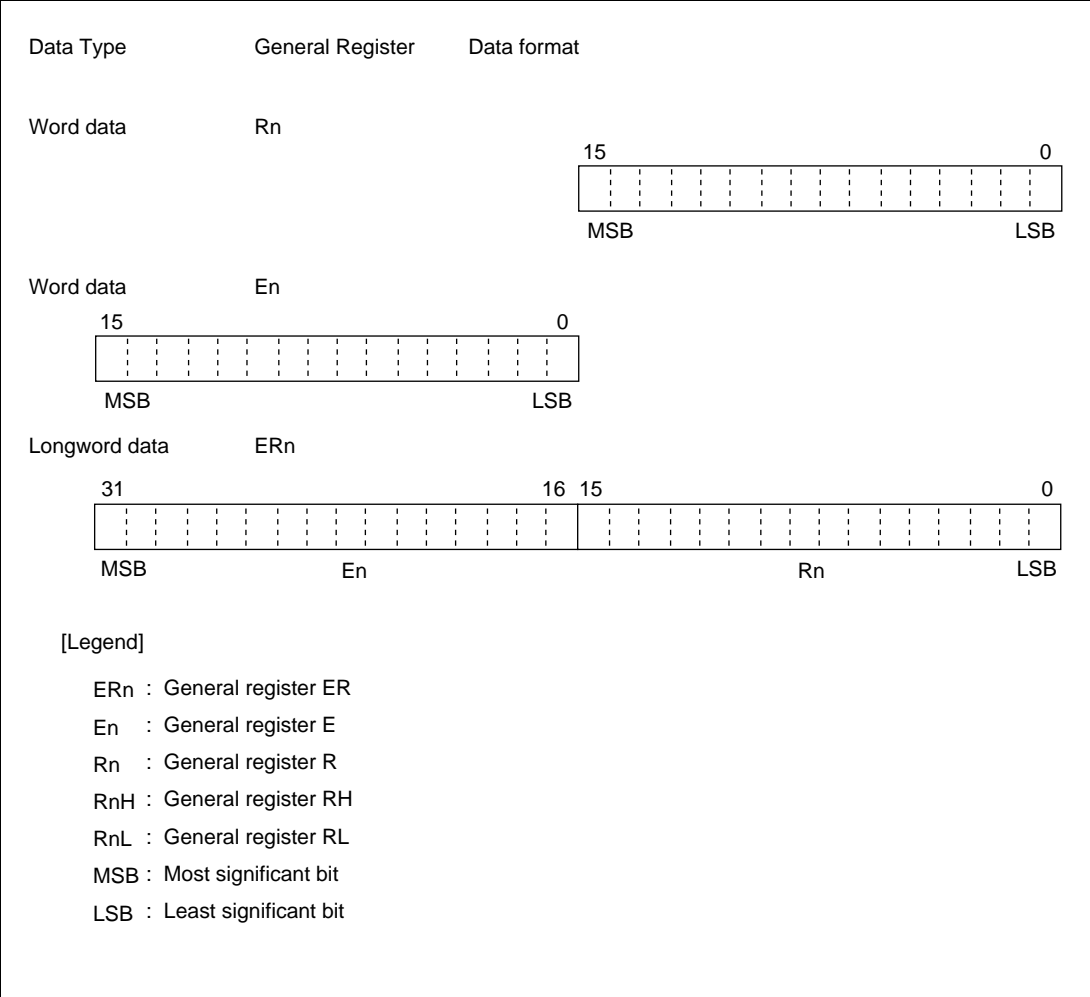


Figure 2.10 General Register Data Formats (2)

2.5.2 Memory Data Formats

Figure 2.11 shows the data formats in memory.

The CPU can access word data and longword data in memory, but word or longword data must begin at an even address. If an attempt is made to access word or longword data at an odd address, no address error occurs but the least significant bit of the address is regarded as 0, so the access starts at the preceding address. This also applies to instruction fetches.

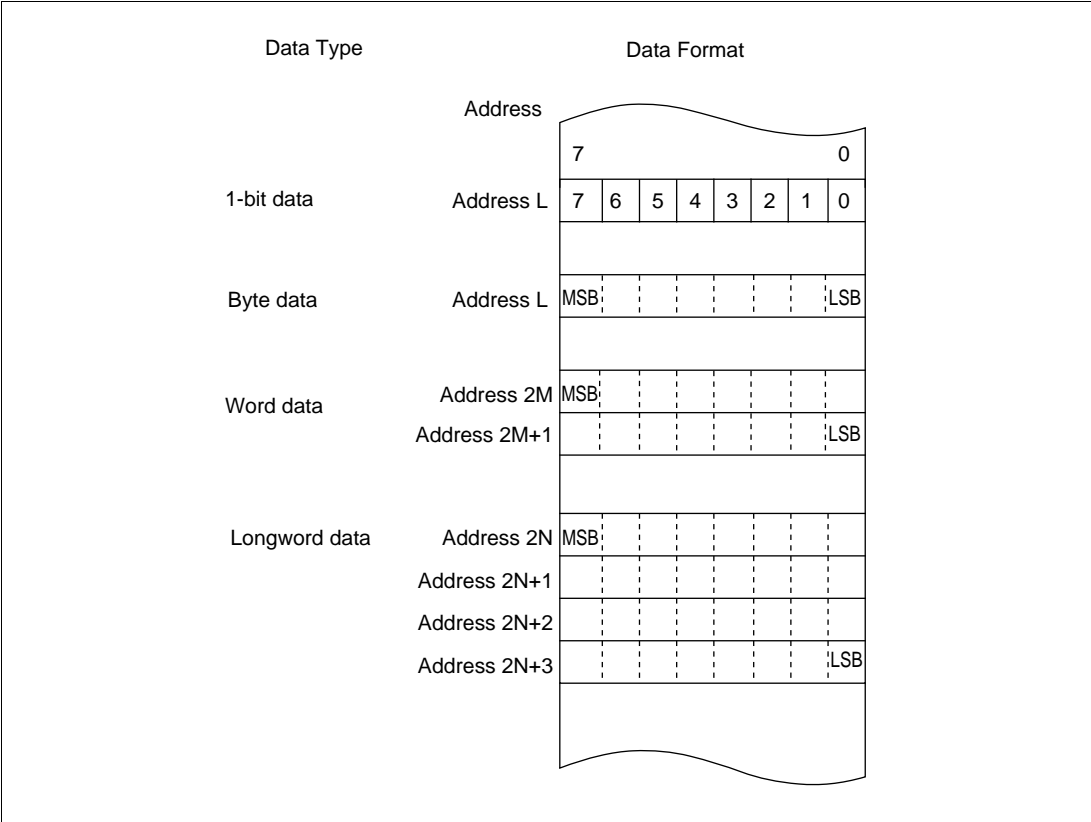


Figure 2.11 Memory Data Formats

When ER7 (SP) is used as an address register to access the stack, the operand size should be word size or longword size.

2.6 Instruction Set

2.6.1 Overview

The H8S/2000 CPU has 65 types of instructions. The instructions are classified by function in table 2.1.

Table 2.1 Instruction Classification

| Function | Instructions | Size | Types |
|---------------------|---|------|-------|
| Data transfer | MOV | BWL | 5 |
| | POP ^{*1} , PUSH ^{*1} | WL | |
| | LDM, STM | L | |
| | MOVFP ^{*3} , MOVTP ^{*3} | B | |
| Arithmetic | ADD, SUB, CMP, NEG | BWL | 19 |
| | ADDX, SUBX, DAA, DAS | B | |
| | INC, DEC | BWL | |
| | ADDS, SUBS | L | |
| | MULXU, DIVXU, MULXS, DIVXS | BW | |
| | EXTU, EXTS | WL | |
| | TAS ^{*4} | B | |
| Logic operations | AND, OR, XOR, NOT | BWL | 4 |
| Shift | SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR | BWL | 8 |
| Bit manipulation | RSET, BCLR, BNOT, BTST, BLD, BILD, BST, BIST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR | B | 14 |
| Branch | Bcc ^{*2} , JMP, BSR, JSR, RTS | — | 5 |
| System control | TRAPA, RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP | — | 9 |
| Block data transfer | EEPMOV | — | 1 |

Total: 65 types

Notes: B: byte size; W: word size; L: longword size.

1. POP.W Rn and PUSH.W Rn are identical to MOV.W @SP+, Rn and MOV.W Rn, @-SP.
POP.L ERn and PUSH.L ERn are identical to MOV.L @SP+, ERn and MOV.L ERn, @-SP.
2. Bcc is the general name for conditional branch instructions.
3. Not available in this LSI.
4. Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.

2.6.2 Instructions and Addressing Modes

Table 2.2 indicates the combinations of instructions and addressing modes that the H8S/2000 CPU can use.

Table 2.2 Combinations of Instructions and Addressing Modes

| Function | Instruction | Addressing Modes | | | | | | | | | | | | | |
|-----------------------|----------------------------|------------------|-----|------|--------------|--------------|--------------|-------|--------|--------|--------|------------|-------------|--------|----|
| | | #xx | Rn | @ERn | @(d:16, ERn) | @(d:32, ERn) | @-ERn/@ERn++ | @aa:8 | @aa:16 | @aa:24 | @aa:32 | @(d:8, PC) | @(d:16, PC) | @@aa:8 | |
| Data transfer | MOV | BWL | BWL | BWL | BWL | BWL | BWL | B | BWL | — | BWL | — | — | — | — |
| | POP, PUSH | — | — | — | — | — | — | — | — | — | — | — | — | — | WL |
| | LDM, STM | — | — | — | — | — | — | — | — | — | — | — | — | — | L |
| | MOVFP, MOVTP ^{*1} | — | — | — | — | — | — | — | B | — | — | — | — | — | — |
| Arithmetic operations | ADD, CMP | BWL | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | SUB | WL | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | ADDX, SUBX | B | B | — | — | — | — | — | — | — | — | — | — | — | — |
| | ADDS, SUBS | — | L | — | — | — | — | — | — | — | — | — | — | — | — |
| | INC, DEC | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | DAA, DAS | — | B | — | — | — | — | — | — | — | — | — | — | — | — |
| | MULXU, DIVXU | — | BW | — | — | — | — | — | — | — | — | — | — | — | — |
| | MULXS, DIVXS | — | BW | — | — | — | — | — | — | — | — | — | — | — | — |
| | NEG | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | EXTU, EXTS | — | WL | — | — | — | — | — | — | — | — | — | — | — | — |
| Logic operation | TAS ^{*2} | — | — | B | — | — | — | — | — | — | — | — | — | — | — |
| | AND, OR, XOR | BWL | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| Shift | NOT | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| | Shift | — | BWL | — | — | — | — | — | — | — | — | — | — | — | — |
| Bit manipulation | Bit manipulation | — | B | B | — | — | — | B | B | — | B | — | — | — | — |
| | Bcc, BSR | — | — | — | — | — | — | — | — | — | — | ○ | ○ | — | — |
| Branch | JMP, JSR | — | — | — | — | — | — | — | — | ○ | — | — | — | ○ | — |
| | RTS | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| System control | TRAPA | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| | RTE | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| | SLEEP | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| | LDC | B | B | W | W | W | W | — | W | — | W | — | — | — | — |
| | STC | — | B | W | W | W | W | — | W | — | W | — | — | — | — |
| | ANDC, ORC, XORC | B | — | — | — | — | — | — | — | — | — | — | — | — | — |
| | NOP | — | — | — | — | — | — | — | — | — | — | — | — | — | ○ |
| | Block data transfer | — | — | — | — | — | — | — | — | — | — | — | — | — | BW |

[Legend]

B: Byte

W: Word

L: Longword

Note: *1 Cannot be used in this LSI.

*2 Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.

2.6.3 Table of Instructions Classified by Function

Table 2.3 to 2.10 summarize the instructions in each functional category. The notation used in table 2.3 is defined below.

Operation Notation

| | |
|----------------|------------------------------------|
| Rd | General register (destination)* |
| Rs | General register (source)* |
| Rn | General register* |
| ERn | General register (32-bit register) |
| (EAd) | Destination operand |
| (EAs) | Source operand |
| EXR | Extended control register |
| CCR | Condition-code register |
| N | N (negative) flag in CCR |
| Z | Z (zero) flag in CCR |
| V | V (overflow) flag in CCR |
| C | C (carry) flag in CCR |
| PC | Program counter |
| SP | Stack pointer |
| #IMM | Immediate data |
| Disp | Displacement |
| + | Addition |
| – | Subtraction |
| × | Multiplication |
| ÷ | Division |
| ^ | Logical AND |
| ∨ | Logical OR |
| ⊕ | Logical exclusive OR |
| → | Move |
| ~ | NOT (logical complement) |
| :8/:16/:24/:32 | 8-, 16-, 24-, or 32-bit length |

Note: * General registers include 8-bit registers (R0H to R7H, R0L to R7L), 16-bit registers (R0 to R7, E0 to E7), and 32-bit registers (ER0 to ER7).

Table 2.3 Data Transfer Instructions

| Instruction | Size* | Function |
|--------------------|--------------|--|
| MOV | B/W/L | (EAs) → Rd, Rs → (EAd) Moves data between two general registers or between a general register and memory, or moves immediate data to a general register |
| MOVFPE | B | Cannot be used in this LSI |
| MOVTPE | B | Cannot be used in this LSI |
| POP | W/L | @SP+ → Rn Pops a general register from the stack POP.W Rn is identical to MOV.W @SP+, Rn POP.L ERn is identical to MOV.L @SP+, ERn |
| PUSH | W/L | Rn → @-SP Pushes a general register onto the stack PUSH.W Rn is identical to MOV.W Rn, @-SP PUSH.L ERn is identical to MOV.L ERn, @-SP |
| LDM | L | @SP+ → Rn (register list) Pops two or more general registers from the stack |
| STM | L | Rn (register list) → @-SP Pushes two or more general registers onto the stack |

Note: * Size refers to the operand size.

B: Byte

W: Word

L: Longword

Table 2.4 Arithmetic Instructions

| Instruction | Size ^{*1} | Function |
|--------------|--------------------|--|
| ADD SUB | B/W/L | $Rd \pm Rs \rightarrow Rd$, $Rd \pm \#IMM \rightarrow Rd$ Performs addition or subtraction on data in two general registers, or on immediate data and data in a general register. (Immediate byte data cannot be subtracted from byte data in a general register. Use the SUBX or ADD instruction) |
| ADDX SUBX | B | $Rd \pm Rs \pm C \rightarrow Rd$, $Rd \pm \#IMM \pm C \rightarrow Rd$ Performs addition or subtraction with carry on byte data in two general registers, or on immediate data and data in a general register |
| INC DEC | B/W/L | $Rd \pm 1 \rightarrow Rd$, $Rd \pm 2 \rightarrow Rd$ Increments or decrements a general register by 1 or 2. (Byte operands can be incremented or decremented by 1 only) |
| ADDS SUBS | B | $Rd \pm 1 \rightarrow Rd$, $Rd \pm 2 \rightarrow Rd$, $Rd \pm 4 \rightarrow Rd$ Adds or subtracts the value 1, 2, or 4 to or from data in a 32-bit register |
| DAA DAS | B/W | $Rd \text{ decimal adjust} \rightarrow Rd$ Decimal-adjusts an addition or subtraction result in a general register by referring to the CCR to produce 4-bit BCD data |
| MULXU | B/W | $Rd \times Rs \rightarrow Rd$ Performs unsigned multiplication on data in two general registers: either 8 bits \times 8 bits \rightarrow 16 bits or 16 bits \times 16 bits \rightarrow 32 bits |
| MULXS | B/W | $Rd \times Rs \rightarrow Rd$ Performs signed multiplication on data in two general registers: either 8 bits \times 8 bits \rightarrow 16 bits or 16 bits \times 16 bits \rightarrow 32 bits |
| DIVXU | B/W | $Rd \div Rs \rightarrow Rd$ Performs unsigned division on data in two general registers: either 16 bits \div 8 bits \times 8-bit quotient and 8-bit remainder or 32 bits \div 16 bits \times 16-bit quotient and 16-bit remainder |

| Instruction | Size ^{*1} | Function |
|-------------|--------------------|--|
| DIVXS | B/W | $Rd \div Rs \rightarrow Rd$ Performs signed division on data in two general registers: either 16 bits \div 8 bits \rightarrow 8-bit quotient and 8-bit remainder or 32 bits \div 16 bits \rightarrow 16-bit quotient and 16-bit remainder |
| CMP | B/W/L | $Rd - Rs, Rd - \#IMM$ Compares data in a general register with data in another general register or with immediate data, and sets CCR bits according to the result |
| NEG | B/W/L | $0 - Rd \rightarrow Rd$ Takes the two's complement (arithmetic complement) of data in a general register |
| EXTU | W/L | $Rd \text{ (zero extension)} \rightarrow Rd$ Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by padding with zeros on the left |
| EXTS | W/L | $Rd \text{ (sign extension)} \rightarrow Rd$ Extends the lower 8 bits of a 16-bit register to word size, or the lower 16 bits of a 32-bit register to longword size, by extending the sign bit |
| TAS | B | $@ERd - 0, 1 \rightarrow (<\text{bit } 7> \text{ of } @ERd)^{*2}$ Tests memory contents, and sets the most significant bit (bit 7) to 1 |

Note: ^{*1} Size refers to the operand size.

B: Byte

W: Word

L: Longword

^{*2} Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.

Table 2.5 Logic Instructions

| Instruction | Size* | Function |
|-------------|-------|---|
| AND | B/W/L | $Rd \wedge Rs \rightarrow Rd, Rd \wedge \#IMM \rightarrow Rd$ Performs a logical AND operation on a general register and another general register or immediate data |
| OR | B/W/L | $Rd \vee Rs \rightarrow Rd, Rd \vee \#IMM \rightarrow Rd$ Performs a logical OR operation on a general register and another general register or immediate data |
| XOR | B/W/L | $Rd \oplus Rs \rightarrow Rd, Rd \oplus \#IMM \rightarrow Rd$ Performs a logical exclusive OR operation on a general register and another general register or immediate data |
| NOT | B/W/L | $\sim Rd \rightarrow Rd$ Takes the one's complement (logical complement) of general register contents |

Note: * Size refers to the operand size.

B: Byte

W: Word

L: Longword

Table 2.6 Shift Instructions

| Instruction | Size* | Function |
|----------------|-------|---|
| SHAL SHAR | B/W/L | $Rd \text{ (shift)} \rightarrow Rd$ Performs an arithmetic shift on general register contents A 1-bit or 2-bit shift is possible |
| SHLL SHLR | B/W/L | $Rd \text{ (shift)} \rightarrow Rd$ Performs a logical shift on general register contents A 1-bit or 2-bit shift is possible |
| ROTL ROTR | B/W/L | $Rd \text{ (rotate)} \rightarrow Rd$ Rotates general register contents 1-bit or 2-bit rotation is possible |
| ROTXL ROTXR | B/W/L | $Rd \text{ (rotate)} \rightarrow Rd$ Rotates general register contents through the carry flag 1-bit or 2-bit rotation is possible |

Note: * Size refers to the operand size.

B: Byte

W: Word

L: Longword

Table 2.7 Bit Manipulation Instructions

| Instruction | Size* | Function |
|--------------------|--------------|--|
| BSET | B | $1 \rightarrow (\text{<bit-No.> of <EAd>})$ Sets a specified bit in a general register or memory operand to 1. The bit number is specified by 3-bit immediate data or the lower three bits of a general register |
| BCLR | B | $0 \rightarrow (\text{<bit-No.> of <EAd>})$ Clears a specified bit in a general register or memory operand to 0. The bit number is specified by 3-bit immediate data or the lower three bits of a general register |
| BNOT | B | $\sim (\text{<bit-No.> of <EAd>}) \rightarrow (\text{<bit-No.> of <EAd>})$ Inverts a specified bit in a general register or memory operand. The bit number is specified by 3-bit immediate data or the lower three bits of a general register |
| BTST | B | $\sim (\text{<bit-No.> of <EAd>}) \rightarrow Z$ Tests a specified bit in a general register or memory operand and sets or clears the Z flag accordingly. The bit number is specified by 3-bit immediate data or the lower three bits of a general register |
| BAND | B | $C \wedge (\text{<bit-No.> of <EAd>}) \rightarrow C$ ANDs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag |
| BIAND | B | $C \wedge [\sim (\text{<bit-No.> of <EAd>})] \rightarrow C$ ANDs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag The bit number is specified by 3-bit immediate data |
| BOR | B | $C \vee (\text{<bit-No.> of <EAd>}) \rightarrow C$ ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag |
| BIOR | B | $C \vee [\sim (\text{<bit-No.> of <EAd>})] \rightarrow C$ ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag The bit number is specified by 3-bit immediate data |

| Instruction | Size* | Function |
|---|-------|--|
| BOXR | B | $C \oplus (\text{<bit-No.> of <EAd>}) \rightarrow C$ Exclusive-ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag |
| BIXOR | B | $C \oplus [\sim (\text{<bit-No.> of <EAd>})] \rightarrow C$ Exclusive-ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag The bit number is specified by 3-bit immediate data |
| BLD | B | $(\text{<bit-No.> of <EAd>}) \rightarrow C$ Transfers a specified bit in a general register or memory operand to the carry flag |
| BILD | B | $\sim (\text{<bit-No.> of <EAd>}) \rightarrow C$ Transfers the inverse of a specified bit in a general register or memory operand to the carry flag The bit number is specified by 3-bit immediate data |
| BST | B | $C \rightarrow (\text{<bit-No.> of <EAd>})$ Transfers the carry flag value to a specified bit in a general register or memory operand |
| BIST | B | $\sim C \rightarrow (\text{<bit-No.> of <EAd>})$ Transfers the inverse of the carry flag value to a specified bit in a general register or memory operand The bit number is specified by 3-bit immediate data |
| Note: * Size refers to the operand size. B: Byte | | |

Table 2.8 Branch Instructions

| Instruction | Size* | Function | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|----------------------------|---|-----------------|--------------------|------------------|----------|---------------|--------|----------|---------------|-------|-----|------|-----------|-----|-------------|-----------|-----------|----------------------------|---------|-----------|-----------------|---------|-----|-----------|---------|-----|-------|---------|-----|----------------|---------|-----|--------------|---------|-----|------|---------|-----|-------|---------|-----|------------------|----------|-----|-----------|------------------|-----|--------------|---------------------------|-----|---------------|---------------------------|
| Bcc | — | Branches to a specified address if a specified condition is true The branching conditions are listed below | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <table> <tr> <th>Mnemonic</th><th>Description</th><th>Condition</th></tr> <tr> <td>BRA (BT)</td><td>Always (True)</td><td>Always</td></tr> <tr> <td>BRN (BF)</td><td>Never (False)</td><td>Never</td></tr> <tr> <td>BHI</td><td>HIgh</td><td>$CVZ = 0$</td></tr> <tr> <td>BLS</td><td>Low of Same</td><td>$CVZ = 1$</td></tr> <tr> <td>BCC (BHS)</td><td>Carry Clear (High or Same)</td><td>$C = 0$</td></tr> <tr> <td>BCS (BLO)</td><td>Carry Set (LOW)</td><td>$C = 1$</td></tr> <tr> <td>BNE</td><td>Not Equal</td><td>$Z = 0$</td></tr> <tr> <td>BEQ</td><td>Equal</td><td>$Z = 1$</td></tr> <tr> <td>BVC</td><td>oVerflow Clear</td><td>$V = 0$</td></tr> <tr> <td>BVS</td><td>oVerflow Set</td><td>$V = 1$</td></tr> <tr> <td>BPL</td><td>PLus</td><td>$N = 0$</td></tr> <tr> <td>BMI</td><td>MInus</td><td>$N = 1$</td></tr> <tr> <td>BGE</td><td>Greater or Equal</td><td>$NV = 0$</td></tr> <tr> <td>BLT</td><td>Less Than</td><td>$N \oplus V = 1$</td></tr> <tr> <td>BGT</td><td>Greater Than</td><td>$Z \vee (N \oplus V) = 0$</td></tr> <tr> <td>BLE</td><td>Less or Equal</td><td>$Z \vee (N \oplus V) = 1$</td></tr> </table> | | | Mnemonic | Description | Condition | BRA (BT) | Always (True) | Always | BRN (BF) | Never (False) | Never | BHI | HIgh | $CVZ = 0$ | BLS | Low of Same | $CVZ = 1$ | BCC (BHS) | Carry Clear (High or Same) | $C = 0$ | BCS (BLO) | Carry Set (LOW) | $C = 1$ | BNE | Not Equal | $Z = 0$ | BEQ | Equal | $Z = 1$ | BVC | oVerflow Clear | $V = 0$ | BVS | oVerflow Set | $V = 1$ | BPL | PLus | $N = 0$ | BMI | MInus | $N = 1$ | BGE | Greater or Equal | $NV = 0$ | BLT | Less Than | $N \oplus V = 1$ | BGT | Greater Than | $Z \vee (N \oplus V) = 0$ | BLE | Less or Equal | $Z \vee (N \oplus V) = 1$ |
| Mnemonic | Description | Condition | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BRA (BT) | Always (True) | Always | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BRN (BF) | Never (False) | Never | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BHI | HIgh | $CVZ = 0$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BLS | Low of Same | $CVZ = 1$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BCC (BHS) | Carry Clear (High or Same) | $C = 0$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BCS (BLO) | Carry Set (LOW) | $C = 1$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BNE | Not Equal | $Z = 0$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BEQ | Equal | $Z = 1$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BVC | oVerflow Clear | $V = 0$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BVS | oVerflow Set | $V = 1$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BPL | PLus | $N = 0$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BMI | MInus | $N = 1$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BGE | Greater or Equal | $NV = 0$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BLT | Less Than | $N \oplus V = 1$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BGT | Greater Than | $Z \vee (N \oplus V) = 0$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BLE | Less or Equal | $Z \vee (N \oplus V) = 1$ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| JMP | — | Branches unconditionally to a specified address | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BSR | — | Branches to a subroutine at a specified address | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| JSR | — | Branches to a subroutine at a specified address | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| RTS | — | Returns from a subroutine | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Table 2.9 System Control Instructions

| Instruction | Size* | Function |
|-------------|-------|---|
| TRAPA | — | Starts trap-instruction exception handling |
| RTE | — | Returns from an exception-handling routine |
| SLEEP | — | Causes a transition to a power-down state |
| LDC | B/W | (EAs) → CCR, (EAs) → EXR Moves contents of a general register or memory or immediate data to CCR or EXR. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid |
| STC | B/W | CCR → (EAd), EXR → (EAd) Transfers CCR or EXR contents to a general register or memory. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid |
| ANDC | B | $CCR \wedge \#IMM \rightarrow CCR$, $EXR \wedge \#IMM \rightarrow EXR$ Logically ANDs the CCR or EXR contents with immediate data |
| ORC | B | $CCR \vee \#IMM \rightarrow CCR$, $EXR \vee \#IMM \rightarrow EXR$ Logically ORs the CCR or EXR contents with immediate data |
| XORC | B | $CCR \oplus \#IMM \rightarrow CCR$, $EXR \oplus \#IMM \rightarrow EXR$ Logically exclusive-ORs the CCR or EXR contents with immediate data |
| NOP | — | $PC + 2 \rightarrow PC$ Only increments the program counter |

Note: * Size refers to the operand size.

B: Byte

W: Word

Table 2.10 Block Data Transfer Instructions

| Instruction | Size* | Function |
|--------------------|--------------|---|
| EEPMOV.B | — | if R4L \neq 0 then Repeat @ER5+ \rightarrow @er6+ R4L-1 \rightarrow R4L Until R4L = 0 else next; |
| EEPMOV.W | — | if R4 \neq 0 then Repeat @ER5+ \rightarrow @er6+ R4-1 \rightarrow R4 Until R4 = 0 else next; Transfers a data block according to parameters set in general registers R4L or R4, ER5, and ER6 R4L or R4: size of block (bytes) ER5: starting source address ER6: starting destination address Execution of the next instruction begins as soon as the transfer is completed |

2.6.4 Basic Instruction Formats

The CPU instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op field), a register field (r field), an effective address extension (EA field), and a condition field (cc).

Figure 2.12 shows examples of instruction formats.

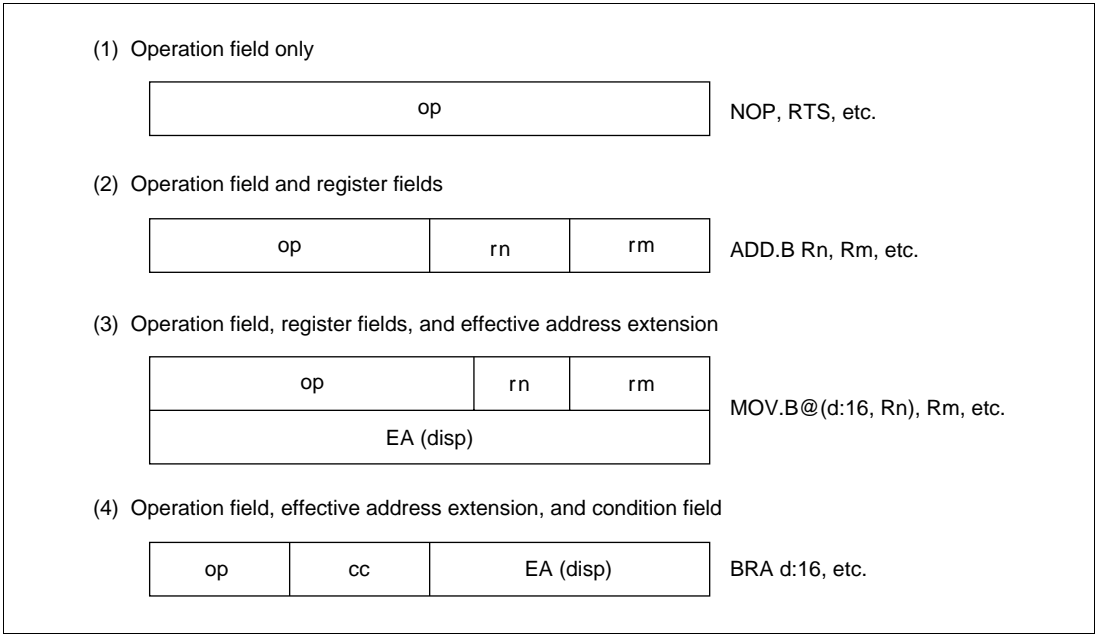


Figure 2.12 Instruction Formats (Examples)

- (1) Operation Field
- Indicates the function of the instruction, the addressing mode, and the operation to be carried out on the operand. The operation field always includes the first four bits of the instruction. Some instructions have two operation fields.
- (2) Register Field
- Specifies a general register. Address registers are specified by 3 bits, data registers by 3 bits or 4 bits. Some instructions have two register fields. Some have no register field.
- (3) Effective Address Extension
- Eight, 16, or 32 bits specifying immediate data, an absolute address, or a displacement.
- (4) Condition Field
- Specifies the branching condition of Bcc instructions.

2.6.5 Notes on Use of Bit-Manipulation Instructions

The BSET, BCLR, BNOT, BST, and BIST instructions read a byte of data, carry out bit manipulation, then write back the byte of data. Caution is therefore required when using these instructions on a register containing write-only bits, or a port.

The BCLR instruction can be used to clear internal I/O register flags to 0. In this case, the relevant flag need not be read beforehand if it is clear that it has been set to 1 in an interrupt handling routine, etc.

2.7 Addressing Modes and Effective Address Calculation

2.7.1 Addressing Mode

The CPU supports the eight addressing modes listed in table 2.11. Each instruction uses a subset of these addressing modes. Arithmetic and logic instructions can use the register direct and immediate modes. Data transfer instructions can use all addressing modes except program-counter relative and memory indirect. Bit-manipulation instructions use register direct, register indirect, or absolute addressing mode to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.

Table 2.11 Addressing Modes

| No. | Addressing Mode | Symbol |
|-----|---|-----------------------------|
| 1 | Register direct | Rn |
| 2 | Register indirect | @ERn |
| 3 | Register indirect with displacement | @(d:16,ERn)/@(d:32,ERn) |
| 4 | Register indirect with post-increment Register indirect with pre-decrement | @ERn+ @-ERn |
| 5 | Absolute address | @aa:8/#@aa:16/@aa:24/@aa:32 |
| 6 | Immediate | #xx:8/#xx:16/#xx:32 |
| 7 | Program-counter relative | @(d:8,PC)/@(d:16,PC) |
| 8 | Memory indirect | @ @aa:8 |

(1) Register Direct–Rn

The register field of the instruction code specifies an 8-, 16-, or 32-bit general register containing the operand. R0H to R7H and R0L to R7L can be specified as 8-bit registers. R0 to R7 and E0 to E7 can be specified as 16-bit registers. ER0 to ER7 can be specified as 32-bit registers.

(2) Register Indirect–@Ern

The register field of the instruction code specifies an address register (ERn) which contains the address of the operand in memory. If the address is a program instruction address, the lower 24 bits are valid and the upper 8 bits are all assumed to be 0 (H'00).

(3) Register Indirect with Displacement–@(d:16, ERn) or @(d:32, ERn)

A 16-bit or 32-bit displacement contained in the instruction is added to an address register (ERn) specified by the register field of the instruction, and the sum gives the address of a memory operand. A 16-bit displacement is sign-extended when added.

(4) Register Indirect with Post-Increment or Pre-Decrement—@ERn+ or @-ERn

(a) Register indirect with post-increment—@ERn+

The register field of the instruction code specifies an address register (ERn) which contains the address of a memory operand. After the operand is accessed, 1, 2, or 4 is added to the address register contents and the sum is stored in the address register. The value added is 1 for byte access, 2 for word access, or 4 for longword access. For word or longword access, the register value should be even.

(b) Register indirect with pre-decrement—@-ERn

The value 1, 2, or 4 is subtracted from an address register (ERn) specified by the register field in the instruction code, and the result becomes the address of a memory operand. The result is also stored in the address register. The value subtracted is 1 for byte access, 2 for word access, or 4 for longword access. For word or longword access, the register value should be even.

(5) Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32

The instruction code contains the absolute address of a memory operand. The absolute address may be 8 bits long (@aa:8), 16 bits long (@aa:16), 24 bits long (@aa:24), or 32 bits long (@aa:32).

To access data, the absolute address should be 8 bits (@aa:8), 16 bits (@aa:16), or 32 bits (@aa:32) long. For an 8-bit absolute address, the upper 24 bits are all assumed to be 1 (H'FFFF). For a 16-bit absolute address the upper 16 bits are a sign extension. A 32-bit absolute address can access the entire address space.

A 24-bit absolute address (@aa:24) indicates the address of a program instruction. The upper 8 bits are all assumed to be 0 (H'00).

Table 2.12 indicates the accessible absolute address ranges.

Table 2.12 Absolute Address Access Ranges

| Absolute Address | | Normal Mode | Advanced Mode |
|-----------------------------|---------------------|------------------|--|
| Data address | 8 bits (@aa:8) | H'FF00 to H'FFFF | H'FFFF00 to H'FFFFFF |
| | 16 bits (@aa:16) | H'0000 to H'FFFF | H'000000 to H'007FFF, H'FF8000 to H'FFFFFF |
| | 32 bits (@aa:32) | | H'000000 to H'FFFFFF |
| Program instruction address | 24 bits (@aa:24) | | |

(6) Immediate—#xx:8, #xx:16, or #xx:32

The instruction contains 8-bit (#xx:8), 16-bit (#xx:16), or 32-bit (#xx:32) immediate data as an operand.

The ADDS, SUBS, INC, and DEC instructions contain immediate data implicitly. Some bit manipulation instructions contain 3-bit immediate data in the instruction code, specifying a bit number. The TRAPA instruction contains 2-bit immediate data in its instruction code, specifying a vector address.

(7) Program-Counter Relative—@(d:8, PC) or @(d:16, PC)

This mode is used in the Bcc and BSR instructions. An 8-bit or 16-bit displacement contained in the instruction is sign-extended and added to the 24-bit PC contents to generate a branch address. Only the lower 24 bits of this branch address are valid; the upper 8 bits are all assumed to be 0 (H'00). The PC value to which the displacement is added is the address of the first byte of the next instruction, so the possible branching range is -126 to +128 bytes (-63 to +64 words) or -32766 to +32768 bytes (-16383 to +16384 words) from the branch instruction. The resulting value should be an even number.

(8) Memory Indirect—@@aa:8

This mode can be used by the JMP and JSR instructions. The instruction code contains an 8-bit absolute address specifying a memory operand. This memory operand contains a branch address. The upper bits of the absolute address are all assumed to be 0, so the address range is 0 to 255 (H'0000 to H'00FF in normal mode, H'000000 to H'0000FF in advanced mode). In normal mode the memory operand is a word operand and the branch address is 16 bits long. In advanced mode the memory operand is a longword operand, the first byte of which is assumed to be all 0 (H'00).

Note that the first part of the address range is also the exception vector area. For further details, see section 5, Exception Handling.

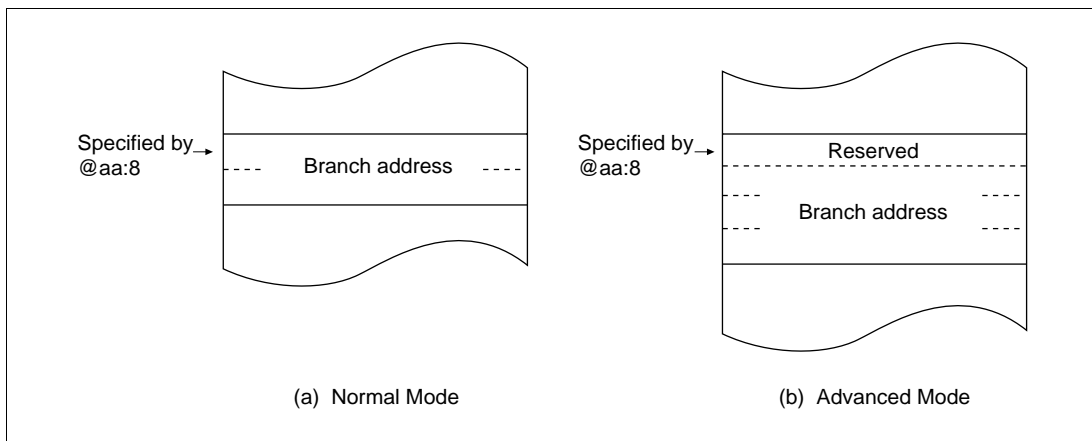


Figure 2.13 Branch Address Specification in Memory Indirect Mode

If an odd address is specified in word or longword memory access, or as a branch address, the least significant bit is regarded as 0, causing data to be accessed or an instruction code to be fetched at the address preceding the specified address. (For further information, see section 2.5.2, Memory Data Formats.)

2.7.2 Effective Address Calculation

Table 2.13 indicates how effective addresses are calculated in each addressing mode.

In normal mode the upper 8 bits of the effective address are ignored in order to generate a 16-bit address.

Table 2.13 Effective Address Calculation

| No. | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) | | | | |
|-----|--|-------------------------------|------------------------|----|------|--------------------------------------|--|
| 1 | Register direct (Rn) <table><tr><td>op</td><td>rm</td><td>rn</td></tr></table> | op | rm | rn | | Operand is general register contents | |
| op | rm | rn | | | | | |
| 2 | Register indirect (@ERn) <table><tr><td>op</td><td>r</td><td></td></tr></table> | op | r | | | | |
| op | r | | | | | | |
| 3 | Register indirect with displacement @(d:16, ERn) or @(d:32, ERn) <table><tr><td>op</td><td>r</td><td></td><td>disp</td></tr></table> | op | r | | disp | | |
| op | r | | disp | | | | |
| 4 | Register indirect with post-increment or pre-decrement <ul style="list-style-type: none">Register indirect with post-increment @ERn+ <table><tr><td>op</td><td>r</td><td></td></tr></table> Register indirect with pre-decrement @-ERn <table><tr><td>op</td><td>r</td><td></td></tr></table> | op | r | | op | r | |
| op | r | | | | | | |
| op | r | | | | | | |

| No. | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) | | | | | | | | | | | | | | | | | | | | | | | | | |
|----------------|---|-------------------------------|---------------------------|---|--|----|----|-------------|----|----|----|------------|--|----------------|------|--|----|----|----|---|--|--|------------|--|--|--|--|--|
| 5 | Absolute address | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | @aa:8 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | <table border="1"><tr><td>op</td><td>abs</td></tr></table> | op | abs | | <table border="1"><tr><td>31</td><td>24</td><td>23</td><td>8</td><td>7</td><td>0</td></tr><tr><td colspan="2">Don't care</td><td colspan="2">H'FFFF</td><td colspan="2"></td></tr></table> | 31 | 24 | 23 | 8 | 7 | 0 | Don't care | | H'FFFF | | | | | | | | | | | | | | |
| op | abs | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 24 | 23 | 8 | 7 | 0 | | | | | | | | | | | | | | | | | | | | | | | |
| Don't care | | H'FFFF | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | @aa:16 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | <table border="1"><tr><td>op</td><td>abs</td></tr></table> | op | abs | | <table border="1"><tr><td>31</td><td>24</td><td>23</td><td>16</td><td>15</td><td>0</td></tr><tr><td colspan="2">Don't care</td><td colspan="2">Sign extension</td><td colspan="2"></td></tr></table> | 31 | 24 | 23 | 16 | 15 | 0 | Don't care | | Sign extension | | | | | | | | | | | | | | |
| op | abs | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 24 | 23 | 16 | 15 | 0 | | | | | | | | | | | | | | | | | | | | | | | |
| Don't care | | Sign extension | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | @aa:24 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | <table border="1"><tr><td>op</td><td>abs</td></tr></table> | op | abs | | <table border="1"><tr><td>31</td><td>24</td><td>23</td><td colspan="3">0</td></tr><tr><td colspan="2">Don't care</td><td colspan="4"></td></tr></table> | 31 | 24 | 23 | 0 | | | Don't care | | | | | | | | | | | | | | | | |
| op | abs | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 24 | 23 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Don't care | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | @aa:32 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | <table border="1"><tr><td>op</td></tr><tr><td>abs</td></tr></table> | op | abs | | <table border="1"><tr><td>31</td><td>24</td><td>23</td><td colspan="3">0</td></tr><tr><td colspan="2">Don't care</td><td colspan="4"></td></tr></table> | 31 | 24 | 23 | 0 | | | Don't care | | | | | | | | | | | | | | | | |
| op | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| abs | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 24 | 23 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Don't care | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | Immediate #xx:8/#xx:16/#xx:32 | | Operand is immediate data | | | | | | | | | | | | | | | | | | | | | | | | | |
| | <table border="1"><tr><td>op</td><td>IMM</td></tr></table> | op | IMM | | | | | | | | | | | | | | | | | | | | | | | | | |
| op | IMM | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | Program-counter relative | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | @(d:8, PC)/@(d:16, PC) | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | <table border="1"><tr><td>op</td><td>disp</td></tr></table> | op | disp | <table border="1"><tr><td>23</td><td colspan="2">0</td></tr><tr><td colspan="3">PC contents</td></tr></table> <table border="1"><tr><td>23</td><td colspan="2">0</td></tr><tr><td>Sign extension</td><td colspan="2">disp</td></tr></table> <table border="1"><tr><td>31</td><td>24</td><td>23</td><td colspan="3">0</td></tr><tr><td colspan="2">Don't care</td><td colspan="4"></td></tr></table> | 23 | 0 | | PC contents | | | 23 | 0 | | Sign extension | disp | | 31 | 24 | 23 | 0 | | | Don't care | | | | | |
| op | disp | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 23 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| PC contents | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 23 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Sign extension | disp | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 24 | 23 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | |
| Don't care | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| No. | Addressing Mode and Instruction Format | Effective Address Calculation | Effective Address (EA) |
|-----|---|-------------------------------|------------------------|
| 8 | Memory indirect @@aa:8 <ul style="list-style-type: none"> Normal mode* | | |
| | <ul style="list-style-type: none"> Advanced mode | | |

Note: * Not available in this LSI.

2.8 Processing States

2.8.1 Overview

The CPU has four main processing states: the reset state, exception-handling state, program execution state, and power-down state. Figure 2.14 shows a diagram of the processing states. Figure 2.15 indicates the state transitions.

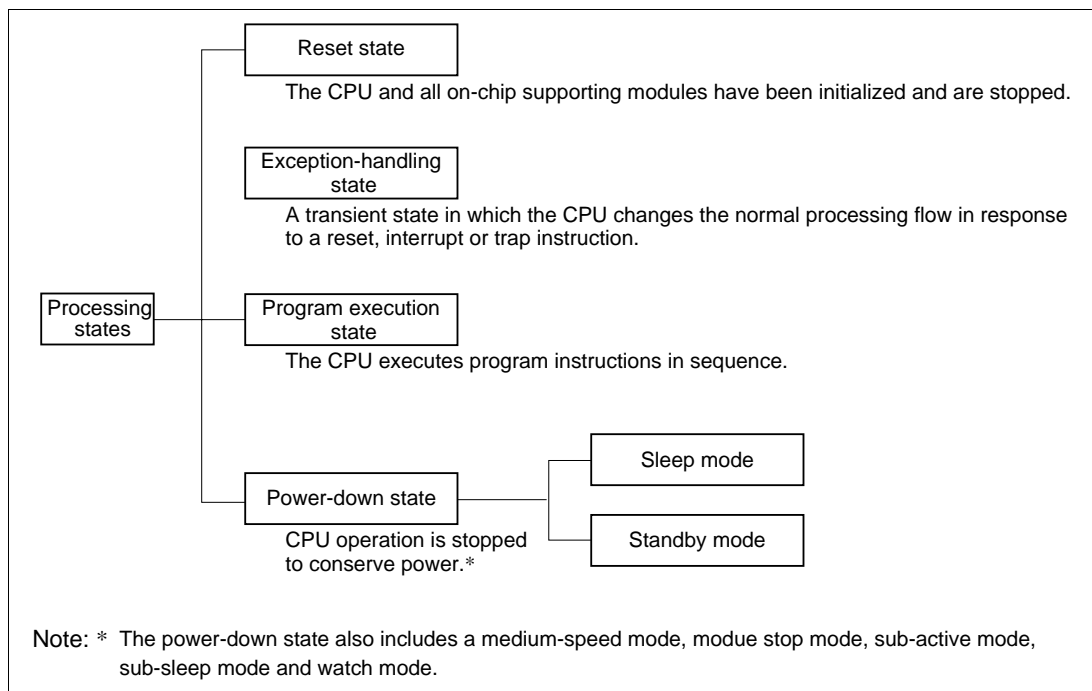


Figure 2.14 Processing States

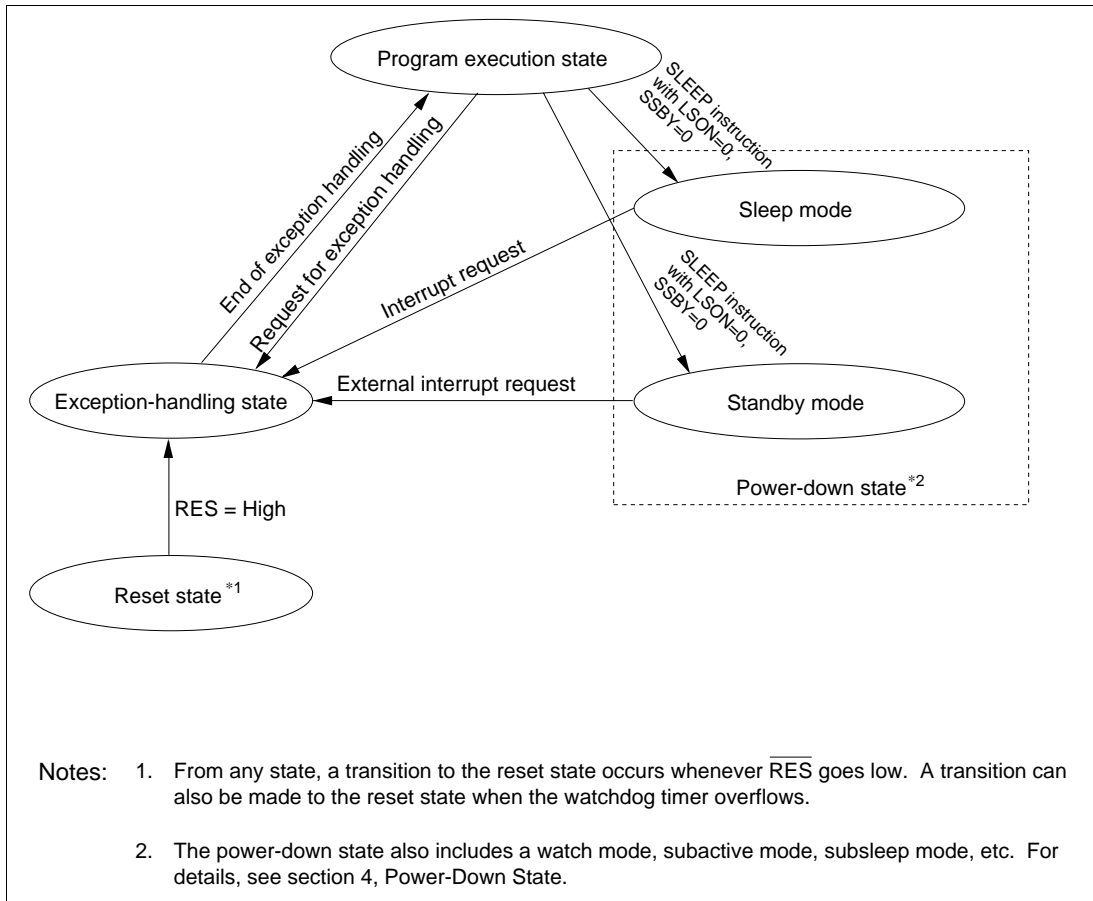


Figure 2.15 State Transitions

2.8.2 Reset State

When the $\overline{\text{RES}}$ input goes low all current processing stops and the CPU enters the reset state. All interrupts are disabled in the reset state. Reset exception handling starts when the $\overline{\text{RES}}$ signal changes from low to high.

The reset state can also be entered by a watchdog timer overflow. For details, see section 17, Watchdog Timer.

2.8.3 Exception-Handling State

The exception-handling state is a transient state that occurs when the CPU alters the normal processing flow due to a reset, interrupt, or trap instruction. The CPU fetches a start address (vector) from the exception vector table and branches to that address.

(1) Types of Exception Handling and Their Priority

Exception handling is performed for resets, interrupts, and trap instructions. Table 2.14 indicates the types of exception handling and their priority. Trap instruction exception handling is always accepted in the program execution state.

Exception handling and the stack structure depend on the interrupt control mode set in SYSCR.

Table 2.14 Exception Handling Types and Priority

| Priority | Type of Exception | Detection Timing | Start of Exception Handling |
|--|-------------------|--|---|
| <div><div>High</div><div>↑</div><div>Low</div></div> | Reset | Synchronized with clock | Exception handling starts immediately after a low-to-high transition at the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows |
| | Interrupt | End of instruction execution or end of exception-handling sequence ^{*1} | When an interrupt is requested, exception handling starts at the end of the current instruction or current exception-handling sequence |
| | Trap instruction | When TRAPA instruction is executed | Exception handling starts when a trap (TRAPA) instruction is executed ^{*2} |

- Notes: 1. Interrupts are not detected at the end of the ANDC, ORC, XORC, and LDC instructions, or immediately after reset exception handling.
2. Trap instruction exception handling is always accepted in the program execution state.

(2) Reset Exception Handling

After the $\overline{\text{RES}}$ pin has gone low and the reset state has been entered, when $\overline{\text{RES}}$ goes high again, reset exception handling starts. When reset exception handling starts the CPU fetches a start address (vector) from the exception vector table and starts program execution from that address. All interrupts, including NMI, are disabled during reset exception handling and after it ends.

(3) Interrupt Exception Handling and Trap Instruction Exception Handling

When interrupt or trap-instruction exception handling begins, the CPU references the stack pointer (ER7) and pushes the program counter and other control registers onto the stack. Next, the CPU alters the settings of the interrupt mask bits in the control registers. Then the CPU fetches a start address (vector) from the exception vector table and program execution starts from that start address.

Figure 2.16 shows the stack after exception handling ends.

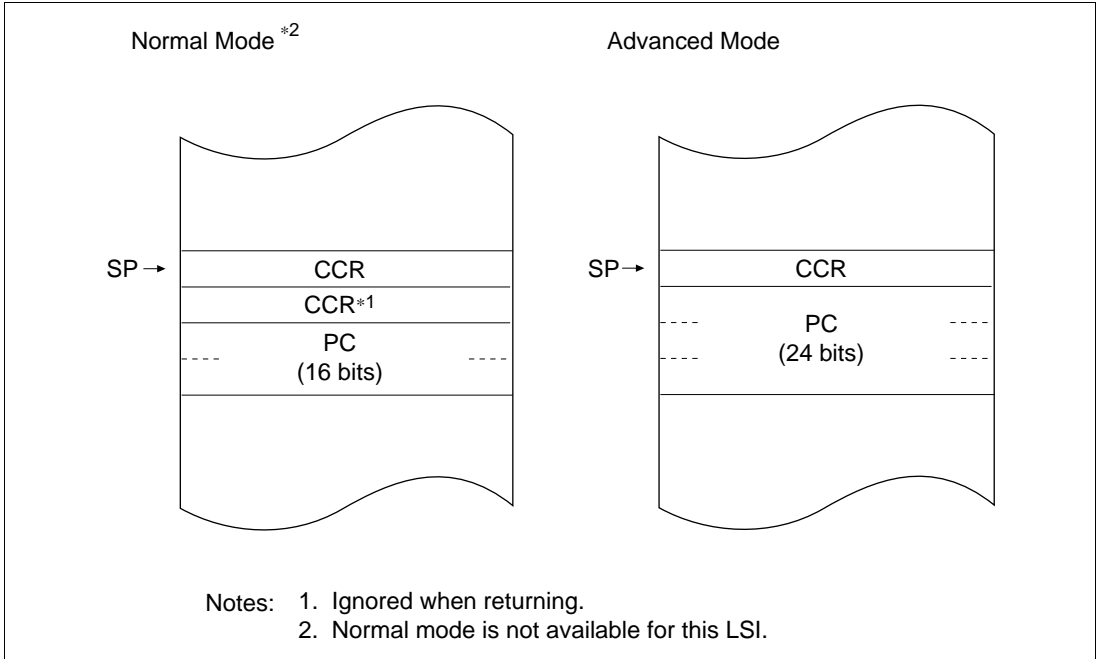


Figure 2.16 Stack Structure after Exception Handling (Examples)

2.8.4 Program Execution State

In this state the CPU executes program instructions in sequence.

2.8.5 Power-Down State

The power-down state includes both modes in which the CPU stops operating and modes in which the CPU does not stop. There are five modes in which the CPU stops operating: sleep mode, standby mode, subsleep mode, and watch mode. There are also three other power-down modes: medium-speed mode, module stop mode, and subactive mode. In medium-speed mode, the CPU operates on a medium-speed clock. Module stop mode permits halting of the operation of individual modules, other than the CPU. Subactive mode, subsleep mode, and watch mode are power-down modes that use subclock input. For details, see section 4, Power-Down State.

(1) Sleep Mode

A transition to sleep mode is made if the SLEEP instruction is executed while the software standby bit (SSBY) in the standby control register (SBYCR) and the LSON bit in the low-power control register (LPWRCR) are both cleared to 0. In sleep mode, CPU operations stop immediately after execution of the SLEEP instruction. The contents of CPU registers are retained.

(2) Standby Mode

A transition to standby mode is made if the SLEEP instruction is executed while the SSBY bit in SBYCR is set to 1 and the LSON bit in LPWRCR and the TMA3 bit in the TMA (timer A) are both cleared to 0. In standby mode, the CPU and clock halt and all MCU operations stop. As long as a specified voltage is supplied, the contents of CPU registers and on-chip RAM are retained.

2.9 Basic Timing

2.9.1 Overview

The CPU is driven by a system clock, denoted by the symbol ϕ . The period from one rising edge of ϕ to the next is referred to as a “state.” The memory cycle or bus cycle consists of one or two states. Different methods are used to access on-chip memory and on-chip supporting modules.

2.9.2 On-Chip Memory (ROM, RAM)

On-chip memory is accessed in one state. The data bus is 16 bits wide, permitting both byte and word transfer instruction. Figure 2.17 shows the on-chip memory access cycle.

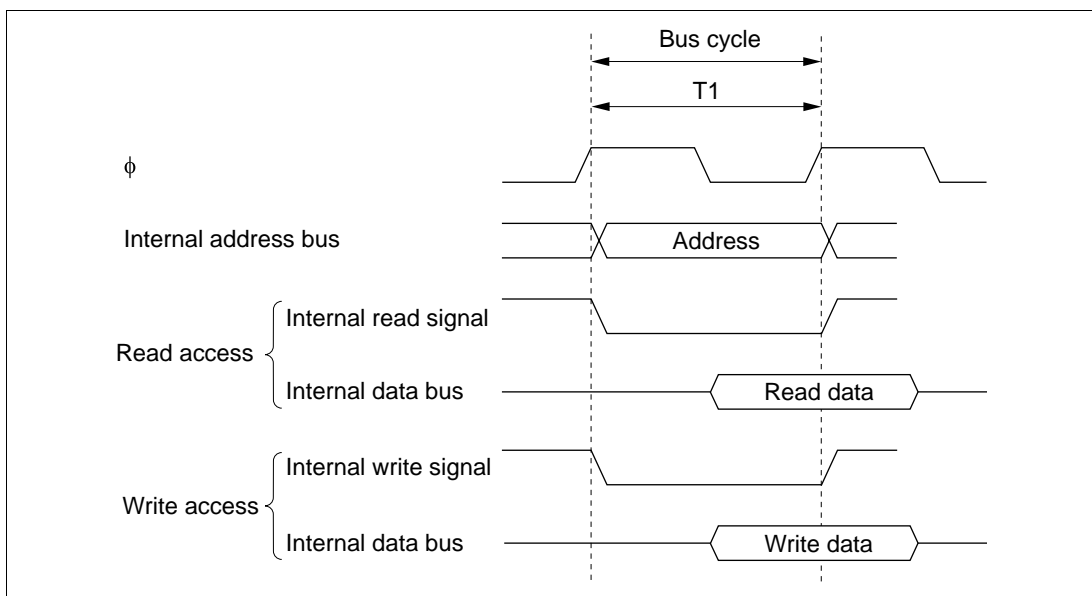


Figure 2.17 On-Chip Memory Access Cycle

2.9.3 On-Chip Supporting Module Access Timing

The on-chip supporting modules are accessed in two states. The data bus is either 8 bits or 16 bits wide, depending on the particular internal I/O register being accessed. Figure 2.18 shows the access timing for the on-chip supporting modules.

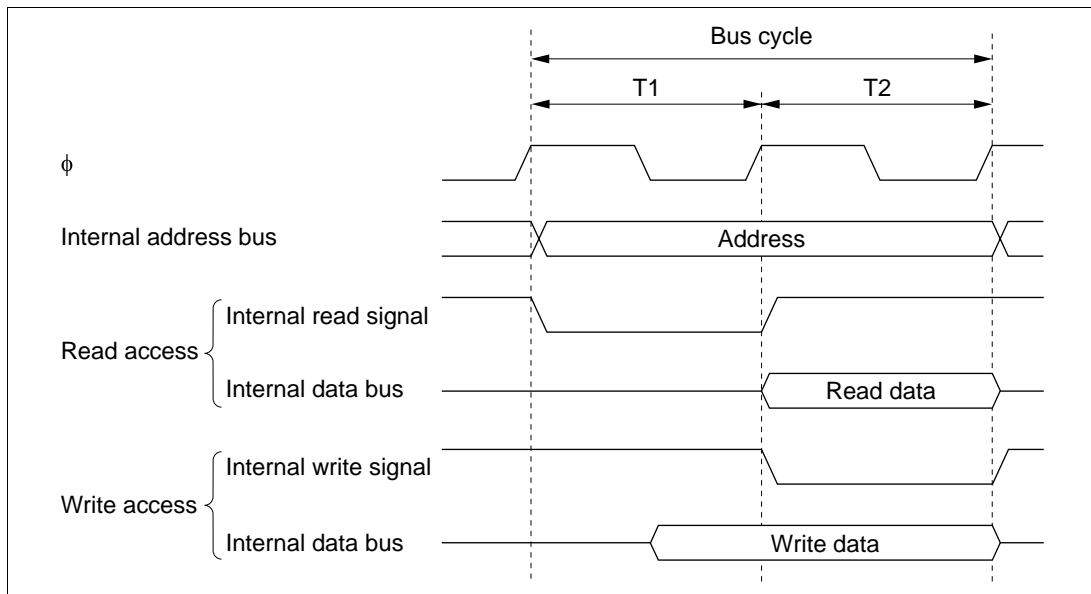


Figure 2.18 On-Chip Supporting Module Access Cycle

2.10 Usage Note

Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction. The TAS instruction is not generated by the Hitachi H8S or H8/300 series C/C++ compilers. If the TAS instruction is used as a user-defined intrinsic function, ensure that only register ER0, ER1, ER4, or ER5 is used.

Section 3 MCU Operating Modes

3.1 Overview

3.1.1 Operating Mode Selection

This LSI has one operating mode (mode 1). This mode is selected depending on settings of the mode pin (MD0).

Table 3.1 lists the MCU operating modes.

Table 3.1 MCU Operating Mode Selection

| MCU Operating Mode | MD0 | CPU Operating Mode | Description |
|--------------------|-----|--------------------|------------------|
| 0 | 0 | — | — |
| 1 | 1 | Advanced | Single-chip mode |

The CPU's architecture allows for 4 Gbytes of address space, but this LSI actually accesses a maximum of 16 Mbytes.

Mode 1 operation starts in single-chip mode after reset release.

This LSI can only be used in mode 1. This means that the mode pins must be set at mode 1. Do not changes the inputs at the mode pins during operation.

3.1.2 Register Configuration

This LSI has a mode control register (MDCR) that indicates the inputs at the mode pin (MD0) and a system control register (SYSCR) and that controls the operation of this LSI. Table 3.2 summarizes these registers.

Table 3.2 MCU Registers

| Name | Abbreviation | R/W | Initial Value | Address* |
|-------------------------|--------------|-----|---------------|----------|
| Mode control register | MDCR | R/W | Undetermined | H'FFE9 |
| System control register | SYSCR | R/W | H'09 | H'FFE8 |

Note: * Lower 16 bits of the address.

3.2 Register Descriptions

3.2.1 Mode Control Register (MDCR)

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | MDS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | —* |
| R/W : | — | — | — | — | — | — | — | R |

Note: * Determined by MD0 pin

MDCR is an 8-bit read-only register monitors the current operating mode of this LSI.

Bit 7 to 1: Reserved.

These bits cannot be modified and are always set at 0.

Bit 0: Mode Select 0 (MDS0)

This bit indicates the value which reflects the input levels at mode pin (MD0) (the current operating mode). Bit MDS0 corresponds to MD0 pin. It is read-only bit and cannot be written to. The mode pin (MD0) input levels are latched into these bits when MDCR is read.

3.2.2 System Control Register (SYSCR)

| | | | | | | | | |
|-----------------|---|---|-------|-------|------|--------|--------|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | INTM1 | INTM0 | XRST | NMIEG1 | NMIEG0 | — |
| Initial value : | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| R/W : | — | — | R | R/W | R | R/W | R/W | — |

Bits 7 and 6: Reserved.

Bits 5 and 4: Interrupt control modes 1 and 0 (INTM1, INTM0)

These bits are for selecting the interrupt control mode of the interrupt controller. For details of the interrupt control modes, see section 6.4, Interrupt Operation.

| Bit 5 INTM1 | Bit 4 INTM0 | Interrupt Control Mode | Description |
|----------------|----------------|---------------------------|---|
| 0 | 0 | 0 | Interrupt is controlled by bit I (Initial value) |
| | 1 | 1 | Interrupt is controlled by bits I and UI, and ICR |
| 1 | 0 | 2 | Cannot be used in this LSI |
| | 1 | 3 | Cannot be used in this LSI |

Bit 3: External Reset (XRST)

Indicates the reset source. When the watchdog timer is used, a reset can be generated by watchdog timer overflow as well as by external reset input. XRST is a read-only bit. It is set to 1 by an external reset and cleared to 0 by watchdog timer overflow.

Bit 3

| XRST | Description |
|------|---|
| 0 | A reset is generated by watchdog timer overflow |
| 1 | A reset is generated by an external reset (Initial value) |

Bits 2 and 1: NMI edge select 1 and 0 (NMIEG1, 0)

Select the input edge for NMI interrupt.

| Bit 2 NIMIEG1 | Bit 1 NIMIEG0 | Description |
|------------------|------------------|--|
| 0 | 0 | An interrupt request occurs at falling edge of NMI input (Initial value) |
| | 1 | An interrupt request occurs at rising edge of NMI input |
| 1 | * | An interrupt request occurs at rising or falling edge of NMI input |

Note: * Don't care

Bit 0: Reserved.

3.3 Operating Mode Descriptions

3.3.1 Mode 1

The CPU can access a 16 Mbyte address space in advanced mode.

3.4 Address Map

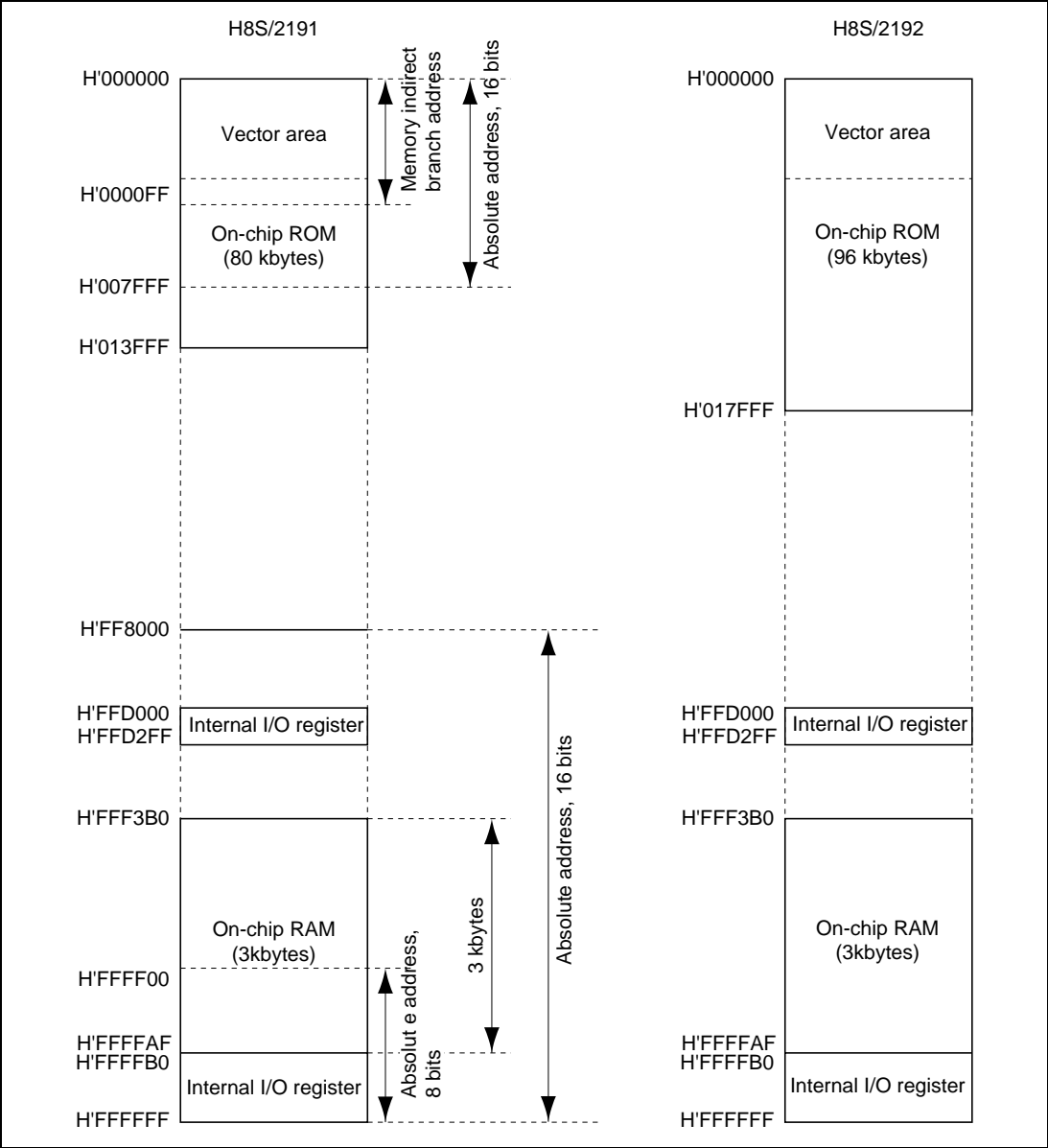


Figure 3.1 Address Map (1)

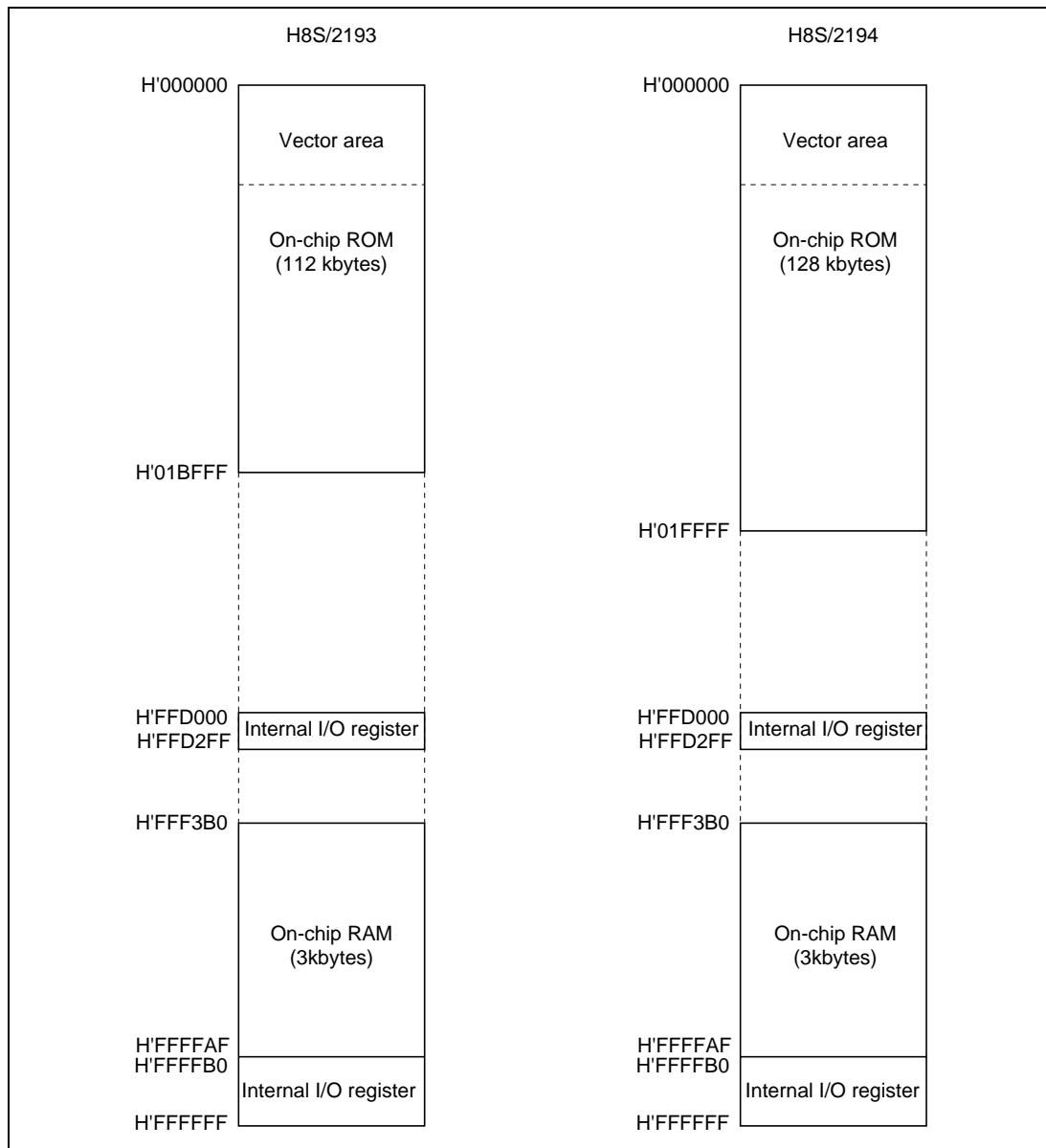


Figure 3.2 Address Map (2)

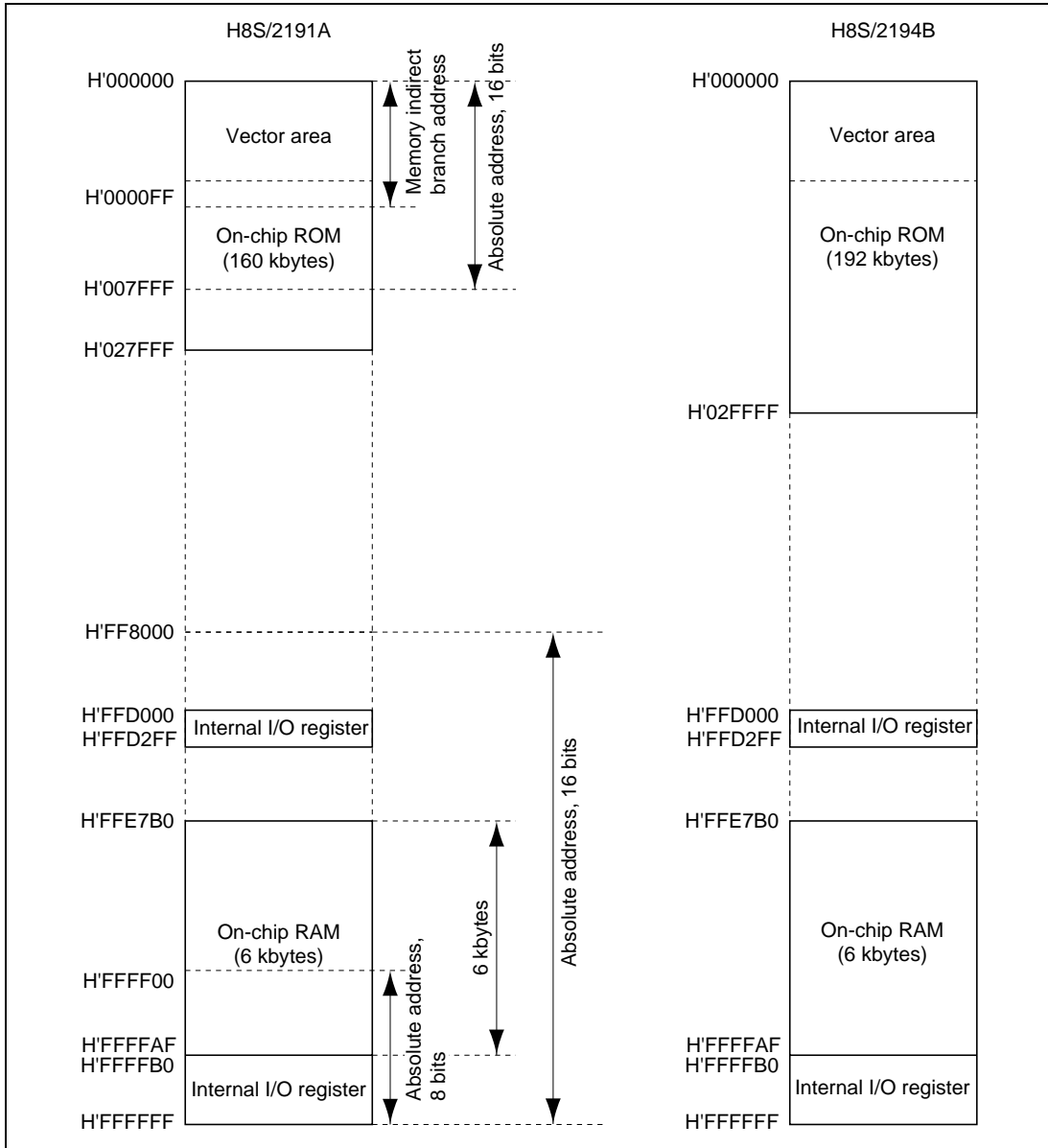


Figure 3.3 Address Map (3)

H8S/2194C

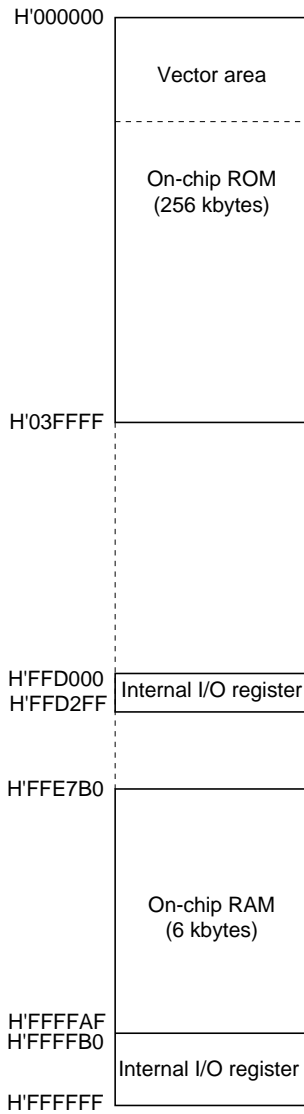


Figure 3.4 Address Map (4)

Section 4 Power-Down State

4.1 Overview

In addition to the normal program execution state, this LSI has a power-down state in which operation of the CPU and oscillator is halted and power dissipation is reduced. Low-power operation can be achieved by individually controlling the CPU, on-chip supporting modules, and so on.

This LSI operating modes are as follows:

1. High-speed mode
2. Medium-speed mode
3. Subactive mode
4. Sleep mode
5. Subsleep mode
6. Watch mode
7. Module stop mode
8. Standby mode

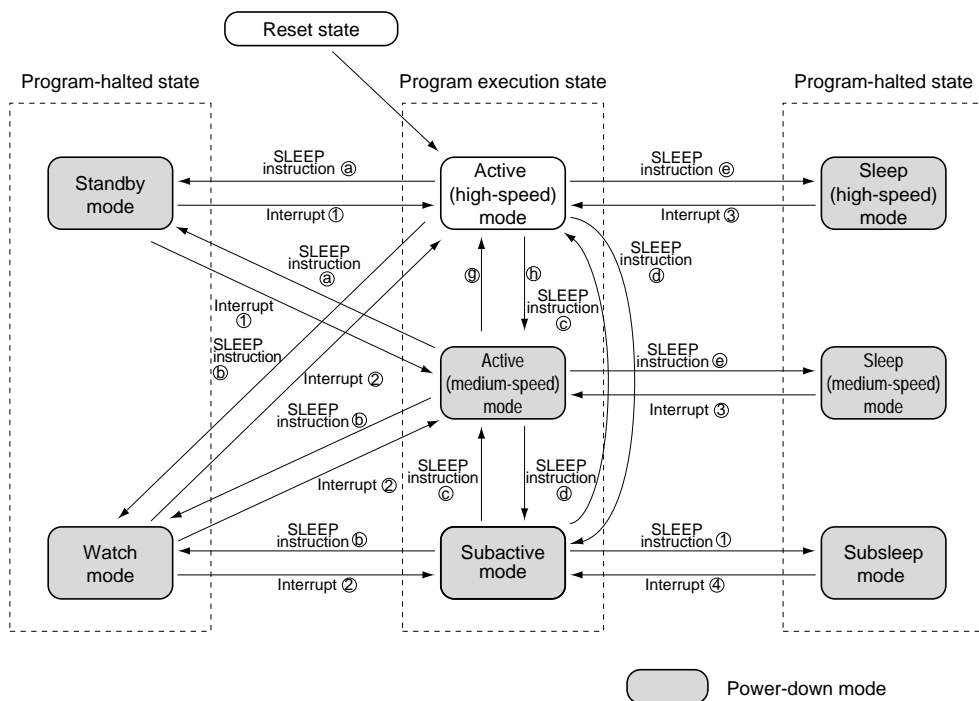
Of these, 2 to 8 are power-down modes. Certain combinations of these modes can be set. After a reset, the MCU is in high-speed mode.

Table 4.1 shows the internal chip states in each mode, and table 4.2 shows the conditions for transition to the various modes. Figure 4.1 shows a mode transition diagram.

Table 4.1 Internal Chip States in Each Mode

| Function | | High-Speed | Medium-Speed | Sleep | Module Stop | Watch | Subactive | Subsleep | Standby |
|-------------------------------------|----------------|-------------|--------------|----------------|--------------------------------|--------------------|--------------------|--------------------|-------------------|
| System clock | | Functioning | Functioning | Functioning | Functioning | Halted | Halted | Halted | Halted |
| Subclock pulse generator | | Functioning | Functioning | Functioning | Functioning | Functioning | Functioning | Functioning | Functioning |
| CPU operation | Instructions | Functioning | Medium-speed | Halted | Functioning | Halted | Subclock operation | Halted | Halted |
| | Registers | | | Retained | | Retained | | Retained | Retained |
| External interrupts | NIMI | Functioning | Functioning | Functioning | Functioning | Functioning | Functioning | Functioning | Functioning |
| | IRQ0 | | | | | | | | |
| | IRQ1 | | | | | | | | |
| | IRQ2 | | | | | Halted | Halted | Functioning | Halted |
| | IRQ3 | | | | | | | | |
| | IRQ4 | | | | | | | | |
| | IRQ5 | | | | | | | | |
| On-chip supporting module operation | Timer A | Functioning | Functioning | Functioning | Functioning /halted (retained) | Subclock operation | Subclock operation | Subclock operation | Halted (retained) |
| | Timer B | Functioning | Functioning | Functioning | Functioning /halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) |
| | Timer J | | | | | | | | |
| | Timer L | | | | | | | | |
| | Timer R | | | | | | | | |
| | Timer X1 | | | | Functioning /halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) |
| | Watchdog timer | Functioning | Functioning | Functioning | Functioning | Halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) |
| | PSU | Functioning | Functioning | Functioning | Functioning | Subclock operation | Subclock operation | Subclock operation | Halted |
| | IIC | | | | Functioning /halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) |
| | SCI1 | | | | | | | | |
| | SCI2 | | | | Functioning /halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) | Halted (retained) |
| | 14-bit PWM | | | | | | | | |
| | 8-bit PWM | | | | | | | | |
| | A/D | | | | Functioning /halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) |
| | I/O | Functioning | Functioning | Retained | Functioning | Halted | Functioning | Retained | Halted |
| | 12-bit PWM | Functioning | Functioning | Halted (reset) | Functioning /halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) | Halted (reset) |
| | Servo | | | | | | | | |

- Notes:
1. "Halted (retained)" means that internal register values are retained. The internal state is "operation suspended."
 2. "Halted (reset)" means that internal register values and internal states are initialized.
 3. In module stop mode, only modules for which a stop setting has been made are halted (reset or retained).
 4. In the power-down mode, the analog section of the servo circuits are not turned off, therefore Vcc (Servo) current does not go low. When power-down is needed, externally shut down the analog system power.



Conditions for mode transition (1)

| Flag | LSON | SSBY | TMA3 | DTON |
|------|----------------------------------|------|------|------|
| a | 0 | 1 | 0 | * |
| b | * | 1 | 1 | 0 |
| c | 0 | 1 | 1 | 1 |
| d | 1 | 1 | 1 | 1 |
| e | 0 | 0 | * | * |
| f | 1 | 0 | 1 | * |
| g | SCK1 to 0 = 0 | | | |
| h | SCK1 to 0 ≠ 0 (either 1 bit = 0) | | | |

Note: * Don't care

Conditions for mode transition (2)

| | Interruption factor |
|---|---|
| 1 | NMI, IRQ0 to 1 |
| 2 | NMI, IRQ0 to 1, Timer A interruption |
| 3 | All interruption (excluding servo system) |
| 4 | NMI, IRQ0 to 5, Timer A interruption |

Note: When a transition is made between modes by means of an interrupt, transition cannot be made on interrupt source generation alone. Ensure that interrupt handling is performed after accepting the interrupt request

Figure 4.1 Mode Transitions

Table 4.2 Power-Down Mode Transition Conditions

| State before Transition | Control Bit States at Time of Transition | | | | State after Transition by SLEEP Instruction | State after Return by Interrupt |
|--------------------------|--|------|------|------|---|--|
| | SSBY | TMA3 | LSON | DTON | | |
| High-speed /medium-speed | 0 | * | 0 | * | Sleep | High-speed /medium-speed ^{*1} |
| | 0 | * | 1 | * | — | — |
| | 1 | 0 | 0 | * | Standby | High-speed /medium-speed ^{*1} |
| | 1 | 0 | 1 | * | — | — |
| | 1 | 1 | 0 | 0 | Watch | High-speed /medium-speed ^{*1} |
| | 1 | 1 | 1 | 0 | Watch | Subactive |
| | 1 | 1 | 0 | 1 | — | — |
| | 1 | 1 | 1 | 1 | Subactive | — |
| | 1 | 1 | 1 | 1 | Subactive | — |
| Subactive | 0 | 0 | * | * | — | — |
| | 0 | 1 | 0 | * | — | — |
| | 0 | 1 | 1 | * | Subsleep | Subactive |
| | 1 | 0 | * | * | — | — |
| | 1 | 1 | 0 | 0 | Watch | High-speed /medium-speed ^{*2} |
| | 1 | 1 | 1 | 0 | Watch | Subactive |
| | 1 | 1 | 0 | 1 | High-speed /medium-speed ^{*2} | — |
| | 1 | 1 | 1 | 1 | — | — |
| | 1 | 1 | 1 | 1 | — | — |

Notes: * Don't care

—: Do not set.

1. Returns to the state before transition.

2. Mode varies depending on the state of SCK1 to SCK0.

4.1.1 Register Configuration

The power-down state is controlled by the SBYCR, LPWRCR, TMA (Timer A), and MSTPCR registers. Table 4.3 summarizes these registers.

Table 4.3 Power-Down State Registers

| Name | Abbreviation | R/W | Initial Value | Address* |
|------------------------------|--------------|-----|---------------|----------|
| Standby control register | SBYCR | R/W | H'00 | H'FFEA |
| Low-power control register | LPWRCR | R/W | H'00 | H'FFEB |
| Module stop control register | MSTPCRH | R/W | H'FF | H'FFEC |
| | MSTPCRL | R/W | H'FF | H'FFED |
| Timer mode register | TMA | R/W | H'30 | H'FFBA |

Note: * Lower 16 bits of the address.

4.2 Register Descriptions

4.2.1 Standby Control Register (SBYCR)

| | | | | | | | | |
|-----------------|------|------|------|------|---|---|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | SSBY | STS2 | STS1 | STS0 | — | — | SCK1 | SCK0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | — | — | R/W | R/W |

SBYCR is an 8-bit readable/writable register that performs power-down mode control.
SBYCR is initialized to H'00 by a reset.

Bit 7: Software Standby (SSBY)

Determines the operating mode, in combination with other control bits, when a power-down mode transition is made by executing a SLEEP instruction. The SSBY setting is not changed by a mode transition due to an interrupt, etc.

Bit 7

| SSBY | Description |
|------|--|
| 0 | Transition to sleep mode after execution of SLEEP instruction in high-speed mode or medium-speed mode Transition to subsleep mode after execution of SLEEP instruction in subactive mode (Initial value) |
| 1 | Transition to standby mode, subactive mode, or watch mode after execution of SLEEP instruction in high-speed mode or medium-speed mode Transition to watch mode or high-speed mode after execution of SLEEP instruction in subactive mode |

Bits 6 to 4: Standby Timer Select 2 to 0 (STS2 to STS0)

These bits select the time the MCU waits for the clock to stabilize when standby mode, watch mode, or subactive mode is cleared and a transition is made to high-speed mode or medium-speed mode by means of a specific interrupt or instruction. With crystal oscillation, see table 4.5 and make a selection according to the operating frequency so that the standby time is at least 10 ms (the oscillation settling time). With an external clock, any selection can be made.
(With FLASH ROM version, do not set the standby time to 16 states.)

| Bit 6 | Bit 5 | Bit 4 | |
|-------|-------|-------|--|
| STS2 | STS1 | STS0 | Description |
| 0 | 0 | 0 | Standby time = 8192 states |
| 0 | 0 | 1 | Standby time = 16384 states |
| 0 | 1 | 0 | Standby time = 32768 states |
| 0 | 1 | 1 | Standby time = 65536 states |
| 1 | 0 | 0 | Standby time = 131072 states |
| 1 | 0 | 1 | Standby time = 262144 states |
| 1 | 1 | * | Standby time = 16 states ^{*1} |

Notes: * Don't care

1. With FLASH ROM version, do not set the standby time to 16 states.
The standby time is 32 states when transited to medium-speed mode $\phi/32$ (SCK1=1, SCK0=0).

Bit 3, 2: Reserved.

These bits cannot be modified and are always read as 0.

Bits 1, 0: System Clock Select 1, 0 (SCK1, SCK0)

These bits select the CPU clock for the bus master in high-speed mode and medium-speed mode.

| Bit 1 | Bit 0 | |
|-------|-------|--|
| SCK1 | SCK0 | Description |
| 0 | 0 | Bus master is in high-speed mode (Initial value) |
| 0 | 1 | Medium-speed clock is $\phi/16$ |
| 1 | 0 | Medium-speed clock is $\phi/32$ |
| 1 | 1 | Medium-speed clock is $\phi/64$ |

4.2.2 Low-Power Control Register (LPWRCR)

| | | | | | | | | |
|-----------------|------|------|-------|---|---|---|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | DTON | LSON | NESEL | — | — | — | SA1 | SA0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | — | — | — | R/W | R/W |

LPWRCR is an 8-bit readable/writable register that performs power-down mode control. LPWRCR is initialized to H'00 by a reset.

Bit 7: Direct-Transfer On Flag (DTON)

Specifies whether a direct transition is made between high-speed mode, medium-speed mode, and subactive mode when making a power-down transition by executing a SLEEP instruction. The operating mode to which the transition is made after SLEEP instruction execution is determined by a combination of other control bits.

Bit 7

| DTON | Description |
|------|---|
| 0 | <ul style="list-style-type: none">When a SLEEP instruction is executed in high-speed mode or medium-speed mode, a transition is made to sleep mode, standby mode, or watch modeWhen a SLEEP instruction is executed in subactive mode, a transition is made to subsleep mode or watch mode (Initial value) |
| 1 | <ul style="list-style-type: none">When a SLEEP instruction is executed in high-speed mode or medium-speed mode, transition is made directly to subactive mode, or a transition is made to sleep mode or standby modeWhen a SLEEP instruction is executed in subactive mode, a transition is made directly to high-speed mode, or a transition is made to subsleep mode |

Bit 6: Low-Speed On Flag (LSON)

Determines the operating mode in combination with other control bits when making a power-down transition by executing a SLEEP instruction. Also controls whether a transition is made to high-speed mode or to subactive mode when watch mode is cleared.

Bit 6

| LSON | Description |
|------|---|
| 0 | <ul style="list-style-type: none"> When a SLEEP instruction is executed in high-speed mode or medium-speed mode, transition is made to sleep mode, standby mode, or watch mode When a SLEEP instruction is executed in subactive mode, a transition is made to watch mode, or directly to high-speed mode After watch mode is cleared, a transition is made to high-speed mode (Initial value) |
| 1 | <ul style="list-style-type: none"> When a SLEEP instruction is executed in high-speed mode a transition is made to watch mode, subactive mode, sleep mode or standby mode When a SLEEP instruction is executed in subactive mode, a transition is made to subsleep mode or watch mode After watch mode is cleared, a transition is made to subactive mode |

Bit 5: Noise Elimination Sampling Frequency Select (NESEL)

Selects the frequency at which the subclock (ϕ_w) generated by the subclock pulse generator is sampled with the clock (ϕ) generated by the system clock oscillator. When $\phi = 5$ MHz or higher, clear this bit to 0.

Bit 5

| NESEL | Description |
|-------|----------------------------------|
| 0 | Sampling at ϕ divided by 16 |
| 1 | Sampling at ϕ divided by 4 |

Bits 4 to 2: Reserved.

These bits cannot be modified and are always read as 0.

Bit 1, 0: Subactive mode clock select 1, 0 (SA1, SA0)

These bits select the CPU operating clock in the subactive mode. These bits cannot be modified in the subactive mode.

| Bit 1 | Bit 0 | Description |
|-------|-------|--|
| SA1 | SA0 | |
| 0 | 0 | Operating clock of CPU is $\phi_w/8$ (Initial value) |
| 0 | 1 | Operating clock of CPU is $\phi_w/4$ |
| 1 | * | Operating clock of CPU is $\phi_w/2$ |

Note: * Don't care

4.2.3 Timer Register A (TMA)

| | | | | | | | | |
|-----------------|--------|-------|-----|-----|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMAOV | TMAIE | — | — | TMA3 | TMA2 | TMA1 | TMA0 |
| Initial value : | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Only 0 can be written, to clear the flag.

The timer register A (TMA) controls timer A interrupts and selects input clock.

Only Bit 3 is explained here. For details of other bits, see section 12.2.1, Timer Mode Register A.

TMA is a readable/writable register which is initialized to H'30 by a reset.

Bit 3: Clock source, prescaler select (TMA3)

Selects Timer A clock source between PSS and PSW.

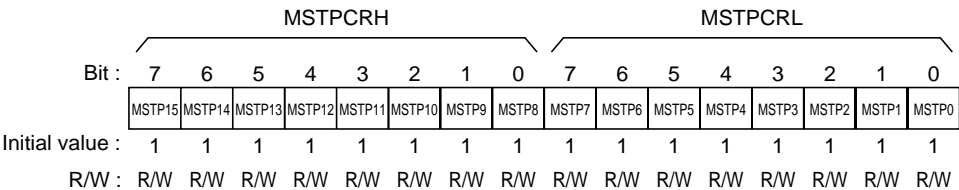
Also controls transition operation to the power-down mode. The operation mode to which the MCU is transited after SLEEP instruction execution is determined by the combination with other control bits than this bit.

For details, see the description of Clock Select 2 to 0 in section 12.2.1, Timer Mode Register A.

Bit 3

| TMA3 | Description |
|------|---|
| 0 | <ul style="list-style-type: none">• Timer A counts ϕ-based prescaler (PSM) divided clock pulses• When a SLEEP instruction is executed in high-speed mode or medium-speed mode, a transition is made to sleep mode or software standby mode (Initial value) |
| 1 | <ul style="list-style-type: none">• Timer A counts ϕ_w-based prescaler (PSM) divided clock pulses• When a SLEEP instruction is executed in high-speed mode or medium-speed mode, a transition is made to sleep mode, watch mode, or subactive mode• When a SLEEP instruction is executed in subactive mode, a transition is made to subsleep mode, watch mode, or high-speed mode |

4.2.4 Module Stop Control Register (MSTPCR)



MSTPCR comprises two 8-bit readable/writable registers that perform module stop mode control.

MSTPCR is initialized to H'FFFF by a reset.

MSTPCRH and MSTPCRL Bits 7 to 0: Module Stop (MSTP 15 to MSTP 0)

These bits specify module stop mode. See table 4.4 for the method of selecting on-chip supporting modules.

MSTPCRH, MSTPCRL
Bits 7 to 0

| MSTP 15 to MSTP 0 | Description |
|-------------------|---|
| 0 | Module stop mode is cleared |
| 1 | Module stop mode is set (Initial value) |

4.3 Medium-Speed Mode

When the SCK1 and SCK0 bits in SBYCR are set to 1 in high-speed mode, the operating mode changes to medium-speed mode at the end of the bus cycle. In medium-speed mode, the CPU operates on the operating clock ($\phi 16$, $\phi 32$ or $\phi 64$) specified by the SCK1 and SCK0 bits. The on-chip supporting modules other than the CPU always operate on the high-speed clock (ϕ).

In medium-speed mode, a bus access is executed in the specified number of states with respect to the bus master operating clock. For example, if $\phi 16$ is selected as the operating clock, on-chip memory is accessed in 16 states, and internal I/O registers in 32 states.

Medium-speed mode is cleared by clearing the both bits SCK1 and SCK0 to 0. A transition is made to high-speed mode and medium-speed mode is cleared at the end of the current bus cycle.

If a SLEEP instruction is executed when the SSBY bit in SBYCR and the LSON bit in LPWRCR are cleared to 0, a transition is made to sleep mode. When sleep mode is cleared by an interrupt, medium-speed mode is restored.

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, and the LSON bit in LPWRCR and the TMA3 bit in TMA (Timer A) are both cleared to 0, a transition is made to software standby mode. When standby mode is cleared by an external interrupt, medium-speed mode is restored.

When the $\overline{\text{RES}}$ pin is driven low, a transition is made to the reset state, and medium-speed mode is cleared. The same applies in the case of a reset caused by overflow of the watchdog timer.

Figure 4.2 shows the timing for transition to and clearance of medium-speed mode.

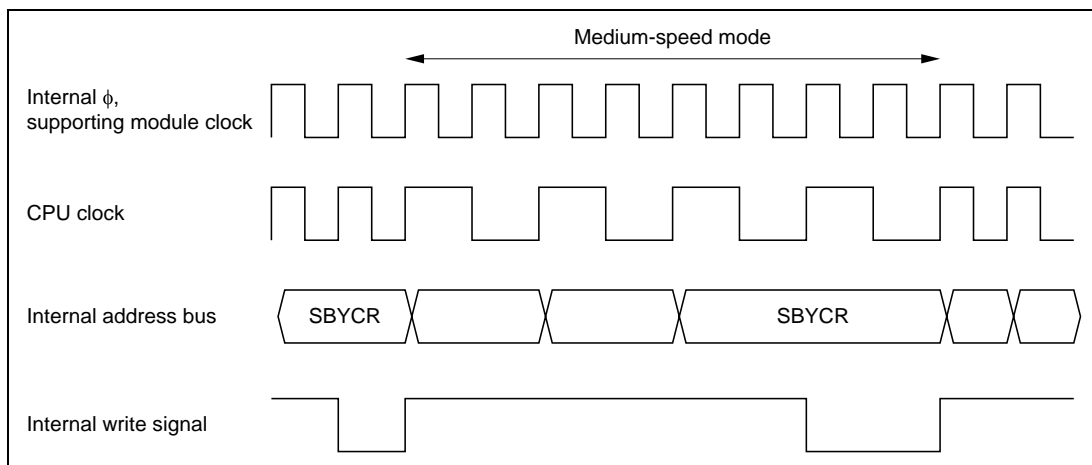


Figure 4.2 Medium-Speed Mode Transition and Clearance Timing

4.4 Sleep Mode

4.4.1 Sleep Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR and the LSON bit in LPWRCR are both cleared to 0, the CPU enters sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other supporting modules (excluding the servo circuit and 12-bit PWM) do not stop.

4.4.2 Clearing Sleep Mode

Sleep mode is cleared by any interrupt, or with the $\overline{\text{RES}}$ pin.

(1) Clearing with an Interrupt

When an interrupt request signal is input, sleep mode is cleared and interrupt exception handling is started. Sleep mode will not be cleared if interrupts are disabled, or if interrupts other than NMI have been masked by the CPU.

(2) Clearing with the $\overline{\text{RES}}$ Pin

When the $\overline{\text{RES}}$ pin is driven low, the reset state is entered. When the $\overline{\text{RES}}$ pin is driven high after the prescribed reset input period, the CPU begins reset exception handling.

4.5 Module Stop Mode

4.5.1 Module Stop Mode

Module stop mode can be set for individual on-chip supporting modules.

When the corresponding MSTP bit in MSTPCR is set to 1, module operation stops at the end of the bus cycle and a transition is made to module stop mode. The CPU continues operating independently.

Table 4.4 shows MSTP bits and the on-chip supporting modules.

When the corresponding MSTP bit is cleared to 0, module stop mode is cleared and the module starts operating again at the end of the bus cycle. In module stop mode, the internal states of modules other than the SCI1, A/D converter, Timer X1, and Servo circuit, are retained.

After reset release, all modules are in module stop mode.

When an on-chip supporting module is in module stop mode, read/write access to its registers is disabled.

Table 4.4 MSTP Bits and Corresponding On-Chip Supporting Modules

| Register | Bit | Module |
|----------|--------|---|
| MSTPCRH | MSTP15 | Timer A |
| | MSTP14 | Timer B |
| | MSTP13 | Timer J |
| | MSTP12 | Timer L |
| | MSTP11 | Timer R |
| | MSTP10 | Timer X1 |
| | MSTP9 | — |
| | MSTP8 | Serial communication interface 1 (SCI1) |
| MSTPCRL | MSTP7 | Serial communication interface 2 (SCI2) |
| | MSTP6 | I ² C bus interface (IIC) |
| | MSTP5 | 14-bit PWM |
| | MSTP4 | 8-bit PWM |
| | MSTP3 | — |
| | MSTP2 | A/D converter |
| | MSTP1 | Servo circuit, 12-bit PWM |
| | MSTP0 | — |

4.6 Standby Mode

4.6.1 Standby Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, the LSON bit in LPWRCR is cleared to 0, and the TMA3 bit in TMA (Timer A) is cleared to 0, standby mode is entered. In this mode, the CPU, on-chip supporting modules, and oscillator (except for subclock oscillator) all stop. However, contents of the CPU's internal registers and data in the built-in RAM as well as functions of the SCII, timer X1 and built-in peripheral circuits (except the servo circuit) are maintained in the current state. The I/O port, at this time, is caused to the high impedance state.

In this mode the oscillator stops, and therefore power dissipation is significantly reduced.

4.6.2 Clearing Standby Mode

Standby mode is cleared by an external interrupt (NMI pin, or pin $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ1}}$, or by means of the $\overline{\text{RES}}$ pin.

(1) Clearing with an Interrupt

When an NMI, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ1}}$ interrupt request signal is input, clock oscillation starts, and after the elapse of the time set in bits STS2 to STS0 in SYSCR, stable clocks are supplied to the entire chip, standby mode is cleared, and interrupt exception handling is started.

Standby mode cannot be cleared with an $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ1}}$ interrupt if the corresponding enable bit has been cleared to 0 or has been masked by the CPU.

(2) Clearing with the $\overline{\text{RES}}$ Pin

When the $\overline{\text{RES}}$ pin is driven low, clock oscillation is started. At the same time as clock oscillation starts, clocks are supplied to the entire chip. Note that the $\overline{\text{RES}}$ pin must be held low until clock oscillation stabilizes. When the $\overline{\text{RES}}$ pin goes high, the CPU begins reset exception handling.

4.6.3 Setting Oscillation Settling Time after Clearing Standby Mode

Bits STS2 to STS0 in SBYCR should be set as described below.

(1) Using a Crystal Oscillator

Set bits STS2 to STS0 so that the standby time is at least 10 ms (the oscillation settling time).

Table 4.5 shows the standby times for different operating frequencies and settings of bits STS2 to STS0.

Table 4.5 Oscillation Settling Time Settings

| STS2 | STS1 | STS0 | Standby Time | 10 MHz | 8 MHz | Unit |
|------|------|------|-------------------------|--------|-------|------|
| 0 | 0 | 0 | 8192 states | 0.8 | 1.0 | ms |
| | | 1 | 16384 states | 1.6 | 2.0 | |
| | 1 | 0 | 32768 states | 3.3 | 4.1 | |
| | | 1 | 65536 states | 6.6 | 8.2 | |
| 1 | 0 | 0 | 131072 states | 13.1 | 16.4 | |
| | | 1 | 262144 states | 26.2 | 32.8 | |
| | 1 | * | 16 states ^{*1} | 1.6 | 2.0 | |

: Recommended time setting

Note: * Don't care

(2) Using an External Clock

Any value can be set.

Note: 1. With Flash memory version, do not set the standby time to 16 states. The standby time is 32 states when transited to medium-speed mode $\phi/32$ (SCK1 = 1, SCK0 = 0).

4.7 Watch Mode

4.7.1 Watch Mode

If a SLEEP instruction is executed in high-speed mode, medium-speed mode or subactive mode when the SSBY in SBYCR is set to 1, the DTON bit in LPWRCR is cleared to 0, and the TMA3 bit in TMA (Timer A) is set to 1, the CPU makes a transition to watch mode.

In this mode, the CPU and all on-chip supporting modules except Timer A stop. As long as the prescribed voltage is supplied, the contents of some of the CPU's internal registers and on-chip RAM are retained, and I/O ports are placed in the high-impedance state.

4.7.2 Clearing Watch Mode

Watch mode is cleared by an interrupt (Timer A interrupt, NMI pin, or pin $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ1}}$), or by means of the $\overline{\text{RES}}$ pin.

(1) Clearing with an Interrupt

When an interrupt request signal is input, watch mode is cleared and a transition is made to high-speed mode or medium-speed mode if the LSON bit in LPWRCR is cleared to 0, or to subactive mode if the LSON bit is set to 1. When making a transition to medium-speed mode, after the elapse of the time set in bits STS2 to STS0 in SBYCR, stable clocks are supplied to the entire chip, and interrupt exception handling is started.

Watch mode cannot be cleared with an $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ1}}$ interrupt if the corresponding enable bit has been cleared to 0, or with an on-chip supporting module interrupt if acceptance of the relevant interrupt has been disabled by the interrupt enable register or masked by the CPU.

See section 4.6.3, Setting Oscillation Settling Time after Clearing Standby Mode, for the oscillation settling time setting when making a transition from watch mode to high-speed mode.

(2) Clearing with the $\overline{\text{RES}}$ Pin

See (2) Clearing with the $\overline{\text{RES}}$ Pin in section 4.6.2, Clearing Standby Mode.

4.8 Subsleep Mode

4.8.1 Subsleep Mode

If a SLEEP instruction is executed in subactive mode when the SSBY in SBYCR is cleared to 0, the LSON bit in LPWRCR is set to 1, and the TMA3 bit in TMA (Timer A) is set to 1, the CPU makes a transition to subsleep mode.

In this mode, the CPU and all on-chip supporting modules other than Timer A stop. As long as the prescribed voltage is supplied, the contents of the CPU, some of its on-chip registers and on-chip RAM are retained, and I/O ports retain their states prior to the transition.

4.8.2 Clearing Subsleep Mode

Subsleep mode is cleared by an interrupt (Timer A interrupt, NMI pin, or pin $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ5}}$), or by means of the $\overline{\text{RES}}$ pin.

(1) Clearing with an Interrupt

When an interrupt request signal is input, subsleep mode is cleared and interrupt exception handling is started. Subsleep mode cannot be cleared with an $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ5}}$ interrupt if the corresponding enable bit has been cleared to 0, or with an on-chip supporting module interrupt if acceptance of the relevant interrupt has been disabled by the interrupt enable register or masked by the CPU.

(2) Clearing with the $\overline{\text{RES}}$ Pin

See (2) Clearing with the $\overline{\text{RES}}$ Pin in section 4.6.2, Clearing Standby Mode.

4.9 Subactive Mode

4.9.1 Subactive Mode

If a SLEEP instruction is executed in high-speed mode when the SSBY bit in SBYCR, the DTON bit in LPWRCR, and the TMA3 bit in TMA (Timer A) are all set to 1, the CPU makes a transition to subactive mode. When an interrupt is generated in watch mode, if the LSON bit in LPWRCR is set to 1, a transition is made to subactive mode. When an interrupt is generated in subsleep mode, a transition is made to subactive mode.

In subactive mode, the CPU performs sequential program execution at low speed on the subclock. In this mode, all on-chip supporting modules other than Timer A stop.

4.9.2 Clearing Subactive Mode

Subsleep mode is cleared by a SLEEP instruction, or by means of the $\overline{\text{RES}}$ pin.

(1) Clearing with a SLEEP Instruction

When a SLEEP instruction is executed while the SSBY bit in SBYCR is set to 1, the DTON bit in LPWRCR is cleared to 0, and the TMA3 bit in TMA (Timer A) is set to 1, subactive mode is cleared and a transition is made to watch mode. When a SLEEP instruction is executed while the SSBY bit in SBYCR is cleared to 0, the LSON bit in LPWRCR is set to 1, and the TMA3 bit in TMA (Timer A) is set to 1, a transition is made to subsleep mode.

When a SLEEP instruction is executed while the SSBY bit in SBYCR is set to 1, the DTON bit is set to 1 and the LSON bit is cleared to 0 in LPWRCR, and the PSS bit in TCSR (WDT1) is set to 1, a transition is made directly to high-speed or medium-speed mode.

For details of direct transition, see section 4.10, Direct Transition.

(2) Clearing with the $\overline{\text{RES}}$ Pin

See (2) Clearing with the $\overline{\text{RES}}$ Pin in section 4.6.2, Clearing Standby Mode.

4.10 Direct Transition

4.10.1 Overview of Direct Transition

There are three operating modes in which the CPU executes programs: high-speed mode, medium-speed mode, and subactive mode. A transition between high-speed mode and subactive mode without halting the program* is called a direct transition. A direct transition can be carried out by setting the DTON bit in LPWRCR to 1 and executing a SLEEP instruction. After the transition, direct transition interrupt exception handling is started.

(1) Direct Transition from High-Speed Mode to Subactive Mode

If a SLEEP instruction is executed in high-speed mode while the SSBY bit in SBYCR, the LSON bit and DTON bit in LPWRCR, and the TMA3 bit in TMA (Timer A) are all set to 1, a transition is made to subactive mode.

(2) Direct Transition from Subactive Mode to High-Speed Mode/Medium-Speed Mode

If a SLEEP instruction is executed in subactive mode while the SSBY bit in SBYCR is set to 1, the LSON bit is cleared to 0 and the DTON bit is set to 1 in LPWRCR, and the TMA3 bit in TMA (Timer A) is set to 1, after the elapse of the time set in bits STS2 to STS0 in SBYCR, a transition is made to directly to high-speed mode.

Note: * At the time of transition from subactive mode to high- or medium-speed mode, an oscillation stabilization wait time is generated.

Section 5 Exception Handling


5.1 Overview

5.1.1 Exception Handling Types and Priority

As table 5.1 indicates, exception handling may be caused by a reset, trap instruction, or interrupt. Exception handling is prioritized as shown in table 5.1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority. Trap instruction exceptions are accepted at all times in the program execution state.

Exception handling sources, the stack structure, and the operation of the CPU vary depending on the interrupt control mode set by the INTM0 and INTM1 bits in SYSCR.

Table 5.1 Exception Types and Priority

| Priority | Exception Type | Start of Exception Handling |
|--|--|--|
|  High | Reset | Starts immediately after a low-to-high transition at the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows |
| | Trace ^{*1} | Starts when execution of the current instruction or exception handling ends, if the trace (T) bit is set to 1 |
| | Interrupt | Starts when execution of the current instruction or exception handling ends, if an interrupt request has been issued ^{*2} |
| | Direct transition | Started by a direct transition resulting from execution of a SLEEP instruction |
| | Trap instruction (TRAPA) ^{*3} | Started by execution of a trap instruction (TRAPA) |
| Low | | |

- Notes: 1. Traces are enabled only in interrupt control modes 2 and 3. (They cannot be used in this LSI.) Trace exception handling is not executed after execution of an RTE instruction.
2. Interrupt detection is not performed on completion of ANDC, ORC, XORC, or LDC instruction execution, or on completion of reset exception handling.
3. Trap instruction exception handling requests are accepted at all times in the program execution state.

5.1.2 **Exception Handling Operation**

Exceptions originate from various sources. Trap instructions and interrupts are handled as follows:

- [1] The program counter (PC) and condition-code register (CCR) are pushed onto the stack.
- [2] The interrupt mask bits are updated. The T bit is cleared to 0.
- [3] A vector address corresponding to the exception source is generated, and program execution starts from that address.

For a reset exception, steps [2] and [3] above are carried out.

5.1.3 **Exception Sources and Vector Table**

The exception sources are classified as shown in figure 5.1. Different vector addresses are assigned to different exception sources.

Table 5.2 lists the exception sources and their vector addresses.

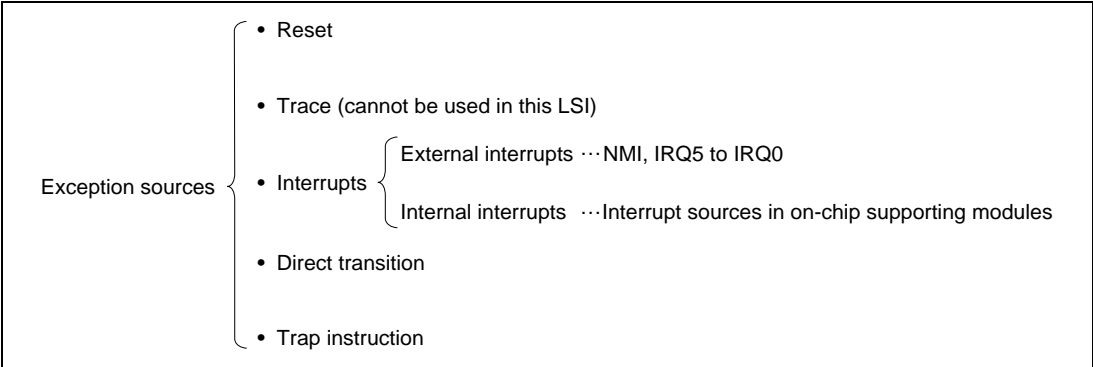


Figure 5.1 Exception Sources

Table 5.2 Exception Vector Table

| Exception Source | | Vector Number | Vector Address ^{*1} |
|----------------------------------|------|---------------|------------------------------|
| Reset | | 0 | H'0000 to H'0003 |
| Reserved for system use | | 1 | H'0004 to H'0007 |
| | | 2 | H'0008 to H'000B |
| | | 3 | H'000C to H'000F |
| | | 4 | H'0010 to H'0013 |
| | | 5 | H'0014 to H'0017 |
| Direct transition | | 6 | H'0018 to H'001B |
| External interrupt | NMI | 7 | H'001C to H'001F |
| Trap instruction (4 sources) | | 8 | H'0020 to H'0023 |
| | | 9 | H'0024 to H'0027 |
| | | 10 | H'0028 to H'002B |
| | | 11 | H'002C to H'002F |
| Reserved for system use | | 12 | H'0030 to H'0033 |
| | | 13 | H'0034 to H'0037 |
| | | 14 | H'0038 to H'003B |
| | | 15 | H'003C to H'003F |
| Address trap | #0 | 16 | H'0040 to H'0043 |
| | #1 | 17 | H'0044 to H'0047 |
| | #2 | 18 | H'0048 to H'004B |
| Internal interrupt (IC) | | 19 | H'004C to H'004F |
| Internal interrupt (HSW1) | | 20 | H'0050 to H'0053 |
| External interrupt | IRQ0 | 21 | H'0054 to H'0057 |
| | IRQ1 | 22 | H'0058 to H'005B |
| | IRQ2 | 23 | H'005C to H'005F |
| | IRQ3 | 24 | H'0060 to H'0063 |
| | IRQ4 | 25 | H'0064 to H'0067 |
| | IRQ5 | 26 | H'0068 to H'006B |
| Reserved | | 27 | H'006C to H'006F |
| | | | |
| | | 33 | H'0084 to H'0087 |
| Internal interrupt ^{*2} | | 30 | H'0088 to H'008B |
| | | | |
| | | 67 | H'010C to H'010F |

Notes: 1. Lower 16 bits of the address.

2. For details on internal interrupt vectors, see section 6.3.3, Interrupt Exception Vector Table.

5.2 Reset

5.2.1 Overview

A reset has the highest exception priority.

When the $\overline{\text{RES}}$ pin goes low, all processing halts and the MCU enters the reset state. A reset initializes the internal state of the CPU and the registers of on-chip supporting modules.

Immediately after a reset, interrupt control mode 0 is set.

Reset exception handling begins when the $\overline{\text{RES}}$ pin changes from low to high.

The MCUs can also be reset by overflow of the watchdog timer. For details, see section 17, Watchdog Timer.

5.2.2 Reset Sequence

The MCU enters the reset state when the $\overline{\text{RES}}$ pin goes low.

To ensure that the chip is reset, hold the $\overline{\text{RES}}$ pin low during the oscillation stabilizing time of the clock oscillator when powering on. To reset the chip during operation, hold the $\overline{\text{RES}}$ pin low for at least 20 states. For pin states in a reset, see Appendix D.1, Pin Circuit Diagrams.

When the $\overline{\text{RES}}$ pin goes high after being held low for the necessary time, the chip starts reset exception handling as follows:

- [1] The internal state of the CPU and the registers of the on-chip supporting modules are initialized, and the I bit is set to 1 in CCR.
- [2] The reset exception vector address is read and transferred to the PC, and program execution starts from the address indicated by the PC.

Figures 5.2 shows examples of the reset sequence.

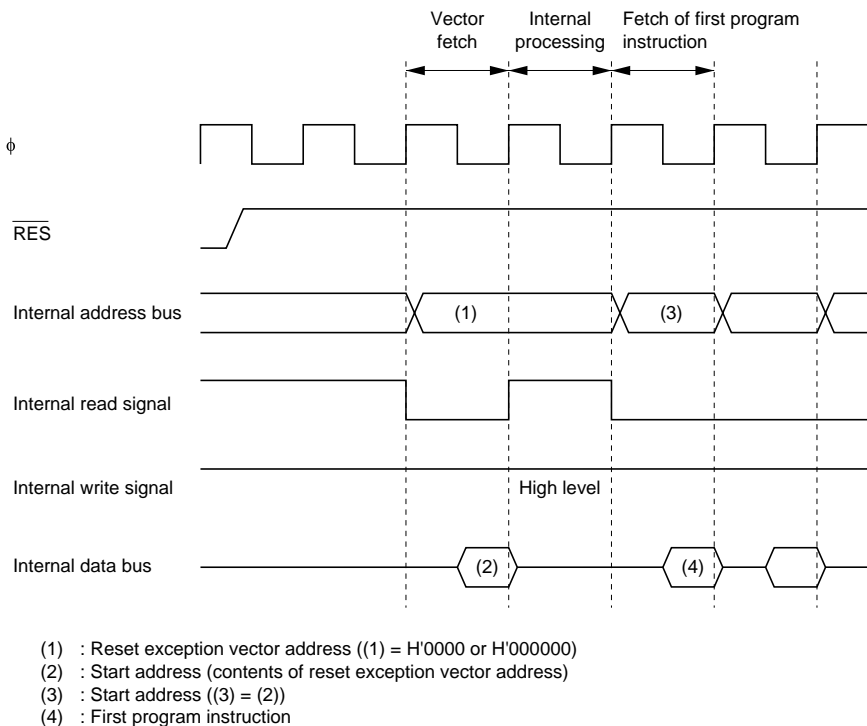


Figure 5.2 Reset Sequence (Mode 1)

5.2.3 Interrupts after Reset

If an interrupt is accepted after a reset but before the stack pointer (SP) is initialized, the PC and CCR will not be saved correctly, leading to a program crash. To prevent this, all interrupt requests, including NMI, are disabled immediately after a reset. Since the first instruction of a program is always executed immediately after the reset state ends, make sure that this instruction initializes the stack pointer (example: `MOV.L #xx:32, SP`).

5.3 Interrupts

Interrupt exception handling can be requested by seven external sources (NMI and IRQ5 to IRQ0) and internal sources in the on-chip supporting modules. Figure 5.3 shows the interrupt sources and the number of interrupts of each type.

The on-chip supporting modules that can request interrupts include the watchdog timer (WDT), prescaler unit (PSU), Timers A, B, J, L, R and X1 (TMR), serial communication interface (SCI), A/D converter (ADC), I²C bus interface (IIC), servo circuits, synchronized detection, address trap, etc. Each interrupt source has a separate vector address.

NMI is the highest-priority interrupt. Interrupts are controlled by the interrupt controller. The interrupt controller has two interrupt control modes and can assign interrupts other than NMI to either three priority/mask levels to enable multiplexed interrupt control.

For details on interrupts, see section 6, Interrupt Controller.

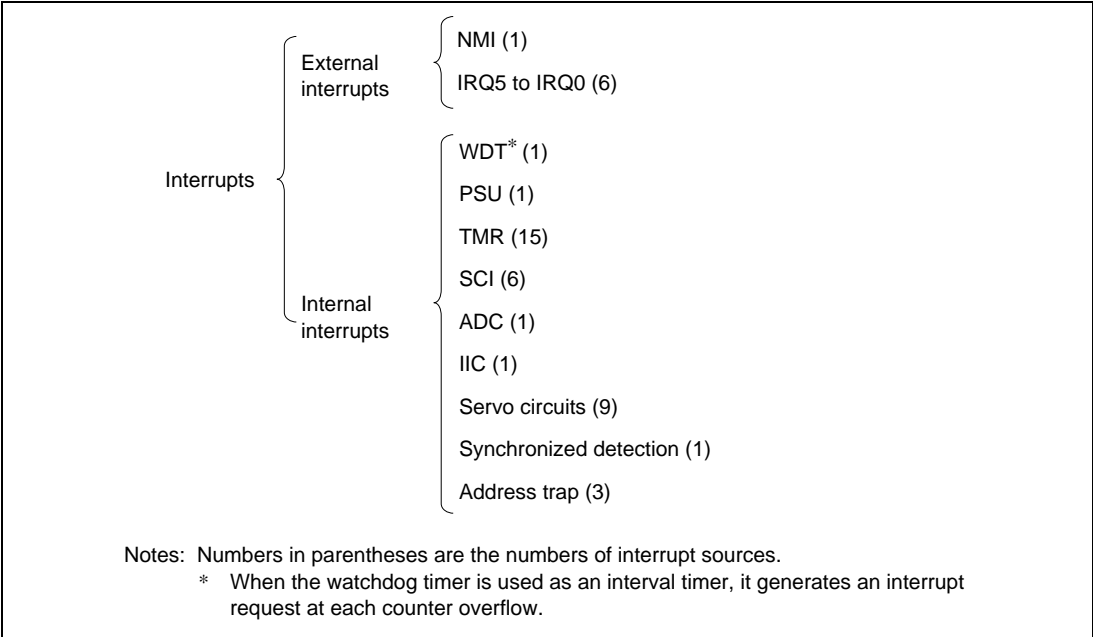


Figure 5.3 Interrupt Sources and Number of Interrupts

5.4 Trap Instruction

Trap instruction exception handling starts when a TRAPA instruction is executed. Trap instruction exception handling can be executed at all times in the program execution state. The TRAPA instruction fetches a start address from a vector table entry corresponding to a vector number from 0 to 3, as specified in the instruction code.

Table 5.3 shows the status of CCR and EXR after execution of trap instruction exception handling.

Table 5.3 Status of CCR and EXR after Trap Instruction Exception Handling

| Interrupt Control Mode | CCR | | EXR* | |
|---------------------------|-----|----|----------|---|
| | I | UI | I2 to I0 | T |
| 0 | 1 | — | — | — |
| 1 | 1 | 1 | — | — |

Legend:

1: Set to 1

0: Cleared to 0

—: Retains value prior to execution.

*: Does not affect operation in this LSI.

5.5 Stack Status after Exception Handling

Figure 5.4 shows the stack after completion of trap instruction exception handling and interrupt exception handling.

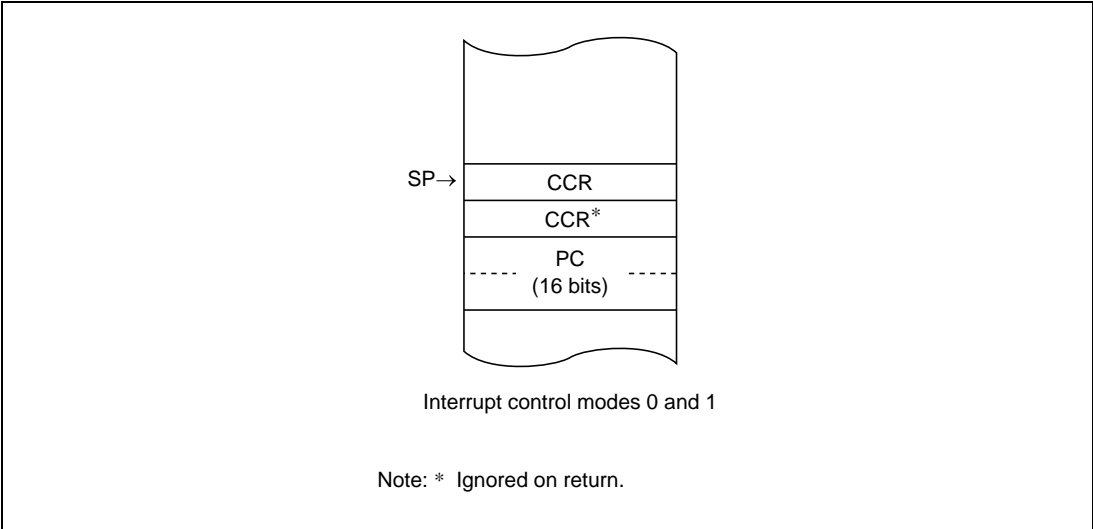


Figure 5.4 (1) Stack Status after Exception Handling (Normal Mode)*

Note: * Normal mode is not available for this LSI.

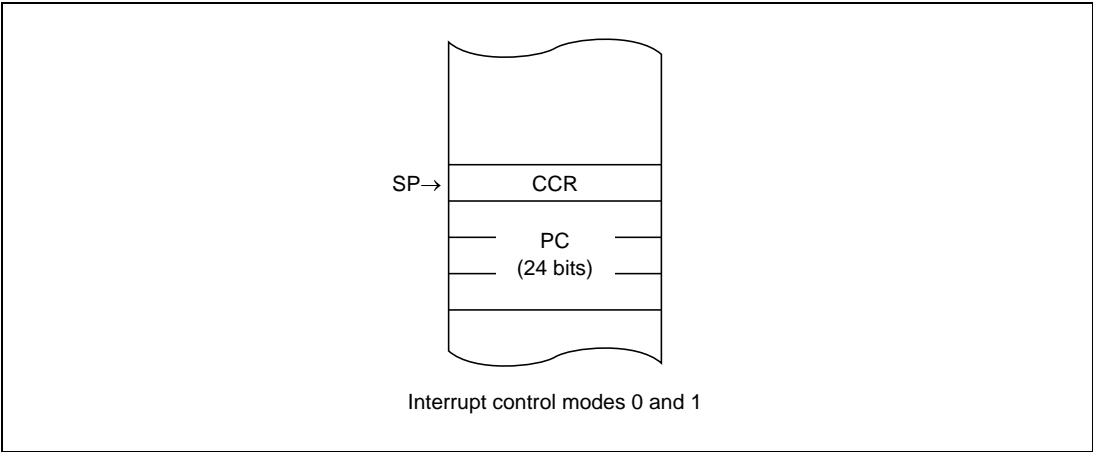


Figure 5.4 (2) Stack Status after Exception Handling (Advanced Mode)

5.6 Notes on Use of the Stack

When accessing word data or longword data, this chip assumes that the lowest address bit is 0. The stack should always be accessed by word transfer instruction or longword transfer instruction, and the value of the stack pointer (SP: ER7) should always be kept even. Use the following instructions to save registers:

```
PUSH.W   Rn      (or MOV.W Rn, @-SP)
PUSH.L   ERn     (or MOV.L ERn, @-SP)
```

Use the following instructions to restore registers:

```
POP.W Rn  (or MOV.W @SP+, Rn)
POP.L ERn (or MOV.L @SP+, ERn)
```

Setting SP to an odd value may lead to a malfunction. Figure 5.5 shows an example of what happens when the SP value is odd.

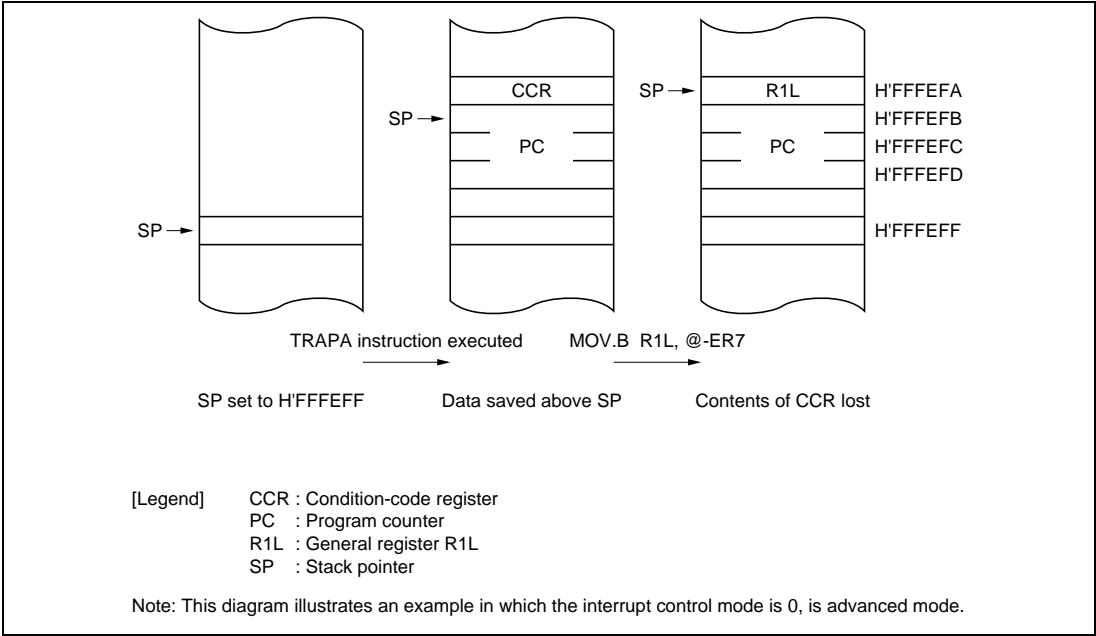


Figure 5.5 Operation when SP Value is Odd

Section 6 Interrupt Controller

6.1 Overview

6.1.1 Features

This LSI controls interrupts by means of an interrupt controller. The interrupt controller has the following features:

(1) Two Interrupt Control Modes

- Either of two interrupt control modes can be set by means of the INTM1 and INTM0 bits in the system control register (SYSCR).

(2) Priorities Settable with ICR

- An interrupt control register (ICR) is provided for setting interrupt priorities. Three priority levels can be set for each module for all interrupts except NMI.

(3) Independent Vector Addresses

- All interrupt sources are assigned independent vector addresses, making it unnecessary for the source to be identified in the interrupt handling routine.

(4) Seven External Interrupt Pins

- NMI is the highest-priority interrupt, and is accepted at all times. Falling edge, rising edge, or both edge detection can be selected for the NMI interrupt.
- Falling edge, rising edge, or both edge detection can be selected for interrupt IRQ0.
- Falling edge or rising edge can be individually selected for interrupts IRQ5 to IRQ1.

6.1.2 Block Diagram

A block diagram of the interrupt controller is shown in figure 6.1.

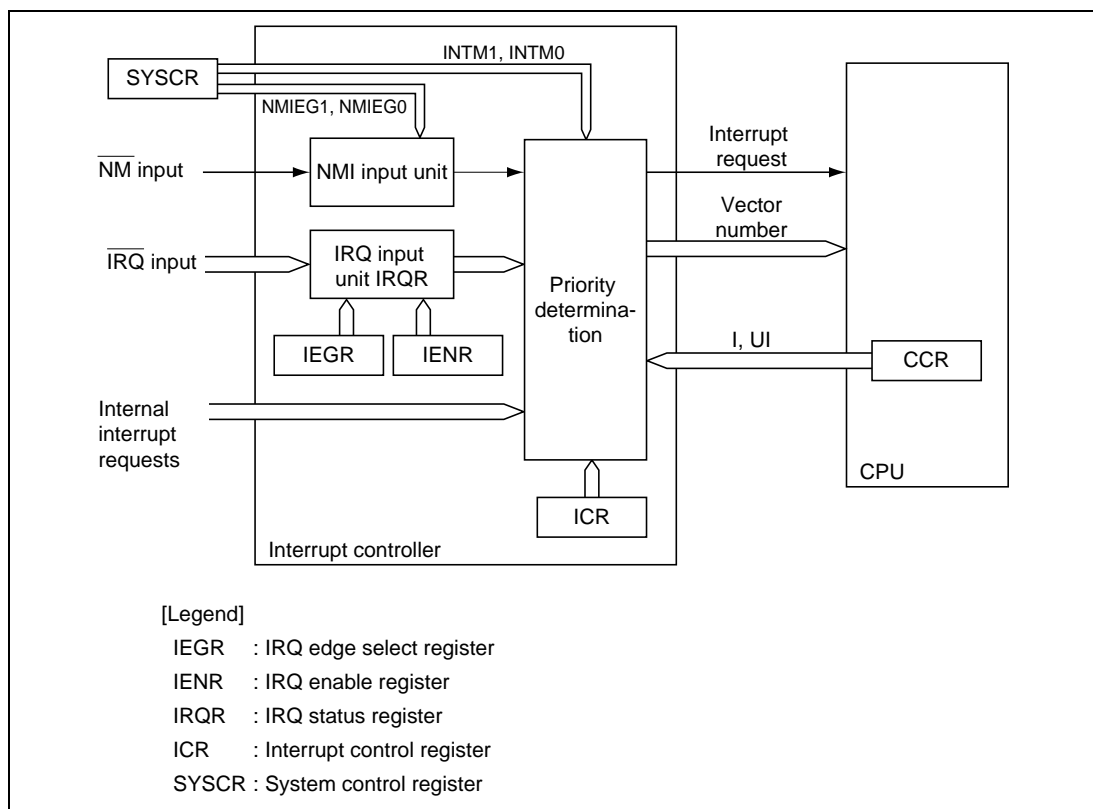


Figure 6.1 Block Diagram of Interrupt Controller

6.1.3 Pin Configuration

Table 6.1 summarizes the pins of the interrupt controller.

Table 6.1 Interrupt Controller Pins

| Name | Symbol | I/O | Function |
|------------------------------------|--|-------|--|
| Nonmaskable interrupt | $\overline{\text{NMI}}$ | Input | Nonmaskable external interrupt; rising, falling, or both edges can be selected |
| External interrupt request | $\overline{\text{IRQ0}}$ | Input | Maskable external interrupts; rising, falling, or both edges can be selected |
| External interrupt requests 1 to 5 | $\overline{\text{IRQ1}}$ to $\overline{\text{IRQ5}}$ | Input | Maskable external interrupts: rising, or falling edges can be selected |

6.1.4 Register Configuration

Table 6.2 summarizes the registers of the interrupt controller.

Table 6.2 Interrupt Controller Registers

| Name | Abbreviation | R/W | Initial Value | Address ^{*1} |
|------------------------------|--------------|----------------------|---------------|-----------------------|
| System control register | SYSCR | R/W | H'00 | H'FFE8 |
| IRQ edge select register | IEGR | R/W | H'00 | H'FFF0 |
| IRQ enable register | IENR | R/W | H'00 | H'FFF1 |
| IRQ status register | IRQR | R/ (W) ^{*2} | H'00 | H'FFF2 |
| Interrupt control register A | ICRA | R/W | H'00 | H'FFF3 |
| Interrupt control register B | ICRB | R/W | H'00 | H'FFF4 |
| Interrupt control register C | ICRC | R/W | H'00 | H'FFF5 |
| Interrupt control register D | ICRD | R/W | H'00 | H'FFF6 |
| Port mode register 1 | PMR1 | R/W | H'00 | H'FFCE |

Notes: 1. Lower 16 bits of the address.

2. Only 0 can be written, for flag clearing.

6.2 Register Descriptions

6.2.1 System Control Register (SYSCR)

| | | | | | | | | |
|-----------------|---|---|-------|-------|------|--------|--------|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | INTM1 | INTM0 | XRST | NMIEG1 | NMIEG0 | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/W | R/W | R | R/W | R/W | — |

SYSCR is an 8-bit readable register that selects the interrupt control mode and the detected edge for $\overline{\text{NMI}}$.

Only bits 5, 4, 2 and 1 are described here; for details on the other bits, see section 3.2.2, System Control Register (SYSCR).

SYSCR is initialized to H'00 by a reset.

Bits 5 and 4: Interrupt Control Mode (INTM1, INTM0)

These bits select one of two interrupt control modes for the interrupt controller. The INTM1 bit must not be set to 1.

| Bit 5 | Bit 4 | Interrupt Control Mode | Description |
|-------|-------|------------------------|--|
| INTM1 | INTM0 | | |
| 0 | 0 | 0 | Interrupts are controlled by I bit (Initial value) |
| | 1 | 1 | Interrupts are controlled by I and UI bits and ICR |
| 1 | 0 | — | Cannot be used in this LSI |
| | 1 | — | Cannot be used in this LSI |

Bit 2 and 1: $\overline{\text{NMI}}$ Pin Detected Edge Select (NMIEG1, NMIEG0)

Selects the detected edge for the $\overline{\text{NMI}}$ pin.

| Bit 2 | Bit 1 | Description |
|--------|--------|---|
| NMIEG1 | NMIEG0 | |
| 0 | 0 | Interrupt request generated at falling edge of $\overline{\text{NMI}}$ pin (Initial value) |
| | 1 | Interrupt request generated at rising edge of $\overline{\text{NMI}}$ pin |
| 1 | * | Interrupt request generated at both falling and rising edges of $\overline{\text{NMI}}$ pin |

Note: * Don't care

6.2.2 Interrupt Control Registers A to D (ICRA to ICRD)

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICR7 | ICR6 | ICR5 | ICR4 | ICR3 | ICR2 | ICR1 | ICR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The ICR registers are four 8-bit readable/writable registers that set the interrupt control level for interrupts other than NMI.

The correspondence between ICR settings and interrupt sources is shown in table 6.3.

The ICR registers are initialized to H'00 by a reset.

Bit 7 to 0: Interrupt Control Level (ICR7 to ICR0)

Sets the control level for the corresponding interrupt source.

Bit n

| ICRn | Description |
|------|--|
| 0 | Corresponding interrupt source is control level 0 (non-priority) (Initial value) |
| 1 | Corresponding interrupt source is control level 1 (priority) |

(n = 7 to 0)

Table 6.3 Correspondence between Interrupt Sources and ICR Settings

| | | | | | | | | |
|------|----------|------------------------|-----------------------------|----------|----------|--------------|---------------------------|----------|
| ICRA | ICRA7 | ICRA6 | ICRA5 | ICRA4 | ICRA3 | ICRA2 | ICRA1 | CIRA0 |
| | Reserved | Input capture | HSW1 | IRQ0 | IRQ1 | IRQ2 IRQ3 | IRQ4 IRQ5 | Reserved |
| ICRB | ICRB7 | ICRB6 | ICRB5 | ICRB4 | ICRB3 | ICRB2 | ICRB1 | ICRB0 |
| | Reserved | Reserved | Servo (drum, capstan latch) | Timer A | Timer B | Timer J | Timer R | Timer L |
| ICRC | ICRC7 | ICRC6 | ICRC5 | ICRC4 | ICRC3 | ICRC2 | ICRC1 | ICRC0 |
| | Timer X1 | Synchronized detection | Watchdog timer | Servo | IIC | SCI1 (UART) | SCI2 (with 32-bit buffer) | A/D |
| ICRD | ICRD7 | ICRD6 | ICRD5 | ICRD4 | ICRD3 | ICRD2 | ICRD1 | ICRD0 |
| | HSW2 | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved |

6.2.3 **IRQ Enable Register (IENR)**

| | | | | | | | | |
|-----------------|---|---|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | IRQ5E | IRQ4E | IRQ3E | IRQ2E | IRQ1E | IRQ0E |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

IENR is an 8-bit readable/writable register that controls enabling and disabling of interrupt requests IRQ5 to IRQ0.
IENR is initialized to H'00 by a reset.

Bits 7 and 6: Reserved
Do not write 1 to them.

Bits 5 to 0: IRQ5 to IRQ0 Enable (IRQ5E to IRQ0E)
These bits select whether IRQ5 to IRQ0 are enabled or disabled.

| Bit n | |
|--------------|---|
| IRQnE | Description |
| 0 | IRQn interrupt disabled (Initial value) |
| 1 | IRQn interrupt enabled |
| (n = 5 to 0) | |

6.2.4 IRQ Edge Select Registers (IEGR)

| | | | | | | | | |
|-----------------|---|--------|--------|--------|--------|--------|---------|---------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | IRQ5EG | IRQ4EG | IRQ3EG | IRQ2EG | IRQ1EG | IRQ0EG1 | IRQ0EG0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

IEGR is an 8-bit readable/writable register that selects detected edge of the input at pins $\overline{\text{IRQ5}}$ to $\overline{\text{IRQ0}}$.

IEGR register is initialized to H'00 by a reset.

Bit 7: Reserved

Do not write 1 to it.

Bits 6 to 2: $\overline{\text{IRQ5}}$ to $\overline{\text{IRQ1}}$ Pins Detected Edge Select (IRQ5EG to IRQ1EG)

These bits select detected edge for interrupts IRQ5 to IRQ1.

Bits 6 to 2

| IRQnEG | Description |
|--------|---|
| 0 | Interrupt request generated at falling edge of $\overline{\text{IRQn}}$ pin input (Initial value) |
| 1 | Interrupt request generated at rising edge of $\overline{\text{IRQn}}$ pin input |

(n = 5 to 1)

Bits 1 and 0: $\overline{\text{IRQ0}}$ Pin Detected Edge Select (IRQ0EG1, IRQ0EG0)

These bits select detected edge for interrupt IRQ0.

| Bit 1 | Bit 0 | Description |
|---------|---------|--|
| IRQ0EG1 | IRQ0EG0 | |
| 0 | 0 | Interrupt request generated at falling edge of $\overline{\text{IRQ0}}$ pin input (Initial value) |
| 0 | 1 | Interrupt request generated at rising edge of $\overline{\text{IRQ0}}$ pin input |
| 1 | * | Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ0}}$ pin input |

Note: * Don't care

6.2.5 IRQ Status Register (IRQR)

| | | | | | | | | |
|-----------------|---|---|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | IRQ5F | IRQ4F | IRQ3F | IRQ2F | IRQ1F | IRQ0F |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Note: * Only 0 can be written, to clear the flag.

IRQR is an 8-bit readable/writable register that indicates the status of IRQ5 to IRQ0 interrupt requests.

IRQR is initialized to H'00 by a reset.

Bit 7 and 6: Reserved

Do not write 1 to them.

Bits 5 to 0: IRQ5 to IRQ0 Flags

These bits indicate the status of IRQ5 to IRQ0 interrupt requests.

Bit n

| IRQnF | Description |
|--------------|---|
| 0 | [Clearing conditions] (Initial value) Cleared by reading IRQnF set to 1, then writing 0 in IRQnF When IRQn interrupt exception handling is executed |
| 1 | [Setting conditions] (1) When a falling edge occurs in $\overline{\text{IRQn}}$ input while falling edge detection is set (IRQnEG = 0) (2) When a rising edge occurs in $\overline{\text{IRQn}}$ input while rising edge detection is set (IRQnEG = 0) (3) When a falling or rising edge occurs in $\overline{\text{IRQ0}}$ input while both-edge detection is set (IRQ0EG1 = 1) |
| (n = 5 to 0) | |

6.2.6 Port Mode Register (PMR1)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR17 | PMR16 | PMR15 | PMR14 | PMR13 | PMR12 | PMR11 | PMR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port Mode Register 1 (PMR1) controls pin function switching-over of port 1. Switching is specified for each bit.
PMR1 is an 8-bit readable/writable register and is initialized to H'00 by a reset.
Only bits 5 to 0 are explained here. For details, see section 10, I/O Port.

Bits 5 to 0: P15/ $\overline{\text{IRQ5}}$ to P10/ $\overline{\text{IRQ0}}$ pin switching (PMR15 to PMR10)

These bits are for setting the P1n/ $\overline{\text{IRQn}}$ pin as the input/output pin for P1n or as the $\overline{\text{IRQn}}$ pin for external interrupt request input.

| Bit n | |
|-------|---|
| PMR1n | Description |
| 0 | P1n/ $\overline{\text{IRQn}}$ pin functions as the P1n input/output pin (Initial value) |
| 1 | P1n/ $\overline{\text{IRQn}}$ pin functions as the $\overline{\text{IRQn}}$ input pin |

(N = 5 to 0)

The following is the notes on switching the pin function by PMR1.

- (1) When the port is set as the $\overline{\text{IC}}$ input pin or $\overline{\text{IRQ5}}$ to $\overline{\text{IRQ0}}$ input pin, the pin level must be High or Low regardless of active mode or power-down mode. Do not set the pin level at Medium.
- (2) Switching the pin function of P16/ $\overline{\text{IC}}$ or P15/ $\overline{\text{IRQ5}}$ to P10/ $\overline{\text{IRQ0}}$ may be mistakenly identified as edge detection and detection signal may be generated. To prevent this, operate as follows:
 - (a) Set the interrupt enable/disable flag to disable before switching the pin function.
 - (b) Clear the applicable interrupt request flag to 0 after switching the pin function and executing another instruction.

(Program example)

```
:
MOV.B R0L,@IENR ..... Interrupt disabled
MOV.B R1L,@PMR1 ..... Pin function change
NOP ..... Optional instruction
BCLR m @IRQR ..... Applicable interrupt clear
MOV.B R1L,@IENR ..... Interrupt enabled
:
```

6.3 Interrupt Sources

Interrupt sources comprise external interrupts (NMI and IRQ5 to IRQ0) and internal interrupts.

6.3.1 External Interrupts

There are seven external interrupt sources; NMI and IRQ5 to IRQ0. Of these, NMI, and IRQ1 to IRQ0 can be used to restore this chip from standby mode.

(1) NMI Interrupt

NMI is the highest-priority interrupt, and is always accepted by the CPU regardless of the interrupt control mode and the status of the CPU interrupt mask bits. The NMIEG1 and NMIEG0 bits in SYSCR can be used to select whether an interrupt is requested at a rising, falling edge or both edges on the $\overline{\text{NMI}}$ pin.

The vector number for NMI interrupt exception handling is 7.

(2) IRQ5 to IRQ0 Interrupts

Interrupts IRQ5 to IRQ0 are requested by an input signal at pins $\overline{\text{IRQ5}}$ to $\overline{\text{IRQ0}}$. Interrupts IRQ5 to IRQ0 have the following features:

- (a) Using IEGR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, at pin $\overline{\text{IRQ0}}$.
- (b) Using IEGR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, at pins $\overline{\text{IRQ5}}$ to $\overline{\text{IRQ0}}$.
- (c) Enabling or disabling of interrupt requests IRQ5 to IRQ0 can be selected with IENR.
- (d) The interrupt control level can be set with ICR.
- (e) The status of interrupt requests IRQ5 to IRQ0 is indicated in IRQR. IRQR flags can be cleared to 0 by software.

A block diagram of interrupts IRQ5 to IRQ0 is shown in figure 6.2.

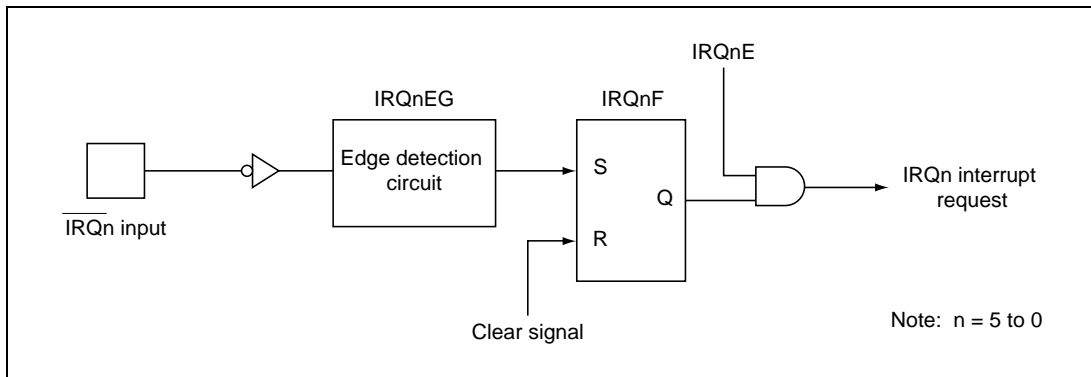


Figure 6.2 Block Diagram of Interrupts IRQ5 to IRQ0

Figure 6.3 shows the timing of IRQnF setting.

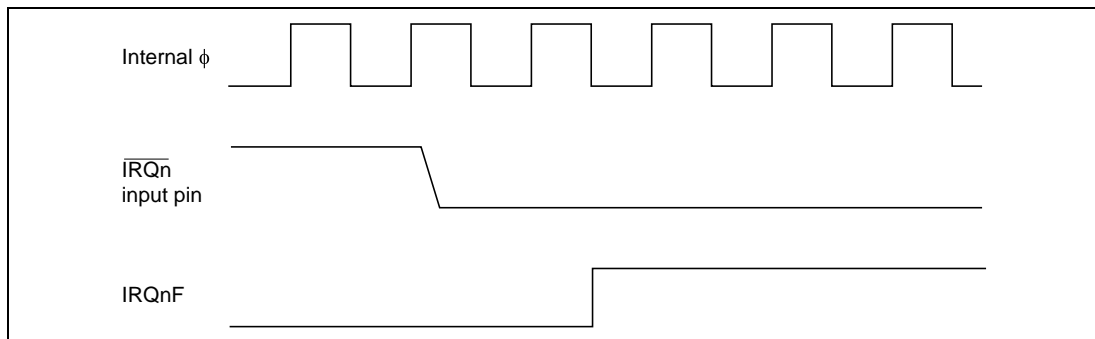


Figure 6.3 Timing of IRQnF Setting

The vector numbers for IRQ5 to IRQ0 interrupt exception handling are 21 to 26.

Upon detection of IRQ5 to IRQ0 interrupts, the applicable pin is set in the port mode register 1 (PMR1) as $\overline{\text{IRQn}}$ pin.

6.3.2 Internal Interrupts

There are 38 sources for internal interrupts from on-chip supporting modules.

- (1) For each on-chip supporting module there are flags that indicate the interrupt request status, and enable bits that select enabling or disabling of these interrupts. If any one of these is set to 1, an interrupt request is issued to the interrupt controller.
- (2) The interrupt control level can be set by means of ICR.


6.3.3 Interrupt Exception Vector Table

Table 6.4 shows interrupt exception handling sources, vector addresses, and interrupt priorities.

For default priorities, the lower the vector number, the higher the priority.

Priorities among modules can be set by means of ICR. The situation when two or more modules are set to the same priority, and priorities within a module, are fixed as shown in table 6.4.

Table 6.4 Interrupt Sources, Vector Addresses, and Interrupt Priorities

| Priority | Interrupt Source | | Origin of Interrupt Source | Vector No. | Vector address | ICR | Remarks |
|---|-------------------|---------|----------------------------|------------|------------------|-----|---------|
|  High | Reset | | External pin | 0 | H'0000 to H'0003 | — | |
| | Reserved | | — | 1 | H'0004 to H'0007 | — | |
| | | | — | 2 | H'0008 to H'000B | — | |
| | | | — | 3 | H'000C to H'000F | — | |
| | | | — | 4 | H'0010 to H'0013 | — | |
| | | | — | 5 | H'0014 to H'0017 | — | |
| | Direct transition | | Instruction | 6 | H'0018 to H'001B | — | |
| | NMI | | External pin | 7 | H'001C to H'001F | — | |
| | Trap instruction | TRAPA#0 | Instruction | 8 | H'0020 to H'0023 | — | |
| | | TRAPA#1 | | 9 | H'0024 to H'0027 | — | |
| | | TRAPA#2 | | 10 | H'0028 to H'002B | — | |
| | | TRAPA#3 | | 11 | H'002C to H'002F | — | |
| | Reserved | | — | 12 | H'0030 to H'0033 | — | |
| | | | | 13 | H'0034 to H'0037 | — | |
| | | | | 14 | H'0038 to H'003B | — | |
| | | | | 15 | H'003C to H'003F | — | |
| Low | | | | | | | |

| Priority | Interrupt Source | | Origin of Interrupt Source | Vector No. | Vector address | ICR | Remarks | | |
|------------------------------|-------------------------|----|----------------------------|------------|------------------|-------|---------|--|--|
| <div>High</div> <div>↑</div> | Address trap | #0 | ATC | 16 | H'0040 to H'0043 | — | | | |
| | | #1 | | 17 | H'0044 to H'0047 | | | | |
| | | #2 | | 18 | H'0048 to H'004B | | | | |
| | IC | | PSU | 19 | H'004C to H'004F | ICRA6 | | | |
| | HSW1 | | Servo circuit | 20 | H'0050 to H'0053 | ICRA5 | | | |
| | IRQ0 | | External pin | 21 | H'0054 to H'0057 | ICRA4 | | | |
| | IRQ1 | | | 22 | H'0058 to H'005B | ICRA3 | | | |
| | IRQ2 | | | 23 | H'005C to H'005F | ICRA2 | | | |
| | IRQ3 | | | 24 | H'0060 to H'0063 | | | | |
| | IRQ4 | | | 25 | H'0064 to H'0067 | ICRA1 | | | |
| | IRQ5 | | | 26 | H'0068 to H'006B | | | | |
| | Reserved | — | | 27 | H'006C to H'006F | — | | | |
| | | | | 28 | H'0070 to H'0073 | | | | |
| | | | | 29 | H'0074 to H'0077 | | | | |
| | | | | 30 | H'0078 to H'007B | | | | |
| | | | | 31 | H'007C to H'007F | | | | |
| | | | | 32 | H'0080 to H'0083 | | | | |
| | | | | 33 | H'0084 to H'0087 | | | | |
| | Drum latch 1 (speed) | | Servo circuit | 34 | H'0088 to H'008B | ICRB5 | | | |
| | Capstan latch 1 (speed) | | | 35 | H'008C to H'008F | | | | |
| | TMAI | | Timer A | 36 | H'0090 to H'0093 | ICRB4 | | | |
| | TMBI | | Timer B | 37 | H'0094 to H'0097 | ICRB3 | | | |
| | TMJ1I | | Timer J | 38 | H'0098 to H'009B | ICRB2 | | | |
| | TMJ2I | | | 39 | H'009C to H'009F | | | | |
| | TMR1I | | Timer R | 40 | H'00A0 to H'00A3 | ICRB1 | | | |
| | TMR2I | | | 41 | H'00A4 to H'00A7 | | | | |
| | TMR3I | | | 42 | H'00A8 to H'00AB | | | | |
| Low | TMLI | | Timer L | 43 | H'00AC to H'00AF | ICRB0 | | | |

| Priority | Interrupt Source | | Origin of Interrupt Source | Vector No. | Vector address | ICR | Remarks | | |
|------------------------------|-------------------------|------|----------------------------|------------------|------------------|-------|---------|-------|--|
| <div>High</div> <div>▲</div> | ICXA | | Timer X1 | 44 | H'00B0 to H'00B3 | ICRC7 | | | |
| | ICXB | | | 45 | H'00B4 to H'00B7 | | | | |
| | ICXC | | | 46 | H'00B8 to H'00BB | | | | |
| | ICXD | | | 47 | H'00BC to H'00BF | | | | |
| | OCX1 | | | 48 | H'00C0 to H'00C3 | | | | |
| | OCX2 | | | 49 | H'00C4 to H'00C7 | | | | |
| | OVFX | | | 50 | H'00C8 to H'00CB | | | | |
| | VD interrupts | | Sync signal detection | 51 | H'00CC to H'00CF | ICRC6 | | | |
| | Reserved | | — | 52 | H'00D0 to H'00D3 | | | | |
| | 8-bit interval timer | | Watchdog timer | 53 | H'00D4 to H'00D7 | | ICRC5 | | |
| | CTL | | Servo circuit | 54 | H'00D8 to H'00DB | | | ICRC4 | |
| | Drum latch 2 (speed) | | | 55 | H'00DC to H'00DF | | | | |
| | Capstan latch 2 (speed) | | | 56 | H'00E0 to H'00E3 | | | | |
| | Drum latch 3 (phase) | | | 57 | H'00E4 to H'00D7 | | | | |
| | Capstan latch 3 (phase) | | | 58 | H'00E8 to H'00EB | | | | |
| | IIC | | IIC | 59 | H'00EC to H'00EF | ICRC3 | | | |
| | SCI1 | ERI | SCI1 (UART) | 60 | H'00F0 to H'00F3 | | ICRC2 | | |
| | | RXI | | 61 | H'00F4 to H'00F7 | | | | |
| | | TXI | | 62 | H'00F8 to H'00FB | | | | |
| | | TEI | | 63 | H'00FC to H'00FF | | | | |
| SCI2 | TEI | SCI2 | 64 | H'0100 to H'0103 | ICRC1 | | | | |
| | ABTI | | 65 | H'0104 to H'0107 | | | | | |
| | A/D conversion end | | A/D | 66 | H'0108 to H'010B | ICRC0 | | | |
| Low | HSW2 | | Servo circuit | 67 | H'010C to H'010F | ICRD7 | | | |

6.4 Interrupt Operation

6.4.1 Interrupt Control Modes and Interrupt Operation

Interrupt operations in this LSI differ depending on the interrupt control mode. NMI interrupts and address trap interrupts are accepted at all times except in the reset state. In the case of IRQ interrupts and on-chip supporting module interrupts, an enable bit is provided for each interrupt. Clearing an enable bit to 0 disables the corresponding interrupt request. Interrupt sources for which the enable bits are set to 1 are controlled by the interrupt controller. Table 6.5 shows the interrupt control modes. The interrupt controller performs interrupt control according to the interrupt control mode set by the INTM1 and INTM0 bits in SYSCR, the priorities set in ICR, and the masking state indicated by the I and UI bits in the CPU's CCR.

Table 6.5 Interrupt Control Modes

| Interrupt Control Mode | SYSCR | | Priority Setting Register | Interrupt Mask Bits | Description |
|------------------------|-------|-------|---------------------------|---------------------|--|
| | INTM1 | INTM0 | | | |
| 0 | 0 | 0 | ICR | I | Interrupt mask control is performed by the I bit Priority can be set with ICR |
| 1 | | 1 | ICR | I, UI | 3-level interrupt mask control is performed by the I and UI bits Priority can be set with ICR |

Figure 6.4 shows a block diagram of the priority decision circuit.

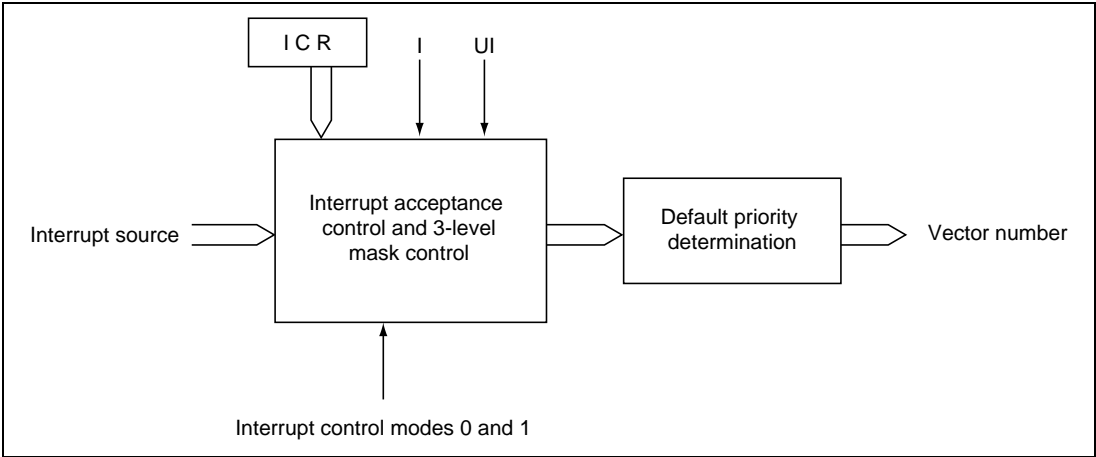


Figure 6.4 Block Diagram of Interrupt Priority Determination Operation

(1) Interrupt Acceptance Control and 3-Level Control

In interrupt control modes 0 and 1, interrupt acceptance control and 3-level mask control is performed by means of the I and UI bits in CCR, and ICR (control level).

Table 6.6 shows the interrupts selected in each interrupt control mode.

Table 6.6 Interrupts Selected in Each Interrupt Control Mode

| Interrupt Control Mode | Interrupt Mask Bit | | Selected Interrupts |
|------------------------|--------------------|----|--|
| | I | UI | |
| 0 | 0 | * | All interrupts (control level 1 has priority) |
| | 1 | * | NMI and address trap interrupts |
| 1 | 0 | * | All interrupts (control level 1 has priority) |
| | 1 | 0 | NMI, address trap and control level 1 interrupts |
| | | 1 | NMI and address trap interrupts |

Note: * Don't care

(2) Default Priority Determination

The priority is determined for the selected interrupt, and a vector number is generated.

If the same value is set for ICR, acceptance of multiple interrupts is enabled, and so only the interrupt source with the highest priority according to the preset default priorities is selected and has a vector number generated.

Interrupt sources with a lower priority than the accepted interrupt source are held pending.

Table 6.7 shows operations and control signal functions in each interrupt control mode.

Table 6.7 Operations and Control Signal Functions in Each Interrupt Control Mode

| Interrupt Control Mode | Setting | | Interrupt Acceptance Control, 3-Level Control | | | | Default Priority Determination |
|------------------------|---------|-------|---|----|-----|----|--------------------------------|
| | INTM1 | INTM0 | I | UI | ICR | | |
| 0 | 0 | 0 | ○ | IM | — | PR | ○ |
| 1 | | 1 | ○ | IM | IM | PR | ○ |

Legend:

○: Interrupt operation control performed

IM: Used as interrupt mask bit

PR: Sets priority

—: Not used

6.4.2 Interrupt Control Mode 0

Enabling and disabling of IRQ interrupts and on-chip supporting module interrupts can be set by means of the I bit in the CPU's CCR, and ICR. Interrupts are enabled when the I bit is cleared to 0, and disabled when set to 1. Control level 1 interrupt sources have higher priority.

Figure 6.5 shows a flowchart of the interrupt acceptance operation in this case.

- (1) If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
- (2) When interrupt requests are sent to the interrupt controller, a control level 1 interrupt, according to the control level set in ICR, has priority for selection, and other interrupt requests are held pending. If a number of interrupt requests with the same control level setting are generated at the same time, the interrupt request with the highest priority according to the priority system shown in table 6.4 is selected.
- (3) The I bit is then referenced. If the I bit is cleared to 0, the interrupt request is accepted. If the I bit is set to 1, only an NMI or an address trap interrupt is accepted, and other interrupt requests are held pending.
- (4) When an interrupt request is accepted, interrupt exception handling starts after execution of the current instruction has been completed.
- (5) The PC and CCR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.
- (6) Next, the I bit in CCR is set to 1. This disables all interrupts except NMI and address trap.
- (7) A vector address is generated for the accepted interrupt, and execution of the interrupt handling routine starts at the address indicated by the contents of that vector address.

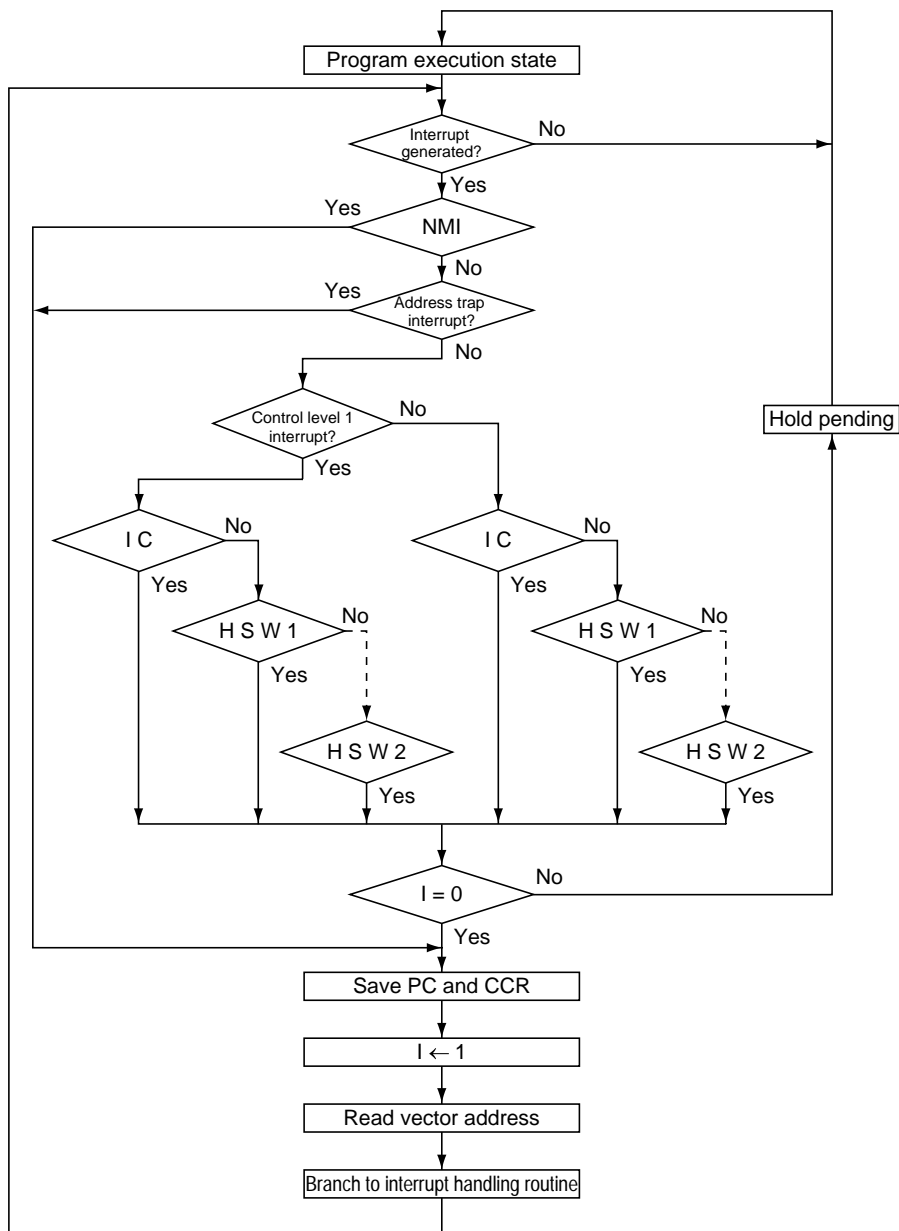


Figure 6.5 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0

6.4.3 Interrupt Control Mode 1

Three-level masking is implemented for IRQ interrupts and on-chip supporting module interrupts by means of the I and UI bits in the CPU's CCR, and ICR.

- (1) Control level 0 interrupt requests are enabled when the I bit is cleared to 0, and disabled when set to 1.
- (2) Control level 1 interrupt requests are enabled when the I bit or UI bit is cleared to 0, and disabled when both the I bit and the UI bit are set to 1.

For example, if the interrupt enable bit for an interrupt request is set to 1, and H'04, H'00, H'00 and H'00 are set in ICRA, ICRB, ICRC and ICRD respectively, (i.e. IRQ2 interrupt is set to control level 1 and other interrupts to control level 0), the situation is as follows:

- (1) When $I = 0$, all interrupts are enabled
(Priority order: $NMI > IRQ2 > IC > HSW1 > \dots$)
- (2) When $I = 1$ and $UI = 0$, only NMI, address trap and IRQ2 interrupts are enabled
- (3) When $I = 1$ and $UI = 1$, only NMI and address trap interrupts are enabled

Figure 6.6 shows the state transitions in these cases.

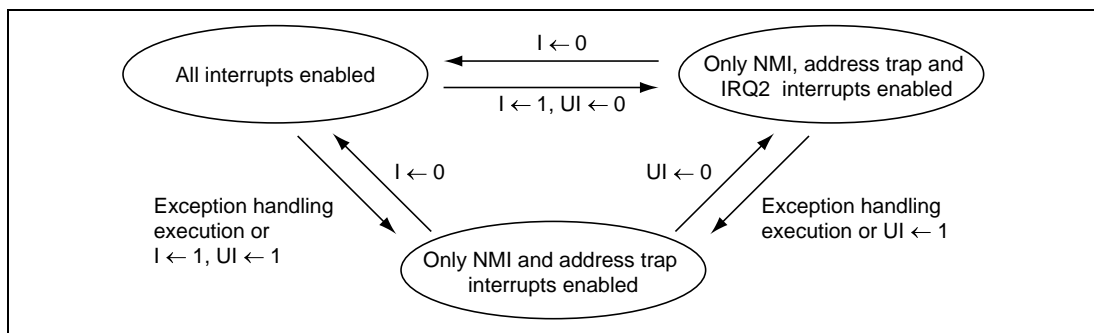


Figure 6.6 Example of State Transitions in Interrupt Control Mode 1

Figure 6.7 shows an operation flowchart of interrupt reception.

- (1) If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
- (2) When interrupt requests are sent to the interrupt controller, a control level 1 interrupt, according to the control level set in ICR, has priority for selection, and other interrupt requests are held pending. If a number of interrupt requests with the same control level setting are generated at the same time, the interrupt request with the highest priority according to the priority system shown in table 6.4 is selected.
- (3) The I bit is then referenced. If the I bit is cleared to 0, the UI bit has no effect.

An interrupt request set to interrupt control level 0 is accepted when the I bit is cleared to 0. If the I bit is set to 1, only NMI and address trap interrupts are accepted, and other interrupt requests are held pending.

An interrupt request set to interrupt control level 1 has priority over an interrupt request set to interrupt control level 0, and is accepted if the I bit is cleared to 0, or if the I bit is set to 1 and the UI bit is cleared to 0.

When both the I bit and the UI bit are set to 1, only NMI and address trap interrupts are accepted, and other interrupt requests are held pending.
- (4) When an interrupt request is accepted, interrupt exception handling starts after execution of the current instruction has been completed.
- (5) The PC and CCR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.
- (6) Next, the I and UI bits in CCR are set to 1. This masks all interrupts except NMI and address trap.
- (7) A vector address is generated for the accepted interrupt, and execution of the interrupt handling routine starts at the address indicated by the contents of that vector address.

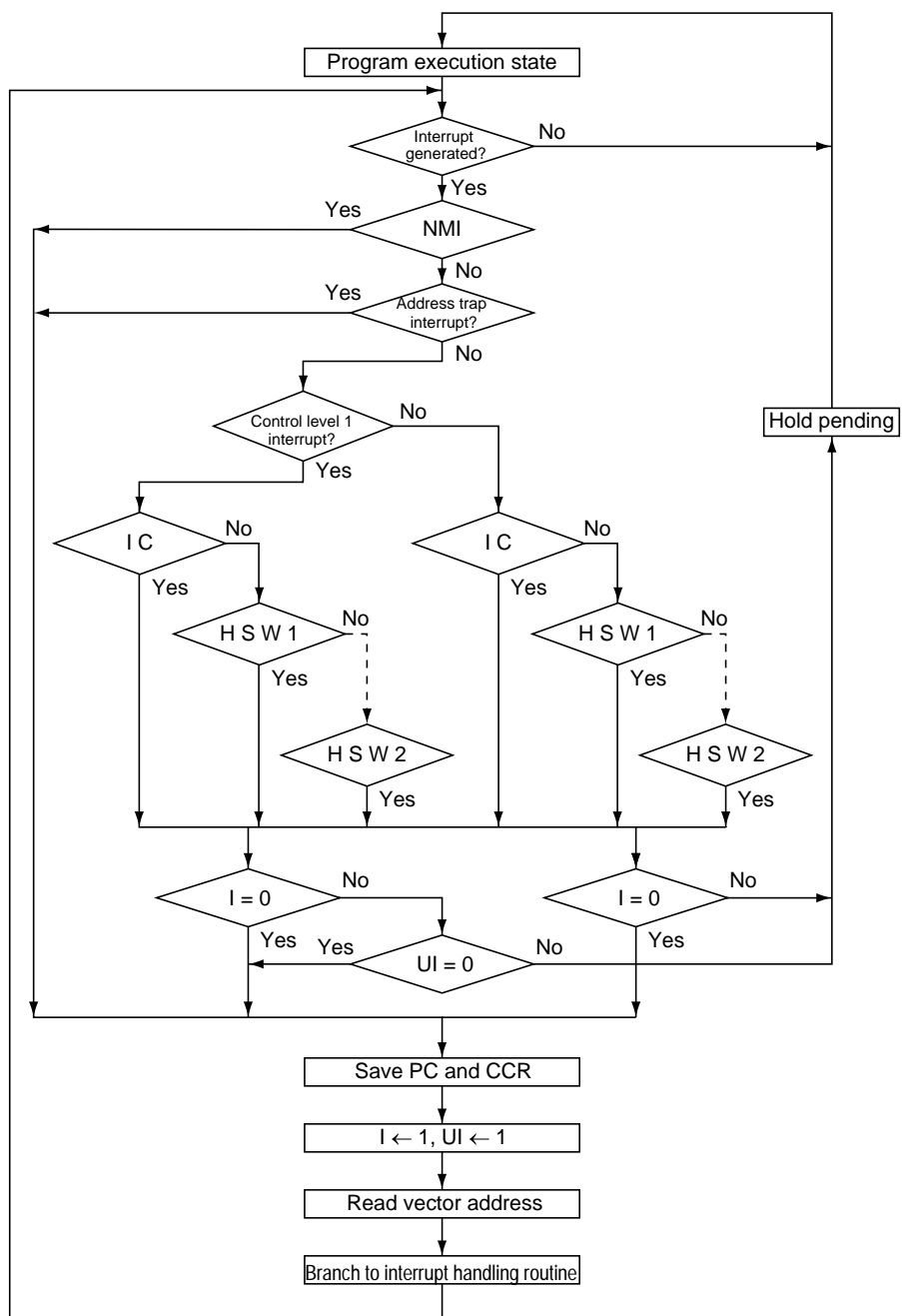


Figure 6.7 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 1

6.4.4 Interrupt Exception Handling Sequence

Figure 6.8 shows the interrupt exception handling sequence. The example shown is for the case where interrupt control 0 is set in advanced mode, and the program area and stack area are in on-chip memory.

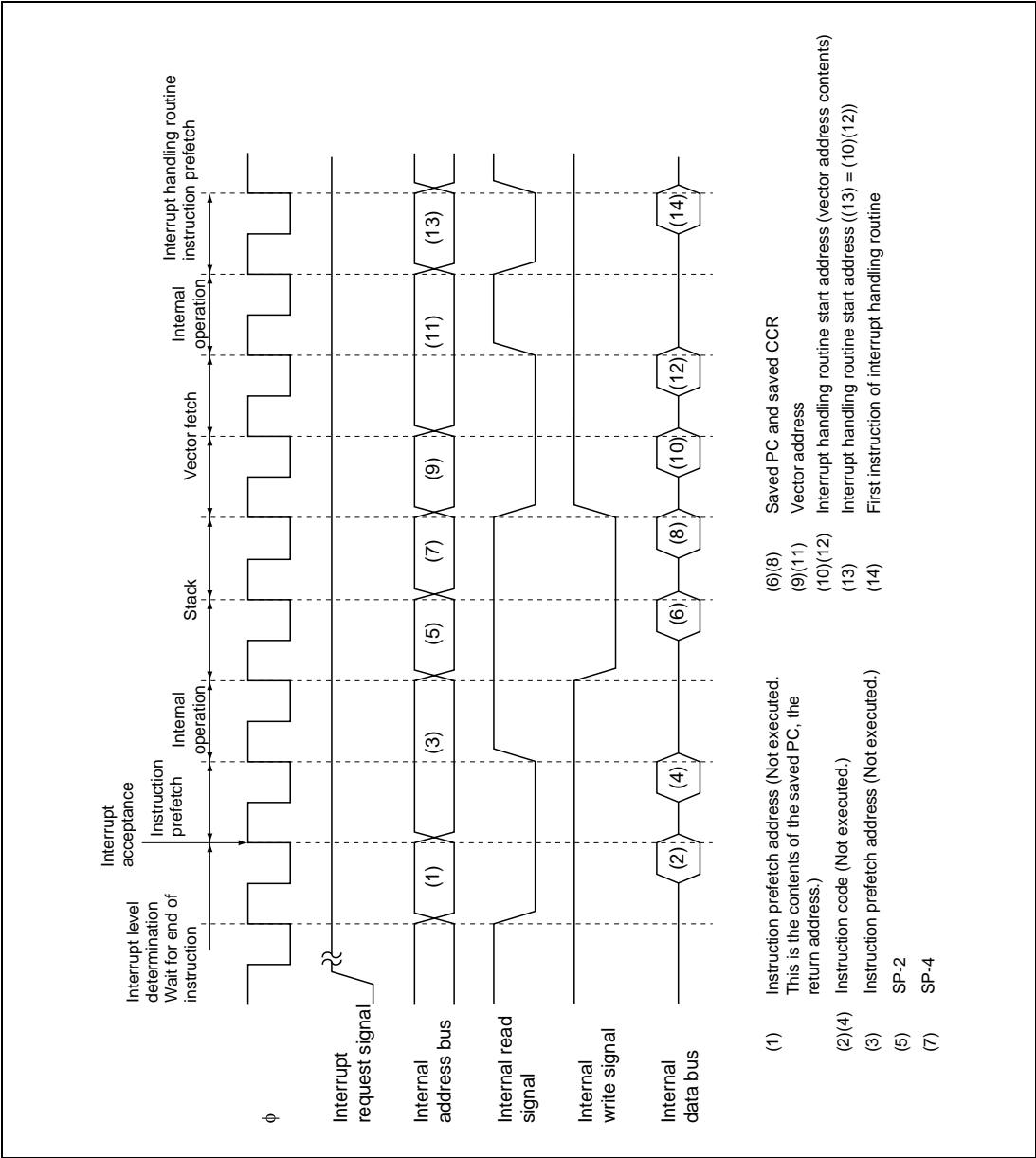


Figure 6.8 Interrupt Exception Handling

6.4.5 Interrupt Response Times

Table 6.8 shows interrupt response times-the interval between generation of an interrupt request and execution of the first instruction in the interrupt handling routine. The symbols used in table 6.8 are explained in table 6.9.

Table 6.8 Interrupt Response Times

| No. | Number of States | Advanced Mode |
|------------------------------|--|-----------------------|
| 1 | Interrupt priority determination ^{*1} | 3 |
| 2 | Number of wait states until executing instruction ends ^{*2} | 1 to $19+2 \cdot S_i$ |
| 3 | PC, CCR stack save | $2 \cdot S_k$ |
| 4 | Vector fetch | $2 \cdot S_i$ |
| 5 | Instruction fetch ^{*3} | $2 \cdot S_i$ |
| 6 | Internal processing ^{*4} | 2 |
| Total (using on-chip memory) | | 12 to 32 |

Notes: 1. Two states in case of internal interrupt.

2. Refers to MULXS and DIVXS instructions.

3. Prefetch after interrupt acceptance and interrupt handling routine prefetch.

4. Internal processing after interrupt acceptance and internal processing after vector fetch.

Table 6.9 Number of States in Interrupt Handling Routine Execution

| Symbol | Object of Access |
|------------------------|------------------|
| | Internal Memory |
| Instruction fetch SI | 1 |
| Branch address read SJ | |
| Stack manipulation SK | |

6.5 Usage Notes

6.5.1 Contention between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to disable interrupts, the disabling becomes effective after execution of the instruction.

In other words, when an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, and so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request of higher priority than that interrupt, interrupt exception handling will be executed for the higher-priority interrupt, and the lower-priority interrupt will be ignored.

The same also applies when an interrupt source flag is cleared to 0.

Figure 6.9 shows an example in which the OCIAE bit in timer X1 TIER is cleared to 0.

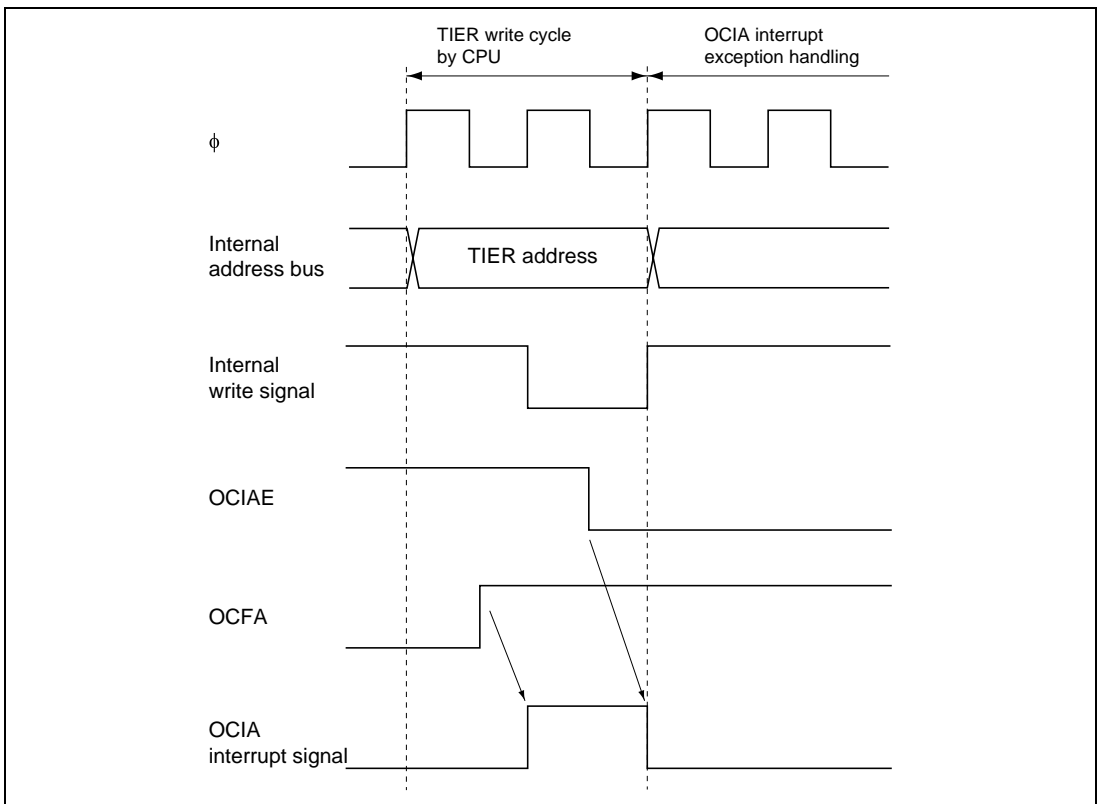


Figure 6.9 Contention between Interrupt Generation and Disabling

The above contention will not occur if an enable bit or interrupt source flag is cleared to 0 while the interrupt is masked.

6.5.2 Instructions that Disable Interrupts

Instructions that disable interrupts are LDC, ANDC, ORC, and XORC. After any of these instructions is executed, all interrupts except NMI are disabled and the next instruction is always executed. When the I bit or UI bit is set by one of these instructions, the new value becomes valid two states after execution of the instruction ends.

6.5.3 Interrupts during Execution of EEPMOV Instruction

Interrupt operation differs between the EEPMOV.B instruction and the EEPMOV.W instruction. With the EEPMOV.B instruction, an interrupt request (including NMI) issued during the transfer is not accepted until the move is completed.

With the EEPMOV.W instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at a break in the transfer cycle. The PC value saved on the stack in this case is the address of the next instruction.

Therefore, if an interrupt is generated during execution of an EEPMOV.W instruction, the following coding should be used.

```
L1: EEPMOV.W
    MOV.W   R4, R4
    BNE     L1
```

6.5.4 When NMI is Disabled

When NMI is disabled, the input level to the $\overline{\text{NMI}}$ pin must be fixed high or low. It is recommended that the NMI interrupt exception handling address be set to the NMI vector address (H'00001C to H'00001F) and that the RTE instruction also be set to the NMI exception handling address.

<Program Example>

```
.ORG      H'00001C
.DATA.L   NMI
.
.
.
.
NMI: RTE
```


Section 7 ROM (H8S/2194 Series)

7.1 Overview

The H8S/2194 has 128 kbytes of on-chip ROM (flash memory or mask ROM), the H8S/2193 has 112 kbytes, the H8S/2192 has 96 kbytes, and the H8S/2191 has 80 kbytes. The ROM is connected to the CPU by a 16-bit data bus. The CPU accesses both byte and word data in one state, enabling faster instruction fetches and higher processing speed.

The flash memory versions of the H8S/2194 can be erased and programmed on-board as well as with a general-purpose PROM programmer.

7.1.1 Block Diagram

Figure 7.1 shows a block diagram of the ROM.

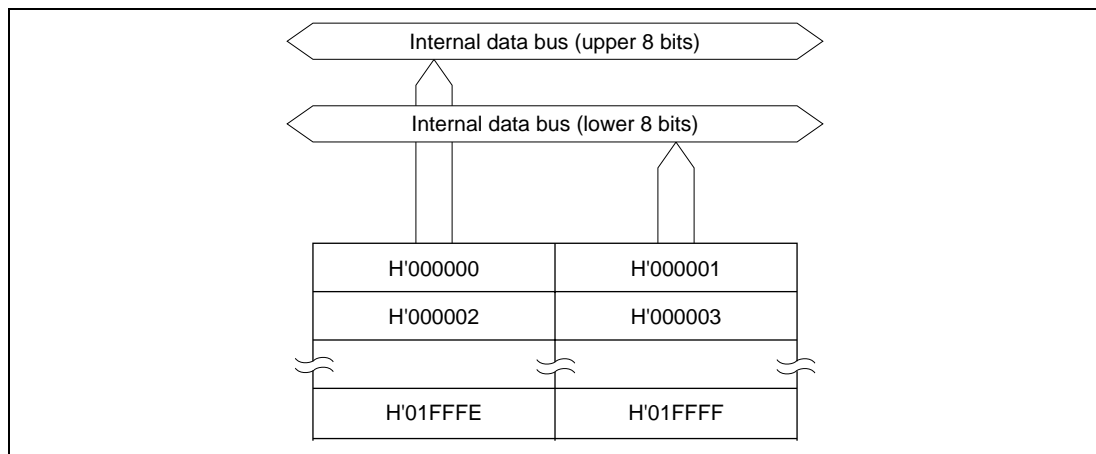


Figure 7.1 ROM Block Diagram (H8S/2194)

7.2 Overview of Flash Memory

7.2.1 Features

The features of the flash memory are summarized below.

- Four flash memory operating modes
 - Program mode
 - Erase mode
 - Program-verify mode
 - Erase-verify mode
- Programming/erase methods

The flash memory is programmed 32 bytes at a time. Erasing is performed by block erase (in single-block units). When erasing all blocks, the individual blocks must be erased sequentially. Block erasing can be performed as required on 1-kbyte, 8-kbyte, 16-kbyte, 28-kbyte, and 32-kbyte blocks.
- Programming/erase times

The flash memory programming time is 10 ms (typ.) for simultaneous 32-byte programming, equivalent to 300 μ s (typ.) per byte, and the erase time is 100 ms (typ.) per block.
- Reprogramming capability

The flash memory can be reprogrammed up to 100 times.
- On-board programming modes

There are two modes in which flash memory can be programmed/erased/verified on-board:

 - Boot mode
 - User program mode
- Automatic bit rate adjustment

If data transfer on boot mode, automatic adjustment is possible at host transfer bit rates and MCU's bit rates.
- Protect modes

There are three protect modes, hardware, software, and error protect, which allow protected status to be designated for flash memory program/erase/verify operations.
- Programmer mode

Flash memory can be programmed/erased in programmer mode, using a PROM programmer, as well as in on-board programming mode.

7.2.2 Block Diagram

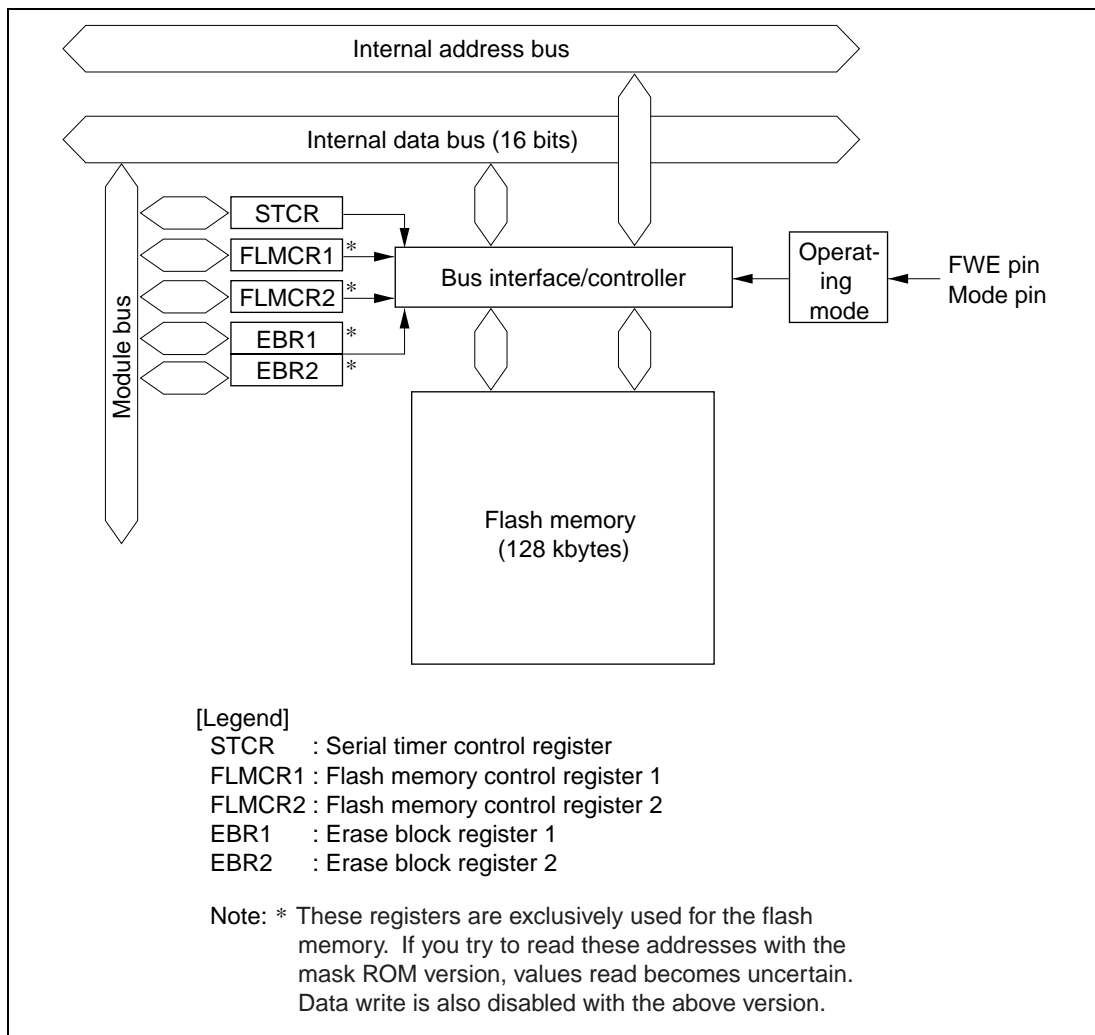


Figure 7.2 Block Diagram of Flash Memory

7.2.3 Flash Memory Operating Modes

(1) Mode Transitions

When each mode pin and the FWE pin are set in the reset state and a reset-start is executed, the MCU enters one of the operating modes shown in figure 7.3. In user mode, flash memory can be read but not programmed or erased.

Flash memory can be programmed and erased in boot mode, user program mode, and programmer mode.

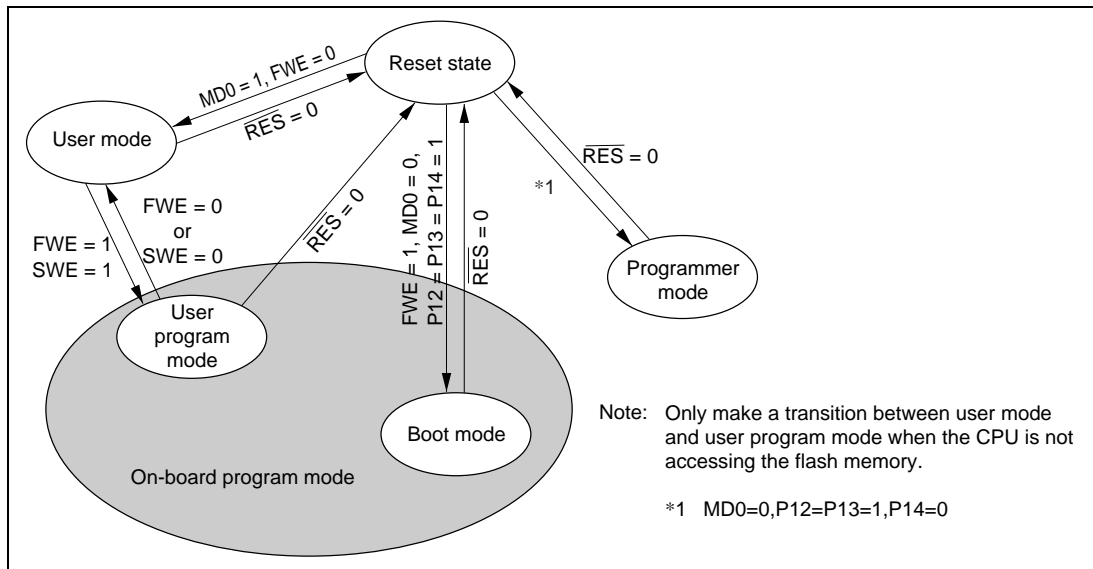


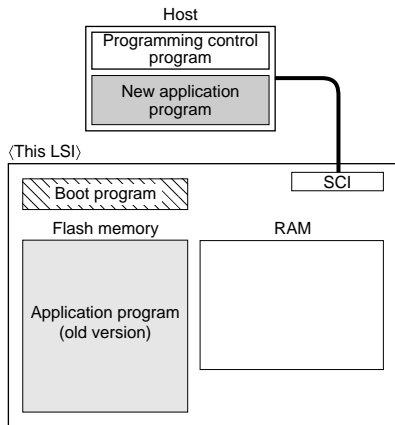
Figure 7.3 Flash Memory Mode Transitions

(2) On-Board Programming Modes

(a) Boot mode

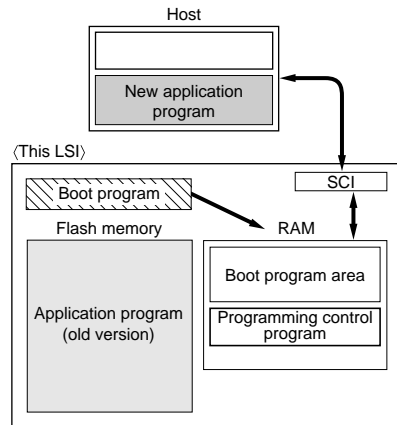
1. Initial state

The old program version or data remains written in the flash memory. The user should prepare the programming control program and new application program beforehand in the host.



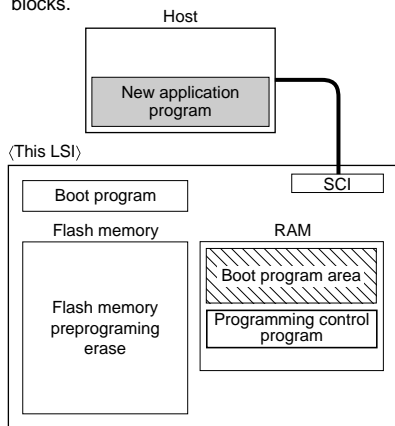
2. Programming control program transfer

When boot mode is entered, the boot program in the LSI (originally incorporated in the chip) is started and the programming control program in the host is transferred to RAM via SCI communication. The boot program required for flash memory erasing is automatically transferred to the RAM boot program area.



3. Flash memory initialization

The erase program in the boot program area (in RAM) is executed, and the flash memory is initialized (to H'FF). In boot mode, entire flash memory erasure is performed, without regard to blocks.



4. Writing new application program

The programming control program transferred from the host to RAM is executed, and the new application program in the host is written into the flash memory.

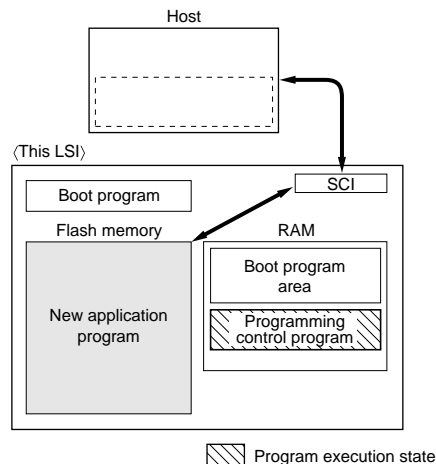
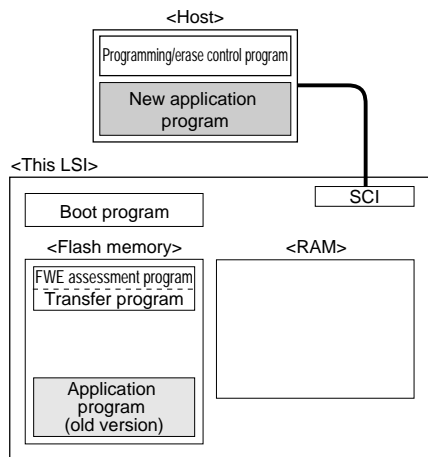


Figure 7.4 Boot Mode

(b) User program mode

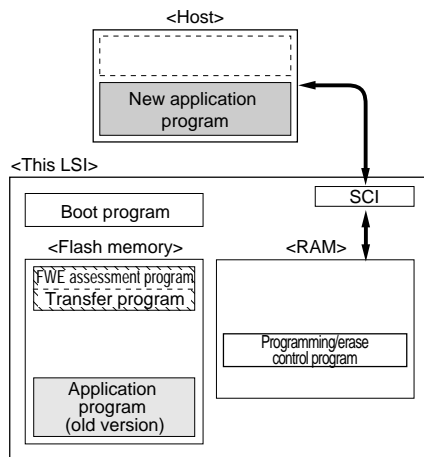
1. Initial state

(1) The FWE assessment program that confirms that the FWE pin has been driven high, and (2) the program that will transfer the programming/erase control program from the flash memory to on-chip RAM should be written into the flash memory by the user beforehand. (3) The programming/erase control program should be prepared in the host or in the flash memory.



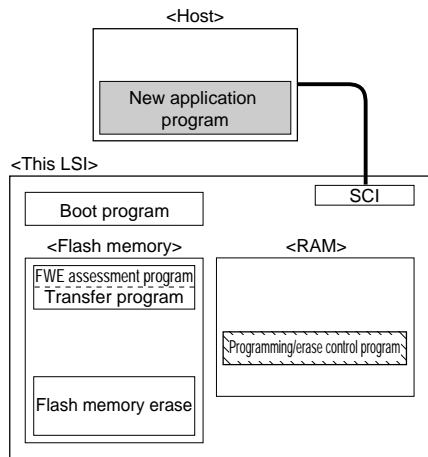
2. Programming/erase control program transfer

When user program mode is entered, user software confirms this fact, executes the transfer program in the flash memory, and transfers the programming/erase control program to RAM.



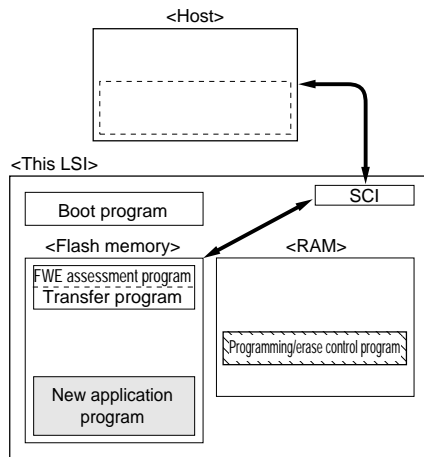
3. Flash memory initialization

The programming/erase control program in RAM is executed, and the flash memory is initialized (to H'FF). Erasing can be performed in block units, but not in byte units.



4. Writing new application program

Next, the new application program in the host is written into the erased flash memory blocks. Do not write to unerased blocks.



Program execution state

Figure 7.5 User Program Mode (Example)

(3) Differences between Boot Mode and User Program Mode

Table 7.1 Differences between Boot Mode and User Program Mode

| | Boot Mode | User Program Mode |
|------------------------------|------------------------|--|
| Entire memory erase | Yes | Yes |
| Block erase | No | Yes |
| Programming control program* | Program/program-verify | Erase/erase-verify Program/program-verify |

Note: * To be provided by the user, in accordance with the recommended algorithm.

(4) Block Configuration

The flash memory is divided into two 32-kbyte blocks, two 8-kbyte blocks, one 16-kbyte block, one 28-kbyte block, and four 1-kbyte blocks.

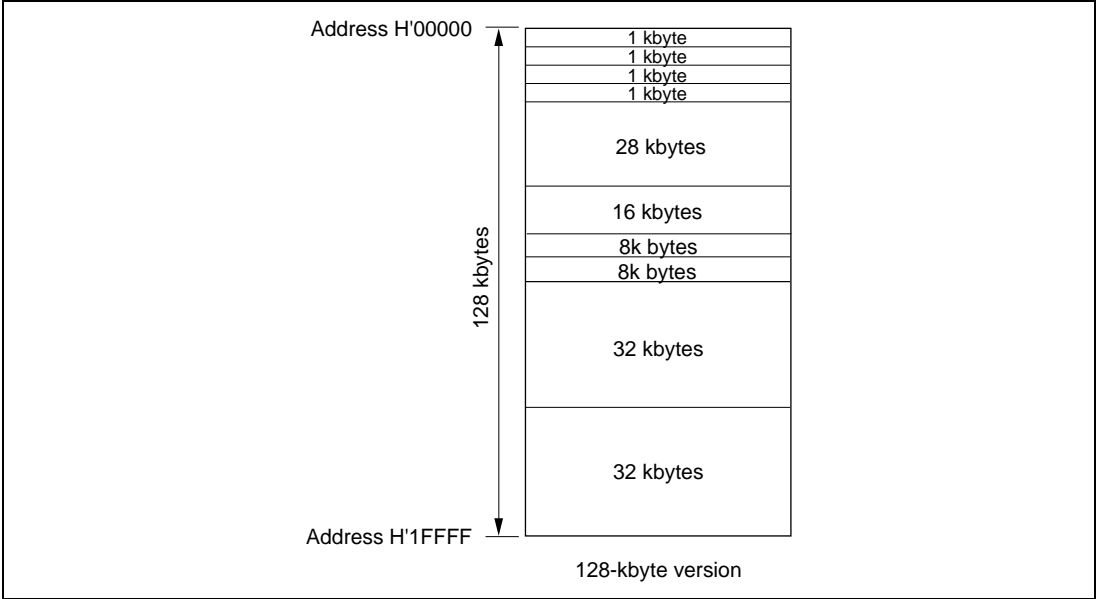


Figure 7.6 Flash Memory Block Configuration

7.2.4 Pin Configuration

The flash memory is controlled by means of the pins shown in table 7.2.

Table 7.2 Flash Memory Pins

| Pin Name | Abbreviation | I/O | Function |
|--------------------|-------------------------|--------|--|
| Reset | $\overline{\text{RES}}$ | Input | Reset |
| Flash write enable | FWE | Input | Flash program/erase protection by hardware |
| Mode 0 | MD0 | Input | Sets this LSI operating mode |
| Port 12 | P12 | Input | Sets this LSI operating mode when MD0 = 0 |
| Port 13 | P13 | Input | Sets this LSI operating mode when MD0 = 0 |
| Port 14 | P14 | Input | Sets this LSI operating mode when MD0 = 0 |
| Transmit data | SO1 | Output | Serial transmit data output |
| Receive data | SI1 | Input | Serial receive data input |

7.2.5 Register Configuration

The registers used to control the on-chip flash memory when enabled are shown in table 7.3. In order to access these registers, the FLSHE bit in STCR must be set to 1.

Table 7.3 Flash Memory Registers

| Register Name | Abbreviation | R/W | Initial Value | Address ^{*5} | Access Size |
|---------------------------------|----------------------|-------------------|--------------------|-----------------------|-------------|
| Flash memory control register 1 | FLMCR1 ^{*4} | R/W ^{*1} | H'00 ^{*2} | H'FFF8 | 8 |
| Flash memory control register 2 | FLMCR2 ^{*4} | R/W ^{*1} | H'00 ^{*3} | H'FFF9 | 8 |
| Erase block register 1 | EBR1 ^{*4} | R/W ^{*1} | H'00 ^{*3} | H'FFFA | 8 |
| Erase block register 2 | EBR2 ^{*4} | R/W ^{*1} | H'00 ^{*3} | H'FFFB | 8 |
| Serial timer control register | STCR | R/W | H'00 | H'FFEE | 8 |

- Notes: 1. When the FWE bit in FLMCR1 is not set at 1, writes are disabled.
2. When a high level is input to the FWE pin, the initial value is H'80.
3. When a low level is input to the FWE pin, or if a high level is input and the SWE bit in FLMCR1 is not set, these registers are initialized to H'00.
4. FLMCR1, FLMCR2, EBR1, and EBR2 are 8-bit registers. Only byte accesses are valid for these registers, the access requiring 2 states.
5. Lower 16 bits of the address.

7.3 Flash Memory Register Descriptions

7.3.1 Flash Memory Control Register 1 (FLMCR1)

| | | | | | | | | |
|-----------------|-----|-----|---|---|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FWE | SWE | — | — | EV | PV | E | P |
| Initial value : | —* | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R/W | — | — | R/W | R/W | R/W | R/W |

Note: * Determined by the state of the FWE pin.

FLMCR1 is an 8-bit register used for flash memory operating mode control. Program-verify mode or erase-verify mode is entered by setting SWE to 1 when FWE = 1. Program mode is entered by setting SWE to 1 when FWE = 1, then setting the PSU bit in FLMCR2, and finally setting the P bit. Erase mode is entered by setting SWE to 1 when FWE = 1, then setting the ESU bit in FLMCR2, and finally setting the E bit. FLMCR1 is initialized by a reset, in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), or when a low level is input to the FWE pin. Its initial value is H'80 when a high level is input to the FWE pin, and H'00 when a low level is input. When on-chip flash memory is disabled, a read will return H'00, and writes are invalid.

Writes to the SWE bit in FLMCR1 are enabled only when FWE = 1; writes to the EV and PV bits only when FWE=1 and SWE=1; writes to the E bit only when FWE = 1, SWE = 1, and ESU = 1; and writes to the P bit only when FWE = 1, SWE = 1, and PSU = 1.

Bit 7: Flash Write Enable (FWE)

Sets hardware protection against flash memory programming/erasing.

Bit 7

| FWE | Description |
|-----|---|
| 0 | When a low level is input to the FWE pin (hardware-protected state) |
| 1 | When a high level is input to the FWE pin |

Bit 6: Software Write Enable (SWE)

Enables or disables flash memory programming. SWE should be set before setting bits ESU, PSU, EV, PV, E, P, and EB9 to EB0, and should not be cleared at the same time as these bits.

Bit 6

| SWE | Description |
|-----|---|
| 0 | Writes are disabled (Initial value) |
| 1 | Writes are enabled [Setting condition] Setting is available when FWE = 1 is selected |

Bit 5 and 4: Reserved

These bits cannot be modified and are always read as 0.

Bit 3: Erase-Verify (EV)

Selects erase-verify mode transition or clearing. Do not set the SWE, ESU, PSU, PV, E, or P bit at the same time.

Bit 3

| EV | Description |
|----|---|
| 0 | Erase-verify mode cleared (Initial value) |
| 1 | Transition to erase-verify mode [Setting condition] Setting is available when FWE = 1 and SWE = 1 are selected |

Bit 2: Program-Verify (PV)

Selects program-verify mode transition or clearing. Do not set the SWE, ESU, PSU, EV, E, or P bit at the same time.

Bit 2

| PV | Description |
|----|---|
| 0 | Program-verify mode cleared (Initial value) |
| 1 | Transition to program-verify mode [Setting condition] Setting is available when FWE = 1 and SWE = 1 are selected |

Bit 1: Erase (E)

Selects erase mode transition or clearing. Do not set the SWE, ESU, PSU, EV, PV, or P bit at the same time.

Bit 1

| E | Description |
|----------|--|
| 0 | Erase mode cleared (Initial value) |
| 1 | Transition to erase mode [Setting condition] Setting is available when FWE = 1, SWE = 1, and ESU = 1 are selected |

Bit 0: Program (P)

Selects program mode transition or clearing. Do not set the SWE, PSU, ESU, EV, PV, or E bit at the same time.

Bit 0

| P | Description |
|----------|--|
| 0 | Program mode cleared (Initial value) |
| 1 | Transition to program mode [Setting condition] Setting is available when FWE = 1, SWE = 1, and PSU = 1 are selected |

7.3.2 Flash Memory Control Register 2 (FLMCR2)

| | | | | | | | | |
|-----------------|------|---|---|---|---|---|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FLER | — | — | — | — | — | ESU | PSU |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | — | — | — | — | — | R/W | R/W |

FLMCR2 is an 8-bit register that monitors the presence or absence of flash memory program/erase protection (error protection) and performs setup for flash memory program/erase mode. FLMCR2 is initialized to H'00 by a reset. The ESU and PSU bits are cleared to 0 in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), hardware protect mode, or software protect mode.

Bit 7: Flash Memory Error (FLER)

Indicates that an error has occurred during an operation on flash memory (programming or erasing). When FLER is set to 1, flash memory goes to the error-protection state.

Bit 7

| FLER | Description |
|------|---|
| 0 | Flash memory is operating normally Flash memory program/erase protection (error protection) is disabled [Clearing condition] Reset or hardware standby mode (Initial value) |
| 1 | An error has occurred during flash memory programming/erasing Flash memory program/erase protection (error protection) is enabled [Setting condition] See section 7.6.3, Error Protection |

Bits 6 to 2: Reserved

These bits cannot be modified and are always read as 0.

Bit 1: Erase Setup (ESU)

Prepares for a transition to erase mode. Set this bit to 1 before setting the E bit to 1 in FLMCR1. Do not set the SWE, PSU, EV, PV, E, or P bit at the same time.

Bit 1

| ESU | Description |
|-----|--|
| 0 | Erase setup cleared (Initial value) |
| 1 | Erase setup [Setting condition] When FWE = 1, and SWE = 1 |

Bit 0: Program Setup (PSU)

Prepares for a transition to program mode. Set this bit to 1 before setting the P bit to 1 in FLMCR1. Do not set the SWE, ESU, EV, PV, E, or P bit at the same time.

Bit 0

| PSU | Description |
|-----|--|
| 0 | Program setup cleared (Initial value) |
| 1 | Program setup [Setting condition] When FWE = 1, and SWE = 1 |

7.3.3 Erase Block Registers 1 and 2 (EBR1, EBR2)

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EBR1 : | — | — | — | — | — | — | EB9 | EB8 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | — | — | R/W | R/W |

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EBR2 : | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

EBR1 and EBR2 are registers that specify the flash memory erase area block by block; bits 1 and 0 in EBR1 (128-kbyte versions only) and bits 7 to 0 in EBR2 are readable/writable bits. EBR1 and EBR2 are each initialized to H'00 by a reset, in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), when a low level is input to the FWE pin, or when a high level is input to the FWE pin and the SWE bit in FLMCR1 is not set. When a bit in EBR1 or EBR2 is set, the corresponding block can be erased. Other blocks are erase-protected. Set only one bit in EBR1 or EBR2 (more than one bit cannot be set).

The flash memory block configuration is shown in table 7.4.

Table 7.4 Flash Memory Erase Blocks

| Block (Size) | |
|---------------------------|----------------------|
| 128-kbyte Versions | Address |
| EB0 (1 kbyte) | H'000000 to H'0003FF |
| EB1 (1 kbyte) | H'000400 to H'0007FF |
| EB2 (1 kbyte) | H'000800 to H'000BFF |
| EB3 (1 kbyte) | H'000C00 to H'000FFF |
| EB4 (28 kbytes) | H'001000 to H'007FFF |
| EB5 (16 kbytes) | H'008000 to H'00BFFF |
| EB6 (8 kbytes) | H'00C000 to H'00DFFF |
| EB7 (8 kbytes) | H'00E000 to H'00FFFF |
| EB8 (32 kbytes) | H'010000 to H'017FFF |
| EB9 (32 kbytes) | H'018000 to H'01FFFF |

7.3.4 Serial/Timer Control Register (STCR)

| | | | | | | | | | |
|---------------|---|---|------|--------|---|-------|---|---|---|
| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | — | IICX | IICRST | — | FLSHE | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | R/W | R/W | — | R/W | — | — | — |

STCR is an 8-bit readable/writable register that controls register access, the I²C bus interface operating mode, and on-chip flash memory (in F-ZTAT versions), and also selects the I²C bus interface serial clock frequency. For details on functions not related to on-chip flash memory, see section 25.2.7, Serial/Timer Control Register (STCR), and descriptions of individual modules. If a module controlled by STCR is not used, do not write 1 to the corresponding bit. STCR is initialized to H'00 by a reset.

Bits 6, 5: I²C Control (IICX, IICRST)

These bits control the operation of the I²C bus interface. For details, see section 24, I²C Bus Interface.

Bit 3: Flash Memory Control Register Enable (FLSHE)

Setting the FLSHE bit to 1 enables read/write access to the flash memory control registers. If FLSHE is cleared to 0, the flash memory control registers are deselected. In this case, the flash memory control register contents are retained.

Bit 3

| FLSHE | Description |
|-------|---|
| 0 | Flash memory control registers deselected (Initial value) |
| 1 | Flash memory control registers selected |

Bits 7, 4 and 2 to 0: Reserved

7.4 On-Board Programming Modes

When pins are set to on-board programming mode, program/erase/verify operations can be performed on the on-chip flash memory. There are two on-board programming modes: boot mode and user program mode. The pin settings for transition to each of these modes are shown in table 7.5. For a diagram of the transitions to the various flash memory modes, see figure 7.3.

Table 7.5 Setting On-Board Programming Modes

| Mode | Pin | | | | |
|-------------------|-----------------|-----|-----------------|-----------------|-----------------|
| Mode Name | FWE | MD0 | P12 | P13 | P14 |
| Boot mode | 1 | 0 | 1 ^{*2} | 1 ^{*2} | 1 ^{*2} |
| User program mode | 1 ^{*1} | 1 | — | — | — |

Notes: 1. In user program mode, the FWE pin should not be constantly set to 1. Set FWE to 1 to make a transition to user program mode before performing a program/erase/verify operation.

2. Can be used as I/O ports after boot mode is initiated.

7.4.1 Boot Mode

When boot mode is used, the flash memory programming control program must be prepared in the host beforehand. The channel 1 SCI to be used is set to asynchronous mode.

When a reset-start is executed after the MCU's pins have been set to boot mode, the boot program built into the MCU is started and the programming control program prepared in the host is serially transmitted to the MCU via the SCI1. In the MCU, the programming control program received via the SCI1 is written into the programming control program area in on-chip RAM. After the transfer is completed, control branches to the start address of the programming control program area and the programming control program execution state is entered (flash memory programming is performed).

The transferred programming control program must therefore include coding that follows the programming algorithm given later.

The system configuration in boot mode is shown in figure 7.7, and the boot program mode execution procedure in figure 7.8.

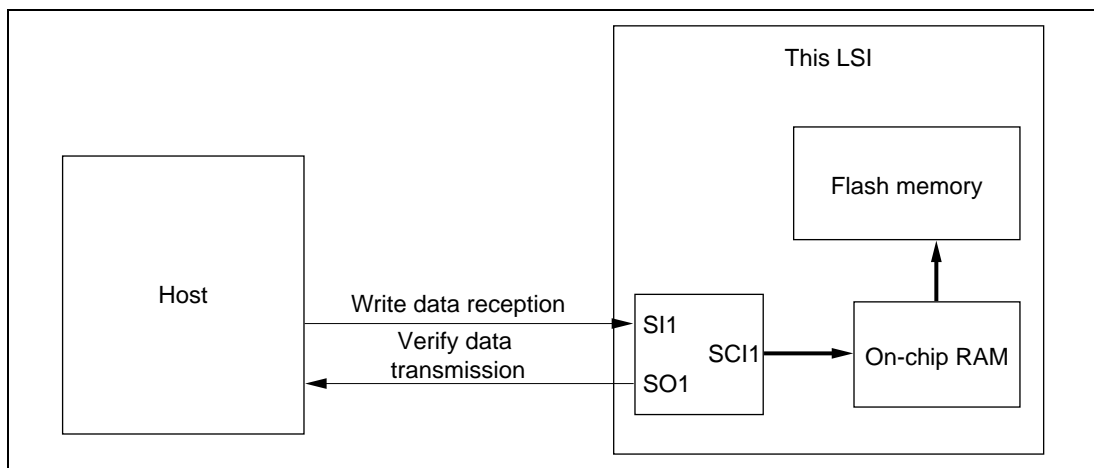
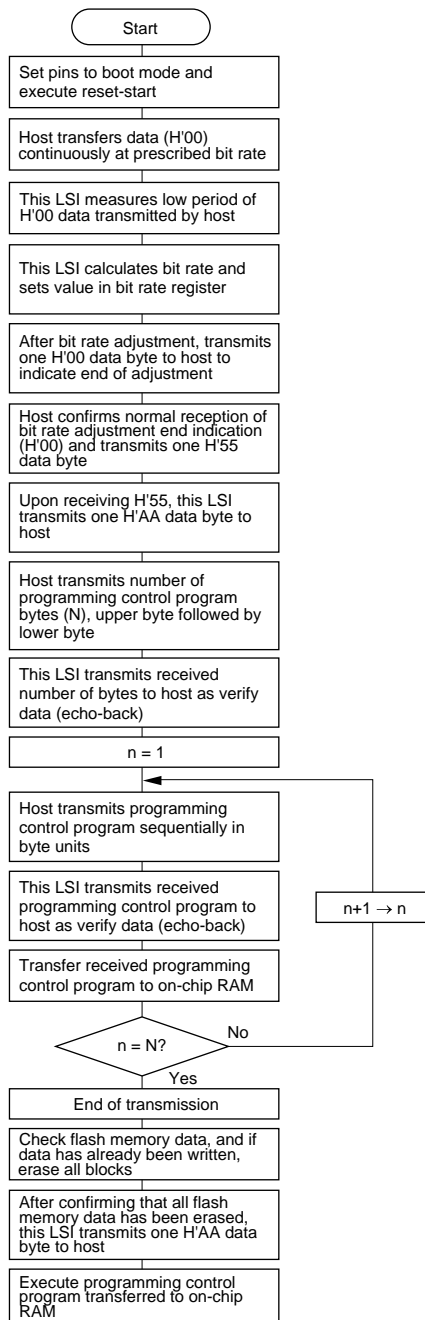


Figure 7.7 System Configuration in Boot Mode



Note : If a memory cell does not operate normally and cannot be erased, one H'FF byte is transmitted as an erase error, and the erase operation and subsequent operations are halted.

Figure 7.8 Boot Mode Execution Procedure

(1) Automatic SCI Bit Rate Adjustment

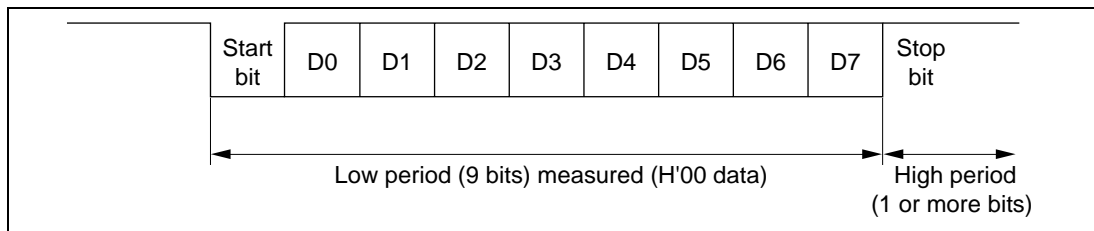


Figure 7.9 Automatic SCI Bit Rate Adjustment

When boot mode is initiated, the MCU measures the low period of the asynchronous SCI communication data (H'00) transmitted continuously from the host. The SCI transmit/receive format should be set as follows: 8-bit data, 1 stop bit, no parity. The MCU calculates the bit rate of the transmission from the host from the measured low period, and transmits one H'00 byte to the host to indicate the end of bit rate adjustment. The host should confirm that this adjustment end indication (H'00) has been received normally, and transmit one H'55 byte to the MCU. If reception cannot be performed normally, initiate boot mode again (reset), and repeat the above operations. Depending on the host's transmission bit rate and the MCU's system clock frequency, there will be a discrepancy between the bit rates of the host and the MCU. To ensure correct SCI operation, the host's transfer bit rate should be set to (2400, 4800, or 9600) bps. Table 7.6 shows typical host transfer bit rates and system clock frequencies for which automatic adjustment of the MCU's bit rate is possible. The boot program should be executed within this system clock range.

Table 7.6 System Clock Frequencies for which Automatic Adjustment of This LSI Bit Rate is Possible

| Host Bit Rate | System Clock Frequency for which Automatic Adjustment of This LSI Bit Rate is Possible |
|---------------|--|
| 9600 bps | 8 MHz to 10 MHz |
| 4800 bps | 4 MHz to 10 MHz |

(2) On-Chip RAM Area Divisions in Boot Mode

In boot mode, the 2048-byte area from H'FFEFB0 to H'FFF7AF is reserved for use by the boot program, as shown in figure 7.10. The area to which the programming control program is transferred is H'FFF7B0 to H'FFFF2F (1920 bytes). The boot program area can be used when the programming control program transferred into RAM enters the execution state. A stack area should be set up as required.

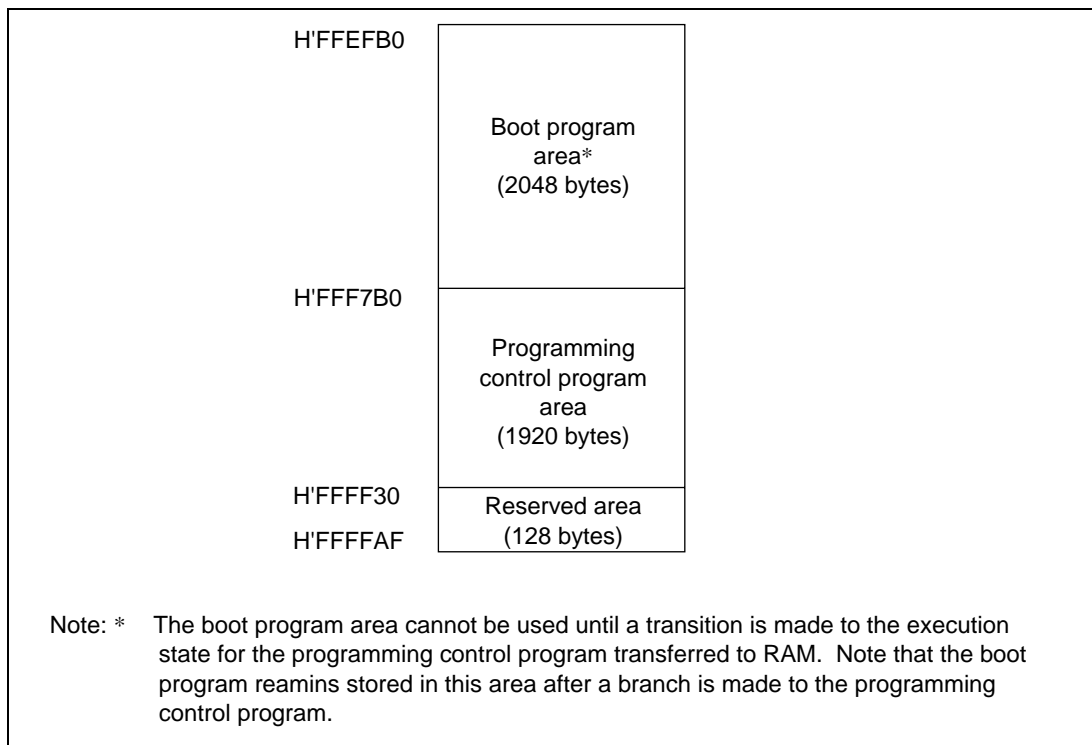


Figure 7.10 RAM Areas in Boot Mode

(3) Notes on Use of Boot Mode:

- (a) When reset is released in boot mode, it measures the low period of the input at the SCI1's SI1 pin. The reset should end with SI1 pin high. After the reset ends, it takes about 100 states for the chip to get ready to measure the low period of the SI1 pin input.
- (b) In boot mode, if any data has been programmed into the flash memory (if all data is not 1), all flash memory blocks are erased. Boot mode is for use when user program mode is unavailable, such as the first time on-board programming is performed, or if the program activated in user program mode is accidentally erased.
- (c) Interrupts cannot be used while the flash memory is being programmed or erased.
- (d) The SI1 and SO1 pins should be pulled up on the board.
- (e) Before branching to the programming control program (RAM area H'FFF3B0), the chip terminates transmit and receive operations by the on-chip SCI (channel 1) (by clearing the RE and TE bits in SCR to 0), but the adjusted bit rate value remains set in BRR. The transmit data output pin, SO1, goes to the high-level output state (P21PCR = 1, P21PDR = 1).

The contents of the CPU's internal general registers are undefined at this time, so these registers must be initialized immediately after branching to the programming control program. In particular, since the stack pointer (SP) is used implicitly in subroutine calls, etc., a stack area must be specified for use by the programming control program.

The initial values of other on-chip registers are not changed.

- (f) Boot mode can be entered by making the pin settings shown in table 7.5 and executing a reset-start.

When the chip detects the boot mode setting at reset release^{*1}, it retains that state internally.

Boot mode can be cleared by driving the reset pin low, waiting at least 20 states, then setting the FWE pin and mode pins, and executing reset release^{*1}. Boot mode can also be cleared by a WDT overflow reset.

If the mode pin input levels are changed in boot mode, the boot mode state will be maintained in the microcomputer, and boot mode continued, unless a reset occurs.

However, the FWE pin must not be driven low while the boot program is running or flash memory is being programmed or erased^{*2}.

- Notes: 1. Mode pin and FWE pin input must satisfy the mode programming setup time ($t_{\text{MDS}} = 4$ states) with respect to the reset release timing.
2. For further information on FWE application and disconnection, see section 7.9, Flash Memory Programming and Erasing Precautions.

7.4.2 User Program Mode

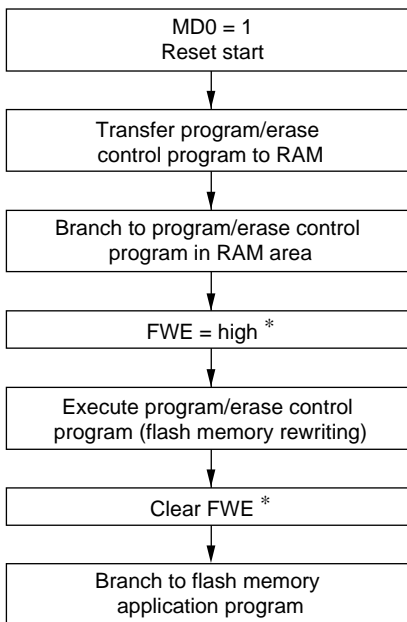
When set to user program mode, the chip can program and erase its flash memory by executing a user program/erase control program. Therefore, on-board reprogramming of the on-chip flash memory can be carried out by providing on-board means of FWE control and supply of programming data, and storing a program/erase control program in part of the program area as necessary.

In this mode, the chip starts up in mode 1 and applies a high level to the FWE pin.

The flash memory itself cannot be read while the SWE bit is set to 1 to perform programming or erasing, so the control program that performs programming and erasing should be run in on-chip RAM or external memory.

Figure 7.11 shows the procedure for executing the program/erase control program when transferred to on-chip RAM.

Write the FWE assessment program and transfer program (and the program/erase control program if necessary) beforehand



Note: Do not apply a constant high level to the FWE pin. Apply a high level to the FWE pin only when the flash memory is programmed or erased. Also, while a high level is applied to the FWE pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc.

* For further information on FWE application and disconnection, see section 7.9, Flash Memory Programming and Erasing Precautions.

Figure 7.11 User Program Mode Execution Procedure

7.5 Programming/Erasing Flash Memory

In the on-board programming modes, flash memory programming and erasing is performed by software, using the CPU. There are four flash memory operating modes: program mode, erase mode, program-verify mode, and erase-verify mode. Transitions to these modes can be made by setting the PSU and ESU bits in FLMCR2, and the P, E, PV, and EV bits in FLMCR1.

The flash memory cannot be read while being programmed or erased. Therefore, the program that controls flash memory programming/erasing (the programming control program) should be located and executed in on-chip RAM or external memory.

- Notes:
1. Operation is not guaranteed if setting/resetting of the SWE, EV, PV, E, and P bits in FLMCR1, and the ESU and PSU bits in FLMCR2, is executed by a program in flash memory.
 2. When programming or erasing, set FWE to 1 (programming/erasing will not be executed if FWE = 0).
 3. Perform programming in the erased state. Do not perform additional programming on previously programmed addresses.

7.5.1 Program Mode

Follow the procedure shown in the program/program-verify flowchart in figure 7.12 to write data or programs to flash memory. Performing program operations according to this flowchart will enable data or programs to be written to flash memory without subjecting the device to voltage stress or sacrificing program data reliability. Programming should be carried out 32 bytes at a time.

Table 29.9 in section 29.2.7, Flash Memory Characteristics, lists wait time (x , y , z , α , β , γ , ϵ and η) after setting or clearing each bit on the flash memory control registers 1 and 2 (FLMCR1 and FLMCR2) and the maximum write count (N).

Following the elapse of (x) μ s or more after the SWE bit is set to 1 in flash memory control register 1 (FLMCR1), 32-byte program data is stored in the program data area and reprogram data area, and the 32-byte data in the reprogram data area written consecutively to the write addresses. The lower 8 bits of the first address written to must be H'00, H'20, H'40, H'60, H'80, H'A0, H'C0, or H'E0. Thirty-two consecutive byte data transfers are performed. The program address and program data are latched in the flash memory. A 32-byte data transfer must be performed even if writing fewer than 32 bytes; in this case, H'FF data must be written to the extra addresses.

Next, the watchdog timer is set to prevent overprogramming in the event of program runaway, etc. Set more than ($y + z + \alpha + \beta$) μ s as the WDT overflow period. After this, preparation for program mode (program setup) is carried out by setting the PSU bit in FLMCR2, and after the elapse of (y) μ s or more, the operating mode is switched to program mode by setting the P bit in FLMCR1. The time during which the P bit is set is the flash memory programming time. Make a program setting so that the time for one programming operation is within the range of (z) μ s.

7.5.2 Program-Verify Mode

In program-verify mode, the data written in program mode is read to check whether it has been correctly written in the flash memory.

After the elapse of a given programming time, the programming mode is exited (the P bit in FLMCR1 is cleared, then the PSU bit in FLMCR2 is cleared at least (α) μ s later). The watchdog timer is cleared after the elapse of (β) μ s or more, and the operating mode is switched to program-verify mode by setting the PV bit in FLMCR1. Before reading in program-verify mode, a dummy write of H'FF data should be made to the addresses to be read. The dummy write should be executed after the elapse of (γ) μ s or more. When the flash memory is read in this state (verify data is read in 16-bit units), the data at the latched address is read. Wait at least (ϵ) μ s after the dummy write before performing this read operation. Next, the originally written data is compared with the verify data, and reprogram data is computed (see figure 7.12) and transferred to the reprogram data area. After 32 bytes of data have been verified, exit program-verify mode, wait for at least (η) μ s, then clear the SWE bit in FLMCR1. If reprogramming is necessary, set program mode again, and repeat the program/program-verify sequence as before. However, ensure that the program/program-verify sequence is not repeated more than (N) times on the same bits.

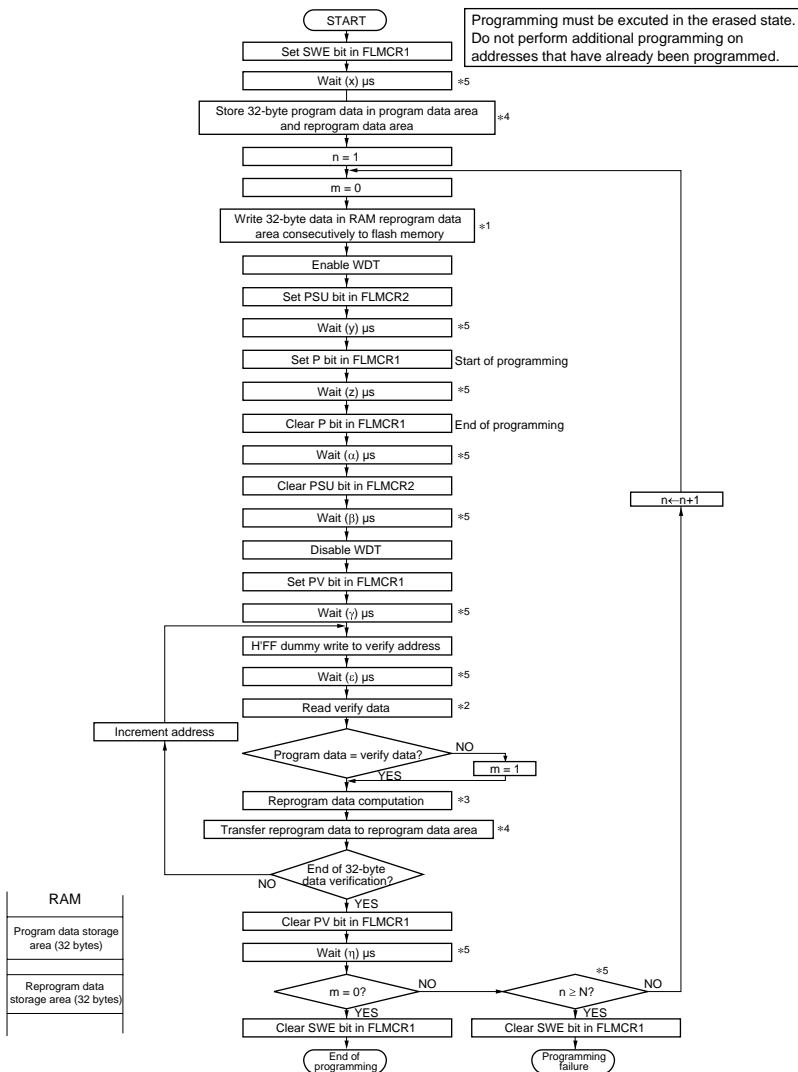


Figure 7.12 Program/Program-Verify Flowchart

7.5.3 Erase Mode

Flash memory erasing should be performed block by block following the procedure shown in the erase/erase-verify flowchart (single-block erase) shown in figure 7.13.

Table 28.9 in section 28.2.7, Flash Memory Characteristics lists wait time (x , y , z , α , β , γ , ε and η) after setting or clearing each bit on the flash memory control registers 1 and 2 (FLMCR1 and FLMCR2) and the maximum clearing count (N).

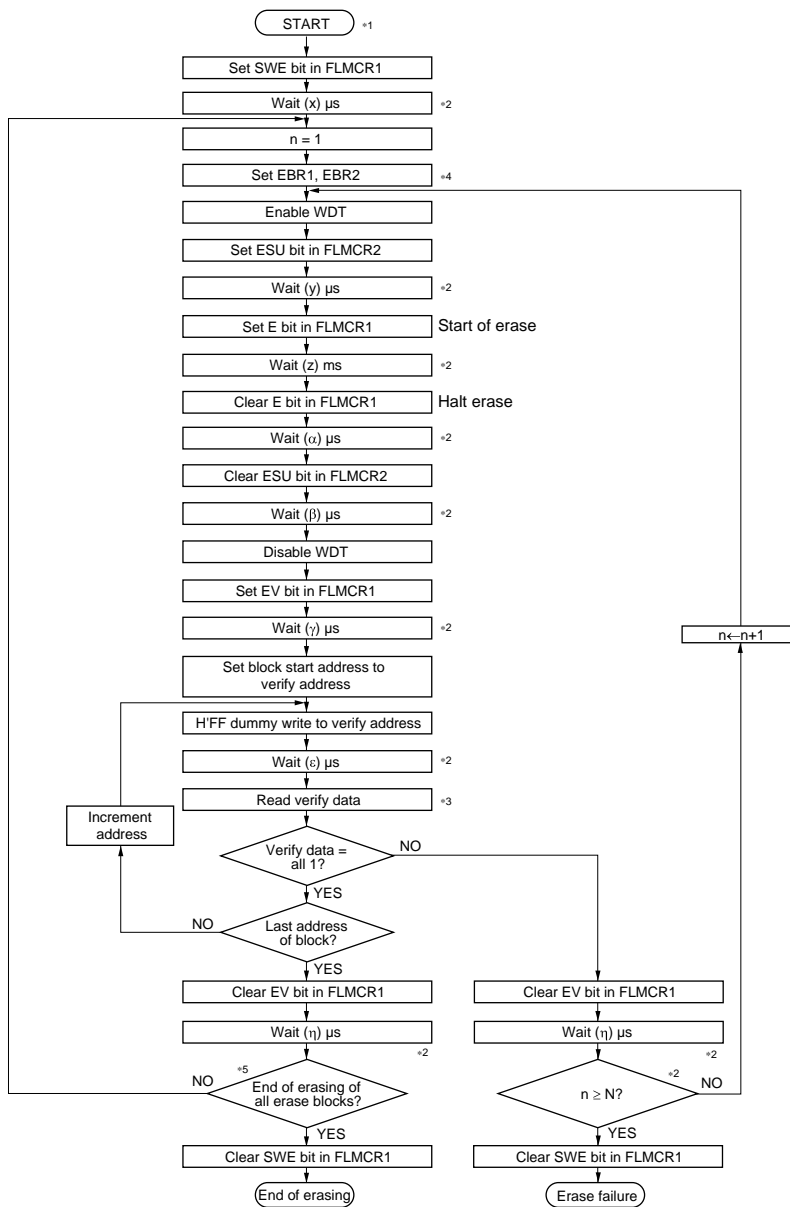
To perform data or program erasure, make a 1 bit setting for the flash memory area to be erased in erase block register 1 or 2 (EBR1 or EBR2) at least (x) μ s after setting the SWE bit to 1 in flash memory control register 1 (FLMCR1). Next, the watchdog timer is set to prevent overerasing in the event of program runaway, etc. Set more than ($y + z + \alpha + \beta$) ms as the WDT overflow period. After this, preparation for erase mode (erase setup) is carried out by setting the ESU bit in FLMCR2, and after the elapse of (y) μ s or more, the operating mode is switched to erase mode by setting the E bit in FLMCR1. The time during which the E bit is set is the flash memory erase time. Ensure that the erase time does not exceed (z) ms.

Note: With flash memory erasing, preprogramming (setting all data in the memory to be erased to 0) is not necessary before starting the erase procedure.

7.5.4 Erase-Verify Mode

In erase-verify mode, data is read after memory has been erased to check whether it has been correctly erased.

After the elapse of the erase time, erase mode is exited (the E bit in FLMCR1 is cleared, then the ESU bit in FLMCR2 is cleared at least (α) μ s later), the watchdog timer is cleared after the elapse of (β) μ s or more, and the operating mode is switched to erase-verify mode by setting the EV bit in FLMCR1. Before reading in erase-verify mode, a dummy write of H'FF data should be made to the addresses to be read. The dummy write should be executed after the elapse of (γ) μ s or more. When the flash memory is read in this state (verify data is read in 16-bit units), the data at the latched address is read. Wait at least (ε) μ s after the dummy write before performing this read operation. If the read data has been erased (all 1), a dummy write is performed to the next address, and erase-verify is performed. If the read data has not been erased, set erase mode again, and repeat the erase/erase-verify sequence in the same way. However, ensure that the erase/erase-verify sequence is not repeated more than (N) times. When verification is completed, exit erase-verify mode, and wait for at least (η) μ s. If erasure has been completed on all the erase blocks, clear the SWE bit in FLMCR1. If there are any unerased blocks, make a 1 bit setting in EBR1 or EBR2 for the flash memory area to be erased, and repeat the erase/erase-verify sequence in the same way.



- Notes:
1. Preprogramming (setting erase block data to all 0) is not necessary.
 2. The values of x, y, z, α, β, γ, ε, η and N are listed in section 29.2.7, Flash Memory Characteristics.
 3. Verify data is read in 16-bit (word) units.
 4. Set only one bit in EBR1 or EBR2. More than one bit cannot be set.
 5. Erasing is performed in block units. To erase a number of blocks, the individual blocks must be erased sequentially.

Figure 7.13 Erase/Erase-Verify Flowchart (Single-Block Erase)

7.6 Flash Memory Protection

There are three kinds of flash memory program/erase protection: hardware protection, software protection, and error protection.

7.6.1 Hardware Protection

Hardware protection refers to a state in which programming/erasing of flash memory is forcibly disabled or aborted. Hardware protection is reset by settings in flash memory control registers 1 and 2 (FLMCR1, FLMCR2) and erase block registers 1 and 2 (EBR1, EBR2). In error protection mode, FLMCR1, FLMCR2, EBR1 and EBR2 settings are retained. (See table 7.7.)

Table 7.7 Hardware Protection

| Item | Description | Functions | |
|--------------------------|--|-----------|-------|
| | | Program | Erase |
| FWE pin protection | <ul style="list-style-type: none">When a low level is input to the FWE pin, FLMCR1, FLMCR2, EBR1, and EBR2 are initialized, and the program/erase-protected state is entered | Yes | Yes |
| Reset/standby protection | <ul style="list-style-type: none">In a reset (including a WDT overflow reset) and in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), FLMCR1, FLMCR2 (excluding the FLER bit), EBR1, and EBR2 are initialized, and the program/erase-protected state is enteredIn a reset via the $\overline{\text{RES}}$ pin, the reset state is not entered unless the $\overline{\text{RES}}$ pin is held low until oscillation stabilizes after powering on. In the case of a reset during operation, hold the $\overline{\text{RES}}$ pin low for the $\overline{\text{RES}}$ pulse width specified in the AC characteristics section | Yes | Yes |

7.6.2 Software Protection

Software protection can be implemented by setting the SWE bit in FLMCR1 and erase block registers 1 and 2 (EBR1, EBR2). When software protection is in effect, setting the P or E bit in flash memory control register 1 (FLMCR1) does not cause a transition to program mode or erase mode. (See table 7.8.)

Table 7.8 Software Protection

| Item | Description | Functions | |
|--------------------------------|--|-----------|-------|
| | | Program | Erase |
| SWE bit protection | <ul style="list-style-type: none">Clearing the SWE bit to 0 in FLMCR1 sets the program/erase-protected state for all blocks (Execute in on-chip RAM or external memory) | Yes | Yes |
| Block specification protection | <ul style="list-style-type: none">Erase protection can be set for individual blocks by settings in erase block registers 1 and 2 (EBR1, EBR2)Setting EBR1 and EBR2 to H'00 places all blocks in the erase-protected state | – | Yes |

7.6.3 Error Protection

In error protection, an error is detected when MCU runaway occurs during flash memory programming/erasing, or operation is not performed in accordance with the program/erase algorithm, and the program/erase operation is aborted. Aborting the program/erase operation prevents damage to the flash memory due to overprogramming or overerasing.

If the MCU malfunctions during flash memory programming/erasing, the FLER bit is set to 1 in FLMCR2 and the error protection state is entered. The FLMCR1, FLMCR2, EBR1, and EBR2 settings are retained, but program mode or erase mode is aborted at the point at which the error occurred. Program mode or erase mode cannot be re-entered by re-setting the P or E bit. However, PV and EV bit setting is enabled, and a transition can be made to verify mode. FLER bit setting conditions are as follows:

- (1) When flash memory is read during programming/erasing (including a vector read or instruction fetch)
- (2) Immediately after exception handling (excluding a reset) during programming/erasing
- (3) When a SLEEP instruction is executed during programming/erasing

Error protection is released only by a reset and in hardware standby mode.

Figure 7.14 shows the flash memory state transition diagram.

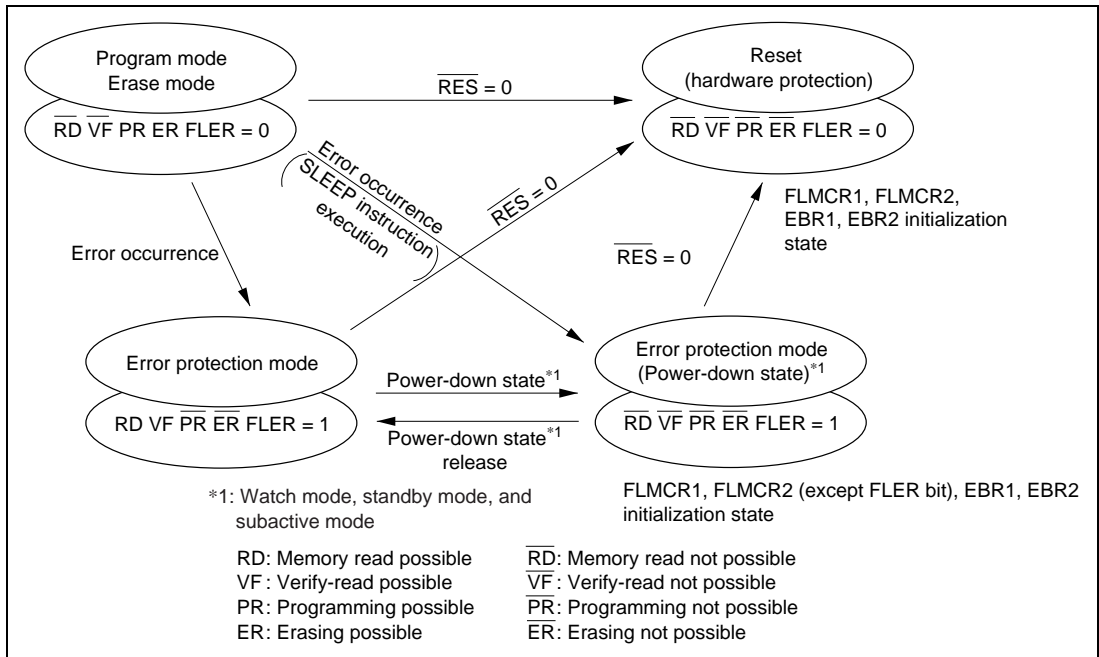


Figure 7.14 Flash Memory State Transitions

7.7 Interrupt Handling when Programming/Erasing Flash Memory

All interrupts, including NMI input is disabled when flash memory is being programmed or erased (when the P or E bit is set in FLMCR1), and while the boot program is executing in boot mode^{*1}, to give priority to the program or erase operation. There are three reasons for this:

- (1) Interrupt during programming or erasing might cause a violation of the programming or erasing algorithm, with the result that normal operation could not be assured.
- (2) In the interrupt exception handling sequence during programming or erasing, the vector would not be read correctly^{*2}, possibly resulting in MCU runaway.
- (3) If interrupt occurred during boot program execution, it would not be possible to execute the normal boot mode sequence.

For these reasons, in on-board programming mode alone there are conditions for disabling interrupt, as an exception to the general rule. However, this provision does not guarantee normal erasing and programming or MCU operation. All requests, including NMI input, must therefore be disabled inside and outside the MCU during FWE application. Interrupt is also disabled in the error-protection state while the P or E bit remains set in FLMCR1.

- Notes:
1. Interrupt requests must be disabled inside and outside the MCU until data write by the write control program is complete.
 2. The vector may not be read correctly in this case for the following two reasons:
 - If flash memory is read while being programmed or erased (while the P or E bit is set in FLMCR1), correct read data will not be obtained (undetermined values will be returned).
 - If the interrupt entry in the interrupt vector table has not been programmed yet, interrupt exception handling will not be executed correctly.

7.8 Flash Memory Programmer Mode

7.8.1 Programmer Mode Setting

Programs and data can be written and erased in programmer mode as well as in the on-board programming modes. In programmer mode, the on-chip ROM can be freely programmed using a PROM programmer that supports Hitachi microcomputer device type with 128-kbyte on-chip flash memory. Flash memory read mode, auto-program mode, auto-erase mode, and status read mode are supported with these device types. In auto-program mode, auto-erase mode, and status read mode, a status polling procedure is used, and in status read mode, detailed internal signals are output after execution of an auto-program or auto-erase operation.

7.8.2 Socket Adapters and Memory Map

In programmer mode, a socket adapter is mounted on the writer programmer. The socket adapter product codes are listed in table 7.9.

Figure 7.15 shows the memory map in programmer mode.

Table 7.9 Socket Adapter Product Codes

| Product Codes | Package | Socket Adapter Product Code |
|---------------|-------------|-----------------------------|
| HD64F2194 | 112-pin QFP | ME2194ESHF1H |

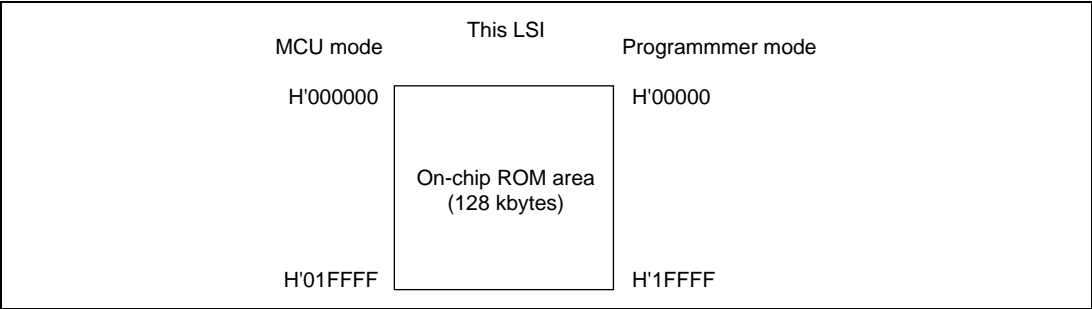


Figure 7.15 Memory Map in Programmer Mode

7.8.3 Programmer Mode Operation

Table 7.10 shows how the different operating modes are set when using programmer mode, and table 7.11 lists the commands used in programmer mode. Details of each mode are given below.

(1) Memory Read Mode

Memory read mode supports byte reads.

(2) Auto-Program Mode

Auto-program mode supports programming of 128 bytes at a time. Status polling is used to confirm the end of auto-programming.

(3) Auto-Erase Mode

Auto-erase mode supports automatic erasing of the entire flash memory. Status polling is used to confirm the end of auto-erasing.

(4) Status Read Mode

Status polling is used for auto-programming and auto-erasing, and normal termination can be confirmed by reading the FO6 signal. In status read mode, error information is output if an error occurs.

Table 7.10 Settings for Each Operating Mode in Programmer Mode

| Mode | Pin Names | | | | | |
|----------------------------|----------------------|------------------------|------------------------|------------------------|-------------|-------------------|
| | FWE | $\overline{\text{CE}}$ | $\overline{\text{OE}}$ | $\overline{\text{WE}}$ | FO0 to FO7 | FA0 to FA17 |
| Read | H or L | L | L | H | Data output | Ain |
| Output disable | H or L | L | H | H | Hi-z | X |
| Command write | H or L ^{*3} | L | H | L | Data input | Ain ^{*2} |
| Chip disable ^{*1} | H or L | L | X | X | Hi-z | X |

Notes: 1. Chip disable is not a standby state; internally, it is an operation state.

2. Ain indicates that there is also address input in auto-program mode.

3. For command writes when making a transition to auto-program or auto-erase mode, input a high level to the FWE pin.

Table 7.11 Programmer Mode Commands

| Command Name | Number of Cycles | 1st Cycle | | | 2nd Cycle | | |
|-------------------|------------------|-----------|---------|------|-----------|---------|------|
| | | Mode | Address | Data | Mode | Address | Data |
| Memory read mode | 1+n | write | X | H'00 | read | RA | Dout |
| Auto-program mode | 129 | write | X | H'40 | write | WA | Din |
| Auto-erase mode | 2 | write | X | H'20 | write | X | H'20 |
| Status read mode | 2 | write | X | H'71 | write | X | H'71 |

Notes: 1. In auto-program mode. 129 cycles are required for command writing by a simultaneous 128-byte write.

2. In memory read mode, the number of cycles depends on the number of address write cycles (n).

7.8.4 Memory Read Mode

- (1) After the end of an auto-program, auto-erase, or status read operation, the command wait state is entered. To read memory contents, a transition must be made to memory read mode by means of a command write before the read is executed.
- (2) Command writes can be performed in memory read mode, just as in the command wait state.
- (3) Once memory read mode has been entered, consecutive reads can be performed.
- (4) After power-on, memory read mode is entered.

Table 7.12 AC Characteristics in Memory Read Mode

(Conditions: $V_{CC} = 5.0 \text{ V} \pm 10\%$, $V_{SS} = 0 \text{ V}$, $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit |
|-----------------------------------|------------|-----|-----|---------------|
| Command write cycle | t_{nxtc} | 20 | — | μs |
| $\overline{\text{CE}}$ hold time | t_{ceh} | 0 | — | ns |
| $\overline{\text{CE}}$ setup time | t_{ces} | 0 | — | ns |
| Data hold time | t_{dh} | 50 | — | ns |
| Data setup time | t_{ds} | 50 | — | ns |
| Write pulse width | t_{wep} | 70 | — | ns |
| $\overline{\text{WE}}$ rise time | t_r | — | 30 | ns |
| $\overline{\text{WE}}$ fall time | t_f | — | 30 | ns |

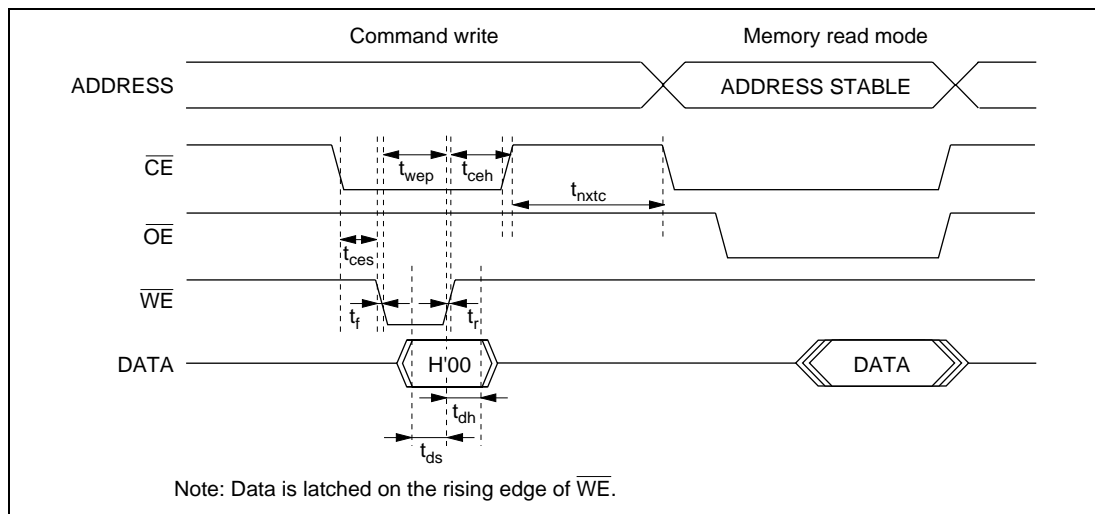


Figure 7.16 Memory Read Mode Timing Waveforms after Command Write

Table 7.13 AC Characteristics when Entering Another Mode from Memory Read Mode(Conditions: $V_{CC} = 5.0 \text{ V} \pm 10\%$, $V_{SS} = 0 \text{ V}$, $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit |
|-----------------------------------|------------|-----|-----|---------------|
| Command write cycle | t_{nxtc} | 20 | — | μs |
| $\overline{\text{CE}}$ hold time | t_{ceh} | 0 | — | ns |
| $\overline{\text{CE}}$ setup time | t_{ces} | 0 | — | ns |
| Data hold time | t_{dh} | 50 | — | ns |
| Data setup time | t_{ds} | 50 | — | ns |
| Write pulse width | t_{wep} | 70 | — | ns |
| $\overline{\text{WE}}$ rise time | t_r | — | 30 | ns |
| $\overline{\text{WE}}$ fall time | t_f | — | 30 | ns |

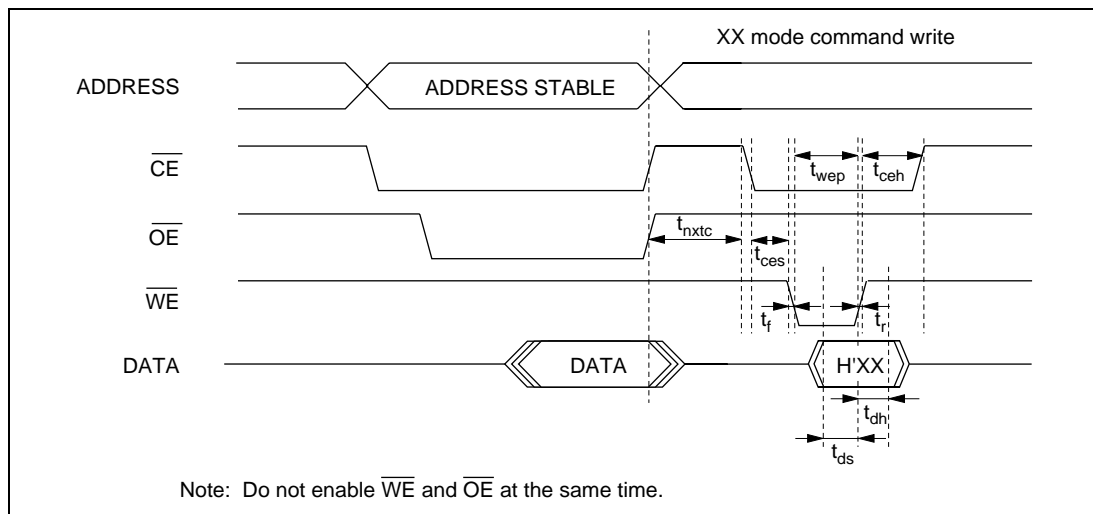
**Figure 7.17 Timing Waveforms when Entering Another Mode from Memory Read Mode**

Table 7.14 AC Characteristics in Memory Read Mode

(Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $V_{SS} = 0\text{ V}$, $T_a = 25^{\circ}\text{C} \pm 5^{\circ}\text{C}$)

| Item | Symbol | Min | Max | Unit |
|--|-----------|-----|-----|---------------|
| Access time | t_{acc} | — | 20 | μs |
| $\overline{\text{CE}}$ output delay time | t_{ce} | — | 150 | ns |
| $\overline{\text{OE}}$ output delay time | t_{oe} | — | 150 | ns |
| Output disable delay time | t_{df} | — | 100 | ns |
| Data output hold time | t_{oh} | 5 | — | ns |

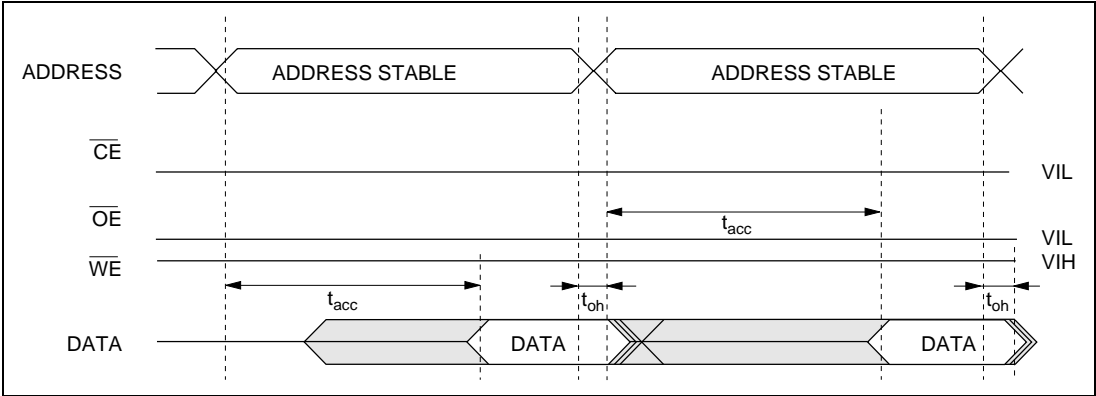


Figure 7.18 Timing Waveforms for $\overline{\text{CE}}/\overline{\text{OE}}$ Enable State Read

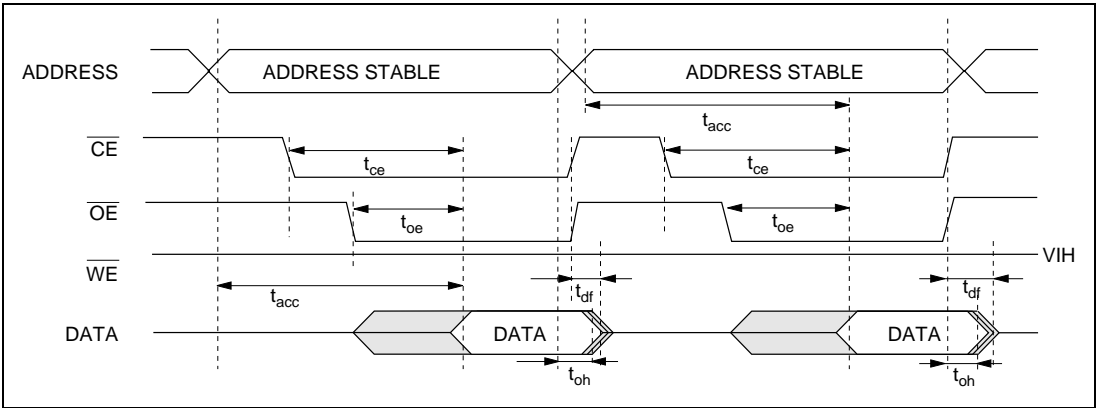


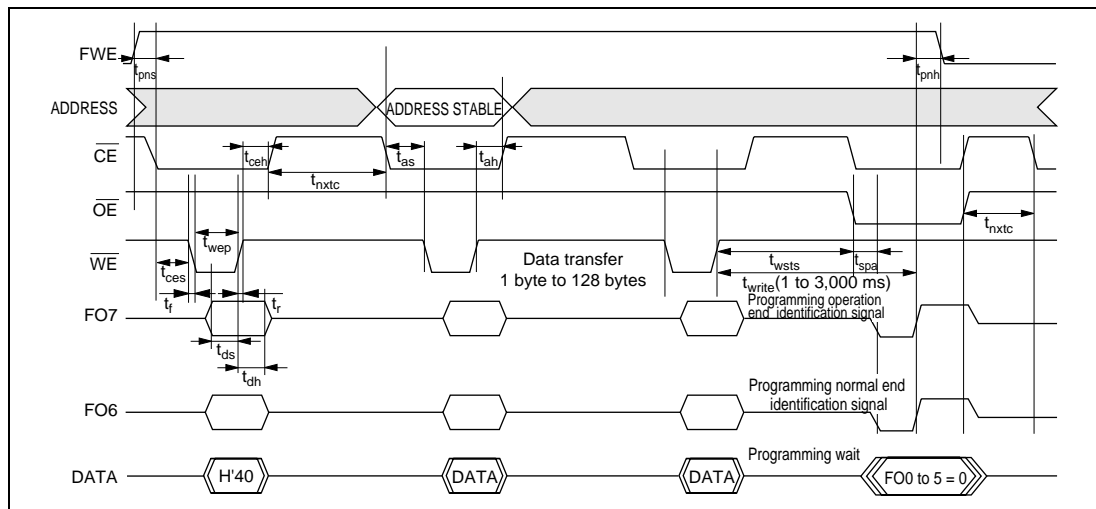
Figure 7.19 Timing Waveforms for $\overline{\text{CE}}/\overline{\text{OE}}$ Clocked Read

7.8.5 Auto-Program Mode

- (a) In auto-program mode, 128 bytes are programmed simultaneously. This should be carried out by executing 128 consecutive byte transfers.
- (b) A 128-byte data transfer is necessary even when programming fewer than 128 bytes. In this case, H'FF data must be written to the extra addresses.
- (c) The lower 8 bits of the transfer address must be H'00 or H'80. If a value other than an effective address is input, processing will switch to a memory write operation but a write error will be flagged.
- (d) Memory address transfer is performed in the second cycle (figure 7.20). Do not perform transfer after the second cycle.
- (e) Do not perform a command write during a programming operation.
- (f) Perform one auto-programming operation for a 128-byte block for each address. Characteristics are not guaranteed for two or more programming operations.
- (g) Confirm normal end of auto-programming by checking FO6. Alternatively, status read mode can also be used for this purpose (FO7 status polling uses the auto-program operation end identification pin).
- (h) The status polling FO6 and FO7 pin information is retained until the next command write. Until the next command write is performed, reading is possible by enabling $\overline{\text{CE}}$ and $\overline{\text{OE}}$.

Table 7.15 AC Characteristics in Auto-Program(Conditions: $V_{CC} = 5.0 \text{ V} \pm 10\%$, $V_{SS} = 0 \text{ V}$, $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit |
|-----------------------------------|-------------|-----|------|---------------|
| Command write cycle | t_{nxtc} | 20 | — | μs |
| $\overline{\text{CE}}$ hold time | t_{ceh} | 0 | — | ns |
| $\overline{\text{CE}}$ setup time | t_{ces} | 0 | — | ns |
| Data hold time | t_{dh} | 50 | — | ns |
| Data setup time | t_{ds} | 50 | — | ns |
| Write pulse width | t_{wep} | 70 | — | ns |
| Status polling start time | t_{wsts} | 1 | — | ms |
| Status polling access time | t_{spa} | — | 150 | ns |
| Address setup time | t_{as} | 0 | — | ns |
| Address hold time | t_{ah} | 60 | — | ns |
| Memory write time | t_{write} | 1 | 3000 | ms |
| $\overline{\text{WE}}$ rise time | t_r | — | 30 | ns |
| $\overline{\text{WE}}$ fall time | t_f | — | 30 | ns |
| Write setup time | t_{pns} | 100 | — | ns |
| Write end setup time | t_{pnh} | 100 | — | ns |

**Figure 7.20 Auto-Program Mode Timing Waveforms**

7.8.6 Auto-Erase Mode

- (a) Auto-erase mode supports only entire memory erasing.
- (b) Do not perform a command write during auto-erasing.
- (c) Confirm normal end of auto-erasing by checking FO6. Alternatively, status read mode can also be used for this purpose (FO7 status polling uses the auto-erase operation end identification pin).
- (d) The status polling FO6 and FO7 pin information is retained until the next command write. Until the next command write is performed, reading is possible by enabling \overline{CE} and \overline{OE} .

Table 7.16 AC Characteristics in Auto-Erase Mode

(Conditions: $V_{CC} = 5.0 \text{ V} \pm 10\%$, $V_{SS} = 0 \text{ V}$, $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit |
|----------------------------|-------------|-----|-------|---------------|
| Command write cycle | t_{nxtc} | 20 | — | μs |
| \overline{CE} hold time | t_{ceh} | 0 | — | ns |
| \overline{CE} setup time | t_{ces} | 0 | — | ns |
| Data hold time | t_{dh} | 50 | — | ns |
| Data setup time | t_{ds} | 50 | — | ns |
| Write pulse width | t_{wep} | 70 | — | ns |
| Status polling start time | t_{ests} | 1 | — | ms |
| Status polling access time | t_{spa} | — | 150 | ns |
| Memory erase time | t_{erase} | 100 | 40000 | ms |
| \overline{WE} rise time | t_r | — | 30 | ns |
| \overline{WE} fall time | t_f | — | 30 | ns |
| Erase setup time | t_{ens} | 100 | — | ns |
| Erase end setup time | t_{enh} | 100 | — | ns |

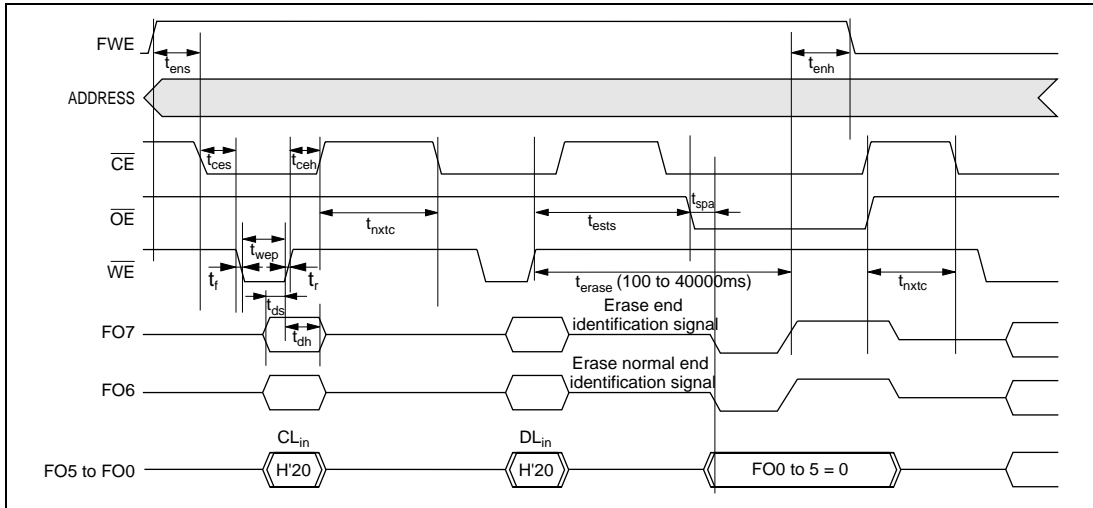


Figure 7.21 Auto-Erase Mode Timing Waveforms

7.8.7 Status Read Mode

- (1) Status read mode is used to identify what type of abnormal end has occurred. Use this mode when an abnormal end occurs in auto-program mode or auto-erase mode.
- (2) The return code is retained until a command write for other than status read mode is performed.

Table 7.17 AC Characteristics in Status Read Mode

(Conditions: $V_{CC} = 5.0 \text{ V} \pm 10\%$, $V_{SS} = 0 \text{ V}$, $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit |
|--|------------|-----|-----|---------------|
| Command write cycle | t_{nxtc} | 20 | — | μs |
| $\overline{\text{CE}}$ hold time | t_{ceh} | 0 | — | ns |
| $\overline{\text{CE}}$ setup time | t_{ces} | 0 | — | ns |
| Data hold time | t_{dh} | 50 | — | ns |
| Data setup time | t_{ds} | 50 | — | ns |
| Write pulse width | t_{wep} | 70 | — | ns |
| $\overline{\text{OE}}$ output delay time | t_{oe} | — | 150 | ns |
| Disable delay time | t_{df} | — | 100 | ns |
| $\overline{\text{CE}}$ output delay time | t_{ce} | — | 150 | ns |
| $\overline{\text{WE}}$ rise time | t_r | — | 30 | ns |
| $\overline{\text{WE}}$ fall time | t_f | — | 30 | ns |

7.8.8 Status Polling

- (1) The FO7 status polling flag indicates the operating status in auto-program or auto-erase mode.
- (2) The FO6 status polling flag indicates a normal or abnormal end in auto-program or auto-erase mode.

Table 7.19 Status Polling Output Truth Table

| Pin Names | Internal Operation in Progress | Abnormal End | — | Normal End |
|------------|-----------------------------------|--------------|---|------------|
| FO7 | 0 | 1 | 0 | 1 |
| FO6 | 0 | 0 | 1 | 1 |
| FO0 to FO5 | 0 | 0 | 0 | 0 |

7.8.9 Programmer Mode Transition Time

Commands cannot be accepted during the oscillation stabilization period or the programmer mode setup time. After the programmer mode setup time, a transition is made to memory read mode.

Table 7.20 Command Wait State Transition Time Specifications

| Item | Symbol | Min | Max | Unit |
|---|------------|-----|-----|------|
| Standby release (oscillation stabilization time) | t_{osc1} | 10 | — | ms |
| Programmer mode setup time | t_{bmV} | 10 | — | ms |
| V_{CC} hold time | t_{dwn} | 0 | — | ms |

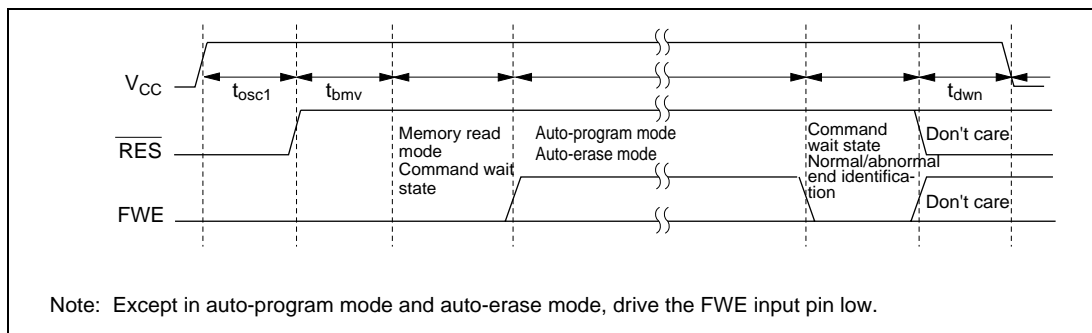


Figure 7.23 Oscillation Stabilization Time, Boot Program Transfer Time, and Power Supply Fall Sequence

7.8.10 Notes On Memory Programming

- (1) When programming addresses which have previously been programmed, carry out auto-erasing before auto-programming.
- (2) When performing programming using programmer mode on a chip that has been programmed/erased in an on-board programming mode, auto-erasing is recommended before carrying out auto-programming.

Notes: 1. The flash memory is initially in the erased state when the device is shipped by Hitachi. For other chips for which the erasure history is unknown, it is recommended that auto-erasing be executed to check and supplement the initialization (erase) level.

2. Auto-programming should be performed once only on the same address block.

7.9 Flash Memory Programming and Erasing Precautions

Precautions concerning the use of on-board programming mode and programmer mode are summarized below.

(1) Use the Specified Voltages and Timing for Programming and Erasing

Applied voltages in excess of the rating can permanently damage the device. Use a PROM programmer that supports Hitachi microcomputer device type with 128-kbyte on-chip flash memory.

Do not select the HN28F101 setting for the PROM programmer, and only use the specified socket adapter. Incorrect use will result in damaging the device.

(2) Powering On and Off

Do not apply a high level to the FWE pin until V_{CC} has stabilized. Also, drive the FWE pin low before turning off V_{CC} .

When applying or disconnecting V_{CC} , fix the FWE pin low and place the flash memory in the hardware protection state.

The power-on and power-off timing requirements should also be satisfied in the event of a power failure and subsequent recovery.

(3) FWE Application/Disconnection

FWE application should be carried out when MCU operation is in a stable condition. If MCU operation is not stable, fix the FWE pin low and set the protection state.

The following points must be observed concerning FWE application and disconnection to prevent unintentional programming or erasing of flash memory:

- (a) Apply FWE when the V_{CC} voltage has stabilized within its rated voltage range.
- (b) In boot mode, apply and disconnect FWE during a reset.
- (c) In user program mode, FWE can be switched between high and low level regardless of the reset state. FWE input can also be switched during program execution in flash memory.
- (d) Do not apply FWE if program runaway has occurred.
- (e) Disconnect FWE only when the SWE, ESU, PSU, EV, PV, P, and E bits in FLMCR1 and FLMCR2 are cleared.

Make sure that the SWE, ESU, PSU, EV, PV, P, and E bits are not set by mistake when applying or disconnecting FWE.

(4) Do Not Apply a Constant High Level to the FWE Pin

Apply a high level to the FWE pin only when programming or erasing flash memory. A system configuration in which a high level is constantly applied to the FWE should be avoided. Also, while a high level is applied to the FWE pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc.

(5) Use the Recommended Algorithm when Programming and Erasing Flash Memory

The recommended algorithm enables programming and erasing to be carried out without subjecting the device to voltage stress or sacrificing program data reliability. When setting the P or E bit in FLMCR1, the watchdog timer should be set beforehand as a precaution against program runaway, etc.

(6) Do Not Set or Clear the SWE Bit During Program Execution in Flash Memory

Clear the SWE bit before executing a program or reading data in flash memory.

When the SWE bit is set, data in flash memory can be rewritten, but flash memory should only be accessed for verify operations (verification during programming/erasing).

(7) Do Not Use Interrupts while Flash Memory is Being Programmed or Erased

All interrupt requests, including NMI, should be disabled during FWE application to give priority to program/erase operations.

(8) Do Not Perform Additional Programming. Erase the Memory before Reprogramming.

In on-board programming, perform only one programming operation on a 32-byte programming unit block. In programmer mode, too, perform only one programming operation on a 128-byte programming unit block. Programming should be carried out with the entire programming unit block erased.

(9) Before Programming, Check that the Chip is Correctly Mounted in the PROM Programmer.

Overcurrent damage to the device can result if the index marks on the PROM programmer socket, socket adapter, and chip are not correctly aligned.

(10) Do Not Touch the Socket Adapter or Chip During Programming.

Touching either of these can cause contact faults and write errors.

7.10 Note on Switching from F-ZTAT Version to Mask ROM Version

The mask ROM version does not have the internal registers for flash memory control that are provided in the F-ZTAT version. Table 7.21 lists the registers that are present in the F-ZTAT version but not in the mask ROM version. If a register listed in table 7.21 is read in the mask ROM version, an undefined value will be returned. Therefore, if application software developed on the F-ZTAT version is switched to a mask ROM version product, it must be modified to ensure that the registers in table 7.21 have no effect.

Table 7.21 Registers Present in F-ZTAT Version but Absent in Mask ROM Version

| Register | Abbreviation | Address |
|---------------------------------|--------------|---------|
| Flash memory control register 1 | FLMCR1 | H'FFF8 |
| Flash memory control register 2 | FLMCR2 | H'FFF9 |
| Erase block register 1 | EBR1 | H'FFFA |
| Erase block register 2 | EBR2 | H'FFFB |

Section 8 ROM (H8S/2194C Series)

8.1 Overview

The H8S/2194C has 256 kbytes of on-chip ROM (flash memory or mask ROM), the H8S/2194B has 192 kbytes, the H8S/2194A has 160 kbytes. The ROM is connected to the CPU by a 16-bit data bus. The CPU accesses both byte and word data in one state, enabling faster instruction fetches and higher processing speed.

The flash memory versions of the H8S/2194C can be erased and programmed on-board as well as with a general-purpose PROM programmer.

8.1.1 Block Diagram

Figure 8.1 shows a block diagram of the ROM.

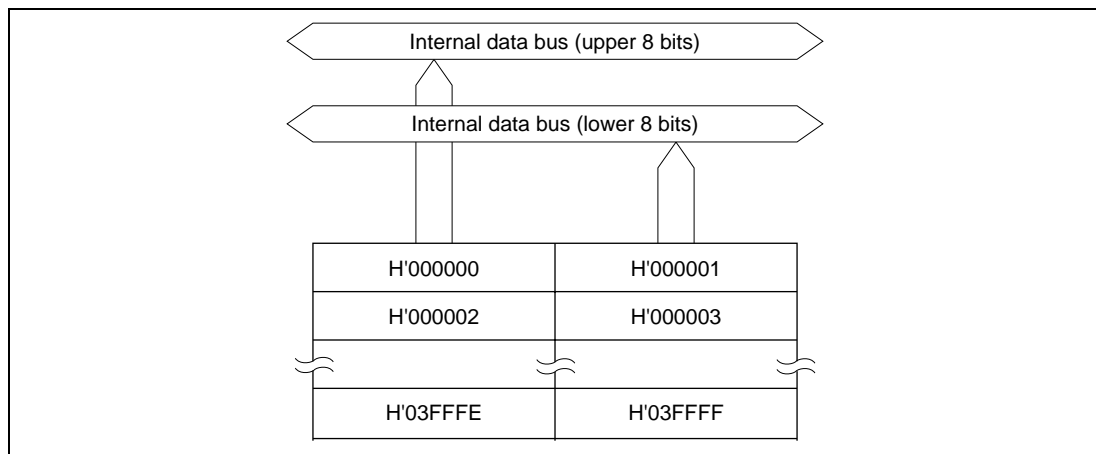


Figure 8.1 ROM Block Diagram (H8S/2194C)

8.2 Overview of Flash Memory

8.2.1 Features

The features of the flash memory are summarized below.

- Four flash memory operating modes
 - Program mode
 - Erase mode
 - Program-verify mode
 - Erase-verify mode
- Programming/erase methods

The flash memory is programmed 32 bytes at a time. Erasing is performed by block erase (in single-block units). When erasing all blocks, the individual blocks must be erased sequentially. Block erasing can be performed as required on 1-kbyte, 8-kbyte, 16-kbyte, 28-kbyte, and 32-kbyte blocks.
- Programming/erase times

The flash memory programming time is 10 ms (typ.) for simultaneous 32-byte programming, equivalent to 300 μ s (typ.) per byte, and the erase time is 100 ms (typ.) per block.
- Reprogramming capability

The flash memory can be reprogrammed up to 100 times.
- On-board programming modes

There are two modes in which flash memory can be programmed/erased/verified on-board:

 - Boot mode
 - User program mode
- Automatic bit rate adjustment

If data transfer on boot mode, automatic adjustment is possible at host transfer bit rates and MCU's bit rates.
- Protect modes

There are three protect modes, hardware, software, and error protect, which allow protected status to be designated for flash memory program/erase/verify operations.
- Programmer mode

Flash memory can be programmed/erased in programmer mode, using a PROM programmer, as well as in on-board programming mode.

8.2.2 Block Diagram

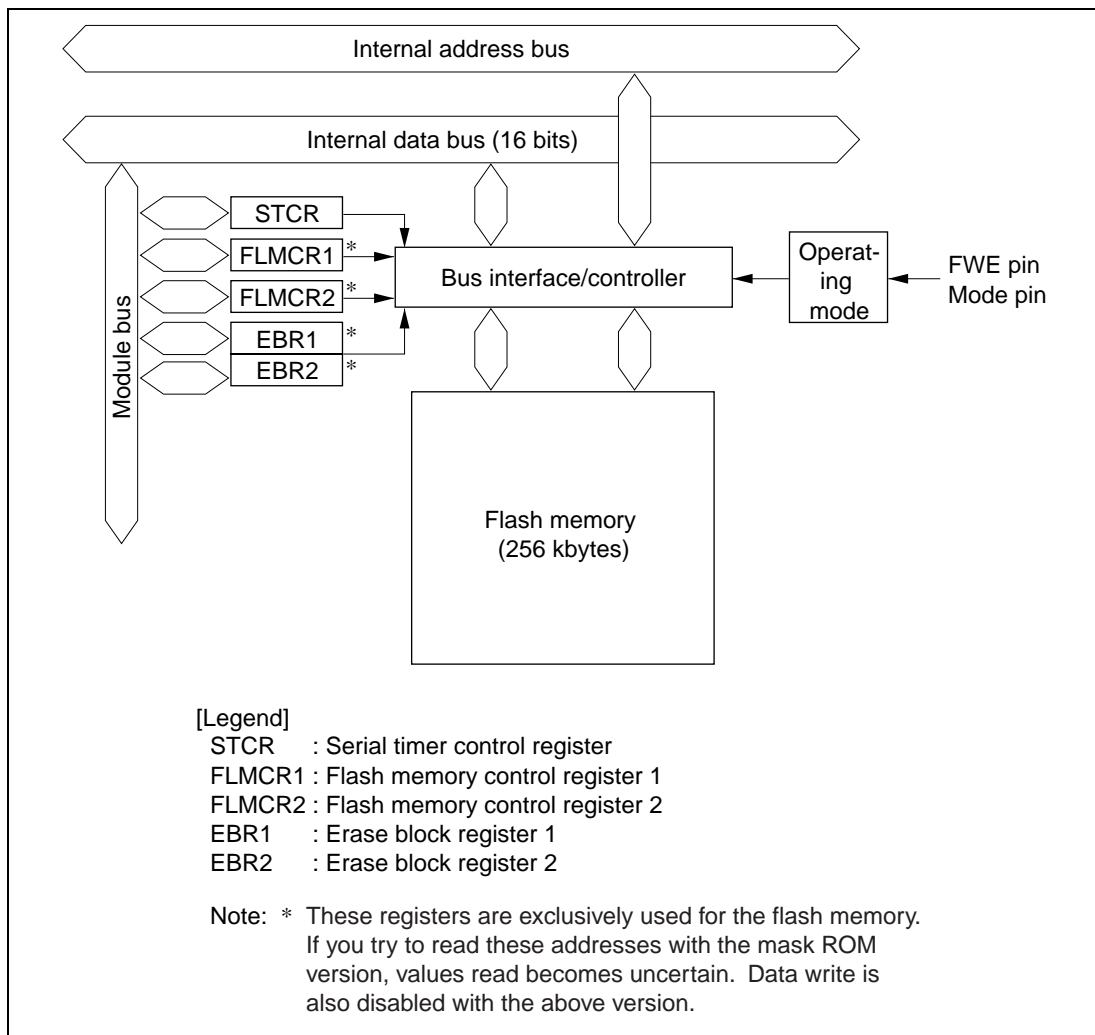


Figure 8.2 Block Diagram of Flash Memory

8.2.3 Flash Memory Operating Modes

(1) Mode Transitions

When each mode pin and the FWE pin are set in the reset state and a reset-start is executed, the MCU enters one of the operating modes shown in figure 8.3. In user mode, flash memory can be read but not programmed or erased.

Flash memory can be programmed and erased in boot mode, user program mode, and programmer mode.

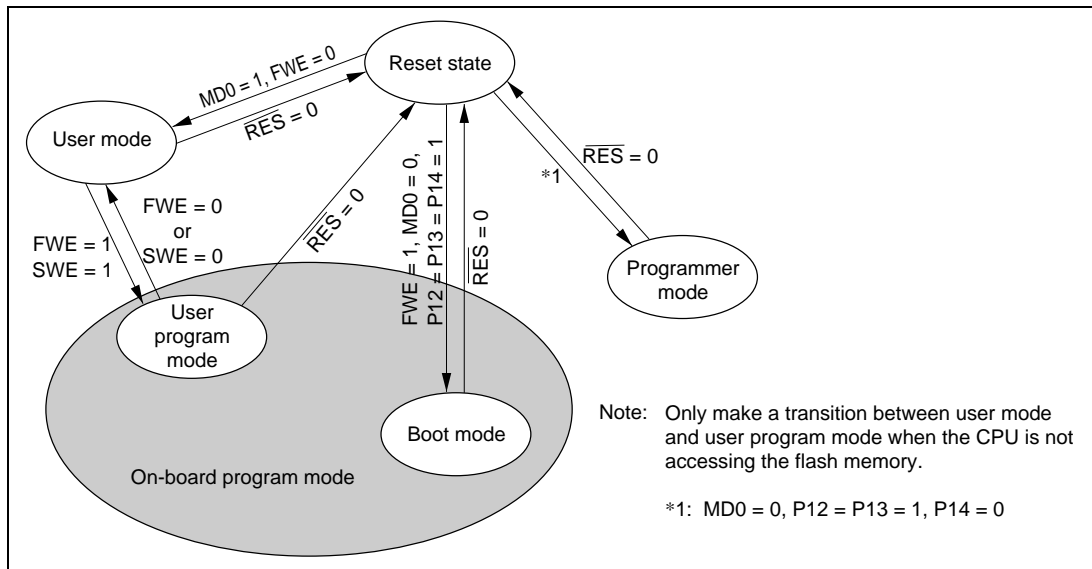


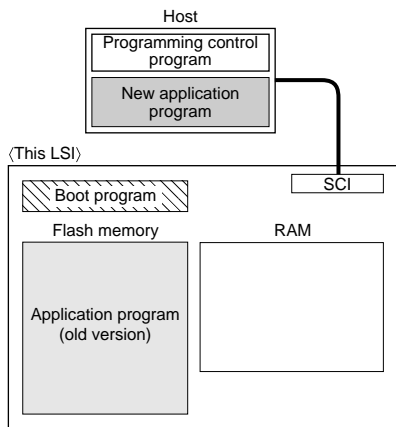
Figure 8.3 Flash Memory Mode Transitions

(2) On-Board Programming Modes

(a) Boot mode

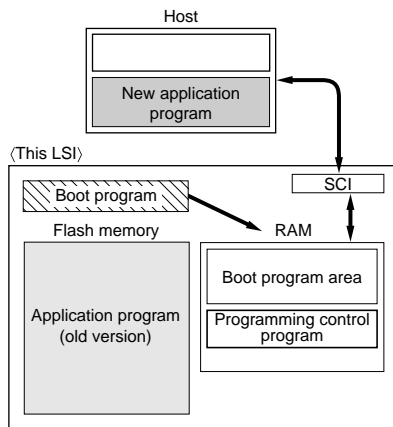
1. Initial state

The old program version or data remains written in the flash memory. The user should prepare the programming control program and new application program beforehand in the host.



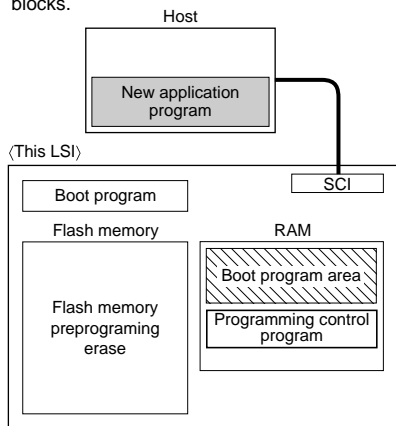
2. Programming control program transfer

When boot mode is entered, the boot program in the LSI (originally incorporated in the chip) is started and the programming control program in the host is transferred to RAM via SCI communication. The boot program required for flash memory erasing is automatically transferred to the RAM boot program area.



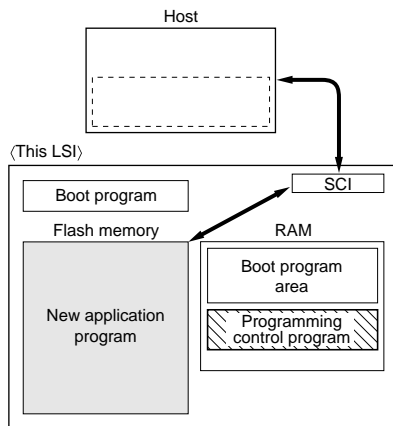
3. Flash memory initialization

The erase program in the boot program area (in RAM) is executed, and the flash memory is initialized (to H'FF). In boot mode, entire flash memory erasure is performed, without regard to blocks.



4. Writing new application program

The programming control program transferred from the host to RAM is executed, and the new application program in the host is written into the flash memory.




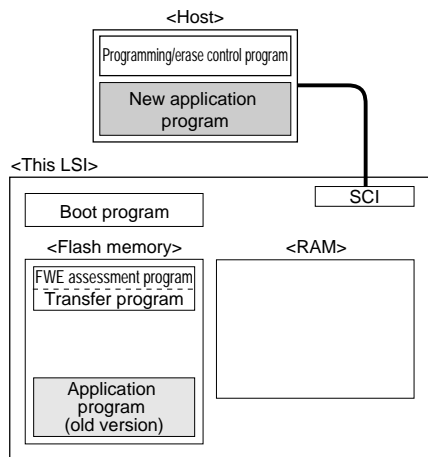
 Program execution state

Figure 8.4 Boot Mode

(b) User program mode

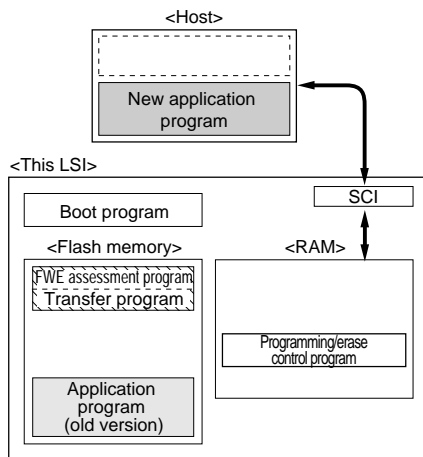
1. Initial state

(1) The FWE assessment program that confirms that the FWE pin has been driven high, and (2) the program that will transfer the programming/erase control program from the flash memory to on-chip RAM should be written into the flash memory by the user beforehand. (3) The programming/erase control program should be prepared in the host or in the flash memory.



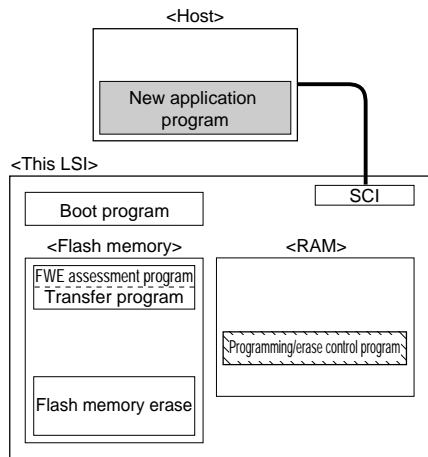
2. Programming/erase control program transfer

When user program mode is entered, user software confirms this fact, executes the transfer program in the flash memory, and transfers the programming/erase control program to RAM.



3. Flash memory initialization

The programming/erase control program in RAM is executed, and the flash memory is initialized (to H'FF). Erasing can be performed in block units, but not in byte units.



4. Writing new application program

Next, the new application program in the host is written into the erased flash memory blocks. Do not write to unerased blocks.

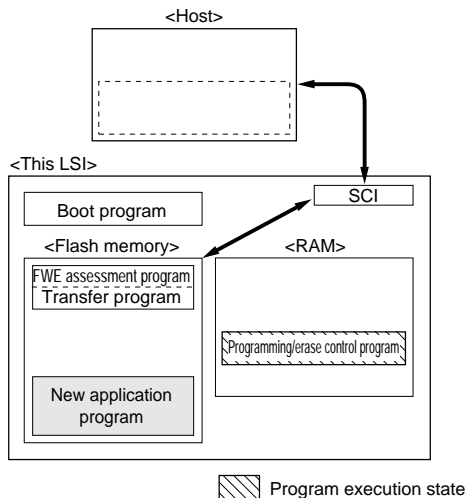


Figure 8.5 User Program Mode (Example)

(3) Differences between Boot Mode and User Program Mode

Table 8.1 Differences between Boot Mode and User Program Mode

| | Boot Mode | User Program Mode |
|------------------------------|------------------------|--|
| Entire memory erase | Yes | Yes |
| Block erase | No | Yes |
| Programming control program* | Program/program-verify | Erase/erase-verify Program/program-verify |

Note: * To be provided by the user, in accordance with the recommended algorithm.

(4) Block Configuration

The flash memory is divided into six 32-kbyte blocks, two 8-kbyte blocks, one 16-kbyte block, one 28-kbyte block, and four 1-kbyte blocks.

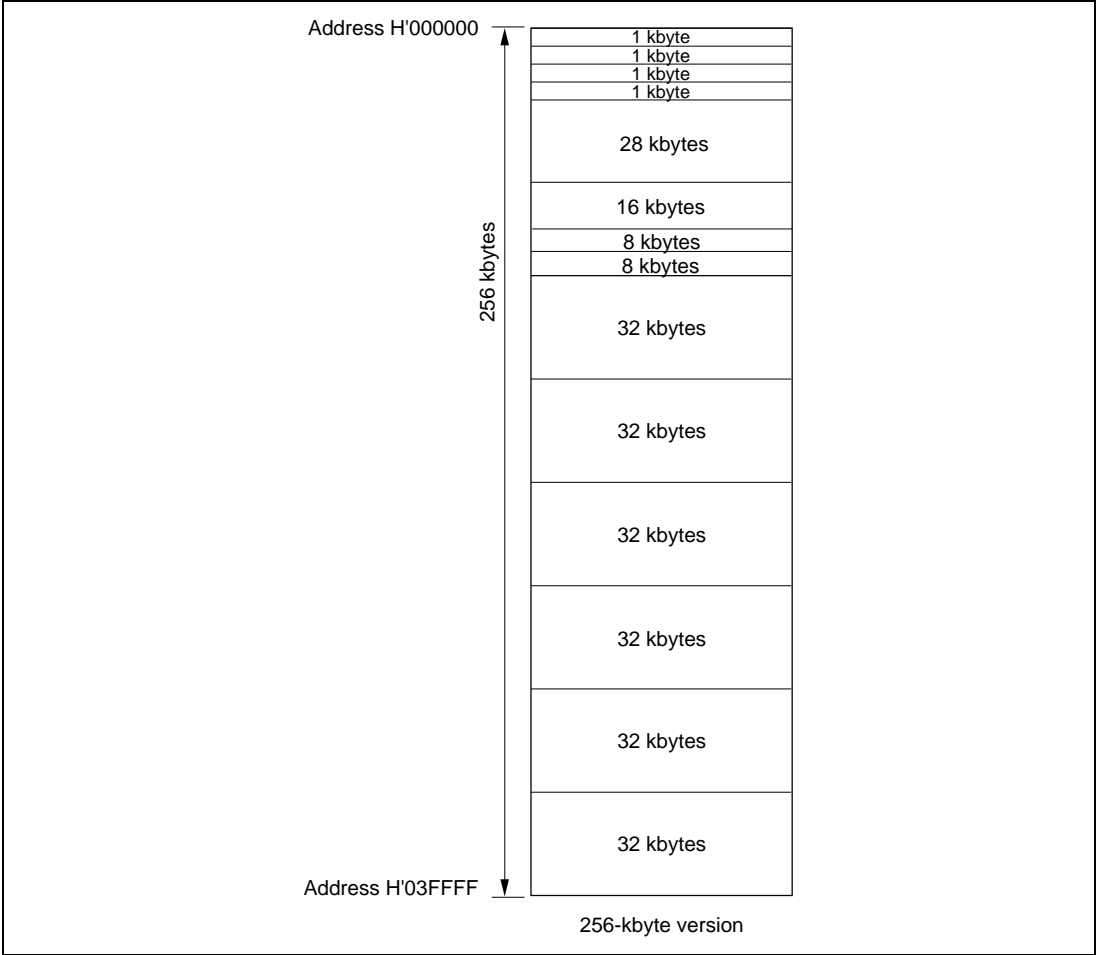


Figure 8.6 Flash Memory Block Configuration

8.2.4 Pin Configuration

The flash memory is controlled by means of the pins shown in table 8.2.

Table 8.2 Flash Memory Pins

| Pin Name | Abbreviation | I/O | Function |
|--------------------|-------------------------|--------|--|
| Reset | $\overline{\text{RES}}$ | Input | Reset |
| Flash write enable | FWE | Input | Flash program/erase protection by hardware |
| Mode 0 | MD0 | Input | Sets this LSI operating mode |
| Port 12 | P12 | Input | Sets this LSI operating mode when MD0 = 0 |
| Port 13 | P13 | Input | Sets this LSI operating mode when MD0 = 0 |
| Port 14 | P14 | Input | Sets this LSI operating mode when MD0 = 0 |
| Transmit data | SO1 | Output | Serial transmit data output |
| Receive data | SI1 | Input | Serial receive data input |

8.2.5 Register Configuration

The registers used to control the on-chip flash memory when enabled are shown in table 8.3. In order to access these registers, the FLSHE bit in STCR must be set to 1.

Table 8.3 Flash Memory Registers

| Register Name | Abbreviation | R/W | Initial Value | Address ^{*5} |
|---------------------------------|----------------------|-------------------|--------------------|-----------------------|
| Flash memory control register 1 | FLMCR1 ^{*4} | R/W ^{*1} | H'00 ^{*3} | H'FFF8 |
| Flash memory control register 2 | FLMCR2 ^{*4} | R/W ^{*1} | H'00 ^{*3} | H'FFF9 |
| Erase block register 1 | EBR1 ^{*4} | R/W ^{*1} | H'00 ^{*3} | H'FFFA |
| Erase block register 2 | EBR2 ^{*4} | R/W ^{*1} | H'00 ^{*3} | H'FFFB |
| Serial timer control register | STCR | R/W | H'00 | H'FFEE |

- Notes: 1. When the FWE bit in FLMCR1 is not set at 1, writes are disabled.
2. When a high level is input to the FWE pin, the initial value is H'80.
3. When a low level is input to the FWE pin, or if a high level is input and the SWE bit in FLMCR1 is not set, these registers are initialized to H'00.
4. FLMCR1, FLMCR2, EBR1, and EBR2 are 8-bit registers. Only byte accesses are valid for these registers, the access requiring 2 states.
5. Lower 16 bits of the address.

8.3 Flash Memory Register Descriptions

8.3.1 Flash Memory Control Register 1 (FLMCR1)

| | | | | | | | | |
|-----------------|-----|-----|------|------|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FWE | SWE | ESU1 | PSU1 | EV1 | PV1 | E1 | P1 |
| Initial value : | —* | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Determined by the state of the FWE pin.

FLMCR1 is an 8-bit register used for flash memory operating mode control. Program-verify mode or erase-verify mode for addresses H'00000 to H'1FFFF is entered by setting SWE to 1 while FWE is 1 and then setting the EV1 bit or PV1 bit. Program mode for addresses H'00000 to H'1FFFF is entered by setting SWE to 1 while FWE is 1, then setting the PSU1 bit, and finally setting the P1 bit. Erase mode for addresses H'00000 to H'1FFFF is entered by setting SWE to 1 while FWE is 1, then setting the ESU1 bit, and finally setting the E1 bit. FLMCR1 is initialized by a reset, in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), or when a low level is input to the FWE pin. When a high level is input to the FWE pin, its initial value is H'80 and when a low level is input, its initial value is H'00.

Writes to the SWE, ESU1, PSU1, EV1, and PV1 bits in FLMCR1 are enabled only when FWE = 1 and SWE = 1; writes to the E1 bit only when FWE = 1, SWE = 1, and ESU1 = 1; and writes to the P1 bit only when FWE = 1, SWE = 1, and PSU1 = 1.

Bit 7: Flash Write Enable (FWE)

Sets hardware protection against flash memory programming/erasing.

Bit 7

| FWE | Description |
|-----|---|
| 0 | When a low level is input to the FWE pin (hardware-protected state) |
| 1 | When a high level is input to the FWE pin |

Bit 6: Software Write Enable (SWE)

Enables or disables flash memory programming. SWE should be set before setting bits 5 to 0, bits 5 to 0 in FLMCR2, bits 5 to 0 in EBR1 and bits 7 to 0 in EBR2.

Bit 6

| SWE | Description |
|-----|---|
| 0 | Writes are disabled (Initial value) |
| 1 | Writes are enabled [Setting condition] Setting is available when FWE = 1 is selected |

Bit 5: Erase-Setup Bit 1 (ESU1)

Prepares erase-mode transition for addresses H'00000 to H'1FFFF. Set ESU1 to 1 before setting the E1 bit to 1. Do not set the SWE, PSU1, EV1, PV1, E1, or P1 bit at the same time.

Bit 5

| ESU1 | Description |
|------|---|
| 0 | Erase-setup cleared (Initial value) |
| 1 | Erase-setup [Setting condition] When FWE = 1 and SWE = 1 |

Bit 4: Program-Setup Bit 1 (PSU1)

Prepares erase-mode transition for addresses H'00000 to H'1FFFF. Set PSU1 to 1 before setting the P1 bit to 1. Do not set the SWE, ESU1, EV1, PV1, E1, or P1 bit at the same time.

Bit 4

| PSU1 | Description |
|------|---|
| 0 | Program-setup cleared (Initial value) |
| 1 | Program-setup [Setting condition] When FWE = 1 and SWE = 1 |

Bit 3: Erase-Verify (EV1)

Selects erase-verify mode transition or clearing. Do not set the SWE, ESU1, PSU1, PV1, E1, or P1 bit at the same time.

Bit 3

| EV1 | Description |
|-----|---|
| 0 | Erase-verify mode cleared (Initial value) |
| 1 | Transition to erase-verify mode [Setting condition] Setting is available when FWE = 1 and SWE = 1 are selected |

Bit 2: Program-Verify (PV1)

Selects program-verify mode transition or clearing. Do not set the SWE, ESU1, PSU1, EV1, E1, or P1 bit at the same time.

Bit 2

| PV1 | Description |
|------------|---|
| 0 | Program-verify mode cleared (Initial value) |
| 1 | Transition to program-verify mode [Setting condition] Setting is available when FWE = 1 and SWE = 1 are selected |

Bit 1: Erase (E1)

Selects erase mode transition or clearing. Do not set the SWE, ESU1, PSU1, EV1, PV1, or P1 bit at the same time.

Bit 1

| E1 | Description |
|-----------|--|
| 0 | Erase mode cleared (Initial value) |
| 1 | Transition to erase mode [Setting condition] Setting is available when FWE = 1, SWE = 1, and ESU = 1 are selected |

Bit 0: Program (P1)

Selects program mode transition or clearing. Do not set the SWE, PSU1, ESU1, EV1, PV1, or E1 bit at the same time.

Bit 0

| P1 | Description |
|-----------|--|
| 0 | Program mode cleared (Initial value) |
| 1 | Transition to program mode [Setting condition] Setting is available when FWE = 1, SWE = 1, and PSU = 1 are selected |

8.3.2 Flash Memory Control Register 2 (FLMCR2)

| | | | | | | | | |
|-----------------|------|---|------|------|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FLER | — | ESU2 | PSU2 | EV2 | PV2 | E2 | P2 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | — | R/W | R/W | R/W | R/W | R/W | R/W |

FLMCR2 is an 8-bit register used for flash memory operating mode control. Program-verify mode or erase-verify mode for addresses H'20000 to H'3FFFF is entered by setting SWE in FLMCR1 to 1 while FWE in FLMCR1 is 1 and then setting the EV2 bit or PV2 bit. Program mode for addresses H'20000 to H'3FFFF is entered by setting SWE in FLMCR1 to 1 while FWE in FLMCR1 is 1, then setting the PSU2 bit, and finally setting the P2 bit. Erase mode for addresses H'20000 to H'3FFFF is entered by setting SWE in FLMCR1 to 1 while FWE in FLMCR1 is 1, then setting the ESU2 bit, and finally setting the E2 bit. FLMCR2 is initialized to H'00 by a reset, in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), when a low level is input to the FWE pin, or when a high level is input to the FWE pin and the SWE bit in FLMCR1 is not set. However, FLER is initialized only by a reset.

Writes to the ESU2, PSU2, EV2, and PV2 bits in FLMCR2 are enabled only when FWE in FLMCR1 = 1 and SWE in FLMCR1 = 1; writes to the E2 bit only when FWE in FLMCR1 = 1, SWE in FLMCR1 = 1, and ESU2 = 1; and writes to the P2 bit only when FWE in FLMCR1 = 1, SWE in FLMCR1 = 1, and PSU2 = 1.

Bit 7: Flash Memory Error (FLER)

Indicates that an error has occurred during an operation on flash memory (programming or erasing). When FLER is set to 1, flash memory goes to the error-protection state.

Bit 7

| FLER | Description |
|------|---|
| 0 | Flash memory is operating normally Flash memory program/erase protection (error protection) is disabled [Clearing condition] Reset or hardware standby mode (Initial value) |
| 1 | An error has occurred during flash memory programming/erasing Flash memory program/erase protection (error protection) is enabled [Setting condition] See section 8.8.3, Error Protection |

Bits 6: Reserved

This bit cannot be modified and is always read as 0.

Bit 5: Erase-Setup Bit 2 (ESU2)

Prepares erase-mode transition for addresses H'20000 to H'3FFFF. Set ESU2 to 1 before setting the E2 bit to 1. Do not set the PSU2, EV2, PV2, E2, or P2 bit at the same time.

Bit 5

| ESU2 | Description |
|------|---|
| 0 | Erase-setup cleared (Initial value) |
| 1 | Erase-setup [Setting condition] When FWE = 1 and SWE = 1 |

Bit 4: Program-Setup Bit 2 (PSU2)

Prepares erase-mode transition for addresses H'20000 to H'3FFFF. Set PSU2 to 1 before setting the P2 bit to 1. Do not set the ESU2, EV2, PV2, E2, or P2 bit at the same time.

Bit 4

| PSU2 | Description |
|------|---|
| 0 | Program-setup cleared (Initial value) |
| 1 | Program-setup [Setting condition] When FWE = 1 and SWE = 1 |

Bit 3: Erase-Verify 2 (EV2)

Selects erase-verify mode transition or clearing for addresses H'20000 to H'3FFFF. Do not set the ESU2, PSU2, PV2, E2, or P2 bit at the same time.

Bit 3

| EV2 | Description |
|-----|---|
| 0 | Erase-verify mode cleared (Initial value) |
| 1 | Transition to erase-verify mode [Setting condition] When FWE = 1 and SWE = 1 |

Bit 2: Program-Verify 2 (PV2)

Selects program-verify mode transition or clearing for addresses H'20000 to H'3FFFF. Do not set the ESU2, PSU2, EV2, E2, or P2 bit at the same time.

Bit 2

| PV2 | Description |
|-----|---|
| 0 | Program-verify mode cleared (Initial value) |
| 1 | Transition to program-verify mode [Setting condition] When FWE = 1 and SWE = 1 |

Bit 1: Erase 2 (E2)

Selects erase mode transition or clearing for addresses H'20000 to H'3FFFF. Do not set the ESU2, PSU2, EV2, PV2, or P2 bit at the same time.

Bit 1

| E2 | Description |
|-----------|---|
| 0 | Erase mode cleared (Initial value) |
| 1 | Transition to erase mode [Setting condition] When FWE = 1, SWE = 1, and ESU2 = 1 |

Bit 0: Program 2 (P2)

Selects program-mode transition or clearing for addresses H'20000 to H'3FFFF. Do not set the ESU2, PSU2, EV2, PV2, or E2 bit at the same time.

Bit 0

| P2 | Description |
|-----------|---|
| 0 | Program-mode cleared (Initial value) |
| 1 | Transition to program-mode [Setting condition] When FWE = 1, SWE = 1, and PSU2 = 1 |

8.3.3 Erase Block Registers 1 (EBR1)

| | | | | | | | | |
|-----------------|---|---|------|------|------|------|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EBR1 : | — | — | EB13 | EB12 | EB11 | EB10 | EB9 | EB8 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

EBR1 is a register that specifies the flash memory erase area block by block. EBR1 is initialized to H'00 by a reset, in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), when a low level is input to the FWE pin, or when a high level is input to the FWE pin and the SWE bit in FLMCR1 is not set. When a bit in EBR1 is set, the corresponding block can be erased. Other blocks are erase-protected. Set only one bit in EBR1 or EBR2 (more than one bit cannot be set).

The flash memory block configuration is shown in table 8.3.

8.3.4 Erase Block Registers 2 (EBR2)

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EBR2 : | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

EBR2 is a register that specifies the flash memory erase area block by block. EBR2 is initialized to H'00 by a reset, in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), when a low level is input to the FWE pin, or when a high level is input to the FWE pin and the SWE bit in FLMCR1 is not set. When a bit in EBR2 is set, the corresponding block can be erased. Other blocks are erase-protected. Set only one bit in EBR1 or EBR2 (more than one bit cannot be set).

The flash memory block configuration is shown in table 8.4.

Table 8.4 Flash Memory Erase Blocks

| Block (Size) | |
|---------------------------|----------------------|
| 128-kbyte Versions | Address |
| EB0 (1 kbyte) | H'000000 to H'0003FF |
| EB1 (1 kbyte) | H'000400 to H'0007FF |
| EB2 (1 kbyte) | H'000800 to H'000BFF |
| EB3 (1 kbyte) | H'000C00 to H'000FFF |
| EB4 (28 kbytes) | H'001000 to H'007FFF |
| EB5 (16 kbytes) | H'008000 to H'00BFFF |
| EB6 (8 kbytes) | H'00C000 to H'00DFFF |
| EB7 (8 kbytes) | H'00E000 to H'00FFFF |
| EB8 (32 kbytes) | H'010000 to H'017FFF |
| EB9 (32 kbytes) | H'018000 to H'01FFFF |
| EB10 (32 kbytes) | H'020000 to H'027FFF |
| EB11 (32 kbytes) | H'028000 to H'02FFFF |
| EB12 (32 kbytes) | H'030000 to H'037FFF |
| EB13 (32 kbytes) | H'038000 to H'03FFFF |

8.3.5 Serial/Timer Control Register (STCR)

| | | | | | | | | | |
|---------------|---|---|------|--------|---|-------|---|---|---|
| Bit | : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | — | IICX | IICRST | — | FLSHE | — | — | — |
| Initial value | : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W | : | — | R/W | R/W | — | R/W | — | — | — |

STCR is an 8-bit readable/writable register that controls register access, the I²C bus interface operating mode, and on-chip flash memory (in F-ZTAT versions), and also selects the I²C bus interface serial clock frequency. For details on functions not related to on-chip flash memory, see section 25.2.7, Serial/Timer Control Register (STCR), and descriptions of individual modules. If a module controlled by STCR is not used, do not write 1 to the corresponding bit. STCR is initialized to H'00 by a reset.

Bits 6 to 5: I²C Control (IICX, IICRST)

These bits control the operation of the I²C bus interface. For details, see section 25, I²C Bus Interface.

Bit 3: Flash Memory Control Register Enable (FLSHE)

Setting the FLSHE bit to 1 enables read/write access to the flash memory control registers. If FLSHE is cleared to 0, the flash memory control registers are deselected. In this case, the flash memory control register contents are retained.

Bit 3

| FLSHE | Description |
|-------|---|
| 0 | Flash memory control registers deselected (Initial value) |
| 1 | Flash memory control registers selected |

Bits 7, 4 and 2 to 0: Reserved

8.4 On-Board Programming Modes

When pins are set to on-board programming mode, program/erase/verify operations can be performed on the on-chip flash memory. There are two on-board programming modes: boot mode and user program mode. The pin settings for transition to each of these modes are shown in table 8.5. For a diagram of the transitions to the various flash memory modes, see figure 8.3.

Table 8.5 Setting On-Board Programming Modes

| Mode | Pin | | | | |
|-------------------|-----------------|-----|-----------------|-----------------|-----------------|
| Mode Name | FWE | MD0 | P12 | P13 | P14 |
| Boot mode | 1 | 0 | 1 ^{*2} | 1 ^{*2} | 1 ^{*2} |
| User program mode | 1 ^{*1} | 1 | — | — | — |

Notes: 1. In user program mode, the FWE pin should not be constantly set to 1. Set FWE to 1 to make a transition to user program mode before performing a program/erase/verify operation.

2. Can be used as I/O ports after boot mode is initiated.

8.4.1 Boot Mode

When boot mode is used, the flash memory programming control program must be prepared in the host beforehand. The channel 1 SCI to be used is set to asynchronous mode.

When a reset-start is executed after the MCU's pins have been set to boot mode, the boot program built into the MCU is started and the programming control program prepared in the host is serially transmitted to the MCU via the SCI1. In the MCU, the programming control program received via the SCI1 is written into the programming control program area in on-chip RAM. After the transfer is completed, control branches to the start address of the programming control program area and the programming control program execution state is entered (flash memory programming is performed).

The transferred programming control program must therefore include coding that follows the programming algorithm given later.

The system configuration in boot mode is shown in figure 8.7, and the boot program mode execution procedure in figure 8.8.

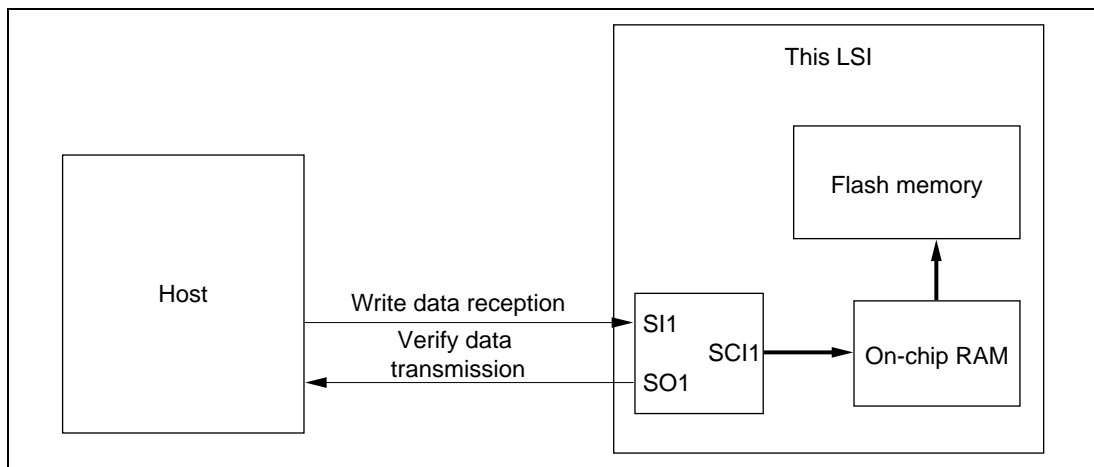
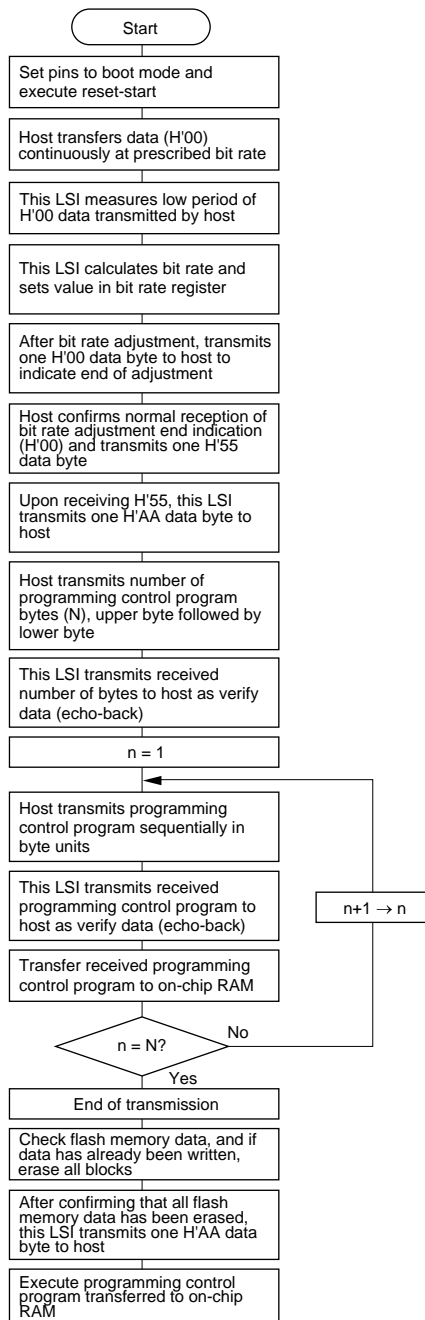


Figure 8.7 System Configuration in Boot Mode



Note : If a memory cell does not operate normally and cannot be erased, one H'FF byte is transmitted as an erase error, and the erase operation and subsequent operations are halted.

Figure 8.8 Boot Mode Execution Procedure

(1) Automatic SCI Bit Rate Adjustment

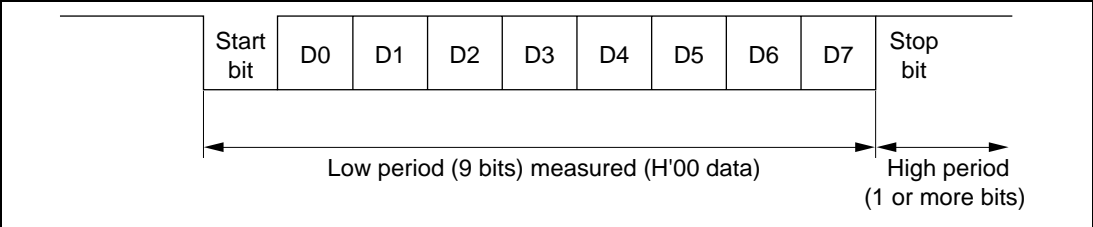


Figure 8.9 Automatic SCI Bit Rate Adjustment

When boot mode is initiated, the MCU measures the low period of the asynchronous SCI communication data (H'00) transmitted continuously from the host. The SCI transmit/receive format should be set as follows: 8-bit data, 1 stop bit, no parity. The MCU calculates the bit rate of the transmission from the host from the measured low period, and transmits one H'00 byte to the host to indicate the end of bit rate adjustment. The host should confirm that this adjustment end indication (H'00) has been received normally, and transmit one H'55 byte to the MCU. If reception cannot be performed normally, initiate boot mode again (reset), and repeat the above operations. Depending on the host's transmission bit rate and the MCU's system clock frequency, there will be a discrepancy between the bit rates of the host and the MCU. To ensure correct SCI operation, the host's transfer bit rate should be set to (2400, 4800, or 9600) bps. Table 8.6 shows typical host transfer bit rates and system clock frequencies for which automatic adjustment of the MCU's bit rate is possible. The boot program should be executed within this system clock range.

Table 8.6 System Clock Frequencies for which Automatic Adjustment of This LSI Bit Rate is Possible

| Host Bit Rate | System Clock Frequency for which Automatic Adjustment of This LSI Bit Rate is Possible |
|---------------|--|
| 9600 bps | 8 MHz to 10 MHz |
| 4800 bps | 4 MHz to 10 MHz |

(2) On-Chip RAM Area Divisions in Boot Mode

In boot mode, the RAM area is divided into; the area for use by the boot program and the area to which programming control program is transferred by the SCI, as shown in figure 8.10. The boot program area cannot be used until a transition is made to the execution state in boot mode for the programming control program transferred to RAM.

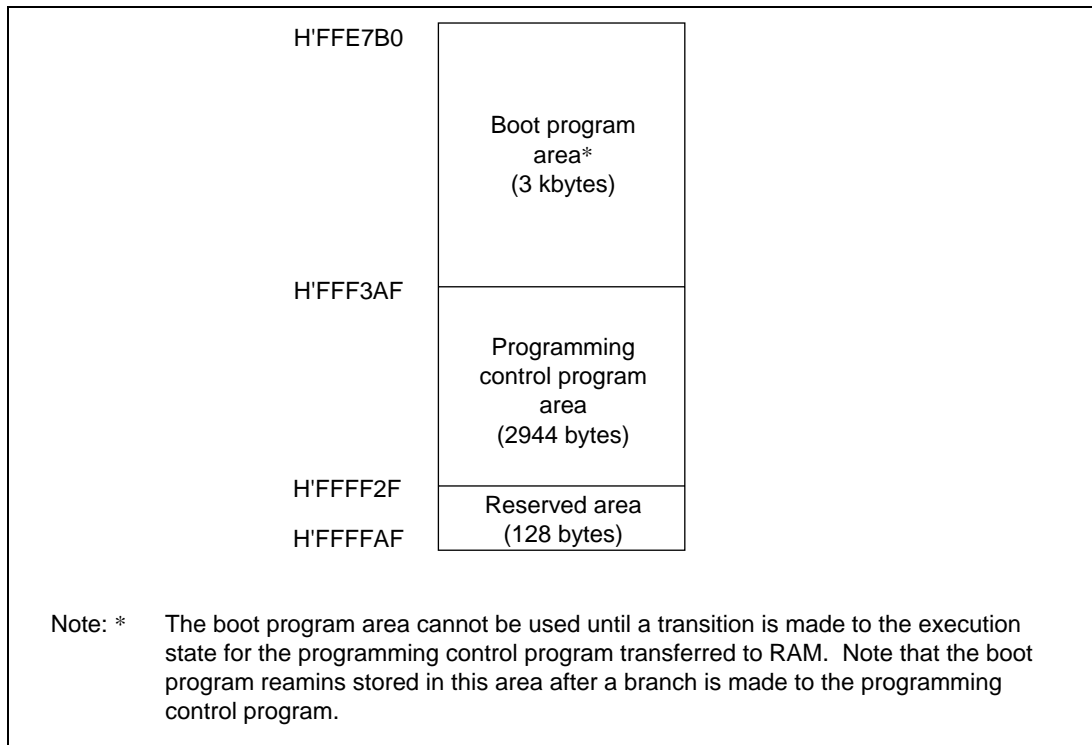


Figure 8.10 RAM Areas in Boot Mode

(3) Notes on Use of Boot Mode:

- (a) When reset is released in boot mode, it measures the low period of the input at the SCI1's SI1 pin. The reset should end with SI1 pin high. After the reset ends, it takes about 100 states for the chip to get ready to measure the low period of the SI1 pin input.
- (b) In boot mode, if any data has been programmed into the flash memory (if all data is not 1), all flash memory blocks are erased. Boot mode is for use when user program mode is unavailable, such as the first time on-board programming is performed, or if the program activated in user program mode is accidentally erased.
- (c) Interrupts cannot be used while the flash memory is being programmed or erased.
- (d) The SI1 and SO1 pins should be pulled up on the board.
- (e) Before branching to the programming control program (RAM area H'FFF3B0), the chip terminates transmit and receive operations by the on-chip SCI (channel 1) (by clearing the RE and TE bits in SCR to 0), but the adjusted bit rate value remains set in BRR. The transmit data output pin, SO1, goes to the high-level output state (P21PCR = 1, P21PDR = 1).

The contents of the CPU's internal general registers are undefined at this time, so these registers must be initialized immediately after branching to the programming control program. In particular, since the stack pointer (SP) is used implicitly in subroutine calls, etc., a stack area must be specified for use by the programming control program.

The initial values of other on-chip registers are not changed.

- (f) Boot mode can be entered by making the pin settings shown in table 8.6 and executing a reset-start.

When the chip detects the boot mode setting at reset release^{*1}, it retains that state internally.

Boot mode can be cleared by driving the reset pin low, waiting at least 20 states, then setting the FWE pin and mode pins, and executing reset release^{*1}. Boot mode can also be cleared by a WDT overflow reset.

If the mode pin input levels are changed in boot mode, the boot mode state will be maintained in the microcomputer, and boot mode continued, unless a reset occurs.

However, the FWE pin must not be driven low while the boot program is running or flash memory is being programmed or erased^{*2}.

- Notes:
- 1. Mode pin and FWE pin input must satisfy the mode programming setup time ($t_{\text{MDS}} = 4$ states) with respect to the reset release timing.
 - 2. For further information on FWE application and disconnection, see section 8.9, Flash Memory Programming and Erasing Precautions.

8.4.2 User Program Mode

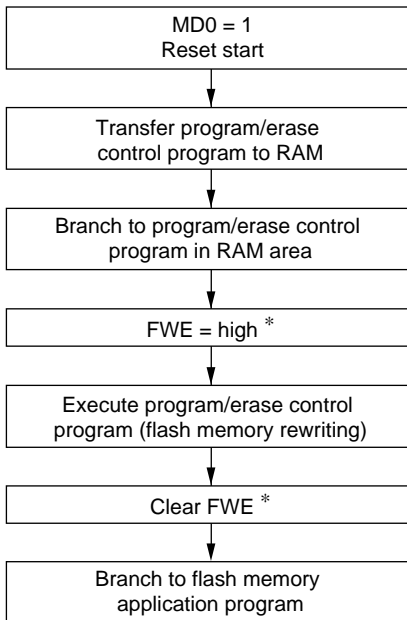
When set to user program mode, the chip can program and erase its flash memory by executing a user program/erase control program. Therefore, on-board reprogramming of the on-chip flash memory can be carried out by providing on-board means of FWE control and supply of programming data, and storing a program/erase control program in part of the program area as necessary.

In this mode, the chip starts up in mode 1 and applies a high level to the FWE pin.

The flash memory itself cannot be read while the SWE bit is set to 1 to perform programming or erasing, so the control program that performs programming and erasing should be run in on-chip RAM or external memory.

Figure 8.11 shows the procedure for executing the program/erase control program when transferred to on-chip RAM.

Write the FWE assessment program and transfer program (and the program/erase control program if necessary) beforehand



Note: Do not apply a constant high level to the FWE pin. Apply a high level to the FWE pin only when the flash memory is programmed or erased. Also, while a high level is applied to the FWE pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc.

* For further information on FWE application and disconnection, see section 8.9, Flash Memory Programming and Erasing Precautions.

Figure 8.11 User Program Mode Execution Procedure (Preliminary)

8.5 Programming/Erasing Flash Memory

In the on-board programming modes, flash memory programming and erasing is performed by software, using the CPU. There are four flash memory operating modes: program mode, erase mode, program-verify mode, and erase-verify mode. For addresses H'00000 to H'1FFFF, transitions to these modes can be made by setting the PSU1, ESU1, P1, E1, PV1 and EV1 bits in FLMCR1 and for addresses H'20000 to H'3FFFF, transitions to these modes can be made by setting the PSU2, ESU2, P2, E2, PV2 and EV1 bits in FLMCR2.

The flash memory cannot be read while being programmed or erased. Therefore, the program that controls flash memory programming/erasing (the programming control program) should be located and executed in on-chip RAM or external memory.

- Notes:
1. Operation is not guaranteed if setting/resetting of the SWE, ESU1, PSU1, EV1, PV1, E1, and P1 bits in FLMCR1, and the ESU2, PSU2, EV2, PV2, E2 and P2 bits in FLMCR2, is executed by a program in flash memory.
 2. When programming or erasing, set FWE to 1 (programming/erasing will not be executed if FWE = 0).
 3. Perform programming in the erased state. Do not perform additional programming on previously programmed addresses.
Do not program addresses H'00000 to H'1FFFF and H'20000 to H'3FFFF at the same time. Operation is not guaranteed if both areas are programmed at the same time.

8.5.1 Program Mode (n = 1 for addresses H'0000 to H'1FFFF and n= 2 for addresses H'20000 to H'3FFFF)

Follow the procedure shown in the program/program-verify flowchart in figure 8.12 to write data or programs to flash memory. Performing program operations according to this flowchart will enable data or programs to be written to flash memory without subjecting the device to voltage stress or sacrificing program data reliability. Programming should be carried out 32 bytes at a time.

Table 29.9 in section 29.2.7, Flash Memory Characteristics, lists wait time (x, y, z, α , β , γ , ϵ and η) after setting or clearing each bit on the flash memory control registers 1 and 2 (FLMCR1 and FLMCR2) and the maximum write count (N).

Following the elapse of (x) μ s or more after the SWE bit is set to 1 in flash memory control register 1 (FLMCR1), 32-byte program data is stored in the program data area and reprogram data area, and the 32-byte data in the reprogram data area written consecutively to the write addresses. The lower 8 bits of the first address written to must be H'00, H'20, H'40, H'60, H'80, H'A0, H'C0, or H'E0. Thirty-two consecutive byte data transfers are performed. The program address and program data are latched in the flash memory. A 32-byte data transfer must be performed even if writing fewer than 32 bytes; in this case, H'FF data must be written to the extra addresses.

Next, the watchdog timer is set to prevent overprogramming in the event of program runaway, etc. Set more than (y + z + α + β) μ s as the WDT overflow period. After this, preparation for

program mode (program setup) is carried out by setting the PSUn bit in FLMCRn, and after the elapse of (γ) μ s or more, the operating mode is switched to program mode by setting the Pn bit in FLMCRn. The time during which the Pn bit is set is the flash memory programming time. Make a program setting so that the time for one programming operation is within the range of (z) μ s.

8.5.2 Program-Verify Mode (n =1 for addresses H'00000 to H'1FFFF and n = 2 for addresses H'20000 to H'3FFFF)

In program-verify mode, the data written in program mode is read to check whether it has been correctly written in the flash memory.

After the elapse of a given programming time, the programming mode is exited (the Pn bit in FLMCRn is cleared, then the PSUn bit in FLMCRn is cleared at least (α) μ s later). The watchdog timer is cleared after the elapse of (β) μ s or more, and the operating mode is switched to program-verify mode by setting the PVn bit in FLMCRn. Before reading in program-verify mode, a dummy write of H'FF data should be made to the addresses to be read. The dummy write should be executed after the elapse of (γ) μ s or more. When the flash memory is read in this state (verify data is read in 16-bit units), the data at the latched address is read. Wait at least (ϵ) μ s after the dummy write before performing this read operation. Next, the originally written data is compared with the verify data, and reprogram data is computed (see figure 8.12) and transferred to the reprogram data area. After 32 bytes of data have been verified, exit program-verify mode, wait for at least (η) μ s, then clear the SWE bit in FLMCR1. If reprogramming is necessary, set program mode again, and repeat the program/program-verify sequence as before. However, ensure that the program/program-verify sequence is not repeated more than (N) times on the same bits.

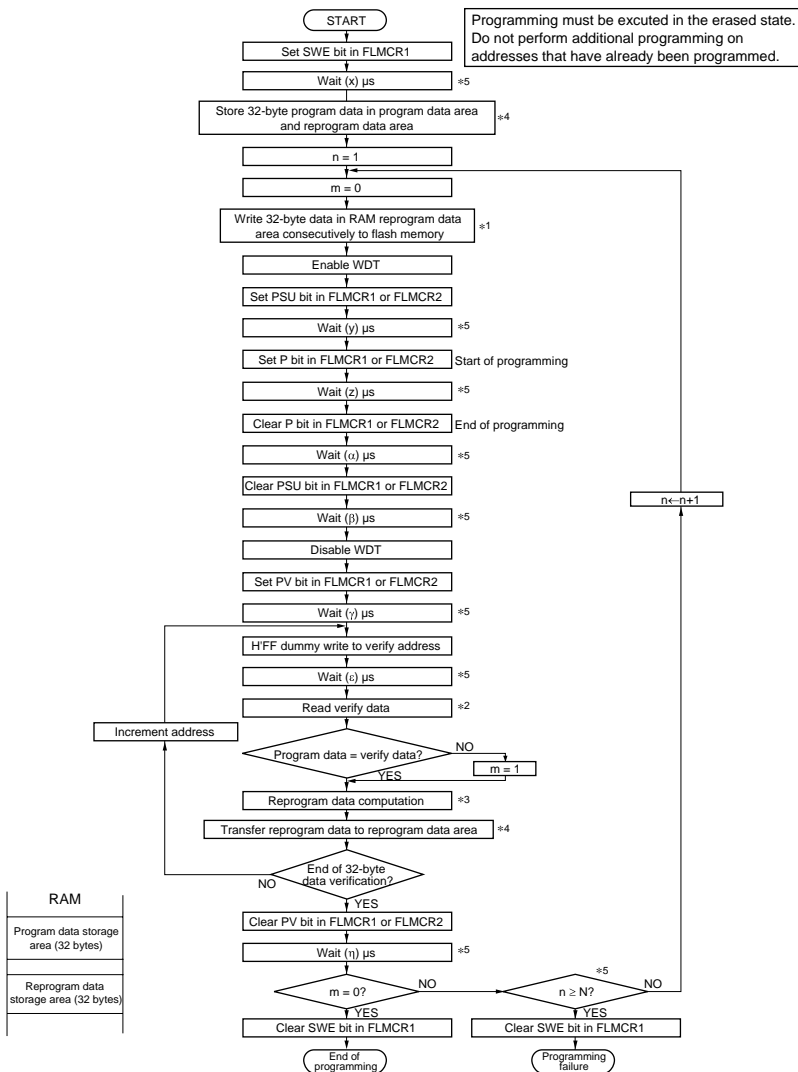


Figure 8.12 Program/Program-Verify Flowchart

8.5.3 Erase Mode (n = 1 for addresses H'00000 to H'1FFFF and n = 2 for address H'20000 to H'3FFFF)

Flash memory erasing should be performed block by block following the procedure shown in the erase/erase-verify flowchart (single-block erase) shown in figure 8.13.

Table 28.9 in section 28.2.7, Flash Memory Characteristics lists wait time (x , y , z , α , β , γ , ϵ and η) after setting or clearing each bit on the flash memory control registers 1 and 2 (FLMCR1 and FLMCR2) and the maximum clearing count (N).

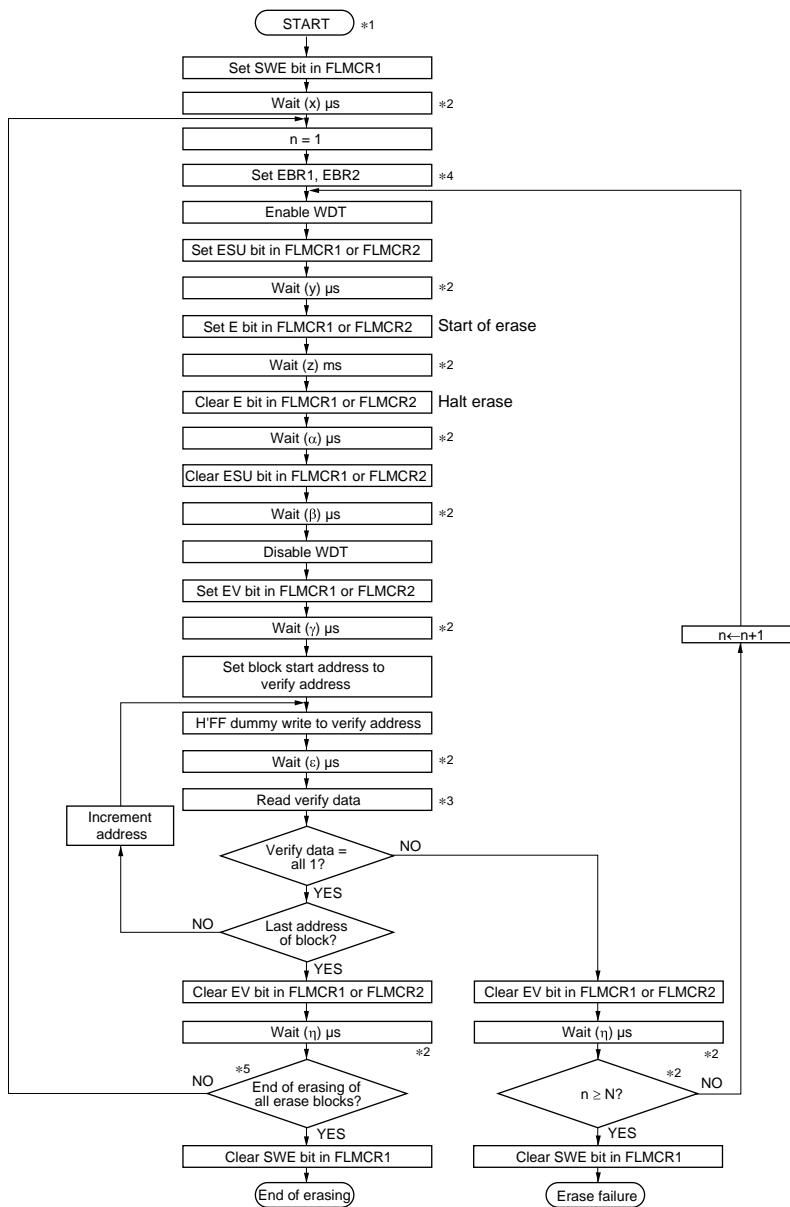
To perform data or program erasure, make a 1 bit setting for the flash memory area to be erased in erase block register 1 or 2 (EBR1 or EBR2) at least (x) μ s after setting the SWE bit to 1 in flash memory control register 1 (FLMCR1). Next, the watchdog timer is set to prevent overerasing in the event of program runaway, etc. Set more than ($y + z + \alpha + \beta$) ms as the WDT overflow period. After this, preparation for erase mode (erase setup) is carried out by setting the ESUn bit in FLMCRn, and after the elapse of (y) μ s or more, the operating mode is switched to erase mode by setting the En bit in FLMCR1. The time during which the En bit is set is the flash memory erase time. Ensure that the erase time does not exceed (z) ms.

Note: With flash memory erasing, preprogramming (setting all data in the memory to be erased to 0) is not necessary before starting the erase procedure.

8.5.4 Erase-Verify Mode (n = 1 for addresses H'00000 to H'1FFFF and n = 2 for address H'20000 to H'3FFFF)

In erase-verify mode, data is read after memory has been erased to check whether it has been correctly erased.

After the elapse of the erase time, erase mode is exited (the En bit in FLMCRn is cleared, then the ESU bit in FLMCR2 is cleared at least (α) μ s later), the watchdog timer is cleared after the elapse of (β) μ s or more, and the operating mode is switched to erase-verify mode by setting the EVn bit in FLMCRn. Before reading in erase-verify mode, a dummy write of H'FF data should be made to the addresses to be read. The dummy write should be executed after the elapse of (γ) μ s or more. When the flash memory is read in this state (verify data is read in 16-bit units), the data at the latched address is read. Wait at least (ϵ) μ s after the dummy write before performing this read operation. If the read data has been erased (all 1), a dummy write is performed to the next address, and erase-verify is performed. If the read data has not been erased, set erase mode again, and repeat the erase/erase-verify sequence in the same way. However, ensure that the erase/erase-verify sequence is not repeated more than (N) times. When verification is completed, exit erase-verify mode, and wait for at least (η) μ s. If erasure has been completed on all the erase blocks, clear the SWE bit in FLMCR1. If there are any unerased blocks, make a 1 bit setting in EBR1 or EBR2 for the flash memory area to be erased, and repeat the erase/erase-verify sequence in the same way.



- Notes:
1. Preprogramming (setting erase block data to all 0) is not necessary.
 2. The values of x, y, z, α, β, γ, ε, η and N are listed in section 29.2.7, Flash Memory Characteristics.
 3. Verify data is read in 16-bit (word) units.
 4. Set only one bit in EBR1 or EBR2. More than one bit cannot be set.
 5. Erasing is performed in block units. To erase a number of blocks, the individual blocks must be erased sequentially.

Figure 8.13 Erase/Erase-Verify Flowchart (Single-Block Erase)

8.6 Flash Memory Protection

There are three kinds of flash memory program/erase protection: hardware protection, software protection, and error protection.

8.6.1 Hardware Protection

Hardware protection refers to a state in which programming/erasing of flash memory is forcibly disabled or aborted. Hardware protection is reset by settings in flash memory control registers 1 and 2 (FLMCR1, FLMCR2) and erase block registers 1 and 2 (EBR1, EBR2). (See table 8.7.)

Table 8.7 Hardware Protection

| Item | Description | Functions | |
|--------------------------|--|-----------|-------|
| | | Program | Erase |
| FWE pin protection | <ul style="list-style-type: none">When a low level is input to the FWE pin, FLMCR1, FLMCR2, EBR1, and EBR2 are initialized, and the program/erase-protected state is entered | Yes | Yes |
| Reset/standby protection | <ul style="list-style-type: none">In a reset (including a WDT overflow reset) and in power-down state (excluding the medium-speed mode, module stop mode, and sleep mode), FLMCR1, FLMCR2 (excluding the FLER bit), EBR1, and EBR2 are initialized, and the program/erase-protected state is enteredIn a reset via the $\overline{\text{RES}}$ pin, the reset state is not entered unless the $\overline{\text{RES}}$ pin is held low until oscillation stabilizes after powering on. In the case of a reset during operation, hold the $\overline{\text{RES}}$ pin low for the $\overline{\text{RES}}$ pulse width specified in the AC characteristics section | Yes | Yes |

8.6.2 Software Protection

Software protection can be implemented by setting the SWE bit in FLMCR1 and erase block registers 1 and 2 (EBR1, EBR2). When software protection is in effect, setting the P1 or E1 bit in flash memory control register 1 (FLMCR1), or setting the P2 or E2 bit in flash memory control register 2 (FLMCR2) does not cause a transition to program mode or erase mode. (See table 8.8.)

Table 8.8 Software Protection

| Item | Description | Functions | |
|--------------------------------|--|-----------|-------|
| | | Program | Erase |
| SWE bit protection | <ul style="list-style-type: none">Clearing the SWE bit to 0 in FLMCR1 sets the program/erase-protected state for all blocks (Execute in on-chip RAM or external memory) | Yes | Yes |
| Block specification protection | <ul style="list-style-type: none">Erase protection can be set for individual blocks by settings in erase block registers 1 and 2 (EBR1, EBR2)Setting EBR1 and EBR2 to H'00 places all blocks in the erase-protected state | – | Yes |

8.6.3 Error Protection

In error protection, an error is detected when MCU runaway occurs during flash memory programming/erasing, or operation is not performed in accordance with the program/erase algorithm, and the program/erase operation is aborted. Aborting the program/erase operation prevents damage to the flash memory due to overprogramming or overerasing.

If the MCU malfunctions during flash memory programming/erasing, the FLER bit is set to 1 in FLMCR2 and the error protection state is entered. The FLMCR1, FLMCR2, EBR1, and EBR2 settings are retained, but program mode or erase mode is aborted at the point at which the error occurred. Program mode or erase mode cannot be re-entered by re-setting the P1, E1, P2 or E2 bit. However, PV1, EV1, PV2 or EV2 bit setting is enabled, and a transition can be made to verify mode.

FLER bit setting conditions are as follows:

- (1) When flash memory is read during programming/erasing (including a vector read or instruction fetch)
- (2) Immediately after exception handling (excluding a reset) during programming/erasing
- (3) When a SLEEP instruction is executed during programming/erasing

Error protection is released only by a reset and in hardware standby mode.

Figure 8.14 shows the flash memory state transition diagram.

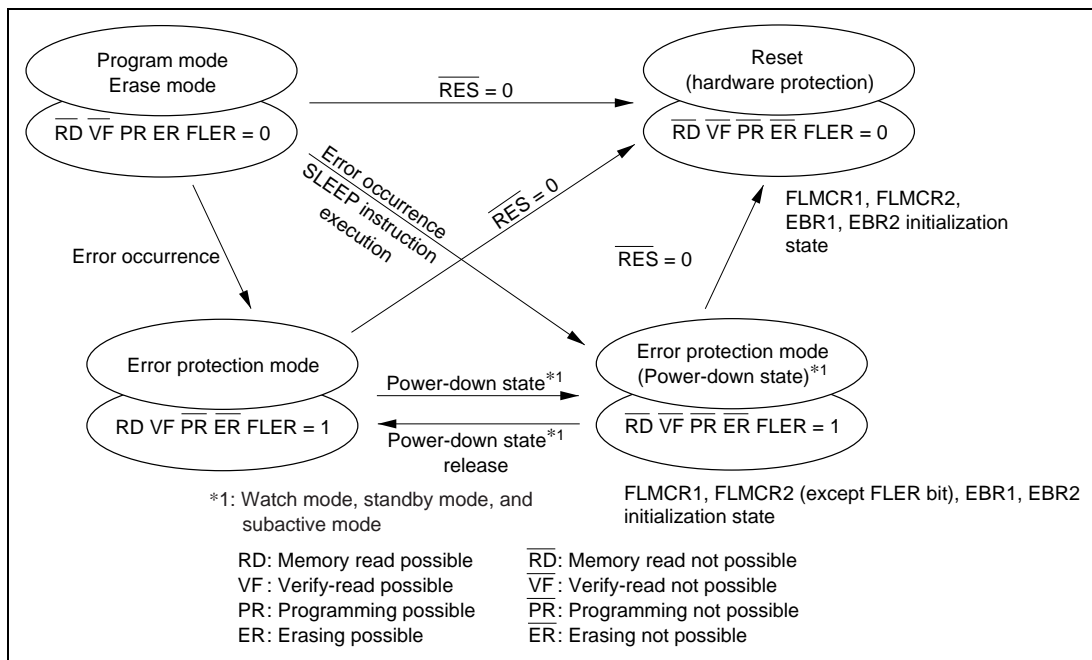


Figure 8.14 Flash Memory State Transitions

8.7 Interrupt Handling when Programming/Erasing Flash Memory

All interrupts, including NMI input is disabled when flash memory is being programmed or erased (when the Pn or En bit is set in FLMCRn), and while the boot program is executing in boot mode^{*1}, to give priority to the program or erase operation. There are three reasons for this:

- (1) Interrupt during programming or erasing might cause a violation of the programming or erasing algorithm, with the result that normal operation could not be assured.
- (2) In the interrupt exception handling sequence during programming or erasing, the vector would not be read correctly^{*2}, possibly resulting in MCU runaway.
- (3) If interrupt occurred during boot program execution, it would not be possible to execute the normal boot mode sequence.

For these reasons, in on-board programming mode alone there are conditions for disabling interrupt, as an exception to the general rule. However, this provision does not guarantee normal erasing and programming or MCU operation. All requests, including NMI input, must therefore be disabled inside and outside the MCU during FWE application. Interrupt is also disabled in the error-protection state while the Pn or En bit remains set in FLMCRn.

- Notes:
1. Interrupt requests must be disabled inside and outside the MCU until data write by the write control program is complete.
 2. The vector may not be read correctly in this case for the following two reasons:
 - If flash memory is read while being programmed or erased (while the Pn or En bit is set in FLMCRn), correct read data will not be obtained (undetermined values will be returned).
 - If the interrupt entry in the interrupt vector table has not been programmed yet, interrupt exception handling will not be executed correctly.

8.8 Flash Memory Programmer Mode

8.8.1 Programmer Mode Setting

Programs and data can be written and erased in programmer mode as well as in the on-board programming modes. In programmer mode, the on-chip ROM can be freely programmed using a PROM programmer that supports Hitachi microcomputer device type with 128-kbyte on-chip flash memory. Flash memory read mode, auto-program mode, auto-erase mode, and status read mode are supported with these device types. In auto-program mode, auto-erase mode, and status read mode, a status polling procedure is used, and in status read mode, detailed internal signals are output after execution of an auto-program or auto-erase operation.

8.8.2 Socket Adapters and Memory Map

In programmer mode, a socket adapter is mounted on the writer programmer. The socket adapter product codes are listed in table 8.9.

Figure 8.15 shows the memory map in programmer mode.

Table 8.9 Socket Adapter Product Codes

| Product Codes | Package | Socket Adapter Product Code |
|---------------|-------------|-----------------------------|
| HD64F2194C | 112-pin QFP | ME2194ESHF1H |

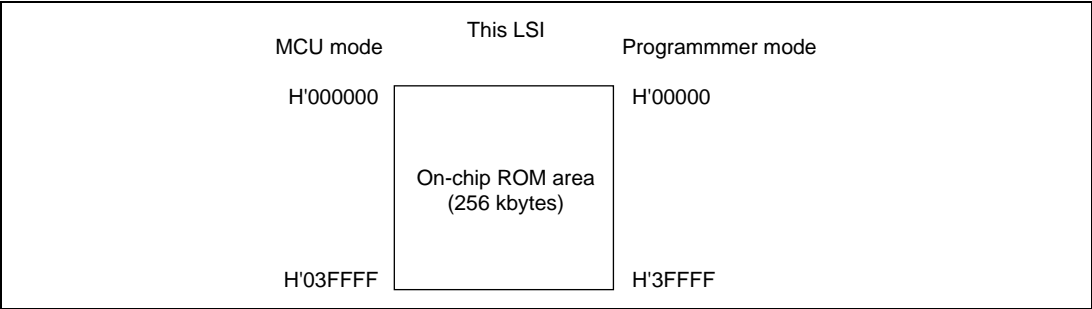


Figure 8.15 Memory Map in Programmer Mode

8.8.3 Programmer Mode Operation

Table 8.10 shows how the different operating modes are set when using programmer mode, and table 8.11 lists the commands used in programmer mode. Details of each mode are given below.

(1) Memory Read Mode

Memory read mode supports byte reads.

(2) Auto-Program Mode

Auto-program mode supports programming of 128 bytes at a time. Status polling is used to confirm the end of auto-programming.

(3) Auto-Erase Mode

Auto-erase mode supports automatic erasing of the entire flash memory. Status polling is used to confirm the end of auto-erasing.

(4) Status Read Mode

Status polling is used for auto-programming and auto-erasing, and normal termination can be confirmed by reading the FO6 signal. In status read mode, error information is output if an error occurs.

Table 8.10 Settings for Each Operating Mode in Programmer Mode

| Mode | Pin Names | | | | | |
|----------------------------|----------------------|------------------------|------------------------|------------------------|-------------|-------------------|
| | FWE | $\overline{\text{CE}}$ | $\overline{\text{OE}}$ | $\overline{\text{WE}}$ | FO0 to FO7 | FA0 to FA17 |
| Read | H or L | L | L | H | Data output | Ain |
| Output disable | H or L | L | H | H | Hi-z | X |
| Command write | H or L ^{*3} | L | H | L | Data input | Ain ^{*2} |
| Chip disable ^{*1} | H or L | L | X | X | Hi-z | X |

Notes: 1. Chip disable is not a standby state; internally, it is an operation state.

2. Ain indicates that there is also address input in auto-program mode.

3. For command writes when making a transition to auto-program or auto-erase mode, input a high level to the FWE pin.

Table 8.11 Programmer Mode Commands

| Command Name | Number of Cycles | 1st Cycle | | | 2nd Cycle | | |
|-------------------|------------------|-----------|---------|------|-----------|---------|------|
| | | Mode | Address | Data | Mode | Address | Data |
| Memory read mode | 1+n | write | X | H'00 | read | RA | Dout |
| Auto-program mode | 129 | write | X | H'40 | write | WA | Din |
| Auto-erase mode | 2 | write | X | H'20 | write | X | H'20 |
| Status read mode | 2 | write | X | H'71 | write | X | H'71 |

Notes: 1. In auto-program mode, 129 cycles are required for command writing by a simultaneous 128-byte write.

2. In memory read mode, the number of cycles depends on the number of address write cycles (n).

8.8.4 Memory Read Mode

- (1) After the end of an auto-program, auto-erase, or status read operation, the command wait state is entered. To read memory contents, a transition must be made to memory read mode by means of a command write before the read is executed.
- (2) Command writes can be performed in memory read mode, just as in the command wait state.
- (3) Once memory read mode has been entered, consecutive reads can be performed.
- (4) After power-on, memory read mode is entered.

Table 8.12 AC Characteristics in Memory Read Mode

(Conditions: $V_{cc} = 5.0\text{ V} \pm 10\%$, $V_{ss} = 0\text{ V}$, $T_a = 25^{\circ}\text{C} \pm 5^{\circ}\text{C}$)

| Item | Symbol | Min | Max | Unit |
|-----------------------------------|------------|-----|-----|---------------|
| Command write cycle | t_{nxtc} | 20 | — | μs |
| $\overline{\text{CE}}$ hold time | t_{ceh} | 0 | — | ns |
| $\overline{\text{CE}}$ setup time | t_{ces} | 0 | — | ns |
| Data hold time | t_{dh} | 50 | — | ns |
| Data setup time | t_{ds} | 50 | — | ns |
| Write pulse width | t_{wep} | 70 | — | ns |
| $\overline{\text{WE}}$ rise time | t_r | — | 30 | ns |
| $\overline{\text{WE}}$ fall time | t_f | — | 30 | ns |

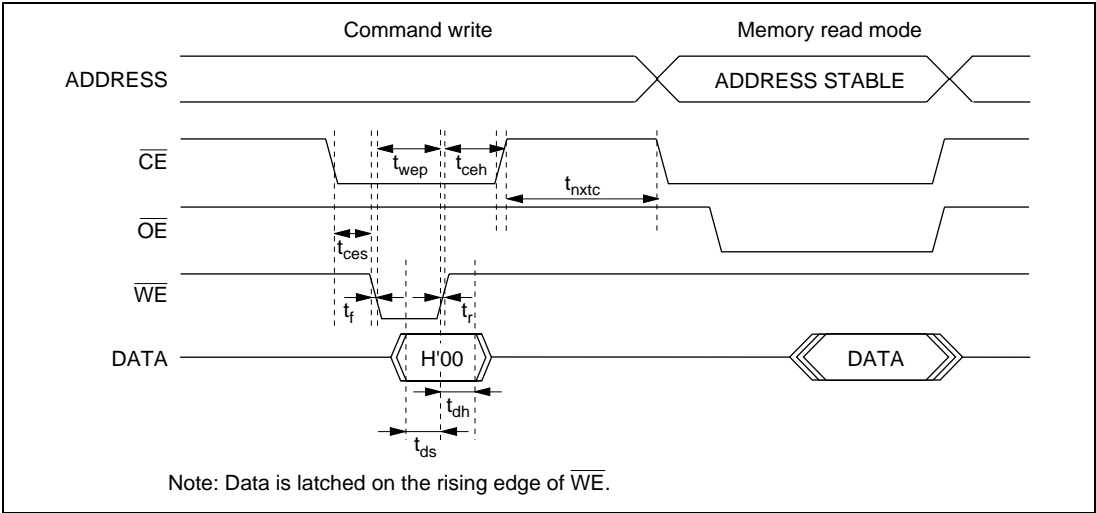


Figure 8.16 Memory Read Mode Timing Waveforms after Command Write

Table 8.13 AC Characteristics when Entering Another Mode from Memory Read Mode(Conditions: $V_{CC} = 5.0 \text{ V} \pm 10\%$, $V_{SS} = 0 \text{ V}$, $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit |
|-----------------------------------|------------|-----|-----|---------------|
| Command write cycle | t_{nxtc} | 20 | — | μs |
| $\overline{\text{CE}}$ hold time | t_{ceh} | 0 | — | ns |
| $\overline{\text{CE}}$ setup time | t_{ces} | 0 | — | ns |
| Data hold time | t_{dh} | 50 | — | ns |
| Data setup time | t_{ds} | 50 | — | ns |
| Write pulse width | t_{wep} | 70 | — | ns |
| $\overline{\text{WE}}$ rise time | t_r | — | 30 | ns |
| $\overline{\text{WE}}$ fall time | t_f | — | 30 | ns |

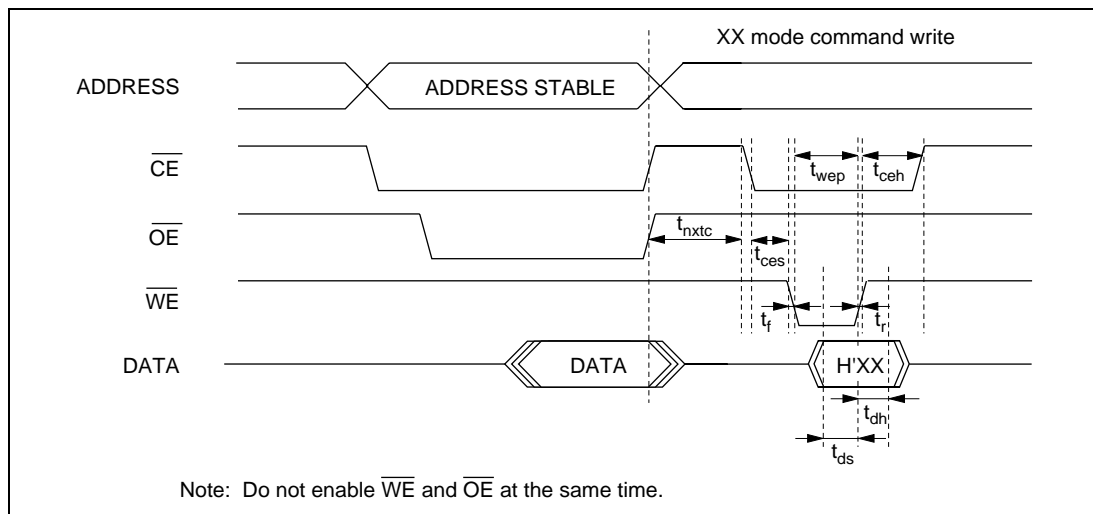
**Figure 8.17 Timing Waveforms when Entering Another Mode from Memory Read Mode**

Table 8.14 AC Characteristics in Memory Read Mode

(Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $V_{SS} = 0\text{ V}$, $T_a = 25^{\circ}\text{C} \pm 5^{\circ}\text{C}$)

| Item | Symbol | Min | Max | Unit |
|--|-----------|-----|-----|---------------|
| Access time | t_{acc} | — | 20 | μs |
| $\overline{\text{CE}}$ output delay time | t_{ce} | — | 150 | ns |
| $\overline{\text{OE}}$ output delay time | t_{oe} | — | 150 | ns |
| Output disable delay time | t_{df} | — | 100 | ns |
| Data output hold time | t_{oh} | 5 | — | ns |

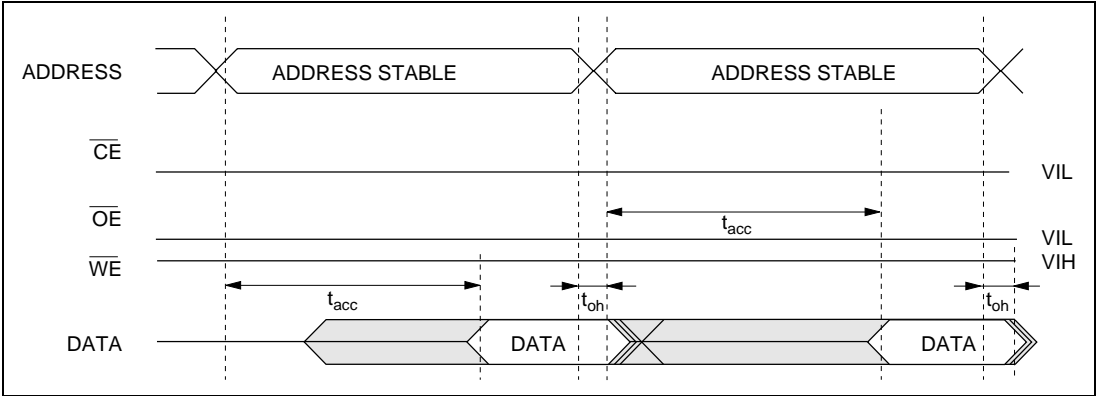


Figure 8.18 Timing Waveforms for $\overline{\text{CE}}/\overline{\text{OE}}$ Enable State Read

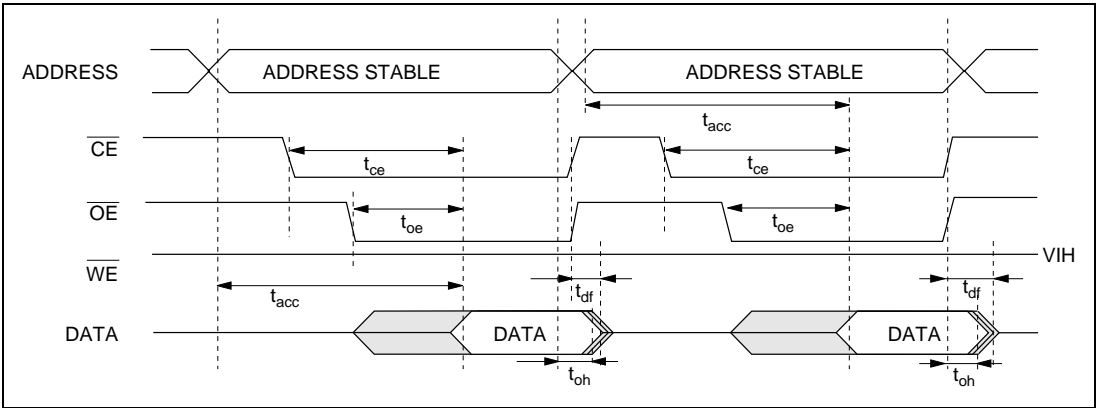


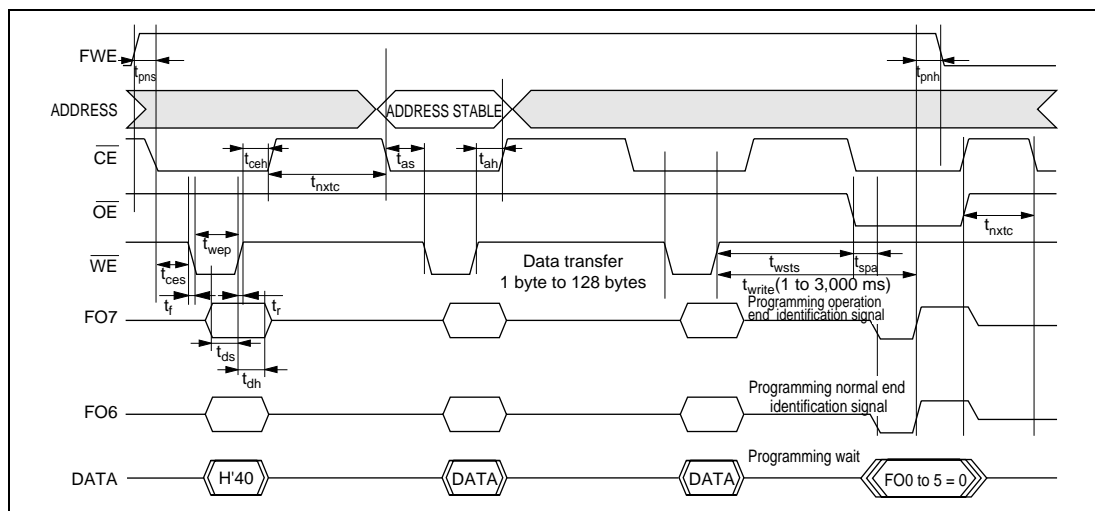
Figure 8.19 Timing Waveforms for $\overline{\text{CE}}/\overline{\text{OE}}$ Clocked Read

8.8.5 Auto-Program Mode

- (a) In auto-program mode, 128 bytes are programmed simultaneously. This should be carried out by executing 128 consecutive byte transfers.
- (b) A 128-byte data transfer is necessary even when programming fewer than 128 bytes. In this case, H'FF data must be written to the extra addresses.
- (c) The lower 8 bits of the transfer address must be H'00 or H'80. If a value other than an effective address is input, processing will switch to a memory write operation but a write error will be flagged.
- (d) Memory address transfer is performed in the second cycle (figure 8.20). Do not perform transfer after the second cycle.
- (e) Do not perform a command write during a programming operation.
- (f) Perform one auto-programming operation for a 128-byte block for each address. Characteristics are not guaranteed for two or more programming operations.
- (g) Confirm normal end of auto-programming by checking FO6. Alternatively, status read mode can also be used for this purpose (FO7 status polling uses the auto-program operation end identification pin).
- (h) The status polling FO6 and FO7 pin information is retained until the next command write. Until the next command write is performed, reading is possible by enabling $\overline{\text{CE}}$ and $\overline{\text{OE}}$.

Table 8.15 AC Characteristics in Auto-Program(Conditions: $V_{CC} = 5.0 \text{ V} \pm 10\%$, $V_{SS} = 0 \text{ V}$, $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit |
|-----------------------------------|-------------|-----|------|---------------|
| Command write cycle | t_{nxtc} | 20 | — | μs |
| $\overline{\text{CE}}$ hold time | t_{ceh} | 0 | — | ns |
| $\overline{\text{CE}}$ setup time | t_{ces} | 0 | — | ns |
| Data hold time | t_{dh} | 50 | — | ns |
| Data setup time | t_{ds} | 50 | — | ns |
| Write pulse width | t_{wep} | 70 | — | ns |
| Status polling start time | t_{wsts} | 1 | — | ms |
| Status polling access time | t_{spa} | — | 150 | ns |
| Address setup time | t_{as} | 0 | — | ns |
| Address hold time | t_{ah} | 60 | — | ns |
| Memory write time | t_{write} | 1 | 3000 | ms |
| $\overline{\text{WE}}$ rise time | t_r | — | 30 | ns |
| $\overline{\text{WE}}$ fall time | t_f | — | 30 | ns |
| Write setup time | t_{pns} | 100 | — | ns |
| Write end setup time | t_{pnh} | 100 | — | ns |

**Figure 8.20 Auto-Program Mode Timing Waveforms**

8.8.6 Auto-Erase Mode

- (a) Auto-erase mode supports only entire memory erasing.
- (b) Do not perform a command write during auto-erasing.
- (c) Confirm normal end of auto-erasing by checking FO6. Alternatively, status read mode can also be used for this purpose (FO7 status polling uses the auto-erase operation end identification pin).
- (d) The status polling FO6 and FO7 pin information is retained until the next command write. Until the next command write is performed, reading is possible by enabling $\overline{\text{CE}}$ and $\overline{\text{OE}}$.

Table 8.16 AC Characteristics in Auto-Erase Mode

(Conditions: $V_{\text{CC}} = 5.0 \text{ V} \pm 10\%$, $V_{\text{SS}} = 0 \text{ V}$, $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit |
|-----------------------------------|--------------------|-----|-------|---------------|
| Command write cycle | t_{nwlc} | 20 | — | μs |
| $\overline{\text{CE}}$ hold time | t_{ceh} | 0 | — | ns |
| $\overline{\text{CE}}$ setup time | t_{ces} | 0 | — | ns |
| Data hold time | t_{dh} | 50 | — | ns |
| Data setup time | t_{ds} | 50 | — | ns |
| Write pulse width | t_{wep} | 70 | — | ns |
| Status polling start time | t_{ests} | 1 | — | ms |
| Status polling access time | t_{spa} | — | 150 | ns |
| Memory erase time | t_{erase} | 100 | 40000 | ms |
| $\overline{\text{WE}}$ rise time | t_{r} | — | 30 | ns |
| $\overline{\text{WE}}$ fall time | t_{f} | — | 30 | ns |
| Erase setup time | t_{ens} | 100 | — | ns |
| Erase end setup time | t_{enh} | 100 | — | ns |

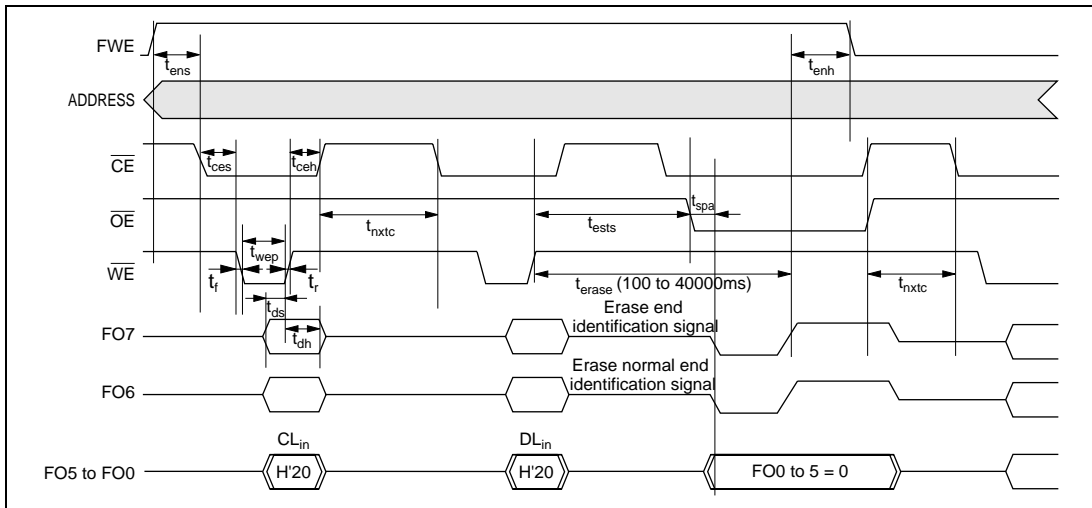


Figure 8.21 Auto-Erase Mode Timing Waveforms

8.8.7 Status Read Mode

- (1) Status read mode is used to identify what type of abnormal end has occurred. Use this mode when an abnormal end occurs in auto-program mode or auto-erase mode.
- (2) The return code is retained until a command write for other than status read mode is performed.

Table 8.17 AC Characteristics in Status Read Mode

(Conditions: $V_{CC} = 5.0 \text{ V} \pm 10\%$, $V_{SS} = 0 \text{ V}$, $T_a = 25^\circ\text{C} \pm 5^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit |
|--|------------|-----|-----|---------------|
| Command write cycle | t_{nxtc} | 20 | — | μs |
| $\overline{\text{CE}}$ hold time | t_{ceh} | 0 | — | ns |
| $\overline{\text{CE}}$ setup time | t_{ces} | 0 | — | ns |
| Data hold time | t_{dh} | 50 | — | ns |
| Data setup time | t_{ds} | 50 | — | ns |
| Write pulse width | t_{wep} | 70 | — | ns |
| $\overline{\text{OE}}$ output delay time | t_{oe} | — | 150 | ns |
| Disable delay time | t_{df} | — | 100 | ns |
| $\overline{\text{CE}}$ output delay time | t_{ce} | — | 150 | ns |
| $\overline{\text{WE}}$ rise time | t_r | — | 30 | ns |
| $\overline{\text{WE}}$ fall time | t_f | — | 30 | ns |

8.8.8 Status Polling

- (1) The FO7 status polling flag indicates the operating status in auto-program or auto-erase mode.
- (2) The FO6 status polling flag indicates a normal or abnormal end in auto-program or auto-erase mode.

Table 8.19 Status Polling Output Truth Table

| Pin Names | Internal Operation in Progress | Abnormal End | — | Normal End |
|------------------|---|---------------------|----------|-------------------|
| FO7 | 0 | 1 | 0 | 1 |
| FO6 | 0 | 0 | 1 | 1 |
| FO0 to FO5 | 0 | 0 | 0 | 0 |

8.8.9 Programmer Mode Transition Time

Commands cannot be accepted during the oscillation stabilization period or the programmer mode setup time. After the programmer mode setup time, a transition is made to memory read mode.

Table 8.20 Command Wait State Transition Time Specifications

| Item | Symbol | Min | Max | Unit |
|---|------------|-----|-----|------|
| Standby release (oscillation stabilization time) | t_{osc1} | 10 | — | ms |
| Programmer mode setup time | t_{bmV} | 10 | — | ms |
| V_{CC} hold time | t_{dwn} | 0 | — | ms |

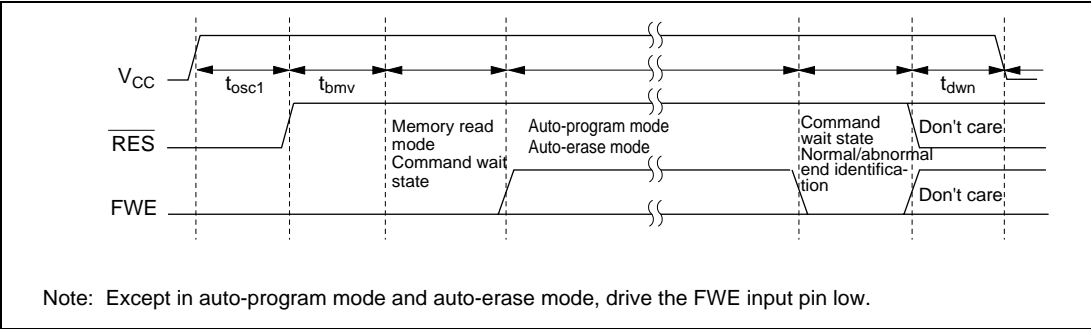


Figure 8.23 Oscillation Stabilization Time, Boot Program Transfer Time, and Power Supply Fall Sequence

8.8.10 Notes On Memory Programming

- (1) When programming addresses which have previously been programmed, carry out auto-erasing before auto-programming.
- (2) When performing programming using programmer mode on a chip that has been programmed/erased in an on-board programming mode, auto-erasing is recommended before carrying out auto-programming.

Notes: 1. The flash memory is initially in the erased state when the device is shipped by Hitachi. For other chips for which the erasure history is unknown, it is recommended that auto-erasing be executed to check and supplement the initialization (erase) level.

2. Auto-programming should be performed once only on the same address block.

8.9 Flash Memory Programming and Erasing Precautions

Precautions concerning the use of on-board programming mode and programmer mode are summarized below.

(1) Use the Specified Voltages and Timing for Programming and Erasing

Applied voltages in excess of the rating can permanently damage the device. Use a PROM programmer that supports Hitachi microcomputer device type with 256-kbyte on-chip flash memory.

Do not select the HN28F101 setting for the PROM programmer, and only use the specified socket adapter. Incorrect use will result in damaging the device.

(2) Powering On and Off

Do not apply a high level to the FWE pin until V_{CC} has stabilized. Also, drive the FWE pin low before turning off V_{CC} .

When applying or disconnecting V_{CC} , fix the FWE pin low and place the flash memory in the hardware protection state.

The power-on and power-off timing requirements should also be satisfied in the event of a power failure and subsequent recovery.

(3) FWE Application/Disconnection

FWE application should be carried out when MCU operation is in a stable condition. If MCU operation is not stable, fix the FWE pin low and set the protection state.

The following points must be observed concerning FWE application and disconnection to prevent unintentional programming or erasing of flash memory:

- (a) Apply FWE when the V_{CC} voltage has stabilized within its rated voltage range.
- (b) In boot mode, apply and disconnect FWE during a reset.
- (c) In user program mode, FWE can be switched between high and low level regardless of the reset state. FWE input can also be switched during program execution in flash memory.
- (d) Do not apply FWE if program runaway has occurred.
- (e) Disconnect FWE only when the SWE, ESU1, ESU2, PSU1, PSU2, EV1, EV2, PV1, PV2, P1, P2, E1 and E2 bits in FLMCR1 and FLMCR2 are cleared.
Make sure that the SWE, ESU1, ESU2, PSU1, PSU2, EV1, EV2, PV1, PV2, P1, P2, E1 and E2 bits are not set by mistake when applying or disconnecting FWE.

(4) Do Not Apply a Constant High Level to the FWE Pin

Apply a high level to the FWE pin only when programming or erasing flash memory. A system configuration in which a high level is constantly applied to the FWE should be avoided. Also, while a high level is applied to the FWE pin, the watchdog timer should be activated to prevent overprogramming or overerasing due to program runaway, etc.

(5) Use the Recommended Algorithm when Programming and Erasing Flash Memory

The recommended algorithm enables programming and erasing to be carried out without subjecting the device to voltage stress or sacrificing program data reliability. When setting the Pn or En bit in FLMCR1 and FLMCR2, the watchdog timer should be set beforehand as a precaution against program runaway, etc.

(6) Do Not Set or Clear the SWE Bit During Program Execution in Flash Memory

Clear the SWE bit before executing a program or reading data in flash memory.

When the SWE bit is set, data in flash memory can be rewritten, but flash memory should only be accessed for verify operations (verification during programming/erasing).

(7) Do Not Use Interrupts while Flash Memory is Being Programmed or Erased

All interrupt requests, including NMI, should be disabled during FWE application to give priority to program/erase operations.

(8) Do Not Perform Additional Programming. Erase the Memory before Reprogramming.

In on-board programming, perform only one programming operation on a 32-byte programming unit block. In programmer mode, too, perform only one programming operation on a 128-byte programming unit block. Programming should be carried out with the entire programming unit block erased.

(9) Before Programming, Check that the Chip is Correctly Mounted in the PROM Programmer.

Overcurrent damage to the device can result if the index marks on the PROM programmer socket, socket adapter, and chip are not correctly aligned.

(10) Do Not Touch the Socket Adapter or Chip During Programming.

Touching either of these can cause contact faults and write errors.

8.10 Note on Switching from F-ZTAT Version to Mask ROM Version

The mask ROM version does not have the internal registers for flash memory control that are provided in the F-ZTAT version. Table 8.21 lists the registers that are present in the F-ZTAT version but not in the mask ROM version. If a register listed in table 8.21 is read in the mask ROM version, an undefined value will be returned. Therefore, if application software developed on the F-ZTAT version is switched to a mask ROM version product, it must be modified to ensure that the registers in table 8.21 have no effect.

Table 8.21 Registers Present in F-ZTAT Version but Absent in Mask ROM Version

| Register | Abbreviation | Address |
|---------------------------------|--------------|---------|
| Flash memory control register 1 | FLMCR1 | H'FFF8 |
| Flash memory control register 2 | FLMCR2 | H'FFF9 |
| Erase block register 1 | EBR1 | H'FFFA |
| Erase block register 2 | EBR2 | H'FFFB |

Section 9 RAM

9.1 Overview

The H8S/2194C, H8S/2194B, and H8S/2194A have 6 kbytes, and the H8S/2194, H8S/2193, H8S/2192 and H8S/2191 have 3 kbytes, of on-chip high-speed static RAM. The on-chip RAM is connected to the CPU by a 16-bit data bus, enabling both byte data and word data to be accessed in one state. This makes it possible to perform fast word data transfer.

9.1.1 Block Diagram

Figure 9.1 shows a block diagram of the on-chip RAM.

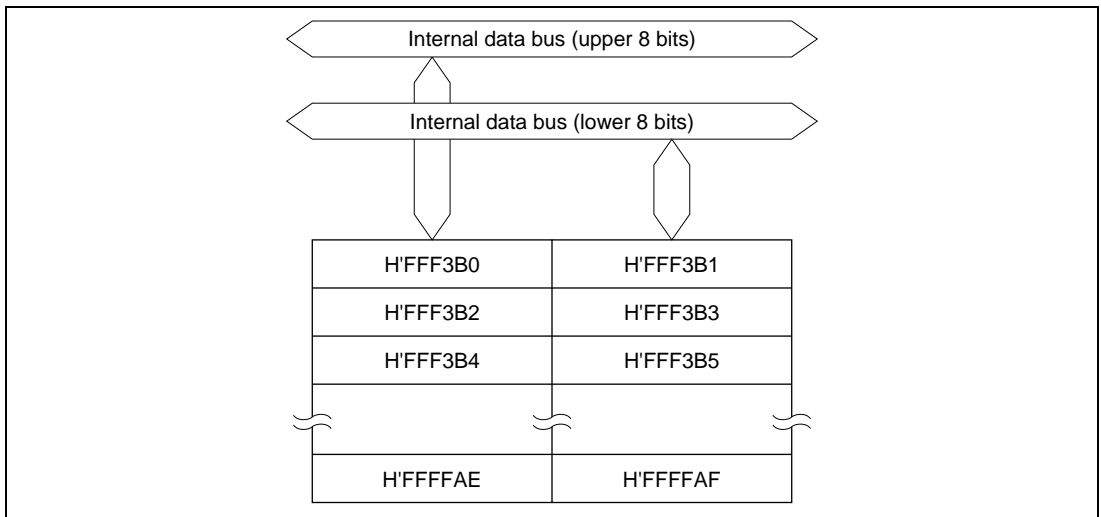


Figure 9.1 Block Diagram of RAM (H8S/2194)

Section 10 Clock Pulse Generator

10.1 Overview

This LSI has a built-in clock pulse generator (CPG) that generates the system clock (ϕ), the bus master clock, and internal clocks.

The clock pulse generator consists of a system clock oscillator, a duty adjustment circuit, clock selection circuit, medium-speed clock divider, subclock oscillator, and subclock division circuit.

10.1.1 Block Diagram

Figure 10.1 shows a block diagram of the clock pulse generator.

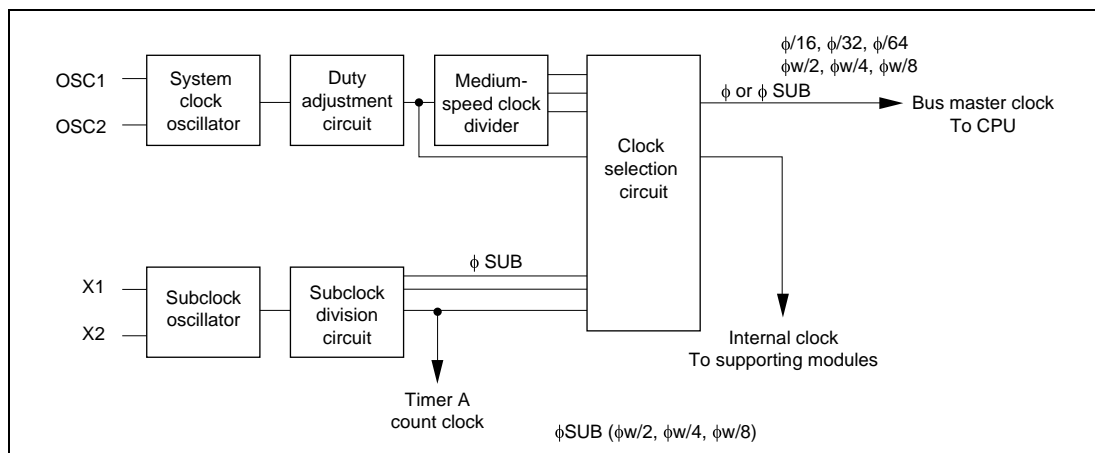


Figure 10.1 Block Diagram of Clock Pulse Generator

10.1.2 Register Configuration

The clock pulse generator is controlled by SBYCR and LPWRCR. Table 10.1 shows the register configuration.

Table 10.1 CPG Registers

| Name | Abbreviation | R/W | Initial Value | Address* |
|----------------------------|--------------|-----|---------------|----------|
| Standby control register | SBYCR | R/W | H'00 | H'FFEA |
| Low-power control register | LPWRCR | R/W | H'00 | H'FFEB |

Note: * Lower 16 bits of the address.

10.2 Register Descriptions

10.2.1 Standby Control Register (SBYCR)

| | | | | | | | | |
|-----------------|------|------|------|------|---|---|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | SSBY | STS2 | STS1 | STS0 | — | — | SCK1 | SCK0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | — | — | R/W | R/W |

SBYCR is an 8-bit readable/writable register that performs power-down mode control. Only bits 0 and 1 are described here. For a description of the other bits, see section 4.2.1, Standby Register (SBYCR). SBYCR is initialized to H'00 by a reset.

Bits 1 and 0: System Clock Select 1 and 0 (SCK1, SCK0)

These bits select the bus master clock for high-speed mode and medium-speed mode.

| Bit 1 | Bit 0 | Description |
|-------|-------|--|
| SCK1 | SCK0 | |
| 0 | 0 | Bus master is in high-speed mode (Initial value) |
| | 1 | Medium-speed clock is $\phi/16$ |
| 1 | 0 | Medium-speed clock is $\phi/32$ |
| | 1 | Medium-speed clock is $\phi/64$ |

10.2.2 Low-Power Control Register (LPWRCR)

| | | | | | | | | |
|-----------------|------|------|-------|---|---|---|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | DTON | LSON | NESEL | — | — | — | SA1 | SA0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | — | — | — | R/W | R/W |

LPWRCR is an 8-bit readable/writable register that performs power-down mode control. Only bit 1 and 0 is described here. For a description of the other bits, see section 4.2.2, Low-Power Control Register (LPWRCR).

LPWRCR is initialized to H'00 by a reset.

Bits 1 and 0: Subactive Mode Clock Select (SA1, SA0)

Selects CPU clock for subactive mode. In subactive mode, writes are disabled.

| Bit 1 | Bit 0 | Description |
|-------|-------|---|
| SA1 | SA0 | |
| 0 | 0 | CPU operating clock is $\phi_w/8$ (Initial value) |
| | 1 | CPU operating clock is $\phi_w/4$ |
| 1 | * | CPU operating clock is $\phi_w/2$ |

Note: * Don't care

10.3 Oscillator

Clock pulses can be supplied by connecting a crystal resonator, or by input of an external clock.

10.3.1 Connecting a Crystal Resonator

(1) Circuit Configuration

A crystal resonator can be connected as shown in the example in figure 10.2. An AT-cut parallel-resonance crystal should be used.

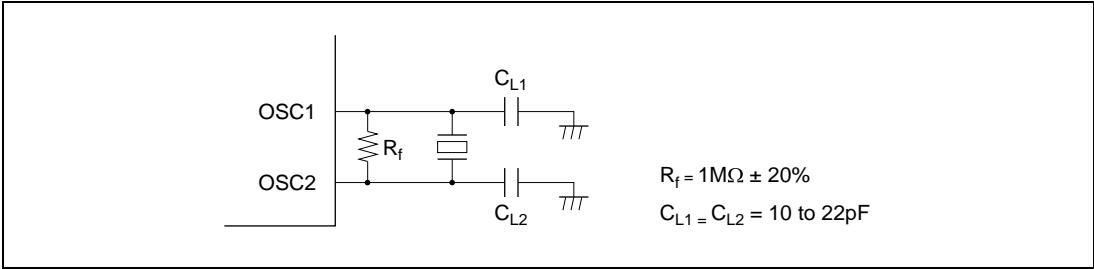


Figure 10.2 Connection of Crystal Resonator (Example)

(2) Crystal Resonator

Figure 10.3 shows the equivalent circuit of the crystal resonator. Use a crystal resonator that has the characteristics shown in table 10.2 and the same frequency as the system clock (ϕ).

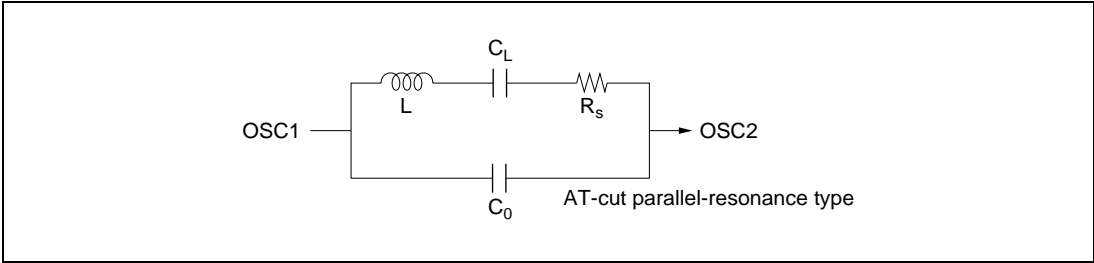


Figure 10.3 Crystal Resonator Equivalent Circuit

Table 10.2 Crystal Resonator Parameters

| Frequency (MHz) | 8 | 10 |
|--------------------------|----|----|
| $R_s\text{max} (\Omega)$ | 80 | 60 |
| $C_o\text{max} (pF)$ | 7 | 7 |

(3) Note on Board Design

When a crystal resonator is connected, the following points should be noted.

Other signal lines should be routed away from the oscillator circuit to prevent induction from interfering with correct oscillation. See figure 10.4.

When designing the board, place the crystal resonator and its load capacitors as close as possible to the OSC1 and OSC2 pins.

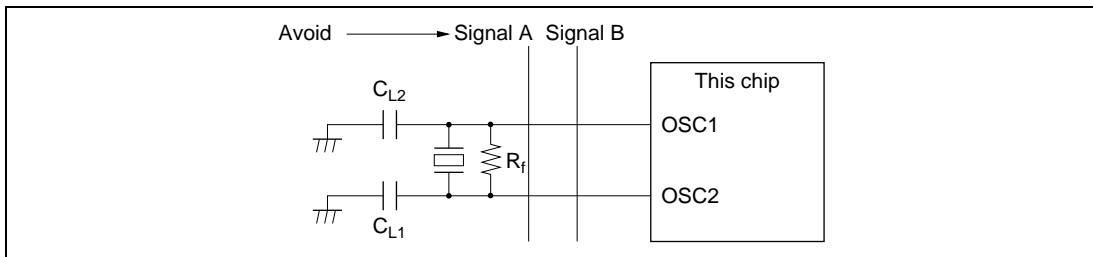


Figure 10.4 Example of Incorrect Board Design

10.3.2 External Clock Input

(1) Circuit Configuration

An external clock signal can be input as shown in the examples in figure 10.5. If the OSC2 pin is left open, make sure that stray capacitance is no more than 10 pF.

In example (b), make sure that the external clock is held high in standby mode, subactive mode, subsleep mode, and watch mode.

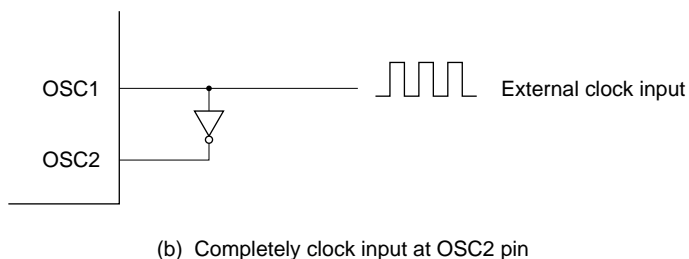
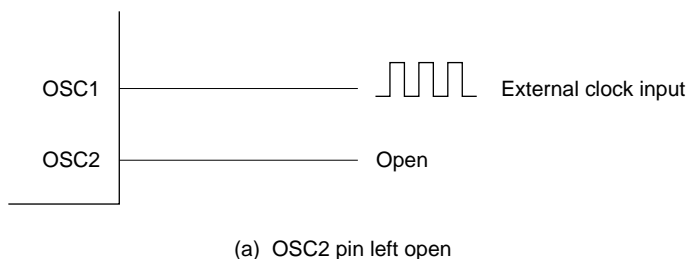


Figure 10.5 External Clock Input (Examples)

(2) External Clock

The external clock signal should have the same frequency as the system clock (ϕ).

Table 10.3 and figure 10.6 show the input conditions for the external clock.

Table 10.3 External Clock Input Conditions

| Item | Symbol | $V_{CC} = 4.0 \text{ to } 5.5 \text{ V}$ | | Unit | Test Conditions |
|---------------------------------------|-----------|--|-----|------|-----------------|
| | | Min | Max | | |
| External clock input low pulse width | t_{CPL} | 40 | — | ns | Figure 10.6 |
| External clock input high pulse width | t_{CPH} | 40 | — | ns | |
| External clock rise time | t_{CPr} | — | 10 | ns | |
| External clock fall time | t_{CPf} | — | 10 | ns | |

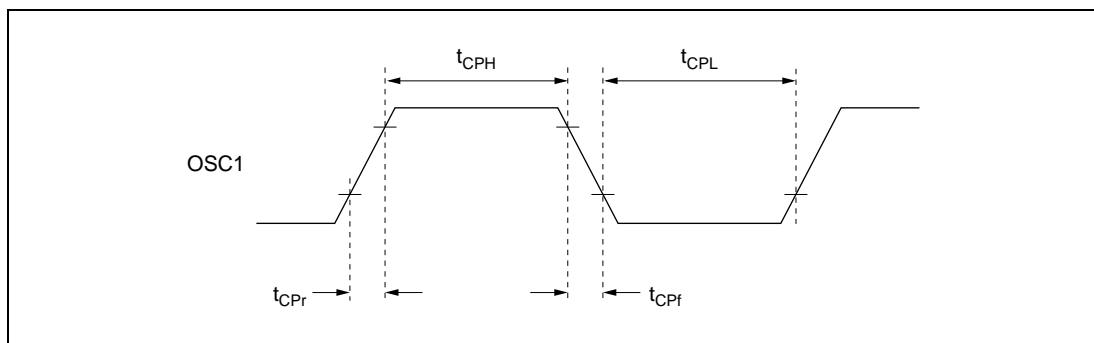


Figure 10.6 External Clock Input Timing

Table 10.4 shows the external clock output settling delay time, and figure 10.7 shows the external clock output settling delay timing. The oscillator and duty adjustment circuit have a function for adjusting the waveform of the external clock input at the OSC1 pin. When the prescribed clock signal is input at the OSC1 pin, internal clock signal output is fixed after the elapse of the external clock output settling delay time (t_{DEXT}). As the clock signal output is not fixed during the t_{DEXT} period, the reset signal should be driven low to maintain the reset state.

Table 10.4 External Clock Output Settling Delay Time

(Conditions: $V_{CC} = 4.0\text{ V}$ to 5.5 V , $AV_{CC} = 4.0\text{ V}$ to 5.5 V , $V_{SS} = AV_{SS} = 0\text{ V}$)

| Item | Symbol | Min | Max | Unit | Notes |
|---|---------------------|-----|-----|---------------|-------------|
| External clock output settling delay time | t_{DEXT}^* | 500 | — | μs | Figure 10.7 |

Note: * t_{DEXT} includes 20 t_{CYC} of $\overline{\text{RES}}$ pulse width (t_{RESW}).

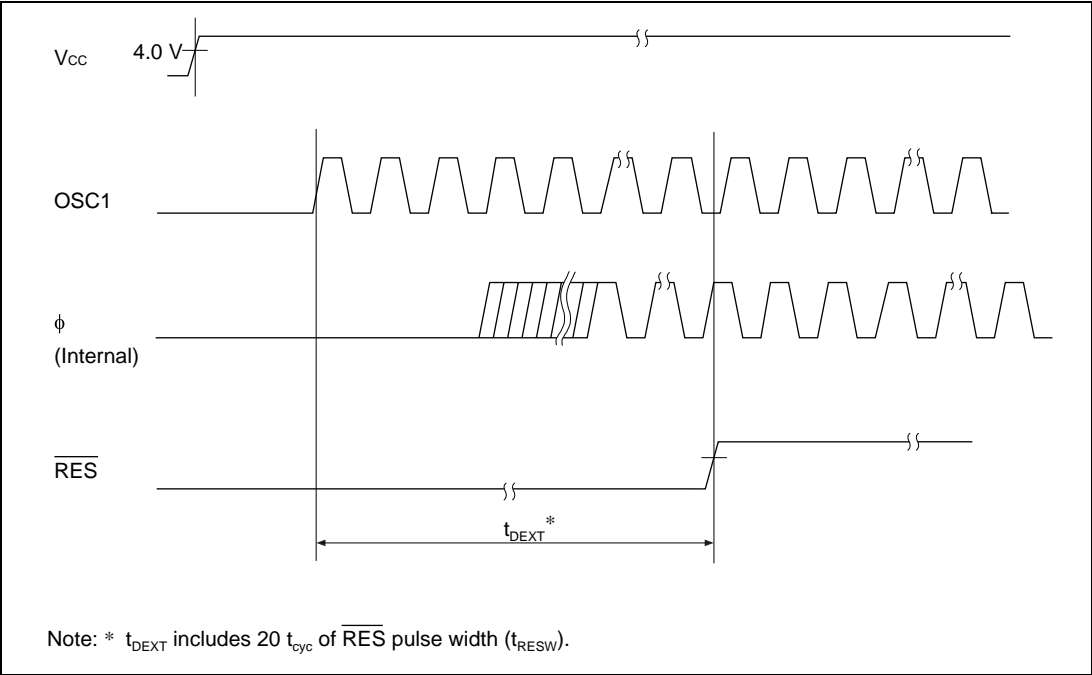


Figure 10.7 External Clock Output Settling Delay Timing

10.4 Duty Adjustment Circuit

When the oscillator frequency is 5 MHz or higher, the duty adjustment circuit adjusts the duty cycle of the clock signal from the oscillator to generate the system clock (ϕ).

10.5 Medium-Speed Clock Divider

The medium-speed divider divides the system clock to generate $\phi/16$, $\phi/32$, and $\phi/64$ clocks.

10.6 Bus Master Clock Selection Circuit

The bus master clock selection circuit selects the system clock (ϕ) or one of the medium-speed clocks ($\phi/16$, $\phi/32$ or $\phi/64$) to be supplied to the bus master (CPU), according to the settings of bits SCK2 to SCK0 in SBYCR.

10.7 Subclock Oscillator Circuit

10.7.1 Connecting 32.768 kHz Crystal Resonator

When using a subclock, connect a 32.768 kHz crystal resonator to X1 and X2 pins as shown in figure 10.8.

For precautions on connecting, see section 10.3.1 (3), Note on Board Design.

The subclock input conditions are shown in figure 10.10.

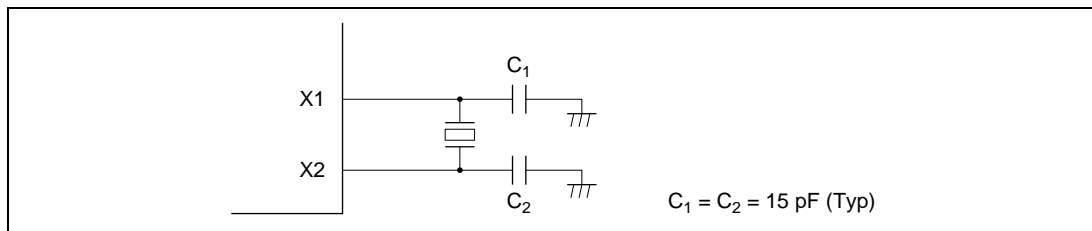


Figure 10.8 Connecting a 32.768 kHz Crystal Resonator (Example)

Figure 10.9 shows a crystal resonator equivalent circuit.

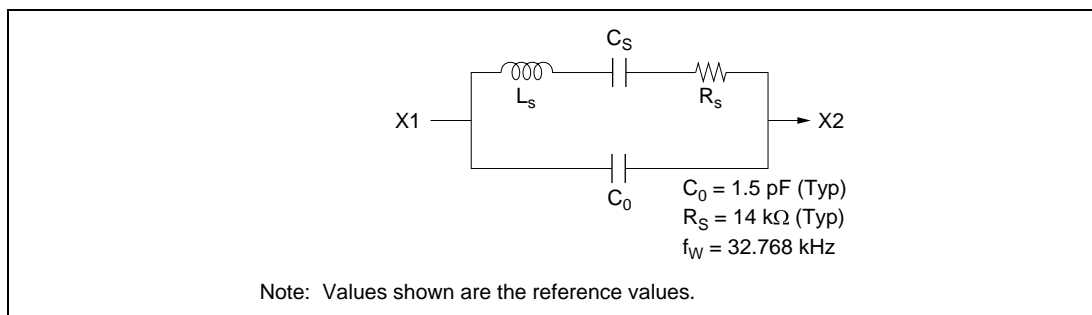


Figure 10.9 32.768 kHz Crystal Resonator Equivalent Circuit

10.7.2 External Clock Input

(1) Circuit Configuration

When external clock input connect to the X1 pin, and X2 pin should remain open as connection example of figure 10.10.

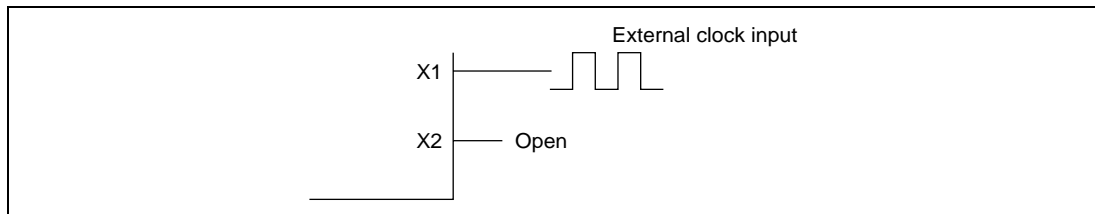


Figure 10.10 Connection Example When Inputting External Clock

10.7.3 When Subclock is not Needed

Connect X1 pin to V_{CC} , and X2 pin should remain open as shown in figure 10.11.

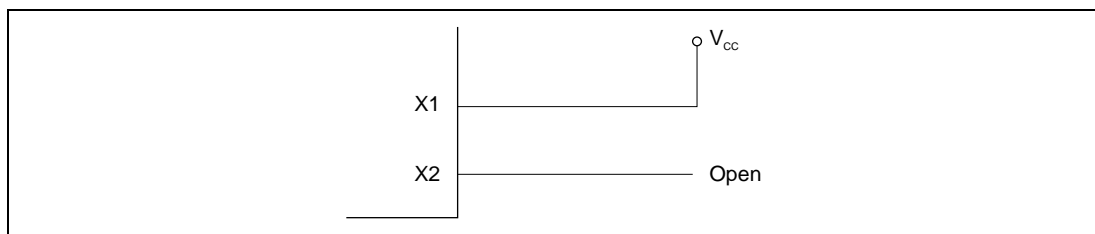


Figure 10.11 Terminal When Subclock is not Needed

10.8 Subclock Waveform Shaping Circuit

To eliminate noise in the subclock input from the X1 pin, this circuit samples the clock using a clock obtained by dividing the ϕ clock. The sampling frequency is set with the NESEL bit in LPWRCR. For details, see section 4.2.2, Low-Power Control Register (LPWRCR). The clock is not sampled in subactive mode, subsleep mode, or watch mode.

10.9 Notes on the Resonator

Resonator characteristics are closely related to the user board design. Perform appropriate assessment of resonator connection, mask version and F-ZTAT, by referring to the connection example given in this section. The resonator circuit rate differs depending on the free capacity of the resonator and the execution circuit, so consult with the resonator manufacturer before determination. Make sure the voltage applied to the resonator pin does not exceed the maximum rated voltage.

Section 11 I/O Port

11.1 Overview

11.1.1 Port Functions

This LSI has seven 8-bit I/O ports (including one CMOS high-current port), one 4-bit I/O port, and one 8-bit input port. Table 11.1 shows the functions of each port. Each I/O port has a port control register (PCR) that controls an input and output and a port data register (PDR) for storing output data. The input and output can be controlled in a unit of bit. The pin whose peripheral function is used both as an alternative function can set the pin function in a unit of bit by a port mode register (PMR).

11.1.2 Port Input

(1) Reading a Port

- When a general port of PCR = 0 (input) is read, the pin level is read.
- When a general port of PCR = 1 (output) is read, the value of the corresponding PDR bit is read.
- When the pins (excluding AN7 to AN0 and RP7 to RP0 pins) set to the peripheral function are read, the results are as given in items (1) and (2) according to the PCR value.

(2) Processing Input Pins

The general input port or general I/O port is gated by read signals. Unused pins can be left open if they are not read. However, if an open pin is read, a feedthrough current may apply during the read period according to an intermediate level. The read period is about one-state. Relevant ports: P0, P1, P2, P3, P4, P5, P6, P7, P8

When an alternative pin is set to an alternative function other than the general I/O, always set the pin level to a high or low level. If the pin is left open, a feedthrough current applies according to an intermediate level, which adversely affects reliability, causes malfunctions, and in the worst case may damage the pin.

Because the PMR is not initialized in low power consumption mode, pay attention to the pin input level after the mode has been shifted to the low power consumption mode.

Relevant pins: \overline{IC} , $\overline{IRQ0}$ to $\overline{IRQ5}$, SCK1, SCK2, SI1, SI2, \overline{CS} , FTIA, FTIB, FTIC, FTID, TRIG, TMBI, \overline{ADTRG} , EXCAP, EXTTRG

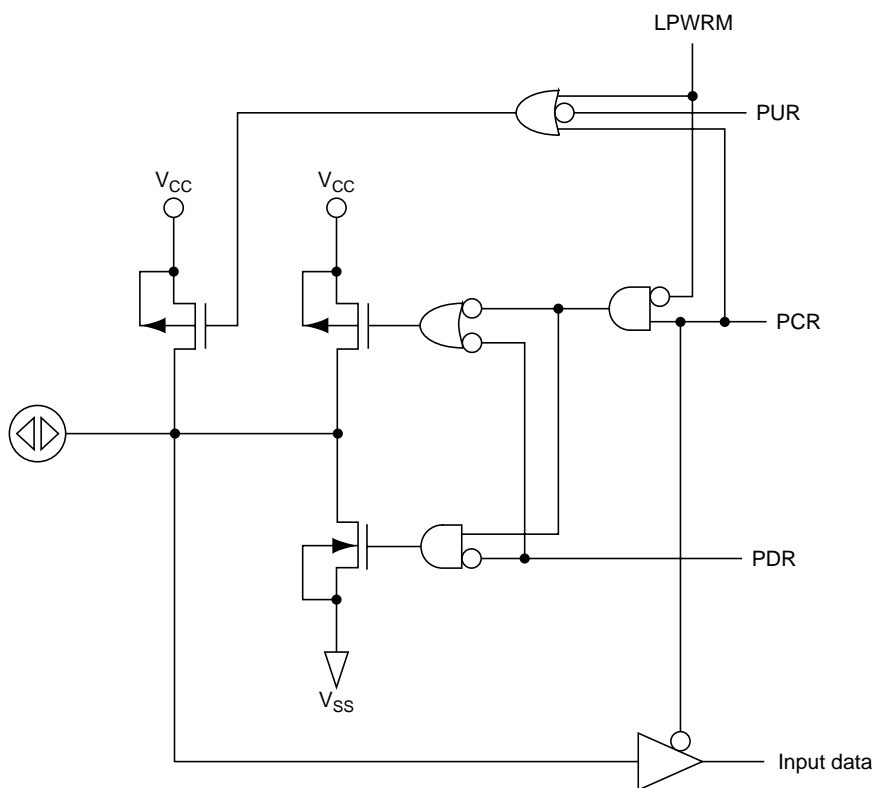
Table 11.1 Port Functions

| Port | Description | Pins | Alternative Functions | Function Switching Register |
|--------|---|-----------------------------|--|-----------------------------|
| Port 0 | P07 to P00 input-only ports | P70/AN7 to P00/AN0 | Analog data input channels 7 to 0 | PMR0 |
| Port 1 | P17 to P10 I/O ports (Built-in MOS pull-up transistors) | P17/TMOW | Prescaler unit frequency division clock output | PMR1 |
| | | P16/ $\overline{\text{IC}}$ | Prescaler unit input capture input | |
| | | P15/IRQ5 to P10/IRQ0 | External interrupt request input | |
| Port 2 | P27 to P20 I/O ports (Built-in MOS pull-up transistors) | P27/SCK2 | SCI2 clock I/O | PMR2 |
| | | P26/SO2 | SCI2 transmit data output | |
| | | P25/SI2 | SCI2 receive data input | |
| | | P24/SCL | I ² C bus interface clock I/O | ICCR |
| | | P23/SDA | I ² C bus interface data I/O | |
| | | P22/SCK1 | SCI1 clock I/O | SMR SCR |
| | | P21/SO1 | SCI1 transmit data output | |
| Port 3 | P37 to P30 I/O ports (Built-in MOS pull-up transistors) | P20/SI1 | SCI1 receive data input | PMR3 |
| | | P37/TMO | Timer J timer output | |
| | | P36/BUZZ | Timer J buzzer output | |
| | | P35/PWM3 to P32/PWM0 | 8-bit PWM output | |
| | | P31/STRB | SCI2 strobe output | |
| Port 4 | P47 to P40 I/O ports | P30/ $\overline{\text{CS}}$ | SCI2 chip select input | TOCR |
| | | P47 | None | |
| | | P46/FTOB | Timer X output compare B output | |
| | | P45/FTOA | Timer X output compare A output | |
| | | P44/FTID | Timer X input capture D input | |
| | | P43/FTIC | Timer X input capture C input | |
| | | P42/FTIB | Timer X input capture B input | |
| | | P41/FTIA | Timer X input capture A input | |
| Port 5 | P53 to P50 I/O ports | P40/PWM14 | 14-bit PWM output | PMR4 |
| | | P53/TRIG | Realtime output port trigger input | PMR5 |
| | | P52/TMBI | Timer B event input | |
| | | P51 | None | ADTSR |
| Port 6 | P67 to P60 I/O ports | P50/ADTRG | A/D conversion start external trigger input | |
| | | P67/RP7 to P60/RP0 | Realtime output port | PMR6 |
| Port 7 | P77 to P70 I/O ports | P77/PPG7 to P70/PPG0 | PPG output | PMR7 |
| Port 8 | P87 to P80 I/O ports (High-current ports) | P87 to P84 | None | PMR8 |
| | | P83/SV2 | Servo monitor output | |
| | | P82/SV1 | | |
| | | P81/EXCAP | Capstan external synchronous signal input | |
| | | P80/EXTTRG | External trigger signal input | |

11.1.3 MOS Pull-Up Transistors

The MOS pull-up transistors in ports 1 to 3 can be switched on or off by the MOS pull-up select registers 1 to 3 (PUR1 to PUR3) in units of bits. Settings in PUR1 to PUR3 are valid when the pin function is set to an input by PCR1 to PCR3. If the pin function is set to an output, the MOS pull-up transistor is turned off. Figure 11.1 shows the circuit configuration of a pin with a MOS pull-up transistor.

When the pin whose peripheral function is used both as an alternative function is set to the alternative output function, the MOS pull-up transistor is turned off. When the pin is set to the alternative input function, the MOS pull-up transistor is controlled according to the PUR setting regardless of PCR.



LPWRM: Low power consumption mode signal
(The MOS pull-up transistor is turned off in reset, standby, and watch modes.)

PUR : MOS pull-up select register

PCR : Port control register

PDR : Port data register

Figure 11.1 Circuit Configuration of Pin with MOS Pull-Up Transistor

11.2 Port 0

11.2.1 Overview

Port 0 is an 8-bit input-only port. Table 11.2 shows the port 0 configuration. Port 0 consists of pins that are used both as standard input ports (P07 to P00) and analog input channels (AN7 to AN0). It is switched by port mode register 0 (PMR0).

Table 11.2 Port 0 Configuration

| Port | Function | Alternative Function |
|--------|---------------------------|----------------------------|
| Port 0 | P07 (standard input port) | AN7 (analog input channel) |
| | P06 (standard input port) | AN6 (analog input channel) |
| | P05 (standard input port) | AN5 (analog input channel) |
| | P04 (standard input port) | AN4 (analog input channel) |
| | P03 (standard input port) | AN3 (analog input channel) |
| | P02 (standard input port) | AN2 (analog input channel) |
| | P01 (standard input port) | AN1 (analog input channel) |
| | P00 (standard input port) | AN0 (analog input channel) |

11.2.2 Register Configuration

Table 11.3 shows the port 0 register configuration.

Table 11.3 Port 0 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|----------------------|---------|-----|------|---------------|----------|
| Port mode register 0 | PMR0 | R/W | Byte | H'00 | H'FFCD |
| Port data register 0 | PDR0 | R | Byte | — | H'FFC0 |

Note: * Lower 16 bits of the address.

(1) Port Mode Register 0 (PMR0)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR07 | PMR06 | PMR05 | PMR04 | PMR03 | PMR02 | PMR01 | PMR00 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port mode register 0 (PMR0) controls switching of each pin function of port 0. The switching is specified in a unit of bit.

PMR0 is an 8-bit read/write enable register. When reset, PMR0 is initialized to H'00.

Bits 7 to 0: P07/AN7 to P00/AN0 Pin Switching (PMR07 to PMR00)

PMR07 to PMR00 sets whether the P0n/ANn pin is used as a P0n input pin or an ANn pin for the analog input channel of an A/D converter.

Bit n

| PMR0n | Description |
|-------|--|
| 0 | The P0n/ANn pin functions as a P0n input pin (Initial value) |
| 1 | The P0n/ANn pin functions as an ANn input pin |

(n = 7 to 0)

(2) Port Data Register 0 (PDR0)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR07 | PDR06 | PDR05 | PDR04 | PDR03 | PDR02 | PDR01 | PDR00 |
| Initial value : | — | — | — | — | — | — | — | — |
| R/W : | R | R | R | R | R | R | R | R |

Port data register 0 (PDR0) reads the port states. When the corresponding bit of PMR0 is 0 (general input port), the pin state is read if PDR0 is read. When the corresponding bit of PMR0 is 1 (analog input channel), 1 is read if PDR0 is read.

PDR0 is an 8-bit read-only register. When PDR0 is reset, its values become undefined.

11.2.3 Pin Functions

This section describes the pin functions of port 0 and their selection methods.

(1) P07/AN7 to P00/AN0

P07/AN7 to P00/AN0 are switched according to the PMR0n bit of PMR0 as shown below.

| PMR0n | Pin Function |
|-------|---------------|
| 0 | P0n input pin |
| 1 | ANn input pin |

(n = 7 to 0)

11.2.4 Pin States

Table 11.4 shows the pin 0 states in each operation mode.

Table 11.4 Port 0 Pin States

| Pins | Reset | Active | Sleep | Standby | Watch | Subactive | Subsleep |
|--------------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| PN7/AN7 to P00/AN0 | High-impedance | High-impedance | High-impedance | High-impedance | High-impedance | High-impedance | High-impedance |

11.3 Port 1

11.3.1 Overview

Port 1 is an 8-bit I/O port. Table 11.5 shows the port 1 configuration.

Port 1 consists of pins that are used both as standard I/O ports (P17 to P10) and frequency division clock output (TMOW), input capture input (\overline{IC}), or external interrupt request inputs ($\overline{IRQ5}$ to $\overline{IRQ0}$). It is switched by port mode register 1 (PMR1) and port control register 1 (PCR1).

Port 1 can select the functions of MOS pull-up transistors.

Table 11.5 Port 1 Configuration

| Port | Function | Alternative Function |
|--------|-------------------------|--|
| Port 1 | P17 (standard I/O port) | TMOW (frequency division clock output) |
| | P16 (standard I/O port) | \overline{IC} (input capture input) |
| | P15 (standard I/O port) | $\overline{IRQ5}$ (external interrupt request input) |
| | P14 (standard I/O port) | $\overline{IRQ4}$ (external interrupt request input) |
| | P13 (standard I/O port) | $\overline{IRQ3}$ (external interrupt request input) |
| | P12 (standard I/O port) | $\overline{IRQ2}$ (external interrupt request input) |
| | P11 (standard I/O port) | $\overline{IRQ1}$ (external interrupt request input) |
| | P10 (standard I/O port) | $\overline{IRQ0}$ (external interrupt request input) |

11.3.2 Register Configuration

Table 11.6 shows the port 1 register configuration.

Table 11.6 Port 1 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-------------------------------|---------|-----|------|---------------|----------|
| Port mode register 1 | PMR1 | R/W | Byte | H'00 | H'FFCE |
| Port control register 1 | PCR1 | W | Byte | H'00 | H'FFD1 |
| Port data register 1 | PDR1 | R/W | Byte | H'00 | H'FFC1 |
| MOS pull-up select register 1 | PUR1 | R/W | Byte | H'00 | H'FFE1 |

Note: * Lower 16 bits of the address.

(1) Port Mode Register 1 (PMR1)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR17 | PMR16 | PMR15 | PMR14 | PMR13 | PMR12 | PMR11 | PMR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port mode register 1 (PMR1) controls switching of each pin function of port 1. The switching is specified in a unit of bit.

PMR1 is an 8-bit read/write enable register. When reset, PMR1 is initialized to H'00.

Note the following items when the pin functions are switched by PMR1.

- (1) If port 1 is set to an \overline{IC} input pin and $\overline{IRQ5}$ to $\overline{IRQ0}$ by PMR1, the pin level needs be set to the high or low level regardless of the active mode and low power consumption mode. The pin level must not be set to an intermediate level.
- (2) When the pin functions of P16/ \overline{IC} and P15/ $\overline{IRQ5}$ to P10/ $\overline{IRQ0}$ are switched by PMR1, they are incorrectly recognized as edge detection according to the state of a pin signal and a detection signal may be generated. To prevent this, perform the operation in the following procedure.
 - (a) Before switching the pin functions, inhibit an interrupt enable flag from being interrupted.
 - (b) After having switched the pin functions, clear the relevant interrupt request flag to 0 by a single instruction.

(Program Example)

```

:
MOV.B ROL,@IENR ..... Interrupt disabled
MOV.B R1L,@PMR1 ..... Pin function change
NOP ..... Optional instruction
BCLR m @IRQR ..... Applicable interrupt clear
MOV.B R1L,@IENR ..... Interrupt enabled
:

```

Bit 7: P17/TMOW Pin Switching (PMR17)

PMR17 sets whether the P17/TMOW pin is used as a P17 I/O pin or a TMOW pin for the frequency division clock output.

Bit 7

| PMR17 | Description |
|-------|---|
| 0 | The P17/TMOW pin functions as a P17 I/O pin (Initial value) |
| 1 | The P17/TMOW pin functions as a TMOW output pin |

Bit 6: P16/ $\overline{\text{IC}}$ Pin Switching (PMR16)

PMR16 sets whether the P16/ $\overline{\text{IC}}$ pin as a P16 I/O pin or an $\overline{\text{IC}}$ pin for the input capture input of the prescaler unit. The $\overline{\text{IC}}$ pin has a built-in noise cancel circuit. See section 21, Prescaler Unit.

Bit 6

| PMR16 | Description |
|-------|--|
| 0 | The P16/ $\overline{\text{IC}}$ pin functions as a P16 I/O pin (Initial value) |
| 1 | The P16/ $\overline{\text{IC}}$ pin functions as an $\overline{\text{IC}}$ input pin |

Bits 5 to 0: P15/ $\overline{\text{IRQ5}}$ to P10/ $\overline{\text{IRQ0}}$ Pin Switching (PMR15 to PMR10)

PMR15 to PMR10 set whether the P1n/ $\overline{\text{IRQn}}$ pin is used as a P1n I/O pin or an $\overline{\text{IRQn}}$ pin for the external interrupt request input.

Bit n

| PMR1n | Description |
|-------|--|
| 0 | The P1n/ $\overline{\text{IRQn}}$ pin functions as a P1n I/O pin (Initial value) |
| 1 | The P1n/ $\overline{\text{IRQn}}$ pin functions as an $\overline{\text{IRQn}}$ input pin |

(n = 5 to 0)

(2) Port Control Register 1 (PCR1)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR17 | PCR16 | PCR15 | PCR14 | PCR13 | PCR12 | PCR11 | PCR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

Port control register 1 (PCR1) controls the I/Os of pins P17 to P10 of port 1 in a unit of bit.

When PCR1 is set to 1, the corresponding P17 to P10 pins become output pins, and when it is set to 0, they become input pins. When the relevant pin is set to a general I/O by PMR1, settings of PCR1 and PDR1 become valid.

PCR1 is an 8-bit write-only register. When PCR1 is read, 1 is read. When reset, PCR1 is initialized to H'00.

Bit n

| PCR1n | Description |
|-------|---|
| 0 | The P1n pin functions as an input pin (Initial value) |
| 1 | The P1n pin functions as an output pin |

(n = 7 to 0)

(3) Port Data Register 1 (PDR1)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR17 | PDR16 | PDR15 | PDR14 | PDR13 | PDR12 | PDR11 | PDR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port data register 1 (PDR1) stores the data for the pins P17 to P10 of port 1. When PCR1 is 1 (output), the PDR1 values are directly read if port 1 is read. Accordingly, the pin states are not affected. When PCR1 is 0 (input), the pin states are read if port 1 is read. PDR1 is an 8-bit read/ write enable register. When reset, PDR1 is initialized to H'00.

(4) MOS Pull-Up Select Register 1 (PUR1)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PUR17 | PUR16 | PUR15 | PUR14 | PUR13 | PUR12 | PUR11 | PUR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MOS pull-up selector register 1 (PUR1) controls the on and off of the MOS pull-up transistor of port 1. Only the pin whose corresponding bit of PCR1 was set to 0 (input) becomes valid. When the corresponding bit of PCR1 is set to 1 (output), the corresponding bit of PUR1 becomes invalid and the MOS pull-up transistor is turned off. PUR1 is an 8-bit read/ write enable register. When reset, PUR1 is initialized to H'00.

Bit n

| PUR1n | Description |
|-------|---|
| 0 | The P1n pin has no MOS pull-up transistor (Initial value) |
| 1 | The P1n pin has a MOS pull-up pin |

(n = 7 to 0)

11.3.3 Pin Functions

This section describes the port 1 pin functions and their selection methods.

(1) P17/TMOW

P17/TMOW is switched as shown below according to the PMR17 bit in PMR1 and the PCR17 bit in PCR1.

| PMR17 | PCR17 | Pin Function |
|-------|-------|-----------------|
| 0 | 0 | P17 input pin |
| | 1 | P17 output pin |
| 1 | * | TMOW output pin |

(2) P16/ \overline{IC}

P16/ \overline{IC} is switched as shown below according to the PMR16 bit in PMR1, the NC on/off bit in prescaler unit control/status register (PCSR), and the PCR16 bit in PCR1.

| PMR16 | PCR16 | NC on/off | Pin Function |
|-------|-------|-----------|---------------------------|
| 0 | 0 | * | P16 input pin |
| | 1 | | P16 output pin |
| 1 | * | 0 | \overline{IC} input pin |
| | | 1 | Noise cancel valid |

(3) P15/ $\overline{IRQ5}$ to P10/ $\overline{IRQ0}$

P15/ $\overline{IRQ15}$ to P10/ $\overline{IRQ0}$ are switched as shown below according to the PMR1n bit in PMR1 and the PCR1n bit in PCR1.

| PMR1n | PCR1n | Pin Function |
|-------|-------|-----------------------------|
| 0 | 0 | P1n input pin |
| | 1 | P1n output pin |
| 1 | * | \overline{IRQn} input pin |

(n = 5 to 0)

Notes: 1. * Don't care.

2. The $\overline{IRQ5}$ to $\overline{IRQ0}$ input pins can select the leading or falling edge as an edge sense (the $\overline{IRQ0}$ pin can select both edges). See section 6.2.4, Edge Select Register (IEGR).

3. $\overline{IRQ1}$ or $\overline{IRQ2}$ can be used as a timer J event input and $\overline{IRQ3}$ can be used as a timer R input capture input. For details, see section 14, "Timer J" or section 16, "Timer R".

11.3.4 Pin States

Table 11.7 shows the port 1 pin states in each operation mode.

Table 11.7 Port 1 Pin States

| Pins | Reset | Active | Sleep | Standby | Watch | Subactive | Subsleep |
|---|--------------------|-----------|---------|--------------------|--------------------|-----------|----------|
| P17/TMOW P16/ \overline{IC} P15/ $\overline{IRQ5}$ to P10/ $\overline{IRQ0}$ | High- impedance | Operation | Holding | High- impedance | High- impedance | Operation | Holding |

Note: If the \overline{IC} input pin and $\overline{IRQ5}$ to $\overline{IRQ0}$ input pins are set, the pin level need be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level.

11.4 Port 2

11.4.1 Overview

Port 2 is an 8-bit I/O port. Table 11.8 shows the port 2 configuration.

Port 2 consists of pins that are used both as standard I/O ports (P27 to P20) and SCI clock I/O (SCK1, SCK2), receive data input (SI1, SI2), send data output (SO1, SO2), I²C bus interface clock I/O (SCL), or data I/O (SCL). It is switched by port mode register 2 (PRM2), serial mode register (SMR), serial control register 2 (SCR), I²C bus control register (ICCR), and port control register (PCR2).

Port 2 can select the MOS pull-up function.

Table 11.8 Port 2 Configuration

| Port | Function | Alternative Function |
|--------|-------------------------|--|
| Port 2 | P27 (standard I/O port) | SCK2 (SCI2 clock I/O) |
| | P26 (standard I/O port) | SO2 (SCI2 transmit data output) |
| | P25 (standard I/O port) | SI2 (SCI2 receive data input) |
| | P24 (standard I/O port) | SCL (I ² C bus interface clock I/O) |
| | P23 (standard I/O port) | SDA (I ² C bus interface data I/O) |
| | P22 (standard I/O port) | SCK1 (SCI1 clock I/O) |
| | P21 (standard I/O port) | SO1 (SCI1 transmit data output) |
| | P20 (standard I/O port) | SI1 (SCI1 receive data input) |

11.4.2 Register Configuration

Table 11.9 shows the port 2 register configuration.

Table 11.9 Port 2 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-------------------------------|---------|-----|------|---------------|----------|
| Port mode register 2 | PMR2 | R/W | Byte | H'1E | H'FFCF |
| Port control register 2 | PCR2 | W | Byte | H'00 | H'FFD2 |
| Port data register 2 | PDR2 | R/W | Byte | H'00 | H'FFC2 |
| MOS pull-up select register 2 | PUR2 | R/W | Byte | H'00 | H'FFE2 |

Note: * Lower 16 bits of the address.

(1) Port Mode Register 2 (PMR2)

| | | | | | | | | |
|-----------------|-------|-------|-------|---|---|---|---|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR27 | PMR26 | PMR25 | — | — | — | — | PMR20 |
| Initial value : | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| R/W : | R/W | R/W | R/W | — | — | — | — | R/W |

Port mode register 2 (PMR0) controls switching of each pin function of port 2. The switching is specified in a unit of bit.

The switching of the P22/SCK1, P21/SO1, and P20/SI1 pin functions is controlled by SMR and SCR. See section 23, SCI1.

PMR2 is an 8-bit read/write enable register. When reset, PMR2 is initialized to H'1E.

If the SCK1, SCK2, SI1, and SI1 input pins are set, the pin level need be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level

Bit 7: P27/SCK2 Pin Switching (PMR27)

PMR27 sets whether the P27/SCK2 pin is used as a P27 I/O pin or an SKC2 pin for the SCI2 clock I/O.

Bit 7

| PMR27 | Description |
|-------|---|
| 0 | The P27/SCK2 pin functions as a P27 I/O pin (Initial value) |
| 1 | The P27/SCK2 pin functions as an SCK2 I/O pin |

Bit 6: P26/SO2 Pin Switching (PMR26)

PMR26 sets whether the P26/SO2 pin as a P26 I/O pin or an SO2 pin for the SCI2 send data output.

Bit 6

| PMR26 | Description |
|-------|--|
| 0 | The P26/SO2 pin functions as a P26 I/O pin (Initial value) |
| 1 | The P26/SO2 pin functions as an SO2 input pin |

Bit 5: P25/SI2 Pin Switching (PMR25)

PMR26 sets whether the P25/SI2 pin as a P25 I/O pin or an SI2 pin for the SCI2 receive data input.

Bit 5

| PMR25 | Description |
|-------|--|
| 0 | The P25/SI2 pin functions as a P25 I/O pin (Initial value) |
| 1 | The P25/SI2 pin functions as an SI2 input pin |

Bits 4 to 1: Reserved Bits

When the bits are read, 1 is always read. The write operation is invalid.

Bit 0: P26/SO2 Pin PMOS Control (PMR20)

PMR20 controls the PMOS ON and OFF of the P26/SO2 pin output buffer.

Bit 0

| PMR20 | Description |
|-------|--|
| 0 | The P26/SO2 pin functions as CMOS output (Initial value) |
| 1 | The P26/SO2 pin functions as NMOS open drain output |

(2) Port Control Register 2 (PCR2)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR27 | PCR26 | PCR25 | PCR24 | PCR23 | PCR22 | PCR21 | PCR20 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

Port control register 2 (PCR2) controls the I/Os of pins P27 to P20 of port 2 in a unit of bit.

When PCR2 is set to 1, the corresponding P27 to P20 pins become output pins, and when it is set to 0, they become input pins. When the relevant pin is set to a general I/O by PMR1, settings of PCR2 and PDR2 are valid.

PCR2 is an 8-bit write-only register. When PCR2 is read, 1 is read. When reset, PCR2 is initialized to H'00.

Bit n

| PCR2n | Description |
|-------|---|
| 0 | The P2n pin functions as an input pin (Initial value) |
| 1 | The P2n pin functions as an output pin |

(n = 7 to 0)

(3) Port Data Register 2 (PDR2)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR27 | PDR26 | PDR25 | PDR24 | PDR23 | PDR22 | PDR21 | PDR20 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port data register 2 (PDR2) stores the data for the pins P27 to P20 of port 2. When PCR2 is 1 (output), the PDR2 values are directly read if port 2 is read. Accordingly, the pin states are not affected. When PCR2 is 0 (input), the pin states are read if port 2 is read. PDR2 is an 8-bit read/write enable register. When reset, PDR2 is initialized to H'00.

(4) MOS Pull-Up Select Register 2 (PUR2)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PUR27 | PUR26 | PUR25 | PUR24 | PUR23 | PUR22 | PUR21 | PUR20 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MOS pull-up selector register 2 (PUR2) controls the ON and OFF of the MOS pull-up transistor of port 2. Only the pin whose corresponding bit of PCR1 was set to 0 (input) becomes valid. If the corresponding bit of PCR2 is set to 1 (output), the corresponding bit of PUR2 becomes invalid and the MOS pull-up transistor is turned off. PUR2 is an 8-bit read/write enable register. When reset, PUR2 is initialized to H'00.

Bit n

| PMR2n | Description |
|-------|---|
| 0 | The P2n pin has no MOS pull-up transistor (Initial value) |
| 1 | The P2n pin has a MOS pull-up transistor |

(n = 7 to 0)

11.4.3 Pin Functions

This section describes the port 2 pin functions and their selection methods.

(1) P27/SCK2

P27/SCK2 is switched as shown below according to the PMR27 bit in PMR2, the PCR27 bit in PCR2, and the SCK2 to SCK0 bits in serial control register 2 (SCR2).

| PMR27 | PCR27 | CKS2 to CKS0 | Pin Function |
|-------|-------|----------------|-----------------|
| 0 | 0 | * | P27 input pin |
| | 1 | | P27 output pin |
| 1 | * | Other than 111 | SCK2 output pin |
| | | 111 | SCK2 input pin |

Note: * Don't care.

(2) P26/SO2

P26/SO2 is switched as shown below according to the PMR26 bit in PMR2 and the PCR26 bit in PCR2.

| PMR26 | PCR26 | Pin Function |
|-------|-------|----------------|
| 0 | 0 | P26 input pin |
| | 1 | P26 output pin |
| 1 | * | SO2 output pin |

Note: * Don't care.

(3) P25/SI2

P25/SI2 is switched as shown below according to the PMR25 bit in PMR2 and the PCR25 bit in PCR2.

| PMR25 | PCR25 | Pin Function |
|-------|-------|----------------|
| 0 | 0 | P25 input pin |
| | 1 | P25 output pin |
| 1 | * | SI2 input pin |

Note: * Don't care.

(4) P24/SCL

P24/SCL2 is switched as shown below according to the ICE bit in the I²C bus control register and the PCR24 bit in PCR2.

| ICE | PCR24 | Pin Function |
|-----|-------|----------------|
| 0 | 0 | P24 input pin |
| | 1 | P24 output pin |
| 1 | * | SCL I/O pin |

Note: * Don't care.

(5) P23/SDA

P23/SDA is switched as shown below according to the ICE bit in the I²C bus control register and the PCR23 bit in PCR2.

| ICE | PCR23 | Pin Function |
|-----|-------|----------------|
| 0 | 0 | P23 input pin |
| | 1 | P23 output pin |
| 1 | * | SDA I/O pin |

Note: * Don't care.

(6) P22/SCK1

P22/SCK1 is switched as shown below according to the PCR22 bit in PCR2, the C/ \bar{A} bit in SMR, and the CKE1 and CKE0 bits in SCR.

| CKE1 | C/ \bar{A} | CKE0 | PCR22 | Pin Function |
|------|--------------|------|-------|-----------------|
| 0 | 0 | 0 | 0 | P22 input pin |
| | | | 1 | P22 output pin |
| | | 1 | * | SCK1 output pin |
| | 1 | * | | |
| 1 | * | | | SCK1 input pin |

Note: * Don't care.

(7) P21/SO1

P21/SO1 is switched as shown below according to the PCR21 bit in PCR2 and the TE bit in SCR.

| TE | PCR21 | Pin Function |
|----|-------|----------------|
| 0 | 0 | P21 input pin |
| | 1 | P21 output pin |
| 1 | * | SO1 output pin |

Note: * Don't care.

(8) P20/SI1

P20/SI1 is switched as shown below according to the PCR20 bit in PCR2 and the RE bit in SCR.

| RE | PCR20 | Pin Function |
|----|-------|----------------|
| 0 | 0 | P20 input pin |
| | 1 | P20 output pin |
| 1 | * | SI1 input pin |

Note: * Don't care.

11.4.4 Pin States

Table 11.10 shows the port 2 pin states in each operation mode.

Table 11.10 Port 2 Pin States

| Pins | Reset | Active | Sleep | Standby | Watch | Subactive | Subsleep |
|----------|----------------|-----------|---------|----------------|----------------|-----------|----------|
| P27/SCK2 | High-impedance | Operation | Holding | High-impedance | High-impedance | Operation | Holding |
| P26/SO2 | | | | | | | |
| P25/SI2 | | | | | | | |
| P24/SCL | | | | | | | |
| P23/SDA | | | | | | | |
| P22/SCK1 | | | | | | | |
| P21/SO1 | | | | | | | |
| P20/SI1 | | | | | | | |

Note: If the SCK1, SCK2, SI1, and SI2 input pins are set, the pin level needs be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level.

11.5 Port 3

11.5.1 Overview

Port 3 is an 8-bit I/O port. Table 11.11 shows the port 3 configuration.

Port 3 consists of pins that are used both as standard I/O ports (P37 to P30) and timer J timer output (TMO), buzzer output (BUZZ), 8-bit PWN outputs (PWN3 to PWN0), SCI2 strobe output (STRB), or chip select input ($\overline{\text{CS}}$). It is switched by port mode register 3 (PMR3) and port control register 3 (PCR3).

Port 3 can select the MOS pull-up function.

Table 11.11 Port 3 Configuration

| Port | Function | Alternative Function |
|--------|-------------------------|---|
| Port 3 | P37 (standard I/O port) | TMO (timer J timer output) |
| | P36 (standard I/O port) | BUZZ (timer J buzzer output) |
| | P35 (standard I/O port) | PWM3 (8-bit PWM output) |
| | P34 (standard I/O port) | PWM2 (8-bit PWM output) |
| | P33 (standard I/O port) | PWM1 (8-bit PWM output) |
| | P32 (standard I/O port) | PWM0 (8-bit PWM output) |
| | P31 (standard I/O port) | STRB (SCI2 strobe output) |
| | P30 (standard I/O port) | $\overline{\text{CS}}$ (SCI2 chip select input) |

11.5.2 Register Configuration

Table 11.12 shows the port 3 register configuration.

Table 11.12 Port 3 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-------------------------------|---------|-----|------|---------------|----------|
| Port mode register 3 | PMR3 | R/W | Byte | H'00 | H'FFD0 |
| Port control register 3 | PCR3 | W | Byte | H'00 | H'FFD3 |
| Port data register 3 | PDR3 | R/W | Byte | H'00 | H'FFC3 |
| MOS pull-up select register 3 | PUR3 | R/W | Byte | H'00 | H'FFE3 |

Note: * Lower 16 bits of the address.

(1) Port Mode Register 3 (PMR3)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR37 | PMR36 | PMR35 | PMR34 | PMR33 | PMR32 | PMR31 | PMR30 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port mode register 3 (PMR3) controls switching of each pin function of port 3. The switching is specified in a unit of bit.

PMR3 is an 8-bit read/write enable register. When reset, PMR3 is initialized to H'00.

If the \overline{CS} input pin is set, the pin level need be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level.

Bit 7: P37/TMO Pin Switching (PMR37)

PMR37 sets whether the P37/TMO pin is used as a P37 I/O pin or a TMO pin for the timer J output timer.

Bit 7

| PMR37 | Description |
|-------|--|
| 0 | The P37/TMO pin functions as a P37 I/O pin (Initial value) |
| 1 | The P37/TMO pin functions as a TMO output pin |

Note: If the TMO pin is used for remote control sending, a careless timer output pulse may be output when the remote control mode is set after the output has been switched to the TMO output. Perform the switching and setting in the following order.

- [1] Set the remote control mode.
- [2] Set the TMJ-1 and 2 counter data of the timer J.
- [3] Switch the P37/TMO pin to the TMO output pin.
- [4] Set the ST bit to 1.

Bit 6: P36/BUZZ Pin Switching (PMR36)

PMR36 sets whether the P36/BUZZ pin as a P36 I/O pin or an BUZZ pin for the timer J buzzer output. For the selection of the BUZZ output, see the 14.2.2, Timer J Control Register (TMJC).

Bit 6

| PMR36 | Description |
|-------|---|
| 0 | The P36/BUZZ pin functions as a P36 I/O pin (Initial value) |
| 1 | The P36/BUZZ pin functions as a BUZZ output pin |

Bits 5 to 2: P35/PWM3 to P32/PWM0 Pin Switching (PMR35 to PMR32)

PMR35 to PMR32 set whether the P3n/PWMm pin is used as a P3n I/O pin or a PWMm pin for the 8-bit PWM output.

Bit n

| PMR3n | Description |
|-------|---|
| 0 | The P3n/PWMm pin functions as a P3n I/O pin (Initial value) |
| 1 | The P3n/PWMm pin functions as a PWMm output pin |

(n = 5 to 2, m = 3 to 0)

Bit 1: P31/STRB Pin Switching (PMR31)

PMR31 sets whether the P31/STRB pin is used as a P31 I/O pin or an STRB pin for the SCI2 strobe output.

Bit 1

| PMR31 | Description |
|-------|---|
| 0 | The P31/STRB pin functions as a P31 I/O pin (Initial value) |
| 1 | The P31/STRB pin functions as an STRB output pin |

Bit 0: P30/ $\overline{\text{CS}}$ Pin Switching (PMR30)

PMR30 sets whether the P30/ $\overline{\text{CS}}$ pin is used as a P30 I/O pin or a $\overline{\text{CS}}$ pin for the SCI2 chip select input.

Bit 0

| PMR30 | Description |
|-------|---|
| 0 | The P30/ $\overline{\text{CS}}$ pin functions as a P30 I/O pin (Initial value) |
| 1 | The P30/ $\overline{\text{CS}}$ pin functions as a $\overline{\text{CS}}$ input pin |

(2) Port Control Register 3 (PCR3)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR37 | PCR36 | PCR35 | PCR34 | PCR33 | PCR32 | PCR31 | PCR30 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

Port control register 3 (PCR3) controls the I/Os of pins P37 to P30 of port 3 in a unit of bit. When PCR3 is set to 1, the corresponding P37 to P30 pins become output pins, and when it is set to 0, they become input pins. When the relevant pin is set to a general I/O by PMR3, settings of PCR3 and PDR3 become valid.

PCR3 is an 8-bit write-only register. When PCR3 is read, 1 is read. When reset, PCR3 is initialized to H'00.

Bit n

| PCR3n | Description |
|-------|---|
| 0 | The P3n pin functions as an input pin (Initial value) |
| 1 | The P3n pin functions as an output pin |

(n = 7 to 0)

(3) Port Data Register 3 (PDR3)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR37 | PDR36 | PDR35 | PDR34 | PDR33 | PDR32 | PDR31 | PDR30 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port data register 3 (PDR3) stores the data for the pins P37 to P30 of port 3. When PCR3 is 1 (output), the PDR3 values are directly read if port 3 is read. Accordingly, the pin states are not affected. When PCR3 is 0 (input), the pin states are read if port 3 is read.

PDR3 is an 8-bit read/write enable register. When reset, PDR3 is initialized to H'00.

(4) MOS Pull-Up Select Register 3 (PUR3)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PUR37 | PUR36 | PUR35 | PUR34 | PUR33 | PUR32 | PUR31 | PUR30 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

MOS pull-up selector register 3 (PUR3) controls the ON and OFF of the MOS pull-up transistor of port 3. Only the pin whose corresponding bit of PCR3 was set to 0 (input) becomes valid. If the corresponding bit of PCR3 is set to 1 (output), the corresponding bit of PUR3 becomes invalid and the MOS pull-up transistor is turned off.

PUR3 is an 8-bit read/write enable register. When reset, PUR3 is initialized to H'00.

Bit n

| PCR3n | Description |
|-------|---|
| 0 | The P3n pin has no MOS pull-up transistor (Initial value) |
| 1 | The P3n pin has a MOS pull-up transistor |

(n = 7 to 0)

11.5.3 Pin Functions

This section describes the port 3 pin functions and their selection methods.

(1) P37/TMO

P37/TMO is switched as shown below according to the PMR37 bit in PMR3 and the PCR37 bit in PCR3.

| PMR37 | PCR37 | Pin Function |
|-------|-------|----------------|
| 0 | 0 | P37 input pin |
| | 1 | P37 output pin |
| 1 | * | TMO output pin |

Note: * Don't care.

(2) P36/BUZZ

P36/BUZZ is switched as shown below according to the PMR36 bit in PMR3 and the PCR36 bit in PCR3.

| PMR36 | PCR36 | Pin Function |
|-------|-------|-----------------|
| 0 | 0 | P36 input pin |
| | 1 | P36 output pin |
| 1 | * | BUZZ output pin |

Note: * Don't care.

(3) P35/PWM3 to P32/PWM0

P35/PWM3 to P32/PWM0 are switched as shown below according to the PMR3n bit in PMR3 and the PCR3n bit in PCR3.

| PMR3n | PCR3n | Pin Function |
|-------|-------|-----------------|
| 0 | 0 | P3n input pin |
| | 1 | P3n output pin |
| 1 | * | PWMm output pin |

(n = 5 to 2, m = 3 to 0)

Note: * Don't care.

(4) P31/STRB

P31/STRB is switched as shown below according to the PMR31 bit in PMR3 and the PCR31 bit in PCR3.

| PMR31 | PCR31 | Pin Function |
|-------|-------|-----------------|
| 0 | 0 | P31 input pin |
| | 1 | P31 output pin |
| 1 | * | STRB output pin |

Note: * Don't care.

(5) P30/ \overline{CS}

P30/ \overline{CS} is switched as shown below according to the PMR30 bit in PMR3 and the PCR30 bit in PCR3.

| PMR30 | PCR30 | Pin Function |
|-------|-------|---------------------------|
| 0 | 0 | P30 input pin |
| | 1 | P30 output pin |
| 1 | * | \overline{CS} input pin |

Note: * Don't care.

11.5.4 Pin States

Table 11.13 shows the port 3 pin states in each operation mode.

Table 11.13 Port 3 Pin States

| Pins | Reset | Active | Sleep | Standby | Watch | Subactive | Subsleep |
|---|----------------|-----------|---------|----------------|----------------|-----------|----------|
| P37/TMO P36/BUZZ P35/PWM3 to P32/PWM0 P31/STRB P30/ \overline{CS} | High-impedance | Operation | Holding | High-impedance | High-impedance | Operation | Holding |

Note: If the \overline{CS} input pin is set, the pin level need be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level.

11.6 Port 4

11.6.1 Overview

Port 4 is an 8-bit I/O port. Table 11.14 shows the port 4 configuration.

Port 4 consists of pins that are used both as standard I/O ports (P47 to P40) and output compare output (FTOA, FTOB), input capture input (FTIA, FTIB, FTIC, FTID) or 14-bit PWM output (PWM14). It is switched by port mode register 4 (PRM4), timer output compare control register (TOCR), and port control register 4 (PCR4).

Table 11.14 Port 4 Configuration

| Port | Function | Alternative Function |
|--------|-------------------------|---------------------------------------|
| Port 4 | P47 (standard I/O port) | None |
| | P46 (standard I/O port) | FTOB (timer X1 output compare output) |
| | P45 (standard I/O port) | FTOA (timer X1 output compare output) |
| | P44 (standard I/O port) | FTID (timer X1 input capture input) |
| | P43 (standard I/O port) | FTIC (timer X1 input capture input) |
| | P42 (standard I/O port) | FTIB (timer X1 input capture input) |
| | P41 (standard I/O port) | FTIA (timer X1 input capture input) |
| | P40 (standard I/O port) | PWM14 (14-bit PWM output) |

11.6.2 Register Configuration

Table 11.15 shows the port 4 register configuration.

Table 11.15 Port 4 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-------------------------|---------|-----|------|---------------|----------|
| Port mode register 4 | PMR4 | R/W | Byte | H'FE | H'FFDB |
| Port control register 4 | PCR4 | W | Byte | H'00 | H'FFD4 |
| Port data register 4 | PDR4 | R/W | Byte | H'00 | H'FFC4 |

Note: * Lower 16 bits of the address.

(1) Port Mode Register 4 (PMR4)

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | PMR40 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| R/W : | — | — | — | — | — | — | — | R/W |

Port mode register 4 (PMR4) controls switching of the P40/PWM14 pin function. The switchings of the P46/FTOB and P45/FTOA functions are controlled by TOCR. See section 17, Timer X1. The FTIA, FTIB, FTIC, and FTID inputs always function. PMR4 is an 8-bit read/write enable register. When reset, PMR4 is initialized to H'FE.

Because the FTIA, FTIB, FTIC, and FTID inputs always function, the alternative pin need always be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level (excluding reset, standby, and watch modes).

Because the FTIA, FTIB, FTIC, and FTID inputs always function, each input uses the input edge to the alternative general I/O pins P44, P43, P42, and P41 as input signals.

Bits 7 to 1: Reserved Bits

When the bits are read, 1 is always read. The write operation is invalid.

Bit 0: P40/PWM14 Pin Switching (PMR40)

PMR40 sets whether the P40/PWM pin is used as a P40 I/O pin or a PWM14 pin for the 14-bit PWM square wave output.

Bit 0

| PMR40 | Description |
|-------|--|
| 0 | The P40/PWM14 pin functions as a P40 I/O pin (Initial value) |
| 1 | The P40/PWM14 pin functions as a PWM14 output pin |

(2) Port Control Register 4 (PCR4)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR47 | PCR46 | PCR45 | PCR44 | PCR43 | PCR42 | PCR41 | PCR40 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

Port control register 4 (PCR4) controls the I/Os of pins P47 to P40 of port 4 in a unit of bit. When PCR4 is set to 1, the corresponding P47 to P40 pins become output pins, and when it is set to 0, they become input pins. When the relevant pin is set to a general I/O by PMR4, settings of PCR4 and PDR4 become valid.

PCR4 is an 8-bit write-only register. When PCR4 is read, 1 is read. When reset, PCR4 is initialized to H'00.

Bit n

| PCR4n | Description |
|-------|---|
| 0 | The P4n pin functions as an input pin (Initial value) |
| 1 | The P4n pin functions as an output pin |

(n = 7 to 0)

(3) Port Data Register 4 (PDR4)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR47 | PDR46 | PDR45 | PDR44 | PDR43 | PDR42 | PDR41 | PDR40 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port data register 4 (PDR4) stores the data for the pins P47 to P40 of port 4. When PCR4 is 1 (output), the PDR4 values are directly read if port 3 is read. Accordingly, the pin states are not affected. When PCR4 is 0 (input), the pin states are read if port 4 is read.

PDR4 is an 8-bit read/write enable register. When reset, PDR4 is initialized to H'00.

11.6.3 Pin Functions

This section describes the port 4 pin functions and their selection methods.

(1) P47/FTCI

P47/FTCI is switched as shown below according to the PCR47 bit in PCR4.

| PCR47 | Pin Function |
|-------|----------------|
| 0 | P47 input pin |
| 1 | P47 output pin |

(2) P46/FTOB

P46/FTOB is switched as shown below according to the PCR46 bit in PCR4 and the OEB bit in TOCR.

| OEB | PCR46 | Pin Function |
|-----|-------|-----------------|
| 0 | 0 | P46 input pin |
| | 1 | P46 output pin |
| 1 | * | FTOB output pin |

Note: * Don't care.

(3) P45/FTOA

P45/FTOA is switched as shown below according to the PCR45 bit in PCR4 and the OEA bit in TOCR.

| OEA | PCR45 | Pin Function |
|-----|-------|-----------------|
| 0 | 0 | P45 input pin |
| | 1 | P45 output pin |
| 1 | * | FTOA output pin |

Note: * Don't care.

(4) P44/FTID

P44/FTID is switched as shown below according to the PCR44 bit in PCR4.

| PCR44 | Pin Function |
|-------|---|
| 0 | P44 input pin FTID input pin |
| 1 | P44 output pin |

(5) P43/FTIC

P43/FTIC is switched as shown below according to the PCR43 bit in PCR4.

| PCR43 | Pin Function | |
|-------|----------------|----------------|
| 0 | P43 input pin | FTIC input pin |
| 1 | P43 output pin | |

(6) P42/FTIB

P42/FTIB is switched as shown below according to the PCR42 bit in PCR4.

| PCR42 | Pin Function | |
|-------|----------------|----------------|
| 0 | P42 input pin | FTIB input pin |
| 1 | P42 output pin | |

(7) P41/FTIA

P41/FTIA is switched as shown below according to the PCR41 bit in PCR4.

| PCR41 | Pin Function | |
|-------|----------------|----------------|
| 0 | P41 input pin | FTIA input pin |
| 1 | P41 output pin | |

(8) P40/PWM14

P40/PWM14 is switched as shown below according to the PMR40 bit in PMR4 and the PCR40 bit in PCR4.

| PMR40 | PCR40 | Pin Function |
|-------|-------|-----------------|
| 0 | 0 | P40 input pin |
| | 1 | P40 output pin |
| 1 | * | PWM14 input pin |

Note: * Don't care.

11.6.4 Pin States

Table 11.16 shows the port 4 pin states in each operation mode.

Table 11.16 Port 4 Pin States

| Pins | Reset | Active | Sleep | Standby | Watch | Subactive | Subsleep |
|--|--------------------|-----------|---------|--------------------|--------------------|-----------|----------|
| P47 P46/FTOB P45/FTOA P44/FTID P43/FTIC P42/FTIB P41/FTIA P40/ PWM14 | High- impedance | Operation | Holding | High- impedance | High- impedance | Operation | Holding |

Note: Because the FTIA, FTIB, FTIC, and FTID inputs always function, the alternative pin need be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level (excluding reset, standby, and watch modes).

11.7 Port 5

11.7.1 Overview

Port 5 is a 4-bit I/O port. Table 11.17 shows the port 5 configuration.

Port 5 consists of pins that are used both as standard I/O ports (P53 to P50) and realtime output port trigger input (TRIG), timer B event input (TMBI), or A/D conversion start external trigger input ($\overline{\text{ADTRG}}$). It is switched by port mode register 5 (PMR5), A/D trigger select register (ADTSR), and port control register 5 (PCR5).

Table 11.17 Port 5 Configuration

| Port | Function | Alternative Function |
|--------|-------------------------|---|
| Port 5 | P53 (standard I/O port) | TRIG (realtime output port trigger input) |
| | P52 (standard I/O port) | TMBI (timer B event input) |
| | P51 (standard I/O port) | None |
| | P50 (standard I/O port) | $\overline{\text{ADTRG}}$ (A/D conversion start external trigger input) |

11.7.2 Register Configuration

Table 11.18 shows the port 5 register configuration.

Table 11.18 Port 5 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-------------------------|---------|-----|------|---------------|----------|
| Port mode register 5 | PMR5 | R/W | Byte | H'F1 | H'FFDC |
| Port control register 5 | PCR5 | W | Byte | H'F0 | H'FFD5 |
| Port data register 5 | PDR5 | R/W | Byte | H'F0 | H'FFC5 |

Note: * Lower 16 bits of the address.

(1) Port Mode Register 5 (PMR5)

| | | | | | | | | |
|-----------------|---|---|---|---|-------|-------|-------|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PMR53 | PMR52 | PMR51 | — |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| R/W : | — | — | — | — | R/W | R/W | R/W | — |

Port mode register 5 (PMR5) controls switching of each pin function of port 5 and specifies the edge sense of the timer B event input (TMBI).

The switching of the P50/ $\overline{\text{ADTRG}}$ pin function is controlled by ADTSR. See section 26, A/D Converter.

PMR5 is an 8-bit read/write enable register. When reset, PMR5 is initialized to H'F1.

If the TRIG, TMBI, and $\overline{\text{ADTRG}}$ pin pins are set, the alternative pin need always be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level.

Bits 7 to 4: Reserved Bits

When the bits are read, 1 is always read. The write operation is invalid.

Bit 3: P53/TRIG Pin Switching (PMR53)

PMR53 sets whether the P53/TRIG pin is used as a P53 I/O pin or a TRIG pin for the realtime output port trigger input.

Bit 3

| PMR53 | Description |
|-------|---|
| 0 | The P53/TRIG pin functions as a P53 I/O pin (Initial value) |
| 1 | The P53/TRIG pin functions as a TRIG input pin |

Bit 2: P52/TMBI Pin Switching (PMR52)

PMR52 sets whether the P52/TMBI pin is used as a P52 I/O pin or a TMBI pin for the timer B event input.

Bit 2

| PMR52 | Description |
|-------|---|
| 0 | The P52/TMBI pin functions as a P52 I/O pin (Initial value) |
| 1 | The P52/TMBI pin functions as a TMBI input pin |

Bit 1: Timer B event input edge select (PMR51)

PMR51 selects the input edge sense of the TMBI pin.

Bit 1

| PMR51 | Description |
|-------|--|
| 0 | The timer B event input detects the falling edge (Initial value) |
| 1 | The timer B event input detects the rising edge |

Bit 0: Reserved Bit

When the bit is read, 1 is always read. The write operation is invalid.

(2) Port Control Register 5 (PCR5)

| | | | | | | | | |
|-----------------|---|---|---|---|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PCR53 | PCR52 | PCR51 | PCR50 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W |

Port control register 5 (PCR5) controls the I/Os of pins P53 to P50 of port 5 in a unit of bit.

When PCR5 is set to 1, the corresponding P53 to P50 pins become output pins, and when it is set to 0, they become input pins. When the relevant pin is set to a general I/O, settings of PCR5 and PDR5 are valid.

PCR5 is an 8-bit write-only register. When PCR5 is read, 1 is read. When reset, PCR5 is initialized to H'F0.

Bits 7 to 4 are reserved bits.

Bit n

| PCR5n | Description |
|-------|---|
| 0 | The P5n pin functions as an input pin (Initial value) |
| 1 | The P5n pin functions as an output pin |

(n = 3 to 0)

(3) Port Data Register 5 (PDR5)

| | | | | | | | | |
|-----------------|---|---|---|---|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PDR53 | PDR52 | PDR51 | PDR50 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | R/W | R/W | R/W | R/W |

Port data register 5 (PDR5) stores the data for the pins P53 to P50 of port 5. When PCR5 is 1 (output), the PDR5 values are directly read if port 5 is read. Accordingly, the pin states are not affected. When PCR5 is 0 (input), the pin states are read if port 5 is read.

PDR5 is an 8-bit read/write enable register. When reset, PDR5 is initialized to H'F0.

Bits 7 to 4 are reserved bits.

11.7.3 Pin Functions

This section describes the port 5 pin functions and their selection methods.

(1) P53/TRIG

P53/TRIG is switched as shown below according to the PMR53 bit in PMR5 and the PCR53 bit in PCR5.

| PMR53 | PCR53 | Pin Function |
|-------|-------|----------------|
| 0 | 0 | P53 input pin |
| | 1 | P53 output pin |
| 1 | * | TRIG input pin |

Note: * Don't care.

(2) P52/TMBI

P52/TMBI is switched as shown below according to the PMR52 bit in PMR5 and the PCR52 bit in PCR5.

| PMR52 | PCR52 | Pin Function |
|-------|-------|----------------|
| 0 | 0 | P52 input pin |
| | 1 | P52 output pin |
| 1 | * | TMBI input pin |

Note: * Don't care.

(3) P51

P51 is switched as shown below according to the PCR51 bit in PCR5.

| PCR51 | Pin Function |
|-------|----------------|
| 0 | P51 input pin |
| 1 | P51 output pin |

(4) P50/ $\overline{\text{ADTRG}}$

P50/ $\overline{\text{ADTRG}}$ is switched as shown below according to the PCR50 bit in PCR5 and the TRGS1 and TRG0 bits in ADTSR.

| TRGS1, TRGS0 | PCR31 | Pin Function |
|---------------|-------|-------------------------------------|
| Other than 11 | 0 | P50 input pin |
| | 1 | P50 output pin |
| 11 | * | $\overline{\text{ADTRG}}$ input pin |

Note: * Don't care.

11.7.4 Pin States

Table 11.19 shows the port 5 pin states in each operation mode.

Table 11.19 Port 3 Pin States

| Pins | Reset | Active | Sleep | Standby | Watch | Subactive | Subsleep |
|--|--------------------|-----------|---------|--------------------|--------------------|-----------|----------|
| P53/TRIG P52/TMBI P51 P50/ $\overline{\text{ADRTG}}$ | High- impedance | Operation | Holding | High- impedance | High- impedance | Operation | Holding |

Note: If the TRIG, TMBI, and $\overline{\text{ADRTG}}$ input pins are set, the alternative pin need always be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level.

11.8 Port 6

11.8.1 Overview

Port 6 is an 8-bit I/O port. Table 11.20 shows the port 6 configuration.

Port 6 consists of pins that are used both as standard I/O ports (P67 to P60) and realtime output ports (RP7 to RP0). It is switched by port mode register 6 (PMR6) and port control register 6 (PCR6).

The realtime output function can instantaneously switch the output data by an external or internal trigger input.

Table 11.20 Port 6 Configuration

| Port | Function | Alternative Function |
|--------|-------------------------|--------------------------------|
| Port 6 | P67 (standard I/O port) | RP7 (realtime output port pin) |
| | P66 (standard I/O port) | RP6 (realtime output port pin) |
| | P65 (standard I/O port) | RP5 (realtime output port pin) |
| | P64 (standard I/O port) | RP4 (realtime output port pin) |
| | P63 (standard I/O port) | RP3 (realtime output port pin) |
| | P62 (standard I/O port) | RP2 (realtime output port pin) |
| | P61 (standard I/O port) | RP1 (realtime output port pin) |
| | P60 (standard I/O port) | RP0 (realtime output port pin) |

11.8.2 Register Configuration

Table 11.21 shows the port 6 register configuration.

Table 11.21 Port 6 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|--|---------|-----|------|---------------|----------|
| Port mode register 6 | PMR6 | R/W | Byte | H'00 | H'FFDD |
| Port control register 6 | PCR6 | W | Byte | H'00 | H'FFD6 |
| Port data register 6 | PDR6 | R/W | Byte | H'00 | H'FFC6 |
| Realtime output trigger select register | RTPSR | R/W | Byte | H'00 | H'FFE5 |
| Realtime output trigger edge select register | RTPEGR | R/W | Byte | H'FC | H'FFE4 |
| Port control register slave 6 | PCRS6 | — | Byte | H'00 | — |
| Port data register slave 6 | PDRS6 | — | Byte | H'00 | — |

Note: * Lower 16 bits of the address.

(1) Port Mode Register 6 (PMR6)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR67 | PMR66 | PMR65 | PMR64 | PMR63 | PMR62 | PMR61 | PMR60 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port mode register 6 (PMR6) controls switching of each pin function of port 6. The switching is specified in units of bits.

PMR6 is an 8-bit read/write enable register. When reset, PMR6 is initialized to H'00.

Bits 7 to 0: P67/RP7 to P60/RP0 Pin Switching (PMR67 to PMR60)

PMR67 to PMR60 set whether the P6n/RPn pin is used as a P6n I/O pin or an RPn pin for the realtime output port.

| Bit n | |
|-------|--|
| PMR6n | Description |
| 0 | The P6n/RPn pin functions as a P6n I/O pin (Initial value) |
| 1 | The P6n/RPn pin functions as an RPn output pin |

(n = 7 to 0)

(2) Port Control Register 6 (PCR6)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR67 | PCR66 | PCR65 | PCR64 | PCR63 | PCR62 | PCR61 | PCR60 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

Port control register 6 (PCR6) selects the general I/O of port 6 and controls the realtime output in a unit of bit together with PMR6.

When PMR6 = 0, the corresponding P67 to P60 pins become general output pins if PCR6 is set to 1, and they become general input pins if it is set to 0.

When PMR6 = 1, PCR6 controls the corresponding RP7 to RP0 realtime output pins. For details, see section 11.8.4, Operation.

PCR6 is an 8-bit write-only register. When PCR6 is read, 1 is read. When reset, PCR6 is initialized to H'00.

| PMR6 | PCR6 | |
|-------|-------|---|
| Bit n | Bit n | |
| PMR6n | PCR6n | Description |
| 0 | 0 | The P6n/RPn pin functions as a P6n general I/O input pin (Initial value) |
| | 1 | The P6n/RPn pin functions as a P6n general output pin |
| 1 | * | The P6n/RPn pin functions as an RPn realtime output pin |

Note: * Don't care. (n = 7 to 0)

(3) Port Data Register 6 (PDR6)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR67 | PDR66 | PDR65 | PDR64 | PDR63 | PDR62 | PDR61 | PDR60 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port data register 6 (PDR6) stores the data for the pins P67 to P60 of port 6.

For PMR6 = 0, when PCR6 is 1 (output), the PDR6 values are directly read if port 6 is read.

Accordingly, the pin states are not affected. When PCR6 is 0 (input), the pin states are read if port 6 is read.

For PMR6 = 1, port 6 becomes a realtime output pin. For details, see section 11.8.4, Operation.

PDR6 is an 8-bit read/write enable register. When reset, PDR6 is initialized to H'00.

(4) Realtime Output Trigger Select Register (RTPSR)

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | RTPSR7 | RTPSR6 | RTPSR5 | RTPSR4 | RTPSR3 | RTPSR2 | RTPSR1 | RTPSR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The realtime output trigger select register (RTPSR) sets whether the external trigger (TRIG pin input) or the internal trigger (HSW) is used as an trigger input for the realtime output in a unit of bit. For the internal trigger HSW, see section 28.4, HSW Timing Generation Circuit. RTPSR is an 8-bit read/write enable register. When reset, RTPSR is initialized to H'00.

Bit n

| RTPSRn | Description |
|--------|--|
| 0 | Selects the external trigger (TRIG pin input) as a trigger input (Initial value) |
| 1 | Selects the internal trigger (HSW) a trigger input |

(n = 7 to 0)

(5) Real Time Output Trigger Edge Select Register (RTPEGR)

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---------|---------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | RTPEGR1 | RTPEGR0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W : | — | — | — | — | — | — | R/W | R/W |

The realtime output trigger edge select register (RTPEGR) specifies the edge sense of the external or internal trigger input for the realtime output.

RTPEGR is an 8-bit read/write enable register. When reset, RTPEGR is initialized to H'FC.

Bits 7 to 2: Reserved Bits

When the bits are read, 1 is always read. The write operation is invalid.

Bits 1 and 0: Realtime Output Trigger Edge Select (RTPEGR1, RTPEGR0)

RTPEGR1 and RTPEGR0 select the edge sense of the external or internal trigger input for the realtime output.

| Bit 1 | Bit 0 | Description |
|---------|---------|---|
| RTPEGR1 | RTPEGR0 | |
| 0 | 0 | Inhibits a trigger input (Initial value) |
| | 1 | Selects the rising edge of a trigger input |
| 1 | 0 | Selects the falling edge of a trigger input |
| | 1 | Selects both the leading and falling edges of a trigger input |

11.8.3 Pin Functions

This section describes the port 6 pin functions and their selection methods.

(1) P67/RP7 to P60/RP0

P67/RP7 to P60/RP0 are switched as shown below according to the PMR6n bit in PMR6 and the PCR6n bit in PCR6.

| PMR6n | PCR6n | Pin Function | Output Value | Value When PDR6n was read |
|-------|-------|----------------|-----------------|---------------------------|
| 0 | 0 | P6n input pin | — | P6n pin |
| | 1 | P6n output pin | PDR6n | PDR6n |
| 1 | 0 | RPn output pin | High-impedance* | |
| | 1 | | PDRS6n* | |

Note: * When PMR6n = 1 (realtime output pin), indicates the state after the PCR6n setup value has been transferred to PCRS6n by a trigger input. (n = 7 to 0)

11.8.4 Operation

Port 6 can be used as a realtime output port or general I/O output port by PMR6. Port 6 functions as a realtime output port when PMR6 = 1 and as a general I/O port when PMR6 = 0. The operation per port 6 function is shown below. (See figure 11.2.)

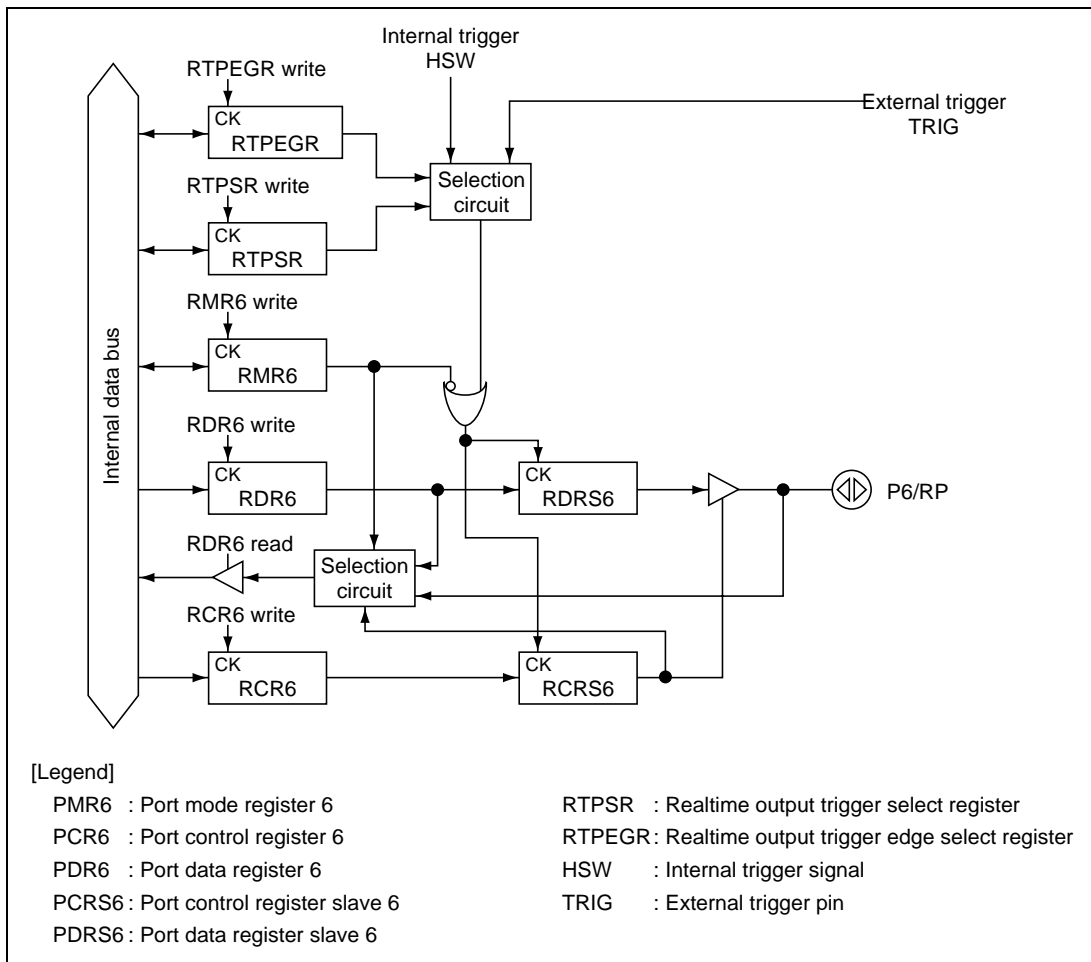


Figure 11.2 Port 6 Function Block Diagram

(1) Operation of the Realtime Output Port (PMR6 = 1)

When PMR6 is 1, it operates as a realtime output port. When a trigger is input, PMR6 transfers the PDR6 data to PDRS6 and the PCR6 data to PCRS6, respectively. In this case, when PCRS6 is 1, the PDRS6 data of the corresponding bit is output to the RP pin. When PCRS6 is 0, the RP pin of the corresponding bit is output to the high-impedance state. In other words, the pin output state (High or Low) or high-impedance state can instantaneously be switched by a trigger input.

Adversely, when PDR6 is read, the PDR6 values are read regardless of the PCR6 and PCRS6 values.

(2) Operation of the general I/O port (PMR6 = 0)

When PMR6 is 0, it operates as a general I/O port. When data is written to PDR6, the same data is also written to PDRS6. Accordingly, because both PDR6 and PDRS6 and both PCR6 and PCRS6 can be handled as one register, respectively, they can be used in the same way as a normal general I/O port. In other words, if PCR6 is 1, the PDR6 data of the corresponding bit is output to the P6 pin. If PCR6 is 0, the P6 pin of the corresponding bit becomes an input.

Adversely, assuming that PDR6 is read, the PDR6 values are read when PCR6 is 1 and the pin values are read when PCR6 is 0.

11.8.5 Pin States

Table 11.22 shows the port 6 pin states in each operation mode.

Table 11.22 Port 6 Pin States

| Pins | Reset | Active | Sleep | Standby | Watch | Subactive | Subsleep |
|-----------------------|--------------------|-----------|---------|--------------------|--------------------|-----------|----------|
| P67/RP7 to P60/RP0 | High- impedance | Operation | Holding | High- impedance | High- impedance | Operation | Holding |

11.9 Port 7

11.9.1 Overview

Port 7 is an 8-bit I/O port. Table 11.23 shows the port 7 configuration.

Port 7 consists of pins that are used both as standard I/O ports (P77 to P70) and HSW timing generation circuit (programmable pattern generator: PPG) outputs (PPG7 to PPG0). It is switched by port mode register 7 (PMR7) and port control register 7 (PCR7).

For the programmable generator (PPG), see section 28.4, HSW Timing Generation Circuit.

Table 11.23 Port 7 Configuration

| Port | Function | Alternative Function |
|--------|-------------------------|--------------------------|
| Port 7 | P77 (standard I/O port) | PPG7 (HSW timing output) |
| | P76 (standard I/O port) | PPG6 (HSW timing output) |
| | P75 (standard I/O port) | PPG5 (HSW timing output) |
| | P74 (standard I/O port) | PPG4 (HSW timing output) |
| | P73 (standard I/O port) | PPG3 (HSW timing output) |
| | P72 (standard I/O port) | PPG2 (HSW timing output) |
| | P71 (standard I/O port) | PPG1 (HSW timing output) |
| | P70 (standard I/O port) | PPG0 (HSW timing output) |

11.9.2 Register Configuration

Table 11.24 shows the port 7 register configuration.

Table 11.24 Port 7 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-------------------------|---------|-----|------|---------------|----------|
| Port mode register 7 | PMR7 | R/W | Byte | H'00 | H'FFDE |
| Port control register 7 | PCR7 | W | Byte | H'00 | H'FFD7 |
| Port control register 7 | PDR7 | R/W | Byte | H'00 | H'FFC7 |

Note: * Lower 16 bits of the address.

(1) Port Mode Register 7 (PMR7)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR77 | PMR76 | PMR75 | PMR74 | PMR73 | PMR72 | PMR71 | PMR70 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port mode register 7 (PMR7) controls switching of each pin function of port 7. The switching is specified in a unit of bit.

PMR7 is an 8-bit read/write enable register. When reset, PMR7 is initialized to H'00.

Bits 7 to 0: P77/PPG7 to P70/PPG0 Pin Switching (PMR77 to PMR70)

PMR77 to PMR70 set whether the P7n/PPGn pin is used as a P7n I/O pin or a PPGn pin for the HSW timing generation circuit output.

Bit n

| PMR7n | Description |
|-------|---|
| 0 | The P7n/PPGn pin functions as a P7n I/O pin (Initial value) |
| 1 | The P7n/PPGn pin functions as a PPGn output pin |

(n = 7 to 0)

(2) Port Control Register 7 (PCR7)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR77 | PCR76 | PCR75 | PCR74 | PCR73 | PCR72 | PCR71 | PCR70 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

Port control register 7 (PCR7) controls the I/Os of pins P77 to P70 of port 7 in a unit of bit.

When PCR7 is set to 1, the corresponding P77 to P70 pins become output pins, and when it is set to 0, they become input pins. When the corresponding pin is set to the general I/O by PMR7, settings of PCR7 and PDR7 become valid.

PCR7 is an 8-bit write-only register. When PCR7 is read, 1 is read. When reset, PCR7 is initialized to H'00.

Bit n

| PCR7n | Description |
|-------|---|
| 0 | The P7n pin functions as an input pin (Initial value) |
| 1 | The P7n pin functions as an output pin |

(n = 7 to 0)

(3) Port Data Register 7 (PDR7)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR77 | PDR76 | PDR75 | PDR74 | PDR73 | PDR72 | PDR71 | PDR70 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port data register 7 (PDR7) stores the data for the pins P77 to P70 of port 7.

When PCR7 is 1 (output), the PDR7 values are directly read if port 7 is read. Accordingly, the pin states are not affected. When PCR7 is 0 (input), the pin states are read if port 7 is read.

PDR7 is an 8-bit read/write enable register. When reset, PDR7 is initialized to H'00.

11.9.3 Pin Functions

This section describes the port 7 pin functions and their selection methods.

(1) P77/PPG7 to P70/PPG0

P77/PPG7 to P70/PPG0 are switched as shown below according to the PMR7n bit in PMR7 and the PCR7n bit in PCR7.

| PMR7n | PCR7n | Pin Function |
|-------|-------|----------------|
| 0 | 0 | P7n input pin |
| | 1 | P7n output pin |
| 1 | * | PPGn input pin |

Note: * Don't care. (n = 7 to 0)

11.9.4 Pin States

Table 11.25 shows the port 7 pin states in each operation mode.

Table 11.25 Port 7 Pin States

| Pins | Reset | Active | Sleep | Standby | Watch | Subactive | Subsleep |
|----------------------------|----------------|-----------|---------|----------------|----------------|-----------|----------|
| P77/PPG7 to P70/PPG0 | High-impedance | Operation | Holding | High-impedance | High-impedance | Operation | Holding |

11.10 Port 8

11.10.1 Overview

Port 8 is an 8-bit I/O port. Table 11.26 shows the port 8 configuration.

Port 8 is a CMOS high-current I/O port. The sink current is 20 mA max. ($V_{OL} = 1.5$ V) and up to four pins can simultaneously be set on.

Port 8 consists of pins that are used both as high-current I/O ports (P87 to P80) and servo monitor output (SV1, SV2), capstan external synchronous signal input (EXCAP), or external trigger signal input (EXTTRG). It is switched by port mode register 8 (PMR8) and port control register 8 (PCR8).

Table 11.26 Port 8 Configuration

| Port | Function | Alternative Function |
|--------|-----------------------------|---|
| Port 8 | P87 (high-current I/O port) | None |
| | P86 (high-current I/O port) | None |
| | P85 (high-current I/O port) | None |
| | P84 (high-current I/O port) | None |
| | P83 (high-current I/O port) | SV2 (servo monitor output) |
| | P82 (high-current I/O port) | SV1 (servo monitor output) |
| | P81 (high-current I/O port) | EXCAP (capstan external synchronous signal input) |
| | P80 (high-current I/O port) | EXTTRG (external trigger signal input) |

11.10.2 Register Configuration

Table 11.27 shows the port 8 register configuration.

Table 11.27 Port 8 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-------------------------|---------|-----|------|---------------|----------|
| Port mode register 8 | PMR8 | R/W | Byte | H'F0 | H'FFDF |
| Port control register 8 | PCR8 | W | Byte | H'00 | H'FFD8 |
| Port data register 8 | PDR8 | R/W | Byte | H'00 | H'FFC8 |

Note: * The address indicates the low-order 16 bits.

(1) Port Mode Register 8 (PMR8)

| | | | | | | | | |
|-----------------|---|---|---|---|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PMR83 | PMR82 | PMR81 | PMR80 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | R/W | R/W | R/W | R/W |

Port mode register 8 (PMR8) controls switching of each pin function of port 8. The switching is specified in a unit of bit.

PMR8 is an 8-bit read/write enable register. When reset, PMR8 is initialized to H'F0.

If the EXCAP and EXTTRG input pins are set, the pin level need always be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level.

Bits 7 to 4: Reserved Bits

When the bits are read, 1 is always read. The write operation is valid.

Bit 3: P83/SV2 Pin Switching (PMR83)

PMR83 sets whether the P83/SV2 pin is used as a P83 I/O pin or an SV2 pin for the servo monitor output. For the selection of the SV2 output, see section 28, Servo Circuit.

Bit 3

| PMR83 | Description |
|-------|--|
| 0 | The P83/SV2 pin functions as a P83 I/O pin (Initial value) |
| 1 | The P83/SV2 pin functions as an SV2 output pin |

Bit 2: P82/SV1 Pin Switching (PMR82)

PMR82 sets whether the P82/SV1 pin is used as a P82 I/O pin or an SV1 pin for the servo monitor output. For the selection of the SV1 output, see section 27, Servo Circuit.

Bit 2

| PMR82 | Description |
|-------|--|
| 0 | The P82/SV1 pin functions as a P82 I/O pin (Initial value) |
| 1 | The P82/SV1 pin functions as an SV1 output pin |

Bit 1: P81/EXCAP Pin Switching (PMR81)

PMR81 sets whether the P81/EXCAP pin is used as a P81 I/O pin or an EXTTRG pin for the capstan external synchronous signal input.

Bit 1

| PMR81 | Description |
|-------|--|
| 0 | The P81/EXCAP pin functions as a P81 I/O pin (Initial value) |
| 1 | The P81/EXCAP pin functions as an EXCAP input pin |

Bit 0: P80/EXTTRG Pin Switching (PMR80)

PMR80 sets whether the P80/EXTTRG pin is used as a P80 I/O pin or an EXTTRG pin for the external trigger signal input.

Bit 0

| PMR80 | Description |
|-------|---|
| 0 | The P80/EXTTRG pin functions as a P80 I/O pin (Initial value) |
| 1 | The P80/EXTTRG pin functions as an EXTTRG input pin |

(2) Port Control Register 8 (PCR8)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR87 | PCR86 | PCR85 | PCR84 | PCR83 | PCR82 | PCR81 | PCR80 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

Port control register 8 (PCR8) controls the I/Os of pins P87 to P80 of port 8 in a unit of bit. When PCR8 is set to 1, the corresponding P87 to P80 pins become output pins, and when it is set to 0, they become input pins. When the corresponding pin is set to a general I/O, settings of PCR8 and PDR8 become valid.

PCR8 is an 8-bit write-only register. When PCR8 is read, 1 is read. When reset, PCR8 is initialized to H'00.

Bit n

| PCR8n | Description |
|-------|---|
| 0 | The P8n pin functions as an input pin (Initial value) |
| 1 | The P8n pin functions as an output pin |

(n = 7 to 0)

(3) Port Data Register 8 (PDR8)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR87 | PDR86 | PDR85 | PDR84 | PDR83 | PDR82 | PDR81 | PDR80 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port data register 8 (PDR8) stores the data for the pins P87 to P80 of port 8.

When PCR8 is 1 (output), the PDR8 values are directly read if port 8 is read. Accordingly, the pin states are not affected. When PCR8 is 0 (input), the pin states are read if port 8 is read.

PDR8 is an 8-bit read/write enable register. When reset, PDR8 is initialized to H'00.

11.10.3 Pin Functions

This section describes the port 8 pin functions and their selection methods.

(1) P87 to P84

P87 to P84 are switched as shown below according to the PCR8n bit in PCR8.

| PCR8n | Pin Function |
|-------|----------------|
| 0 | P8n input pin |
| 1 | P8n output pin |

(n = 7 to 4)

Note: * Don't care.

(2) P83/SV2

P83/SV2 is switched as shown below according to the PMR83 bit in PMR8 and the PCR83 bit in PCR8.

| PMR83 | PCR83 | Pin Function |
|-------|-------|-----------------|
| 0 | 0 | P83 input pin |
| | 1 | P83n output pin |
| 1 | * | SV2 output pin |

Note: * Don't care.

(3) P82/SV1

P82/SV1 is switched as shown below according to the PMR82 bit in PRM8 and the PCR82 bit in PCR8.

| PMR82 | PCR82 | Pin Function |
|-------|-------|----------------|
| 0 | 0 | P82 input pin |
| | 1 | P82 output pin |
| 1 | * | SV1 output pin |

Note: * Don't care.

(4) P81/EXCAP

P81/EXCAP is switched as shown below according to the PMR81 bit in PRM8 and the PCR81 bit in PCR8.

| PMR81 | PCR81 | Pin Function |
|-------|-------|-----------------|
| 0 | 0 | P81 input pin |
| | 1 | P81 output pin |
| 1 | * | EXCAP input pin |

Note: * Don't care.

(5) P80/EXTTRG

P80/EXTTRG is switched as shown below according to the PMR80 bit in PRM8 and the PCR80 bit in PCR8.

| PMR80 | PCR80 | Pin Function |
|-------|-------|------------------|
| 0 | 0 | P80 input pin |
| | 1 | P80 output pin |
| 1 | * | EXTTRG input pin |

Note: * Don't care.

11.10.4 Pin States

Table 11.28 shows the port 8 pin states in each operation mode.

Table 11.28 Port 8 Pin States

| Pins | Reset | Active | Sleep | Standby | Watch | Subactive | Subsleep |
|---|--------------------|-----------|---------|--------------------|--------------------|-----------|----------|
| P87 to P84 P83/SV2 P83/SV1 P81/ EXCAP P80/ EXTTRG | High- impedance | Operation | Holding | High- impedance | High- impedance | Operation | Holding |

Note: If the EXCAP and EXTTRG input pins are set, the pin level need always be set to the high or low level regardless of the active mode and low power consumption mode. Note that the pin level must not reach an intermediate level.

Section 12 Timer A

12.1 Overview

The Timer A is an 8-bit interval timer. It can be used as a clock timer when connected to a 32.768 kHz crystal oscillator.

12.1.1 Features

Features of the Timer A are as follows:

- Choices of eight different types of internal clocks ($\phi/16384$, $\phi/8192$, $\phi/4096$, $\phi/1024$, $\phi/512$, $\phi/256$, $\phi/64$ and $\phi/16$) are available for your selection.
- Four different overflowing cycles (1s, 0.5s, 0.25s and 0.03125s) are selectable as a clock timer. (When using a 32.768 kHz crystal oscillator.)
- Requests for interrupt will be output when the counter overflows.

12.1.2 Block Diagram

Figure 12.1 shows a block diagram of the Timer A.

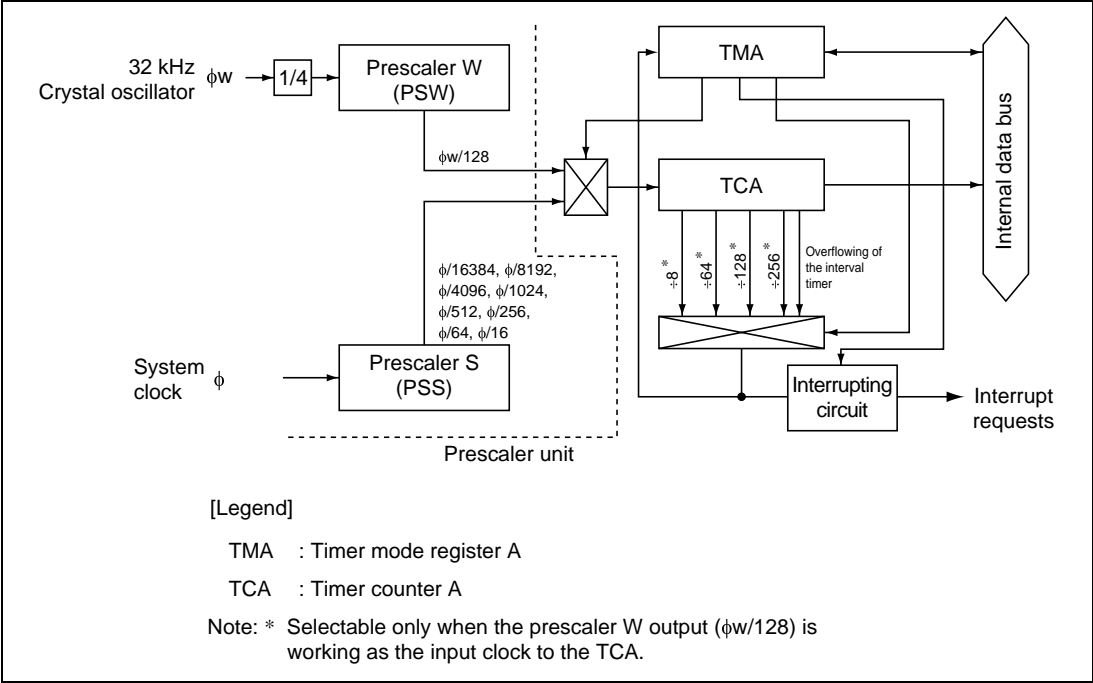


Figure 12.1 Block Diagram of the Timer A

12.1.3 Register Configuration

Table 12.1 shows the register configuration of the Timer A.

Table 12.1 Register configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-----------------------|---------|-----|------|---------------|----------|
| Timer mode register A | TMA | R/W | Byte | H'30 | H'FFBA |
| Timer counter A | TCA | R | Byte | H'00 | H'FFBB |

Note: * Lower 16 bits of the address.

12.2 Descriptions of Respective Registers

12.2.1 Timer Mode Register A (TMA)

| | | | | | | | | |
|-----------------|--------|-------|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMAOV | TMAIE | — | — | TMA3 | TMA2 | TMA1 | TMA0 |
| Initial value : | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/W | — | — | R/W | R/W | R/W | R/W |

Note: * Only 0 can be written to clear the flag.

The timer mode register A (TMA) works to control the interrupts of the Timer A and to select the input clock.

TMA is an 8-bit read/write register. When reset, the TMA will be initialized to H'30.

Bit 7: Timer A Overflow Flag (TMAOV)

This is a status flag indicating the fact that the TCA is overflowing (H'FF → H'00).

Bit 7

| TMAOV | Description |
|-------|--|
| 0 | [Clearing conditions] (Initial value) When 0 is written to the TMAOV flag after reading the TMAOV flag under the status where TMAOV = 1 |
| 1 | [Setting conditions] When the TCA overflows |

Bit 6: Enabling Interrupt of the Timer A (TMAIE)

This bit works to permit/prohibit occurrence of interrupt of the Timer A (TMAI) when the TCA overflows and when the TMAOV of the TMA is set to 1.

Bit 6

| TMAIE | Description |
|-------|---|
| 0 | Prohibits occurrence of interrupt of the Timer A (TMAI) (Initial value) |
| 1 | Permits occurrence of interrupt of the Timer A (TMAI) |

Bits 5 and 4: Reserved

When they are read, 1 will always be readout. Writes are disabled.

Bit 3: Selection of the Clock Source and Prescaler (TMA3)

This bit works to select the PSS or PSW as the clock source for the Timer A.

Bit 3

| TMA3 | Description |
|------|---|
| 0 | Selects the PSS as the clock source for the Timer A (Initial value) |
| 1 | Selects the PSW as the clock source for the Timer A |

Bits 2 to 0: Clock Selection (TMA2 to TMA0)

These bits work to select the clock to input to the TCA. In combination with the TMA3 bit, the choices are as follows:

| Bit 3 | Bit 2 | Bit 1 | Bit 0 | Prescaler division ratio (interval timer) or overflow cycle (time base) | Operation mode |
|-------|-------|-------|-------|--|-------------------------|
| TMA3 | TMA2 | TMA1 | TMA0 | | |
| 0 | 0 | 0 | 0 | PSS, $\phi/16384$ (Initial value) | Interval timer mode |
| | | | 1 | PSS, $\phi/8192$ | |
| | | 1 | 0 | PSS, $\phi/4096$ | |
| | | | 1 | PSS, $\phi/1024$ | |
| | 1 | 0 | 0 | PSS, $\phi/512$ | |
| | | | 1 | PSS, $\phi/256$ | |
| | | 1 | 0 | PSS, $\phi/64$ | |
| | | | 1 | PSS, $\phi/16$ | |
| 1 | 0 | 0 | 0 | 1s | Clock time base mode |
| | | | 1 | 0.5s | |
| | | 1 | 0 | 0.25s | |
| | | | 1 | 0.03125s | |
| | 1 | 0 | 0 | Works to clear the PSW and TCA to H'00 | |
| | | | 1 | | |
| | | 1 | 0 | | |
| | | | 1 | | |

Note: $\phi = f_{osc}$

12.2.2 **Timer Counter A (TCA)**

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TCA7 | TCA6 | TCA5 | TCA4 | TCA3 | TCA2 | TCA1 | TCA0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R |

The timer counter A (TCA) is an 8-bit up-counter which counts up on inputs from the internal clock. The inputting clock can be selected by TMA3 to TMA0 bits of the TMA

When the TCA overflows, the TMAOV bit of the TMA is set to 1.

The TCA can be cleared by setting the TMA3 and TMA2 bits of the TMA to 11.

The TCA is always readable. When reset, the TCA will be initialized into H'00.

12.2.3 **Module Stop Control Register (MSTPCR)**

| | | | | | | | | | | | | | | | | |
|-----------------|---------|--------|--------|--------|--------|--------|-------|-------|---------|-------|-------|-------|-------|-------|-------|-------|
| | MSTPCRH | | | | | | | | MSTPCRL | | | | | | | |
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | MSTP15 | MSTP14 | MSTP13 | MSTP12 | MSTP11 | MSTP10 | MSTP9 | MSTP8 | MSTP7 | MSTP6 | MSTP5 | MSTP4 | MSTP3 | MSTP2 | MSTP1 | MSTP0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 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 |

The MSTPCR are 8-bit read/write twin registers which work to control the module stop mode.

When the MSTP15 bit is set to 1, the Timer A stops its operation at the ending point of the bus cycle to shift to the module stop mode. For more information, see section 4.5, Module Stop Mode. When reset, the MSTPCR will be initialized into H'FFFF.

Bit 7: Module Stop (MSTP15)

This bit works to designate the module stop mode for the Timer A.

MSTPCRH

Bit 7

| MSTP15 | Description |
|--------|--|
| 0 | Cancels the module stop mode of the Timer A |
| 1 | Sets the module stop mode of the Timer A (Initial value) |

12.3 Operation

The Timer A is an 8-bit timer for use as an interval timer and as a clock time base connecting to a 32.768 kHz crystal oscillator.

12.3.1 Operation as the Interval Timer

When the TMA3 bit of the TMA is cleared to 0, the Timer A works as an 8-bit interval timer. When reset the TCA is cleared to H'00 and as the TMA3 bit is cleared to 0, the Timer A continues counting up as the interval counter without interrupts right after resetting.

As the operation clock for the Timer A, selection can be made from eight different types of internal clocks being output from the PSS by the TMA2 to TMA0 bits of the TMA.

When the clock signal is input after the reading of the TCA reaches H'FF, the Timer A overflows and the TMAOV bit of the TMA will be set to 1. At this time, when the TMAIE bit of the TMA is 1, interrupt occurs.

When overflowing occurs, the reading of the TCA returns to H'00 before resuming counting up. Consequently, it works as the interval timer to produce overflow outputs periodically at every 256 input clocks.

12.3.2 Operation of the Timer for Clocks

When the TMA3 bit of the TMA is set to 1, the Timer A works as a time base for the clock. As the overflow cycles for the Timer A, selection can be made from four different types by counting the clock being output from the PSW by the TMA1 bit and TMA0 bit of the TMA.

12.3.3 Initializing the Counts

When the TMA3 and TMA2 bits are set to 11, the PSW and TCA will be cleared to H'00 to come to a stop.

At this state, writing 10 to the TMA3 bit and TMA2 bit makes the Timer A to start counting from H'00 under the time base mode for clocks.

After clearing the PSW and TCA using the TMA3 and TMA2 bits, writing 00 or 01 to the TMA3 bit and TMA2 bit work to make the Timer A to start counting from H'00 under the interval timer mode. However, since the PSS is not cleared, the period to the first count is not constant.

Section 13 Timer B

13.1 Overview

The Timer B is an 8-bit up-counter. The Timer B is equipped with two different types of functions namely, the interval function and the auto reloading function.

13.1.1 Features

- Selection from choices of seven different types of internal clocks ($\phi/16384$, $\phi/4096$, $\phi/1024$, $\phi/512$, $\phi/128$, $\phi/32$ and $\phi/8$) or selection of external clock are possible.
- When the counter overflows, a interrupt request will be issued.

13.1.2 Block Diagram

Figure 13.1 shows a block diagram of the Timer B.

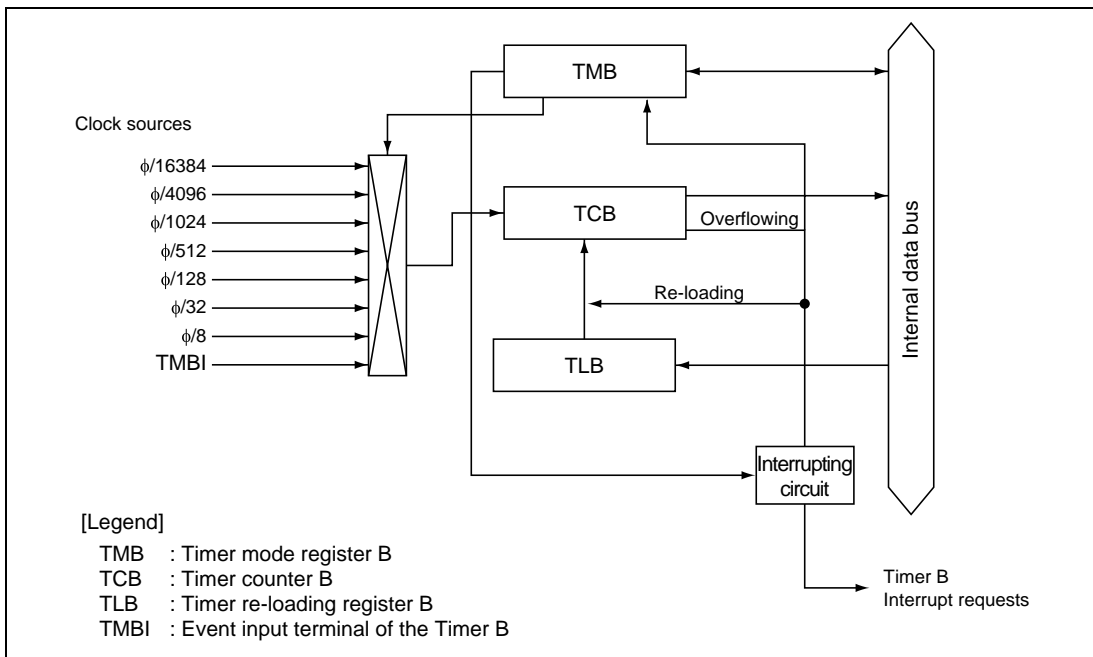


Figure 13.1 Block diagram of the Timer B

13.1.3 Pin Configuration

Table 13.1 shows the pin configuration of the Timer B.

Table 13.1 Pin Configuration

| Name | Abbrev. | I/O | Function |
|-----------------------------|---------|-------|---------------------------------------|
| Event inputs to the Timer B | TMBI | Input | Event input pin for inputs to the TCB |

13.1.4 Register Configuration

Table 13.2 shows the register configuration of the Timer B.

The TCB and TLB are being allocated to the same address. Reading or writing determines the accessing register.

Table 13.2 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-----------------------|---------|-----|------|---------------|----------|
| Timer mode register B | TMB | R/W | Byte | H'18 | H'D110 |
| Timer counter B | TCB | R | Byte | H'00 | H'D111 |
| Timer load register B | TLB | W | Byte | H'00 | H'D111 |
| Port mode register 5 | PMR5 | R/W | Byte | H'F1 | H'FFDC |

Note: * Lower 16 bits of the address.

13.2 Descriptions of Respective Registers

13.2.1 Timer Mode Register B (TMB)

| | | | | | | | | |
|-----------------|-------|--------|-------|---|---|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMB17 | TMBIF | TMBIE | — | — | TMB12 | TMB11 | TMB10 |
| Initial value : | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| R/W : | R/W | R/(W)* | R/W | — | — | R/W | R/W | R/W |

Note: Only 0 can be written to clear the flag.

The TMB is an 8-bit read/write register which works to control the interrupts, to select the auto reloading function and to select the input clock.

When reset, the TMB is initialized to H'18.

Bit 7: Selecting the Auto Reloading Function (TMB17)

This bit works to select the auto reloading function of the Timer B.

Bit 7

| TMB17 | Description |
|-------|---|
| 0 | Selects the interval function (Initial value) |
| 1 | Selects the auto reloading function |

Bit 6: Interrupt Requesting Flag for the Timer B (TMBIF)

This is an interrupt requesting flag for the Timer B. It indicates the fact that the TCB is overflowing.

Bit 6

| TMBIF | Description |
|-------|--|
| 0 | [Clearing conditions] When 0 is written after reading 1 (Initial value) |
| 1 | [Setting conditions] When the TCB overflows |

Bit 5: Enabling Interrupt of the Timer B (TMBIE)

This bit works to permit/prohibit occurrence of interrupt of the Timer B when the TCB overflows and when the TMBIF is set to 1.

Bit 5

| TMBIE | Description |
|-------|--|
| 0 | Prohibits occurrence of interrupt of the Timer B (Initial value) |
| 1 | Permits occurrence of interrupt of the Timer B |

Bits 4 to 3: Reserved

When they are read, 1 will always be readout. Writes are disabled.

Bits 2 to 0: Clock Selection (TMB12 to TMB10)

These bits work to select the clock to input to the TCB. Selection of the rising edge or the falling edge is workable with the external event inputs.

| Bit 2 | Bit 1 | Bit 0 | |
|-------|-------|-------|--|
| TMB12 | TMB11 | TMB10 | Descriptions |
| 0 | 0 | 0 | Internal clock: Counts at $\phi/16384$ (Initial value) |
| 0 | 0 | 1 | Internal clock: Counts at $\phi/4096$ |
| 0 | 1 | 0 | Internal clock: Counts at $\phi/1024$ |
| 0 | 1 | 1 | Internal clock: Counts at $\phi/512$ |
| 1 | 0 | 0 | Internal clock: Counts at $\phi/128$ |
| 1 | 0 | 1 | Internal clock: Counts at $\phi/32$ |
| 1 | 1 | 0 | Internal clock: Counts at $\phi/8$ |
| 1 | 1 | 1 | Counts at the rising edge and the falling edge of external event inputs (TMBI) * |

Note: * The edge selection for the external event inputs is made by setting the PMR51 of the port mode register 5 (PMR5). See section 13.2.4, Port Mode Register 5 (PMR5).

13.2.2 Timer Counter B (TCB)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TCB17 | TCB16 | TCB15 | TCB14 | TCB13 | TCB12 | TCB11 | TCB10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R |

The TCB is an 8-bit readable register which works to count up by the internal clock inputs and external event inputs. The input clock can be selected by the TMB12 to TMB10 of the TMB. When the TCB overflows (H'FF → H'00 or H'FF → TLB setting), a interrupt request of the Timer B will be issued.

When reset, the TCB is initialized to H'00.

13.2.3 Timer Load Register B (TLB)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TLB17 | TLB16 | TLB15 | TLB14 | TLB13 | TLB12 | TLB11 | TLB10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

The TLB is an 8-bit write only register which works to set the reloading value of the TCB. When the reloading value is set to the TLB, the value will be simultaneously loaded to the TCB and the TCB starts counting up from the set value. Also, during an auto reloading operation, when the TCB overflows, the value of the TLB will be loaded to the TCB. Consequently, the overflowing cycle can be set within the range of 1 to 256 input clocks.

When reset, the TLB is initialized to H'00.

13.2.4 Port Mode Register 5 (PMR5)

| | | | | | | | | |
|-----------------|---|---|---|---|-------|-------|-------|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PMR53 | PMR52 | PMR51 | — |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| R/W : | — | — | — | — | R/W | R/W | R/W | — |

The port mode register 5 (PMR5) works to changeover the pin functions of the port 5 and to designate the edge sense of the event inputs of the Timer B (TMBI).
The PMR5 is an 8-bit read/write register. When reset, the PMR5 will be initialized to H'F1.
See section 11.7, Port 5 for other information than bit 1.

Bit 1: Selecting the Edges of the Event Inputs to the Timer B (PMR51)

This bit works to select the input edge sense of the TMBI pins.

Bit 1

| PMR51 | Description |
|-------|---|
| 0 | Detects the falling edge of the event inputs to the Timer B (Initial value) |
| 1 | Detects the rising edge of the event inputs to the Timer B |

13.2.5 Module Stop Control Register (MSTPCR)

| | | | | | | | | | | | | | | | | |
|-----------------|---------------------|--------|--------|--------|--------|--------|-------|-------|---------------------|-------|-------|-------|-------|-------|-------|-------|
| | MSTPCR ^H | | | | | | | | MSTPCR ^L | | | | | | | |
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | MSTP15 | MSTP14 | MSTP13 | MSTP12 | MSTP11 | MSTP10 | MSTP9 | MSTP8 | MSTP7 | MSTP6 | MSTP5 | MSTP4 | MSTP3 | MSTP2 | MSTP1 | MSTP0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 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 |

The MSTPCR are 8-bit read/write twin registers which work to control the module stop mode.
When the MSTP14 bit is set to 1, the Timer B stops its operation at the ending point of the bus cycle to shift to the module stop mode. For more information, see section 4.5, Module stop mode.
When reset, the MSTPCR is initialized to H'FFFF.

Bit 6: Module Stop (MSTP14)

This bit works to designate the module stop mode for the Timer B.

MSTPCRH

Bit 6

| MSTP14 | Description |
|--------|--|
| 0 | Cancels the module stop mode of the Timer B |
| 1 | Sets the module stop mode of the Timer B (Initial value) |

13.3 Operation

13.3.1 Operation as the Interval Timer

When the TMB17 bit of the TMB is set to 0, the Timer B works as an 8-bit interval timer.

When reset, since the TCB is cleared to H'00 and as the TMB17 bit is cleared to 0, the Timer B continues counting up as the interval timer without interrupts right after resetting.

As the clock source for the Timer B, selection can be made from seven different types of internal clocks being output from the prescaler unit by the TMB12 to TMB10 bits of the TMB or an external clock through the TMBI input pin can be chosen instead.

When the clock signal is input after the reading of the TCB reaches H'FF, the Timer B overflows and the TMBIF bit of the TMB will be set to 1. At this time, when the TMBIE bit of the TMB is 1, interrupt occurs.

When overflowing occurs, the reading of the TCB returns to H'00 before resuming counting up.

When a value is set to the TLB while the interval timer is in operation, the value which has been set to the TLB will be loaded to the TCB simultaneously.

13.3.2 Operation as the Auto Reload Timer

When the TMB17 of the TMB is set to 1, the Timer B works as an 8-bit auto reload timer.

When a reload value is set in the TLB, the value is loaded onto the TCB at the same time, and the TCB starts counting up from the value.

When the clock signal is input after the reading of the TCB reaches H'FF, the Timer B overflows and the TLB value is loaded onto the TCB, then the TCB continues counting up from the loaded value. Accordingly, overflow interval can be set within the range of 1 to 256 clocks depending on the TLB value.

Clock source and interrupts in the auto reload operation are the same as those in the interval operation. When the TLB value is re-set while the auto reload timer is in operation, the value which has been set to the TLB will be loaded onto the TCB simultaneously.

13.3.3 Event Counter

The Timer B works as an event counter using the TMBI pin as the event input pin. When the TMB12 to TMB10 are set to 111, the external event will be selected as the clock source and the TCB counts up at the leading edge or the trailing edge of the TMBI pin inputs.

Section 14 Timer J

14.1 Overview

The Timer J consists of twin 8-bit counters. It carries seven different operation modes such as reloading and event counting.

14.1.1 Features

The Timer J consists of twin 8-bit reloading timers and it is usable under the various functions as follows:

- (a) Twin 8-bit reloading timers (Among the two, one is capable to make timer outputs)
- (b) Twin 8-bit event counters (Capable to make reloading)
- (c) 8-bit event counter (Capable to make reloading) + 8-bit reload timer
- (d) 16-bit event counter (Capable to make 16-bit reloading)
- (e) 16-bit reload timer (Capable to make 16-bit reloading)
- (f) Remote controlled transmissions
- (g) "Take up/Supply reel pulse" dividing ($8 \text{ bit} \times 2 \text{ units}$)

14.1.2 Block Diagram

Figure 14.1 is a block diagram of the Timer J. The Timer J consists of two reload timers namely, TMJ-1 and TMJ-2.

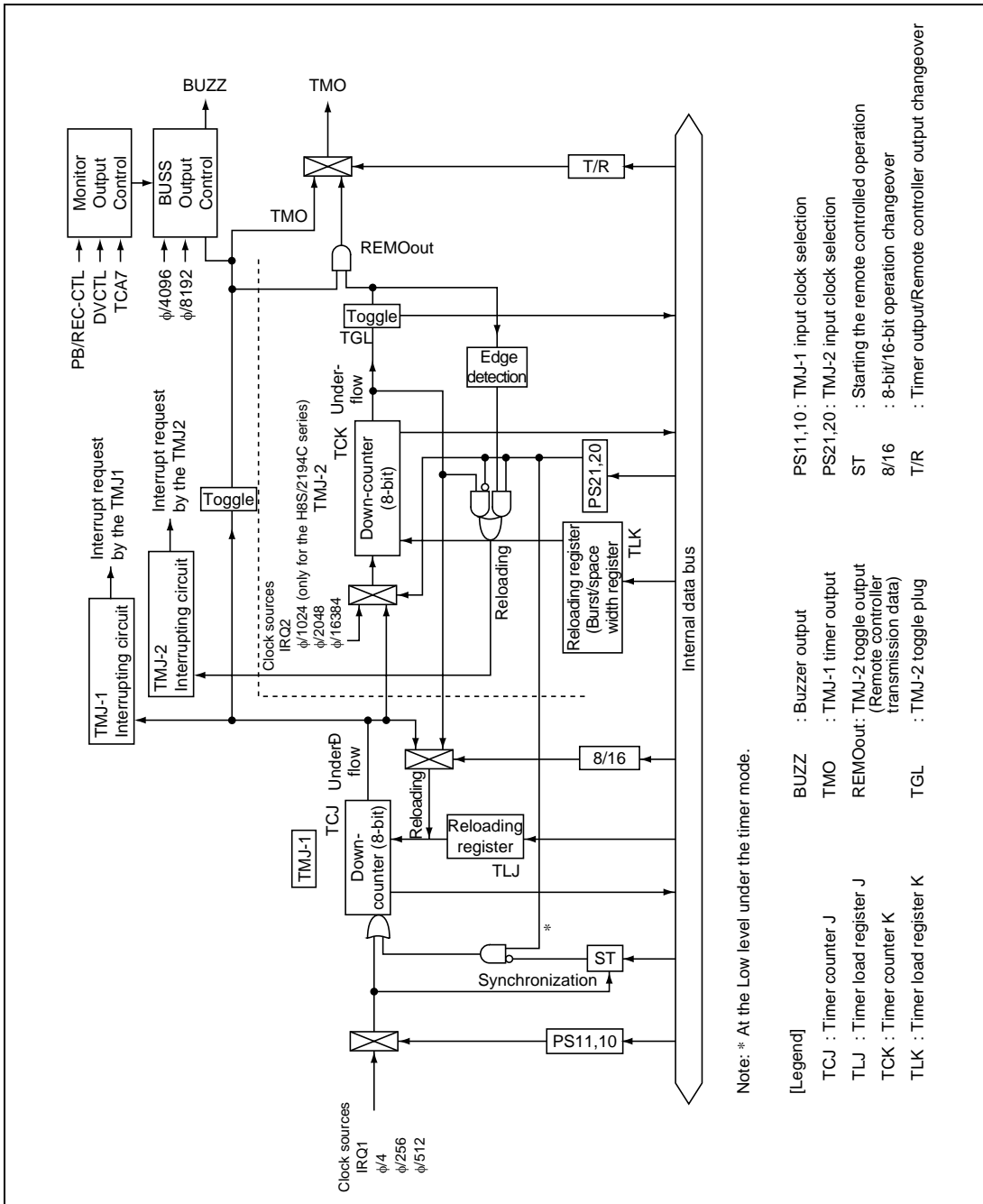


Figure 14.1 Block Diagram of the Timer J

14.1.3 Pin Configuration

Table 14.1 shows the pin configuration of the Timer J.

Table 14.1 Pin Configuration

| Name | Abbrev. | I/O | Function |
|-----------------|--------------------------|-------|---------------------------|
| Event input pin | $\overline{\text{IRQ1}}$ | Input | Event inputs to the TMJ-1 |
| Event input pin | $\overline{\text{IRQ2}}$ | Input | Event inputs to the TMJ-2 |

14.1.4 Register Configuration

Table 14.2 shows the register configuration of the Timer J.

The TCJ and TLJ or the TCK and TLK are being allocated to the same address respectively.

Reading or writing determines the accessing register.

Table 14.2 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address ^{*2} |
|--------------------------|---------|---------------------|------|---------------|-----------------------|
| Timer mode register J | TMJ | R/W | Byte | H'00 | H'D13A |
| Timer J control register | TMJC | R/W | Byte | H'09 | H'D13B |
| Timer J status register | TMJS | R/(W) ^{*1} | Byte | H'3F | H'D13C |
| Timer counter J | TCJ | R | Byte | H'FF | H'D139 |
| Timer counter K | TCK | R | Byte | H'FF | H'D138 |
| Timer load register J | TLJ | W | Byte | H'FF | H'D139 |
| Timer load register K | TLK | W | Byte | H'FF | H'D138 |

Notes: 1. Only 0 can be written to clear the flag.

2. Lower 16 bits of the address.

14.2 Descriptions of Respective Registers

14.2.1 Timer Mode Register J (TMJ)

| | | | | | | | | |
|-----------------|------|------|-----|------|------|------|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PS11 | PS10 | ST | 8/16 | PS21 | PS20 | TGL | T/R |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R | R/W |

The timer mode register J (TMJ) works to select the inputting clock for the TMJ-1 and TMJ-2 and to set the operation mode.

The TMJ is an 8-bit register and Bit-1 is for read only and all the remaining bits are applicable to read/write.

When reset, the TMJ is initialized to H'00.

Under all other modes than the remote controlling mode, writing into the TMJ works to initialize the counters (TCJ and TCK) to H'FF.

Bits 7 and 6: Selecting the Inputting Clock to the TMJ-1 (PS11 and PS10)

These bits work to select the clock to input to the TMJ-1. Selection of the rising edge or the falling edge is workable for counting by use of an external clock.

| Bit 7 | Bit 6 | Description |
|-------|-------|--|
| PS11 | PS10 | |
| 0 | 0 | Counting by the PSS, $\phi/512$ (Initial value) |
| | 1 | Counting by the PSS, $\phi/256$ |
| 1 | 0 | Counting by the PSS, $\phi/4$ |
| | 1 | Counting at the rising edge or the falling edge of the external clock inputs ($\overline{\text{IRQ1}}$)* |

Note: * The edge selection for the external clock inputs is made by setting the edge select register (IEGR). See section 6.2.4, Edge Select Register (IEGR) for more information. When using an external clock under the remote controlling mode, set the opposite edge with the IRQ1 and the IRQ2 when using an external clock under the remote controlling mode. (When IRQ1 falling, select IRQ2 rising and when IRQ1 rising, select IRQ2 falling)

Bit 5: Starting the Remote Controlled Operation (ST)

This bit works to start the remote controlled operations.

When this bit is set to 1, clock signal is supplied to the TMJ-1 to start signal transmissions.

When this bit is cleared to 0, clock supply stops to discontinue the operation. The ST bit will be valid under the remote controlling mode, namely, when the Bit 0 (T/R bit) is 1 and the Bit 4 (8/16 bit) is 0.

Under other modes than the remote controlling mode, it will be fixed to 0. When a shift to the low power consumption mode is made during remote controlled operation, the ST bit will be cleared to 0. When resuming operation after returning to the active mode, write 1.

Bit 5

| ST | Description |
|----|---|
| 0 | Works to stop clock signal supply to the TMJ-1 under the remote controlling mode (Initial value) |
| 1 | Works to supply clock signal to the TMJ-1 under the remote controlling mode |

Bit 4: Switching Over Between 8-bit/16-bit Operations (8/16)

This bit works to choose if using the Timer J as two units of 8-bit timer/counter or if using it as a single unit of 16-bit timer/counter. Even under 16-bit operations, TMJ1I interrupt requests from the TMJ-1 will be valid.

Bit 4

| 8/16 | Description |
|------|--|
| 0 | Makes the TMJ-1 and TMJ-2 operate separately (Initial value) |
| 1 | Makes the TMJ-1 and TMJ-2 operate altogether as 16-bit timer/counter |

Bits 3 and 2: Selecting the Inputting Clock to the TMJ-2 (PS21 and PS20)

This bit works to select the clock to input to the TMJ-2. Selection of the leading edge or the trailing edge is workable for counting by use of an external clock.

| TMJC:Bit0 | Bit 3 | Bit 2 | Description |
|--------------------|-----------------|-----------------|---|
| PS22 ^{*3} | PS21 | PS20 | |
| 1 | 0 | 0 | Counting by the PSS, $\phi/16384$ (Initial value) |
| | | 1 | Counting by the PSS, $\phi/2048$ |
| | 1 | 0 | Counting at underflowing of the TMJ-1 |
| | | 1 | Counting at the leading edge or the trailing edge of the external clock inputs (IRQ2) ^{*1} |
| 0 | * ^{*2} | * ^{*2} | Counting by the PSS, $\phi/1024$ (available only the H8S/2194C series) |

- Notes: 1. The edge selection for the external clock inputs is made by setting the edge select register (IEGR). See section 6.2.4, Edge Select Register (IEGR) for more information.
2. Don't care.
3. Available only in the H8S/2194C series.

Bit 1: TMJ-2 Toggle Flag (TGL)

This flag indicates the toggled status of the underflowing with the TMJ-2. Reading only is workable.

It will be cleared to 0 under the low power consumption mode.

Bit 1

| TGL | Description |
|-----|---|
| 0 | The toggle output of the TMJ-2 is 0 (Initial value) |
| 1 | The toggle output of the TMJ-2 is 0 |

Bit 0: Switching Over Between Timer Output/Remote Controlling Output (T/R)

This bit works to select if using the timer outputs from the TMJ-1 as the output signal through the TMO pin or if using the toggle outputs (remote controlled transmission data) from the TMJ-2 as the output signal through the TMO pin.

Bit 0

| T/R | Description |
|-----|---|
| 0 | Timer outputs from the TMJ-1 (Initial value) |
| 1 | Toggle outputs from the TMJ-2 (remote controlled transmission data) |

Selecting the Operation Mode

The operation mode of the Timer J is determined by the Bit 4 (8/16) and Bit 0 (T/R) of the TMJ.

TMJ

| Bit 4 | Bit 0 | Description |
|-------|-------|---------------------------------|
| 8/16 | T/R | |
| 0 | 0 | 8-bit timer × 2 (Initial value) |
| | 1 | Remote controlling mode |
| 1 | * | 16-bit timer |

Note: * Don't care.

When writing is made into the TMJ under the timer mode, the counters (TCJ and TCK) will be initialized (H'FF). Consequently, writing into the reloading registers (TLJ and TLK) should be conducted after finishing settings with the TMJ.

Under the remote controlling mode, although the TLJ and the TLK will not be initialized even when writing is made into the TMJ, follow the sequence listed below when starting a remote controlling operation.

- (1) Make setting to the remote controlling mode with the TMJ.
- (2) Write the data into the TLJ and TLK.
- (3) Start the remote controlled operation by use of the TMJ. (ST bit = 1)

Even under 16-bit operations, TMJ1I interrupt requests from the TMJ-1 will be valid.

14.2.2 Timer J Control Register (TMJC)

| | | | | | | | | |
|-----------------|-------|-------|------|------|---|--------|--------|---------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | BUZZ1 | BUZZ0 | MON1 | MON0 | — | TMJ2IE | TMJ1IE | (PS22)* |
| Initial value : | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| R/W : | R/W | R/W | R/W | R/W | — | R/W | R/W | (R/W)* |

Note: * Bit 0 is readable/writable only in the H8S/2194C series.

The timer J control register works to select the buzzer output frequency and to control permission/prohibition of interrupts.

The TMJC is an 8-bit read/write register.

When reset, the TMJC is initialized to H'09.

Bits 7 and 6: Selecting the Buzzer Output (BUZZ1 or BUZZ0)

This bit works to select if using the buzzer outputs as the output signal through the BUZZ pin or if using the monitor signals as the output signal through the BUZZ pin.

When setting is made to the monitor signals, choose the monitor signal using the MON1 bit and MON0 bit.

| Bit 7 | Bit 6 | Frequency when $\phi = 10\text{MHz}$ | |
|-------|-------|---|----------|
| BUZZ1 | BUZZ0 | Description | |
| 0 | 0 | $\phi/4096$ (Initial value) | 2.44 kHz |
| | 1 | $\phi/8192$ | 1.22 kHz |
| 1 | 0 | Works to output monitor signals | |
| | 1 | Works to output BUZZ signals from the Timer J | |

Bits 5 and 4: Selecting the Monitor Signals (MON1 or MON0)

These bits work to select the type of signals being output through the BUZZ pin for monitoring purpose. These settings are valid only when the BUZZ1 and BUZZ0 bits are being set to 1 and 0.

When PB-CTL or REC-CTL is chosen, signal duties will be output as they are.

In case of DVCTL signals, signals from the CTL dividing circuit will be toggled before being output. Signal waveforms divided by the CTL dividing circuit into "n-divisions" will further be divided into halves. (Namely, "2n" divisions, 50% duty waveform).

In case of TCA7, Bit 7 of the counter of the Timer A will be output. (50% duty)

When the prescaler is being used with the Timer A, 1Hz outputs are available.

| Bit 5 | Bit 4 | |
|-------|-------|-------------------------------|
| MON1 | MON0 | Description |
| 0 | 0 | PB or REC-CTL (Initial value) |
| | 1 | DVCTL |
| 1 | * | Outputs TCA7 |

Note: * Don't care.

Bits 3: Reserved

When this is read, 1 will always be readout. Writes are disabled.

Bit 2: Enabling Interrupt of the TMJ2I (TMJ2IE)

This bit works to permit/prohibit occurrence of TMJ2I interrupt of the TMJS in 1-set of the TMJ2I.

| Bit 2 | |
|--------|---|
| TMJ2IE | Description |
| 0 | Prohibits occurrence of TMJ2I interrupt (Initial value) |
| 1 | Permits occurrence of TMJ2I interrupt |

Bit 1: Enabling Interrupt of the TMJ1I (TMJ1IE)

This bit works to permit/prohibit occurrence of TMJ1I interrupt of the TMJS in 1-set of the TMJ1I.

| Bit 1 | |
|--------|---|
| TMJ1IE | Description |
| 0 | Prohibits occurrence of TMJ1I interrupt (Initial value) |
| 1 | Permits occurrence of TMJ1I interrupt |

Bit 0: Reserved (for H8S/2194 series)

When this is read, 1 will always be readout. Writes are disabled.

Bit 0: Selecting the Input clock for TMJ-2 (PS22) (for H8S/2194C series)

This bit, together with bits 3 and 2 (PS21, PS20) in TMJ, selects the input clock for TMJ-2. For details, see section 14.2.1, Timer Mode Register J (TMJ).

14.2.3 Timer J Status Register (TMJS)

| | | | | | | | | |
|-----------------|--------|--------|---|---|---|---|---|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMJ2I | TMJ1I | — | — | — | — | — | — |
| Initial value : | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R/(W)* | R/(W)* | — | — | — | — | — | — |

Note: * Only 0 can be written to clear the flag.

The timer J status register (TMJS) works to indicate issuance of the interrupt request of the Timer J. The TMJS is an 8-bit read/write register. When reset, the TMJS is initialized to H'3F.

Bit 7: TMJ2I Interrupt Requesting Flag (TMJ2I)

This is the TMJ2I interrupt requesting flag. This flag is set when the TMJ-2 underflows.

Bit 7

| TMJ2I | Description |
|-------|--|
| 0 | [Clearing conditions] When 0 is written after reading 1 (Initial value) |
| 1 | [Setting conditions] When the TMJ-2 underflows |

Bit 6: TMJ1I Interrupt Requesting Flag (TMJ1I)

This is the TMJ1I interrupt requesting flag. This flag is set out when the TMJ-1 underflows. TMJ1I interrupt requests will also be made under a 16-bit operation.

Bit 6

| TMJ1I | Description |
|-------|--|
| 0 | [Clearing conditions] When 0 is written after reading 1 (Initial value) |
| 1 | [Setting conditions] When the TMJ-1 underflows |

Bits 5 to 0: Reserved

When they are read, 1 will always be readout. Writes are disabled.

14.2.4 Timer Counter J (TCJ)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TDR17 | TDR16 | TDR15 | TDR14 | TDR13 | TDR12 | TDR11 | TDR10 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R | R | R | R | R | R | R | R |

The time counter J (TCJ) is an 8-bit readable down-counter which works to count down by the internal clock inputs or external clock inputs. The inputting clock can be selected by the PS11 and PS10 bits of the TMJ. TCJ values can be readout always. Nonetheless, when the 8/16 bit of the TMJ is being set to 1 (means when setting is made to 16-bit operation), reading is possible under the word command only.

At this time, the TCK of the TMJ-2 can be read by the upper 8 bits and the TCJ can be read by the lower 8 bits.

When the TCJ underflows (H'00 → Reloading value), regardless of the operation mode setting of the 8/16 bit, the TMJ1I bit of the TMJS will be set to 1. The TCJ and TLJ are being allocated to the same address.

When reset, the TCJ is initialized to H'FF.

14.2.5 Timer Counter K (TCK)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TDR27 | TDR26 | TDR25 | TDR24 | TDR23 | TDR22 | TDR21 | TDR20 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R | R | R | R | R | R | R | R |

The time counter K (TCK) is an 8-bit readable down-counter which works to count down by the internal clock inputs or external clock inputs. The inputting clock can be selected by the PS21 and PS20 bits of the TMJ. TCK values can be readout always. Nonetheless, when the 8/16 bit of the TMJ is being set to 1 (means when setting is made to 16-bit operation), reading is possible under the word command only.

At this time, the TCK can be read by the upper 8 bits and the TCJ of the TMJ-1 can be read by the lower 8 bits.

When the TCK underflows (H'00 → Reloading value), the TMJ2I bit of the TMJS will be set to 1.

The TCK and TLK are being allocated to the same address.

When reset, the TCK is initialized to H'FF.

14.2.6 Timer Load Register J (TLJ)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TLR17 | TLR16 | TLR15 | TLR14 | TLR13 | TLR12 | TLR11 | TLR10 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W |

The timer load register J (TLJ) is an 8-bit write only register which works to set the reloading value of the TCJ.

When the reloading value is set to the TLJ, the value will be simultaneously loaded to the TCJ and the TCJ starts counting down from the set value. Also, during an auto reloading operation, when the TCJ underflows, the value of the TLJ will be loaded to the TCJ. Consequently, the underflowing cycle can be set within the range of 1 to 256 input clocks. Nonetheless, when the 8/16 bit of the TMJ is being set to 1 (means when setting is made to 16-bit operation), writing is possible under the word command only.

At this time, the upper 8 bits can be written into the TLK of the TMJ-2 and the lower 8 bits can be written into the TLJ.

The TLJ and TCJ are being allocated to the same address.

When reset, the TLJ is initialized to H'FF.

14.2.7 Timer Load Register K (TLK)

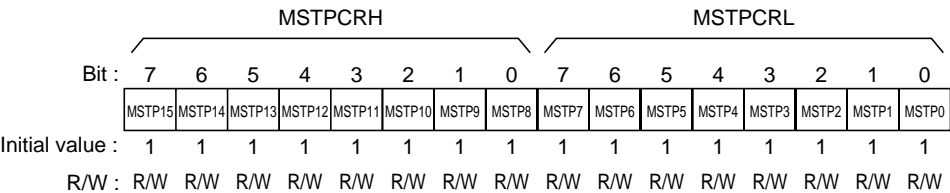
| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TLR27 | TLR26 | TLR25 | TLR24 | TLR23 | TLR22 | TLR21 | TLR20 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W |

The timer load register K (TLK) is an 8-bit write only register which works to set the reloading value of the TCK.

When the reloading value is set to the TLK, the value will be simultaneously loaded to the TCK and the TCK starts counting down from the set value. Also, during an auto reloading operation, when the TCK underflows, the value of the TLK will be loaded to the TCK. Consequently, the underflowing cycle can be set within the range of 1 to 256 input clocks. Nonetheless, when the 8/16 bit of the TMJ is being set to 1 (means when setting is made to 16-bit operation), writing is possible under the word command only. At this time, the upper 8 bits can be written into the TLK and the lower 8 bits can be written into the TLJ of the TMJ-1. The TLK and TCK are being allocated to the same address.

When reset, the TLK is initialized to H'FF.

14.2.8 Module Stop Control Register (MSTPCR)



The MSTPCR are 8-bit read/write twin registers which work to control the module stop mode. When the MSTP13 bit is set to 1, the Timer J stops its operation at the ending point of the bus cycle to shift to the module stop mode. For more information, see section 4.5, Module Stop Mode.

When reset, the MSTPCR is initialized to H'FFFF.

Bit 5: Module Stop (MSTP13)

This bit works to designate the module stop mode for the Timer J.

MSTPCRH

Bit 5

| MSTP13 | Description |
|--------|--|
| 0 | Cancels the module stop mode of the Timer J |
| 1 | Sets the module stop mode of the Timer J (Initial value) |

14.3 Operation

14.3.1 8-bit Reload Timer (TMJ-1)

The TMJ-1 is an 8-bit reload timer. As the clock source, dividing clock or edge signals through the $\overline{\text{IRQ1}}$ pin are being used. By selecting the edge signals through the $\overline{\text{IRQ1}}$ pin, it can also be used as an event counter. While it is working as an event counter, its reloading function is workable simultaneously. When data are written into the reloading register TLJ, these data will be written into the counter TCJ simultaneously. Also, when the counter TCJ underflows, the data of the reloading register TLJ will be reloaded to the counter TCJ.

When the counter underflows, TMJ1I interrupt requests will be issued.

The underflow will be toggled and, by a appropriate selection of the dividing clock, buzzer outputs will be issued or carrier frequencies for remote controlling transmissions can be acquired.

The TMJ-1 and TMJ-2, in combination, can be used as a 16-bit reload timer. Nonetheless, when they are being used, in combination, as a 16-bit timer, word command only is valid and the TCK works as the down counter for the upper 8 bits and the TCJ works as the down counter for the lower 8 bits, means a reloading register of total 16 bits.

When data are written into a 16-bit reloading register, the same data will be written into the 16-bit counter.

Also, when the 16-bit counter underflows, the data of the 16-bit reloading register will be reloaded into the counter.

Even when they are making a 16-bit operation, the TMJ1I interrupt requests of the TMJ-1 and BUZZ outputs are effective. In case these functions are not necessary, make them invalid by programming.

The TMJ-1 and TMJ-2, in combination, can be used for remote controlled data transmission. Regarding the remote controlled data transmission, see section 14.3.3, Remote Controlled Data Transmission.

14.3.2 8-bit Reload Timer (TMJ-2)

The TMJ-2 is an 8-bit down-counting reload timer. As the clock source, dividing clock, edge signals through the IRQ2 pin or the underflow signals from the TMJ-1 are being used. By selecting the edge signals through the $\overline{\text{IRQ2}}$ pin, it can also be used as an event counter. While it is working as an event counter, its reloading function is workable simultaneously.

When data are written into the reloading register TLK, these data will be written into the counter TCK simultaneously. Also, when the counter TCK underflows, the data of the reloading register TLK will be made to the data counter TCK.

When the counter underflows, TMJ2I interrupt requests will be issued.

The TMJ-2 and TMJ-1, in combination, can be used as a 16-bit reload timer. For more information on the 16-bit reload timer, see section 14.3.1, 8-bit Reload Timer (TMJ-1).

The TMJ-2 and TMJ-1, in combination, can be operated by remote controlled data transmission.

Regarding the remote controlled data transmission, see section 14.3.3, Remote Controlled Data Transmission.

14.3.3 Remote Controlled Data Transmission

The Timer J is capable of making remote controlled data transmission. The carrier frequencies for the remote controlled data transmission can be generated by the TMJ-1 and the burst width duration and the space width duration can be setup by the TMJ-2.

The data having been written into the reloading register TMJ-1 and into the burst/space duration register (TLK) of the TMJ-2 will be loaded to the counter at the same time as the remote controlled data transmission starts. (Remote controlled data transmission starting bit (ST) \leftarrow 1)

While remote controlled data transmission is being made, the contents of the burst/space duration register will be loaded to the counter only while reloading is being made by underflow signals. Even when a writing is made to the burst/space duration register while remote controlled data transmission is being made, reloading operation will not be made until an underflow signal is issued. The TMJ-2 issues TMJ2I interrupt requests by the underflow signals. The TMJ-1 performs normal reloading operation (including the TMJ1I interrupt requests).

Figure 14.2 shows the output waveform for the remote controlled data transmission function. When a shift to the low power consumption mode is effected while remote controlled data transmission is being made, the ST bit will be cleared to 0. When resuming the remote controlled data transmission after returning to the active mode, write 1.

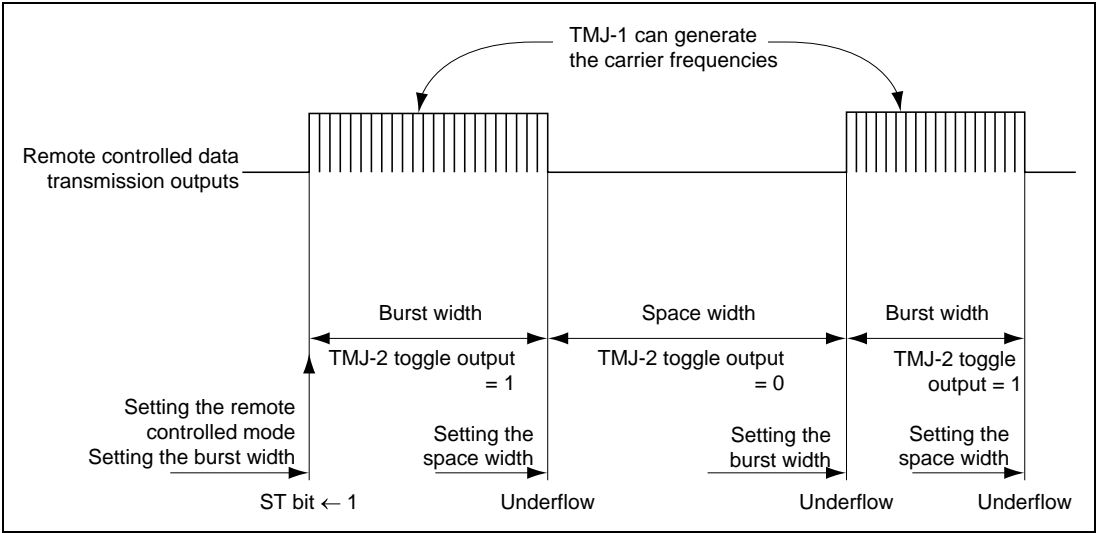


Figure 14.2 Remote Controlled Data Transmission Output Waveform

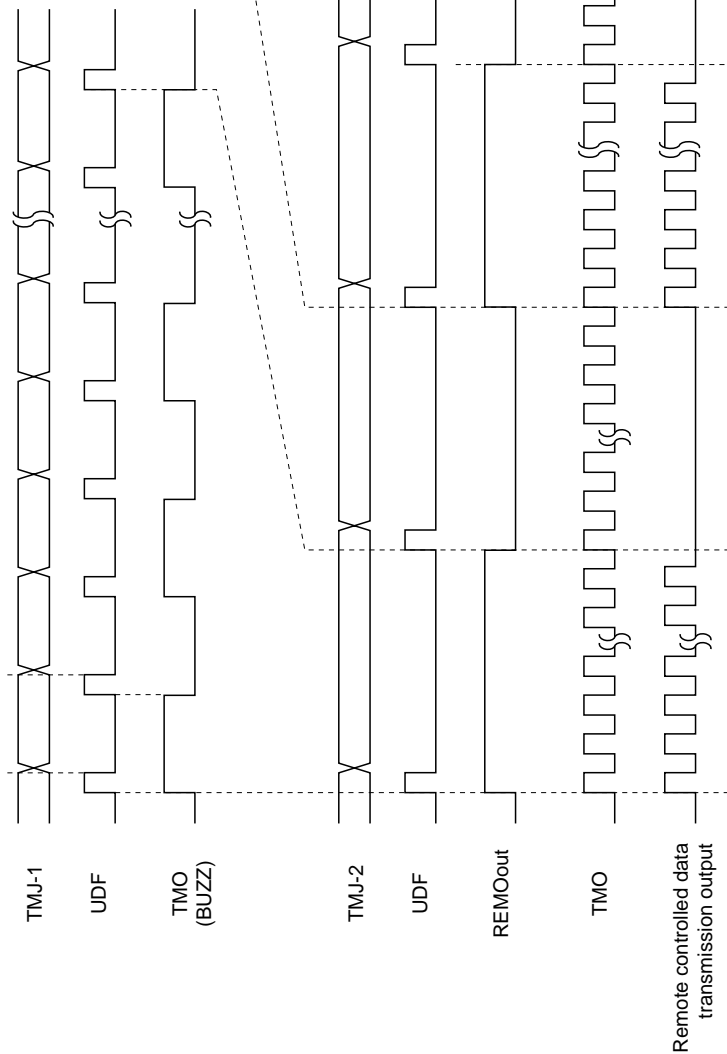


Figure 14.3 Timer Output Timing

When the Timer J is set to the remote controlled operation mode, since the start bit (ST) is being set or cleared in synchronization with the inputting clock to the TMJ-2, a delay upto a cycle of the inputting clock at the maximum occurs, namely, after the ST bit has been set to 1 until the remote controlled data transmission starts. Consequently, when the TLK is updated during the period after setting the ST bit to 1 until the next cycle of the inputting clock comes, the initial burst width will be changed like shown in figure 14.4.

Therefore, when making remote controlled data transmission, determine I/O of the TGL bit at the time of the first burst width control operation without fail. (Or, set the space width to the TLK after waiting for a cycle of the inputting clock.)

After that, operations can be continued by interrupts.

Similarly, pay attention to the control works when ending remote controlled data transmission.

Example)

- 1) Set the burst width with the TLK.
- 2) ST bit $\leftarrow 1$
- 3) Execute the procedure 4) if the TGL flag = 1.
- 4) Set the space width with the TLK under the status where the TGL flag = 1.
- 5] Make TMJ-2 interrupt.
- 6] Set the burst width with the TLK.
- :
- n) After making TMJ-2 interrupt, make setting of the ST $\leftarrow 0$ under the status where the TGL flag = 0.

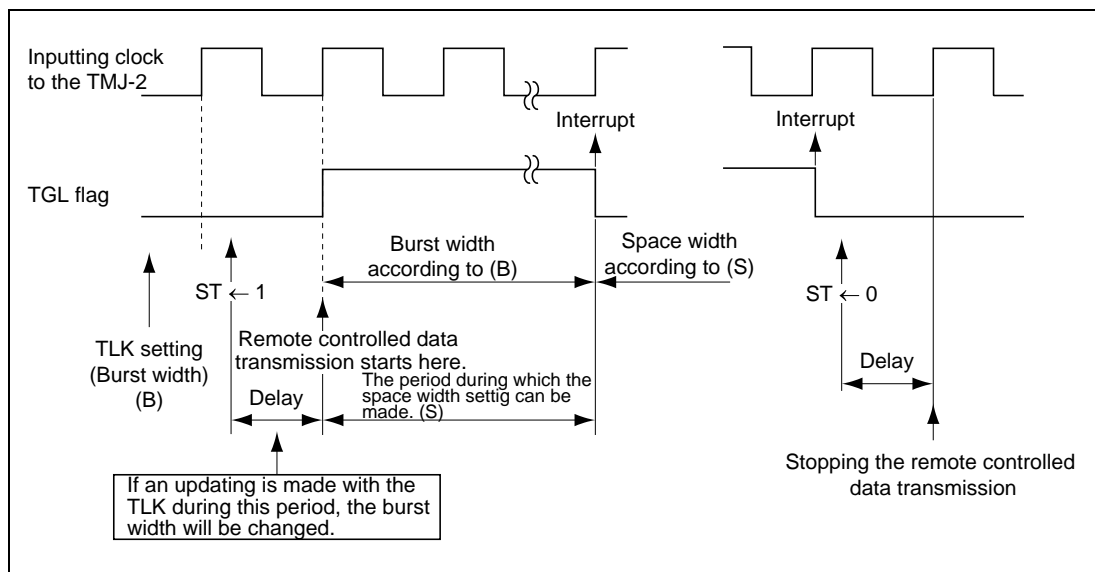


Figure 14.4 Controls of the Remote Controlled Data Transmission

Section 15 Timer L

15.1 Overview

The Timer L is an 8-bit up/down counter using the control pulses or the CFG division signals as the clock source.

15.1.1 Features

Features of the Timer L are as follows:

- Choices of two different types of internal clocks ($\phi/128$ and $\phi/64$), DVCFG2 (CFG division signal 2), PB and REC-CTL (control pulses) are available for your selection.
 - In case the PB-CTL is not available, such as when reproducing un-recorded tapes, tape count can be made by the DVCFG2.
 - Selection of the leading edge or the trailing edge is workable with the CTL pulse counting.
- Interrupts occur when the counter overflows or underflows and at occurrences of compare match clear.
- It is possible to switch over between the up-counting and down-counting functions with the counter.

15.1.2 Block Diagram

Figure 15.1 shows a block diagram of the Timer L.

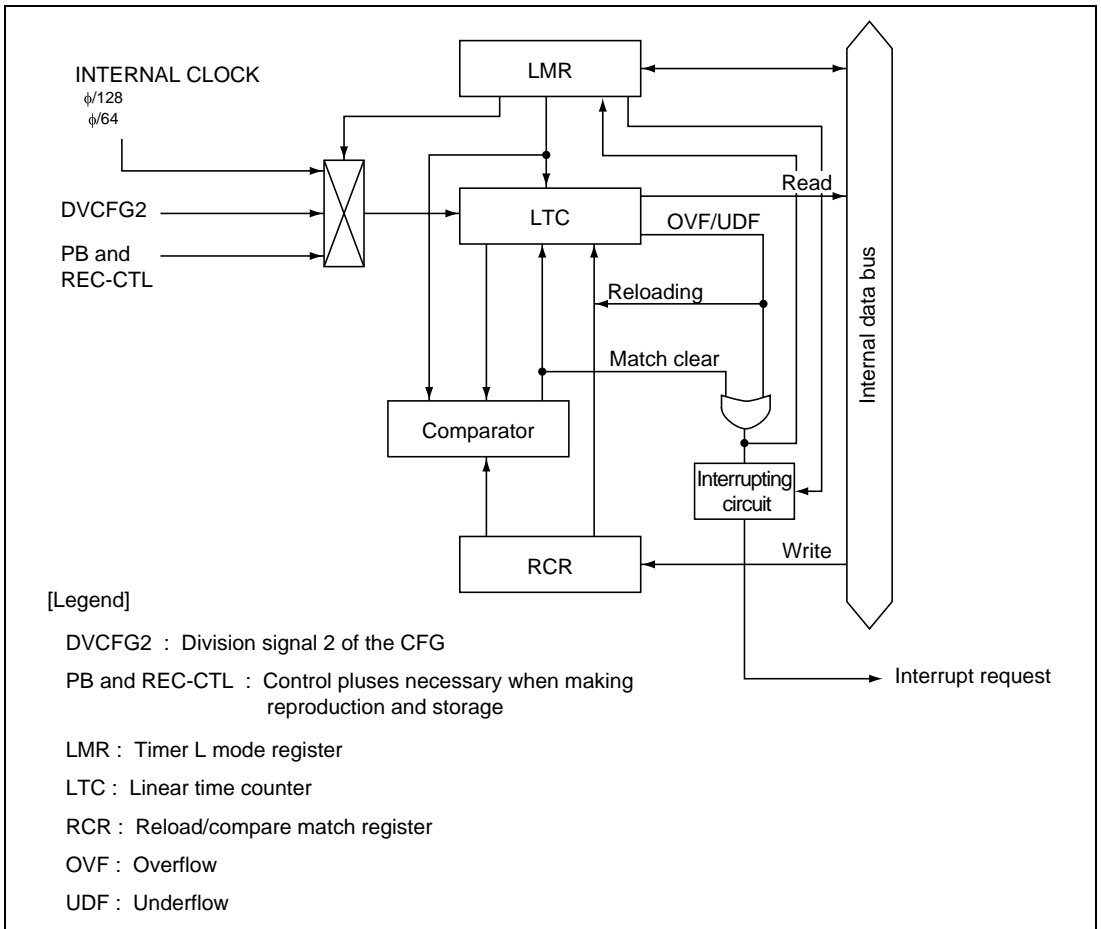


Figure 15.1 Block Diagram of the Timer L

15.1.3 Register Configuration

Table 15.1 shows the register configuration of the Timer L. The linear time counter (LTC) and the reload compare patch register (RCR) are being allocated to the same address.

Reading or writing determines the accessing register.

Table 15.1 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-------------------------------|----------------|------------|-------------|----------------------|-----------------|
| Timer L mode register | LMR | R/W | Byte | H'30 | H'D112 |
| Linear time counter | LTC | R | Byte | H'00 | H'D113 |
| Reload/compare match register | RCR | W | Byte | H'00 | H'D113 |

Note: * Lower 16 bits of the address.

15.2 Descriptions of Respective Registers

15.2.1 Timer L Mode Register (LMR)

| | | | | | | | | |
|-----------------|--------|------|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | LMIF | LMIE | — | — | IMR3 | IMR2 | IMR1 | IMR0 |
| Initial value : | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/W | — | — | R/W | R/W | R/W | R/W |

Note: * Only 0 can be written to clear the flag.

The timer L mode register (LMR) is an 8-bit read/write register which works to control the interrupts, to select between up-counting and down-counting and to select the clock source. When reset, the LMR is initialized to H'30.

Bit 7: Timer L Interrupt Requesting Flag (LMIF)

This is the Timer L interrupt requesting flag. It indicates occurrence of overflow or underflow of the LTC or occurrence of compare match clear.

Bit 7

| LMIF | Description |
|------|---|
| 0 | [Clearing conditions] When 0 is written after reading 1 (Initial value) |
| 1 | [Setting conditions] When the LTC overflows, underflows or when compare match clear has occurred |

Bit 6: Enabling Interrupt of the Timer L (LMIE)

This bit works to permit/prohibit occurrence of interrupt of the Timer L when the LTC overflows, underflows or when compare match clear has occurred.

Bit 6

| LMIE | Description |
|------|--|
| 0 | Prohibits occurrence of interrupt of the Timer L (Initial value) |
| 1 | Permits occurrence of interrupt of the Timer L |

Bits 5 and 4: Reserved

When they are read, 1 will always be readout. Writes are disabled.

Bit 3: Up-count/Down-count Control (LMR3)

This bit is for selection if the Timer L is to be controlled to the up-counting function or down-counting function.

(1) When Controlled to the Up-counting Function

- When any other values than H'00 are input to the RCR, the LTC will be cleared to H'00 before starting counting up. When the LTC value and the RCR value match, the LTC will be cleared to H'00. Also, interrupt requests will be issued by the match signal. (Compare match clear function)
- When H'00 is set to the RCR, the counter makes 8-bit interval timer operation to issue a interrupt request when overflowing occurs. (Interval timer function)

(2) When Controlled to the Down-counting Function

- When a value is set to the RCR, the set value is reloaded to the LTC and counting down starts from that value. When the LTC underflows, the value of the RCR will be reloaded to the LTC. Also, when the LTC underflows, a interrupt request will be issued. (Auto reload timer function)

Bit 3

| LMR3 | Description |
|-------------|---|
| 0 | Controlling to the up-counting function (Initial value) |
| 1 | Controlling to the up-counting function |

Bits 2 to 0: Clock Selection (LMR2 to LMR0)

The bits LMR2 to LMR0 work to select the clock to input to the Timer L. Selection of the leading edge or the trailing edge is workable for counting by the PB and the REC-CTL.

| Bit 2 | Bit 1 | Bit 0 | Description |
|--------------|--------------|--------------|---|
| R2 | LMR1 | LMR0 | |
| 0 | 0 | 0 | Counts at the rising edge of the PB and REC-CTL (Initial value) |
| | | 1 | Counts at the falling edge of the PB and REC-CTL |
| | 1 | * | Counts the DVCFG2 |
| 1 | 0 | * | Counts at $\phi/128$ of the internal clock |
| | 1 | * | Counts at $\phi/64$ of the internal clock |

Note: * Don't care.

15.2.2 Linear Time Counter (LTC)

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | LTC7 | LTC6 | LTC5 | LTC4 | LTC3 | LTC2 | LTC1 | LTC0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R |

The linear time counter (LTC) is a readable 8-bit up/down-counter. The inputting clock can be selected by the LMR2 to LMR0 bits of the LMR.

When reset, the LTC is initialized to H'00.

15.2.3 Reload/Compare Match Register (RCR)

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | RCR7 | RCR6 | RCR5 | RCR4 | RCR3 | RCR2 | RCR1 | RCR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

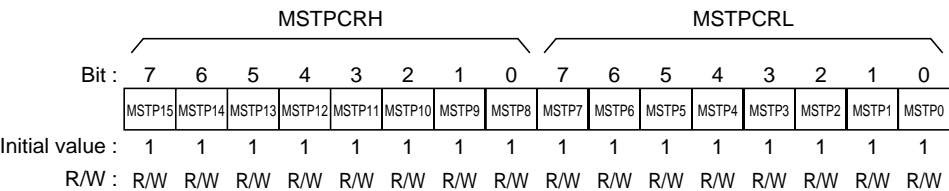
The reload/compare match register (RCR) is an 8-bit write only register.

When the Timer L is being controlled to the up-counting function, when a compare match value is set to the RCR, the LTC will be cleared at the same time and the LTC will then start counting up from the initial value (H'00).

While, when the Timer L is being controlled to the down-counting function, when a reloading value is set to the RCR, the same value will be loaded to the LTC at the same time and the LTC will then start counting up from said value. Also, when the LTC underflows, the value of the RCR will be reloaded to the LTC.

When reset, the RCR is initialized to H'00.

15.2.4 Module Stop Control Register (MSTPCR)



The MSTPCR are 8-bit read/write twin registers which work to control the module stop mode. When the MSTP12 bit is set to 1, the Timer L stops its operation at the ending point of the bus cycle to shift to the module stop mode. For more information, see section 4.5, Module Stop Mode.

When reset, the MSTPCR is initialized to H'FFFF.

Bit 4: Module Stop (MSTP12)

This bit works to designate the module stop mode for the Timer L.

MSTPCRH

Bit 4

| MSTP12 | Description |
|--------|--|
| 0 | Cancels the module stop mode of the Timer L |
| 1 | Sets the module stop mode of the Timer L (Initial value) |

15.3 Operation

The Timer L is an 8-bit up/down counter.

The inputting clock for the Timer L can be selected by the LMR2 to LMR0 bits of the LMR from the choices of the internal clock ($\phi/128$ and $\phi/64$), DVCDG2, PB and REC-CTL.

The Timer L is provided with three different types of operation modes, namely, the compare match clear mode when controlled to the up-counting function, the auto reloading mode when controlled to the down-counting function and the interval timer mode.

Respective operation modes and operation methods will be explained below.

15.3.1 Compare Match Clear Operation

When the LMR3 bit of the LMR is cleared to 0, the Timer L will be controlled to the up-counting function.

When any other values than H'00 are written into the RCR, the LTC will be cleared to H'00 simultaneously before starting counting up.

Figure 15.2 shows the clear timing of the LTC. When the LTC value and the RCR value match (compare match), the LTC readings will be cleared to H'00 to resume counting from H'00.

Figure 15.3 indicated on the next page shows the compare match clear timing.

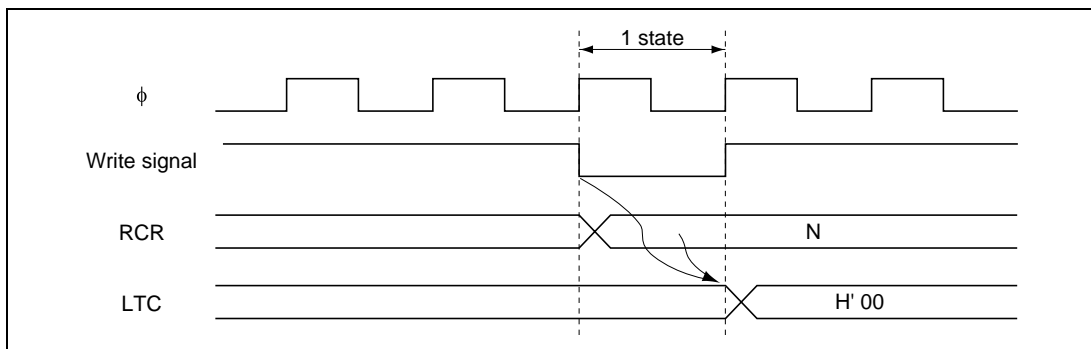


Figure 15.2 RCR Writing and LTC Clearing Timing Chart

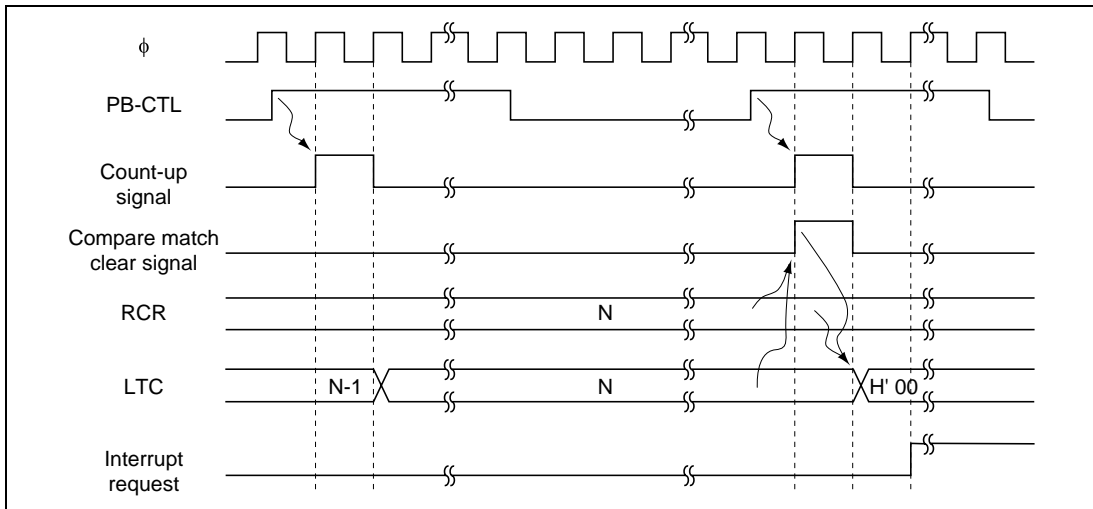


Figure 15.3 Compare Match Clearing Timing Chart
(In case the rising edge of the PB-CTL is selected)

Section 16 Timer R

16.1 Overview

The Timer R consists of triple 8-bit down-counters. It carries VCR mode identification function and slow tracking function in addition to the reloading function and event counter function.

16.1.1 Features

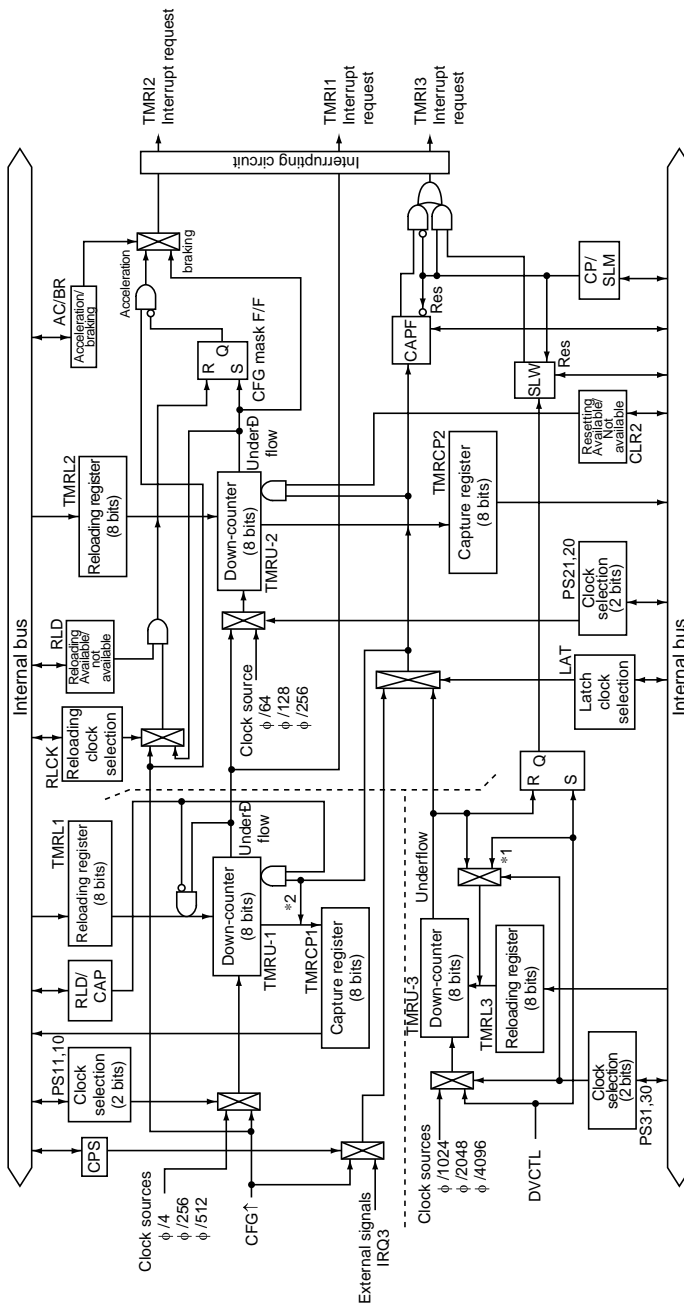
The Timer R consists of triple 8-bit reloading timers. By combining the functions of three units of reloading timers/counters and by combining three units of timers, it can be used for the following applications:

- (1) Applications making use of the functions of three units of reloading timers.
- (2) For identification of the VCR mode.
- (3) For reel controls.
- (4) For acceleration and braking of the capstan motor when being applied to intermittent movements.
- (5) Slow tracking mono-multi applications.

16.1.2 Block Diagram

The Timer R consists of three units of reload timer counters, namely, two units of reload timer counters equipped with capturing function (TMRU-1 and TMRU-2) and a unit of reload timer counter (TMRU-3).

Figure 16.1 is a block diagram of the Timer R.



- Notes:
1. When the DVCCTL is being used as the clock source, reloading will be made when the counter underflows and when the dividing clock is being used as the clock source, reloading will be made by the DVCCTL.
 2. When the LAT bit = 0, the capture signal against the TMRL-1 will not be output.

Figure 16.1 Block Diagram of the Timer R

16.1.3 Pin Configuration

Table 16.1 shows the pin configuration of the Timer R.

Table 16.1 Pin Configuration

| Name | Abbrev. | I/O | Function |
|-----------------------------|--------------------------|-------|---|
| Input capture inputting pin | $\overline{\text{IRQ3}}$ | Input | Input capture inputting for the Timer R |

16.1.4 Register Configuration

Table 16.2 shows the register configuration of the Timer R.

Table 16.2 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|---------------------------------|---------|-----|------|---------------|---------|
| Timer R mode register 1 | TMRM1 | R/W | Byte | H'00 | H'D118 |
| Timer R mode register 2 | TMRM2 | R/W | Byte | H'00 | H'D119 |
| Timer R control/status register | TMRCs | R/W | Byte | H'03 | H'D11F |
| Timer R capture register 1 | TMRCp1 | R | Byte | H'FF | H'D11A |
| Timer R capture register 2 | TMRCp2 | R | Byte | H'FF | H'D11B |
| Timer R load register 1 | TMRL1 | W | Byte | H'FF | H'D11C |
| Timer R load register 2 | TMRL2 | W | Byte | H'FF | H'D11D |
| Timer R load register 3 | TMRL3 | W | Byte | H'FF | H'D11E |

Note: Memories of respective registers will be preserved even under the low power consumption mode. Nonetheless, the CAPF flag and SLW flag of the TMRM2 will be cleared to 0.

16.2 Descriptions of Respective Registers

16.2.1 Timer R Mode Register 1 (TMRM1)

| | | | | | | | | |
|-----------------|------|-------|-----|------|------|------|---------|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CLR2 | AC/BR | RLD | RLCK | PS21 | PS20 | RLD/CAP | CPS |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The timer R mode register 1 (TMRM1) works to control the acceleration and braking processes and to select the inputting clock for the TMRU-2. This is an 8-bit read/write register. When reset, the TMRM1 is initialized to H'00.

Bit 7: Selecting Clearing/Not Clearing of TMRU-2 (CLR2)

This bit is used for selecting if the TMRU-2 counter reading is to be cleared or not as it is captured.

Bit 7

| CLR2 | Description |
|------|--|
| 0 | TMRU-2 counter reading is not to be cleared as soon as it is captured. (Initial value) |
| 1 | TMRU-2 counter reading is to be cleared as soon as it is captured |

Bit 6: Selecting the Acceleration/Braking Processing (AC/BR)

This bit works to control occurrences of interrupt requests to detect completion of acceleration or braking while the capstan motor is making intermittent revolutions. For more information, see section 16.3.6, Acceleration and Braking of the Capstan Motor.

Bit 6

| AC/BR | Description |
|-------|------------------------------|
| 0 | Acceleration (Initial value) |
| 1 | Braking |

Bit 5: Selection if Using the TMRU-2 for Reloading or Not Doing So (RLD)

This bit is used for selecting if the TMRU-2 reload function is to be turned on or not.

Bit 5

| RLD | Description |
|-----|--|
| 0 | Not using the TMRU-2 as the reload timer (Initial value) |
| 1 | Using the TMRU-2 as the reload timer |

Bit 4: Selection of the Reloading Timing for the TMRU-2 (RLCK)

This bit works to select if the TMRU-2 is reloading by the CFG or by underflowing of the TMRU-2 counter. This choice is valid only when the bit 5 (RLD) is being set to 1.

Bit 4

| RLCK | Description |
|------|---|
| 0 | Reloading at the rising edge of the CFG (Initial value) |
| 1 | Reloading by underflowing of the TMRU-2 |

Bits 3 and 2: Selecting the Clock Source for the TMRU-2 (PS21 and PS20)

These bits work to select the inputting clock to the TMRU-2.

| Bit 3 | Bit 2 | Description |
|-------|-------|--|
| 0 | 0 | Counting by underflowing of the TMRU-1 (Initial value) |
| | 1 | Counting by the PSS, $\phi/256$ |
| 1 | 0 | Counting by the PSS, $\phi/128$ |
| | 1 | Counting by the PSS, $\phi/64$ |

Bit 1: Selection of the Operation Mode of the TMRU-1 (RLD/CAP)

This bit works to select if the operation mode of the TMRU-1 is reload timer mode or capture timer mode.

Under the capture timer mode, reloading operation will not be made. Also, the counter reading will be cleared as soon as capture has been made.

Bit 1

| RLD/CAP | Description |
|---------|---|
| 0 | The TMRU-1 works as the reloading timer (Initial value) |
| 1 | The TMRU-1 works as the capture timer |

Bit 0: Selection of the Capture Signals of the TMRU-1 (CPS)

In combination with the LAT bit (Bit 7) of the TMR2, this bit works to select the capture signals of the TMRU-1. This bit becomes valid when the LAT bit is being set to 1. It will also become valid when the RLD/CAP bit (Bit 1) is being set to 1. Nonetheless, it will be invalid when the RLD/CAP bit (Bit 1) is being set to 0.

Bit 0

| CPS | Description |
|-----|---|
| 0 | Capture signals at the rising edge of the CFG (Initial value) |
| 1 | Capture signals at the edge of the IRQ3 |

16.2.2 Timer R Mode Register 2 (TMRM2)

| | | | | | | | | |
|-----------------|-----|------|------|------|------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | LAT | PS11 | PS10 | PS31 | PS30 | CP/SLM | CAPF | SLW |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/(W)* | R/(W)* |

The timer R mode register 2 (TMRM2) is an 8-bit read/write register which works to identify the operation mode and to control the slow tracking processing.

When reset, the TMRM2 is initialized to H'00.

Note: * The CAPF bit and the SLW bit, respectively, works to latch the interrupt causes and writing 0 only is valid. Consequently, when these bits are being set to 1, respective interrupt requests will not be issued. Therefore, it is necessary to check these bits during the course of the interrupt processing routine to have them cleared.

Also, priority is given to the set and, when an interrupt cause occur while the a clearing command (BCLR, MOV, etc.) is being executed, the CAPF bit and the SLW bit will not be cleared respectively and it thus becomes necessary to pay attention to the clearing timing.

Bit 7: Selection of the Capture Signals of the TMRU-2 (LAT)

In combination with the CPS bit (Bit 0) of the TMRM1, this bit works to select the capture signals of the TMRU-2.

| TMRM2 | TMRM1 | |
|--------------|--------------|---|
| Bit 7 | Bit 0 | |
| LAT | CPS | Description |
| 0 | * | Captures when the TMRU-3 underflows (Initial value) |
| 1 | 0 | Captures at the rising edge of the CFS |
| | 1 | Captures at the edge of the IRQ3 |

Note: * Don't care.

Bits 6 and 5: Selecting the Clock Source for the TMRU-1 (PS11 and PS10)

These bits work to select the inputting clock to the TMRU-1.

| Bit 6 | Bit 5 | |
|--------------|--------------|--|
| PS11 | PS10 | Description |
| 0 | 0 | Counting at the rising edge of the CFG (Initial value) |
| | 1 | Counting by the PSS, $\phi/4$ |
| 1 | 0 | Counting by the PSS, $\phi/256$ |
| | 1 | Counting by the PSS, $\phi/512$ |

Bits 4 and 3: Selecting the Clock Source for the TMRU-3 (PS31 and PS30)

These bits work to select the inputting clock to the TMRU-3.

| Bit 4 | Bit 3 | |
|--------------|--------------|---|
| PS31 | PS30 | Description |
| 0 | 0 | Counting at the rising edge of the DVCTL from the dividing circuit. (Initial value) |
| | 1 | Counting by the PSS, $\phi/4096$ |
| 1 | 0 | Counting by the PSS, $\phi/2048$ |
| | 1 | Counting by the PSS, $\phi/1024$ |

Bit 2: Selection of Interrupt Causes (CP/SLM)

This bit works to select the interrupt causes for the TMRI3.

Bit 2

| CP/SLM | Description |
|--------|---|
| 0 | Makes interrupt requests upon the capture signals of the TMRU-2 valid (Initial value) |
| 1 | Makes interrupt requests upon ending of the slow tracking mono-multi valid |

Bit 1: Capture Signal Flag (CAPF)

This is a flag being set out by the capture signal of the TMRU-2. Although both reading/writing are possible, 0 only is valid for writing.

Also, priority is being given to the set and, when the "capture signal" and "writing 0" occur simultaneously, this flag bit remains being set to 1 and the interrupt request will not be issued and it is necessary to be attentive about this fact.

When the CP/SLM bit (Bit 2) is being set to 1, this CAPF bit should always be set to 0.

The CAPF flag is cleared to 0 under the low power consumption mode.

Bit 1

| CAPF | Description |
|------|---|
| 0 | [Clearing conditions] When 0 is written after reading 1 (Initial value) |
| 1 | [Setting conditions] At occurrences of the TMRU-2 capture signals while the CP/SLM bit is being set to 0 |

Bit 0: Slow Tracking Mono-multi Flag (SLW)

This is a flag being set out when the slow tracking mono-multi processing ends. Although both reading/writing are possible, 0 only is valid for writing.

Also, priority is being given to the set and, when "ending of the slow tracking mono-multi processing" and "writing 0" occur simultaneously, this flag bit remains being set to 1 and the interrupt request will not be issued and it is necessary to be attentive about this fact.

When the CP/SLM bit (Bit 2) is being set to 0, this SLW bit should always be set to 0.

The SLW flag is cleared to 0 under the low power consumption mode.

Bit 0

| SLW | Description |
|-----|--|
| 0 | [Clearing conditions] When 0 is written after reading 1 (Initial value) |
| 1 | [Setting conditions] When the slow tracking mono-multi processing ends while the CP/SLM bit is being set to 1 |

16.2.3 Timer R Control/Status Register (TMRCS)

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|---|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMRI3E | TMRI2E | TMRI1E | TMRI3 | TMRI2 | TMRI1 | — | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| R/W : | R/W | R/W | R/W | R/(W)* | R/(W)* | R/(W)* | — | — |

Note: * Only 0 can be written to clear the flag.

The timer R control/status register (TMRCS) works to control the interrupts of the Timer R. The TMRCS is an 8-bit read/write register. When reset, the TMRCS is initialized to H'03.

Bit 7: Enabling the TMRI3 Interrupt (TMRI3E)

This bit works to permit/prohibit occurrence of the TMRI3 interrupt when an interrupt cause being selected by the CP/SLM bit of the TMRM2 has occurred, such as occurrences of the TMRU-2 capture signals or when the slow tracking mono-multi processing ends, and the TMRI3 has been set to 1.

Bit 7

| TMRI3E | Description |
|--------|---|
| 0 | Prohibits occurrences of TMRI3 interrupts (Initial value) |
| 1 | Permits occurrences of TMRI3 interrupts |

Bit 6: Enabling the TMRI2 Interrupt (TMRI2E)

This bit works to permit/prohibit occurrence of the TMRI2 interrupt when the TMRI2 has been set to 1 by issuance of the underflow signal of the TMRU-2 or by ending of the slow tracking mono-multi processing.

Bit 6

| TMRI2E | Description |
|--------|---|
| 0 | Prohibits occurrences of TMRI2 interrupts (Initial value) |
| 1 | Permits occurrences of TMRI2 interrupts |

Bit 5: Enabling the TMRI1 Interrupt (TMRI1E)

This bit works to permit/prohibit occurrence of the TMRI1 interrupt when the TMRI1 has been set to 1 by issuance of the underflow signal of the TMRU-1.

Bit 5

| TMRI1E | Description |
|--------|---|
| 0 | Prohibits occurrences of TMRI1 interrupts (Initial value) |
| 1 | Permits occurrences of TMRI1 interrupts |

Bit 4: TMRI3 Interrupt Requesting Flag (TMRI3)

This is the TMRI3 interrupt requesting flag.

It indicates occurrence of an interrupt cause being selected by the CP/SLM bit of the TMRM2, such as occurrences of the TMRU-2 capture signals or ending of the slow tracking mono-multi processing.

Bit 4

| TMRI3 | Description |
|-------|--|
| 0 | [Clearing conditions] (Initial value) When 0 is written after reading 1 |
| 1 | [Setting conditions] At occurrence of the interrupt cause being selected by the CP/SLM bit of the TMRM2 |

Bit 3: TMRI2 Interrupt Requesting Flag (TMRI2)

This is the TMRI2 interrupt requesting flag.

It indicates occurrences of the TMRU-2 underflow signals or ending of the acceleration/braking processing of the capstan motor.

Bit 3

| TMRI2 | Description |
|-------|---|
| 0 | [Clearing conditions] (Initial value) When 0 is written after reading 1 |
| 1 | [Setting conditions] At occurrences of the TMRU-2 underflow signals or ending of the acceleration /braking processing of the capstan motor |

Bit 2: TMRI1 Interrupt Requesting Flag (TMRI1)

This is the TMRI1 interrupt requesting flag.

It indicates occurrences of the TMRU-1 underflow signals.

Bit 2

| TMRI1 | Description |
|-------|---|
| 0 | [Clearing conditions] When 0 is written after reading 1. (Initial value) |
| 1 | [Setting conditions] When the TMRU-1 underflows. |

Bits 1 and 0: Reserved

When they are read, 1 will always be readout. Writes are disabled.

16.2.4 Timer R Capture Register 1 (TMRCP1)

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMRC17 | TMRC16 | TMRC15 | TMRC14 | TMRC13 | TMRC12 | TMRC11 | TMRC10 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R | R | R | R | R | R | R | R |

The timer R capture register 1 (TMRCP1) works to store the capture data of the TMRU-1.

During the course of the capturing operation, the TMRU-1 counter readings are captured by the TMRCP1 at the CFG edge or the IRQ3 edge. The capturing operation of the TMRU-1 is being performed using 16 bits, in combination with the capturing operation of the TMRU-2.

The TMRCP1 is an 8-bit read only register. When reset, the TMRC1 is initialized to H'FF.

- Notes:
1. When the TMRCP1 is readout while the capture signal is being received, the reading data become unstable. Pay attention to the timing for reading out.
 2. When a shift to the low power consumption mode is made while the capturing operating is in progress, the counter reading becomes unstable. After returning to the active mode, always write "H'FF" into the TMRL1 to initialize the counter.

16.2.5 Timer R Capture Register 2 (TMRC2P2)

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMRC27 | TMRC26 | TMRC25 | TMRC24 | TMRC23 | TMRC22 | TMRC21 | TMRC20 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R | R | R | R | R | R | R | R |

The timer R capture register 2 (TMRC2P2) works to store the capture data of the TMRU-2. At each CFG edge, IRQ3 edge, or at occurrence of underflow of the TMRU-3, the TMRU-2 counter readings are captured by the TMRC2P2.

The TMRC2P2 is an 8-bit read only register. When reset, the TMRC2P2 will be initialized into H'FF.

- Notes:
1. When the TMRC2P2 is readout while the capture signal is being received, the reading data become unstable. Pay attention to the timing for reading out.
 2. When a shift to the low power consumption mode is made, the counter reading becomes unstable. After returning to the active mode, always write "H'FF" into the TMRL2 to initialize the counter.

16.2.6 Timer R Load Register 1 (TMRL1)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMR17 | TMR16 | TMR15 | TMR14 | TMR13 | TMR12 | TMR11 | TMR10 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W |

The timer R load register 1 (TMRL1) is an 8-bit write only register which works to set the load value of the TMRU-1.

When a load value is set to the TMRL1, the same value will be set to the TMRU-1 counter simultaneously and the counter starts counting down from the set value. Also, when the counter underflows during the course of the reload timer operation, the TMRL1 value will be set to the counter.

When reset, the TMRL1 is initialized to H'FF.

16.2.7 Timer R Load Register 2 (TMRL2)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMR27 | TMR26 | TMR25 | TMR24 | TMR23 | TMR22 | TMR21 | TMR20 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W |

The timer R load register 2 (TMRL2) is an 8-bit write only register which works to set the load value of the TMRU-2.

When a load value is set to the TMRL2, the same value will be set to the TMRU-2 counter simultaneously and the counter starts counting down from the set value. Also, when the counter underflows or a CFG edge is detected during the course of the reload timer operation, the TMRL2 value will be set to the counter.

When reset, the TMRL2 is initialized to H'FF.

16.2.8 Timer R Load Register 3 (TMRL3)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMR37 | TMR36 | TMR35 | TMR34 | TMR33 | TMR32 | TMR31 | TMR30 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W |

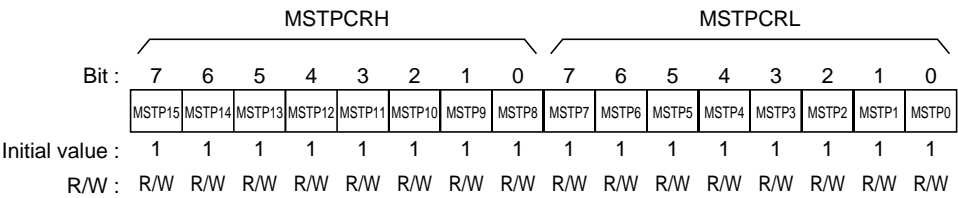
The timer R load register 3 (TMRL3) is an 8-bit write only register which works to set the load value of the TMRU-3.

When a load value is set to the TMRL3, the same value will be set to the TMRU-3 counter simultaneously and the counter starts counting down from the set value. Also, when the counter underflows or a DVCTL edge is detected, the TMRL2 value will be set to the counter.

(Reloading will be made by the underflowing signals when the DVCTL signal is selected as the clock source, and reloading will be made by the DVCTL signals when the dividing clock is selected as the clock source.)

When reset, the TMRL3 is initialized to H'FF.

16.2.9 Module Stop Control Register (MSTPCR)



The MSTPCR are 8-bit read/write twin registers which work to control the module stop mode. When the MSTP11 bit is set to 1, the Timer R stops its operation at the ending point of the bus cycle to shift to the module stop mode. For more information, see section 4.5, Module Stop Mode.

When reset, the MSTPCR is initialized to H'FFFF.

Bit 3: Module Stop (MSTP11)

This bit works to designate the module stop mode for the Timer R.

MSTPCRH

Bit 3

MSTP11

Description

| | | |
|---|---|-----------------|
| 0 | Cancels the module stop mode of the Timer R | |
| 1 | Sets the module stop mode of the Timer R | (Initial value) |

16.3 Operation

16.3.1 Reload Timer Counter Equipped with Capturing Function TMRU-1

The reload timer counter equipped with capturing function, TMRU-1, consists of an 8-bit down-counter, a reloading register and a capture register.

The clock source can be selected from among the leading edge of the CFG signals and three types of dividing clocks. It is also selectable whether using it as a reload counter or as a capture counter. Even when the capturing function is selected, the counter readings can be updated by writing the values into the reloading register.

When the counter underflows, the TMRI1 interrupt request will be issued.

The initial values of the TMRU-1 counter, reloading register and capturing register are all H'FF.

(1) Operation of the Reload Timer

When a value is written into the reloading register, the same value will be written into the counter simultaneously. Also, when the counter underflows, the reloading register value will be reloaded to the counter. The TMRU-1 is a dividing circuit for the CFG. In combination with the TMRU-2 and TMRU-3, it can also be used for the mode identification purpose.

(2) Capturing Operation

Capturing operation is carried out in combination with the TMRU-2 using the combined 16 bits. It can be so programmed that the counter may be cleared by the capture signal. The CFG edges or IRQ3 edges are used as the capture signals. It is possible to issue the TMRI3 interrupt request by the capture signal.

In addition to the capturing function being worked out in combination with the TMRU-2, the TMRU-1 can be used as a 16-bit CFG counter. Selecting the IRQ3 as the capture signal, the CFG within the duration of the reel pulse being input into the $\overline{\text{IRQ3}}$ pin can be counted by the TMRU-1.

16.3.2 Reload Timer Counter Equipped with Capturing Function TMRU-2

The reload timer counter equipped with capturing function, TMRU-2, consists of an 8-bit down-counter, a reloading register and a capture register.

The clock source can be selected from among the underflowing signal of the TMRU-1 and three types of dividing clocks. Also, although the reloading function is workable during its capturing operation, equipping or not of the reloading function is selectable. Even when without-reloading-function is chosen, the counter reading can be updated by writing the values to the reloading register.

When the counter underflows, the TMRI2 interrupt request will be issued.

The initial values of the TMRU-2 counter, reloading register and capturing register are all H'FF.

(1) Operation of the Reload Timer

When a value is written into to the reloading register, the same value will be written into the counter, simultaneously. Also, when the counter underflows, the reloading register value will be reloaded to the counter.

The TMRU-2 can make acceleration and braking work for the capstan motor using the reload timer operation.

(2) Capturing Operation

Using the capture signals, the counter reading can be latched into the capturing register. As the capture signal, you can choose from among edges of the CFG, edges of the IRQ3 or the underflow signals of the TMRU-3. It is possible to issue the TMRI3 interrupt request by the capture signal.

The capturing function (stopping the reloading function) of the TMRU-2, in combination with the TMRU-1 and TMRU-3, can also be used for the mode identification purpose.

16.3.3 Reload Counter Timer TMRU-3

The reload counter timer TMRU-3 consists of an 8-bit down-counter and a reloading register.

Its clock source can be selected from between the underflowing signal of the counter and the edges of the DVCTL signals. (When the DVCTL signal is selected as the clock source, reloading will be effected by the underflowing signals and when the dividing clock is selected as the clock source, reloading will be effected by the DVCTL signals.) The reloading signal works to reload the reloading register value into the counter. Also, when a value is written into to the reloading register, the same value will be written into the counter, simultaneously.

The initial values of the counter and the reloading register are H'FF.

The underflowing signals can be used as the capturing signal for the TMRU-2.

The TMRU-3 can also be used as a dividing circuit for the DVCTL. Also, in combination with the TMRU-1 and TMRU-2 (capturing function), the TMRU-3 can be used for the mode identification purpose. Since the divided signals of the DVCTL are being used as the clock source, CTL signals (DVCTL) conforming to the double speed can be input when making

searches. These DVCTL signals can also be used for phase controls of the capstan motor. Also, by selecting the dividing clock as the clock source, it is possible to make a delay with the edges of the DVCTL to provide the slow tracking mono-multi function.

16.3.4 Mode Identification

When making mode identification (2/4/6 identification) of the SP/LP/EP modes of reproducing tapes, the TMRU-1 (CFG dividing circuit), TMRU-2 (capturing function/without reloading function) and TMRU-3 (DVCTL dividing circuit) of the Timer R should be used.

The Timer R will become to the aforementioned status after a reset.

Under the aforementioned status, the divided CFG should be written into the reloading register of the TMRU-1 and divided DVCTL should be written into the reloading register of the TMRU-

3. When the TMRU-3 underflows, the counter value of the TMRU-2 is captured. Such capturing register value represents the number of the CFG within the DVCTL cycle.

As aforementioned, the Timer R can work to count the number of the CFG corresponding to "n" times of DVCTL's or to identify the mode being searched.

For exemplary settings for the register, see section 16.5.1, Mode Identification.

16.3.5 Reeling Controls

CFG counts can be captured by making 16-bit capturing operation combining the TMRU-1 and TMRU-2. By choosing the IRQ3 as the capture signal, and by counting the CFG within the duration of the reel pulse being input through the IRQ3 pin, reeling controls, etc. can be effected.

For exemplary settings for the register, see section 16.5.2, Reeling Controls.

16.3.6 Acceleration and Braking Processes of the Capstan Motor

When making intermittent movements such as those for slow reproductions or for still reproductions, it is necessary to conduct quick accelerations and abrupt stoppings of the capstan motor. The acceleration and braking processes will function to check if the revolution of a capstan motor has reached the prescribed rate when accelerated or braked. For this purpose, the TMRU-2 (reloading function) should be used.

When making accelerations:

- (1) Set the AC/BR bit of the TMRM1 to acceleration. (Set to 1). Also, use the rising edge of the CFG as the reloading signal.
- (2) Set the prescribed time on the CFG frequency to deem as the acceleration has been finished, into the reloading register.
- (3) The TMRU-2 will work to down-count the reloading data.

- (4) In case the acceleration has not been finished (in case the CFG signal is not input even when the prescribed time has elapsed = underflowing of down-counting has occurred), such underflowing works to set to CFG mask F/F (masking movement) and the reload timer will be cleared by the CFG.
- (5) When the acceleration has been finished (when the CFG signal is input before the prescribed time has elapsed = reloading movement has been made before the down counter underflows), an interrupt request will be issued because of the CFG.

When making braking:

- (1) Set the AC/BR bit of the TMRM1 to braking. (Clear to 0). Also, use the rising edge of the CFG as the reloading signal.
- (2) Set the prescribed time on the CFG frequency to deem as the braking has been finished, into the reloading register.
- (3) The TMRU-2 will work to down-count the reloading data.
- (4) In case the braking has not been finished (when the CFG signal is input before the prescribed time has elapsed = reloading movement has been made before the down counter underflows), the reload timer movement will continue.
- (5) When the acceleration has been finished (when the CFG signal is not input even when the prescribed time has elapsed = underflowing of down-counting has occurred), interrupt request will be issued because of the underflowing signal.

The acceleration and braking processes should be employed when making special reproductions, in combination with the slow tracking mono-multi function being outlined below.

For exemplary settings for the register, see section 16.5.4, Acceleration and Braking Processes of the Capstan Motor.

16.3.7 Slow Tracking Mono-multi Function

When performing slow reproductions or still reproductions, the braking timing for the capstan motor is determined by use of the edge of the DVCTL signal. The slow tracking mono-multi function works to measure the time from the rising edge of the DVCTL signal down to the desired point to issue the interrupt request. In actual programming, this interrupt should be used to activate the brake of the capstan motor. The TMRU-3 should be used to perform time measurements for the slow tracking mono-multi function. Also, the braking process can be made using the TMRU-2. Figure 16.2 below shows the exemplary time series movements when a slow reproduction is being performed.

For exemplary settings for the register, see section 16.5.3, Slow Tracking Mono-multi Function.

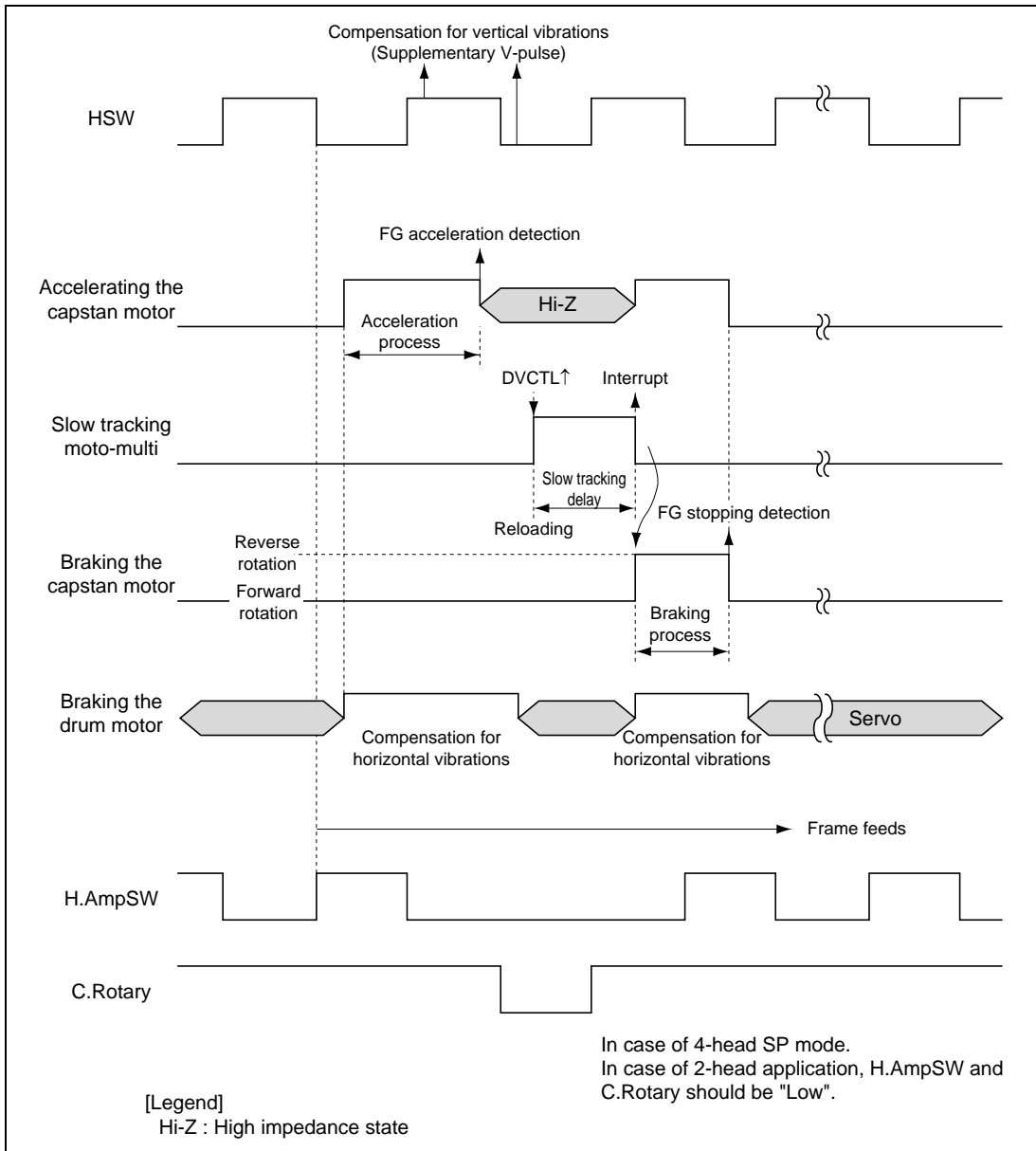


Figure 16.2 Exemplary Time Series Movements when a Slow Reproduction Is Being Performed

16.4 Interrupt Cause

The interrupt causes for the Timer R are 3-causes of the TMRI3 bit through TMRI1 bit of the timer R control/status register (TMRCS).

- (a) Interrupts being caused by the underflowing of the TMRU-1 (TMRI1)

These interrupts will constitute the timing for reloading with the TMRU-1.

- (b) Interrupts being caused by the underflowing of the TMRU-2 or by an end of the acceleration or braking process (TMRI2)

When interrupts occur at the reload timing of the TMRU-2, clear the AC/BR (acceleration/braking) bit of the timer R mode register 1 (TMRM1) to 0.

- (c) Interrupts being caused by the capture signals of the TMRU-3 and by ending the slow tracking mono-multi process (TMRI3)

Since these two interrupt causes are constituting the OR, it becomes necessary to determine which interrupt cause is occurring using the software.

Respective interrupt causes are being set to the CAPF flag or the SLW flag of the timer R mode register 2 (TMRM2), have the software determine which.

Since the CAPF flag and the SLW flag will not be cleared automatically, program the software to clear them. (Writing 0 only is valid for these flags.) Unless these flags are cleared, detection of the next cause becomes unworkable. Also, if the CP/SLM bit is changed leaving these flags un-cleared as they are, these flags will get cleared.

16.5 Exemplary Settings for Respective Functions

16.5.1 Mode Identification

When making mode identification (2/4/6 identification) of the SP/LP/EP modes of reproducing tapes, the TMRU-1 (CFG dividing circuit), TMRU-2 (capturing function/without reloading function) and TMRU-3 (DVCTL dividing circuit) of the Timer R should be used.

The Timer R will become to the aforementioned status after a reset.

Under the aforementioned status, the divided CFG should be written into the reloading register of the TMRU-1 and divided DVCTL should be written into the reloading register of the TMRU-3. When the TMRU-3 underflows, the counter value of the TMRU-2 is captured. Such capturing register value represents the number of the CFG within the DVCTL cycle.

As aforementioned, the Timer R can work to count the number of the CFG corresponding to "n" times of DVCTL's or to identify the mode being searched.

- Exemplary settings

- (1) Setting the timer R mode register 1 (TMRM1)

CLR2 bit (Bit 7) = 1: Works to clear after making the TMRU-2 capture.

RLD bit (Bit 5) = 0: Sets the TMRU-3 without reloading function.

PS21 and PS20 (Bits 3 and 2) = (0 and 0): The underflowing signals of the TMRU-1 are to be used as the clock source for the TMRU-2.

RLD/CAP bit (Bit 1) = 0: The TMRU-1 has been set to make the reload timer operation.

- (2) Setting the timer R mode register 2 (TMRM2)

LAT bit (Bit 7) = 0: The underflowing signals of the TMRU-3 are to be used as the capture signal for the TMRU-2.

PS11 and PS10 (Bits 6 and 5) = (0 and 0): The leading edge of the CFG signal is to be used as the clock source for the TMRU-1.

PS31 and PS30 (Bits 4 and 3) = (0 and 0): The leading edge of the DVCTL signal is to be used as the clock source for the TMRU-3.

CP/SLM bit (Bit 2) = 0: The capture signal is to work to issue the TMRI3 interrupt request.

- (3) Setting the timer R load register 1 (TMRL1)

Set the dividing value for the CFG. The set value should become (n - 1) when divided by "n".

- (4) Setting the timer R load register 3 (TMRL3)

Set the dividing value for the DVCTL. The set value should become (n - 1) when divided by "n".

16.5.2 Reeling Controls

CFG counts can be captured by making 16-bit capturing operation combining the TMRU-1 and TMRU-2. By choosing the IRQ3 as the capture signal, and by counting the CFG within the duration of the reel pulse being input through the $\overline{\text{IRQ3}}$ pin, reeling controls, etc. can be effected.

- Exemplary settings

- (1) Setting P13/ $\overline{\text{IRQ3}}$ pin as the $\overline{\text{IRQ3}}$ pin

- Set the PMR13 bit (Bit 3) of the port mode register 1 (PMR1) to 1. See section 22.2.2, Port Mode Register (PMR1).

- (2) Setting the timer R mode register 1 (TMRM1)

- CLR2 bit (Bit 7) = 1: Works to clear after making the TMRU-2 capture.

- PS21 and PS20 (Bits 3 and 2) = (0 and 0): The underflowing signals of the TMRU-1 are to be used as the clock source for the TMRU-2.

- RLD/CAP bit (Bit 1) = 1: The TMRU-1 has been set to make the capturing operation.

- CPS bit (Bit 0) = 1: The edge of the IRQ3 signal is to be used as the capture signal for the TMRU-1 and TMRU-2.

- (3) Setting the timer R mode register 2 (TMRM2)

- LAT bit (Bit 7) = 1: The edge of the IRQ3 signal is to be used as the capture signal for the TMRU-1 and TMRU-2.

- PS11 and PS10 (Bits 6 and 5) = (0 and 0): The rising edge of the CFG signal is to be used as the clock source for the TMRU-1.

- CP/SLM bit (Bit 2) = 0: The capture signal is to work to issue the TMRI3 interrupt request.

16.5.3 Slow Tracking Mono-multi Function

When performing slow reproductions or still reproductions, the braking timing for the capstan motor is determined by use of the edge of the DVCTL signal. The slow tracking mono-multi function works to measure the time from the leading edge of the DVCTL signal down to the desired point to issue the interrupt request. In actual programming, this interrupt should be used to activate the brake of the capstan motor. The TMRU-3 should be used to perform time measurements for the slow tracking mono-multi function. Also, the braking process can be made using the TMRU-2.

- Exemplary settings

- (1) Setting the timer R mode register 2 (TMRM2)

- PS31 and PS30 (Bits 4 and 3) = Other than (0, 0): The dividing clock is to be used as the clock source for the TMRU-3.

- CP/SLM bit (Bit 2) = 1: The slow tracking delay signal is to work to issue the TMRI3 interrupt request.

(2) Setting the timer R load register 3 (TMRL3)

Set the slow tracking delay value. When the delay count is "n", the set value should be (n - 1).

Regarding the delaying duration, see figure 16.2 Exemplary time series movements when a slow reproduction is being performed.

16.5.4 Acceleration and Braking Processes of the Capstan Motor

When making intermittent movements such as those for slow reproductions or for still reproductions, it is necessary to conduct quick accelerations and abrupt stoppings of the capstan motor. The acceleration and braking processes will function to check if the revolution of a capstan motor has reached the prescribed rate when accelerated or braked. For this purpose, the TMRU-2 (reloading function) should be used.

The acceleration and braking processes should be employed when making special reproductions, in combination with the slow tracking mono-multi function.

- Exemplary settings for the acceleration process

(1) Setting the timer R mode register 1 (TMRM1)

AC/BR bit (Bit 6) = 1: Acceleration process

RLD bit (Bit 5) = 1: The TMRU-2 is to be used as the reload timer.

RLCK bit (Bit 4) = 0: The TMRU-2 is to reload at the rising edge of the CFG.

PS21 and PS20 (Bits 3 and 2) = Other than (0, 0): The dividing clock is to be used as the clock source for the TMRU-2.

(2) Setting the timer R load register 2 (TMRL2)

Set the count reading for the duration until the acceleration process finishes. When the count is "n", the set value should be (n - 1).

Regarding the duration until the acceleration process finishes, see figure 16.2 Exemplary time series movements when a slow reproduction is being performed.

- Exemplary settings for the braking process

(1) Setting the timer R mode register 1 (TMRM1)

AC/BR bit (Bit 6) = 0: Braking process

RLD bit (Bit 5) = 1: The TMRU-2 is to be used as the reload timer.

RLCK bit (Bit 4) = 0: The TMRU-2 is to reload at the rising edge of the CFG.

PS21 and PS20 (Bits 3 and 2) = Other than (0, 0): The dividing clock is to be used as the clock source for the TMRU-2.

(2) Setting the timer R load register 2 (TMRL2)

Set the count reading for the duration until the braking process finishes. When the count is "n", the set value should be (n - 1).

Regarding the duration until the braking process finishes, see figure 16.2 Exemplary time series movements when a slow reproduction is being performed.

Section 17 Timer X1

17.1 Overview

The Timer X1 is capable of outputting two different types of independent waveforms using the free running counter (FRC) as the basic means and it is also applicable to measurements of the durations of input pulses and the cycles external clocks.

17.1.1 Features

Listed below are the features of the Timer X1.

- Choices of 4 different types of counter inputting clocks are available for your selection.
You can select from among three different types of internal clocks ($\phi/4$, $\phi/16$ and $\phi/64$) and the DVCFG.
- Two independent output comparing functions
Capable of outputting two different types of independent waveforms.
- Four independent input capturing functions
The rising edge or falling edge can be selected for use. The buffer operation can also be designated.
- Counter clearing designation is workable.
The counter readings can be cleared by compare match A.
- Seven types of interrupt causes
Comparing match $\times 2$ causes, input capture $\times 4$ causes and overflow $\times 1$ cause are available for use and they can make respective interrupt requests independently.

17.1.2 Block Diagram

Figure 17.1 shows a block diagram of the Timer X1.

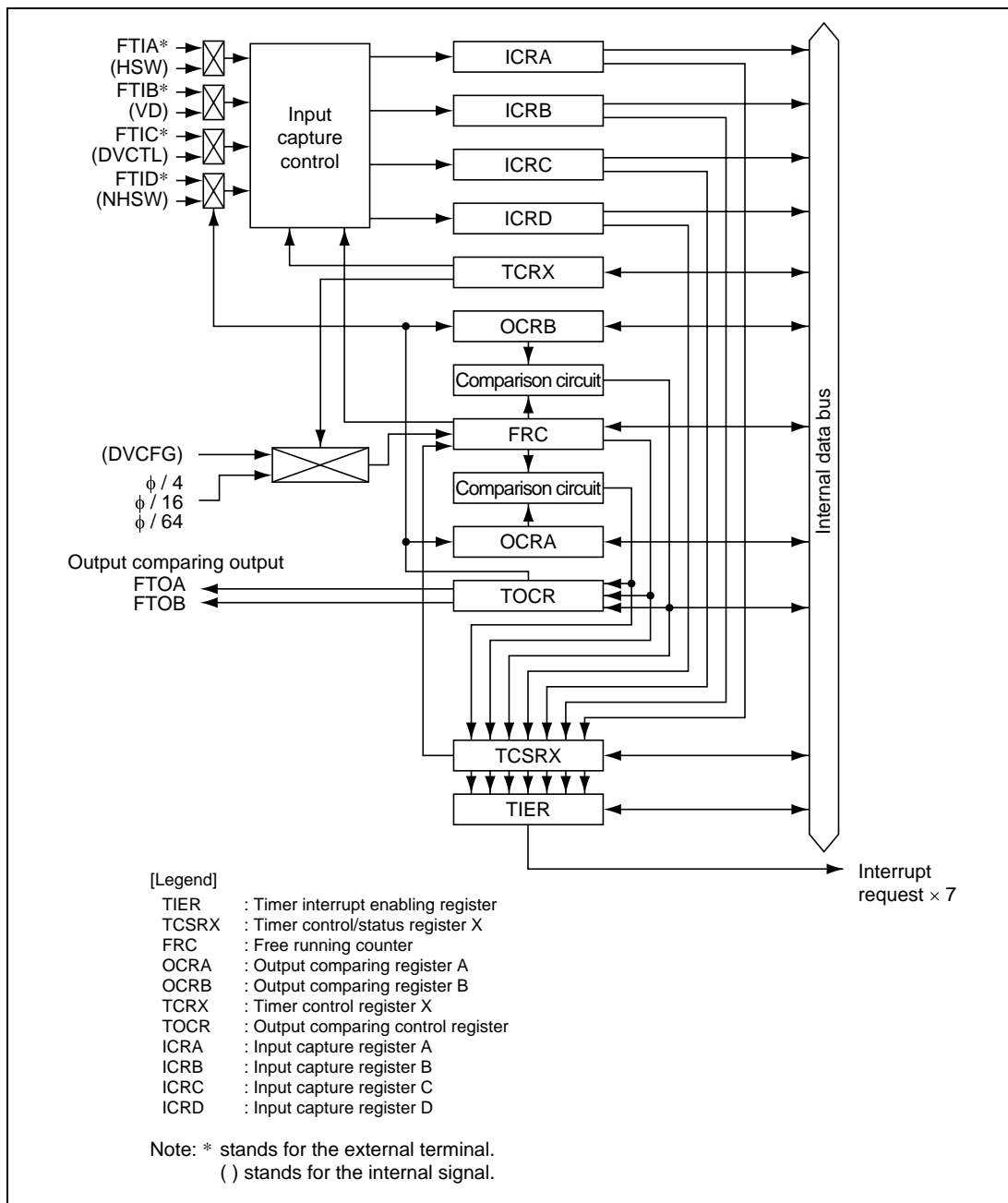


Figure 17.1 Block Diagram of the Timer X1

17.1.3 Pin Configuration

Table 17.1 shows the pin configuration of the Timer X1.

Table 17.1 Pin Configuration

| Name | Abbrev. | I/O | Function |
|-------------------------------|----------------|------------|---------------------------------------|
| Output comparing A output-pin | FTOA | Output | Output pin for the output comparing A |
| Output comparing B output-pin | FTOB | Output | Output pin for the output comparing B |
| Input capture A input-pin | FTIA | Input | Input-pin for the input capture A |
| Input capture B input-pin | FTIB | Input | Input-pin for the input capture B |
| Input capture C input-pin | FTIC | Input | Input-pin for the input capture C |
| Input capture D input-pin | FTID | Input | Input-pin for the input capture D |

17.1.4 Register Configuration

Table 17.2 shows the register configuration of the Timer X1.

Table 17.2 Register Configuration

| Name | Abbrev. | R/W | Initial Value | Address ^{*3} |
|---|---------|----------------------|---------------|-----------------------|
| Timer interrupt enabling register | TIER | R/W | H'00 | H'D100 |
| Timer control/status register X | TCSRX | R/ (W) ^{*1} | H'00 | H'D101 |
| Free running counter H | FRCH | R/W | H'00 | H'D102 |
| Free running counter L | FRCL | R/W | H'00 | H'D103 |
| Output comparing register AH | OCRAH | R/W | H'FF | H'D104 ^{*2} |
| Output comparing register AL | OCRAL | R/W | H'FF | H'D105 ^{*2} |
| Output comparing register BH | OCRBH | R/W | H'FF | H'D104 ^{*2} |
| Output comparing register BL | OCRBL | R/W | H'FF | H'D105 ^{*2} |
| Timer control register X | TCRX | R/W | H'00 | H'D106 |
| Timer output comparing control register | TOCR | R/W | H'00 | H'D107 |
| Input capture register AH | ICRAH | R | H'00 | H'D108 |
| Input capture register AL | ICRAL | R | H'00 | H'D109 |
| Input capture register BH | ICRBH | R | H'00 | H'D10A |
| Input capture register BL | ICRBL | R | H'00 | H'D10B |
| Input capture register CH | ICRCH | R | H'00 | H'D10C |
| Input capture register CL | ICRCL | R | H'00 | H'D10D |
| Input capture register DH | ICRDH | R | H'00 | H'D10E |
| Input capture register DL | ICRDL | R | H'00 | H'D10F |

Notes: 1. Only 0 can be written to clear the flag for Bits 7 to 1. Bit 0 is readable/writable.

2. The addresses of the OCRA and OCRB are the same. Changeover between them are to be made by use of the TOCR bit and OCRS bit.

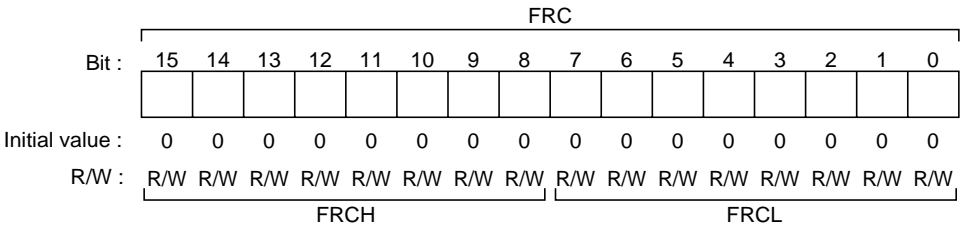
3. Lower 16 bits of the address.

17.2 Descriptions of Respective Registers

17.2.1 Free Running Counter (FRC)

Free running counter H (FRCH)

Free running counter L (FRCL)



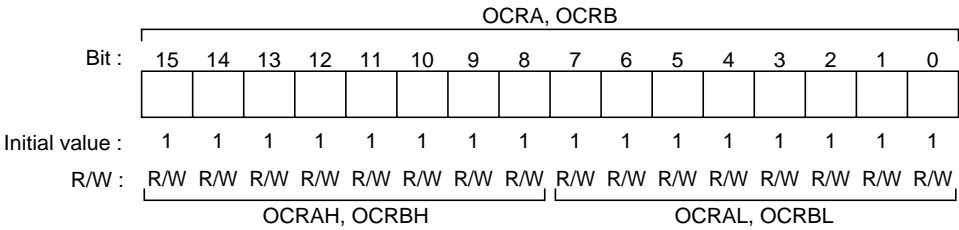
The FRC is a 16-bit read/write up-counter which counts up by the inputting internal clock/external clock. The inputting clock is to be selected from the CKS1 and CKS0 of the TCRX.

By the setting of the CCLRA bit of the TCSRX, the FRC can be cleared by comparing match A. When the FRC overflows (H'FFFF → H'0000), the OVF of the TCSRX will be set to 1. At this time, when the OVIE of the TIER is being set to 1, an interrupt request will be issued to the CPU.

Reading/writing can be made from and to the FRC through the CPU at 8-bit or 16-bit. The FRC is initialized to H'0000 when reset or under the standby mode, watch mode, subsleep mode, module stop mode or subactive mode.

17.2.2 Output Comparing Register A and B (OCRA and OCRB)

Output comparing register AH and BH (OCRAH and OCRBH)
Output comparing register AL and BL (OCRAL and OCRBL)



The OCR consists of twin 8-bit read/write registers (OCRA and OCRB). The contents of the OCR are always being compared with the FRC and, when the value of these two match, the OCFA and OCRB of the TCSR_X will be set to 1. At this time, if the OCIAE and OCIB of the TIER are being set to 1, an interrupt request will be issued to the CPU.

When performing compare matching, if the OEA and OEB of the TOCR are being set to 1, the level value having been set to the OL_VLA and OL_VLB of the TOCR will be output through the FTOA and FTOB pins. After resetting, 0 will be output through the FTOA and FTOB pins until the first compare matching occurs.

Reading/writing can be made from and to the OCR through the CPU at 8-bit or 16-bit.

The OCR is cleared to H'FFFF when reset or under the standby mode, watch mode, subsleep mode, module stop mode or subactive mode.

17.2.3 Input Capture Register A Through D (ICRA Through ICRD)

Input capture register AH to DH (ICRAH to ICRDH)

Input capture register AL to DL (ICRAL to ICRDL)

| ICRA, ICRB, ICRC, ICRD | | | | | | | | | | | | | | | | |
|----------------------------|----|----|----|----|----|----|---|----------------------------|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| ICRAH, ICRBH, ICRCH, ICRDH | | | | | | | | ICRAL, ICRBL, ICRCL, ICRDL | | | | | | | | |

The ICR consists of four 16-bit read only registers (ICRA through ICRD).

When the falling edge of the input capture input signal is detected, the value is transferred to the ICRA through ICRD. At this time, the ICFA through ICFD of the TCSR_X are set to 1 simultaneously. At this time, if the IDIAE through IDIDE of the TCR_X are all being set to 1, due interrupt request will be issued to the CPU. The edge of the input signal can be selected by setting the IEDGA through IEDGD of the TCR_X.

Also, the ICRC and ICRD can be used as the buffer register, respectively, of the ICRA and ICRB by setting the BUFEA and BUFE_B of the TCR_X to perform buffer operations. Figure 17.2 shows the connections necessary when using the ICRC as the buffer register of the ICRA. (BUFEA = 1)

When the ICRC is used as the buffer of the ICRA, by setting IEDGA ≠ IEDGC, both of the rising and falling edges can be designated for use. In case of IEDGA = IEDGC, either one of the rising edge or the falling edge only is usable. Regarding selection of the input signal edge, see table 17.3.

Note: Transference from the FRC to the ICR will be performed regardless of the value of the ICF.

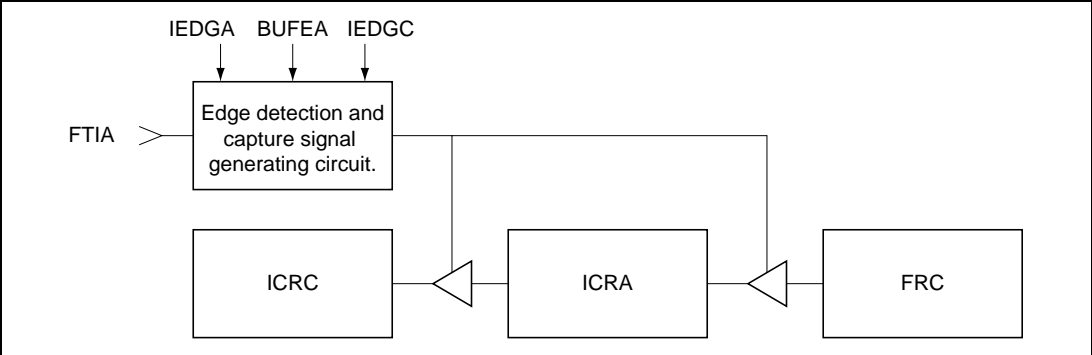


Figure 17.2 Buffer Operation (an example)

Table 17.3 Input Signal Edge Selection when Making Buffer Operation

| IEDGA | IEDGC | Selection of the Input Signal Edge |
|-------|-------|--|
| 0 | 0 | Captures at the rising edge of the input capture input A (Initial value) |
| | 1 | Captures at both rising and falling edges of the input capture input A |
| 1 | 0 | |
| | 1 | Captures at the rising edge of the input capture input A |

Reading can be made from the ICR through the CPU at 8-bit or 16-bit.

For stable input capturing operation, maintain the pulse duration of the input capture input signals at 1.5 system clock (ϕ) or more in case of single edge capturing and at 2.5 system clock (ϕ) or more in case of both edge capturing.

The ICR is initialized to H'0000 when reset or under the standby mode, watch mode, subsleep mode, module stop mode or subactive mode.

17.2.4 Timer Interrupt Enabling Register (TIER)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICIAE | ICIBE | ICICE | ICIDE | OCIAE | OCIBE | OVIE | ICSA |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The TIER is an 8-bit read/write register which works to control permission/prohibition of respective interrupt requests.

The TIER is initialized to H'00 when reset or under the standby mode, watch mode, subsleep mode, module stop mode or subactive mode.

Bit 7: Enabling the Input Capture Interrupt A (ICIAE)

This bit works to permit/prohibit interrupt requests (ICIA) by the ICFA when the ICFA of the TCSR_X is being set to 1.

Bit 7

| ICIAE | Description |
|-------|---|
| 0 | Prohibits interrupt requests (ICIA) by the ICFA (Initial value) |
| 1 | Permits interrupt requests (ICIA) by the ICFA |

Bit 6: Enabling the Input Capture Interrupt B (ICIBE)

This bit works to permit/prohibit interrupt requests (ICIB) by the ICFB when the ICFB of the TCSR_X is being set to 1.

Bit 6

| ICIBE | Description |
|-------|---|
| 0 | Prohibits interrupt requests (ICIB) by the ICFB (Initial value) |
| 1 | Permits interrupt requests (ICIB) by the ICFB |

Bit 5: Enabling the Input Capture Interrupt C (ICICE)

This bit works to permit/prohibit interrupt requests (ICIC) by the ICFC when the ICFC of the TCSR_X is being set to 1.

Bit 5

| ICICE | Description |
|-------|---|
| 0 | Prohibits interrupt requests (ICIC) by the ICFC (Initial value) |
| 1 | Permits interrupt requests (ICIC) by the ICFC |

Bit 4: Enabling the Input Capture Interrupt D (ICIDE)

This bit works to permit/prohibit interrupt requests (ICID) by the ICFD when the ICFD of the TCSR_X is being set to 1.

Bit 4

| ICIDE | Description |
|-------|---|
| 0 | Prohibits interrupt requests (ICID) by the ICFD (Initial value) |
| 1 | Permits interrupt requests (ICID) by the ICFD |

Bit 3: Enabling the Output Comparing Interrupt A (OCIAE)

This bit works to permit/prohibit interrupt requests (OCIA) by the OCFA when the OCFA of the TCSR_X is being set to 1.

Bit 3

| OCIAE | Description |
|-------|---|
| 0 | Prohibits interrupt requests (OCIA) by the OCFA (Initial value) |
| 1 | Permits interrupt requests (OCIA) by the OCFA |

Bit 2: Enabling the Output Comparing Interrupt B (OCIBE)

This bit works to permit/prohibit interrupt requests (OCIB) by the OCFB when the OCFB of the TCSR_X is being set to 1.

Bit 2

| OCIBE | Description |
|-------|---|
| 0 | Prohibits interrupt requests (OCIB) by the OCFB (Initial value) |
| 1 | Permits interrupt requests (OCIB) by the OCFB |

Bit 1: Enabling the Timer Overflow Interrupt (OVIE)

This bit works to permit/prohibit interrupt requests (FOVI) by the OVF when the OVF of the TCSR_X is being set to 1.

Bit 1

| OVIE | Description |
|------|--|
| 0 | Prohibits interrupt requests (FOVI) by the OVF (Initial value) |
| 1 | Permits interrupt requests (FOVI) by the OVF |

Bit 0: Selecting the Input Capture A Signals (ICSA)

This bit works to select the input capture A signals.

Bit 0

| ICSA | Description |
|------|---|
| 0 | Selects the FTIA pin for inputting of the input capture A signals (Initial value) |
| 1 | Selects the HSW for inputting of the input capture A signals |

17.2.5 Timer Control/Status Register X (TCSR_X)

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICFA | ICFB | ICFC | ICFD | OCFA | OCFB | OVF | CCLR_A |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/W |

Note: * Only 0 can be written to clear the flag for Bits 7 to 1.

The TCSR_X is an 8-bit register which works to select counter clearing timing and to control respective interrupt requesting signals. The TCSR_X is initialized to H'00 when reset or under the standby mode, watch mode, subsleep mode, module stop mode or subactive mode. Meanwhile, as for the timing, see section 17.3, Operation.

The FTIA through FTID pins are for fixed inputs inside the LSI under the low power consumption mode excluding the sleep mode. Consequently, when such shifts as "active mode → low power consumption mode → active mode" are made, wrong edges may be detected depending on the pin status or on the type of the detecting edge.

To avoid such error, clear the interrupt requesting flag once immediately after shifting to the active mode from the low power consumption mode.

Bit 7: Input Capture Flag A (ICFA)

This is a status flag indicating the fact that the value of the FRC has been transferred to the ICRA by the input capture signals.

When the BUFEA of the TCR_X is being set to 1, the ICFA indicates the status that the FRC value has been transferred to the ICRA by the input capture signals and that the ICRA value before being updated has been transferred to the ICRC.

This flag should be cleared by use of of the software. Such setting should only be made by use of the hardware. It is not possible to make this setting using a software.

Bit 7

| ICFA | Description |
|------|---|
| 0 | [Clearing conditions] (Initial value) When 0 is written into the ICFA after reading the ICFA under the setting of ICFA = 1 |
| 1 | [Setting conditions] When the value of the FRC has been transferred to the ICRA by the input capture signals |

Bit 6: Input Capture Flag B (ICFB)

This is a status flag indicating the fact that the value of the FRC has been transferred to the ICRB by the input capture signals.

When the BUFEB of the TCRX is being set to 1, the ICFB indicates the status that the FRC value has been transferred to the ICRB by the input capture signals and that the ICRB value before being updated has been transferred to the ICRC.

This flag should be cleared by use of the software. Such setting should only be made by use of the hardware. It is not possible to make this setting using a software.

Bit 6

| ICFB | Description |
|------|---|
| 0 | [Clearing conditions] (Initial value) When 0 is written into the ICFB after reading the ICFB under the setting of ICFB = 1 |
| 1 | [Setting conditions] When the value of the FRC has been transferred to the ICRB by the input capture signals |

Bit 5: Input Capture Flag C (ICFC)

This is a status flag indicating the fact that the value of the FRC has been transferred to the ICRC by the input capture signals.

When an input capture signal occurs while the BUFEA of the TCRX is being set to 1, although the ICFC will be set out, data transference to the ICRC will not be performed.

Therefore, in buffer operation, the ICFC can be used as an external interrupt by setting the ICICE bit to 1.

This flag should be cleared by use of the software. Such setting should only be made by use of the hardware. It is not possible to make this setting using a software.

Bit 5

| ICFC | Description |
|------|---|
| 0 | [Clearing conditions] (Initial value) When 0 is written into the ICFC after reading the ICFC under the setting of ICFC = 1 |
| 1 | [Setting conditions] When the input capture signal has occurred |

Bit 4: Input Capture Flag D (ICFD)

This is a status flag indicating the fact that the value of the FRC has been transferred to the ICRD by the input capture signals.

When an input capture signal occurs while the BUFEB of the TCRX is being set to 1, although the ICFD will be set out, data transference to the ICRD will not be performed.

Therefore, in buffer operation, the ICFD can be used as an external interrupt by setting the ICIDE bit to 1.

This flag should be cleared by use of the software. Such setting should only be made by use of the hardware. It is not possible to make this setting using a software.

Bit 4

| ICFD | Description |
|------|---|
| 0 | [Clearing conditions] (Initial value) When 0 is written into the ICFD after reading the ICFD under the setting of ICFD = 1 |
| 1 | [Setting conditions] When the input capture signal has occurred |

Bit 3: Output Comparing Flag A (OCFA)

This is a status flag indicating the fact that the FRC and the OCRA have come to a comparing match.

This flag should be cleared by use of the software. Such setting should only be made by use of the hardware. It is not possible to make this setting using a software.

Bit 3

| OCFA | Description |
|------|---|
| 0 | [Clearing conditions] (Initial value) When 0 is written into the OCFA after reading the OCFA under the setting of OCFA = 1 |
| 1 | [Setting conditions] When the FRC and the OCRA have come to the comparing match |

Bit 2: Output Comparing Flag B (OCFB)

This is a status flag indicating the fact that the FRC and the OCRB have come to a comparing match.

This flag should be cleared by use of the software. Such setting should only be made by use of the hardware. It is not possible to make this setting using a software.

Bit 2

| OCFB | Description |
|------|---|
| 0 | [Clearing conditions] (Initial value) When 0 is written into the OCFB after reading the OCFB under the setting of OCFB = 1 |
| 1 | [Setting conditions] When the FRC and the OCRB have come to the comparing match |

Bit 1: Time Over Flow (OVF)

This is a status flag indicating the fact that the FRC overflowed. (H'FFFF → H'0000).

This flag should be cleared by use of the software. Such setting should only be made by use of the hardware. It is not possible to make this setting using a software.

Bit 1

| OVF | Description |
|-----|--|
| 0 | [Clearing conditions] (Initial value) When 0 is written into the OVF after reading the OVF under the setting of OVF = 1 |
| 1 | [Setting conditions] When the FRC value has become H'FFFF → H'0000 |

Bit 0: Counter Clearing (CCLRA)

This bit works to select if or not to clear the FRC by occurrence of comparing match A (matching signal of the FRC and OCRA).

Bit 0

| CCLRA | Description |
|-------|--|
| 0 | Prohibits clearing of the FRC by occurrence of comparing match A (Initial value) |
| 1 | Permits clearing of the FRC by occurrence of comparing match A |

17.2.6 Timer Control Register X (TCRX)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | IEDGA | IEDGB | IEDGC | IEDGD | BUFEA | BUFEB | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The TCRX is an 8-bit read/write register which works to select the input capture signal edge, to designate the buffer operation and to select the inputting clock for the FRC.
The TCRX is initialized to H'00 when reset or under the standby mode, watch mode, subsleep mode, module stop mode or subactive mode.

Bit 7: Input Capture Signal Edge Selection A (IEDGA)

This bit works to select the rising edge or falling edge of the input capture signal A (FTIA).

Bit 7

| IEDGA | Description |
|-------|---|
| 0 | Captures the falling edge of the input capture signal A (Initial value) |
| 1 | Captures the rising edge of the input capture signal A |

Bit 6: Input Capture Signal Edge Selection B (IEDGB)

This bit works to select the rising edge or falling edge of the input capture signal B (FTIB).

Bit 6

| IEDGB | Description |
|-------|---|
| 0 | Captures the falling edge of the input capture signal B (Initial value) |
| 1 | Captures the rising edge of the input capture signal B |

Bit 5: Input Capture Signal Edge Selection C (IEDGC)

This bit works to select the rising edge or falling edge of the input capture signal C (FTIC). However, when the DVCTL has been selected as the signal for the input capture signal edge selection C, this bit will not influence the operation.

Bit 5

| IEDGC | Description |
|-------|---|
| 0 | Captures the falling edge of the input capture signal C (Initial value) |
| 1 | Captures the rising edge of the input capture signal C |

Bit 4: Input Capture Signal Edge Selection D (IEDGD)

This bit works to select the rising edge or falling edge of the input capture signal D (FTID).

Bit 4

| IEDGD | Description |
|-------|---|
| 0 | Captures the falling edge of the input capture signal D (Initial value) |
| 1 | Captures the rising edge of the input capture signal D |

Bit 3: Buffer Enabling A (BUFEA)

This bit works to select if or not to use the ICRC as the buffer register for the ICRA.

Bit 3

| BUFEA | Description |
|-------|--|
| 0 | Using the ICRC as the buffer register for the ICRA (Initial value) |
| 1 | Not using the ICRC as the buffer register for the ICRA |

Bit 2: Buffer Enabling B (BUFEB)

This bit works to select if or not to use the ICRD as the buffer register for the ICRB.

Bit 2

| BUFEB | Description |
|-------|--|
| 0 | Using the ICRD as the buffer register for the ICRB (Initial value) |
| 1 | Not using the ICRD as the buffer register for the ICRB |

Bits 1 and 0: Clock Select (CKS1, 0)

These bits work to select the inputting clock to the FRC from among three types of internal clocks and the DVCFG.

The DVCFG is the edge detecting pulse selected by the CFG dividing timer.

| Bit 1 | Bit 0 | Description |
|-------|-------|--|
| CKS1 | CKS0 | |
| 0 | 0 | Internal clock: Counts at $\phi/4$ (Initial value) |
| 0 | 1 | Internal clock: Counts at $\phi/16$ |
| 1 | 0 | Internal clock: Counts at $\phi/64$ |
| 1 | 1 | DVCFG: The edge detecting pulse selected by the CFG dividing timer |

17.2.7 Timer Output Comparing Control Register (TOCR)

| | | | | | | | | |
|-----------------|------|------|------|------|-----|-----|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICSB | ICSC | ICSD | OSRS | OEA | OEB | OLVLA | OLVLB |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The TOCR is an 8-bit read/write register which works to select input capture signals and output comparing output level, to permit output comparing outputs and to control switching over of the access of the OCRA and OCB. See the section 17.2.4 Timer Interrupt Enabling Register (TIER) regarding the input capture inputs A.

The TOCR is initialized to H'00 when reset or under the standby mode, watch mode, subsleep mode, module stop mode or subactive mode.

Bit 7: Selecting the Input Capture B Signals (ICSB)

This bit works to select the input capture B signals.

Bit 7

| ICSB | Description |
|------|---|
| 0 | Selects the FTIB pin for inputting of the input capture B signals (Initial value) |
| 1 | Selects the VD as the input capture B signals |

Bit 6: Selecting the Input Capture C Signals (ICSC)

This bit works to select the input capture C signals. The DVCTL is the edge detecting pulse selected by the CTL dividing timer.

Bit 6

| ICSC | Description |
|------|---|
| 0 | Selects the FTIC pin for inputting of the input capture C signals (Initial value) |
| 1 | Selects the DVCTL as the input capture C signals |

Bit 5: Selecting the Input Capture D Signals (ICSD)

This bit works to select the input capture D signals.

Bit 5

| ICSD | Description |
|------|---|
| 0 | Selects the FTID pin for inputting of the input capture D signals (Initial value) |
| 1 | Selects the NHSW as the input capture D signals |

Bit 4: Selecting the Output Comparing Register (OCRS)

The addresses of the OCRA and OCRB are the same. The OCRS works to control which register to choose when reading/writing this address. The choice will not influence the operation of the OCRA and OCRB.

Bit 4

| OCRS | Description |
|------|---|
| 0 | Selects the OCRA register (Initial value) |
| 1 | Selects the OCRB register |

Bit 3: Enabling the Output A (OEA)

This bit works to control the output comparing A signals.

Bit 3

| OEA | Description |
|-----|---|
| 0 | Prohibits the output comparing A signal outputs (Initial value) |
| 1 | Permits the output comparing A signal outputs |

Bit 2: Enabling the Output B (OEB)

This bit works to control the output comparing B signals.

Bit 2

| OEB | Description |
|-----|---|
| 0 | Prohibits the output comparing B signal outputs (Initial value) |
| 1 | Permits the output comparing B signal outputs |

Bit 1: Output Level A (OLVLA)

This bit works to select the output level to output through the FTOA pin by use of the comparing match A (matching signal between the FRC and OCRA).

Bit 1

| OLVLA | Description |
|-------|---------------------------|
| 0 | Low level (Initial value) |
| 1 | High level |

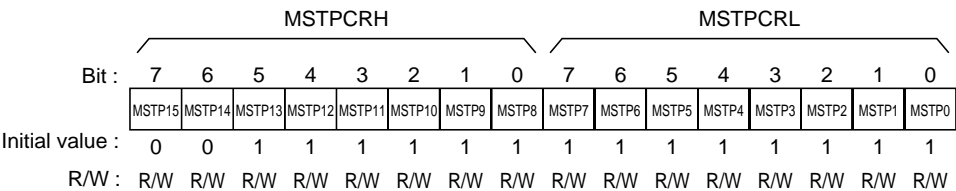
Bit 0: Output Level B (OLVLB)

This bit works to select the output level to output through the FTOB pin by use of the comparing match B (matching signal between the FRC and OCRB).

Bit 0

| OLVLB | Description | |
|--------------|--------------------|-----------------|
| 0 | Low level | (Initial value) |
| 1 | High level | |

17.2.8 Module Stop Control Register (MSTPCR)



The MSTPCR consists of twin 8-bit read/write registers and it works to control the module stop mode.

When the MSTP10 bit is set to 1, the Timer X1 stops its operation at the ending point of the bus cycle to shift to the module stop mode. For more information, see section 4.5, Module Stop Mode.

When reset, the MSTPCR is initialized to H'FFFF.

Bit 2: Module Stop (MSTP10)

This bit works to designate the module stop mode for the Timer X1.

MSTPCRH

Bit 2

| MSTP10 | Description |
|--------|---|
| 0 | Cancels the module stop mode of the Timer X1 |
| 1 | Sets the module stop mode of the Timer X1 (Initial value) |

17.3 Operation

17.3.1 Operation of the Timer X1

(1) Output Comparing Operation

Right after resetting, the FRC is initialized to H'0000 to start counting up. The inputting clock can be selected from among three different types of internal clocks or the external clock by setting the CKS1 and CKS0 of the TCRX.

The contents of the FRC are always being compared with the OCRA and OCRB and, when the value of these two match, the level set by the OLVLA and OLVLB of the TOCR is output through the FTOA pin and FTOB pin.

After resetting, 0 will be output through the FTOA and FTOB pins until the first compare matching occurs.

Also, when the CCLRA of the TCSRX is being set to 1, the FRC will be cleared to H'0000 when the comparing match A occurs.

(2) Input Capturing Operation

Right after resetting, the FRC is initialized to H'0000 to start counting up. The inputting clock can be selected from among three different types of internal clocks or the external clock by setting the CKS1 and CKS0 of the TCRX.

The inputs are transferred to the IEDGA through IEDGD of the TCRX through the FTIA through FTID pins and, at the same time, the ICFA through ICFD of the TCSRX are set to 1. At this time, if the ICIAE through ICIED of the TIER are being set to 1, due interrupt request will be issued to the CPU.

When the BUFEA and BUFEB of the TCRX are set to 1, the ICRC and ICRD work as the buffer register, respectively, of the ICRA and ICRB. When the edge selected by setting the IEDGA through IEDGD of the TCRX is input through the FTIA and FTIB pins, the value at the time of the FRC is transferred to the ICRA and ICRB and, at the same time, the values of the ICRA and ICRB before updating are transferred to the ICRC and ICRD. At this time, when the ICFA and ICFB are being set to 1 and if the ICIAE and ICIBE of the TIER are being set to 1, due interrupt request will be issued to the CPU.

17.3.2 Counting Timing of the FRC

The FRC is counted up by the inputting clock. By setting the CKS1 and CKS0 of the TCRX, the inputting clock can be selected from among three different types of clocks ($\phi/4$, $\phi/16$ and $\phi/64$) and the DVCFG.

(1) In Case of Internal Clock Operation

By setting the CKS1 and CKS0 bits of the TCRX, three types of internal clocks ($\phi/4$, $\phi/16$ and $\phi/64$), generated by dividing the system clock (ϕ) can be selected. Figure 17.3 shows the timing chart at this time.

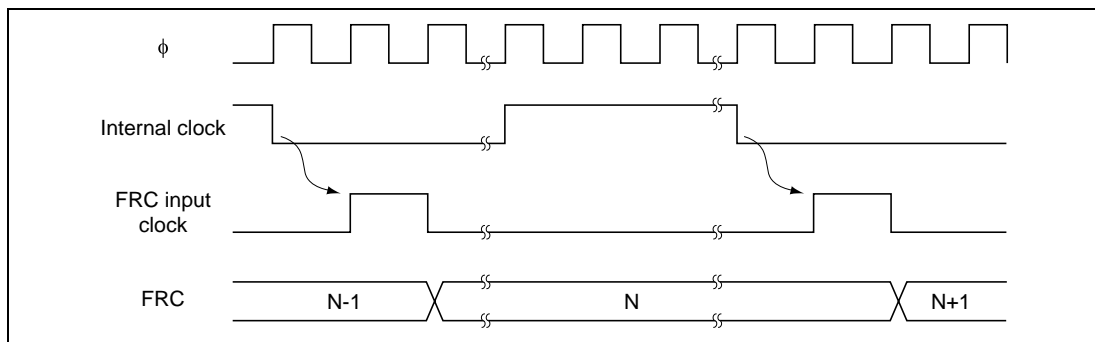


Figure 17.3 Count Timing in Case of Internal Clock Operation

(2) In Case of DVCFG Clock Operation

By setting the CKS1 and CKS0 bits of the TCRX to 1, DVCFG clock input can be selected. The DVCFG clock makes counting by use of the edge detecting pulse being selected by the CFG dividing timer.

Figure 17.4 shows the timing chart at this time.

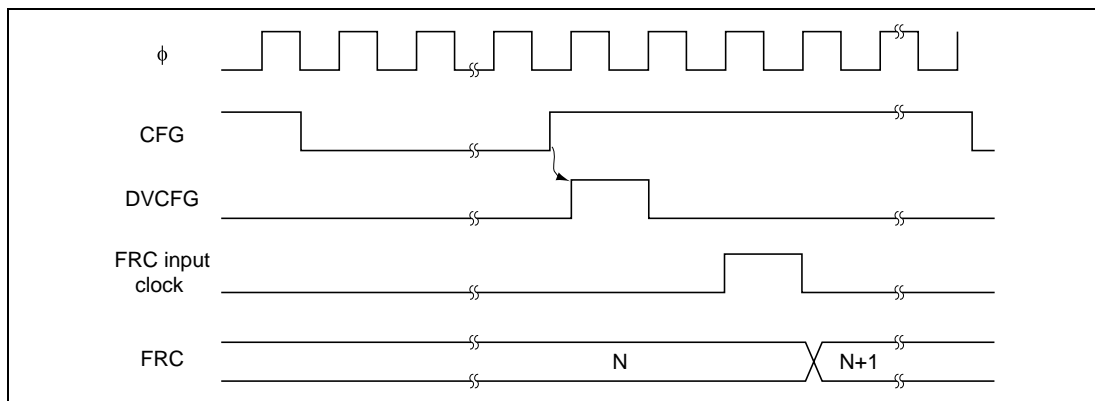


Figure 17.4 Count Timing in Case of CFG Clock Operation

17.3.3 Output Comparing Signal Outputting Timing

When a comparing match occurs, the output level having been set by the OLVL of the TOCR is output through the output comparing signal outputting pins (FTOA and FTOB).
Figure 17.5 shows the timing chart in case of the output comparing signal outputting A.

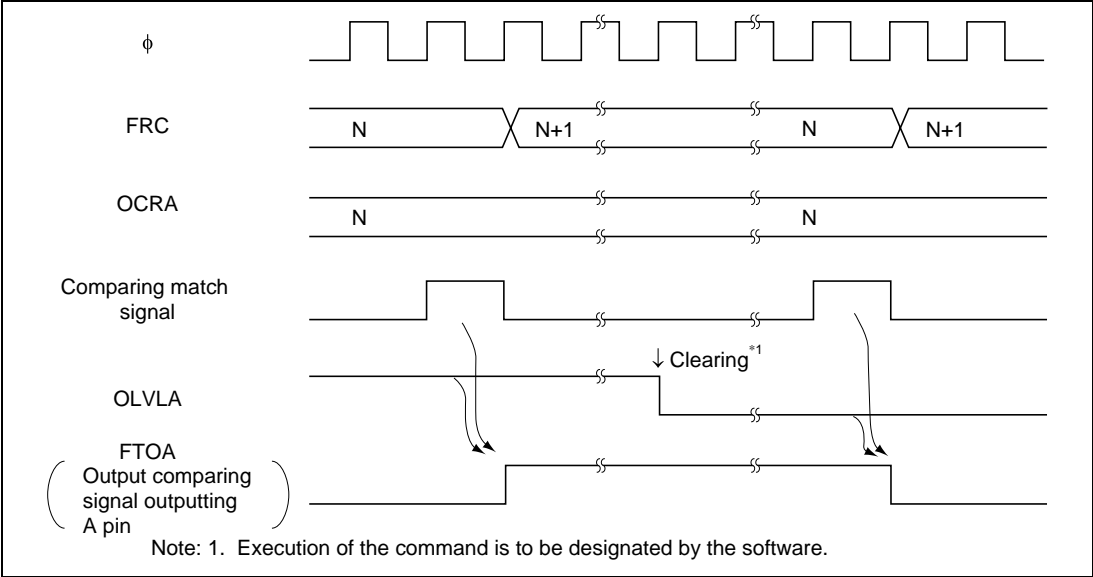


Figure 17.5 Output Comparing Signal Outputting A Timing

17.3.4 FRC Clearing Timing

The FRC can be cleared when the comparing match A occurs. Figure 17.6 shows the timing chart when doing so.

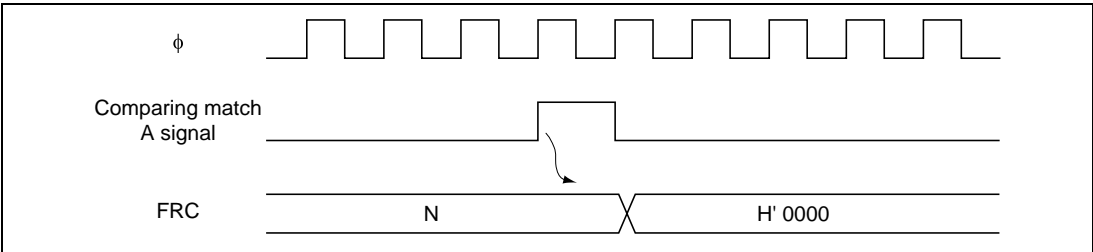


Figure17.6 FRC Clearing Timing by Occurrence of the Comparing Match A

17.3.5 Input Capture Signal Inputting Timing

(1) Input Capture Signal Inputting Timing

As for the input capture signal inputting, rising or falling edge is selected by settings of the IEDGA through IEDGD bits of the TCRX.

Figure 17.7 shows the timing chart when the rising edge is selected (IEDGA through IEDGD = 1).

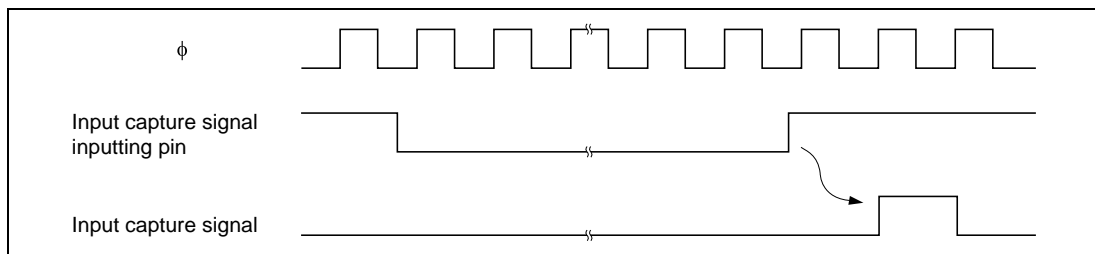


Figure 17.7 Input Capture Signal Inputting Timing (under normal state)

(2) Input Capture Signal Inputting Timing when Making Buffer Operation

Buffer operation can be made using the ICRA or ICRD as the buffer of the ICRA or ICRB.

Figure 17.8 shows the input capture signal inputting timing chart in case both of the rising and falling edges are designated (IEDGA = 1 and IEDGC = 0, or IEDGA = 0 and IEDGC = 1), using the ICRC as the buffer register for the ICRA (BUFEA = 1).

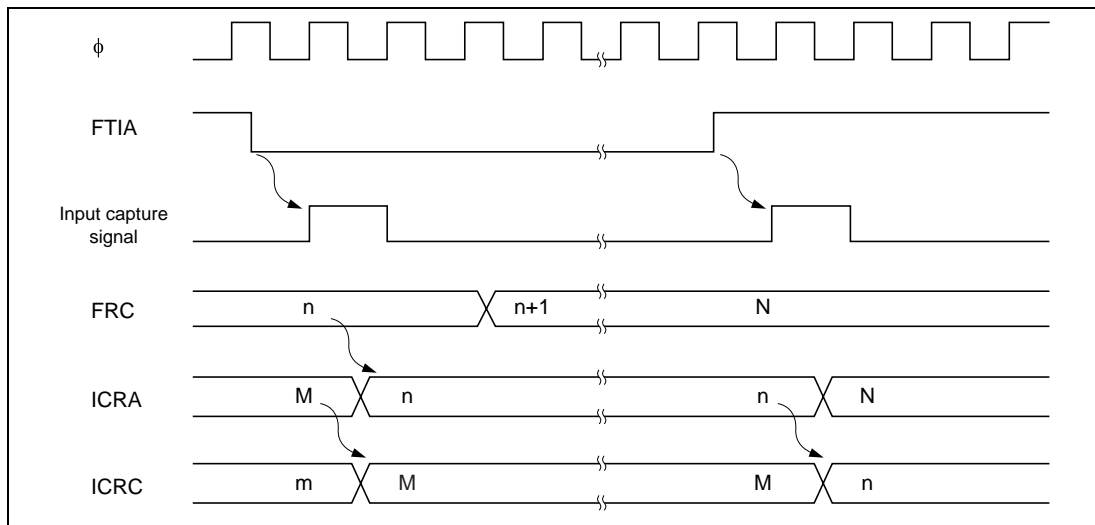


Figure 17.8 Input Capture Signal Inputting Timing Chart Under the Buffer Mode (under normal state)

Even when the ICRC or ICRD is used as the buffer register, the input capture flag will be set up corresponding to the designated edge change of respective input capture signals.

For example, when using the ICRC as the buffer register for the ICRA, when an edge change having been designated by the IEDGC bit is detected with the input capture signals C and if the ICIEC bit is duly set, an interrupt request will be issued.

However, in this case, the FRC value will not be transferred to the ICRC.

17.3.6 Input Capture Flag (ICFA through ICFD) Setting Up Timing

The input capture signal works to set the ICFA through ICFD to "1" and, simultaneously, the FRC value is transferred to the corresponding ICRA through ICRD. Figure 17.9 shows the timing chart for the above.

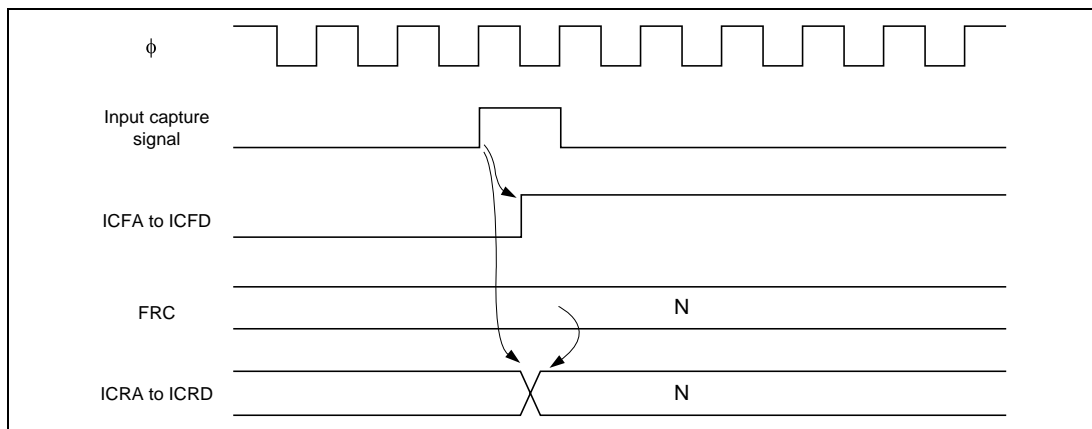


Figure 17.9 ICFA through ICFD Setting Up Timing

17.3.7 **Output Comparing Flag (OCFA and OCFB) Setting Up Timing**

The OCFA and OCFB are being set to 1 by the comparing match signal being output when the values of the OCRA, OCRB and FRC match. The comparing match signal is generated at the last state of the value match (the timing of the FRC's updating the matching count reading). After the values of the OCRA, OCRB and FRC match, up until the count up clock signal is generated, the comparing match signal will not be issued. Figure 17.10 shows the OCFA and OCFB setting timing chart.

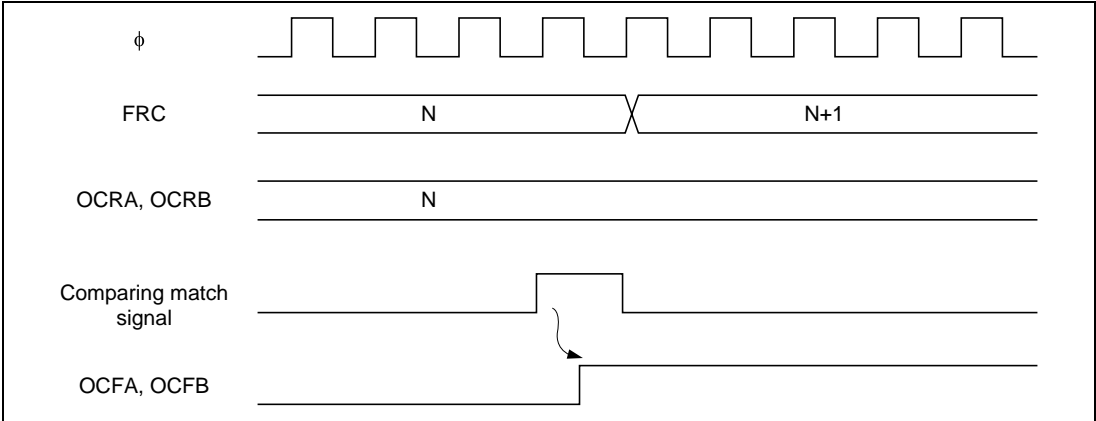


Figure 17.10 OCF Setting Up Timing

17.3.8 **Overflow Flag (CVF) Setting Up Timing**

The OVF is set to when the FRC overflows (H'FFFF → H'0000). Figure 17.11 shows the timing chart for this case.

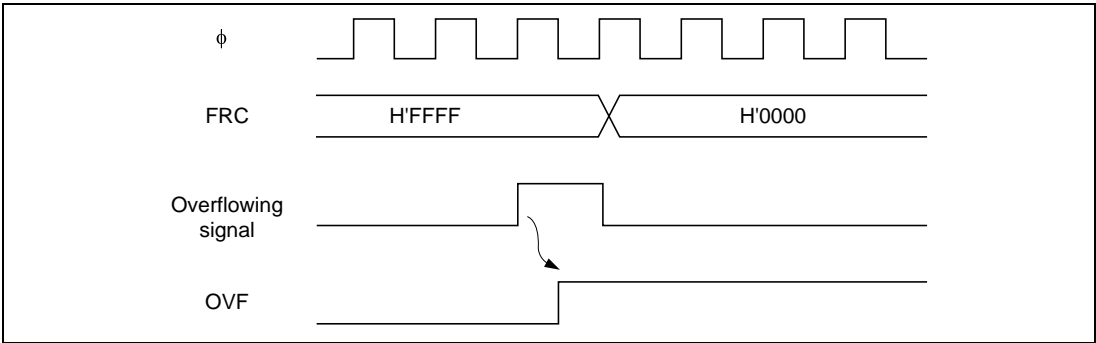


Figure 17.11 OVF Setting Up Timing

17.4 Operation Mode of the Timer X1

Table 17.4 indicated below shows the operation mode of the Timer X1.

Table 17.4 Operation Mode of the Timer X1

| Operation mode | Reset | Active | Sleep | Watch | Subactive | Standby | Subsleep | Module stop |
|-----------------------|--------------|---------------|--------------|--------------|------------------|----------------|-----------------|--------------------|
| FRC | Reset | Functions | Functions | Reset | Reset | Reset | Reset | Reset |
| OCRA, OCRB | Reset | Functions | Functions | Reset | Reset | Reset | Reset | Reset |
| ICRA to ICRD | Reset | Functions | Functions | Reset | Reset | Reset | Reset | Reset |
| TIER | Reset | Functions | Functions | Reset | Reset | Reset | Reset | Reset |
| TCRX | Reset | Functions | Functions | Reset | Reset | Reset | Reset | Reset |
| TOCR | Reset | Functions | Functions | Reset | Reset | Reset | Reset | Reset |
| TCSRX | Reset | Functions | Functions | Reset | Reset | Reset | Reset | Reset |

17.5 Interrupt Causes

Total seven interrupt causes exist with the Timer X1, namely, ICIA through ICID, OCIA, OCIB and FOVI. Table 17.5 given below lists the contents of respective interrupt causes. Respective interrupt requests can be permitted or prohibited by setting of respective interrupt enabling bits of the TIER. Also, independent vector addresses are being allocated to respective interrupt causes.

Table 17.5 Interrupt Causes of the Timer X1

| Abbreviations of the Interrupt Causes | Priority Degree | Contents |
|---------------------------------------|-------------------------------|---|
| ICIA | Interrupt request by the ICFA | <div>High</div> <div>↑</div> <div>Low</div> |
| ICIB | Interrupt request by the ICFB | |
| ICIC | Interrupt request by the ICFC | |
| ICID | Interrupt request by the ICFD | |
| OCIA | Interrupt request by the OCFA | |
| OCIB | Interrupt request by the OCFB | |
| FOVI | Interrupt request by the OVF | |

17.6 Exemplary Uses of the Timer X1

Figure 17.12 indicated below shows an example of outputting at optional phase difference of the pulses of the 50% duty. For this setting, follow the procedures listed below.

- (1) set the CCLRA bit of the TCSR_X to "1".
- (2) Each time a comparing match occurs, the OL_{VIA} bit and the OL_{VLB} bit are reversed by use of the software.

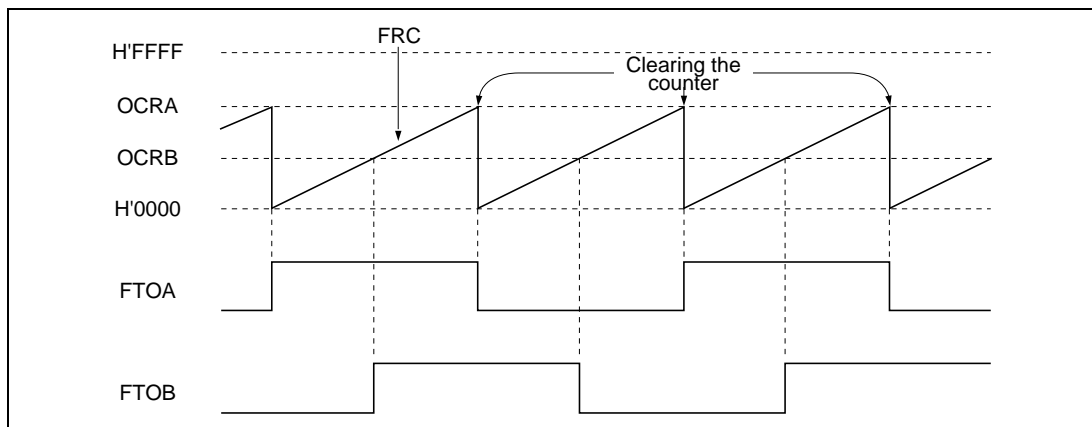


Figure 17.12 An Exemplary Pulse Outputting

17.7 Precautions when Using the Timer X1

Pay great attention to the fact that the following competitions and operations occur during operation of the Timer X1.

17.7.1 Competition between Writing and Clearing with the FRC

When a counter clearing signal is issued under the T2 state where the FRC is under the writing cycle, writing into the FRC will not be effected and the priority will be given to clearing of the FRC.

Figure 17.13 shows the timing chart in this case.

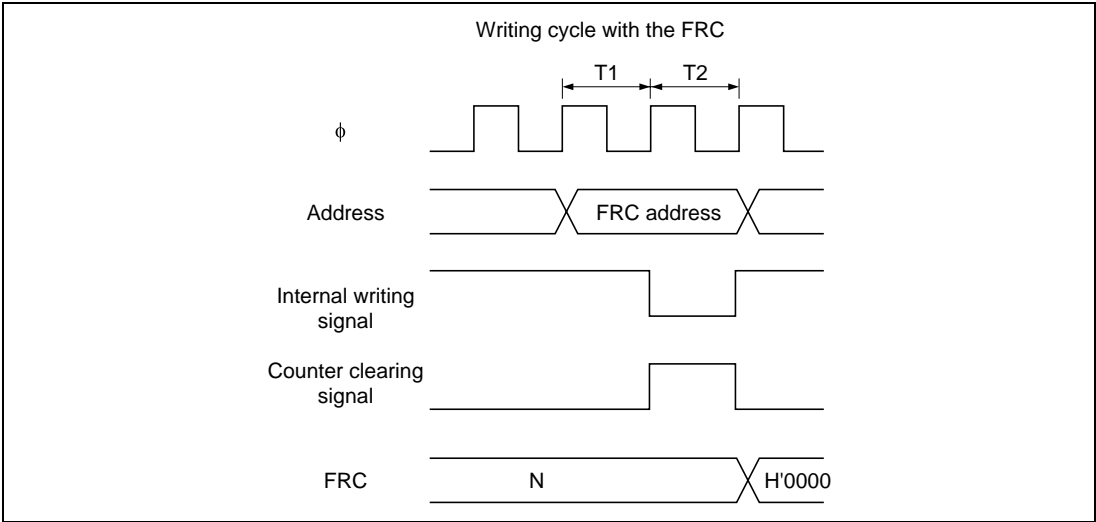


Figure 17.13 Competition between Writing and Clearing with the FRC

17.7.2 Competition between Writing and Counting Up with the FRC

When a counting up cause occurs under the T2 state where the FRC is under the writing cycle, the counting up will not be effected and the priority will be given to count writing.

Figure 17.14 shows the timing chart in this case.

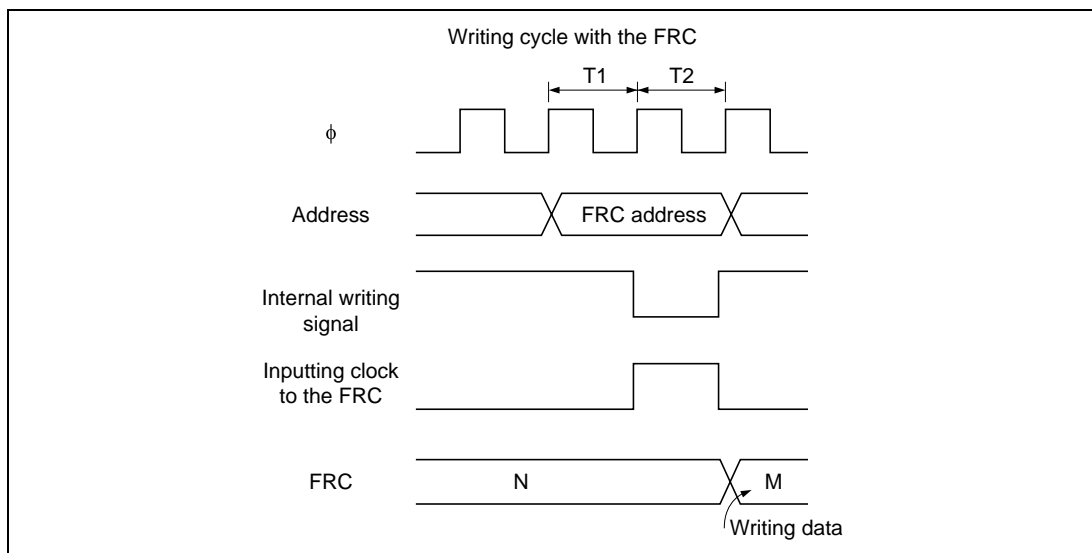


Figure 17.14 Competition between Writing and Counting Up with the FRC

17.7.3 Competition between Writing and Comparing Match with the OCR

When a comparing match occurs under the T2 state where the OCRA and OCRB are under the writing cycle, the priority will be given to writing of the OCR and the comparing match signal will be prohibited.

Figure 17.15 shows the timing chart in this case.

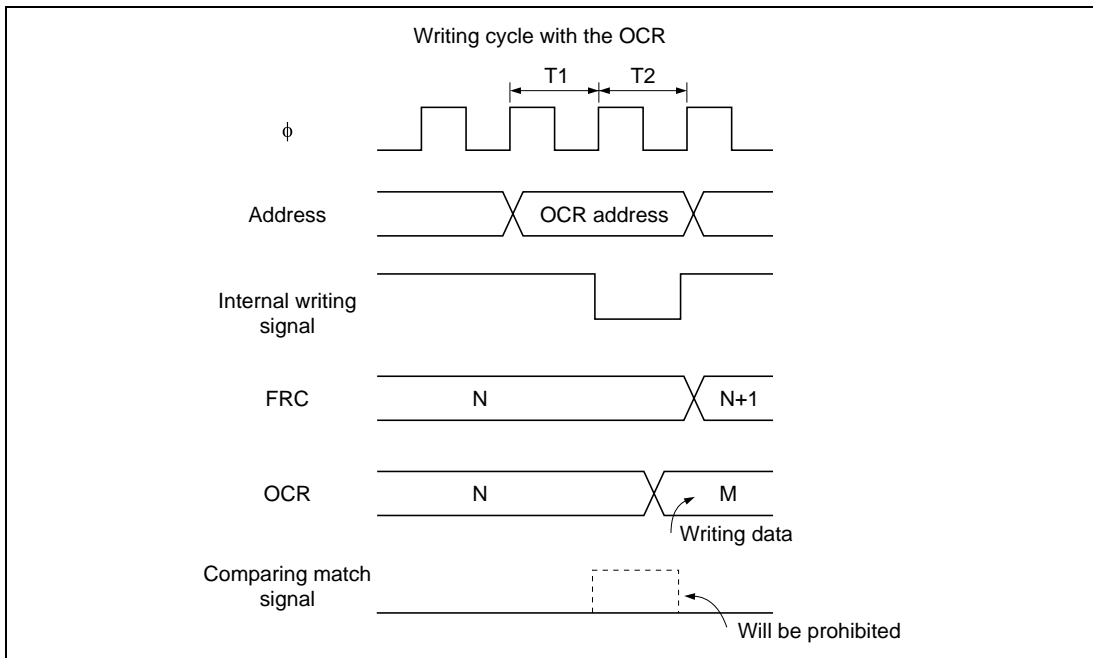


Figure 17.15 Competition between Writing and Comparing Match with the OCR

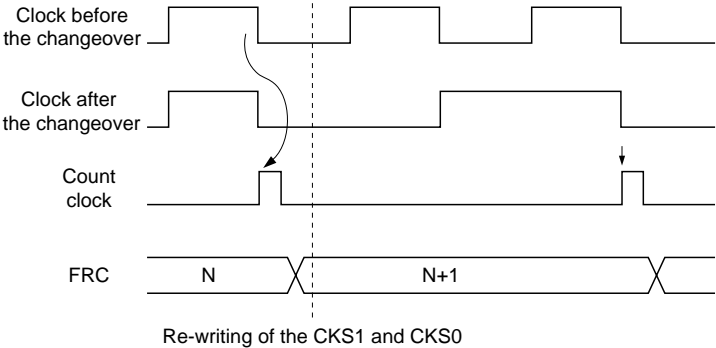
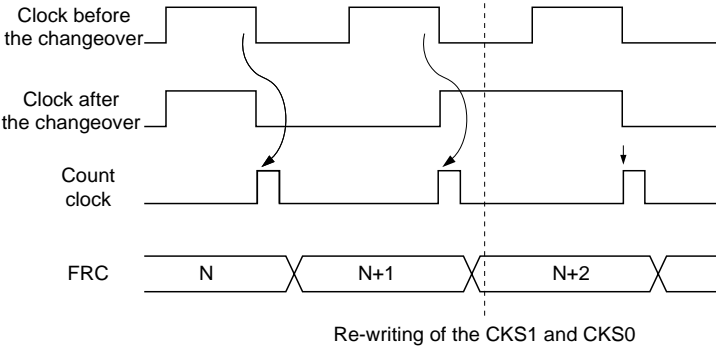
17.7.4 Changing Over the Internal Clocks and Counter Operations

Depending on the timing of changing over the internal clocks, the FRC may count up. Table 17.6 indicated below shows the relations between the timing of changing over the internal clocks (Re-writing of the CKS1 and CKS0) and the FRC operations.

When using an internal clock, the counting clock is being generated detecting the falling edge of the internal clock dividing the system clock (ϕ). For this reason, like Item No. 3 of table 17.6, count clock signals are issued deeming the timing before the changeover as the falling edge to have the FRC to count up.

Also, when changing over between an internal clock and the external clock, the FRC may count up.

Table 17.6 Changing Over the Internal Clocks and the FRC Operation

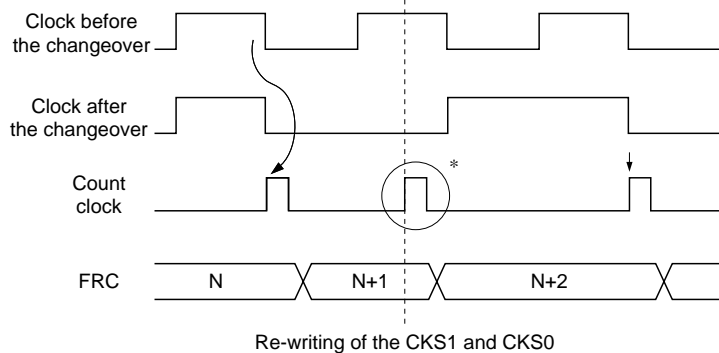
| No. | Re-writing timing for the CKS1 and CKS0 | FRC operation |
|-----|---|---|
| 1 | Low → Low level changeover |  <p>Re-writing of the CKS1 and CKS0</p> |
| 2 | Low → High level changeover |  <p>Re-writing of the CKS1 and CKS0</p> |

Re-writing timing for the CKS1 and CKS0

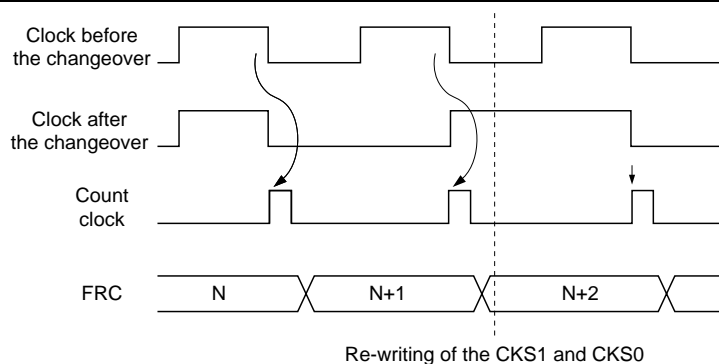
No.

FRC operation

3 High → Low level
changeover



4 High → High level
changeover



Note: * The count clock signals are issued deeming the changeover timing as the falling edge to have the FRC to count up.

Section 18 Watchdog Timer (WDT)

18.1 Overview

This LSI has an on-chip watchdog timer with one channel (WDT) for monitoring system operation. The WDT outputs an overflow signal if a system crash prevents the CPU from writing to the timer counter, allowing it to overflow. At the same time, the WDT can also generate an internal reset signal or internal NMI interrupt signal.

When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer mode, an interval timer interrupt is generated each time the counter overflows.

18.1.1 Features

WDT features are listed below.

- Switchable between watchdog timer mode and interval timer mode
 - WOVI interrupt generation in interval timer mode
- Internal reset or internal interrupt generated when the timer counter overflows
 - Choice of internal reset or NMI interrupt generation in watchdog timer mode
- Choice of 8 counter input clocks
 - Maximum WDT interval: system clock period $\times 131072 \times 256$

18.1.2 Block Diagram

Figure 18.1 shows block diagram of WDT.

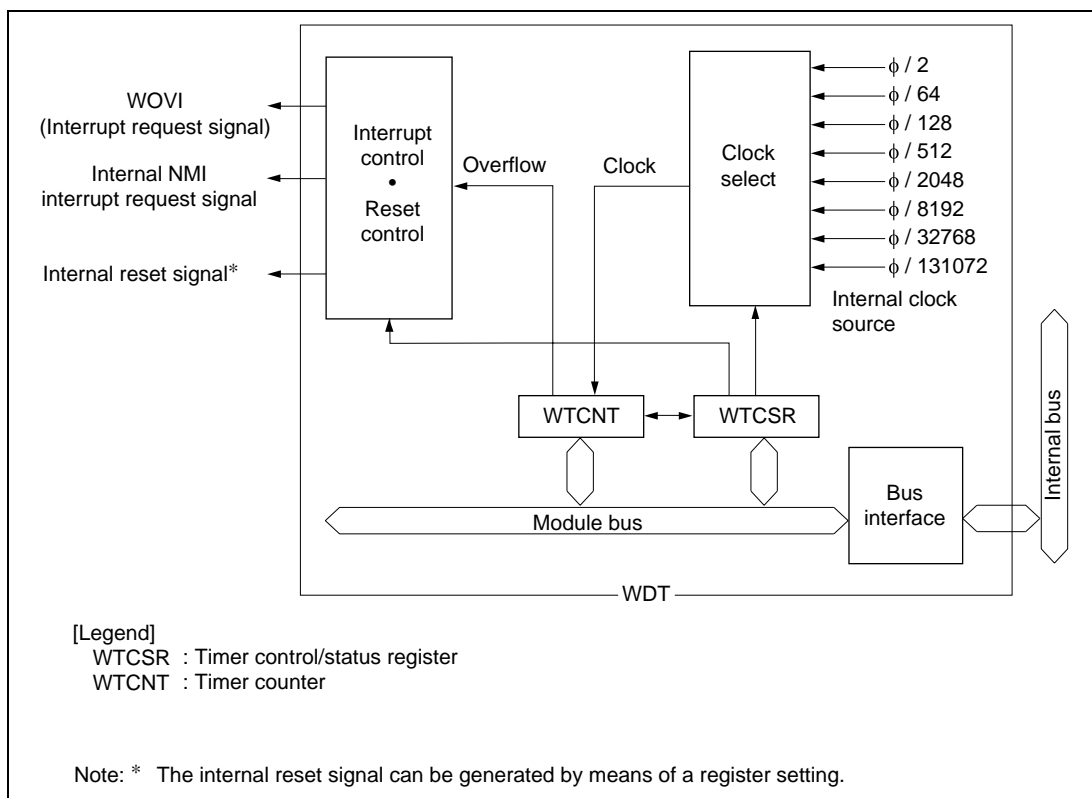


Figure 18.1 Block Diagram of WDT

18.1.3 Register Configuration

The WDT has two registers, as summarized in table 18.2. These registers control clock selection, WDT mode switching, the reset signal, etc.

Table 18.2 WDT Registers

| Name | Abbrev. | R/W | Initial Value | Address ^{*1} | |
|--|---------|----------------------|---------------|-----------------------|--------|
| | | | | Write ^{*2} | Read |
| Watchdog timer control/status register | WTCSR | R/ (W) ^{*3} | H'00 | H'FFBC | H'FFBC |
| Watchdog timer counter | WTCNT | R/W | H'00 | H'FFBC | H'FFBD |
| System control register | SYSCR | R/W | H'09 | H'FFE8 | H'FFE8 |

Notes: 1. Lower 16 bits of the address.

2. For details of write operations, see section 18.2.4, Notes on Register Access.

3. Only 0 can be written in bit 7, to clear the flag.

18.2 Register Descriptions

18.2.1 Watchdog Timer Counter (WTCNT)

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

TCNT is an 8-bit readable/writable* up-counter.

When the TME bit is set to 1 in WTCSR, WTCNT starts counting pulses generated from the internal clock source selected by bits CKS2 to CKS0 in WTCSR. When the count overflows (changes from H'FF to H'00), the OVF flag in WTCSR is set to 1.

WTCNT is initialized to H'00 by a reset, or when the TME bit is cleared to 0.

Note: * WTCNT is write-protected by a password to prevent accidental overwriting. For details see section 18.2.4, Notes on Register Access.

18.2.2 Watchdog Timer Control/Status Register (WTCSR)

| | | | | | | | | |
|-----------------|--------|-------|-----|------|---------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | OVF | WT/IT | TME | RSTS | RST/NMI | CKS2 | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Only 0 can be written to clear the flag.

WTCSR is an 8-bit readable/writable* register. Its functions include selecting the clock source to be input to WTCNT, and the timer mode.

WTCSR is initialized to H'00 by a reset.

Note: * WTCSR is write-protected by a password to prevent accidental overwriting. For details see section 18.2.4, Notes on Register Access.

Bit 7: Overflow Flag (OVF)

A status flag that indicates that WTCNT has overflowed from H'FF to H'00.

Bit 7

| OVF | Description |
|------------|--|
| 0 | [Clearing conditions] (1) Write 0 in the TME bit (2) Read WTCNT when OVF = 1, then write 0 in OVF (Initial value) |
| 1 | [Setting condition] When WTCNT overflows (changes from H'FF to H'00) When internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset |

Bit 6: Timer Mode Select (WT/ $\overline{\text{IT}}$)

Selects whether the WDT is used as a watchdog timer or interval timer. If used as an interval timer, the WDT generates an interval timer interrupt request (WOVI) when TCNT overflows. If used as a watchdog timer, the WDT generates a reset or NMI interrupt when TCNT overflows.

Bit 6

| WT/$\overline{\text{IT}}$ | Description |
|---|---|
| 0 | Interval timer mode: Sends the CPU an interval timer interrupt request (WOVI) when WTCNT overflows (Initial value) |
| 1 | Watchdog timer mode: Sends the CPU a reset or NMI interrupt request when WTCNT overflows |

Bit 5: Timer Enable (TME)

Selects whether WTCNT runs or is halted.

Bit 5

| TME | Description |
|------------|--|
| 0 | WTCNT is initialized to H'00 and halted (Initial value) |
| 1 | WTCNT counts |

Bit 4: Reset Select (RSTS)

Reserved. This bit should not be set to 1.

Bit 3: Reset or NMI (RST/ $\overline{\text{NMI}}$)

Specifies whether an internal reset or NMI interrupt is requested on WTCNT overflow in watchdog timer mode.

Bit 3

| RST/$\overline{\text{NMI}}$ | Description |
|---|---|
| 0 | An NMI interrupt request is generated (Initial value) |
| 1 | An internal reset request is generated |

Bits 2 to 0: Clock Select 2 to 0 (CKS2 to CKS0)

These bits select an internal clock source, obtained by dividing the system clock (ϕ) for input to WTCNT.

■ WDT input clock selection

| Bit 2 | Bit 1 | Bit 0 | Description |
|--------------|--------------|--------------|--|
| CSK2 | CSK1 | CSK0 | Clock Overflow Period* (when $\phi = 10 \text{ MHz}$) |
| 0 | 0 | 0 | $\phi/2$ (Initial value) 51.2 μs |
| | | 1 | $\phi/64$ 1.6 ms |
| | 1 | 0 | $\phi/128$ 3.3 ms |
| | | 1 | $\phi/512$ 13.1 ms |
| 1 | 0 | 0 | $\phi/2048$ 52.4 ms |
| | | 1 | $\phi/8192$ 209.7 ms |
| | 1 | 0 | $\phi/32768$ 838.9 ms |
| | | 1 | $\phi/131072$ 3.36 s |

Note: * The overflow period is the time from when WTCNT starts counting up from H'00 until overflow occurs.

18.2.3 System Control Register (SYSCR)

| | | | | | | | | |
|-----------------|---|---|-------|-------|------|--------|--------|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | INTM1 | INTM0 | XRST | NMIEG1 | NMIEG0 | — |
| Initial value : | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| R/W : | — | — | R | R/W | R | R/W | R/W | — |

Only bit 3 is described here. For details on functions not related to the watchdog timer, see sections 3.2.2 and 6.2.1, System Control Register (SYSCR), and the descriptions of the relevant modules.

Bit 3: External Reset (XRST)

Indicates the reset source. When the watchdog timer is used, a reset can be generated by watchdog timer overflow in addition to external reset input. XRST is a read-only bit. It is set to 1 by an external reset, and cleared to 0 by watchdog timer overflow.

Bit 3

| XRST | Description |
|------|--|
| 0 | Reset is generated by watchdog timer overflow |
| 1 | Reset is generated by external reset input (Initial value) |

18.2.4 Notes on Register Access

The watchdog timer's WTCNT and WTCSR registers differ from other registers in being more difficult to write to. The procedures for writing to and reading these registers are given below.

(1) Writing to WTCNT and WTCSR

These registers must be written to by a word transfer instruction. They cannot be written to with byte transfer instructions.

Figure 18.2 shows the format of data written to WTCNT and WTCSR. WTCNT and WTCSR both have the same write address. For a write to WTCNT, the upper byte of the written word must contain H'5A and the lower byte must contain the write data. For a write to WTCSR, the upper byte of the written word must contain H'A5 and the lower byte must contain the write data. This transfers the write data from the lower byte to WTCNT or WTCSR.

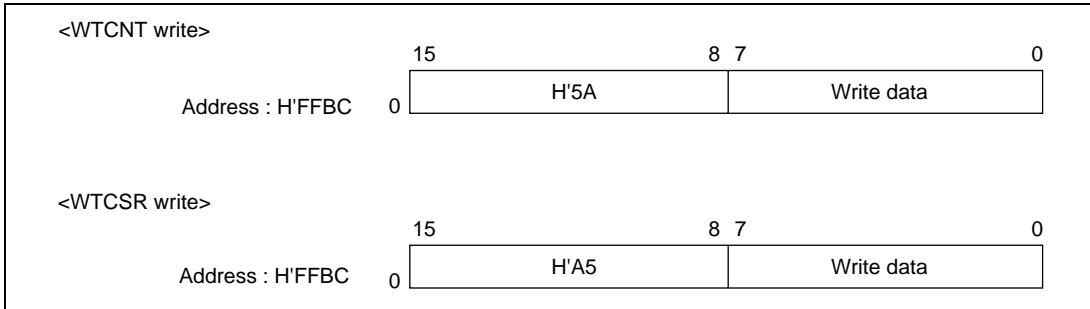


Figure 18.2 Format of Data Written to WTCNT and WTCSR

(2) Reading WTCNT and WTCSR

These registers are read in the same way as other registers. The read addresses are H'FFBC for WTCSR, and H'FFBD for WTCNT.

18.3 Operation

18.3.1 Watchdog Timer Operation

To use the WDT as a watchdog timer, set the $\overline{\text{WT/IT}}$ and TME bits in WTCSR to 1. Software must prevent WTCNT overflows by rewriting the WTCNT value (normally by writing H'00) before overflow occurs. This ensures that WTCNT does not overflow while the system is operating normally. If WTCNT overflows without being rewritten because of a system crash or other error, the chip is reset, or an NMI interrupt is generated, for 518 system clock periods (518 ϕ). This is illustrated in figure 18.3.

An internal reset request from the watchdog timer and reset input from the $\overline{\text{RES}}$ pin are handled via the same vector. The reset source can be identified from the value of the XRST bit in SYSCR.

If a reset caused by an input signal from the $\overline{\text{RES}}$ pin and a reset caused by WDT overflow occur simultaneously, the $\overline{\text{RES}}$ pin reset has priority, and the XRST bit in SYSCR is set to 1.

An NMI interrupt request from the watchdog timer and an interrupt request from the NMI pin are handled via the same vector. Simultaneous handling of a watchdog timer NMI interrupt request and an NMI pin interrupt request must therefore be avoided.

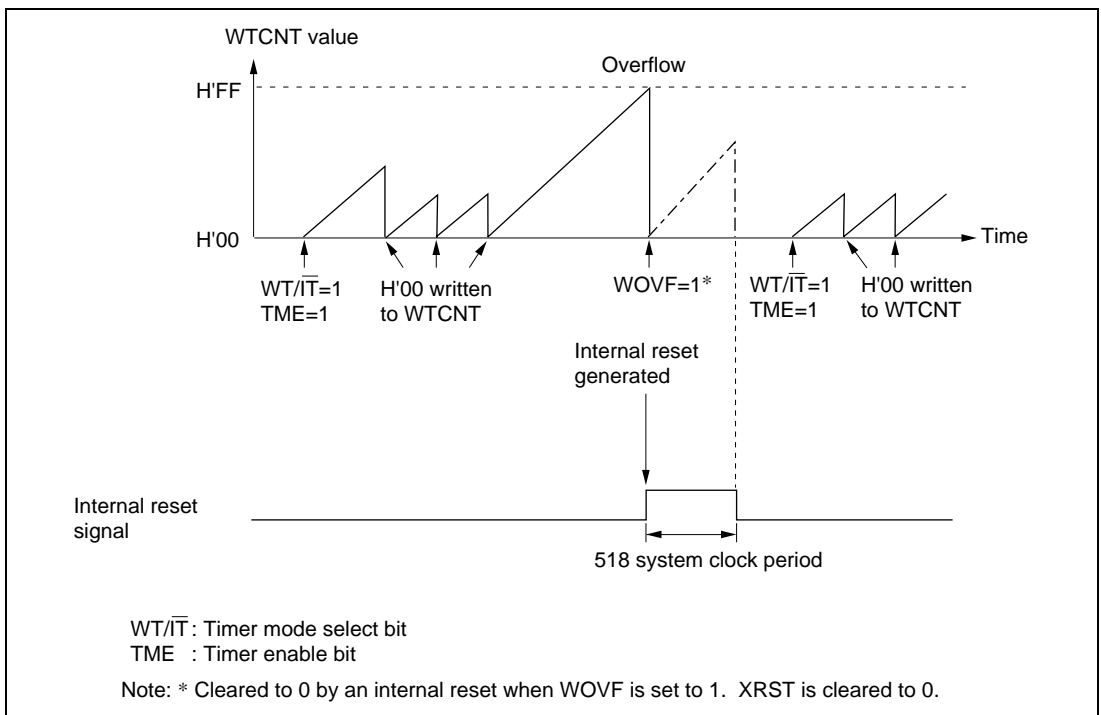


Figure 18.3 Operation in Watchdog Timer Mode

18.3.2 Interval Timer Operation

To use the WDT as an interval timer, clear the WT/\overline{IT} bit in WTCSR to 0 and set the TME bit to 1. An interval timer interrupt (WOVI) is generated each time WTCNT overflows, provided that the WDT is operating as an interval timer, as shown in figure 18.4. This function can be used to generate interrupt requests at regular intervals.

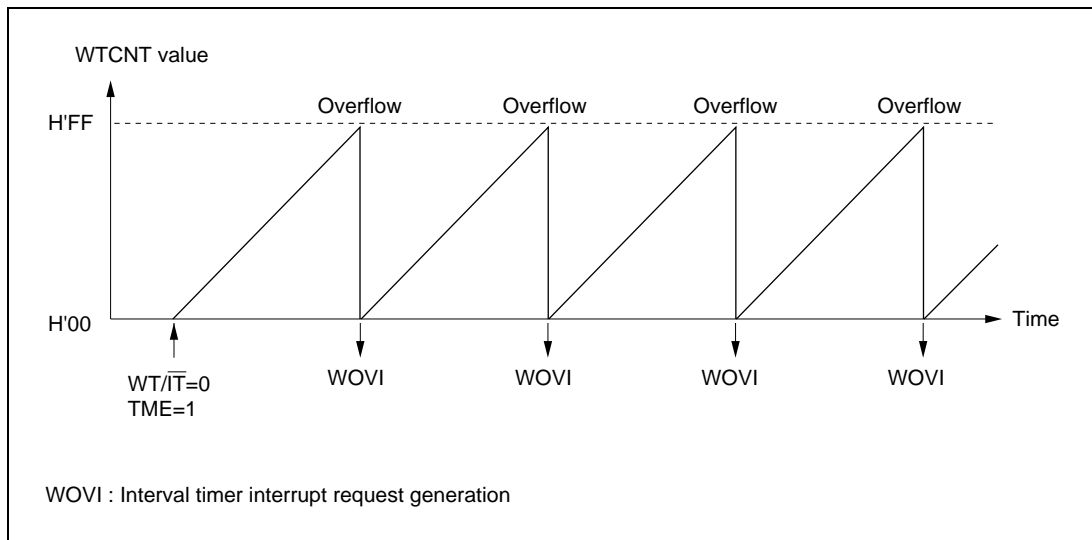


Figure 18.4 Operation in Interval Timer Mode

18.3.3 Timing of Setting of Overflow Flag (OVF)

The OVF bit in WTCSR is set to 1 if WTCNT overflows during interval timer operation. At the same time, an interval timer interrupt (WOVI) is requested. This timing is shown in figure 18.5. If NMI request generation is selected in watchdog timer mode, when WTCNT overflows the OVF bit in WTCSR is set to 1 and at the same time an NMI interrupt is requested.

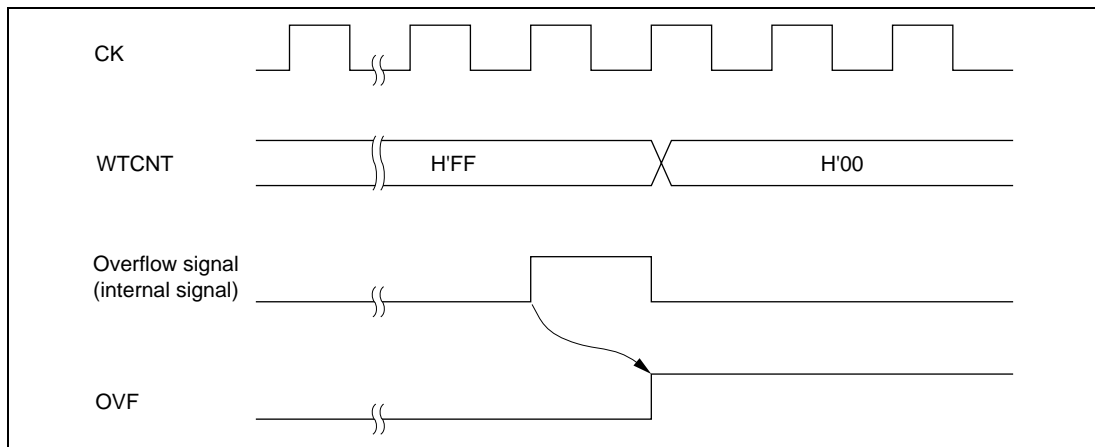


Figure 18.5 Timing of OVF Setting

18.4 Interrupts

During interval timer mode operation, an overflow generates an interval timer interrupt (WOVI). The interval timer interrupt is requested whenever the OVF flag is set to 1 in WTCSR. OVF must be cleared to 0 in the interrupt handling routine. When NMI interrupt request generation is selected in watchdog timer mode, an overflow generates an NMI interrupt request.

18.5 Usage Notes

18.5.1 Contention between Watchdog Timer Counter (WTCNT) Write and Increment

If a timer counter clock pulse is generated during the T2 state of a WTCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 18.6 shows this operation.

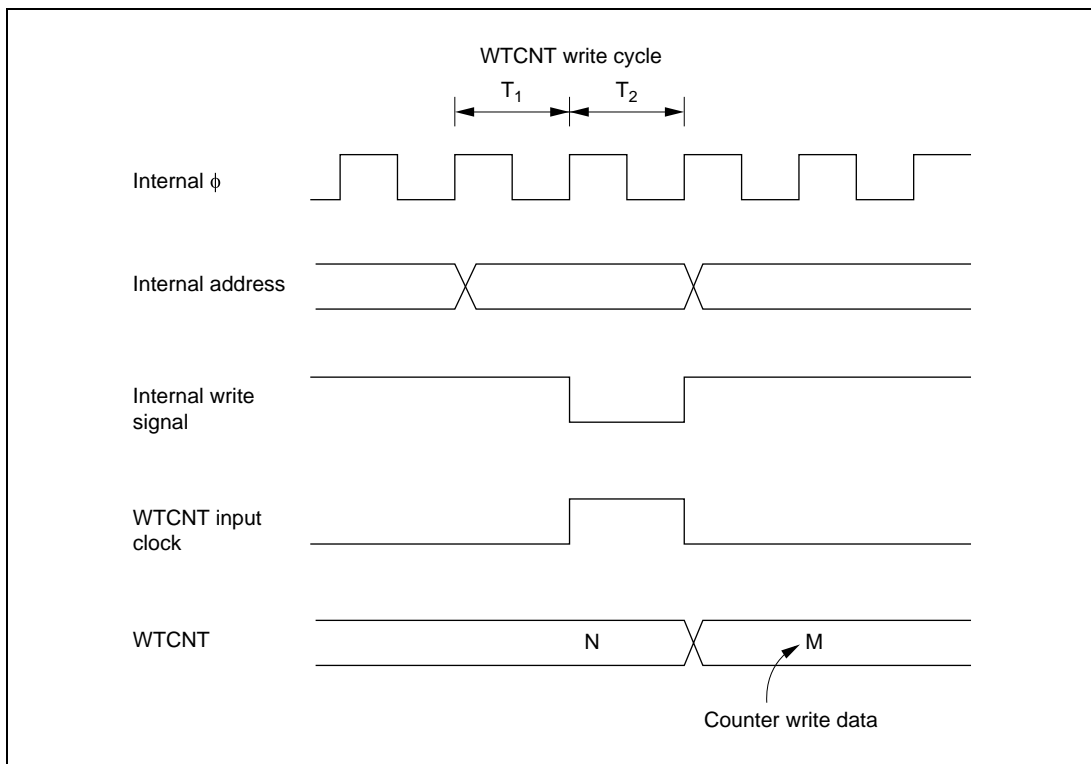


Figure 18.6 Contention between WTCNT Write and Increment

18.5.2 Changing Value of CKS2 to CKS0

If bits CKS2 to CKS0 in WTC SR are written to while the WDT is operating, errors could occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before changing the value of bits CKS2 to CKS0.

18.5.3 Switching between Watchdog Timer Mode and Interval Timer Mode

If the mode is switched from watchdog timer to interval timer, or vice versa, while the WDT is operating, errors could occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before switching the mode.

Section 19 8-Bit PWM

19.1 Overview

The 8-bit PWM incorporates 4 channels of the duty control method. Its outputs can be used to control a reel motor or loading motor.

19.1.1 Features

- Conversion period: 256-state
- Duty control method

19.1.2 Block Diagram

Figure 19.1 shows a block diagram of the 8-bit PWM (1 channel).

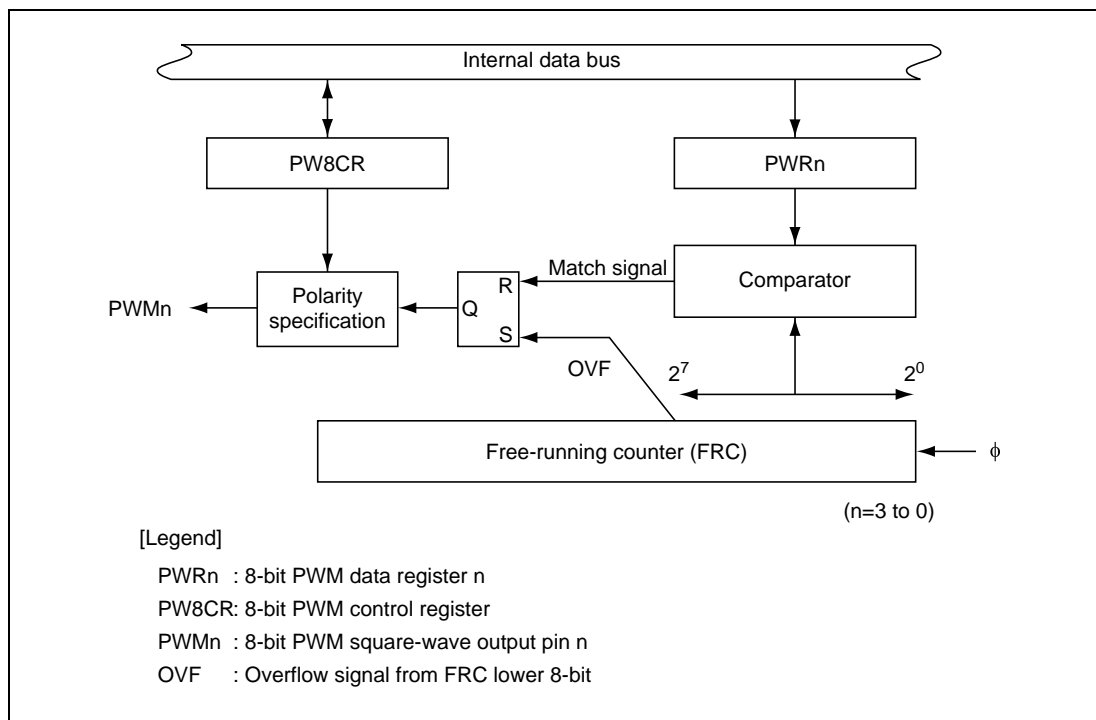


Figure 19.1 Block Diagram of 8-Bit PWM

19.1.3 Pin Configuration

Table 19.1 shows the 8-bit PWM pin configuration.

Table 19.1 Pin Configuration

| Name | Abbrev. | I/O | Function |
|------------------------------------|---------|--------|--------------------------------|
| 8-bit PWM square-wave output pin 0 | PWM0 | Output | 8-bit PWM square-wave output 0 |
| 8-bit PWM square-wave output pin 1 | PWM1 | Output | 8-bit PWM square-wave output 1 |
| 8-bit PWM square-wave output pin 2 | PWM2 | Output | 8-bit PWM square-wave output 2 |
| 8-bit PWM square-wave output pin 3 | PWM3 | Output | 8-bit PWM square-wave output 3 |

19.1.4 Register Configuration

Table 19.2 shows the 8-bit PWM register configuration.

Table 19.2 8-Bit PWM Registers

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|----------------------------|---------|-----|------|---------------|----------|
| 8-bit PWM data register 0 | PWR0 | W | Byte | H'00 | H'D126 |
| 8-bit PWM data register 1 | PWR1 | W | Byte | H'00 | H'D127 |
| 8-bit PWM data register 2 | PWR2 | W | Byte | H'00 | H'D128 |
| 8-bit PWM data register 3 | PWR3 | W | Byte | H'00 | H'D129 |
| 8-bit PWM control register | PW8CR | R/W | Byte | H'F0 | H'D12A |
| Port mode register 3 | PMR3 | R/W | Byte | H'00 | H'FFD0 |

Note: * Lower 16 bits of the address.

19.2 Register Descriptions

19.2.1 Bit PWM Data Registers 0, 1, 2 and 3 (PWR0, PWR1, PWR2, PWR3)

(1) PWR0

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PW07 | PW06 | PW05 | PW04 | PW03 | PW02 | PW01 | PW00 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

(2) PWR1

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PW17 | PW16 | PW15 | PW14 | PW13 | PW12 | PW11 | PW10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

(3) PWR2

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PW27 | PW26 | PW25 | PW24 | PW23 | PW22 | PW21 | PW20 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

(4) PWR3

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PW37 | PW36 | PW35 | PW34 | PW33 | PW32 | PW31 | PW30 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

8-bit PWM data registers 0, 1, 2 and 3 (PWR0, PWR1, PWR2, PWR3) control the duty cycle at 8-bit PWM pins. The data written in PWR0, PWR1, PWR2 and PWR3 correspond to the high-level width of one PWM output waveform cycle (256 states).

When data is set in PWR0, PWR1, PWR2 and PWR3, the contents of the data are latched in the PWM waveform generators, updating the PWM waveform generation data.

PWR0, PWR1, PWR2 and PWR3 are write-only registers. When read, all bits are always read as 1.

PWR0, PWR1, PWR2 and PWR3 are initialized to H'00 by a reset.

19.2.2 8-bit PWM Control Register (PW8CR)

| | | | | | | | | |
|-----------------|---|---|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PWC3 | PWC2 | PWC1 | PWC0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | R/W | R/W | R/W | R/W |

The 8-bit PWM control register (PW8CR) is an 8-bit readable/writable register that controls PWM functions. PW8CR is initialized to H'00 by a reset.

Bits 7 to 4: Reserved

They are always read as 1. Writes are disabled.

Bits 3 to 0: Output Polarity Select (PWC3 to PWC0)

These bits select the output polarity of PWMn pin between positive or negative (reverse).

| Bit n | |
|-------|---|
| PWCn | Description |
| 0 | PWMn pin output has positive polarity (Initial value) |
| 1 | PWMn pin output has negative polarity |

(n = 3 to 0)

19.2.3 Port Mode Register 3 (PMR3)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR37 | PMR36 | PMR35 | PMR34 | PMR33 | PMR32 | PMR31 | PMR30 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The port mode register 3 (PMR3) controls function switching of each pin in the port 3. Switching is specified for each bit.

The PMR3 is a 8-bit readable/writable register and is initialized to H'00 by a reset.

For bits other than 5 to 2, see section 11.5, Port 3.

Bits 5 to 2: P35/PWM3 to P32/PWM0 Pin Switching (PMR35 to PMR32)

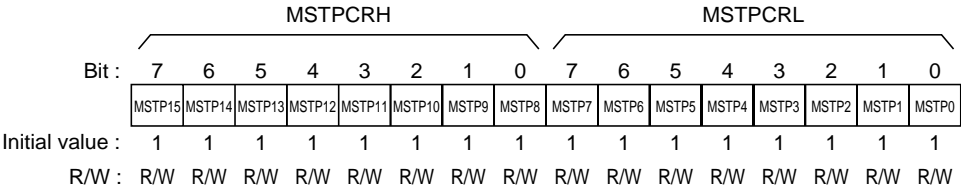
These bits set whether the P3n/PWMn pin is used as I/O pin or it is used as 8-bit PWM output PWMm pin.

Bit n

| PMR3n | Description |
|-------|---|
| 0 | P3n/PMWm pin functions as P3n I/O pin (Initial value) |
| 1 | P3n/PMWm pin functions as PWMm output pin |

(n = 5 to 2, m = 3 to 0)

19.2.4 Module Stop Control Register (MSTPCR)



The MSTPCR consists of two 8-bit readable/writable registers that control module stop mode. When MSTP4 bit is set to 1, the 8-bit PWM stops its operation upon completion of the bus cycle and transits to the module stop mode. For details, see section 4.5, Module Stop Mode. The MSTPCR is initialized to H'FFFF by a reset.

Bit 4: Module Stop (MSTP4)
This bit sets the module stop mode of the 8-bit PWM.

MSTPCRL

Bit 4

| MSTP4 | Description |
|-------|---|
| 0 | 8-bit PWM module stop mode is released |
| 1 | 8-bit PWM module stop mode is set (Initial value) |

19.3 8-Bit PWM Operation

The 8-bit PWM outputs PWM pulses having a cycle length of 256 states and a pulse width determined by the data registers (PWR).
The output PWM pulse can be converted to a DC voltage through integration in a low-pass filter.
Figure 19.2 shows the output waveform example of 8-bit PWM. The pulse width (Twidth) can be obtained by the following expression:
$$Twidth = (1/\phi) \times (PWR \text{ setting value})$$

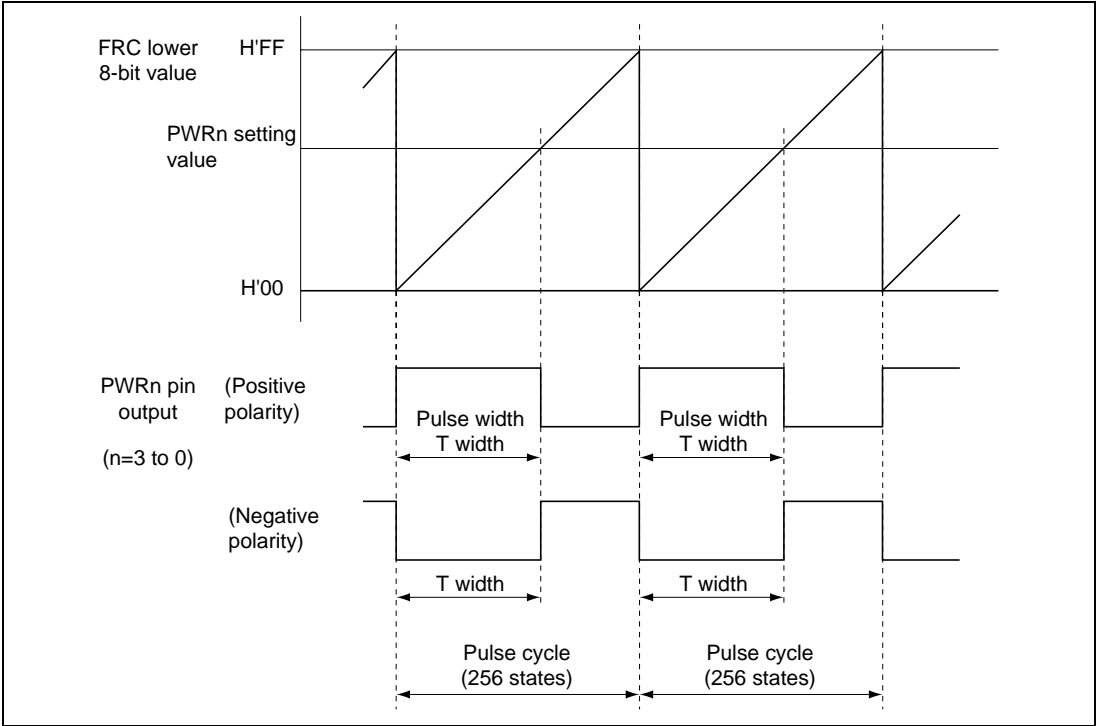


Figure 19.2 8-bit PWM Output Waveform (Example)

20.1 Overview

The 12-bit PWM incorporates 2 channels of the pulse pitch control method and functions as the drum and capstan motor controller.

20.1.1 Features

Two on-chip 12-bit PWM signal generators are provided to control motors. These PWMs use the pulse-pitch control method (periodically overriding part of the output). This reduces low-frequency components in the pulse output, enabling a quick response without increasing the clock frequency. The pitch of the PWM signal is modified in response to error data (representing lead or lag in relation to a preset speed and phase).

20.1.2 Block Diagram

Figure 20.1 shows a block diagram of the 12-bit PWM (1 channel). The PWM signal is generated by combining quantizing pulses from a 12-bit pulse generator with quantizing pulses derived from the contents of a data register. Low-frequency components are reduced because the two quantizing pulses have different frequencies. The error data is represented by an unsigned 12-bit binary number.

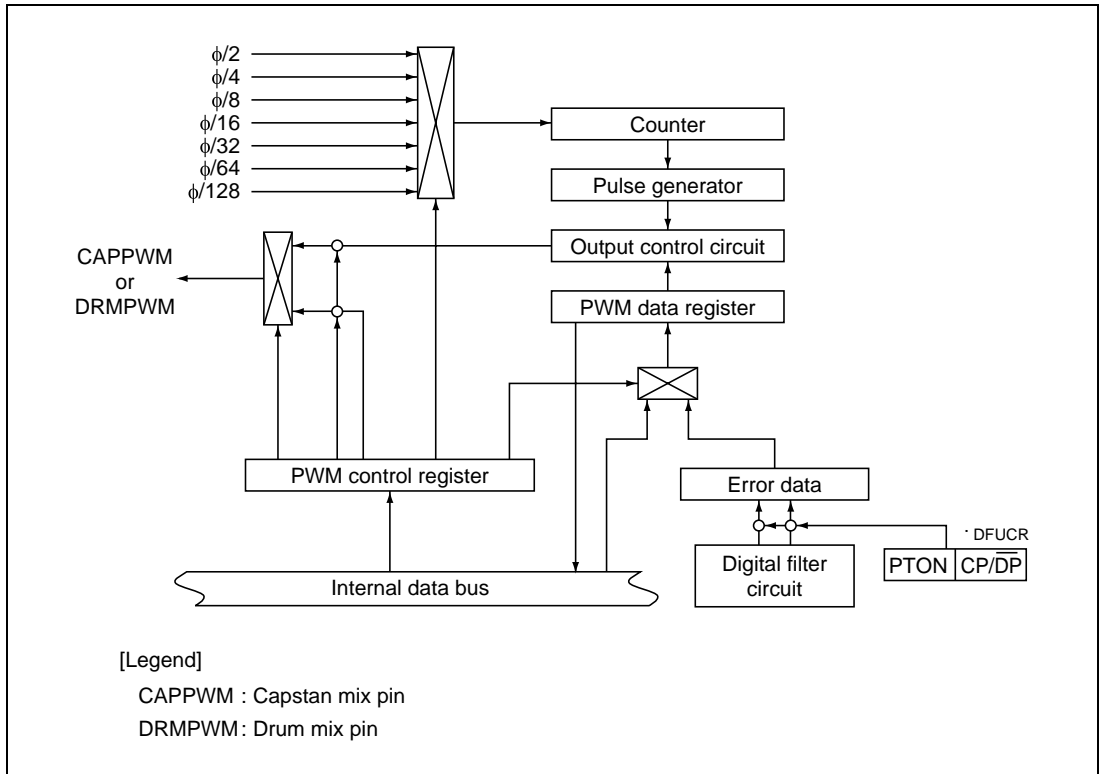


Figure 20.1 Block Diagram of 12-Bit PWM (1 channel)

20.1.3 Pin Configuration

Table 20.1 shows the 12-bit PWM pin configuration.

Table 20.1 Pin Configuration

| Name | Abbrev. | I/O | Function |
|-------------|---------|--------|-------------------------------|
| Capstan mix | CAPPWM | Output | 12-bit PWM square-wave output |
| Drum mix | DRMPWM | | |

20.1.4 Register Configuration

Table 20.2 shows the 12-bit PWM register configuration.

Table 20.2 12-Bit PWM Registers

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|-----------------------------|---------|-----|------|---------------|----------|
| 12-bit PWM control register | CPWCR | W | Byte | H'42 | H'D07B |
| | DPWCR | W | Byte | H'42 | H'D07A |
| 12-bit PWM data register | CPWDR | R/W | Word | H'F000 | H'D07C |
| | DPWDR | R/W | Word | H'F000 | H'D078 |

Note: * Lower 16 bits of the address.

20.2 Register Descriptions

20.2.1 12-Bit PWM Control Registers (CPWCR, DPWCR)

(1) CPWCR

| | | | | | | | | |
|-----------------|------|-----|------|------|--------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CPOL | CDC | CHiZ | CH/L | CSF/DF | CCK2 | CCK1 | CCK0 |
| Initial value : | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

(2) DPWCR

| | | | | | | | | |
|-----------------|------|-----|------|------|--------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | DPOL | DDC | DHiZ | DH/L | DSF/DF | DCK2 | DCK1 | DCK0 |
| Initial value : | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

CPWCR is the PWM output control register for the capstan motor. DPWCR is the PWM output control register for the drum motor. Both are 8-bit writable registers. CPWCR and DPWCR are initialized to H'42 by a reset, or in sleep mode, standby mode, watch mode, subactive mode, subsleep mode, or module stop mode of the servo circuit.

Bit 7: Polarity Invert (POL)

This bit can invert the polarity of the modulated PWM signal for noise suppression and other purposes. This bit is invalid when fixed output is selected (when bit DC is set to 1).

Bit 7

| POL | Description |
|-----|---|
| 0 | Output with positive polarity (Initial value) |
| 1 | Output with inverted polarity |

Bit 6: Output Select (DC)

Selects either PWM modulated output, or fixed output controlled by the pin output bits (Bits 5 and 4).

Bits 5 and 4: PWM Pin Output (HiZ, H/L)

When bit DC is set to 1, the 12-bit PWM output pins (CAPPWM, DRMPWM) output a value determined by the HiZ and H/L bits. The output is not affected by bit POL.

In power-down modes, the 12-bit PWM circuit and pin statuses are retained. Before making a transition to a power-down mode, first set bits 6 (DC), 5 (HiZ), and 4 (H/L) of the 12-bit PWM control registers (CPWCR and DPWCR) to select a fixed output level. Choose one of the following settings:

| Bit 6 | Bit 5 | Bit 4 | Output state |
|-------|-------|-------|----------------------------|
| DC | HiZ | H/L | |
| 1 | 0 | 0 | Low output (Initial value) |
| | | 1 | High output |
| | 1 | * | High-impedance |
| 0 | * | * | Modulation signal output |

Note: * Don't care

Bit 3: Output Data Select (SF/DF)

Selects whether the data to be converted to PWM output is taken from the data register or from the digital filter circuit.

| Bit | | |
|-------|--|-----------------|
| SF/DF | Description | |
| 0 | Modulation by error data from the digital filter circuit | (Initial value) |
| 1 | Modulation by error data written in the data register | |

Note: When PWMs output data from the digital filter circuit, the data consisting of the speed and phase filtering results are modulated by PWMs and output from the CAPPWM and DRMPWM pins. However, it is possible to output only drum phase filter results from CAPPWM pin and only capstan phase filter result from DRMPWM pin, by DFUCR settings of the digital filter circuit. See the section 28.11 Digital Filters.

Bit 2 to 0: Carrier Frequency Select (CK2 to CK0)

Selects the carrier frequency of the PWM modulated signal. Do not set them to 111.

| Bit 2 | Bit 1 | Bit 0 | Description |
|-------|-------|-------|--------------------------|
| CK2 | CK1 | CK0 | |
| 0 | 0 | 0 | $\phi 2$ |
| | | 1 | $\phi 4$ |
| | 1 | 0 | $\phi 8$ (Initial value) |
| | | 1 | $\phi 16$ |
| 1 | 0 | 0 | $\phi 32$ |
| | | 1 | $\phi 64$ |
| | 1 | 0 | $\phi 128$ |
| | | 1 | (Do not set) |

20.2.2 12-Bit PWM Data Registers (CPWDR, DPWDR)

(1) CPWDR

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CPWDR11 | CPWDR10 | CPWDR9 | CPWDR8 | CPWDR7 | CPWDR6 | CPWDR5 | CPWDR4 | CPWDR3 | CPWDR2 | CPWDR1 | CPWDR0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

(2) DPWDR

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | DPWDR11 | DPWDR10 | DPWDR9 | DPWDR8 | DPWDR7 | DPWDR6 | DPWDR5 | DPWDR4 | DPWDR3 | DPWDR2 | DPWDR1 | DPWDR0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

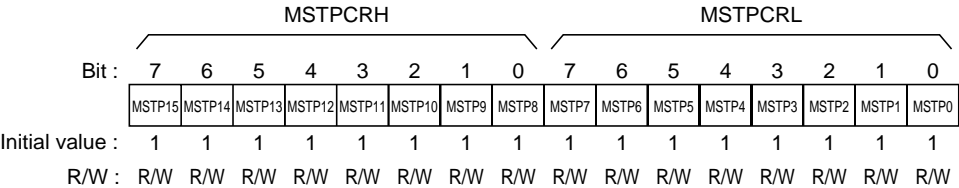
The 12-bit PWM data registers (CPWDR and DPWDR) are 12-bit readable/writable registers in which the data to be converted to PWM output is written.

The data in these registers is converted to PWM output only when bit SF/DF of the corresponding control register is set to 1. The error data from the digital filter circuit is written in the data register, and then modulated by PWM. At this time, the error data from the digital filter circuit can be monitored by reading the data register.

These registers can be accessed by word only, and cannot be accessed by byte. Byte access gives unassured results.

CPWDR and DPWDR are initialized to H'F000 by a reset, or in sleep mode, standby mode, watch mode, subactive mode, subsleep mode, or module stop mode of the servo circuit.

20.2.3 Module Stop Control Register (MSTPCR)



The MSTPCR consists of two 8-bit readable/writable registers that control module stop mode. When the MSTP1 bit is set to 1, the 12-bit PWM and Servo circuit, stops their operation upon completion of the bus cycle and transits to the module stop mode. For details, see section 4.5, Module Stop Mode.

The MSTPCR is initialized to H'FFFF by a reset.

Bit 1: Module Stop (MSTP1)

This bit sets the module stop mode of the 12-bit PWM. This bit also controls the module stop mode of the servo circuit.

MSTPCRL

Bit 1

| MSTP1 | Description |
|-------|---|
| 0 | Module stop mode of the 12-bit PWM and servo circuit is released |
| 1 | Module stop mode of the 12-bit PWM and servo circuit is set (Initial value) |

20.3 Operation

20.3.1 Output Waveform

The PWM signal generator combines the error data with the output from an internal pulse generator to produce a pulse-width modulated signal.

When $V_{cc}/2$ is set as the reference value, the following conditions apply:

- When the motor is running at the correct speed and phase, the PWM signal is output with a 50% duty cycle.
- When the motor is running behind the correct speed or phase, it is corrected by periodically holding part of the PWM signal low. The part held low depends on the size of the error.
- When the motor is running ahead of the correct speed or phase, it is corrected by periodically holding part of the PWM signal high. The part held high depends on the size of the error.

When the motor is running at the correct speed and phase, the error data is a 12-bit value representing $1/2$ (1000 0000 0000), and the PWM output has the same frequency as the selected division clock.

After the error data has been converted into a PWM signal, the PWM signal can be smoothed into a DC voltage by an external low-pass filter (LPF). The smoothed error data can be used to control the motor.

Figure 20.2 shows sample waveform outputs.

The 12-bit PWM pin outputs a low-level signal upon reset, in power-down mode or at module-stop.

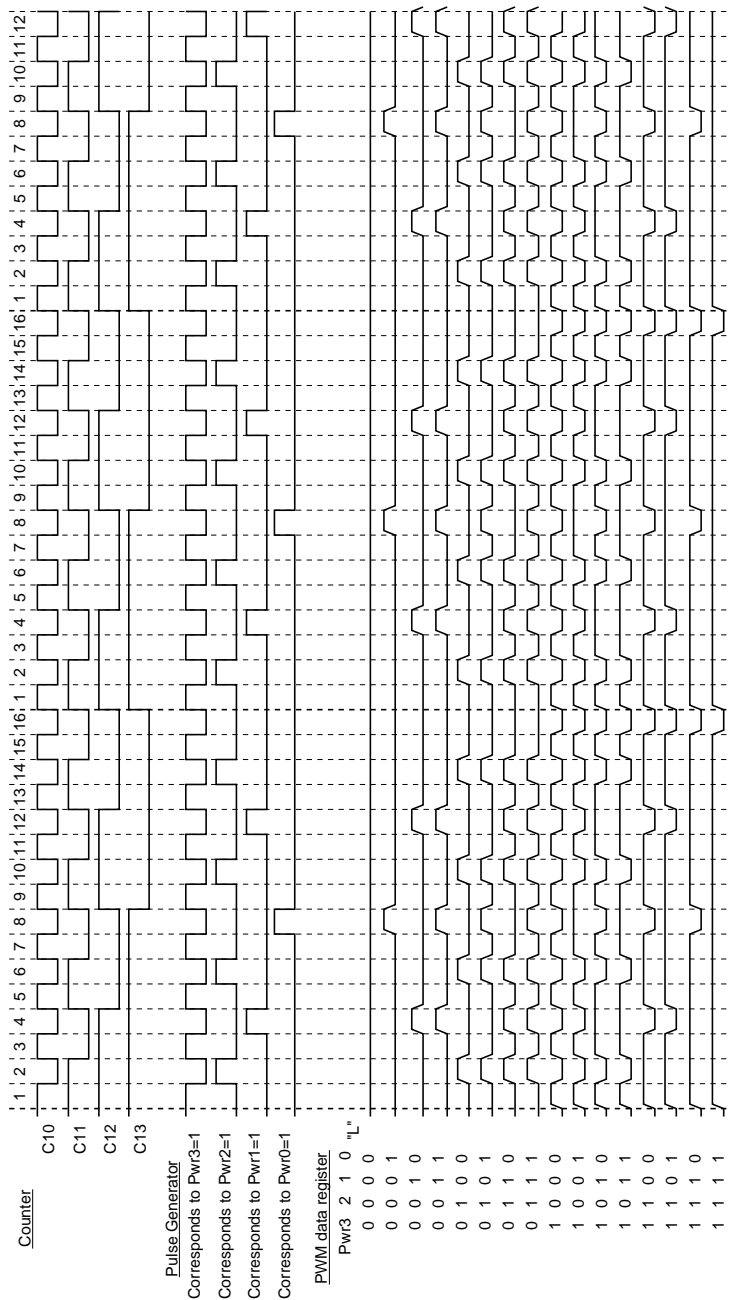


Figure 20.2 Sample Waveform Output by 12-Bit PWM (4 Bits)

Section 21 14-Bit PWM

21.1 Overview

The 14-bit PWM is a pulse division type PWM which can be used for V-synthesizer, etc.

21.1.1 Features

Features of the 14-bit PWM are given below:

- Choice of two conversion periods
A conversion period of $32768/\phi$ with a minimum modulation width of $2/\phi$, or a conversion period of $16384/\phi$ with a minimum modulation width of $1/\phi$, can be selected.
- Pulse division method for less ripple

21.1.2 Block Diagram

Figure 21.1 shows a block diagram of the 14-bit PWM.

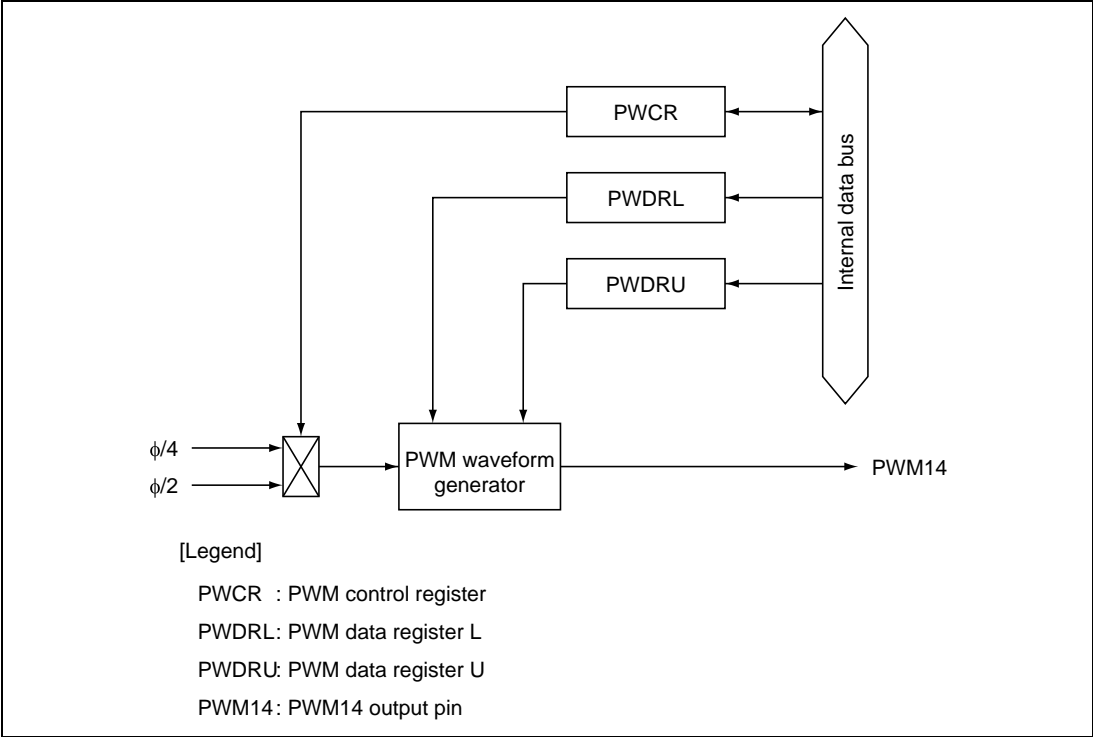


Figure 21.1 Block Diagram of 14-Bit PWM

21.1.3 Pin Configuration

Table 21.1 shows the 14-bit PWM pin configuration.

Table 21.1 Pin Configuration

| Name | Abbrev. | I/O | Function |
|-----------------------------------|---------|--------|-------------------------------|
| PWM 14-bit square-wave output pin | PWM14* | Output | 14-bit PWM square-wave output |

Note: * This pin also functions as P40 general I/O pin. When using this pin, set the pin function by the port mode register 4 (PMR4). For details, see section 11.6, Port 4.

21.1.4 Register Configuration

Table 21.2 shows the 14-bit PWM register configuration.

Table 21.2 14-Bit PWM Registers

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|----------------------|----------------|------------|-------------|----------------------|-----------------|
| PWM control register | PWCR | R/W | Byte | H'FE | H'D122 |
| PWM data register U | PWDRU | W | Byte | H'00 | H'D121 |
| PWM data register L | PWDRL | W | Byte | H'00 | H'D120 |

Note: * Lower 16 bits of the address.

21.2 Register Descriptions

21.2.1 PWM Control Register (PWCR)

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | PWCR0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| R/W : | — | — | — | — | — | — | — | R/W |

The PWM control register (PWCR) is an 8-bit read/write register that controls the 14-bit PWM functions. PWCR is initialized to H'FE by a reset.

Bits 7 to 1: Reserved

They are always read as 1. Writes are disabled.

Bit 0: Clock Select (PWCR0)

Selects the clock supplied to the 14-bit PWM.

Bit 0

| PWCR0 | Description |
|-------|---|
| 0 | The input clock is $\phi/2$ ($t\phi = 2/\phi$) (Initial value) The conversion period is $16384/\phi$, with a minimum modulation width of $1/\phi$ |
| 1 | The input clock is $\phi/4$ ($t\phi = 4/\phi$) The conversion period is $32768/\phi$, with a minimum modulation width of $2/\phi$ |

Note: t/ϕ : Period of PWM clock input

21.2.2 PWM Data Registers U and L (PWDRU, PWDRL)

(1) PWDRU

| | | | | | | | | |
|-----------------|---|---|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | PWDRU5 | PWDRU4 | PWDRU3 | PWDRU2 | PWDRU1 | PWDRU0 |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | W | W | W | W | W | W |

(2) PWDRL

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PWDRL7 | PWDRL6 | PWDRL5 | PWDRL4 | PWDRL3 | PWDRL2 | PWDRL1 | PWDRL0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

PWM data registers U and L (PWDRU and PWDRL) indicate high level width in one PWN waveform cycle.

PWDRU and PWDRL form a 14-bit write-only register, with the upper 6 bits assigned to PWDRU and the lower 8 bits to PWDRL. The value written in PWDRU and PWDRL gives the total high-level width of one PWM waveform cycle. Both PWDRU and PWDRL are accessible by byte access only. Word access gives unassured results.

When 14-bit data is written in PWDRU and PWDRL, the contents are latched in the PWM waveform generator and the PWM waveform generation data is updated. When writing the 14-bit data, follow these steps:

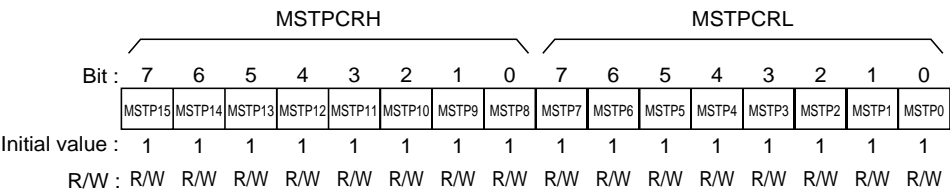
- (1) Write the lower 8 bits to PWDRL.
- (2) Write the upper 6 bits to PWDRU.

Write the data first to PWDRL and then to PWDRU.

PWDRU and PWDRL are write-only registers. When read, all bits always read 1.

PWDRU and PWDRL are initialized to H'C000 by a reset.

21.2.3 Module Stop Control Register (MSTPCR)



The module stop control register (MSTPCR) consists of two 8-bit readable/writable registers that control the module stop mode functions.

When the MSTP5 bit is set to 1, the 14-bit PWM operation stops at the end of the bus cycle and a transition is made to module stop mode. For details, see section 4.5, Module Stop Mode.

MSTPCR is initialized to H'FFFF by a reset.

Bit 5: Module Stop (MSTP5)

Specifies the module stop mode of the 14-bit PWM.

MSTPCRL

Bit 5

| MSTP5 | Description |
|-------|--|
| 0 | 14-bit PWM module stop mode is released |
| 1 | 14-bit PWM module stop mode is set (Initial value) |

21.3 14-Bit PWM Operation

When using the 14-bit PWM, set the registers in this sequence:

- (1) Set bit PMR40 to 1 in port mode register 4 (PMR4) so that pin P40/PWM14 is designated for PWM output.
- (2) Set bit PWCR0 in the PWM control register (PWCR) to select a conversion period of either $32768/\phi$ (PWCR0 = 1) or $16384/\phi$ (PWCR0 = 0).
- (3) Set the output waveform data in PWM data registers U and L (PWDRU, PWDRL). Be sure to write byte data first to PWDRL and then to PWDRU. When the data is written in PWDRU, the contents of these registers are latched in the PWM waveform generator, and the PWM waveform generation data is updated in synchronization with internal signals.

One conversion period consists of 64 pulses, as shown in figure 21.2. The total high-level width during this period (T_H) corresponds to the data in PWDRU and PWDRL. This relation can be expressed as follows:

$$T_H = (\text{data value in PWDRU and PWDRL} + 64) \times t\phi/2$$

where $t\phi$ is the period of PWM clock input: $2/\phi$ (bit PWCR0 = 0) or $4/\phi$ (bit PWCR0 = 1).

If the data value in PWDRU and PWDRL is from H'3FC0 to H'3FFF, the PWM output stays high.

When the data value is H'0000, T_H is calculated as follows:

$$T_H = 64 \times t\phi/2 = 32 \cdot t\phi$$

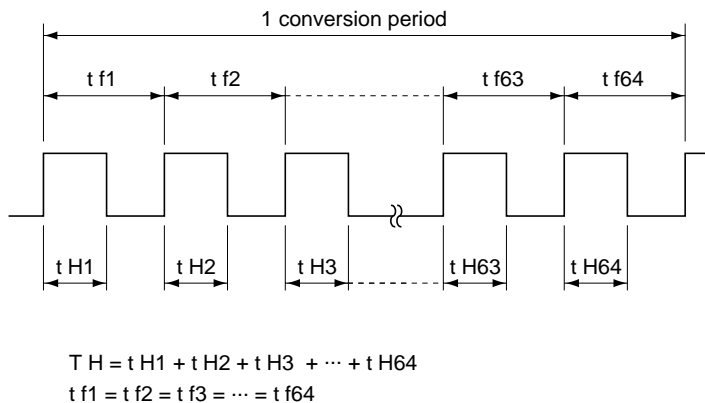


Figure 21.2 Waveform Output by 14-Bit PWM

Section 22 Prescalar Unit

22.1 Overview

The prescalar unit (PSU) has a 18-bit free running counter (FRC) that uses ϕ as a clock source and a 5-bit counter that uses ϕW as a clock source.

22.1.1 Features

- Prescalar S (PSS):
Generates frequency division clocks that are input to peripheral functions.
- Prescalar W (PSW):
When a timer A is used as a clock time base, the PSW frequency-divides subclocks and generates input clocks.
- Stable oscillation wait time count:
During the return from the low power consumption mode excluding the sleep mode, the FRC counts the stable oscillation wait time.
- 8-bit PWM
The lower 8 bits of the FRC is used as 8-bit PWM cycle and duty cycle generation counters. (Conversion cycle: 256 states)
- 8-bit input capture by \overline{IC} pins
Catches the 8 bits of 2^{15} to 2^8 of the FRC according to the edge of the \overline{IC} pin for remote control receiving.
- Frequency division clock output:
Can output the frequency division clock for the system clock or the frequency division clock for the subclock from the frequency division clock output pin (TMOW).

22.1.2 Block Diagram

Figure 22.1 shows a block diagram of the prescaler unit.

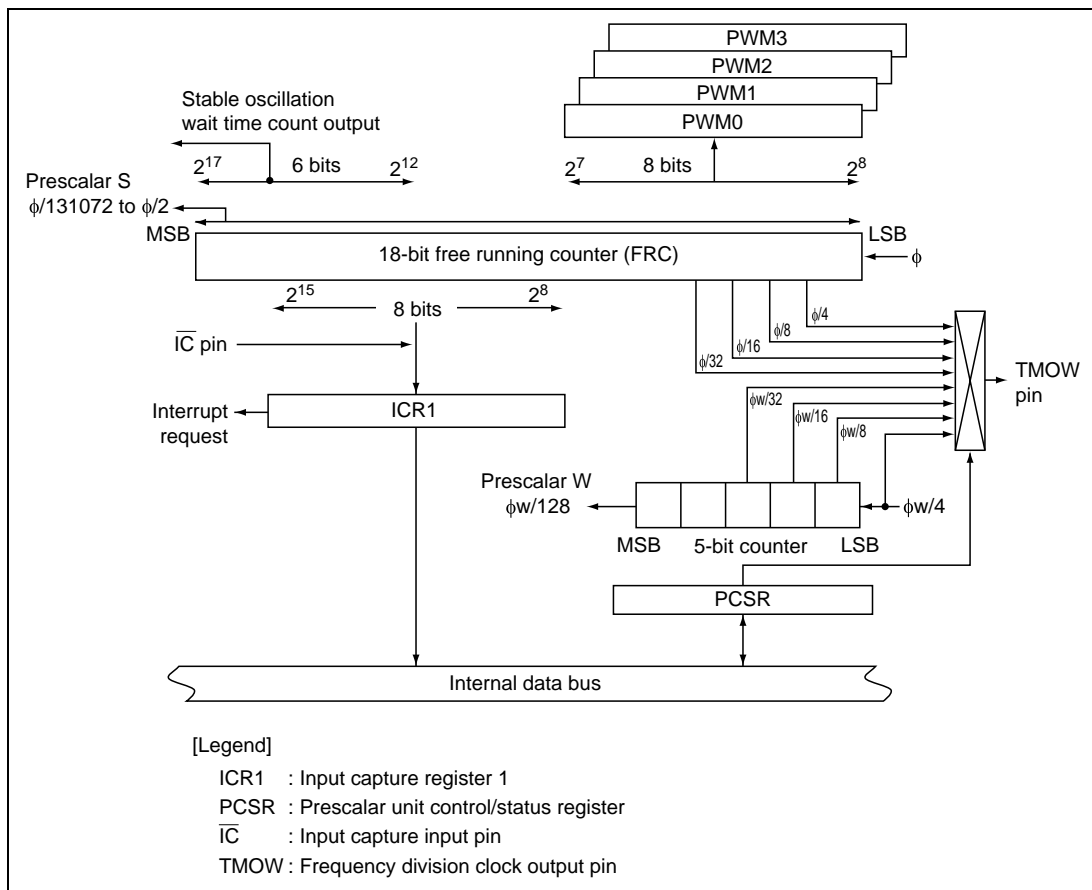


Figure 22.1 Block Diagram of Prescaler Unit

22.1.3 Pin Configuration

Table 22.1 shows the pin configuration of the prescalar unit.

Table 22.1 Pin Configuration

| Name | Abbrev. | I/O | Function |
|---------------------------------|------------------------|--------|--|
| Input capture input | $\overline{\text{IC}}$ | Input | Prescalar unit input capture input pin |
| Frequency division clock output | TMOW | Output | Prescalar unit frequency division clock output pin |

22.1.4 Register Configuration

Table 22.2 shows the register configuration of the prescalar unit.

Table 22.2 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|--|---------|-----|------|---------------|----------|
| Input capture register 1 | ICR1 | R | Byte | H'00 | H'D12C |
| Prescalar unit control/status register | PCSR | R/W | Byte | H'08 | H'D12D |

Note: * Lower 16 bits of the address.

22.2 Registers

22.2.1 Input Capture Register 1 (ICR1)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICR17 | ICR16 | ICR15 | ICR14 | ICR13 | ICR12 | ICR11 | ICR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R |

Input capture register 1 (ICR1) captures 8-bit data of 2^{15} to 2^8 of the FRC according to the edge of the \overline{IC} pin.

ICR1 is an 8-bit read-only register. The write operation becomes invalid. The ICR1 values are undefined until the first capture is generated after the mode has been set to the standby mode, watch mode, subactive mode, and subsleeve mode. When reset, ICR1 is initialized to H'00.

22.2.2 Prescaler Unit Control/Status Register (PCSR)

| | | | | | | | | |
|-----------------|--------|------|------|----------|---|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICIF | ICIE | ICEG | NCon/off | — | DCS2 | DCS1 | DCS0 |
| Initial value : | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/W | R/W | R/W | — | R/W | R/W | R/W |

Note: * Only 0 can be written to clear the flag.

The prescaler unit control/status register (PCSR) controls the input capture function and selects the frequency division clock that is output from the TMOW pin.

PCSR is an 8-bit read/write enable register. When reset, PCSR is initialized to H'08.

Bit 7: Input Capture Interrupt Flag (ICIF)

Input capture interrupt request flag. This indicates that the input capture was performed according to the edge of the \overline{IC} pin.

Bit 7

| ICIF | Description |
|------|--|
| 0 | [Clear condition] When 0 is written after 1 has been read (Initial value) |
| 1 | [Set condition] When the input capture was performed according to the edge of the \overline{IC} pin |

Bit 6: Input Capture Interrupt Enable (ICIE)

When ICIF was set to 1 by the input capture according to the edge of the \overline{IC} pin, ICIE enables and disables the generation of an input capture interrupt.

Bit 6

| ICIE | Description |
|------|---|
| 0 | Disables the generation of an input capture interrupt (Initial value) |
| 1 | Enables the generation of an input capture interrupt |

Bit 5: \overline{IC} Pin Edge Select (ICEG)

ICEG selects the input edge sense of the \overline{IC} pin.

Bit 5

| ICEG | Description |
|------|---|
| 0 | Detects the falling edge of the \overline{IC} pin input (Initial value) |
| 1 | Detects the rising edge of the \overline{IC} pin input |

Bit 4: Noise Cancel ON/OFF (NCon/off)

NCon/off selects enable/disable of the noise cancel function of the \overline{IC} pin. For the noise cancel function, see section 22.3, Noise Cancel Circuit.

Bit 4

| NCon/off | Description |
|----------|---|
| 0 | Disables the noise cancel function of the \overline{IC} pin (Initial value) |
| 1 | Enables the noise cancel function of the \overline{IC} pin |

Bit 3: Reserved Bit

When the bit is read, 1 is always read. The write operation is invalid.

Bits 2 to 0: Frequency Division Clock Output Select (DCS2 to DCS0)

DCS2 to DCS0 select eight types of frequency division clocks that are output from the TMOW pin.

| Bit 2 | Bit 1 | Bit 0 | Description |
|-------|-------|-------|--|
| DCS2 | DCS1 | DCS0 | |
| 0 | 0 | 0 | Outputs PSS, $\phi/32$ (Initial value) |
| | | 1 | Outputs PSS, $\phi/16$ |
| | 1 | 0 | Outputs PSS, $\phi/8$ |
| | | 1 | Outputs PSS, $\phi/4$ |
| 1 | 0 | 0 | Outputs PSW, $\phi W/32$ |
| | | 1 | Outputs PSW, $\phi W/16$ |
| | 1 | 0 | Outputs PSW, $\phi W/8$ |
| | | 1 | Outputs PSW, $\phi W/4$ |

22.2.3 Port Mode Register 1 (PMR1)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR17 | PMR16 | PMR15 | PMR14 | PMR13 | PMR12 | PMR11 | PMR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The port mode register 1 (PMR1) controls switching of each pin function of port 1. The switching is specified in a unit of bit.

PMR1 is an 8-bit read/write enable register. When reset, PMR1 is initialized to H'00.

Bit 7: P17/TMOW Pin Switching (PMR17)

PMR17 sets whether the P17/TMOW pin is used as a P17 I/O pin or a TMOW pin for division clock output.

Bit 7

| PMR17 | Description |
|-------|--|
| 0 | The P17/TMOW pin functions as a P17 I/O pin (Initial value) |
| 1 | The P17/TMOW pin functions as a TMOW pin for division clock output |

Bit 6: P16/ $\overline{\text{IC}}$ Pin Switching (PMR16)

PMR16 sets whether the P16/ $\overline{\text{IC}}$ pin is used as a P16 I/O pin or an $\overline{\text{IC}}$ pin for the input capture input of the prescaler unit.

Bit 6

| PMR16 | Description |
|-------|---|
| 0 | The P16/ $\overline{\text{IC}}$ pin functions as a P16 I/O pin (Initial value) |
| 1 | The P16/ $\overline{\text{IC}}$ pin functions as an $\overline{\text{IC}}$ input function |

22.3 Noise Cancel Circuit

The $\overline{\text{IC}}$ pin has a built-in a noise cancel circuit. The circuit can be used for noise protection such as remote control receiving. The noise cancel circuit samples the input values of the $\overline{\text{IC}}$ pin twice at an interval of 256 states. If the input values are different, they are assumed to be noise. The $\overline{\text{IC}}$ pin can specify enable/disable of the noise cancel function according to the bit 4 (NCon/off) of the prescaler unit control/status register (PCSR).

22.4 Operation

22.4.1 Prescaler S (PSS)

The PSS is a 17-bit counter that uses the system clock ($\phi=f_{osc}$) as an input clock and generates the frequency division clocks ($\phi/131072$ to $\phi/2$) of the peripheral function. The low-order 17 bits of the 18-bit free running counter (FRC) correspond to the PSS. The FRC is incremented by one clock. The PSS output is shared by the timer and serial communication interface (SCI), and the frequency division ratio can independently be set by each built-in peripheral function.

When reset, the FRC is initialized to H'00000, and starts increment after reset has been released. Because the system clock oscillator is stopped in standby mode, watch mode, subactive mode, and subsleep mode, the PSS operation is also stopped. In this case, the FRC is also initialized to H'00000.

The FRC cannot be read and written from the CPU.

22.4.2 Prescalar W (PSW)

PSW is a counter that uses the subclock as an input clock. The PSW also generates the input clock of the timer A. In this case, the timer A functions as a clock time base.

When reset, the PSW is initialized to H'00, and starts increment after reset has been released.

Even if the mode has been shifted to the standby mode *, watch mode *, subactive mode *, and subsleep mode *, the PSW continues the operation as long as the clocks are supplied by the X1 and X2 pins.

The PSW can also be initialized to H'00 by setting the TMA3 and TMA2 bits of the timer mode register A (TMA) to 11.

Note: * When the timer A is in module stop mode, the operation is stopped.

Figure 22.2 shows the supply of the clocks to the peripheral function by the PSS and PSW.

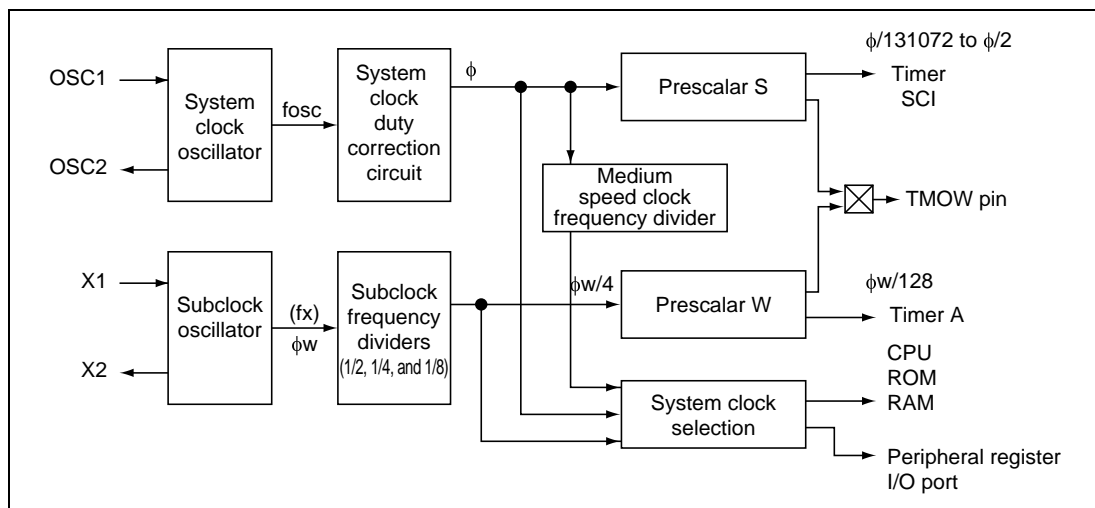


Figure 22.2 Clock Supply

22.4.3 Stable Oscillation Wait Time Count

For the count of the stable oscillation stable wait time during the return from the low power consumption mode excluding the sleep mode, see section 4, Power-Down State.

22.4.4 8-Bit PWM

This 8-bit PWM controls the duty control PWM signal in the conversion cycle 256 states. It counts the cycle and the duty cycle at 2^7 to 2^0 of the FRC. It can be used for controlling reel motors and loading motors. For details, see section 19, 8-Bit PWM.

22.4.5 8-Bit Input Capture Using \overline{IC} Pin

This function catches the 8-bit data of 2^{15} to 2^8 of the FRC according to the edge of the \overline{IC} pin. It can be used for remote control receiving.

For the edge of the \overline{IC} pin, the rising and falling edges can be selected.

The \overline{IC} pin has a built-in noise cancel circuit. See section 22.3, Noise Cancel Circuit.

An interrupt request is generated due to the input capture using the \overline{IC} pin.

Note: Rewriting the ICEG bit, NCon/off bit, or PMR16 bit is incorrectly recognized as edge detection according to the combinations between the state and detection edge of the \overline{IC} pin and the ICIF bit may be set after up to 384 ϕ seconds.

22.4.6 Frequency Division Clock Output

The frequency division clock can be output from the TMOW pin. For the frequency division clock, eight types of clocks can be selected according to the DCS2 to DCS0 bits in PCSR.

The clock in which the system clock was frequency-divided is output in active mode and sleep mode and the clock in which the subclock was frequency-divided is output in active mode*, sleep mode*, and subactive mode.

Note: * When the timer A is in module stop mode, no clock is output.

Section 23 Serial Communication Interface 1 (SCI1)

23.1 Overview

The serial communication interface 1 (SCI1) can handle both asynchronous and clocked synchronous serial communication. A function is also provided for serial communication between processors (multiprocessor communication function).

23.1.1 Features

SCI1 features are listed below.

(1) Choice of asynchronous or clock synchronous serial communication mode

(a) Asynchronous mode

- Serial data communication is executed using an asynchronous system in which synchronization is achieved character by character
Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA)
- A multiprocessor communication function is provided that enables serial data communication with a number of processors
- Choice of 12 serial data transfer formats
Data length: 7 or 8 bits
Stop bit length: 1 or 2 bits
Parity: Even, odd, or none
Multiprocessor bit: 1 or 0
- Receive error detection: Parity, overrun, and framing errors
- Break detection: Break can be detected by reading the SII pin level directly in case of a framing error

(b) Clock synchronous mode

- Serial data communication is synchronized with a clock
Serial data communication can be carried out with other chips that have a synchronous communication function
- One serial data transfer format
Data length: 8 bits
- Receive error detection: Overrun errors detected

(2) Full-duplex communication capability

- The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously
- Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data

(3) Built-in baud rate generator allows any bit rate to be selected

(4) Choice of serial clock source: internal clock from baud rate generator or external clock from SCK1 pin

(5) Four interrupt sources

- Four interrupt sources (transmit-data-empty, transmit-end, receive-data-full, and receive error) that can issue requests independently

23.1.2 Block Diagram

Figure 23.1 shows a block diagram of the SCI1.

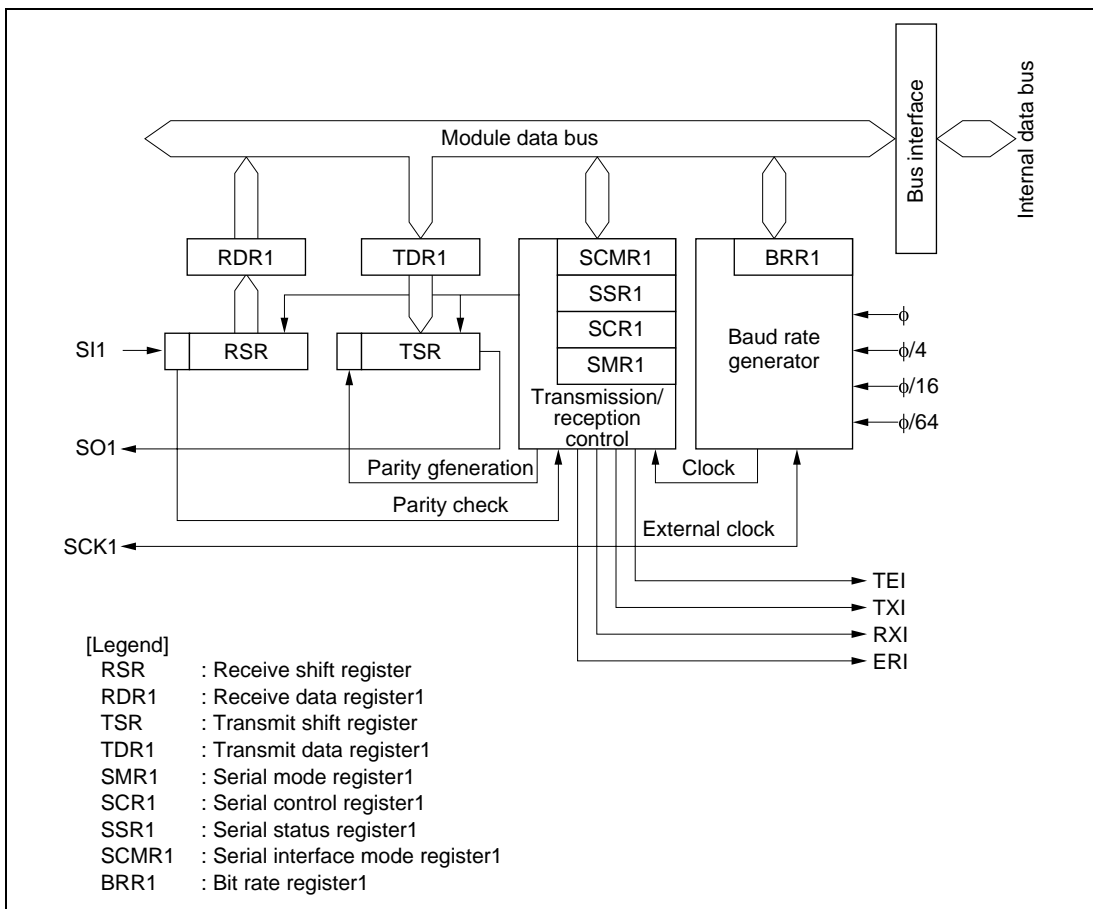


Figure 23.1 Block Diagram of SCI1

23.1.3 Pin Configuration

Table 23.1 shows the serial pins used by the SCI1.

Table 23.1 SCI Pins

| Channel | Pin Name | Symbol | I/O | Function |
|---------|---------------------|--------|--------|---------------------------|
| 1 | Serial clock pin 1 | SCK1 | I/O | SCI1 clock input/output |
| | Receive data pin 1 | SI1 | Input | SCI1 receive data input |
| | Transmit data pin 1 | SO1 | Output | SCI1 transmit data output |

23.1.4 Register Configuration

The SCI1 has the internal registers shown in table 23.2. These registers are used to specify asynchronous mode or synchronous mode, the data format, and the bit rate, and to control the transmitter/receiver.

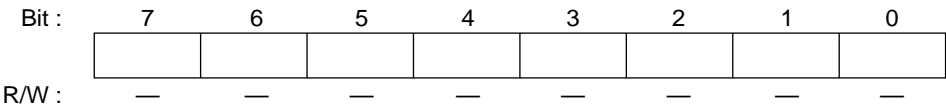
Table 23.2 SCI Registers

| Channel | Name | Abbrev. | R/W | Initial Value | Address ^{*1} |
|---------|----------------------------------|---------------------|---------------------|---------------|-----------------------|
| 1 | Serial mode register 1 | SMR1 | R/W | H'00 | H'D148 |
| | Bit rate register 1 | BRR1 | R/W | H'FF | H'D149 |
| | Serial control register 1 | SCR1 | R/W | H'00 | H'D14A |
| | Transmit data register 1 | TDR1 | R/W | H'FF | H'D14B |
| | Serial status register 1 | SSR1 | R/(W) ^{*2} | H'84 | H'D14C |
| | Receive data register 1 | RDR1 | R | H'00 | H'D14D |
| | Serial interface mode register 1 | SCMR1 | R/W | H'F2 | H'D14E |
| Common | Module stop control register | MSTPCR ^H | R/W | H'FF | H'FFEC |
| | | MSTPCR ^L | R/W | H'FF | H'FFED |

- Notes: 1. Lower 16 bits of the address.
2. Only 0 can be written, to clear flags.

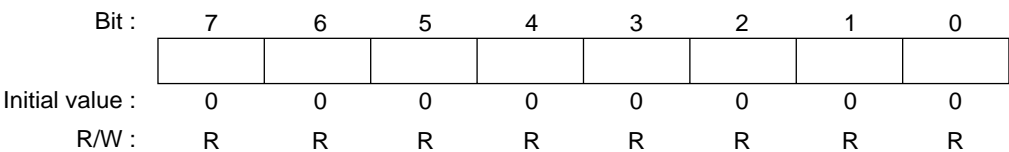
23.2 Register Descriptions

23.2.1 Receive Shift Register (RSR)



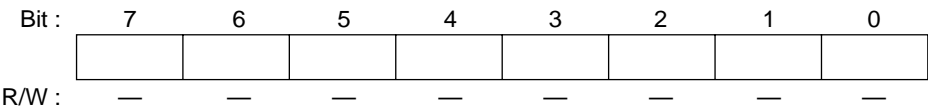
RSR is a register used to receive serial data.
The SCI1 sets serial data input from the SI1 pin in RSR in the order received, starting with the LSB (bit 0), and converts it to parallel data. When one byte of data has been received, it is transferred to RDR1 automatically.
RSR cannot be directly read or written to by the CPU.

23.2.2 Receive Data Register (RDR1)



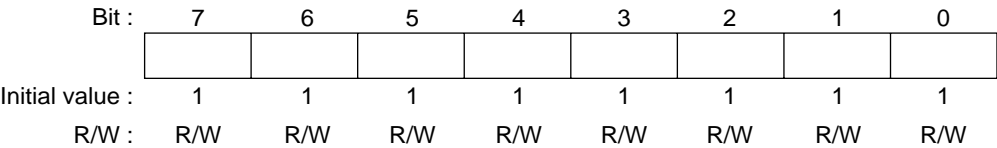
RDR1 is a register that stores received serial data.
When the SCI1 has received one byte of serial data, it transfers the received serial data from RSR to RDR1 where it is stored, and completes the receive operation. After this, RSR is receive-enabled.
Since RSR and RDR1 function as a double buffer in this way, continuous receive operations can be performed.
RDR1 is a read-only register, and cannot be written to by the CPU.
RDR1 is initialized to H'00 by a reset, and in standby mode, watch mode, subactive mode, subsleep mode, and module stop mode.

23.2.3 Transmit Shift Register (TSR)



TSR is a register used to transmit serial data. To perform serial data transmission, the SCI1 first transfers transmit data from TDR1 to TSR, then sends the data to the SO1 pin starting with the LSB (bit 0). When transmission of one byte is completed, the next transmit data is transferred from TDR1 to TSR, and transmission started, automatically. However, data transfer from TDR1 to TSR is not performed if the TDRE bit in SSR1 is set to 1. TSR cannot be directly read or written to by the CPU.

23.2.4 Transmit Data Register (TDR1)



TDR1 is an 8-bit register that stores data for serial transmission. When the SCI1 detects that TSR is empty, it transfers the transmit data written in TDR1 to TSR and starts serial transmission. Continuous serial transmission can be carried out by writing the next transmit data to TDR1 during serial transmission of the data in TSR. TDR1 can be read or written to by the CPU at all times. TDR1 is initialized to H'FF by a reset, and in standby mode, watch mode, subactive mode, subsleep mode, and module stop mode.

23.2.5 Serial Mode Register (SMR1)

| | | | | | | | | |
|-----------------|--------------|-----|-----|--------------|------|-----|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | C/ \bar{A} | CHR | PE | O/ \bar{E} | STOP | MP | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SMR1 is an 8-bit register used to set the SCI1's serial transfer format and select the baud rate generator clock source.

SMR1 can be read or written to by the CPU at all times.

SMR1 is initialized to H'00 by a reset, and in standby mode, watch mode, subactive mode, subsleep mode, and module stop mode.

Bit 7: Communication Mode (C/ \bar{A})

Selects asynchronous mode or clock synchronous mode as the SCI1 operating mode.

Bit 7

| C/ \bar{A} | Description |
|--------------|-----------------------------------|
| 0 | Asynchronous mode (Initial value) |
| 1 | Clock synchronous mode |

Bit 6: Character Length (CHR)

Selects 7 or 8 bits as the data length in asynchronous mode. In synchronous mode, a fixed data length of 8 bits is used regardless of the CHR setting.

Bit 6

| CHR | Description |
|-----|----------------------------|
| 0 | 8-bit data (Initial value) |
| 1 | 7-bit data* |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR1 is not transmitted, and LSB-first/MSB-first selection is not available.

Bit 5: Parity Enable (PE)

In asynchronous mode, selects whether or not parity bit addition is performed in transmission, and parity bit checking in reception. In synchronous mode, or when a multiprocessor format is used, parity bit addition and checking is not performed, regardless of the PE bit setting.

Bit 5

| PE | Description |
|----|---|
| 0 | Parity bit addition and checking disabled (Initial value) |
| 1 | Parity bit addition and checking enabled* |

Note: * When the PE bit is set to 1, the parity (even or odd) specified by the O/\bar{E} bit is added to transmit data before transmission. In reception, the parity bit is checked for the parity (even or odd) specified by the O/\bar{E} bit.

Bit 4: Parity Mode (O/\bar{E})

Selects either even or odd parity for use in parity addition and checking.

The O/\bar{E} bit setting is only valid when the PE bit is set to 1, enabling parity bit addition and checking, in asynchronous mode. The O/\bar{E} bit setting is invalid in synchronous mode, when parity bit addition and checking is disabled in asynchronous mode, and when a multiprocessor format is used.

Bit 4

| O/\bar{E} | Description |
|-------------|---|
| 0 | Even parity ^{*1} (Initial value) |
| 1 | Even parity ^{*2} |

- Notes:
1. When even parity is set, parity bit addition is performed in transmission so that the total number of 1 bits in the transmit character plus the parity bit is even. In reception, a check is performed to see if the total number of 1 bits in the receive character plus the parity bit is even.
 2. When odd parity is set, parity bit addition is performed in transmission so that the total number of 1 bits in the transmit character plus the parity bit is odd. In reception, a check is performed to see if the total number of 1 bits in the receive character plus the parity bit is odd.

Bit 3: Stop Bit Length (STOP)

Selects 1 or 2 bits as the stop bit length in asynchronous mode. The STOP bit setting is only valid in asynchronous mode. If synchronous mode is set the STOP bit setting is invalid since stop bits are not added.

Bit 3

| STOP | Description |
|------|--|
| 0 | 1 stop bit ^{*1} (Initial value) |
| 1 | 2 stop bits ^{*2} |

Notes: 1. In transmission, a single 1 bit (stop bit) is added to the end of a transmit character before it is sent.
2. In transmission, two 1 bits (stop bits) are added to the end of a transmit character before it is sent.

In reception, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, it is treated as a stop bit; if it is 0, it is treated as the start bit of the next transmit character.

Bit 2: Multiprocessor Mode (MP)

Selects multiprocessor format. When multiprocessor format is selected, the PE bit and O/\bar{E} bit parity settings are invalid. The MP bit setting is only valid in asynchronous mode; it is invalid in synchronous mode.

For details of the multiprocessor communication function, see section 23.3.3, Multiprocessor Communication Function.

Bit 2

| MP | Description |
|----|--|
| 0 | Multiprocessor function disabled (Initial value) |
| 1 | Multiprocessor format selected |

Bits 1 and 0: Clock Select 1 and 0 (CKS1, CKS0)

These bits select the clock source for the baud rate generator. The clock source can be selected from ϕ , $\phi/4$, $\phi/16$, and $\phi/64$, according to the setting of bits CKS1 and CKS0.

For the relation between the clock source, the bit rate register setting, and the baud rate, see section 23.2.8, Bit Rate Register.

| Bit 1 | Bit 0 | Description |
|-------|-------|------------------------------|
| CKS1 | CKS0 | |
| 0 | 0 | ϕ clock (Initial value) |
| | 1 | $\phi/4$ clock |
| 1 | 0 | $\phi/16$ clock |
| | 1 | $\phi/64$ clock |

23.2.6 Serial Control Register (SCR1)

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SCR1 is a register that performs enabling or disabling of SCI1 transfer operations, serial clock output in asynchronous mode, and interrupt requests, and selection of the serial clock source.

SCR1 can be read or written to by the CPU at all times.

SCR1 is initialized to H'00 by a reset, and in standby mode, watch mode, subactive mode, subsleep mode, and module stop mode.

Bit 7: Transmit Interrupt Enable (TIE)

Enables or disables transmit-data-empty interrupt (TXI) request generation when serial transmit data is transferred from TDR1 to TSR and the TDRE flag in SSR1 is set to 1.

Bit 7

| TIE | Description |
|-----|---|
| 0 | Transmit-data-empty interrupt (TXI) request disabled* (Initial value) |
| 1 | Transmit-data-empty interrupt (TXI) request enabled |

Note: * TXI interrupt request cancellation can be performed by reading 1 from the TDRE flag, then clearing it to 0, or clearing the TIE bit to 0.

Bit 6: Receive Interrupt Enable (RIE)

Enables or disables receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request generation when serial receive data is transferred from RSR to RDR1 and the RDRF flag in SSR1 is set to 1.

Bit 6

| RIE | Description |
|-----|---|
| 0 | Receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request disabled* (Initial value) |
| 1 | Receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request enabled |

Note: * RXI and ERI interrupt request cancellation can be performed by reading 1 from the RDRF, FER, PER, or ORER flag, then clearing the flag to 0, or clearing the RIE bit to 0.

Bit 5: Transmit Enable (TE)

Enables or disables the start of serial transmission by the SCI1.

Bit 5

| TE | Description |
|----|---|
| 0 | Transmission disabled ^{*1} (Initial value) |
| 1 | Transmission enabled ^{*2} |

Notes: 1. The TDRE flag in SSR1 is fixed at 1.
2. In this state, serial transmission is started when transmit data is written to TDR1 and the TDRE flag in SSR1 is cleared to 0.
SMR1 setting must be performed to decide the transmission format before setting the TE bit to 1.

Bit 4: Receive Enable (RE)

Enables or disables the start of serial reception by the SCI1.

Bit 4

| RE | Description |
|----|--|
| 0 | Reception disabled ^{*1} (Initial value) |
| 1 | Reception enabled ^{*2} |

Notes: 1. Clearing the RE bit to 0 does not affect the RDRF, FER, PER, and ORER flags, which retain their states.
2. Serial reception is started in this state when a start bit is detected in asynchronous mode or serial clock input is detected in synchronous mode.
SMR1 setting must be performed to decide the reception format before setting the RE bit to 1.

Bit 3: Multiprocessor Interrupt Enable (MPIE)

Enables or disables multiprocessor interrupts. The MPIE bit setting is only valid in asynchronous mode when receiving with the MP bit in SMR1 set to 1.

The MPIE bit setting is invalid in clock synchronous mode or when the MP bit is cleared to 0.

Bit 3

| MPIE | Description |
|------|---|
| 0 | Multiprocessor interrupts disabled (normal reception performed) (Initial value) [Clearing conditions] (1) When the MPIE bit is cleared to 0 (2) When data with MPB = 1 is received |
| 1 | Multiprocessor interrupts enabled* Receive interrupt (RXI) requests, receive-error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR are disabled until data with the multiprocessor bit set to 1 is received. |

Note: * When receive data including MPB = 0 is received, receive data transfer from RSR to RDR1, receive error detection, and setting of the RDRF, FER, and ORER flags in SSR1, is not performed. When receive data with MPB = 1 is received, the MPB bit in SSR1 is set to 1, the MPIE bit is cleared to 0 automatically, and generation of RXI and ERI interrupts (when the TIE and RIE bits in SCR1 are set to 1) and FER and ORER flag setting is enabled.

Bit 2: Transmit End Interrupt Enable (TEIE)

Enables or disables transmit-end interrupt (TEI) request generation if there is no valid transmit data in TDR1 when the MSB is transmitted.

Bit 2

| TEIE | Description |
|------|--|
| 0 | Transmit-end interrupt (TEI) request disabled* (Initial value) |
| 1 | Transmit-end interrupt (TEI) request enabled* |

Note: * TEI cancellation can be performed by reading 1 from the TDRE flag in SSR1, then clearing it to 0 and clearing the TEND flag to 0, or clearing the TEIE bit to 0.

Bits 1 and 0: Clock Enable 1 and 0 (CKE1, CKE0)

These bits are used to select the SCI1 clock source and enable or disable clock output from the SCK1 pin. The combination of the CKE1 and CKE0 bits determines whether the SCK1 pin functions as an I/O port, the serial clock output pin, or the serial clock input pin.

The setting of the CKE0 bit, however, is only valid for internal clock operation (CKE1 = 0) in asynchronous mode. The CKE0 bit setting is invalid in synchronous mode, and in the case of external clock operation (CKE1 = 1). Note that the SCI1's operating mode must be decided using SMR1 before setting the CKE1 and CKE0 bits.

For details of clock source selection, see table 23.9 in section 23.3, Operation.

| Bit 1 | Bit 0 | Description | |
|-------|-------|------------------------|--|
| CKE1 | CKE0 | | |
| 0 | 0 | Asynchronous mode | Internal clock/SCK1 pin functions as I/O port ^{*1} |
| | | Clock synchronous mode | Internal clock/SCK1 pin functions as serial clock output ^{*1} |
| | 1 | Asynchronous mode | Internal clock/SCK1 pin functions as clock output ^{*2} |
| | | Clock synchronous mode | Internal clock/SCK1 pin functions as serial clock output |
| 1 | 0 | Asynchronous mode | External clock/SCK1 pin functions as clock input ^{*3} |
| | | Clock synchronous mode | External clock/SCK1 pin functions as serial clock input |
| | 1 | Asynchronous mode | External clock/SCK1 pin functions as clock input ^{*3} |
| | | Clock synchronous mode | External clock/SCK1 pin functions as serial clock input |

- Notes: 1. Initial value
2. Outputs a clock of the same frequency as the bit rate.
3. Inputs a clock with a frequency 16 times the bit rate.

23.2.7 Serial Status Register (SSR1)

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|------|-----|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| R/W : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R | R | R/W |

Note: * Only 0 can be written to clear the flag.

SSR1 is an 8-bit register containing status flags that indicate the operating status of the SCI1, and multiprocessor bits.

SSR1 can be read or written to by the CPU at all times. However, 1 cannot be written to flags TDRE, RDRF, ORER, PER, and FER. Also note that in order to clear these flags they must be read as 1 beforehand. The TEND flag and MPB flag are read-only flags and cannot be modified. SSR1 is initialized to H'84 by a reset, and in standby mode, watch mode, subactive mode, subsleep mode, and module stop mode.

Bit 7: Transmit Data Register Empty (TDRE)

Indicates that data has been transferred from TDR1 to TSR and the next serial data can be written to TDR1.

Bit 7

| TDRE | Description |
|------|---|
| 0 | [Clearing conditions] When 0 is written in TDRE after reading TDRE = 1 |
| 1 | [Setting conditions] (Initial value) (1) When the TE bit in SCR1 is 0 (2) When data is transferred from TDR1 to TSR and data can be written to TDR1 |

Bit 6: Receive Data Register Full (RDRF)

Indicates that the received data is stored in RDR1.

Bit 6

| RDRF | Description |
|------|--|
| 0 | [Clearing conditions] (Initial value) When 0 is written in RDRF after reading RDRF = 1 |
| 1 | [Setting conditions] When serial reception ends normally and receive data is transferred from RSR to RDR1 |

Note: RDR1 and the RDRF flag are not affected and retain their previous values when an error is detected during reception or when the RE bit in SCR1 is cleared to 0.
If reception of the next data is completed while the RDRF flag is still set to 1, an overrun error will occur and the receive data will be lost.

Bit 5: Overrun Error (ORER)

Indicates that an overrun error occurred during reception, causing abnormal termination.

Bit 5

| ORER | Description |
|------|---|
| 0 | [Clearing conditions] When 0 is written in ORER after reading ORER = 1 (Initial value) ^{*1} |
| 1 | [Setting conditions] When the next serial reception is completed while RDRF = 1 ^{*2} |

- Notes:
1. The ORER flag is not affected and retains its previous state when the RE bit in SCR1 is cleared to 0.
 2. The receive data prior to the overrun error is retained in RDR1, and the data received subsequently is lost. Also, subsequent serial reception cannot be continued while the ORER flag is set to 1. In clock synchronous mode, serial transmission cannot be continued, either.

Bit 4: Framing Error (FER)

Indicates that a framing error occurred during reception in asynchronous mode, causing abnormal termination.

Bit 4

| FER | Description |
|-----|---|
| 0 | [Clearing conditions] When 0 is written in FER after reading FER = 1 (Initial value) ^{*1} |
| 1 | [Setting conditions] When the SCI1 checks the stop bit at the end of the receive data when reception ends, and the stop bit is 0 ^{*2} |

- Notes:
1. The FER flag is not affected and retains its previous state when the RE bit in SCR1 is cleared to 0.
 2. In 2-stop-bit mode, only the first stop bit is checked for a value of 1; the second stop bit is not checked. If a framing error occurs, the receive data is transferred to RDR1 but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the FER flag is set to 1. In clock synchronous mode, serial transmission cannot be continued, either.

Bit 3: Parity Error (PER)

Indicates that a parity error occurred during reception using parity addition in asynchronous mode, causing abnormal termination.

Bit 4

| PER | Description |
|-----|---|
| 0 | [Clearing conditions] When 0 is written in PER after reading PER = 1 (Initial value) ^{*1} |
| 1 | [Setting conditions] When, in reception, the number of 1 bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/Ē bit in SMR1 ^{*2} |

- Notes:
1. The PER flag is not affected and retains its previous state when the RE bit in SCR1 is cleared to 0.
 2. If a parity error occurs, the receive data is transferred to RDR1 but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the PER flag is set to 1. In clock synchronous mode, serial transmission cannot be continued, either.

Bit 2: Transmit End (TEND)

Indicates that there is no valid data in TDR1 when the last bit of the transmit character is sent, and transmission has been ended.

The TEND flag is read-only and cannot be modified.

Bit 2

| TEND | Description |
|------|---|
| 0 | [Clearing conditions] When 0 is written in TDRE after reading TDRE = 1 |
| 1 | [Setting conditions] (1) When the TE bit in SCR1 is 0 (2) When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character (Initial value) |

Bit 1: Multiprocessor Bit (MPB)

When reception is performed using a multiprocessor format in asynchronous mode, MPB stores the multiprocessor bit in the receive data.
MPB is a read-only bit, and cannot be modified.

Bit 1

| MPB | Description |
|-----|---|
| 0 | [Clearing conditions] When data with a 0 multiprocessor bit is received (Initial value)* |
| 1 | [Setting conditions] When data with a 1 multiprocessor bit is received |

Note: * Retains its previous state when the RE bit in SCR1 is cleared to 0 with multiprocessor format.

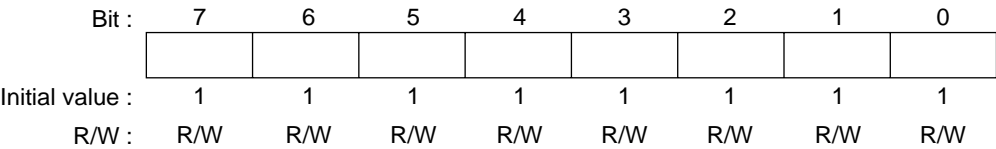
Bit 0: Multiprocessor Bit Transfer (MPBT)

When transmission is performed using a multiprocessor format in asynchronous mode, MPBT stores the multiprocessor bit to be added to the transmit data.
The MPBT bit setting is invalid when a multiprocessor format is not used, when not transmitting, and in synchronous mode.

Bit 0

| MPBT | Description |
|------|---|
| 0 | Data with a 0 multiprocessor bit is transmitted (Initial value) |
| 1 | Data with a 1 multiprocessor bit is transmitted |

23.2.8 Bit Rate Register (BRR1)



BRR1 is an 8-bit register that sets the serial transfer bit rate in accordance with the baud rate generator operating clock selected by bits CKS1 and CKS0 in SMR1.
BRR1 can be read or written to by the CPU at all times.
BRR1 is initialized to H'FF by a reset, and in standby mode, watch mode, subactive mode, subsleep mode, and module stop mode.
Table 23.3 shows sample BRR1 settings in asynchronous mode, and table 23.4 shows sample BRR1 settings in clock synchronous mode.

Table 23.3 BRR1 Settings for Various Bit Rates (Asynchronous Mode)

| Operating Frequency ϕ (MHz) | | | | | | | | | | | | |
|----------------------------------|---|-----|-----------|----------|-----|-----------|--------|-----|-----------|---|-----|-----------|
| Bit Rate (bits/s) | 2 | | | 2.097152 | | | 2.4576 | | | 3 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 1 | 141 | 0.03 | 1 | 148 | −0.04 | 1 | 174 | −0.26 | 1 | 212 | 0.03 |
| 150 | 1 | 103 | 0.16 | 1 | 108 | 0.21 | 1 | 127 | 0.00 | 1 | 155 | 0.16 |
| 300 | 0 | 207 | 0.16 | 0 | 217 | 0.21 | 0 | 255 | 0.00 | 1 | 77 | 0.16 |
| 600 | 0 | 103 | 0.16 | 0 | 108 | 0.21 | 0 | 127 | 0.00 | 0 | 155 | 0.16 |
| 1200 | 0 | 51 | 0.16 | 0 | 54 | −0.70 | 0 | 63 | 0.00 | 0 | 77 | 0.16 |
| 2400 | 0 | 25 | 0.16 | 0 | 26 | 1.14 | 0 | 31 | 0.00 | 0 | 38 | 0.16 |
| 4800 | 0 | 12 | 0.16 | 0 | 13 | −2.48 | 0 | 15 | 0.00 | 0 | 19 | −2.34 |
| 9600 | — | — | — | 0 | 6 | −2.48 | 0 | 7 | 0.00 | 0 | 9 | −2.34 |
| 19200 | — | — | — | — | — | — | 0 | 3 | 0.00 | 0 | 4 | −2.34 |
| 31250 | 0 | 1 | 0.00 | — | — | — | 0 | — | — | 0 | 2 | 0.00 |
| 38400 | — | — | — | — | — | — | 0 | 1 | 0.00 | — | — | — |

| Operating Frequency ϕ (MHz) | | | | | | | | | | | | |
|----------------------------------|--------|-----|-----------|---|-----|-----------|--------|-----|-----------|---|-----|-----------|
| Bit Rate (bits/s) | 3.6864 | | | 4 | | | 4.9152 | | | 5 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 64 | 0.70 | 2 | 70 | 0.03 | 2 | 86 | 0.31 | 2 | 88 | −0.25 |
| 150 | 1 | 191 | 0.00 | 1 | 207 | 0.16 | 1 | 255 | 0.00 | 2 | 64 | 0.16 |
| 300 | 1 | 95 | 0.00 | 1 | 103 | 0.16 | 1 | 127 | 0.00 | 1 | 129 | 0.16 |
| 600 | 0 | 191 | 0.00 | 0 | 207 | 0.16 | 0 | 255 | 0.00 | 1 | 64 | 0.16 |
| 1200 | 0 | 95 | 0.00 | 0 | 103 | 0.16 | 0 | 127 | 0.00 | 0 | 129 | 0.16 |
| 2400 | 0 | 47 | 0.00 | 0 | 51 | 0.16 | 0 | 63 | 0.00 | 0 | 64 | 0.16 |
| 4800 | 0 | 23 | 0.00 | 0 | 25 | 0.16 | 0 | 31 | 0.00 | 0 | 32 | −1.36 |
| 9600 | 0 | 11 | 0.00 | 0 | 12 | 0.16 | 0 | 15 | 0.00 | 0 | 15 | 1.73 |
| 19200 | 0 | 5 | 0.00 | — | — | — | 0 | 7 | 0.00 | 0 | 7 | 1.73 |
| 31250 | — | — | — | 0 | 3 | 0.00 | 0 | 4 | −1.70 | 0 | 4 | 0.00 |
| 38400 | 0 | 2 | 0.00 | — | — | — | 0 | 3 | 0.00 | 0 | 3 | 1.73 |

Operating Frequency ϕ (MHz)

| Bit Rate (bits/s) | 6 | | | 6.144 | | | 7.3728 | | | 8 | | |
|----------------------|---|-----|-----------|-------|-----|-----------|--------|-----|-----------|---|-----|-----------|
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 106 | -0.44 | 2 | 108 | 0.08 | 2 | 130 | -0.07 | 2 | 141 | 0.03 |
| 150 | 2 | 77 | 0.16 | 2 | 79 | 0.00 | 2 | 95 | 0.00 | 2 | 103 | 0.16 |
| 300 | 1 | 155 | 0.16 | 1 | 159 | 0.00 | 1 | 191 | 0.00 | 1 | 207 | 0.16 |
| 600 | 1 | 77 | 0.16 | 1 | 79 | 0.00 | 1 | 95 | 0.00 | 1 | 103 | 0.16 |
| 1200 | 0 | 155 | 0.16 | 0 | 159 | 0.00 | 0 | 191 | 0.00 | 0 | 207 | 0.16 |
| 2400 | 0 | 77 | 0.16 | 0 | 79 | 0.00 | 0 | 95 | 0.00 | 0 | 103 | 0.16 |
| 4800 | 0 | 38 | 0.16 | 0 | 39 | 0.00 | 0 | 47 | 0.00 | 0 | 51 | 0.16 |
| 9600 | 0 | 19 | -2.34 | 0 | 19 | 0.00 | 0 | 23 | 0.00 | 0 | 25 | 0.16 |
| 19200 | 0 | 9 | -2.34 | 0 | 9 | 0.00 | 0 | 11 | 0.00 | 0 | 12 | 0.16 |
| 31250 | 0 | 5 | 0.00 | 0 | 5 | 2.40 | — | — | — | 0 | 7 | 0.00 |
| 38400 | 0 | 4 | -2.34 | 0 | 4 | 0.00 | 0 | 5 | 0.00 | — | — | — |

Operating Frequency ϕ (MHz)

| Bit Rate (bits/s) | 9.8304 | | | 10 | | |
|----------------------|--------|-----|-----------|----|-----|-----------|
| | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 174 | -0.26 | 2 | 177 | -0.25 |
| 150 | 2 | 127 | 0.00 | 2 | 129 | 0.16 |
| 300 | 1 | 255 | 0.00 | 2 | 64 | 0.16 |
| 600 | 1 | 127 | 0.00 | 1 | 129 | 0.16 |
| 1200 | 0 | 255 | 0.00 | 1 | 64 | 0.16 |
| 2400 | 0 | 127 | 0.00 | 0 | 129 | 0.16 |
| 4800 | 0 | 63 | 0.00 | 0 | 64 | 0.16 |
| 9600 | 0 | 31 | 0.00 | 0 | 32 | -1.36 |
| 19200 | 0 | 15 | 0.00 | 0 | 15 | 1.73 |
| 31250 | 0 | 9 | -1.70 | 0 | 9 | 0.00 |
| 38400 | 0 | 7 | 0.00 | 0 | 7 | 1.73 |

Table 23.4 BRR1 Settings for Various Bit Rates (Clock Synchronous Mode)

| Bit Rate (bits/s) | Operating Frequency ϕ (MHz) | | | | | | | |
|----------------------|----------------------------------|-----|---|-----|---|-----|----|-----|
| | 2 | | 4 | | 8 | | 10 | |
| | n | N | n | N | n | N | n | N |
| 110 | 3 | 70 | — | — | | | | |
| 250 | 2 | 124 | 2 | 249 | 3 | 124 | — | — |
| 500 | 1 | 249 | 2 | 124 | 1 | 249 | — | — |
| 1 k | 1 | 124 | 1 | 249 | 2 | 124 | — | — |
| 2.5 k | 0 | 199 | 1 | 99 | 1 | 199 | 1 | 249 |
| 5 k | 0 | 99 | 0 | 199 | 1 | 99 | 1 | 124 |
| 10 k | 0 | 49 | 0 | 99 | 0 | 199 | 0 | 249 |
| 25 k | 0 | 19 | 0 | 39 | 0 | 79 | 0 | 99 |
| 50 k | 0 | 9 | 0 | 19 | 0 | 39 | 0 | 49 |
| 100 k | 0 | 4 | 0 | 9 | 0 | 19 | 0 | 24 |
| 250 k | 0 | 1 | 0 | 3 | 0 | 7 | 0 | 9 |
| 500 k | 0 | 0* | 0 | 1 | 0 | 3 | 0 | 4 |
| 1 M | | | 0 | 0* | 0 | 1 | | |
| 2.5 M | | | | | | | 0 | 0* |
| 5 M | | | | | | | | |

Note: As far as possible, the setting should be made so that the error is no more than 1%.

Legend:

Blank: Cannot be set.

—: Can be set, but there will be a degree of error.

*: Continuous transfer is not possible.

The BRR1 setting is found from the following equations.

- Asynchronous mode:

$$N = \frac{\phi}{64 \times 2^{2n-1} \times B} \times 10^{6-1}$$

- Clock synchronous mode:

$$N = \frac{\phi}{8 \times 2^{2n-1} \times B} \times 10^{6-1}$$

Where

B: Bit rate (bits/s)

N: BRR1 setting for baud rate generator ($0 \leq N \leq 255$)

ϕ : Operating frequency (MHz)

n: Baud rate generator input clock ($n = 0$ to 3)

(See the table below for the relation between n and the clock.)

| n | Clock | SMR1 Setting | |
|---|-----------|--------------|------|
| | | CKS1 | CKS0 |
| 0 | ϕ | 0 | 0 |
| 1 | $\phi/4$ | 0 | 1 |
| 2 | $\phi/16$ | 1 | 0 |
| 3 | $\phi/64$ | 1 | 1 |

The bit rate error in asynchronous mode is found from the following equation:

$$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{(N+1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

Table 23.5 shows the maximum bit rate for each frequency in asynchronous mode. Tables 23.6 and 23.7 show the maximum bit rates with external clock input.

Table 23.5 Maximum Bit Rate for Each Frequency (Asynchronous Mode)

| ϕ (MHz) | Maximum Bit Rate (bits/s) | n | N |
|--------------|---------------------------|---|---|
| 2 | 62500 | 0 | 0 |
| 2.097152 | 65536 | 0 | 0 |
| 2.4576 | 76800 | 0 | 0 |
| 3 | 93750 | 0 | 0 |
| 3.6864 | 115200 | 0 | 0 |
| 4 | 125000 | 0 | 0 |
| 4.9152 | 153600 | 0 | 0 |
| 5 | 156250 | 0 | 0 |
| 6 | 187500 | 0 | 0 |
| 6.144 | 192000 | 0 | 0 |
| 7.3728 | 230400 | 0 | 0 |
| 8 | 250000 | 0 | 0 |
| 9.8304 | 307200 | 0 | 0 |
| 10 | 312500 | 0 | 0 |

Table 23.6 Maximum Bit Rate with External Clock Input (Asynchronous Mode)

| ϕ (MHz) | External Input Clock (MHz) | Maximum Bit Rate (bits/s) |
|--------------|----------------------------|---------------------------|
| 2 | 0.5000 | 31250 |
| 2.097152 | 0.5243 | 32768 |
| 2.4576 | 0.6144 | 38400 |
| 3 | 0.7500 | 46875 |
| 3.6864 | 0.9216 | 57600 |
| 4 | 1.0000 | 62500 |
| 4.9152 | 1.2288 | 76800 |
| 5 | 1.2500 | 78125 |
| 6 | 1.5000 | 83750 |
| 6.144 | 1.5360 | 96000 |
| 7.3728 | 1.8432 | 115200 |
| 8 | 2.0000 | 125000 |
| 9.8304 | 2.4576 | 153600 |
| 10 | 2.5000 | 156250 |

Table 23.7 Maximum Bit Rate with External Clock Input (Clock Synchronous Mode)

| ϕ (MHz) | External Input Clock (MHz) | Maximum Bit Rate (bits/s) |
|--------------|----------------------------|---------------------------|
| 2 | 0.3333 | 333333.3 |
| 4 | 0.6667 | 666666.7 |
| 6 | 1.0000 | 1000000.0 |
| 8 | 1.3333 | 1333333.3 |
| 10 | 1.6667 | 1666666.7 |

23.2.9 Serial Interface Mode Register (SCMR1)

| | | | | | | | | |
|-----------------|---|---|---|---|------|------|---|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| R/W : | — | — | — | — | R/W | R/W | — | R/W |

SCMR1 is an 8-bit readable/writable register used to select SCI1 functions.

SCMR1 is initialized to H'F2 by a reset, and in standby mode, watch mode, subactive mode, subsleep mode, and module stop mode.

Bits 7 to 4: Reserved

These bits cannot be modified and are always read as 1.

Bit 3: Data Transfer Direction (SDIR)

Selects the serial/parallel conversion format.

Bit 3

| SDIR | Description |
|------|---|
| 0 | TDR1 contents are transmitted LSB-first Receive data is stored in RDR1 LSB-first (Initial value) |
| 1 | TDR1 contents are transmitted MSB-first Receive data is stored in RDR1 MSB-first |

Bit 2: Data Invert (SINV)

Specifies inversion of the data logic level. The SINV bit does not affect the logic level of the parity bit(s): parity bit inversion requires inversion of the O/ \bar{E} bit in SMR1.

Bit 2

| SINV | Description |
|------|---|
| 0 | TDR1 contents are transmitted without modification Receive data is stored in RDR1 without modification (Initial value) |
| 1 | TDR1 contents are inverted before being transmitted Receive data is stored in RDR1 in inverted form |

Bit 1: Reserved

This bit cannot be modified and is always read as 1.

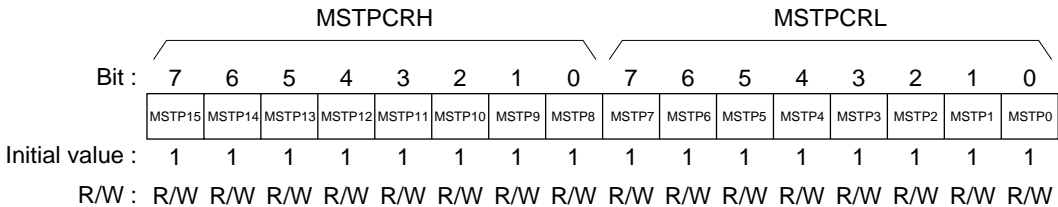
Bit 0: Serial Communication Interface Mode Select (SMIF)

1 should not be written in this bit.

Bit 0

| SMIF | Description |
|------|---------------------------------|
| 0 | Normal SCI mode (Initial value) |
| 1 | Reserved mode |

23.2.10 Module Stop Control Register (MSTPCR)



MSTPCR, comprising two 8-bit readable/writable registers, performs module stop mode control. When bit MSTP8 is set to 1, SCI1 operation stops at the end of the bus cycle and a transition is made to module stop mode. For details, see section 4.5, Module Stop Mode. MSTPCR is initialized to H'FFFF by a reset.

Bit 0: Module Stop (MSTP8)

Specifies the SCI1 module stop mode.

MSTPCRH

Bit 0

| MSTP8 | Description |
|-------|--|
| 0 | SCI1 module stop mode is cleared |
| 1 | SCI1 module stop mode is set (Initial value) |

23.3 Operation

23.3.1 Overview

The SCI1 can carry out serial communication in two modes: asynchronous mode in which synchronization is achieved character by character, and synchronous mode in which synchronization is achieved with clock pulses.

Selection of asynchronous or synchronous mode and the transmission format is made using SMR1 as shown in table 23.8. The SCI1 clock is determined by a combination of the C/A bit in SMR1 and the CKE1 and CKE0 bits in SCR1, as shown in table 23.9.

(1) Asynchronous Mode

- Data length: Choice of 7 or 8 bits
- Choice of parity addition, multiprocessor bit addition, and addition of 1 or 2 stop bits (the combination of these parameters determines the transfer format and character length)
- Detection of framing, parity, and overrun errors, and breaks, during reception
- Choice of internal or external clock as SCI1 clock source
 - When internal clock is selected:
The SCI1 operates on the baud rate generator clock and a clock with the same frequency as the bit rate can be output
 - When external clock is selected:
A clock with a frequency of 16 times the bit rate must be input (the built-in baud rate generator is not used)

(2) Clock Synchronous Mode

- Transfer format: Fixed 8-bit data
- Detection of overrun errors during reception
- Choice of internal or external clock as SCI1 clock source
 - When internal clock is selected:
The SCI1 operates on the baud rate generator clock and a serial clock is output off-chip
 - When external clock is selected:
The built-in baud rate generator is not used, and the SCI1 operates on the input serial clock

Table 23.8 SMR1 Settings and Serial Transfer Format Selection

| SMR1 Settings | | | | | SCI1 Transfer Format | | | | | | | |
|-------------------|-------|-------|-------|-------|---|---------------|--------------------------|----------------|-----------------|--|--|--------|
| Bit 7 | Bit 6 | Bit 2 | Bit 5 | Bit 3 | Mode | Data length | Multiproc- -essor bit | Parit y bit | Stop bit length | | | |
| C/ \overline{A} | CHR | MP | PE | STOP | | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | Asynchro- nous mode | 8-bit data | No | No | 1 bit | | | |
| | | | | 1 | | | | | 2 bits | | | |
| | | | 1 | 0 | | | | Yes | 1 bit | | | |
| | | | | 1 | | | | | 2 bits | | | |
| | | 1 | 0 | 0 | | 7-bit data | | No | 1 bit | | | |
| | | | | 1 | | | | | 2 bits | | | |
| | | | 1 | 0 | | | | Yes | 1 bit | | | |
| | | | | 1 | | | | | 2 bits | | | |
| | 0 | 1 | — | 0 | Asynchro- nous mode (multi- processor format) | 8-bit data | Yes | No | 1 bit | | | |
| | | | | 1 | | | | | 2 bits | | | |
| | | | 1 | — | | 0 | | | 7-bit data | | | 1 bit |
| | | | | — | | 1 | | | | | | 2 bits |
| 1 | — | — | — | — | Clock synchronous mode | 8-bit data | No | | | | | |

Table 23.9 SMR1 and SCR1 Settings and SCI1 Clock Source Selection

| SMR1 SCR1 Setting | | | SCI1 Transfer Clock | | |
|----------------------|-------|-------|------------------------------|--------------|---|
| Bit 7 | Bit 1 | Bit 0 | Mode | Clock Source | SCK1 Pin Function |
| C/ \bar{A} | CKE1 | CKE0 | | | |
| 0 | 0 | 0 | Asynchronous mode | Internal | SCI1 does not use SCK1 pin |
| | | 1 | | | Outputs clock with same frequency as bit rate |
| | 1 | 0 | | External | Inputs clock with frequency of 16 times the bit rate |
| | | 1 | | | |
| 1 | 0 | 0 | Clock synchronous mode | Internal | Outputs serial clock |
| | | 1 | | | |
| | 1 | 0 | | External | Inputs serial clock |
| | | 1 | | | |

23.3.2 **Operation in Asynchronous Mode**

In asynchronous mode, characters are sent or received, each preceded by a start bit indicating the start of communication and followed by one or two stop bits indicating the end of communication. Serial communication is thus carried out with synchronization established on a character-by-character basis.

Inside the SCI1, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 23.2 shows the general format for asynchronous serial communication.

In asynchronous serial communication, the transmission line is usually held in the mark state (high level). The SCI1 monitors the transmission line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication.

One serial communication character consists of a start bit (low level), followed by data (in LSB-first order), a parity bit (high or low level), and finally one or two stop bits (high level).

In asynchronous mode, the SCI1 performs synchronization at the falling edge of the start bit in reception. The SCI1 samples the data on the 8th pulse of a clock with a frequency of 16 times the length of one bit, so that the transfer data is latched at the center of each bit.

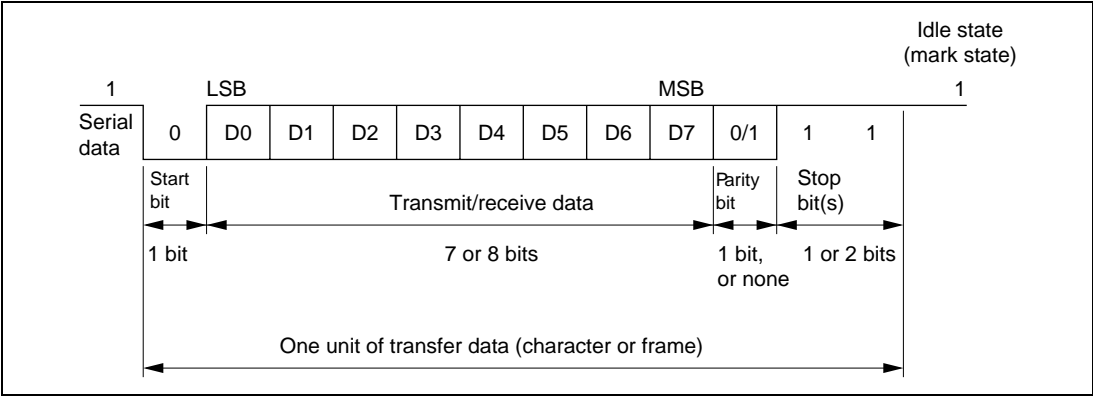


Figure 23.2 Data Format in Asynchronous Communication
(Example with 8-Bit Data, Parity, Two Stop Bits)

(1) Data Transfer Format

Table 23.10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected by settings in SMR1.

Table 23.10 Serial Transfer Formats (Asynchronous Mode)

| SMR1 Settings | | | | Serial Transfer Format and Frame Length | | | | | | | | | | | |
|---------------|----|----|------|---|------------|---|---|---|---|---|---|------|------|------|------|
| CHR | PE | MP | STOP | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 0 | 0 | 0 | 0 | S | 8-bit data | | | | | | | | STOP | | |
| 0 | 0 | 0 | 1 | S | 8-bit data | | | | | | | | STOP | STOP | |
| 0 | 1 | 0 | 0 | S | 8-bit data | | | | | | | | P | STOP | |
| 0 | 1 | 0 | 1 | S | 8-bit data | | | | | | | | P | STOP | STOP |
| 1 | 0 | 0 | 0 | S | 7-bit data | | | | | | | STOP | | | |
| 1 | 0 | 0 | 1 | S | 7-bit data | | | | | | | STOP | STOP | | |
| 1 | 1 | 0 | 0 | S | 7-bit data | | | | | | | P | STOP | | |
| 1 | 1 | 0 | 1 | S | 7-bit data | | | | | | | P | STOP | STOP | |
| 0 | — | 1 | 0 | S | 8-bit data | | | | | | | | MPB | STOP | |
| 0 | — | 1 | 1 | S | 8-bit data | | | | | | | | MPB | STOP | STOP |
| 1 | — | 1 | 0 | S | 7-bit data | | | | | | | MPB | STOP | | |
| 1 | — | 1 | 1 | S | 7-bit data | | | | | | | MPB | STOP | STOP | |

[Legend]

S : Start bit

STOP : Stop bit

P : Parity bit

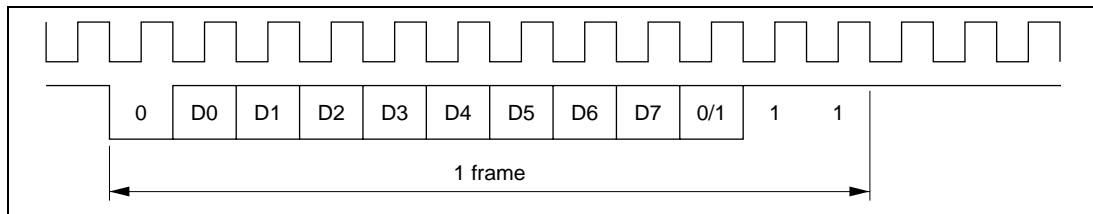
MPB : Multiprocessor bit

(2) Clock

Either an internal clock generated by the built-in baud rate generator or an external clock input at the SCK1 pin can be selected as the SCI1's serial clock, according to the setting of the C/\overline{A} bit in SMR1 and the CKE1 and CKE0 bits in SCR1. For details of SCI1 clock source selection, see table 23.9.

When an external clock is input at the SCK1 pin, the clock frequency should be 16 times the bit rate used.

When the SCI1 is operated on an internal clock, the clock can be output from the SCK1 pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is at the center of each transmit data bit, as shown in figure 23.3.



**Figure 23.3 Relation between Output Clock and Transfer Data Phase
(Asynchronous Mode)**

(3) Data Transfer Operations

(a) SCI1 Initialization (Asynchronous Mode)

Before transmitting and receiving data, first clear the TE and RE bits in SCR1 to 0, then initialize the SCI1 as described below.

When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and TSR is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of RDR1.

When an external clock is used the clock should not be stopped during operation, including initialization, since operation is uncertain.

Figure 23.4 shows a sample SCI1 initialization flowchart.

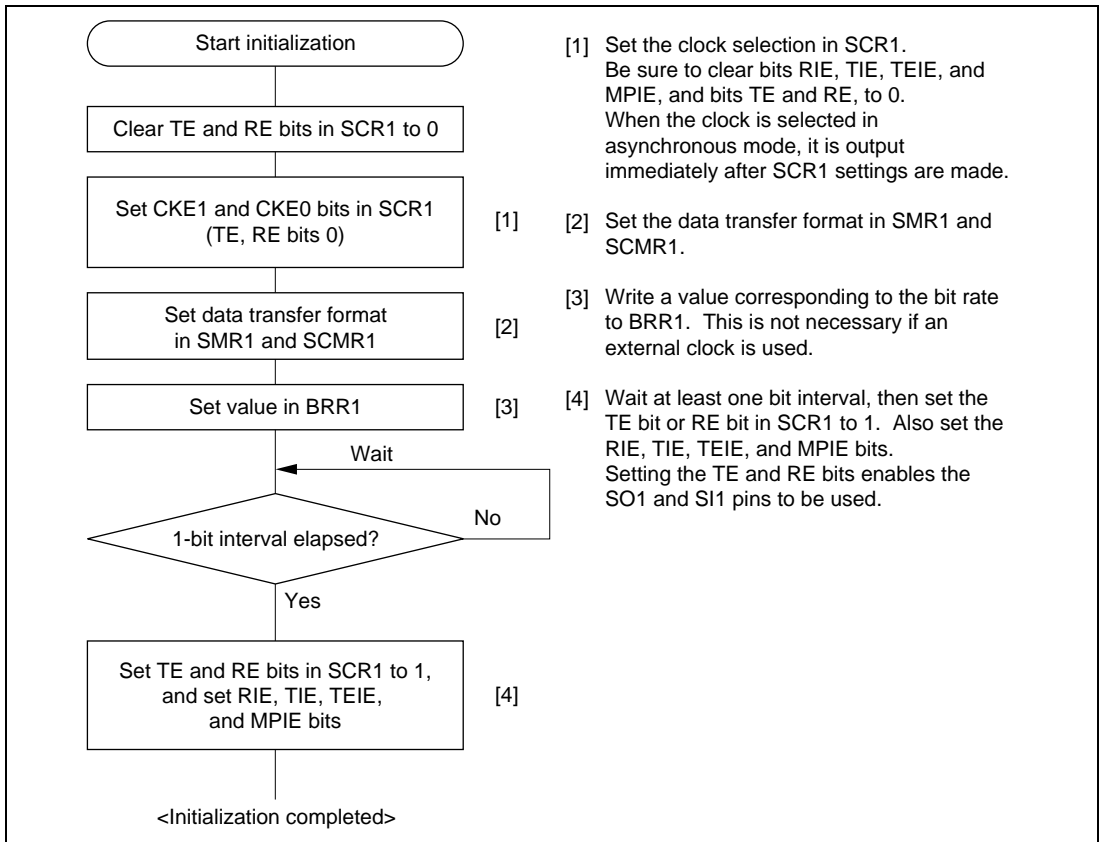


Figure 23.4 Sample SCI Initialization Flowchart

(b) Serial Data Transmission (Asynchronous Mode)

Figure 23.5 shows a sample flowchart for serial transmission.

The following procedure should be used for serial data transmission.

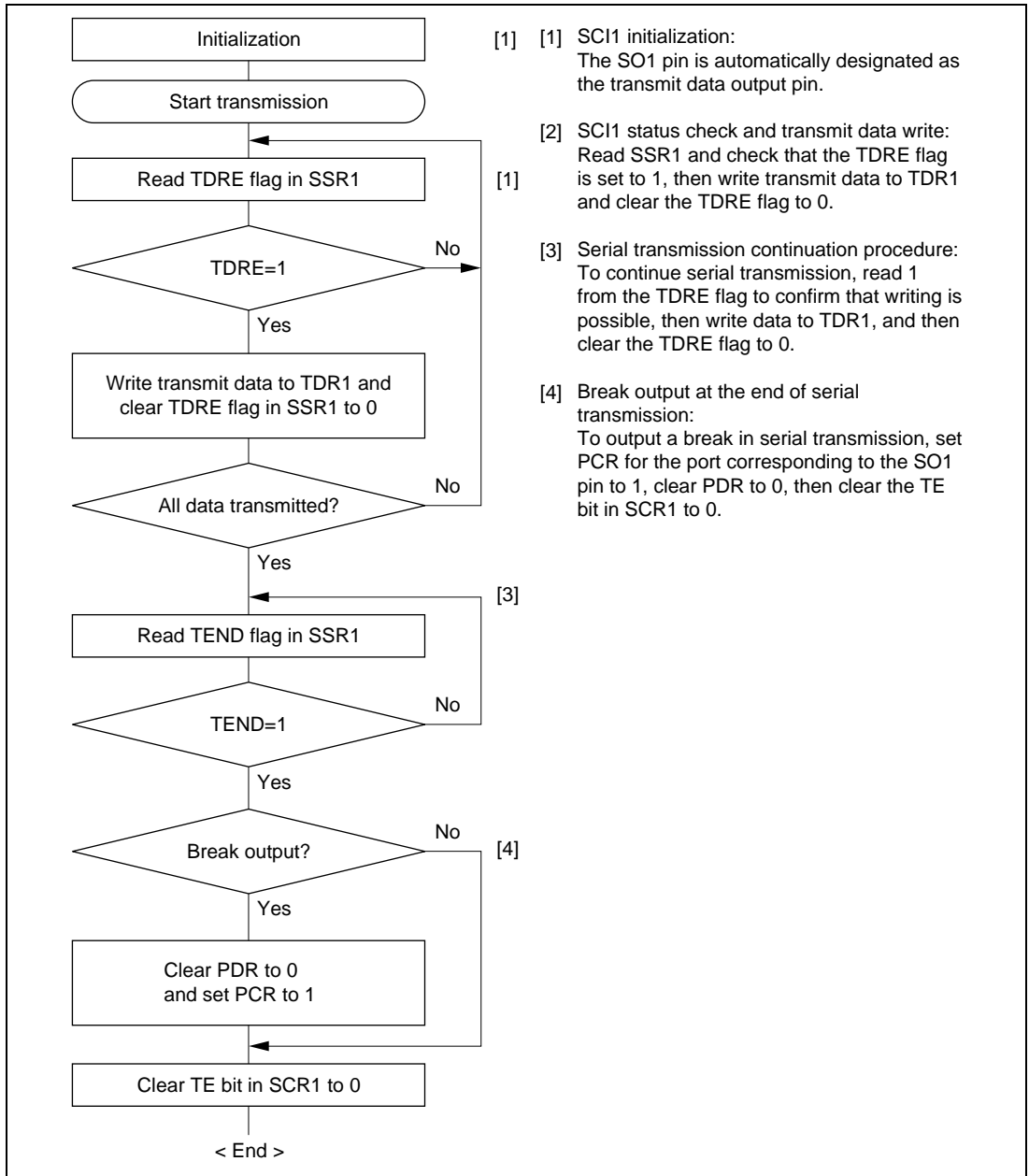


Figure 23.5 Sample Serial Transmission Flowchart

In serial transmission, the SCI1 operates as described below.

- [1] The SCI1 monitors the TDRE flag in SSR1, and if it is 0, recognizes that data has been written to TDR1, and transfers the data from TDR1 to TSR.
- [2] After transferring data from TDR1 to TSR, the SCI1 sets the TDRE flag to 1 and starts transmission.

If the TIE bit is set to 1 at this time, a transmit data empty interrupt (TXI) is generated. The serial transmit data is sent from the SO1 pin in the following order.

[a] Start bit:

One 0-bit is output.

[b] Transmit data:

8-bit or 7-bit data is output in LSB-first order.

[c] Parity bit or multiprocessor bit:

One parity bit (even or odd parity), or one multiprocessor bit is output.

A format in which neither a parity bit nor a multiprocessor bit is output can also be selected.

[d] Stop bit(s):

One or two 1-bits (stop bits) are output.

[e] Mark state:

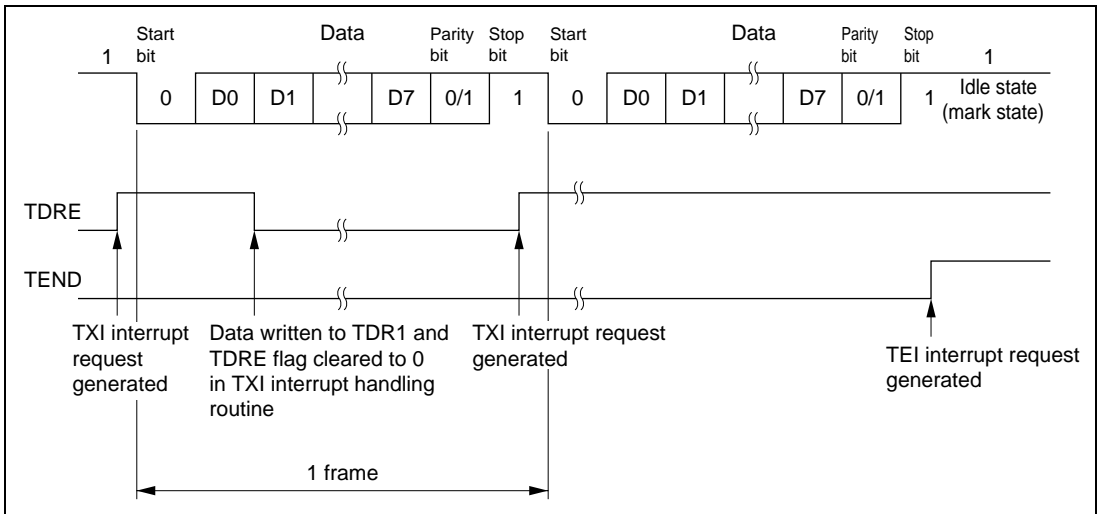
1 is output continuously until the start bit that starts the next transmission is sent.

- [3] The SCI1 checks the TDRE flag at the timing for sending the stop bit.

If the TDRE flag is cleared to 0, the data is transferred from TDR1 to TSR, the stop bit is sent, and then serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR1 is set to 1, the stop bit is sent, and then the mark state is entered in which 1 is output continuously. If the TEIE bit in SCR1 is set to 1 at this time, a TEI interrupt request is generated.

Figure 23.6 shows an example of the operation for transmission in asynchronous mode.



**Figure 23.6 Example of Operation in Transmission in Asynchronous Mode
(Example with 8-Bit Data, Parity, One Stop Bit)**

(c) Serial Data Reception (Asynchronous Mode)

Figure 23.7 shows a sample flowchart for serial reception.

The following procedure should be used for serial data reception.

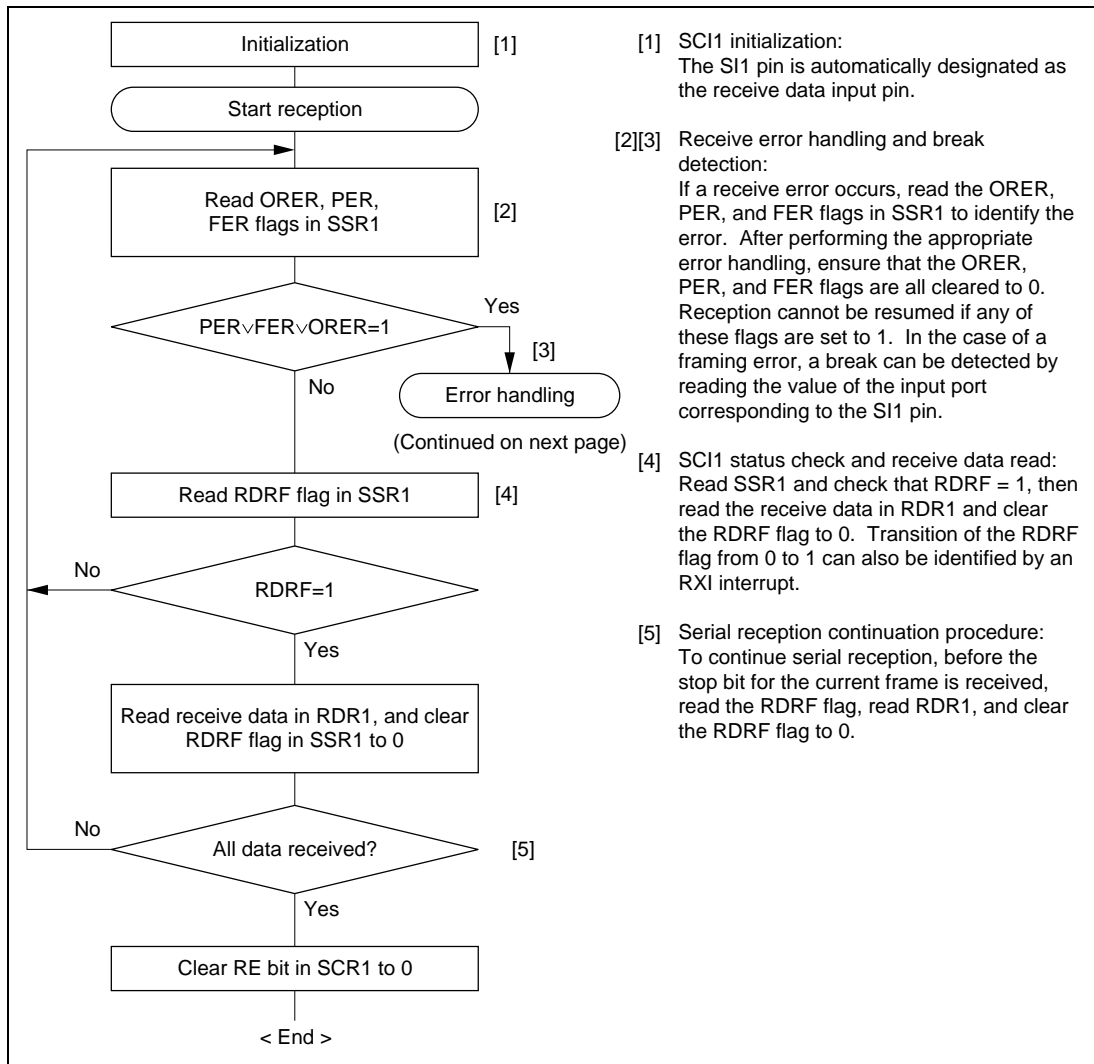


Figure 23.7 Sample Serial Reception Data Flowchart (1)

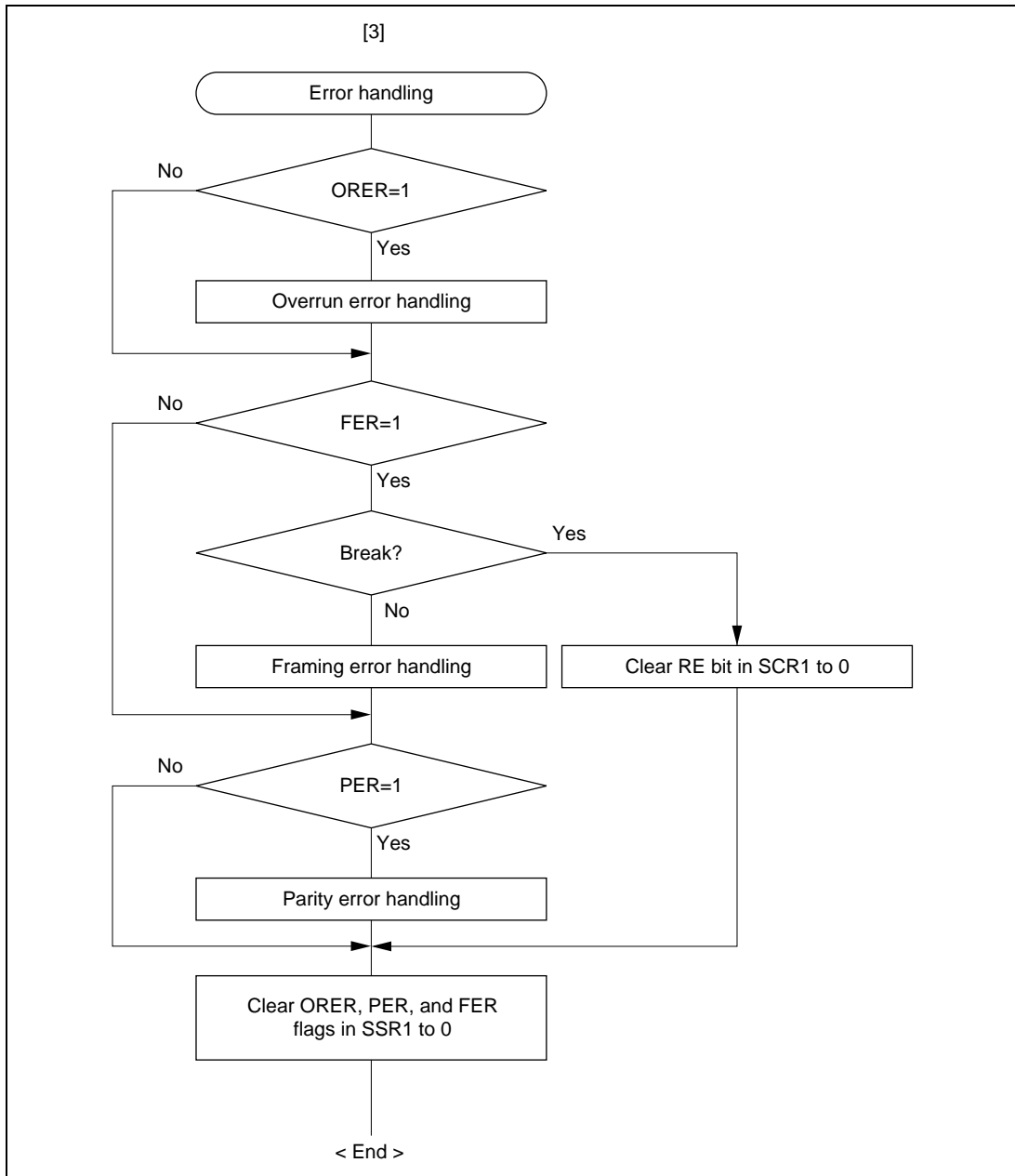


Figure 23.7 Sample Serial Reception Data Flowchart (2)

In serial reception, the SCI1 operates as described below.

- [1] The SCI1 monitors the transmission line, and if a 0 stop bit is detected, performs internal synchronization and starts reception.
- [2] The received data is stored in RSR in LSB-to-MSB order.
- [3] The parity bit and stop bit are received.

After receiving these bits, the SCI1 carries out the following checks.

[a] Parity check:

The SCI1 checks whether the number of 1 bits in the receive data agrees with the parity (even or odd) set in the O/ \bar{E} bit in SMR1.

[b] Stop bit check:

The SCI1 checks whether the stop bit is 1.

If there are two stop bits, only the first is checked.

[c] Status check:

The SCI1 checks whether the RDRF flag is 0, indicating that the receive data can be transferred from RSR to RDR1.

If all the above checks are passed, the RDRF flag is set to 1, and the receive data is stored in RDR1.

If a receive error* is detected in the error check, the operation is as shown in table 23.11.

Note: * Subsequent receive operations cannot be performed when a receive error has occurred.

Also note that the RDRF flag is not set to 1 in reception, and so the error flags must be cleared to 0.

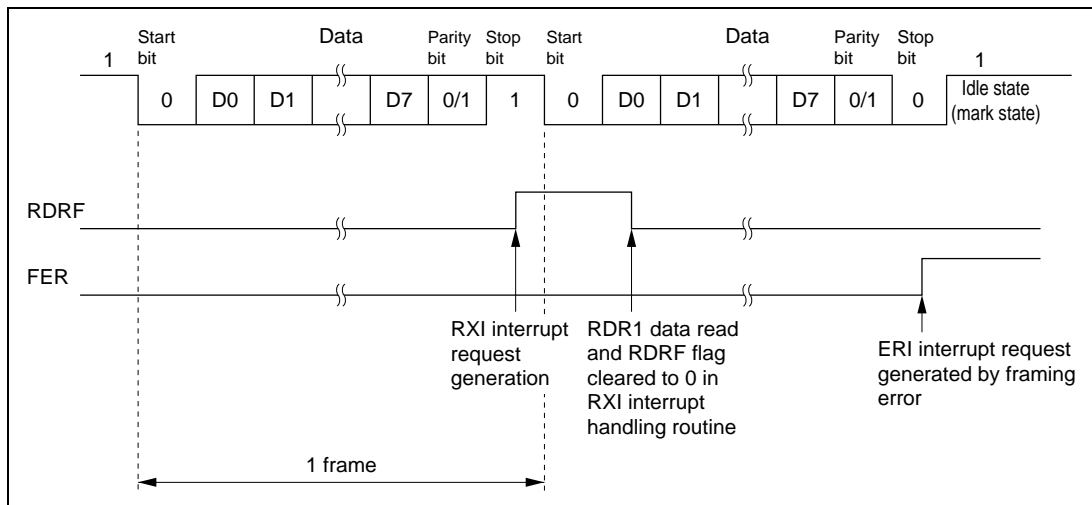
- [4] If the RIE bit in SCR1 is set to 1 when the RDRF flag changes to 1, a receive-data-full interrupt (RXI) request is generated.

Also, if the RIE bit in SCR1 is set to 1 when the ORER, PER, or FER flag changes to 1, a receive-error interrupt (ERI) request is generated.

Table 23.11 Receive Errors and Conditions for Occurrence

| Receive Error | Abbrev. | Occurrence Condition | Data Transfer |
|---------------|---------|---|--|
| Overrun error | ORER | When the next data reception is completed while the RDRF flag in SSR1 is set to 1 | Receive data is not transferred from RSR to RDR1 |
| Framing error | FER | When the stop bit is 0 | Receive data is transferred from RSR to RDR1 |
| Parity error | PER | When the received data differs from the parity (even or odd) set in SMR1 | Receive data is transferred from RSR to RDR1 |

Figure 23.8 shows an example of the operation for reception in asynchronous mode.



**Figure 23.8 Example of SCI1 Operation in Reception
(Example with 8-Bit Data, Parity, One Stop Bit)**

23.3.3 Multiprocessor Communication Function

The multiprocessor communication function performs serial communication using a multiprocessor format, in which a multiprocessor bit is added to the transfer data, in asynchronous mode. Use of this function enables data transfer to be performed among a number of processors sharing transmission lines.

When multiprocessor communication is carried out, each receiving station is addressed by a unique ID code.

The serial communication cycle consists of two component cycles: an ID transmission cycle which specifies the receiving station, and a data transmission cycle. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle.

The transmitting station first sends the ID of the receiving station with which it wants to perform serial communication as data with a 1 multiprocessor bit added. It then sends transmit data as data with a 0 multiprocessor bit added.

The receiving station skips the data until data with a 1 multiprocessor bit is sent.

When data with a 1 multiprocessor bit is received, the receiving station compares that data with its own ID. The station whose ID matches then receives the data sent next. Stations whose ID does not match continue to skip the data until data with a 1 multiprocessor bit is again received. In this way, data communication is carried out among a number of processors.

Figure 23.9 shows an example of inter-processor communication using a multiprocessor format.

(1) Data Transfer Format

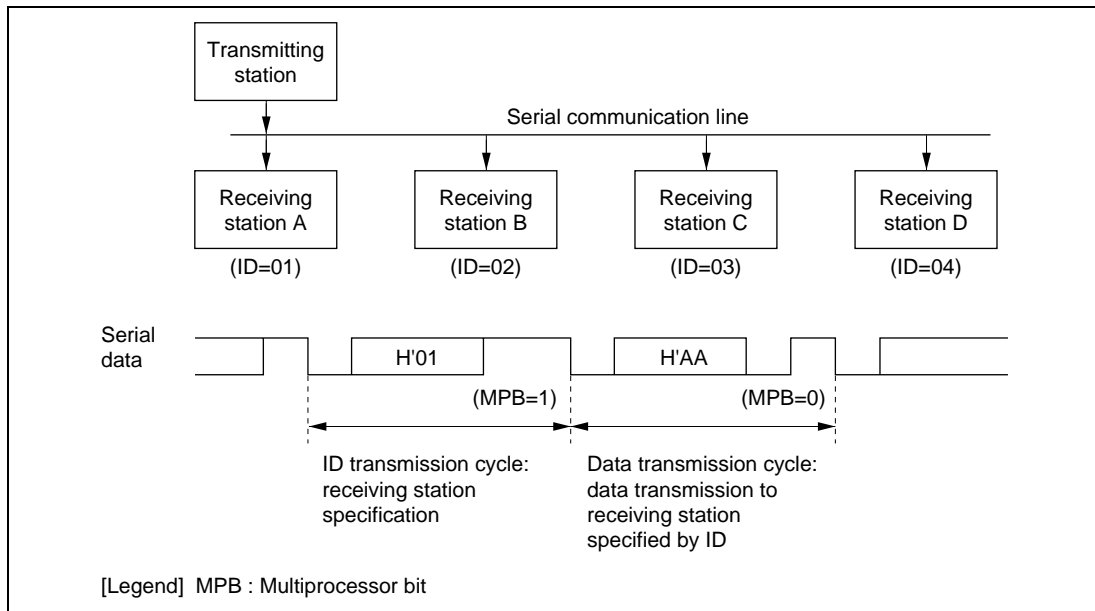
There are four data transfer formats.

When a multiprocessor format is specified, the parity bit specification is invalid.

For details, see table 23.10.

(2) Clock

See the section on asynchronous mode.



**Figure 23.9 Example of Inter-Processor Communication Using Multiprocessor Format
(Transmission of Data H'AA to Receiving Station A)**

(3) Data Transfer Operations

(a) Multiprocessor Serial Data Transmission

Figure 23.10 shows a sample flowchart for multiprocessor serial data transmission.

The following procedure should be used for multiprocessor serial data transmission.

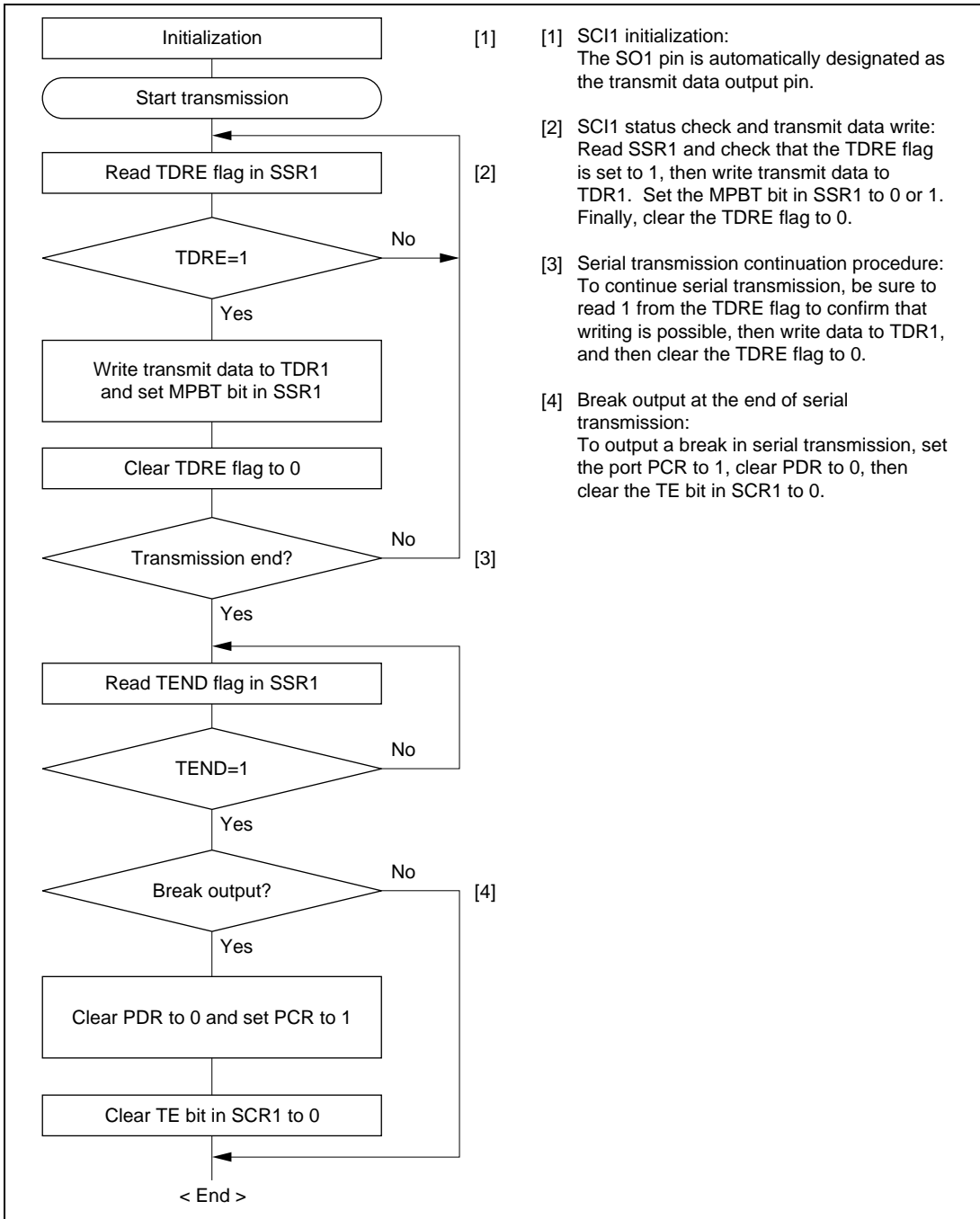


Figure 23.10 Sample Multiprocessor Serial Transmission Flowchart

In serial transmission, the SCI1 operates as described below.

- [1] The SCI1 monitors the TDRE flag in SSR1, and if it is 0, recognizes that data has been written to TDR1, and transfers the data from TDR1 to TSR.
- [2] After transferring data from TDR1 to TSR, the SCI1 sets the TDRE flag to 1 and starts transmission.

If the TIE bit is set to 1 at this time, a transmit-data-empty interrupt (TXI) is generated. The serial transmit data is sent from the SO2 pin in the following order.

[a] Start bit:

One 0-bit is output.

[b] Transmit data:

8-bit or 7-bit data is output in LSB-first order.

[c] Multiprocessor bit

One multiprocessor bit (MPBT value) is output.

[d] Stop bit(s):

One or two 1-bits (stop bits) are output.

[e] Mark state:

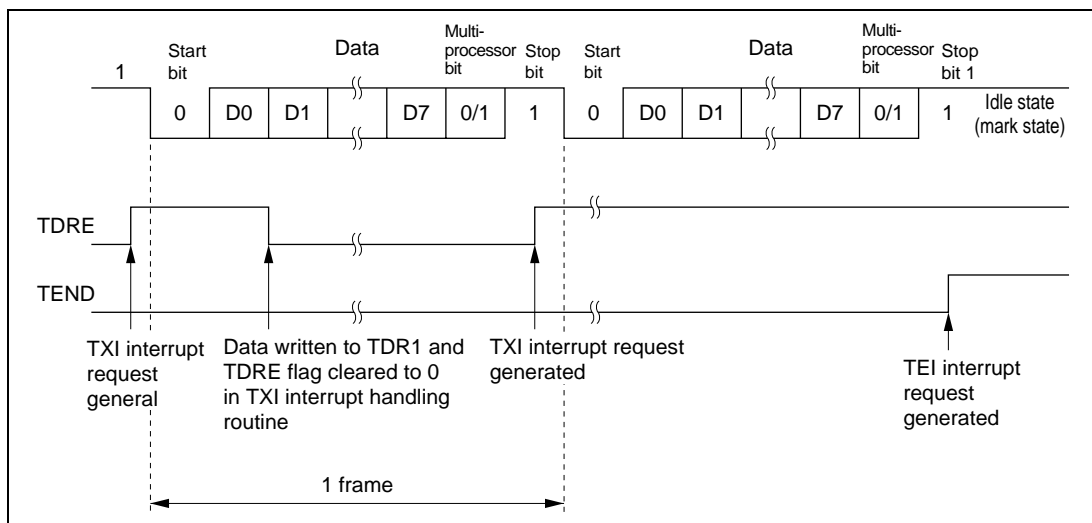
1 is output continuously until the start bit that starts the next transmission is sent.

- [3] The SCI1 checks the TDRE flag at the timing for sending the stop bit.

If the TDRE flag is cleared to 0, data is transferred from TDR1 to TSR, the stop bit is sent, and then serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR1 is set to 1, the stop bit is sent, and then the mark state is entered in which 1 is output continuously. If the TEIE bit in SCR1 is set to 1 at this time, a transmit-end interrupt (TEI) request is generated.

Figure 23.11 shows an example of SCI1 operation for transmission using a multiprocessor format.



**Figure 23.11 Example of SCI1 Operation in Transmission
(Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)**

(b) Multiprocessor Serial Data Reception

Figure 23.12 shows a sample flowchart for multiprocessor serial reception.

The following procedure should be used for multiprocessor serial data reception.

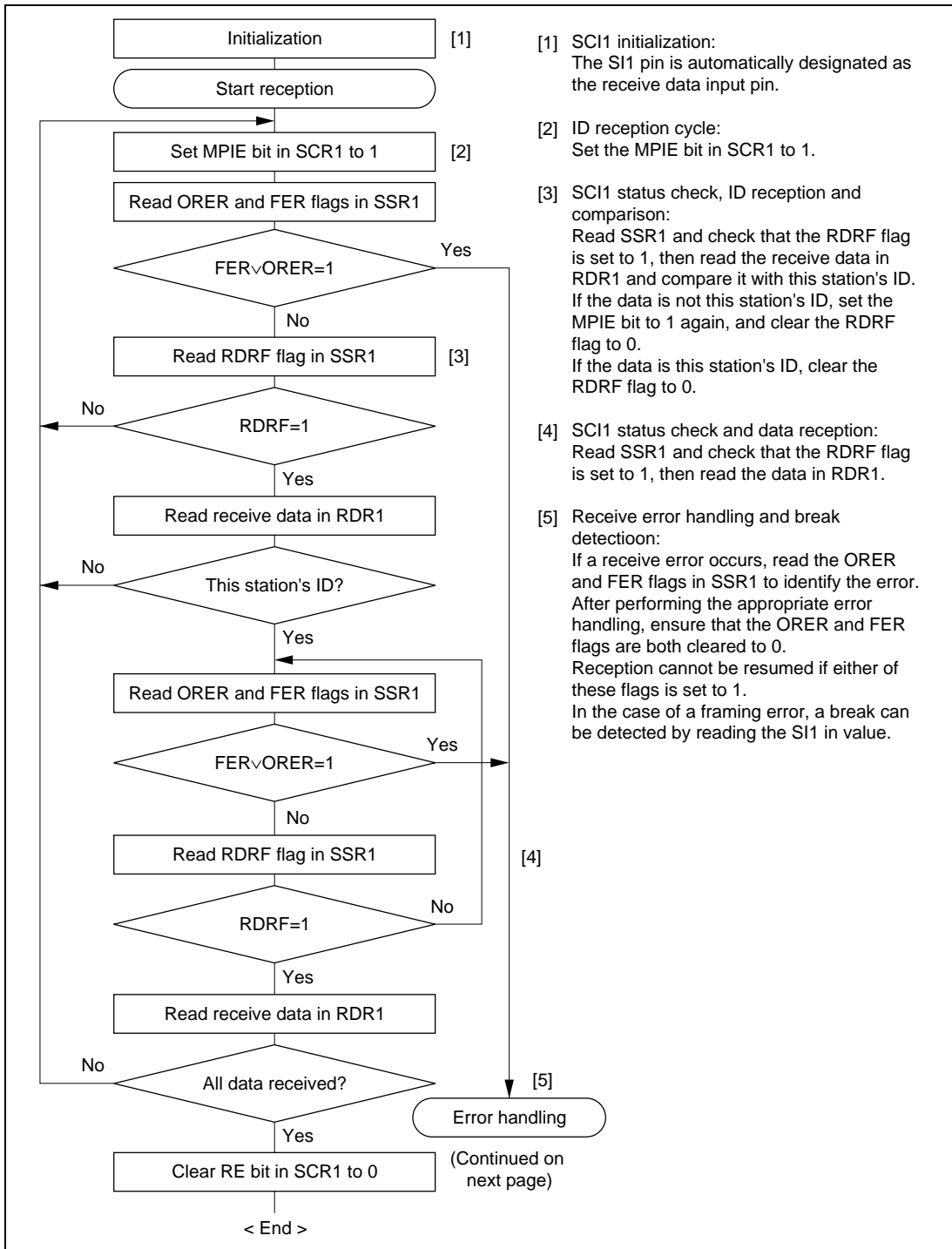


Figure 23.12 Sample Multiprocessor Serial Reception Flowchart (1)

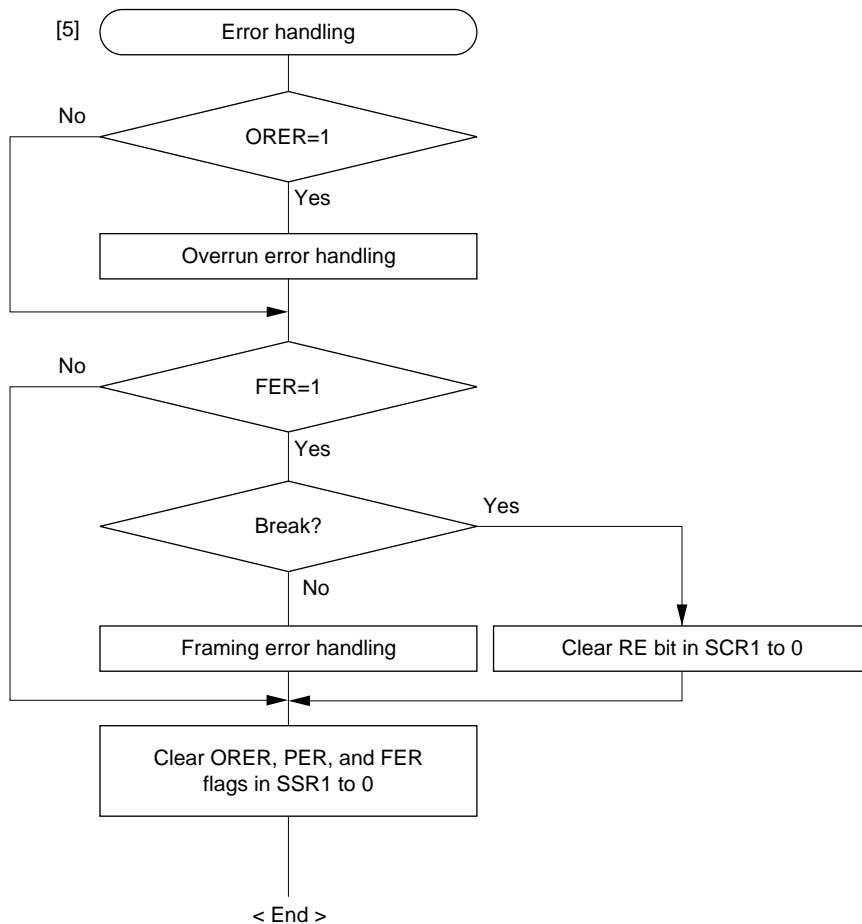
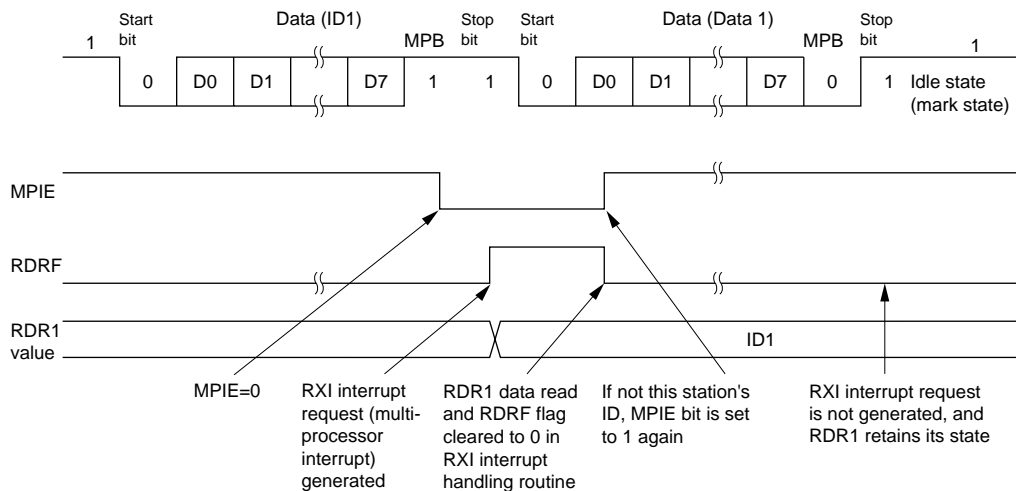
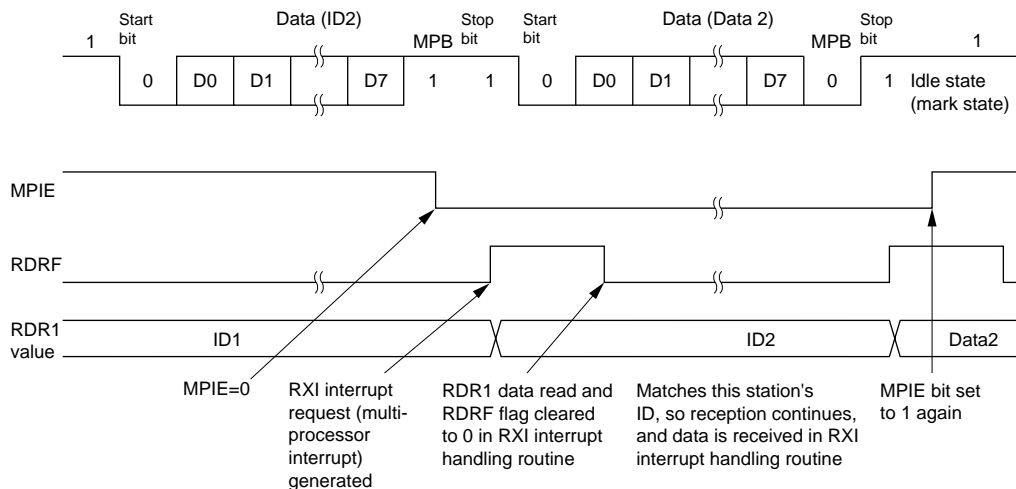


Figure 23.12 Sample Multiprocessor Serial Reception Flowchart (2)

Figure 23.13 shows an example of SCI1 operation for multiprocessor format reception.



(a) Data does not match station's ID



(b) Data matches station's ID

Figure 23.13 Example of SCI Operation in Reception
(Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)

23.3.4 Operation in Clock Synchronous Mode

In clock synchronous mode, data is transmitted or received in synchronization with clock pulses, making it suitable for high-speed serial communication.

Inside the SCI1, the transmitter and receiver are independent units, enabling full-duplex communication by use of a common clock. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 23.14 shows the general format for clock synchronous serial communication.

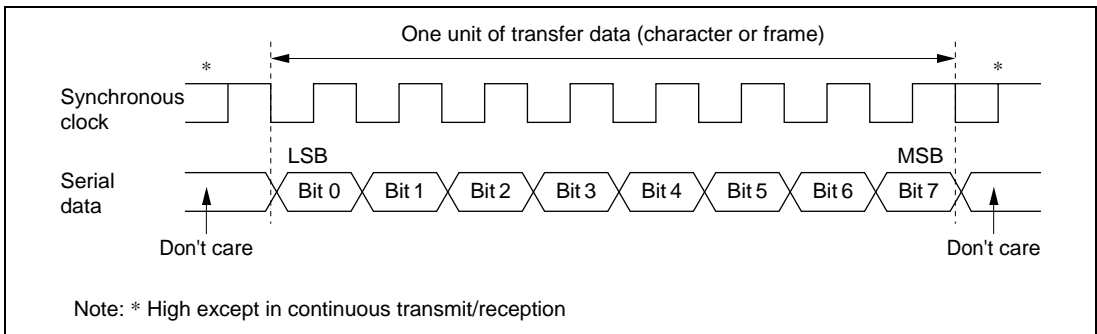


Figure 23.14 Data Format in Clock Synchronous Communication

In clock synchronous serial communication, data on the transmission line is output from one falling edge of the serial clock to the next. Data is guaranteed valid at the rising edge of the serial clock.

In clock synchronous serial communication, one character consists of data output starting with the LSB and ending with the MSB. After the MSB is output, the transmission line holds the MSB state.

In clock synchronous mode, the SCI1 receives data in synchronization with the rising edge of the serial clock.

(1) Data Transfer Format

A fixed 8-bit data format is used.

No parity or multiprocessor bits are added.

(2) Clock

Either an internal clock generated by the built-in baud rate generator or an external serial clock input at the SCK1 pin can be selected, according to the setting of the C/\bar{A} bit in SMR1 and the CKE1 and CKE0 bits in SCR1. For details on SCI clock source selection, see table 23.9.

When the SCI1 is operated on an internal clock, the serial clock is output from the SCK1 pin.

Eight serial clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high. When only receive operations are performed, however, the serial clock is output until an overrun error occurs or the RE bit is cleared to 0. To perform receive operations in units of one character, select an external clock as the clock source.

(3) Data Transfer Operations

(a) SCI1 Initialization (Synchronous Mode)

Before transmitting and receiving data, first clear the TE and RE bits in SCR1 to 0, then initialize the SCI1 as described below.

When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and TSR is initialized. Note that clearing the RE bit to 0 does not change the settings of the RDRF, PER, FER, and ORER flags, or the contents of RDR1.

Figure 23.15 shows a sample SCI1 initialization flowchart.

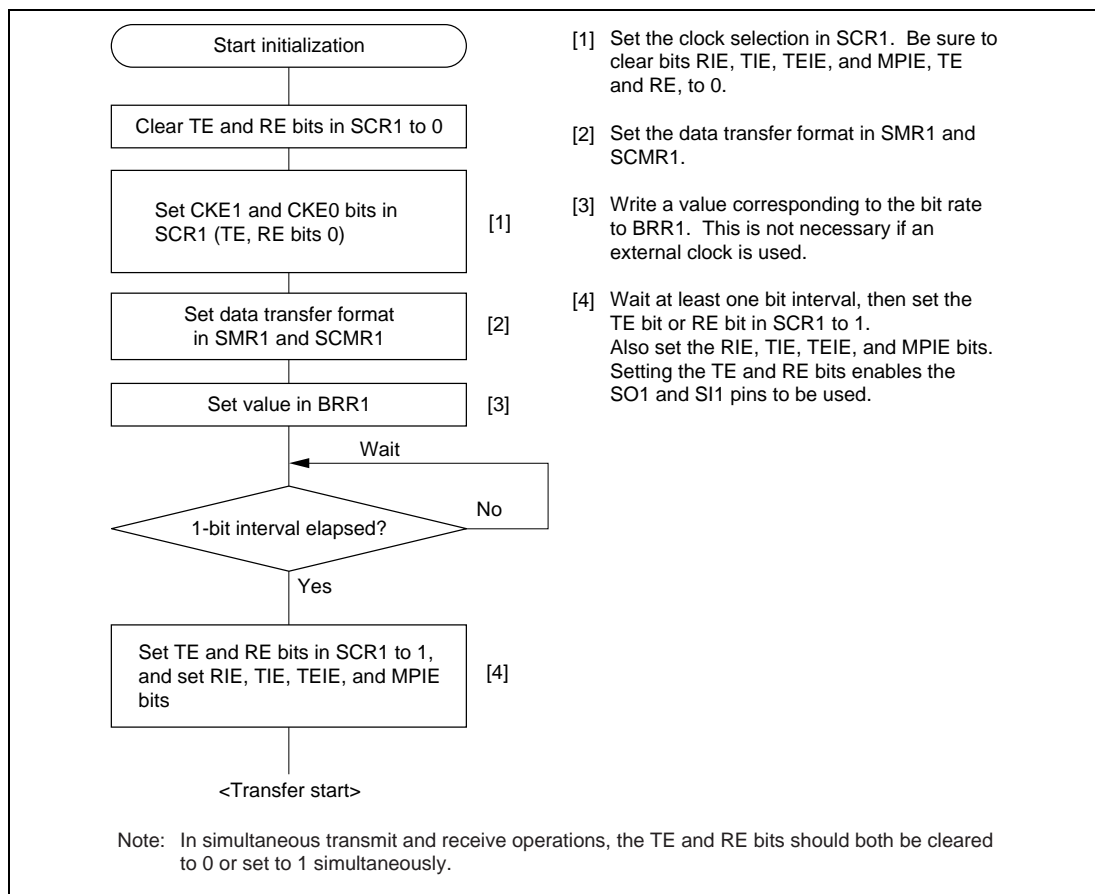


Figure 23.15 Sample SCI Initialization Flowchart

(b) Serial Data Transmission (Clock Synchronous Mode)

Figure 23.16 shows a sample flowchart for serial transmission.

The following procedure should be used for serial data transmission.

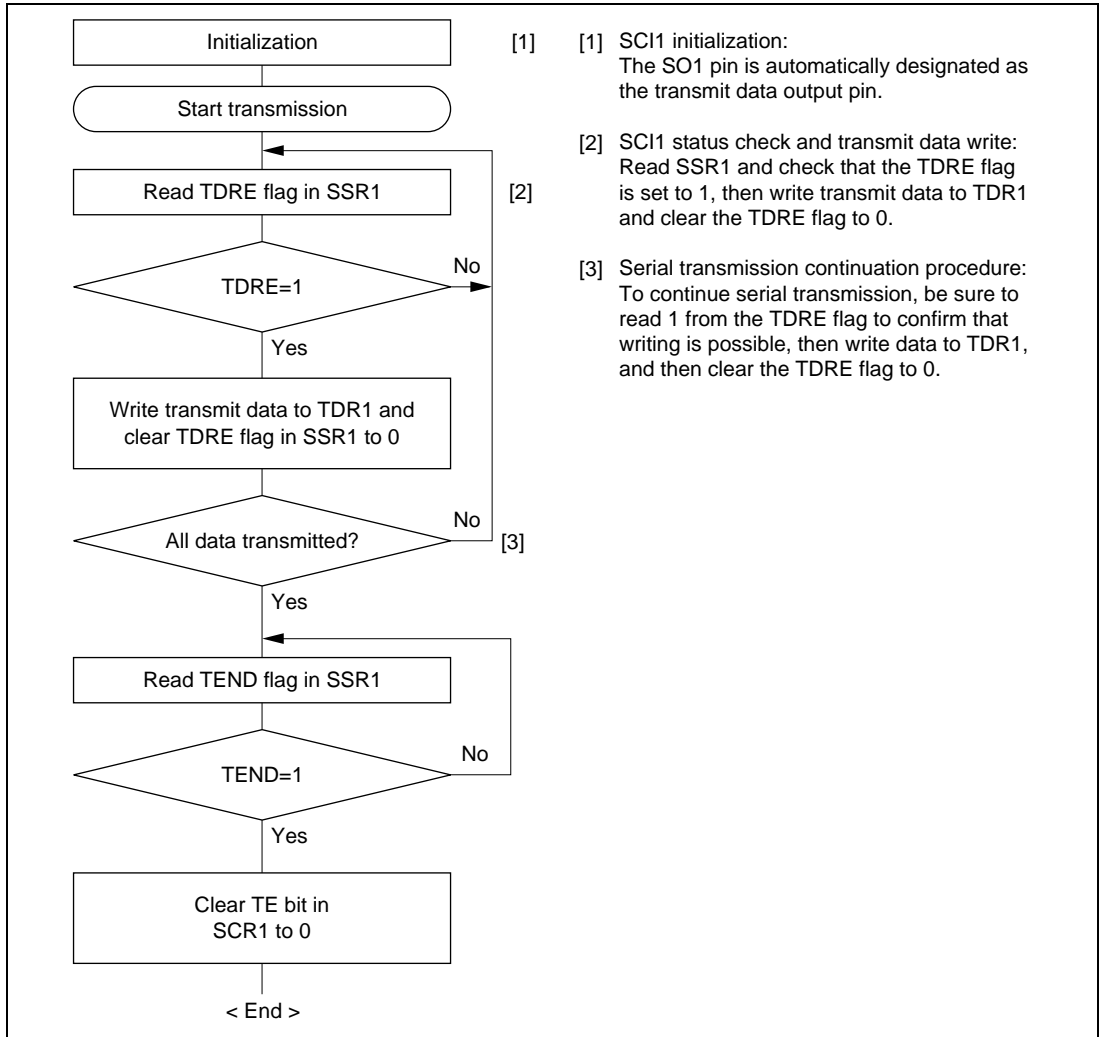


Figure 23.16 Sample Serial Transmission Flowchart

In serial transmission, the SCI1 operates as described below.

- [1] The SCI1 monitors the TDRE flag in SSR1, and if it is 0, recognizes that data has been written to TDR1, and transfers the data from TDR1 to TSR.
- [2] After transferring data from TDR1 to TSR, the SCI1 sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a transmit-data-empty interrupt (TXI) is generated.

When clock output mode has been set, the SCI1 outputs 8 serial clock pulses. When use of an external clock has been specified, data is output synchronized with the input clock.

The serial transmit data is sent from the SO1 pin starting with the LSB (bit 0) and ending with the MSB (bit 7).

- [3] The SCI1 checks the TDRE flag at the timing for sending the MSB (bit 7).

If the TDRE flag is cleared to 0, data is transferred from TDR1 to TSR, and serial transmission of the next frame is started.

If the TDRE flag is set to 1, the TEND flag in SSR1 is set to 1, the MSB (bit 7) is sent, and the SO1 pin maintains its state.

If the TEIE bit in SCR1 is set to 1 at this time, a transmit-end interrupt (TEI) request is generated.

- [4] After completion of serial transmission, the SCK1 pin is held in a constant state.

Figure 23.17 shows an example of SCI1 operation in transmission.

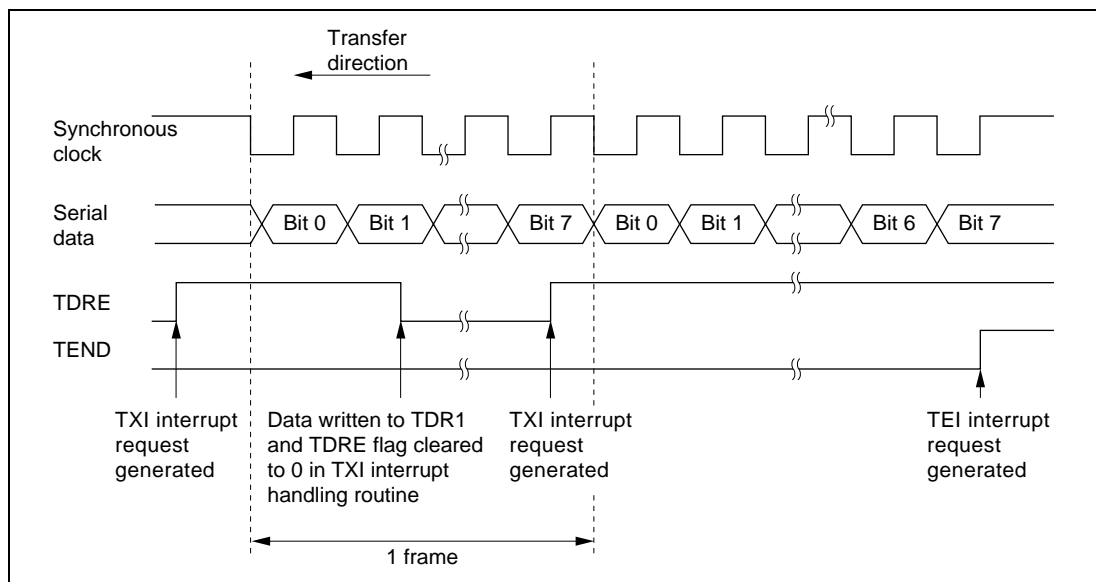


Figure 23.17 Example of SCI1 Operation in Transmission

(c) Serial Data Reception (Clock Synchronous Mode)

Figure 23.18 shows a sample flowchart for serial reception.

The following procedure should be used for serial data reception.

When changing the operating mode from asynchronous to synchronous, be sure to check that the ORER, PER, and FER flags are all cleared to 0.

The RDRF flag will not be set if the FER or PER flag is set to 1, and neither transmit nor receive operations will be possible.

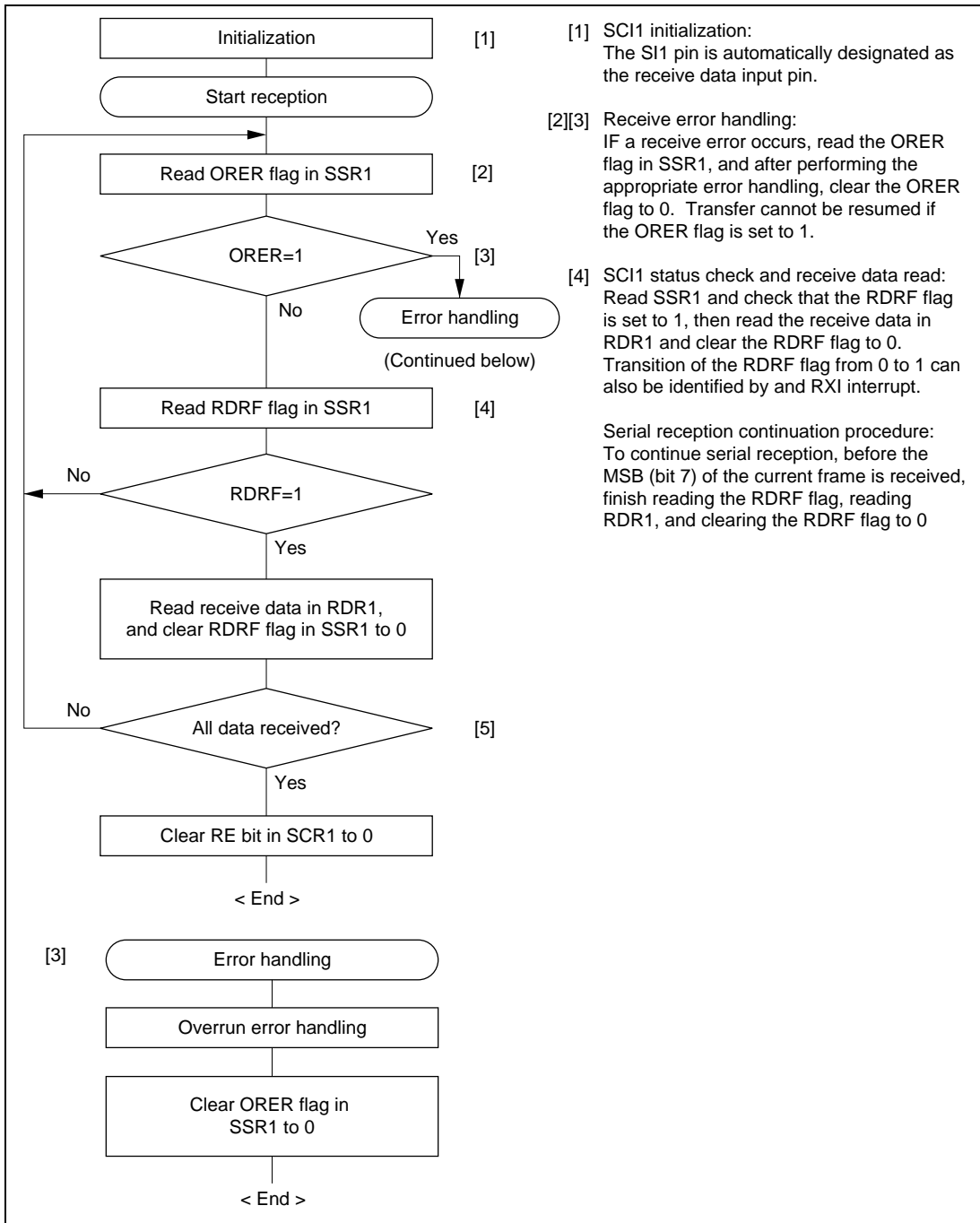


Figure 23.18 Sample Serial Reception Flowchart

In serial reception, the SCI1 operates as described below.

[1] The SCI1 performs internal initialization in synchronization with serial clock input or output.

[2] The received data is stored in RSR in LSB-to-MSB order.

After reception, the SCI1 checks whether the RDRF flag is 0 and the receive data can be transferred from RSR to RDR1.

If this check is passed, the RDRF flag is set to 1, and the receive data is stored in RDR1. If a receive error is detected in the error check, the operation is as shown in table 23.11.

Neither transmit nor receive operations can be performed subsequently when a receive error has been found in the error check.

[3] If the RIE bit in SCR1 is set to 1 when the RDRF flag changes to 1, a receive-data-full interrupt (RXI) request is generated.

Also, if the RIE bit in SCR1 is set to 1 when the ORER flag changes to 1, a receive-error interrupt (ERI) request is generated.

Figure 23.19 shows an example of SCI1 operation in reception.

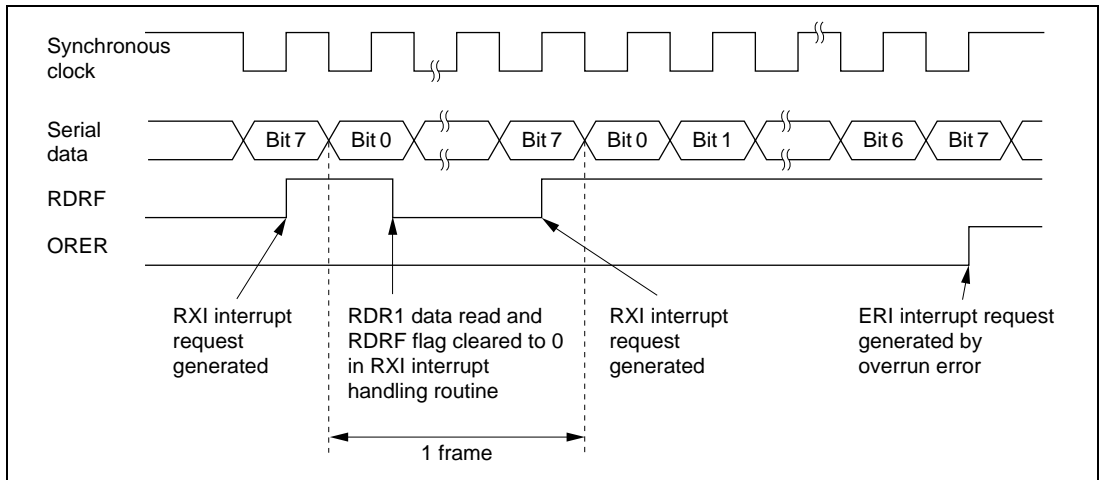
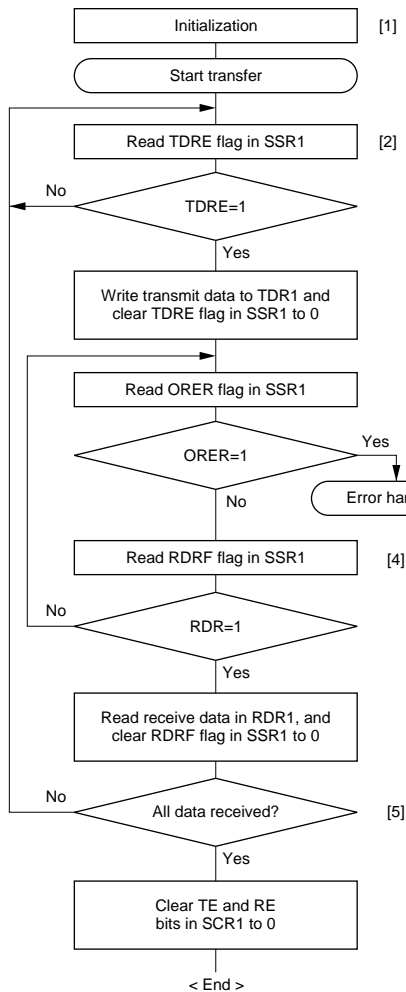


Figure 23.19 Example of SCI1 Operation in Reception

(d) Simultaneous Serial Data Transmission and Reception (Clock Synchronous Mode)

Figure 23.20 shows a sample flowchart for simultaneous serial transmit and receive operations.

The following procedure should be used for simultaneous serial data transmit and receive operations.



[1] SCI1 initialization:
The SO1 pin is designated as the transmit data output pin, and the SI1 pin is designated as the receive data input pin, enabling simultaneous transmit and receive operations.

[2] SCI1 status check and transmit data write:
Read SSR1 and check that the TDRE flag is set to 1, then write transmit data to TDR1 and clear the TDRE flag to 0. Transition of the TDRE flag from 0 to 1 can also be identified by a TXI interrupt.

[3] Receive error handling:
If a receive error occurs, read the ORER flag in SSR1, and after performing the appropriate error handling, clear the ORER flag to 0. Transmission/reception cannot be resumed if the ORER flag is set to 1.

[4] SCI1 status check and receive data read:
Read SSR1 and check that the RDRF flag is set to 1, then read the receive data in RDR1 and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial transmission/reception continuation procedure:
To continue serial transmission/reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR1, and clearing the RDRF flag to 0. Also before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR1 and clear the TDRE flag to 0.


Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE bit and RE bit to 0, then set both these bits to 1 simultaneously.

Figure 23.20 Sample Flowchart of Simultaneous Serial Transmit and Receive Operations

23.4 SCI1 Interrupts

The SCI1 has four interrupt sources: the transmit-end interrupt (TEI) request, receive-error interrupt (ERI) request, receive-data-full interrupt (RXI) request, and transmit-data-empty interrupt (TXI) request. Table 23.13 shows the interrupt sources and their relative priorities. Individual interrupt sources can be enabled or disabled with the TIE, RIE, and TEIE bits in SCR1. Each kind of interrupt request is sent to the interrupt controller independently. When the TDRE flag in SSR1 is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR1 is set to 1, a TEI interrupt request is generated. When the RDRF flag in SSR1 is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR1 is set to 1, an ERI interrupt request is generated.

Table 23.13 SCI1 Interrupt Sources

| Channel | Interrupt Source | Description | Priority* |
|---------|------------------|--|--|
| 1 | ERI | Interrupt by receive error (ORER, FER, or PER) | High |
| | RXI | Interrupt by receive data register full (RDRF) |  |
| | TXI | Interrupt by transmit data register empty (TDRE) | |
| | TEI | Interrupt by transmit end (TEND) | Low |

The TEI interrupt is requested when the TEND flag is set to 1 while the TEIE bit is set to 1. The TEND flag is cleared at the same time as the TDRE flag. Consequently, if a TEI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt will have priority for acceptance, and the TDRE flag and TEND flag may be cleared. Note that the TEI interrupt will not be accepted in this case.

23.5 Usage Notes

The following points should be noted when using the SCI1.

(1) Relation between Writes to TDR1 and the TDRE Flag

The TDRE flag in SSR1 is a status flag that indicates that transmit data has been transferred from TDR1 to TSR. When the SCI transfers data from TDR1 to TSR, the TDRE flag is set to 1.

Data can be written to TDR1 regardless of the state of the TDRE flag. However, if new data is written to TDR1 when the TDRE flag is cleared to 0, the data stored in TDR1 will be lost since it has not yet been transferred to TSR. It is therefore essential to check that the TDRE flag is set to 1 before writing transmit data to TDR1.

(2) Operation when Multiple Receive Errors Occur Simultaneously

If a number of receive errors occur at the same time, the state of the status flags in SSR1 is as shown in table 23.14. If there is an overrun error, data is not transferred from RSR to RDR1, and the receive data is lost.

Table 23.14 State of SSR1 Status Flags and Transfer of Receive Data

| SSR1 Status Flags | | | | Receive Data Transfer RSR → RDR1 | Receive Errors |
|-------------------|------|-----|-----|-------------------------------------|--|
| RDRF | ORER | FER | PER | | |
| 1 | 1 | 0 | 0 | × | Overrun error |
| 0 | 0 | 1 | 0 | ○ | Framing error |
| 0 | 0 | 0 | 1 | ○ | Parity error |
| 1 | 1 | 1 | 0 | × | Overrun error + framing error |
| 1 | 1 | 0 | 1 | × | Overrun error + parity error |
| 0 | 0 | 1 | 1 | ○ | Framing error + parity error |
| 1 | 1 | 1 | 1 | × | Overrun error + framing error + parity error |

Notes: ○: Receive data is transferred from RSR to RDR1.

×: Receive data is not transferred from RSR to RDR1.

(3) Break Detection and Processing

When a framing error (FER) is detected, a break can be detected by reading the SI1 pin value directly. In a break, the input from the SI1 pin becomes all 0s, and so the FER flag is set, and the parity error flag (PER) may also be set.

Note that, since the SCI1 continues the receive operation after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

(4) Sending a Break

The SO1 pin has a dual function as an I/O port whose direction (input or output) is determined by PDR and PCR. This feature can be used to send a break.

Between serial transmission initialization and setting of the TE bit to 1, the mark state is replaced by the value of PDR (the pin does not function as the SO1 pin until the TE bit is set to 1). Consequently, PCR and PDR for the port corresponding to the SO1 pin should first be set to 1.

To send a break during serial transmission, first clear PDR to 0, then clear the TE bit to 0.

When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, the SO1 pin becomes an I/O port, and 0 is output from the SO1 pin.

(5) Receive Error Flags and Transmit Operations (Clock Synchronous Mode Only)

Transmission cannot be started when a receive error flag (ORER, PER, or FER) is set to 1, even if the TDRE flag is cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission.

Note also that receive error flags cannot be cleared to 0 even if the RE bit is cleared to 0.

(6) Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI1 operates on a base clock with a frequency of 16 times the transfer rate.

In reception, the SCI1 samples the falling edge of the start bit using the base clock, and performs internal synchronization. Receive data is latched internally at the rising edge of the 8th pulse of the base clock. This is illustrated in figure 23.21.

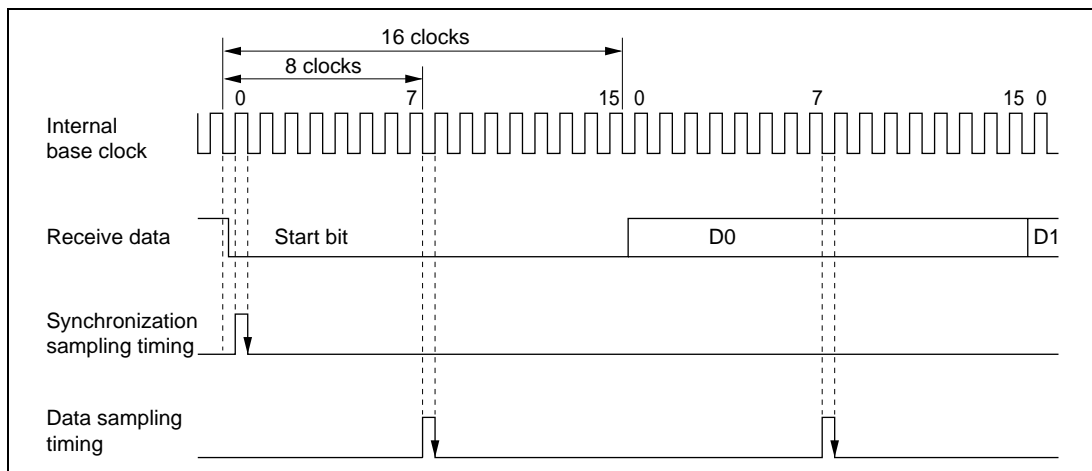


Figure 23.21 Receive Data Sampling Timing in Asynchronous Mode

Thus the receive margin in asynchronous mode is given by equation (1) below.

$$M = \left| \left(0.5 - \frac{1}{2N} \right) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \right| \times 100\% \quad \dots(1)$$

Where M: Receive margin (%)

N: Ratio of bit rate to clock (N = 16)

D: Clock duty (D = 0 to 1.0)

L: Frame length (L = 9 to 12)

F: Absolute value of clock rate deviation

Assuming values of F = 0 and D = 0.5 in equation (1), a receive margin of 46.875% is given by equation (2) below.

When D = 0.5 and F = 0,

$$\begin{aligned} M &= \left(0.5 - \frac{1}{2 \times 16} \right) \times 100\% \\ &= 46.875\% \end{aligned} \quad \dots(2)$$

However, this is only a theoretical value, and a margin of 20% to 30% should be allowed in system design.

Section 24 Serial Communication Interface 2 (SCI2)

24.1 Overview

The serial communication interface 2 (SCI2) that has a 32-byte data buffer carries out clocked synchronous serial transmission of 32 bytes by a single operation.

24.1.1 Features

SCI2 features are listed below.

- 32 bytes data transfer can be automatically carried out
- Choice of 7 internal clocks ($\phi/256$, $\phi/64$, $\phi/32$, $\phi/16$, $\phi/8$, $\phi/4$, and $\phi/2$) and an external clock as serial clock source
- Interrupt occurs when transmission has been completed or an error has occurred
- Data transfer at intervals of 1 byte can be set
Data transfer can be carried out at intervals of 1 byte. The interval can be selected from a multiple of internal clock cycle by 56, 24, or 8 times
- Start of data transfer can be controlled by input of chip select
- Strobe pulse is output for each 1-byte transfer

24.1.2 Block Diagram

Figure 24.1 shows a block diagram of the SCI2.

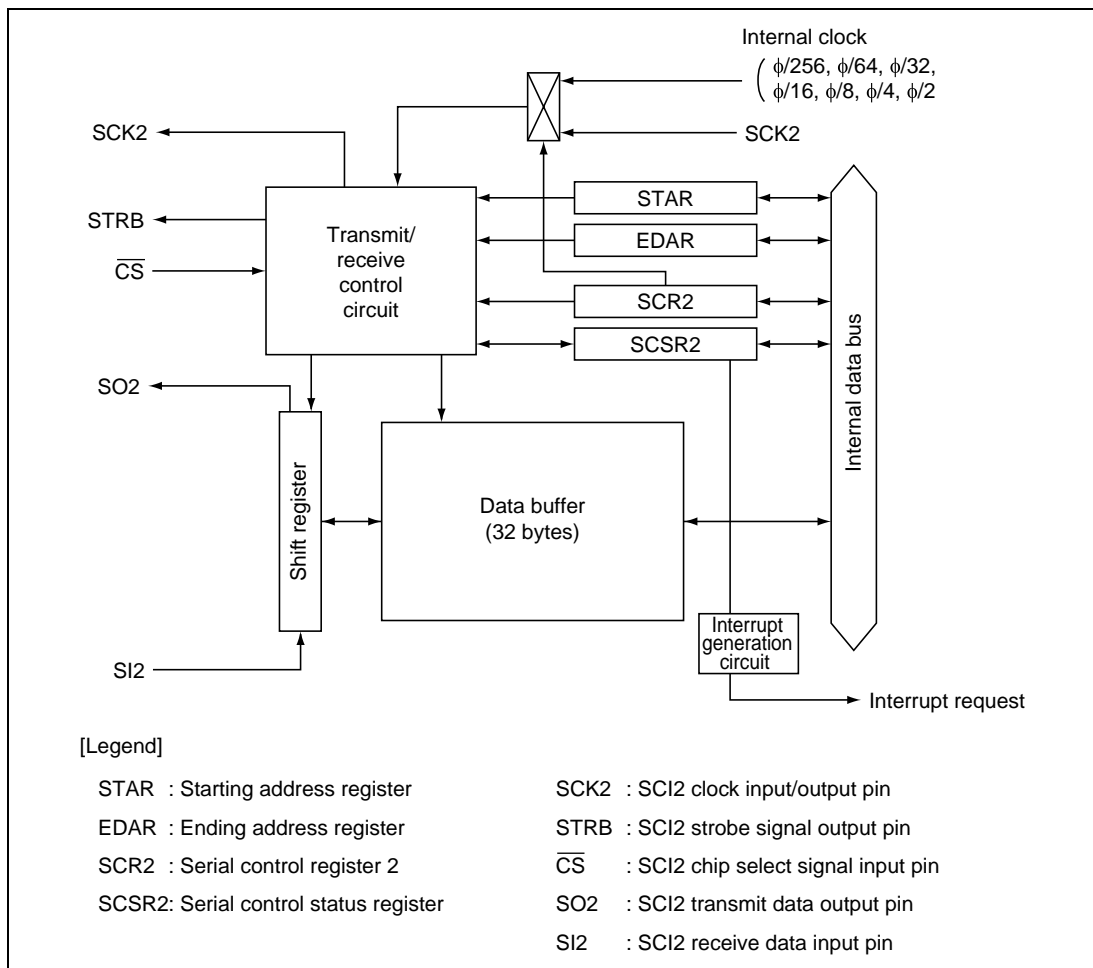


Figure 24.1 Block Diagram of SCI2

24.1.3 Pin Configuration

Table 24.1 shows pin configuration of the SCI2.

Table 24.1 Pin Configuration

| Name | Abbrev. | I/O | Function |
|------------------|------------------------|--------|-----------------------------------|
| SCI2 Clock | SCK2 | I/O | SCI2 clock input/output pin |
| SCI2 Data input | SI2 | Input | SCI2 receive data input pin |
| SCI2 Data output | SO2 | Output | SCI2 transmit data output pin |
| SCI2 Strobe | STRB | Output | SCI2 strobe signal output pin |
| SCI2 Chip select | $\overline{\text{CS}}$ | Input | SCI2 chip select signal input pin |

24.1.4 Register Configuration

Table 24.2 shows register configuration of the SCI2.

Table 24.2 Register Configuration

| Name | Abbrev. | R/W | Initial Value | Address* |
|----------------------------------|---------|-----|---------------|------------------|
| Starting address register | STAR | R/W | H'E0 | H'D0E0 |
| Ending address register | EDAR | R/W | H'E0 | H'D0E1 |
| Serial control register 2 | SCR2 | R/W | H'20 | H'D0E2 |
| Serial control status register 2 | SCSR2 | R/W | H'60 | H'D0E3 |
| Serial data buffer (32 bytes) | — | R/W | Undefined | H'D0C0 to H'D0DF |

Note: * Lower 16 bits of the address..

24.2 Register Descriptions

24.2.1 Starting Address Register (STAR)

| | | | | | | | | |
|-----------------|---|---|---|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | STA4 | STA3 | STA2 | STA1 | STA0 |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | R/W | R/W | R/W | R/W | R/W |

The STAR is a readable/writable register that specifies the transfer starting address within the address space (H'FFD0C0 to H'FFD0DF) to which a 32-byte data buffer is assigned. The 5 low-order bits of the STAR correspond to the 5 low-order bits of the address of 32-byte buffer. The range for executing continuous data transfer on STAR and EDAR is specified. When the value of STAR is equal to that of EDAR, only one-byte transfer is carried out.

Since the 7 to 5 bits of the STAR are reserved, writes are disabled. When each bit is read, 1 is read at all times.

The STAR is initialized to H'E0 by a reset.

24.2.2 Ending Address Register (EDAR)

| | | | | | | | | |
|-----------------|---|---|---|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | EDA4 | EDA3 | EDA2 | EDA1 | EDA0 |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | R/W | R/W | R/W | R/W | R/W |

The EDAR is a readable/writable register that specifies the transfer ending address within the address space (H'FFD0C0 to H'FFD0DF) to which 32-byte data buffer is assigned. The 5 low-order bits of EDAR correspond to the 5 low-order bits of the address of 32-byte buffer. The range for executing continuous data transfer is specified by the EDAR and the STAR. If the value of the STAR is equal to that of the EDAR, only one-byte transfer is carried out.

Since the 7 to 5 bits of the EDAR are reserved, writes are disabled. When each bit is read, 1 is read at all times.

The EDAR is initialized to H'E0 by a reset.

24.2.3 Serial Control Register 2 (SCR2)

| | | | | | | | | |
|-----------------|------|-------|---|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TEIE | ABTIE | — | GAP1 | GAP0 | CKS2 | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | — | R/W | R/W | R/W | R/W | R/W |

The SCR 2 is a readable/writable register that enables or disables generation of SCI2 interrupt and selects an data transfer interval and transfer clock when an internal clock is used.

The SCR2 is initialized to H'20 by a reset.

Bit 7: Transmit End Interrupt Enable (TEIE)

Enables or disables the occurrence of transmit-end interrupt when data transfer has been completed and TEI of the SCR2 has been set to 1.

Bit 7

| TEIE | Description |
|------|---|
| 0 | Transmit-end interrupt disabled (Initial value) |
| 1 | Transmit-end interrupt enabled |

Bit 6: Transmit Cutoff Interrupt (ABTIE)

Enables or disables the occurrence of transmit-cutoff interrupt when the $\overline{\text{CS}}$ pin has entered a high level during transmission and ABT of the SCRS2 has been set to 1.

Bit 6

| ABTIE | Description |
|-------|--|
| 0 | Transmit-cutoff interrupt disabled (Initial value) |
| 1 | Transmit-cutoff interrupt enabled |

Bit 5: Reserved

When read, 1 is read at all times. Writes are disabled.

Bits 4 and 3: Transmit Data Interval Select 1 and 0 (GAP1, GAP0)

When an internal clock is used, data can be transmitted at 1-byte intervals. During that time, the SCK2 pin retains the high level. When data is transmitted without intervals, the STRB signal retains the low level.

| Bit 4 | Bit 3 | Description |
|-------|-------|---|
| GAP1 | GAP0 | |
| 0 | 0 | Data transmission without intervals (Initial value) |
| 0 | 1 | Data intervals: 8 clocks |
| 1 | 0 | Data intervals: 24 clocks |
| 1 | 1 | Data intervals: 56 clocks |

Bits 2 to 0: Transfer Clock Select 2 to 0 (CKS2 to CKS0)

Selects transfer clock.

| Bit 2 | Bit 1 | Bit 0 | SCK2 pin | Clock source | Prescaler division ratio | Transfer clock cycle | |
|-------|-------|-------|-------------|----------------|----------------------------|-------------------------|------------------------|
| CKS2 | CKS1 | CKS0 | | | | $\phi = 10 \text{ MHz}$ | $\phi = 5 \text{ MHz}$ |
| 0 | 0 | 0 | SCK2 output | Prescaler S | $\phi/256$ (Initial value) | 25.6 μs | 51.2 μs |
| 0 | 0 | 1 | | | $\phi/64$ | 6.4 μs | 12.8 μs |
| 0 | 1 | 0 | | | $\phi/32$ | 3.2 μs | 6.4 μs |
| 0 | 1 | 1 | | | $\phi/16$ | 1.6 μs | 3.2 μs |
| 1 | 0 | 0 | | | $\phi/8$ | 0.8 μs | 1.6 μs |
| 1 | 0 | 1 | | | $\phi/4$ | 0.4 μs | 0.8 μs |
| 1 | 1 | 0 | | | $\phi/2$ | — | 0.4 μs |
| 1 | 1 | 1 | SCK2 input | External clock | — | — | — |

24.2.4 Serial Control Status Register 2 (SCSR2)

| | | | | | | | | |
|-----------------|--------|---|---|-----|--------|--------|--------|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TEI | — | — | SOL | ORER | WT | ABT | STF |
| Initial value : | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | — | — | R/W | R/(W)* | R/(W)* | R/(W)* | R/W |

Note: * Only 0 can be written to clear the flag.

The SCSR2 is an 8-bit register that indicates the SCI2's state of operation and error. The SCSR2 is initialized to H'60 by a reset.

Bit 7: Transmit End Interrupt Request Flag (TEI)

Indicates that data transmission or reception has been completed.

Bit 7

| TEI | Description |
|-----|---|
| 0 | [Clearing condition] When 0 is written after reading 1 (Initial value) |
| 1 | [Setting condition] When transmission or reception has been completed |

Bits 6 and 5: Reserved

When each bit is read, 1 is read at all times. Writes are disabled.

Bit 4: Extension Data Bit (SOL)

The SOL sets the output level of the SO2 pin. When read, the output level of the SO2 pin is read. Output of the SO2 pin after completion of transmission retains the value of final bit of transfer data, but the output level of the SO2 pin can be changed by operating this bit before or after transmission. However, setting of the SOL bit becomes invalid when the next transmission is started. Therefore, if the output level of the SO2 pin is changed after completion of transmission, write operation for SOL must be performed every time when transmission is terminated. Since writing to this register during data transfer may cause malfunction, write operation must not be performed during transmission.

Bit 4

| SOL | Description |
|-----|--|
| 0 | Read The SO2 pin output is at a low level (Initial value) |
| | Write The SO2 pin output is changed to a low level |
| 1 | Read The SO2 pin output is at a high level |
| | Write The SO2 pin output is changed to a high level |

Bit 3: Overrun Error Flag (ORER)

The ORER indicates an occurrence of overrun error while an external clock is used. When excessive pulses are overlapped with the normal transfer clock caused by external noise, etc. during transmission, this bit is set to 1. At this time data transfer cannot be assured. When a clock is input after completion of transmission, it is also found to be in the state of overrun and this bit is set to 1. However, overrun is not detected when the \overline{CS} pin is at a high level.

Bit 3

| ORER | Description |
|------|---|
| 0 | [Clearing condition] When 0 is written after reading 1 (Initial value) |
| 1 | [Setting condition] When excessive pulses are overlapped with a normal transfer clock while an external clock is used, or when a clock is input after completion of transmission |

Bit 2: Wait Flag (WT)

The WT indicates that read/ or write to serial data buffer (32 bytes) has been executed during transmission and in the \overline{CS} input standby mode. The instruction at that time is ignored and this bit is set to 1.

Bit 2

| WT | Description |
|----|--|
| 0 | [Clearing condition] When 0 is written after reading 1 (Initial value) |
| 1 | [Setting condition] When an instruction to read/write to serial data buffer (32 bits) is directed during transmission and in the \overline{CS} input standby mode |

Bit 1: Abort Flag (ABT)

The ABT indicates that the \overline{CS} pin has entered a high level during transmission. When a high level of the \overline{CS} pin is detected during transfer, the transfer is immediately cut off, and this bit is set to 1, and then the SCK2 and SO2 pins go into the high impedance state. At this time values of internal registers other than SCSR2 and serial data buffer (32 bytes) are retained. Transfer cannot be carried out while this bit is set to 0. Resume transfer after clearing to 0.

Bit 1

| ABT | Description |
|-----|--|
| 0 | [Clearing condition] When 0 is written after reading 1 (Initial value) |
| 1 | [Setting condition] During transfer and when \overline{CS} pin has entered a high level |

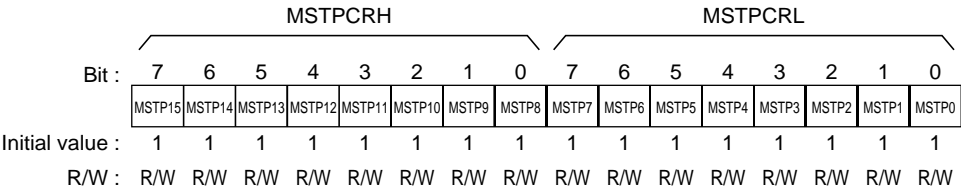
Bit 0: Start Flag (STF)

The STF controls the start of transfer operations. When this bit is set to 1 and PMR30 of PMR3 is 0, transfer operation of the SCI2 is started. When PMR30 of PMR3 is 1, the low level of the \overline{CS} pin is detected and transfer is started. This bit retains 1 during transfer and in the \overline{CS} input standby mode, and it is cleared to 0 after completion of transfer and when transfer is cut off by the \overline{CS} pin. Therefore, this bit can be used as a busy flag. When this bit is cleared to 0 during transfer, the transfer is cut off and the SCI2 is initialized. At this time the contents of internal registers other than the SCR2 and the serial data buffer (32 bytes) are retained.

Bit 0

| STF | Description | |
|-----|-------------|--|
| 0 | Read | Transfer operations stops (Initial value) |
| | Write | Transfer operation discontinues and the SCI2 is initialized |
| 1 | Read | During transfer operation or in \overline{CS} input standby mode |
| | Write | Transfer operation starts |

24.2.5 Module Stop Control Register (MSTPCR)



The MSTPCR, comprising two 8-bit readable/writable registers, performs module stop mode control. When the MSTPCR is set to 1, the SCI2 stops at the end of bus cycle and a transition is made to the module stop mode. For details, see section 4.5 Module Stop Mode. The MSTPCR is initialized to H'FFFF by a reset.

Bit 7: Module Stop (MSTP7)

Specifies the SCI2 module stop mode.

MSTPCRL

| Bit 7 | Description | |
|-------|--|--|
| MSTP7 | | |
| 0 | SCI2 module stop mode is cleared | |
| 1 | SCI2 module stop mode is set (Initial value) | |

24.3 Operation

The SCI2, comprising 32 bytes serial data buffer, can continuously transmit a maximum of 32 bytes data by a single operation, synchronized with clock pulse. Installation of a register enables to select transmit, receive, or simultaneous transmit/receive. When transmit is set, the value of serial data buffer is retained even after completion of transmission.

An internal or external clock can be selected as transfer clock. When an internal clock is selected, data can be transmitted at 1-byte intervals. The strobe signal can also be output from the STRB pin. When an external clock is selected, malfunction due to clock can be detected by the overrun flag.

The start of transfer and its forced cutoff can be controlled by \overline{CS} input. Forced cutoff can be detected by the abort flag.

24.3.1 Clock

Selection of a transfer clock can be made from seven internal clocks and an external clock. When an internal clock is selected, the SCK2 pin becomes a clock output pin.

24.3.2 Data Transfer Format

Figures 24.2 and 24.3 show transfer format of the SCI2.

LSB-first transfer that allows to transmit/receive from the lowest-order bit of data is performed. Transmit data is output from the fall of the transfer clock to its next fall. Receive data is collected at the rise of the transfer clock.

When an internal clock is selected as a transfer clock, data can be transferred at intervals of 1 byte. The SCK2 output is retained at a high level between transfer data. The strobe signal can be output from the STRB pin.

Selection of interval of transfer data is set at GAP1 or GAP0.

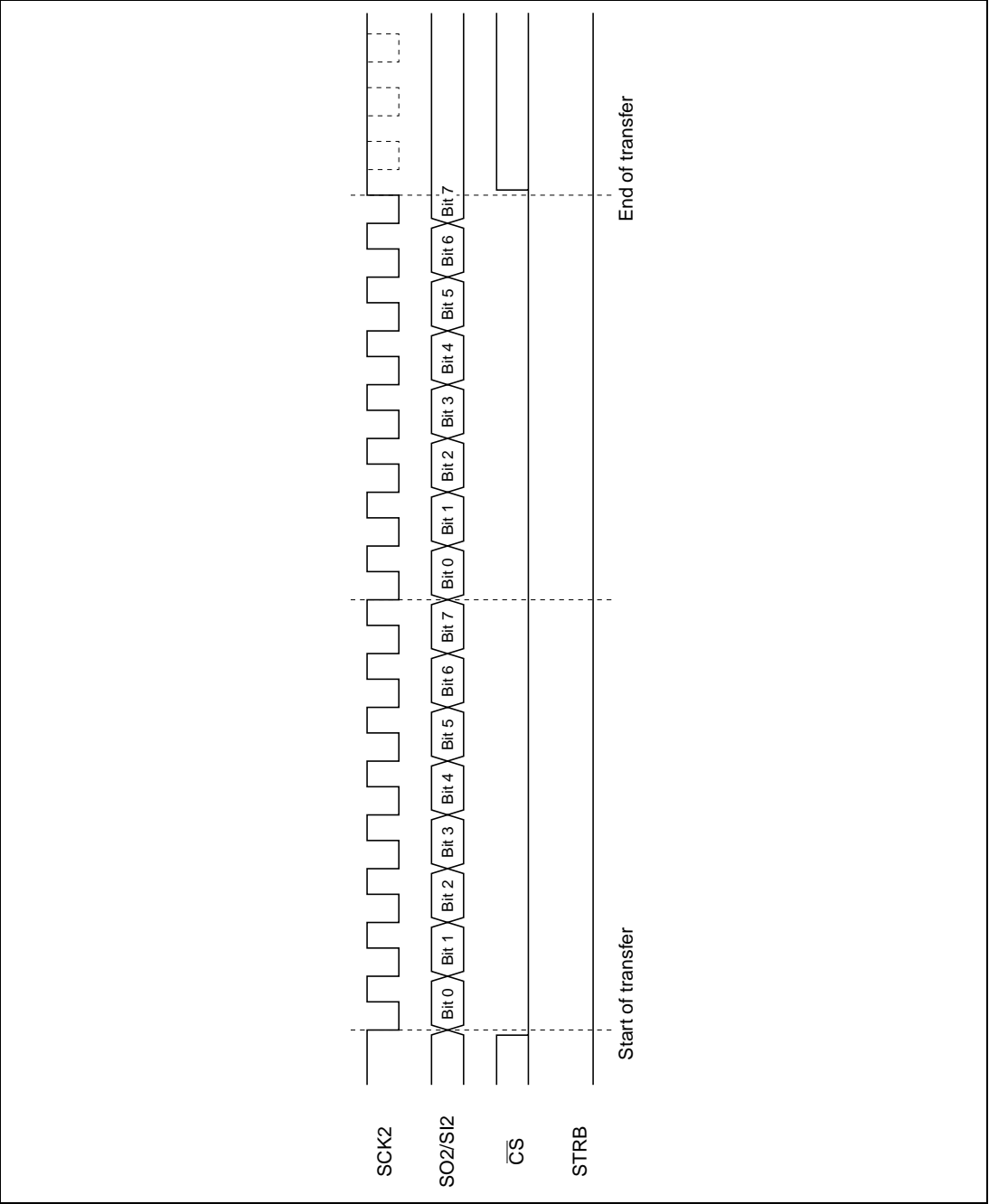


Figure 24.2 Transfer Format (Transfer Data without Intervals)

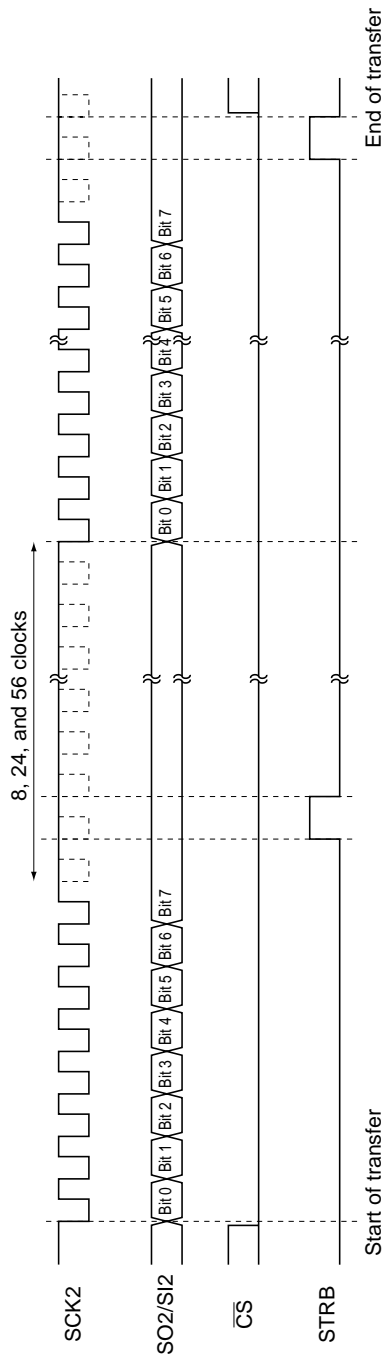


Figure 24.3 Transfer Format (Transfer Data with Intervals)

24.3.3 Data Transfer Operations

(1) SCI2 Initialization

To carry out data transfer, first initialize the SCI2 using software. Initialization is performed as described below:

(1) Use PMR2, PMR3, STAR, EDAR and SCR2 to set the pin and transmission mode while STF of SCSR2 is set to 0.

(2) The SCI2 pin is also used as a port. Switching of a port is performed on PMR.

The SO2 pin allows to select CMOS output or NMOS open drain output on PMR2.

Transfer clock and transfer data intervals can be set on SCR2.

(3) The starting and ending addresses in the transfer data area are set on STAR and EDAR.

If the value of the ending address is smaller than that of the starting address, transfer data at H'FFD0DF and then return to H'FFD0C0 so that transfer to the ending address can be carried out as shows in figure 24.4. If the value of the starting address is equal to that of the ending address is equal, perform one-byte transfer.

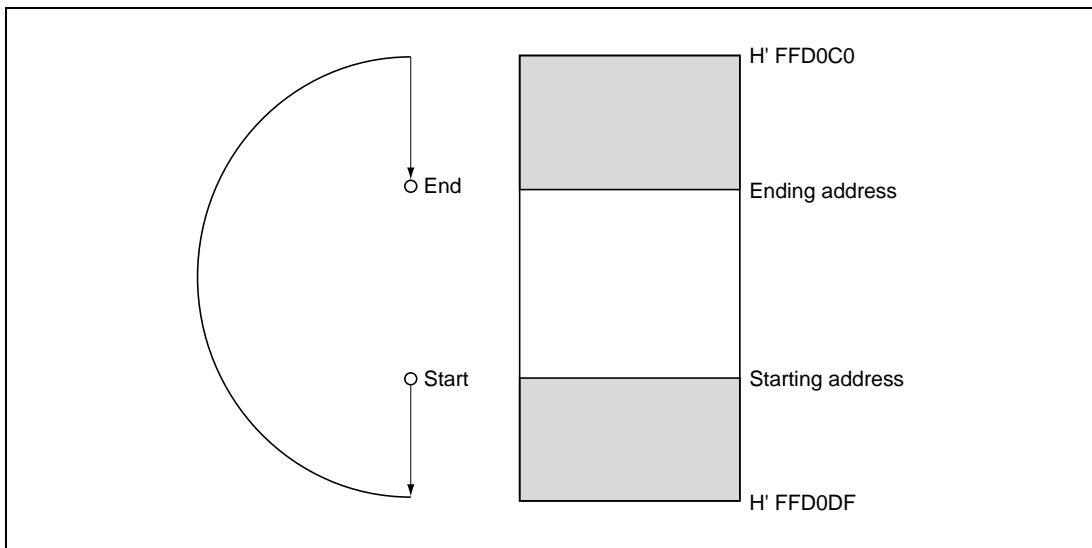


Figure 24.4 If The Value of The Ending Address is Smaller Than That of The Starting Address

(2) Transmit Operations

Transmit operations are performed as described below:

- (1) Set PMR26 and PMR27 of PMR2 to 1 and set them to the SO2 and SCK2 pins, respectively.
Set the SO2 pin to the open drain output using PMR20 of PMR2 and set them to the \overline{CS} and STRB pins, respectively, using PMR30 and PMR31 of PMR3, as necessary.
- (2) Set the transfer clock and transfer data intervals (only when an internal clock is in operation) by setting SCR2.
- (3) Write transmit data to serial data buffer. In transmit operations, the contents of the data buffer will be retained even after the end of transmission. When the same data is transmitted again, it is not necessary to write data.
- (4) Set STAR to the 5 low-order bits at the transmission starting address and EDAR to the 5 low-order bits at the transmission ending address.
- (5) Set STF to 1. When PMR30 of PMR3 is set to 0, transmission is started by setting STF. While PMR30 of PMR3 is set to 1, transmission is started when low level of the \overline{CS} pin is detected.
- (6) After completion of transmission, TEI of SCSR2 is set to 1. STF is cleared to 0.

When an internal clock is selected, synchronous clock is output from the SCK2 pin at the time of starting transmission. When transmission has been completed, synchronous clock is not output until the next STF is set. During that time, the SO2 pin continues to output the value of final bit of the immediately preceding data.

When an external clock is selected, data is transmitted, synchronized with the clock input from the SCK2 pin. If the synchronous clock is continuously input after completion of transmission, no transmission is performed as the overrun state has been found and then ORER of the SCSR2 is set to 1. The SO2 pin continues to retain the value of final bit of the preceding data. However, if the \overline{CS} of PMR3 is set to 1, overrun is not detected when the \overline{CS} pin is at a high level.

The output value of the SO2 pin while transmission is being stopped can be changed by SOL of SCSR2. Data buffer cannot be read or written from CPU during transmission or in the \overline{CS} standby mode.

When a Read instruction has been executed, H'FF is read. Even if a Write instruction is executed, buffer does not change. When a Read/Write instruction has been executed during transmission or in the \overline{CS} input standby mode, WT of the SCSR2 is set.

While PMR30 of PMR3 is set to 0, transmission is immediately cut off when a high level of the \overline{CS} pin has been detected during transmission, and ABT is set to 1, and then STF is cleared to 0. The SCK2 and SO2 pins enter the high impedance state. Therefore, note that transmission may not be carried out while ABT is set to 1, and thus transmission must be resumed after clearing to 0.

(3) Receive Operations

Receive operations are performed as described below:

- (1) Set PMR25 and PMR27 of PMR2 to 1 and set them to the SI2 and SCK2 pins, respectively. Set them to the \overline{CS} pin, using PMR30 of PMR3 as necessary.
- (2) Set the transfer clock and transfer data intervals (only when an internal clock is in operation) by setting SCR2.
- (3) Set STAR to 5 low-order bits at the receive starting address and EDAR to 5 low-order bits at the receive ending address. This enables to determine the area in the serial data buffer where receive data is stored.
- (4) Set STF to 1. When PMR30 of PMR3 is set to 0, reception is started by setting STF. While PMR30 of PMR3 is set to 1, reception is started when low level of the \overline{CS} pin is detected.
- (5) After completion of reception, TEI of SCSR2 is set to 1. STF is cleared to 0.
- (6) Read the receive data stored from the serial data buffer.

When an internal clock is selected, synchronous clock is output from the SCK2 pin at the time of starting reception. When reception has been completed, synchronous clock is not output until the next STF is set.

When an external clock is selected, data is received, synchronized with the clock input from the SCK2 pin. If the synchronous clock is continuously input after completion of reception, no reception is performed as the overrun state has been found and then ORER of the SCSR2 is set to 1. However, if the \overline{CS} of PMR3 is set to 1, overrun is not detected when the \overline{CS} pin is at a high level.

Data buffer cannot be read or written from CPU during reception or in the \overline{CS} standby mode.

When a Read instruction has been executed, H'FF is read. Even if a Write instruction is executed, buffer does not change. When a Read/Write instruction has been executed during reception or in the \overline{CS} input standby mode, WT of the SCSR2 is set.

While \overline{CS} of PMR3 is set to 1, transmission is immediately cut off when a high level of the \overline{CS} pin has been detected during transmission, and ABT is set to 1, and then STF is cleared to 0.

The SCK2 and SO2 pins enter the high impedance state. Therefore, note that transmission may not be carried out while ABT is set to 1, and thus transmission must be resumed after clearing to 0.

(4) Simultaneous Transmit/Receive Operations

Simultaneous transmit/receive operations are performed as described below:

- (1) Set PMR25, PMR26 and PMR27 of PMR2 to 1 and set them to the SI2, SO2 and SCK2 pins, respectively.
Set the SO2 pin to open drain output, using PMR20 of PMR2, and set them to the \overline{CS} and STRB pins, respectively, using PMR30 and PMR31, as necessary.
- (2) Set the transfer clock and transfer data intervals (only when an internal clock is in operation) by setting SCR2.
- (3) Write transmit data to serial data buffer. In the simultaneous transmit/receive operations, the receive data is stored in the same address alternately with the transmit data.
- (4) Set STAR to 5 low-order bits at the transmission starting address and EDAR to 5 low-order bits at the transmission ending address.
- (5) Set STF to 1. When PMR30 of PMR3 is set to 0, transmission is started by setting STF. While PMR30 of PMR3 is set to 1, transmission is started when low level of the \overline{CS} pin is detected.
- (6) After completion of transmission, TEI of SCSR2 is set to 1. STF is cleared to 0.
- (7) Read the receive data stored from the serial data buffer.

When an internal clock is selected, synchronous clock is output from the SCK2 pin at the time of starting transmission. When transmission has been completed, synchronous clock is not output until the next STF is set. During that time, the SO2 pin continues to output the value of final bit of the preceding data.

When an external clock is selected, data is transmitted, synchronized with the clock input from the SCK2 pin. If the synchronous clock is continuously input after completion of transmission, no transmission is performed as the overrun state has been found and then ORER of the SCSR2 is set to 1. The SO2 pin continues to retain the value of final bit of the preceding data.

However, if the CS of PMR3 is set to 1, overrun is not detected when the \overline{CS} pin is at a high level.

The output value of the SO2 pin while transmission is being stopped can be changed by SOL of SCSR2. Data buffer cannot be read or written from CPU during transmission or in the \overline{CS} standby mode. When a Read instruction has been executed, H'FF is read. Even if a Write instruction is executed, buffer does not change. When a Read/Write instruction has been executed during transmission or in the \overline{CS} input standby mode, WT of the SCSR2 is set.

While the CS of PMR3 is set to 1, transmission is immediately cut off when a high level of the \overline{CS} pin has been detected during transmission, and ABT is set to 1, and then STF is cleared to 0. The SCK2 and SO2 pins enter the high impedance state. Therefore, note that transmission may not be carried out while ABT is set to 1, and thus transmission must be resumed after clearing to 0.

24.4 Interrupt Sources

An interrupt source of the SCI2 is transmission cutoff by completion of transmission and the $\overline{\text{CS}}$ pin, to which different vector addresses are assigned.

On completion of data transfer, TEI of SCSR2 is set to 1, and transfer-end interrupt request is generated. This interrupt can specify enable/disable by setting TEIE of SCR2.

While PMR30 of PMR3 is set to 1, transfer is cut off when the $\overline{\text{CS}}$ pin enters a high level during data transfer, and ABT of SCSR2 is set to 1 and then transfer cutoff interrupt request is generated.

This interrupt can specify enable/disable by setting ABTIE of SCR2. In the case of transfer cutoff by the $\overline{\text{CS}}$ pin, overrun error, and read/write to serial data buffer during transfer and in the $\overline{\text{CS}}$ standby mode, ABT, ORER, and WT of the SCSR2 is set to 1, respectively. These bits allow to determine error factors.

Section 25 I²C Bus Interface (IIC)

25.1 Overview

The I²C bus interface conforms to and provides a subset of the Philips I²C bus (inter-IC bus) interface functions. The register configuration that controls the I²C bus differs partly from the Philips configuration, however.

Each I²C bus interface channel uses only one data line (SDA) and one clock line (SCL) to transfer data, saving board and connector space.

25.1.1 Features

- Selection of addressing format or non-addressing format
 - I²C bus format: addressing format with acknowledge bit, for master/slave operation
 - Serial format: non-addressing format without acknowledge bit, for master operation only
- Conforms to Philips I²C bus interface (I²C bus format)
- Two ways of setting slave address (I²C bus format)
- Start and stop conditions generated automatically in master mode (I²C bus format)
- Selection of acknowledge output levels when receiving (I²C bus format)
- Automatic loading of acknowledge bit when transmitting (I²C bus format)
- Wait function in master mode (I²C bus format)
 - A wait can be inserted by driving the SCL pin low after data transfer, excluding acknowledgement. The wait can be cleared by clearing the interrupt flag.
- Wait function in slave mode (I²C bus format)
 - A wait request can be generated by driving the SCL pin low after data transfer, excluding acknowledgement. The wait request is cleared when the next transfer becomes possible.
- Three interrupt sources
 - Data transfer end (including transmission mode transition with I²C bus format and address reception after loss of master arbitration)
 - Address match: when any slave address matches or the general call address is received in slave receive mode (I²C bus format)
 - Stop condition detection
- Selection of 16 internal clocks (in master mode)
- Direct bus drive (with SCL and SDA pins)
 - Two pins-P24/SCL and P23/SDA- (normally CMOS pins) function as NMOS-only outputs when the bus drive function is selected.

25.1.2 Block Diagram

Figure 25.1 shows a block diagram of the I²C bus interface.

Figure 25.2 shows an example of I/O pin connections to external circuits. I/O pins are driven only by NMOS and apparently function as NMOS open-drain outputs. However, applicable voltages to input pins depend on the power (V_{cc}) voltage of this LSI.

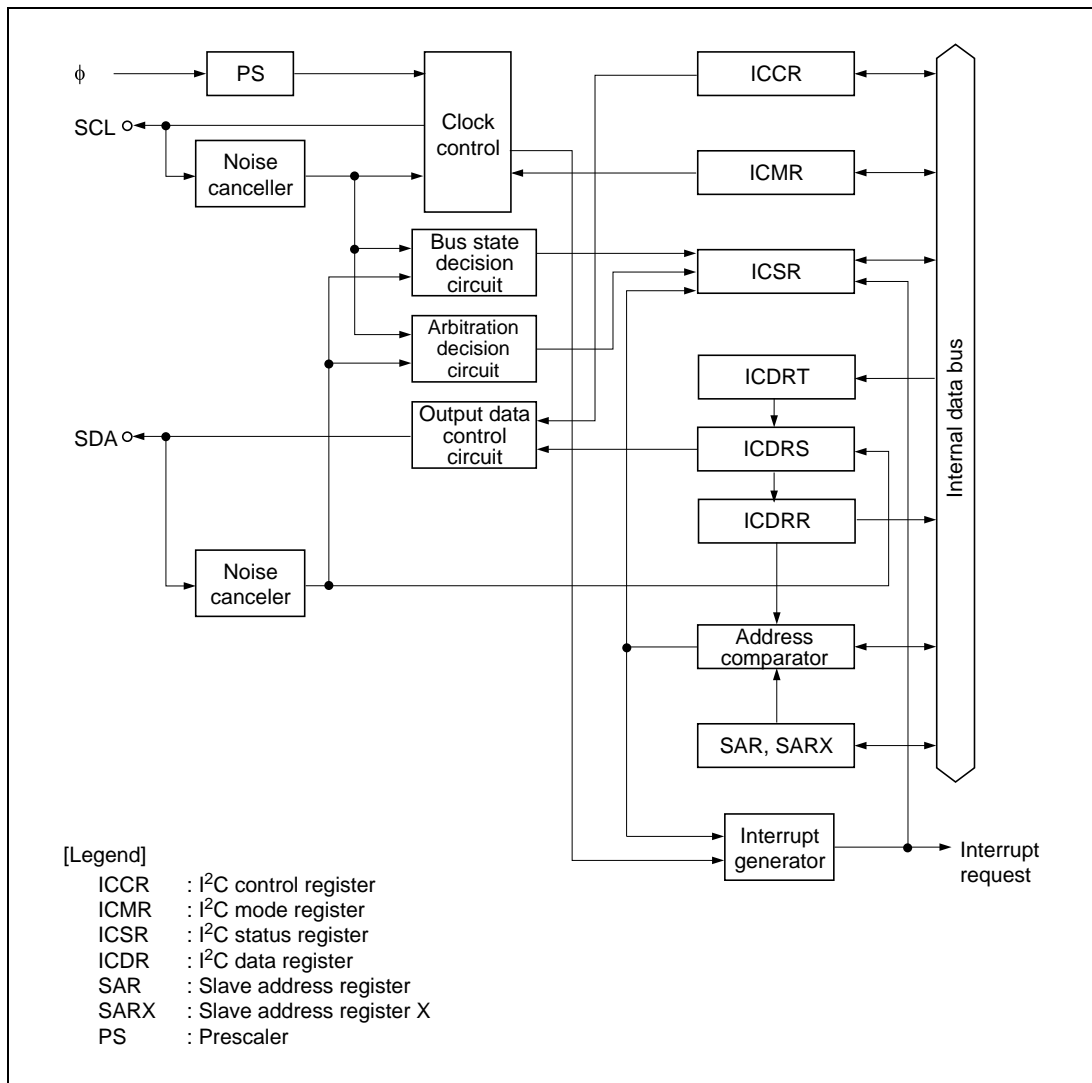


Figure 25.1 Block Diagram of I²C Bus Interface

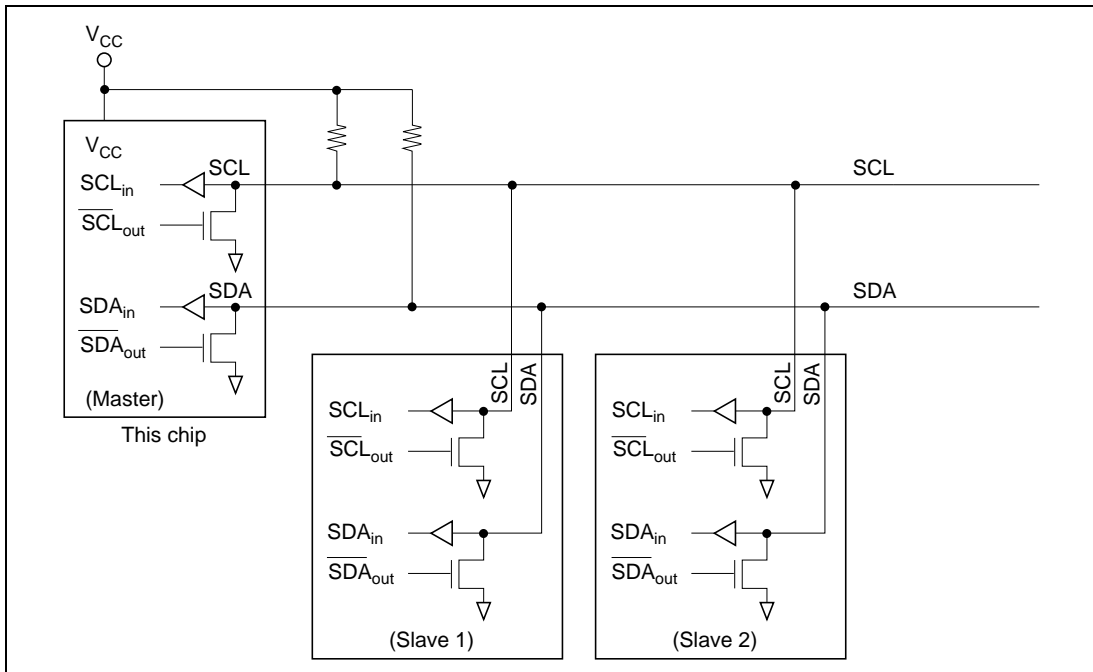


Figure 25.2 I²C Bus Interface Connections (Example: This Chip as Master)

25.1.3 Pin Configuration

Table 25.1 summarizes the input/output pins used by the I²C bus interface.

Table 25.1 I²C Bus Interface Pins

| Name | Abbrev. | I/O | Function |
|------------------|---------|-----|-------------------------------|
| Serial clock pin | SCL | I/O | IIC serial clock input/output |
| Serial data pin | SDA | I/O | IIC serial data input/output |

25.1.4 Register Configuration

Table 25.2 summarizes the registers of the I²C bus interface.

Table 25.2 Register Configuration

| Name | Abbrev. | R/W | Initial Value | Address ^{*1} |
|---------------------------------------|---------|-----|---------------|-----------------------|
| I ² C bus control register | ICCR | R/W | H'01 | H'D158 |
| I ² C bus status register | ICSR | R/W | H'00 | H'D159 |
| I ² C bus data register | ICDR | R/W | — | H'D15E ^{*2} |
| I ² C bus mode register | ICMR | R/W | H'00 | H'D15F ^{*2} |
| Slave address register | SAR | R/W | H'00 | H'D15F ^{*2} |
| Second slave address register | SARX | R/W | H'01 | H'D15E ^{*2} |
| Serial timer control register | STCR | R/W | H'00 | H'FFEE |
| Module stop control register | MSTPCRH | R/W | H'FF | H'FFEC |
| | MSTPCRL | R/W | H'FF | H'FFED |

Notes: 1. Lower 16 bits of the address.

2. The register that can be written or read depends on the ICE bit in the I²C bus control register. The slave address register can be accessed when ICE = 0, and the I²C bus mode register can be accessed when ICE = 1.

25.2 Register Descriptions

25.2.1 I²C Bus Data Register (ICDR)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICDR7 | ICDR6 | ICDR5 | ICDR4 | ICDR3 | ICDR2 | ICDR1 | ICDR0 |
| Initial value : | — | — | — | — | — | — | — | — |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

ICDRR

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICDRR7 | ICDRR6 | ICDRR5 | ICDRR4 | ICDRR3 | ICDRR2 | ICDRR1 | ICDRR0 |
| Initial value : | — | — | — | — | — | — | — | — |
| R/W : | R | R | R | R | R | R | R | R |

ICDRS

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICDRS7 | ICDRS6 | ICDRS5 | ICDRS4 | ICDRS3 | ICDRS2 | ICDRS1 | ICDRS0 |
| Initial value : | — | — | — | — | — | — | — | — |
| R/W : | — | — | — | — | — | — | — | — |

ICDRT

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICDRT7 | ICDRT6 | ICDRT5 | ICDRT4 | ICDRT3 | ICDRT2 | ICDRT1 | ICDRT0 |
| Initial value : | — | — | — | — | — | — | — | — |
| R/W : | W | W | W | W | W | W | W | W |

TDRE, RDRF (Internal flag)

| | | |
|-----------------|------|------|
| Bit : | — | — |
| | TDRE | RDRF |
| Initial value : | 0 | 0 |
| R/W : | — | — |

ICDR is an 8-bit readable/writable register that is used as a transmit data register when transmitting and a receive data register when receiving. ICDR is divided internally into a shift register (ICDRS), receive buffer (ICDRR), and transmit buffer (ICDRT). ICDRS cannot be read or written by the CPU, ICDRR is read-only, and ICDRT is write-only. Data transfers among the three registers are performed automatically in coordination with changes in the bus state, and affect the status of internal flags such as TDRE and RDRF.

After transmission/reception of one frame of data using ICDRS, if the I²C bus is in transmit mode and the next data is in ICDRT (the TDRE flag is 0), data is transferred automatically from ICDRT to ICDRS. After transmission/reception of one frame of data using ICDRS, if the I²C bus is in receive mode and no previous data remains in ICDRR (the RDRF flag is 0), data is transferred automatically from ICDRS to ICDRR.

If the number of bits in a frame, excluding the acknowledge bit, is less than 8, transmit data and receive data are stored differently. Transmit data should be written justified toward the MSB side when MLS = 0, and toward the LSB side when MLS = 1. Receive data bits read from the LSB side should be treated as valid when MLS = 0, and bits read from the MSB side when MLS = 1.

ICDR is assigned to the same address as SARX, and can be written and read only when the ICE bit is set to 1 in ICCR.

The value of ICDR is undefined after a reset.

The TDRE and RDRF flags are set and cleared under the conditions shown below. Setting the TDRE and RDRF flags affects the status of the interrupt flags.

| TDRE | Description |
|-------------|--|
| 0 | <p>The next transmit data is in ICDR (ICDRT), or transmission cannot be started [Clearing conditions] (Initial value)</p> <ul style="list-style-type: none"> (1) When transmit data is written in ICDR (ICDRT) in transmit mode (TRS = 1) (2) When a stop condition is detected in the bus line state after a stop condition is issued with the I²C bus format or serial format selected (3) When a stop condition is detected with the I²C bus format selected (4) In receive mode (TRS = 0) (A 0 write to TRS during transfer is valid after reception of a frame containing an acknowledge bit) |
| 1 | <p>The next transmit data can be written in ICDR (ICDRT) [Setting conditions]</p> <ul style="list-style-type: none"> (1) In transmit mode (TRS = 1), when a start condition is detected in the bus line state after a start condition is issued in master mode with the I²C bus format or serial format selected (2) When data is transferred from ICDRT to ICDRS (Data transfer from ICDRT to ICDRS when TRS = 1 and TDRE = 0, and ICDRS is empty) (3) When a switch is made from receive mode (TRS = 0) to transmit mode (TRS = 1) after detection of a start condition |
| RDRF | Description |
| 0 | <p>The data in ICDR (ICDRR) is invalid (Initial value) [Clearing condition] When ICDR (ICDRR) receive data is read in receive mode</p> |
| 1 | <p>The ICDR (ICDRR) receive data can be read [Setting condition] When data is transferred from ICDRS to ICDRR (Data transfer from ICDRS to ICDRR in case of normal termination with TRS = 0 and RDRF = 0)</p> |

25.2.2 Slave Address Register (SAR)

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | SVA6 | SVA5 | SVA4 | SVA3 | SVA2 | SVA1 | SVA0 | FS |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SAR is an 8-bit readable/writable register that stores the slave address and selects the communication format. When the chip is in slave mode (and the addressing format is selected), if the upper 7 bits of SAR match the upper 7 bits of the first frame received after a start condition, the chip operates as the slave device specified by the master device. SAR is assigned to the same address as ICMR, and can be written and read only when the ICE bit is cleared to 0 in ICCR.

SAR is initialized to H'00 by a reset.

Bits 7 to 1: Slave Address (SVA6 to SVA0)

Set a unique address in bits SVA6 to SVA0, differing from the addresses of other slave devices connected to the I²C bus.

Bit 0: Format Select (FS)

Used together with the FSX bit in SARX to select the communication format.

- I²C bus format: addressing format with acknowledge bit
- Synchronous serial format: non-addressing format without acknowledge bit, for master mode only

The FS bit also specifies whether or not SAR slave address recognition is performed in slave mode.

| SAR Bit 0 | SARX Bit 0 | Operating Mode |
|--------------|---------------|---|
| FS | FSX | |
| 0 | 0 | I ² C bus format <ul style="list-style-type: none">• SAR and SARX slave addresses recognized |
| | 1 | I ² C bus format (Initial value) <ul style="list-style-type: none">• SAR slave address recognized• SARX slave address ignored |
| 1 | 0 | I ² C bus format <ul style="list-style-type: none">• SAR slave address ignored• SARX slave address recognized |
| | 1 | Clock synchronous serial format <ul style="list-style-type: none">• SAR and SARX slave addresses ignored |

25.2.3 Second Slave Address Register (SARX)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | SVAX6 | SVAX5 | SVAX4 | SVAX3 | SVAX2 | SVAX1 | SVAX0 | FSX |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SARX is an 8-bit readable/writable register that stores the second slave address and selects the communication format. When the chip is in slave mode (and the addressing format is selected), if the upper 7 bits of SARX match the upper 7 bits of the first frame received after a start condition, the chip operates as the slave device specified by the master device. SARX is assigned to the same address as ICDR, and can be written and read only when the ICE bit is cleared to 0 in ICCR.

SARX is initialized to H'01 by a reset and in hardware standby mode.

Bits 7 to 1: Second Slave Address (SVAX6 to SVAX0)

Set a unique address in bits SVAX6 to SVAX0, differing from the addresses of other slave devices connected to the I²C bus.

Bit 0: Format Select X (FSX)

Used together with the FS bit in SAR to select the communication format.

- I²C bus format: addressing format with acknowledge bit
- Synchronous serial format: non-addressing format without acknowledge bit, for master mode only

The FSX bit also specifies whether or not SARX slave address recognition is performed in slave mode. For details, see the description of the FS bit in SAR.

25.2.4 I²C Bus Mode Register (ICMR)

| | | | | | | | | |
|-----------------|-----|------|------|------|------|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | MLS | WAIT | CKS2 | CKS1 | CKS0 | BC2 | BC1 | BC0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

ICMR is an 8-bit readable/writable register that selects whether the MSB or LSB is transferred first, performs master mode wait control, and selects the master mode transfer clock frequency and the transfer bit count. ICMR is assigned to the same address as SAR. ICMR can be written and read only when the ICE bit is set to 1 in ICCR. ICMR is initialized to H'00 by a reset.

Bit 7: MSB-First/LSB-First Select (MLS)

Selects whether data is transferred MSB-first or LSB-first.

If the number of bits in a frame, excluding the acknowledge bit, is less than 8, transmit data and receive data are stored differently. Transmit data should be written justified toward the MSB side when MLS = 0, and toward the LSB side when MLS = 1. Receive data bits read from the LSB side should be treated as valid when MLS = 0, and bits read from the MSB side when MLS = 1.

Do not set this bit to 1 when the I²C bus format is used.

Bit 7

| MLS | Description |
|-----|---------------------------|
| 0 | MSB-first (Initial value) |
| 1 | LSB-first |

Bit 6: Wait Insertion Bit (WAIT)

Selects whether to insert a wait between the transfer of data and the acknowledge bit, in master mode with the I²C bus format. When WAIT is set to 1, after the fall of the clock for the final data bit, the IRIC flag is set to 1 in ICCR, and a wait state begins (with SCL at the low level). When the IRIC flag is cleared to 0 in ICCR, the wait ends and the acknowledge bit is transferred. If WAIT is cleared to 0, data and acknowledge bits are transferred consecutively with no wait inserted.

The IRIC flag in ICCR is set to 1 on completion of the acknowledge bit transfer, regardless of the WAIT setting.

The setting of this bit is invalid in slave mode.

Bit 6

| WAIT | Description |
|------|---|
| 0 | Data and acknowledge bits transferred consecutively (Initial value) |
| 1 | Wait inserted between data and acknowledge bits |

Bits 5 to 3: Transfer Clock Select (CKS2 to CKS0)

These bits, together with the IICX bit in the STCR register, select the serial clock frequency in master mode. They should be set according to the required transfer rate.

| STCR | | | | | Transfer Rate | | |
|-------|-------|-------|-------|------------|------------------------|------------------------|-------------------------|
| Bit 6 | Bit 5 | Bit 4 | Bit 3 | | | | |
| IICX | CKS2 | CKS1 | CKS0 | Clock | $\phi = 5 \text{ MHz}$ | $\phi = 8 \text{ MHz}$ | $\phi = 10 \text{ MHz}$ |
| 0 | 0 | 0 | 0 | $\phi/28$ | 179 kHz | 286 kHz | 357 kHz |
| | | | 1 | $\phi/40$ | 125 kHz | 200 kHz | 250 kHz |
| | | 1 | 0 | $\phi/48$ | 104 kHz | 167 kHz | 208 kHz |
| | | | 1 | $\phi/64$ | 78.1 kHz | 125 kHz | 156 kHz |
| | 1 | 0 | 0 | $\phi/80$ | 62.5 kHz | 100 kHz | 125 kHz |
| | | | 1 | $\phi/100$ | 50.0 kHz | 80.0 kHz | 100 kHz |
| | | 1 | 0 | $\phi/112$ | 44.6 kHz | 71.4 kHz | 89.3 kHz |
| | | | 1 | $\phi/128$ | 39.1 kHz | 62.5 kHz | 78.1 kHz |
| 1 | 0 | 0 | 0 | $\phi/56$ | 89.3 kHz | 143 kHz | 179 kHz |
| | | | 1 | $\phi/80$ | 62.5 kHz | 100 kHz | 125 kHz |
| | | 1 | 0 | $\phi/96$ | 52.1 kHz | 83.3 kHz | 104 kHz |
| | | | 1 | $\phi/128$ | 39.1 kHz | 62.5 kHz | 78.1 kHz |
| | 1 | 0 | 0 | $\phi/160$ | 31.3 kHz | 50.0 kHz | 62.5 kHz |
| | | | 1 | $\phi/200$ | 25.0 kHz | 40.0 kHz | 50.0 kHz |
| | | 1 | 0 | $\phi/224$ | 22.3 kHz | 35.7 kHz | 44.6 kHz |
| | | | 1 | $\phi/256$ | 19.5 kHz | 31.3 kHz | 39.1 kHz |

Bits 2 to 0: Bit Counter (BC2 to BC0)

Bits BC2 to BC0 specify the number of bits to be transferred next. With the I²C bus format (when the FS bit in SAR or the FSX bit in SARX is 0), the data is transferred with one additional acknowledge bit. Bit BC2 to BC0 settings should be made during an interval between transfer frames. If bits BC2 to BC0 are set to a value other than 000, the setting should be made while the SCL line is low.

The bit counter is initialized to 000 by a reset and when a start condition is detected. The value returns to 000 at the end of a data transfer, including the acknowledge bit.

| Bit 2 | Bit 1 | Bit 0 | Bits/Frame | |
|-------|-------|-------|---------------------------|-----------------------------|
| BC2 | BC1 | BC0 | Synchronous Serial Format | I ² C Bus Format |
| 0 | 0 | 0 | 8 | 9 (Initial value) |
| | | 1 | 1 | 2 |
| | 1 | 0 | 2 | 3 |
| | | 1 | 3 | 4 |
| 1 | 0 | 0 | 4 | 5 |
| | | 1 | 5 | 6 |
| | 1 | 0 | 6 | 7 |
| | | 1 | 7 | 8 |

25.2.5 I²C Bus Control Register (ICCR)

| | | | | | | | | |
|-----------------|-----|------|-----|-----|------|------|--------|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICE | IEIC | MST | TRS | ACKE | BBSY | IRIC | SCP |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/(W)* | W |

Note: * Only 0 can be written to clear the flag.

ICCR is an 8-bit readable/writable register that enables or disables the I²C bus interface, enables or disables interrupts, selects master or slave mode and transmission or reception, enables or disables acknowledgement, confirms the I²C bus interface bus status, issues start/stop conditions, and performs interrupt flag confirmation.

ICCR is initialized to H'01 by a reset.

Bit 7: I²C Bus Interface Enable (ICE)

Selects whether or not the I²C bus interface is to be used. When ICE is set to 1, port pins function as SCL and SDA input/output pins and transfer operations are enabled. When ICE is cleared to 0, the I²C bus interface module is disabled, and the internal state is initialized. The SAR and SARX registers can be accessed when ICE is 0. The ICMR and ICDR registers can be accessed when ICE is 1.

Bit 7

| ICE | Description |
|-----|---|
| 0 | I ² C bus interface module disabled, with SCL and SDA signal pins set to port function SAR and SARX can be accessed. The internal state of the I ² C bus interface module is initialized. (Initial value) |
| 1 | I ² C bus interface module enabled for transfer operations (pins SCL and SCA are driving the bus) ICMR and ICDR can be accessed |

Bit 6: I²C Bus Interface Interrupt Enable (IEIC)

Enables or disables interrupts from the I²C bus interface to the CPU.

Bit 6

| IEIC | Description |
|------|-------------------------------------|
| 0 | Interrupts disabled (Initial value) |
| 1 | Interrupts enabled |

Bit 5: Master/Slave Select (MST)

Bit 4: Transmit/Receive Select (TRS)

MST selects whether the I²C bus interface operates in master mode or slave mode.

TRS selects whether the I²C bus interface operates in transmit mode or receive mode.

In master mode with the I²C bus format, when arbitration is lost, MST and TRS are both reset by hardware, causing a transition to slave receive mode. In slave receive mode with the addressing format (FS = 0 or FSX = 0), hardware automatically selects transmit or receive mode according to the R/W bit in the first frame after a start condition.

Modification of the TRS bit during transfer is deferred until transfer of the frame containing the acknowledge bit is completed, and the changeover is made after completion of the transfer.

MST and TRS select the operating mode as follows.

| Bit 5 | Bit 4 | Description |
|-------|-------|------------------------------------|
| MST | TRS | |
| 0 | 0 | Slave receive mode (Initial value) |
| | 1 | Slave transmit mode |
| 1 | 0 | Master receive mode |
| | 1 | Master transmit mode |

| Bit 5 | Description |
|-------|--|
| MST | |
| 0 | Slave mode (Initial value) |
| | [Clearing conditions] (1) When 0 is written by software (2) When bus arbitration is lost after transmission is started in I ² C bus format master mode |
| 1 | Master mode |
| | [Setting conditions] (1) When 1 is written by software (in cases other than clearing condition 2) (2) When 1 is written in MST after reading MST = 0 (in case of clearing condition 2) |

Bit 4

| TRS | Description |
|-----|--|
| 0 | Receive mode (Initial value) [Clearing conditions] (1) When 0 is written by software (in cases other than setting condition 3) (2) When 0 is written in TRS after reading TRS = 1 (in case of setting condition 3) (3) When bus arbitration is lost after transmission is started in I ² C bus format master mode |
| 1 | Transmit mode [Setting conditions] (1) When 1 is written by software (in cases other than clearing conditions 3) (2) When 1 is written in TRS after reading TRS = 0 (in case of clearing conditions 3) (3) When a 1 is received as the R/W bit of the first frame in I ² C bus format slave mode |

Bit 3: Acknowledge Bit Judgement Selection (ACKE)

Specifies whether the value of the acknowledge bit returned from the receiving device when using the I²C bus format is to be ignored and continuous transfer is performed, or transfer is to be aborted and error handling, etc., performed if the acknowledge bit is 1. When the ACKE bit is 0, the value of the received acknowledge bit is not indicated by the ACKB bit, which is always 0.

When the ACKE bit is 0, the TDRE, IRIC, and IRTR flags are set on completion of data transmission, regardless of the value of the acknowledge bit. When the ACKE bit is 1, the TDRE, IRIC, and IRTR flags are set on completion of data transmission when the acknowledge bit is 0, and the IRIC flag alone is set on completion of data transmission when the acknowledge bit is 1.

Depending on the receiving device, the acknowledge bit may be significant, in indicating completion of processing of the received data, for instance, or may be fixed at 1 and have no significance.

Bit 3

| ACKE | Description |
|------|---|
| 0 | The value of the acknowledge bit is ignored, and continuous transfer is performed (Initial value) |
| 1 | If the acknowledge bit is 1, continuous transfer is interrupted |

Bit 2: Bus Busy (BBSY)

The BBSY flag can be read to check whether the I²C bus (SCL, SDA) is busy or free. In master mode, this bit is also used to issue start and stop conditions.

A high-to-low transition of SDA while SCL is high is recognized as a start condition, setting BBSY to 1. A low-to-high transition of SDA while SCL is high is recognized as a stop condition, clearing BBSY to 0.

To issue a start condition, use a MOV instruction to write 1 in BBSY and 0 in SCP. A retransmit start condition is issued in the same way. To issue a stop condition, use a MOV instruction to write 0 in BBSY and 0 in SCP.

It is not possible to write to BBSY in slave mode; the I²C bus interface must be set to master transmit mode before issuing a start condition. MST and TRS should both be set to 1 before writing 1 in BBSY and 0 in SCP.

Bit 2

| BBSY | Description |
|------|--|
| 0 | Bus is free [Clearing condition] When a stop condition is detected (Initial value) |
| 1 | Bus is busy [Setting condition] When a start condition is detected |

Bit 1: I²C Bus Interface Interrupt Request Flag (IRIC)

Indicates that the I²C bus interface has issued an interrupt request to the CPU. IRIC is set to 1 at the end of a data transfer, when a slave address or general call address is detected in slave receive mode, when bus arbitration is lost in master transmit mode, and when a stop condition is detected. IRIC is set at different times depending on the FS bit in SAR and the WAIT bit in ICMR. See section 25.3.6, IRIC Setting Timing and SCL Control. The conditions under which IRIC is set also differ depending on the setting of the ACKIE bit in ICCR.

IRIC is cleared by reading IRIC after it has been set to 1, then writing 0 in IRIC.

When the DTC is used, IRIC is cleared automatically and transfer can be performed continuously without CPU intervention.

| IRIC | Description |
|------|---|
| 0 | <p>Waiting for transfer, or transfer in progress (Initial value)</p> <p>[Clearing condition]</p> <p>(1) When 0 is written in IRIC after reading IRIC = 1</p> |
| 1 | <p>Interrupt requested</p> <p>[Setting conditions]</p> <p>■ I²C bus format master mode</p> <p>(1) When a start condition is detected in the bus line state after a start condition is issued (when the TDRE flag is set to 1 because of first frame transmission)</p> <p>(2) When a wait is inserted between the data and acknowledge bit when WAIT = 1</p> <p>(3) At the end of data transfer (at the rise of the 9th transmit clock pulse, and at the fall of the 8th transmit/receive clock pulse when a wait is inserted)</p> <p>(4) When a slave address is received after bus arbitration is lost (when the AL flag is set to 1)</p> <p>(5) When 1 is received as the acknowledge bit when the ACKE bit is 1 (when the ACKB bit is set to 1)</p> <p>■ I²C bus format slave mode</p> <p>(1) When the slave address (SVA, SVAX) matches (when the AAS and AASX flags are set to 1) and at the end of data transfer up to the subsequent retransmission start condition or stop condition detection (when the TDRE or RDRF flag is set to 1)</p> <p>(2) When the general call address is detected (when FS = 0 and the ADZ flag is set to 1) and at the end of data transfer up to the subsequent retransmission start condition or stop condition detection (when the TDRE or RDRF flag is set to 1)</p> <p>(3) When 1 is received as the acknowledge bit when the ACKE bit is 1 (when the ACKB bit is set to 1)</p> <p>(4) When a stop condition is detected (when the STOP or ESTP flag is set to 1)</p> <p>■ Synchronous serial format</p> <p>(1) At the end of data transfer (when the TDRE or RDRF flag is set to 1)</p> <p>(2) When a start condition is detected with serial format selected</p> <p>When conditions are occurred such that the TDRE or RDRF flag is set to 1</p> |

When, with the I²C bus format selected, IRIC is set to 1 and an interrupt is generated, other flags must be checked in order to identify the source that set IRIC to 1. Although each source has a corresponding flag, caution is needed at the end of a transfer.

When the TDRE or RDRF internal flag is set, the readable IRTR flag may or may not be set. The IRTR flag (the DTC* start request flag) is not set at the end of a data transfer up to detection of a retransmission start condition or stop condition after a slave address (SVA) or general call address match in I²C bus format slave mode.

Even when the IRIC flag and IRTR flag are set, the TDRE or RDRF internal flag may not be set. The IRIC and IRTR flags are not cleared at the end of the specified number of transfers in continuous transfer using the DTC*. The TDRE or RDRF flag is cleared, however, since the specified number of ICDR reads or writes have been completed.

Table 25.3 shows the relationship between the flags and the transfer states.

Note: * This LSI does not incorporate DTC.

Table 25.3 Flags and Transfer States

| MST | TRS | BBSY | ESTP | STOP | IRTR | AASX | AL | AAS | ADZ | ACKB | State |
|-----|-----|------|------|------|------|------|----|-----|-----|------|---|
| 1/0 | 1/0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Idle state (flag clearing required) |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Start condition issuance |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | Start condition established |
| 1 | 1/0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | Master mode wait |
| 1 | 1/0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0/1 | Master mode transmit/receive end |
| 0 | 0 | 1 | 0 | 0 | 0 | 1/0 | 1 | 1/0 | 1/0 | 0 | Arbitration lost |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | SAR match by first frame in slave mode |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | General call address match |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | SARX match |
| 0 | 1/0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | Slave mode transmit/receive end (except after SARX match) |
| 0 | 1/0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | Slave mode transmit/receive end (after SARX match) |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | |
| 0 | 1/0 | 0 | 1/0 | 1/0 | 0 | 0 | 0 | 0 | 0 | 0/1 | Stop condition detected |

Bit 0: Start Condition/Stop Condition Prohibit (SCP)

Controls the issuing of start and stop conditions in master mode. To issue a start condition, write 1 in BBSY and 0 in SCP. A retransmit start condition is issued in the same way. To issue a stop condition, write 0 in BBSY and 0 in SCP. This bit is always read as 1. If 1 is written, the data is not stored.

Bit 0

| SCP | Description |
|-----|---|
| 0 | Writing 0 issues a start or stop condition, in combination with the BBSY flag |
| 1 | Reading always returns a value of 1 Writing is ignored |

25.2.6 I²C Bus Status Register (ICSR)

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ESTP | STOP | IRTR | AASX | AL | AAS | ADZ | ACKB |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/W |

Note: * Only 0 can be written to clear the flag.

ICSR is an 8-bit readable/writable register that performs flag confirmation and acknowledge confirmation and control.

ICSR is initialized to H'00 by a reset.

Bit 7: Error Stop Condition Detection Flag (ESTP)

Indicates that a stop condition has been detected during frame transfer in I²C bus format slave mode.

Bit 7

| ESTP | Description |
|------|--|
| 0 | No error stop condition (Initial value) [Clearing condition] (1) When 0 is written in ESTP after reading ESTP = 1 (2) When the IRIC flag is cleared to 0 |
| 1 | <div> <div>■ In I²C bus format slave mode</div> <div>Error stop condition detected</div> <div>[Setting condition]</div> <div> <ul style="list-style-type: none"> When a stop condition is detected during frame transfer </div> <div>■ In other modes</div> <div>No meaning</div> </div> |

Bit 6: Normal Stop Condition Detection Flag (STOP)

Indicates that a stop condition has been detected after completion of frame transfer in I²C bus format slave mode.

Bit 6

| STOP | Description |
|------|---|
| 0 | No normal stop condition (Initial value) [Clearing condition] (1) When 0 is written in STOP after reading STOP = 1 (2) When the IRIC flag is cleared to 0 |
| 1 | ■ In I ² C bus format slave mode Error stop condition detected [Setting condition] <ul style="list-style-type: none">• When a stop condition is detected after completion of frame transfer ■ In other modes No meaning |

Bit 5: I²C Bus Interface Continuous Transmission/Reception Interrupt Request Flag (IRTR)

Indicates that the I²C bus interface has issued an interrupt request to the CPU, and the source is completion of reception/transmission of one frame in continuous transmission/reception for which DTC* activation is possible. When the IRTR flag is set to 1, the IRIC flag is also set to 1 at the same time.

IRTR flag setting is performed when the TDRE or RDRF flag is set to 1. IRTR is cleared by reading IRTR after it has been set to 1, then writing 0 in IRTR. IRTR is also cleared automatically when the IRIC flag is cleared to 0.

Note: * This LSI does not incorporate DTC.

Bit 5

| IRTR | Description |
|------|--|
| 0 | Waiting for transfer, or transfer in progress (Initial value) [Clearing condition] (1) When 0 is written in IRTR after reading IRTR = 1 (2) When the IRIC flag is cleared to 0 |
| 1 | Continuous transfer state [Setting condition] ■ In I ² C bus interface slave mode • When the TDRE or RDRF flag is set to 1 when AASX = 1 ■ In other modes • When the TDRE or RDRF flag is set to 1 |

Bit 4: Second Slave Address Recognition Flag (AASX)

In I²C bus format slave receive mode, this flag is set to 1 if the first frame following a start condition matches bits SVAX6 to SVAX0 in SARX.

AASX is cleared by reading AASX after it has been set to 1, then writing 0 in AASX. AASX is also cleared automatically when a start condition is detected.

Bit 4

| AASX | Description |
|------|---|
| 0 | Second slave address not recognized (Initial value) [Clearing condition] (1) When 0 is written in AASX after reading AASX = 1 (2) When a start condition is detected (3) In master mode |
| 1 | Second slave address recognized [Setting condition] <ul style="list-style-type: none">• When the second slave address is detected in slave receive mode while FSX = 0 |

Bit 3: Arbitration Lost (AL)

This flag indicates that arbitration was lost in master mode. The I²C bus interface monitors the bus. When two or more master devices attempt to seize the bus at nearly the same time, if the I²C bus interface detects data differing from the data it sent, it sets AL to 1 to indicate that the bus has been taken by another master.

AL is cleared by reading AL after it has been set to 1, then writing 0 in AL. In addition, AL is reset automatically by write access to ICDR in transmit mode, or read access to ICDR in receive mode.

Bit 3

| AL | Description |
|----|--|
| 0 | Bus arbitration won (Initial value) [Clearing conditions] (1) When ICDR data is written (transmit mode) or read (receive mode) (2) When 0 is written in AL after reading AL = 1 |
| 1 | Arbitration lost [Setting conditions] (1) If the internal SDA and SDA pin disagree at the rise of SCL in master transmit mode (2) If the internal SCL line is high at the fall of SCL in master transmit mode |

Bit 2: Slave Address Recognition Flag (AAS)

In I²C bus format slave receive mode, this flag is set to 1 if the first frame following a start condition matches bits SVA6 to SVA0 in SAR, or if the general call address (H'00) is detected. AAS is cleared by reading AAS after it has been set to 1, then writing 0 in AAS. In addition, AAS is reset automatically by write access to ICDR in transmit mode, or read access to ICDR in receive mode.

Bit 2

| AAS | Description |
|-----|---|
| 0 | Slave address or general call address not recognized (Initial value) [Clearing conditions] (1) When ICDR data is written (transmit mode) or read (receive mode) (2) When 0 is written in AAS after reading AAS = 1 (3) In master mode |
| 1 | Slave address or general call address recognized [Setting condition] <ul style="list-style-type: none">• When the slave address or general call address is detected in slave receive mode |

Bit 1: General Call Address Recognition Flag (ADZ)

In I²C bus format slave receive mode, this flag is set to 1 if the first frame following a start condition is the general call address (H'00). ADZ is cleared by reading ADZ after it has been set to 1, then writing 0 in ADZ. In addition, ADZ is reset automatically by write access to ICDR in transmit mode, or read access to ICDR in receive mode.

Bit 1

| ADZ | Description |
|-----|--|
| 0 | General call address not recognized (Initial value) [Clearing conditions] (1) When ICDR data is written (transmit mode) or read (receive mode) (2) When 0 is written in ADZ after reading ADZ = 1 (3) In master mode |
| 1 | General call address recognized [Setting condition] <ul style="list-style-type: none">• If the general call address is detected when FSX = 0 or FS = 0 is selected in the slave receive mode. |

Bit 0: Acknowledge Bit (ACKB)

Stores acknowledge data. In transmit mode, after the receiving device receives data, it returns acknowledge data, and this data is loaded into ACKB. In receive mode, after data has been received, the acknowledge data set in this bit is sent to the transmitting device.

When this bit is read, in transmission (when TRS = 1), the value loaded from the bus line (returned by the receiving device) is read. In reception (when TRS = 0), the value set by internal software is read.

Bit 0

| ACKB | Description |
|------|--|
| 0 | Receive mode: 0 is output at acknowledge output timing (Initial value) Transmit mode: Indicates that the receiving device has acknowledged the data (signal is 0) |
| 1 | Receive mode: 1 is output at acknowledge output timing Transmit mode: Indicates that the receiving device has not acknowledged the data (signal is 1) |

25.2.7 Serial/Timer Control Register (STCR)

| | | | | | | | | |
|-----------------|---|------|--------|---|-------|---|---|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | IICX | IICRST | — | FLSHE | — | — | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | R/W | R/W | — | R/W | — | — | — |

STCR is an 8-bit readable/writable register that controls the I²C bus interface operating mode. STCR is initialized to H'00 by a reset.

Bit 7: Reserved

Bit 6: I²C Transfer Select (IICX)

This bit, together with bits CKS2 to CKS0 in ICMR of I²C, selects the transfer rate in master mode. For details, see section 25.2.4, I²C Bus Mode Register (ICMR).

Bit 5: I²C Controller Reset (IICRST)

This bit controls the initialization of the internal state of the I²C bus interface. When the I²C bus interface operating mode is hung because of communications error, and the IICRST bit is then set to 1, the I²C bus interface controller is initialized of the internal state, and this allows the internal state of the I²C bus interface to be initialized without making port settings or initializing registers.

For the detail, refer to section 25.3.9, Initialization of Internal State.

The initialization is continuous and the I²C bus interface cannot operate, when the IICST bit remains set to 1. Therefore, be sure to clear the IICST bit after setting it.

Bit 5

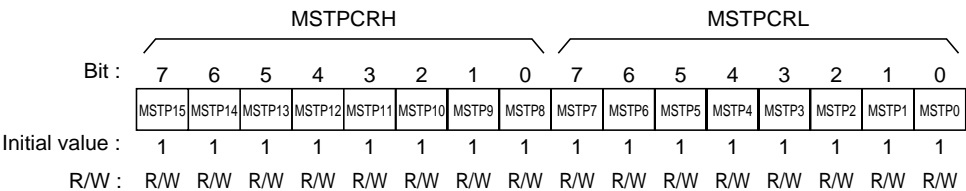
| IICRST | Description |
|--------|--|
| 0 | I ² C bus interface controller is not reset (Initial value) |
| 1 | I ² C bus interface controller is reset |

Bits 3: Flash Memory Control Resister Enable (FLSHE)

This bit selects the control resister of the flash memory. For details, refer to section 7.3.4 or 8.5.5, Serial Timer Control Resister.

Bits 4 and 2 to 0: Reserved

25.2.8 Module Stop Control Register (MSTPCR)



MSTPCR comprises two 8-bit readable/writable registers, and is used to perform module stop mode control.

When the corresponding bit in MSTPCR is set to 1, operation of the corresponding I²C module is halted at the end of the bus cycle, and a transition is made to module stop mode. For details, see section 4.5, Module Stop Mode.

MSTPCR is initialized to H'FFFF by a reset. It is not initialized in standby mode.

MSTPCRL Bit 6: Module Stop (MSTP6)

Specifies I²C module stop mode.

MSTPCRL

Bit 6

| MSTP6 | Description |
|-------|--|
| 0 | I ² C module stop mode is cleared |
| 1 | I ² C module stop mode is set (Initial value) (Initial value) |

25.3 Operation

25.3.1 I²C Bus Data Format

The I²C bus interface has serial and I²C bus formats.

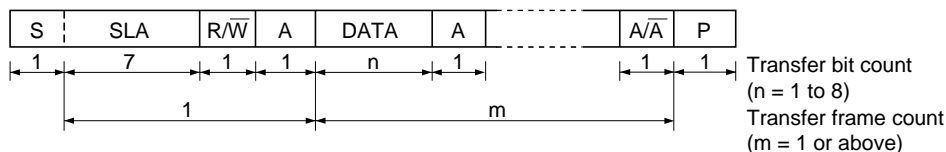
The I²C bus formats are addressing formats with an acknowledge bit. These are shown in figure 25.3. The first frame following a start condition always consists of 8 bits.

The serial format is a non-addressing format with no acknowledge bit. This is shown in figure 25.4.

Figure 25.5 shows the I²C bus timing.

The symbols used in figures 25.3 to 25.5 are explained in table 25.4.

(a) FS = 0 or FSX = 0



(b) Start condition transmission, FS = 0 or FSX = 0

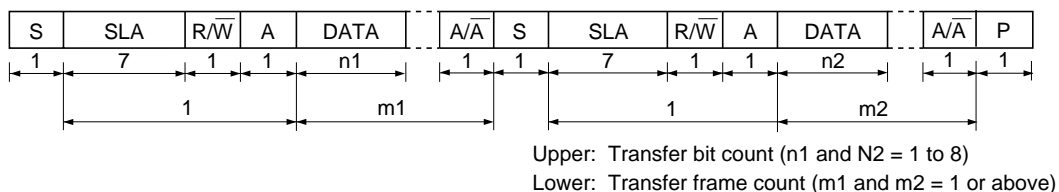


Figure 25.3 I²C Bus Data Formats (I²C Bus Formats)

FS = 1 and FSX = 1

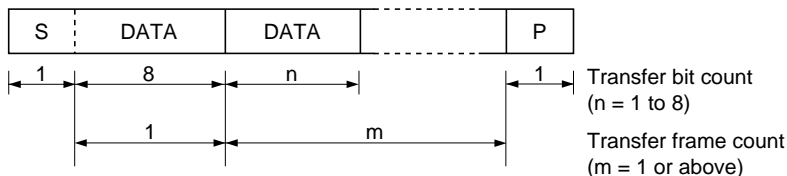


Figure 25.4 I²C Bus Data Format (Serial Format)

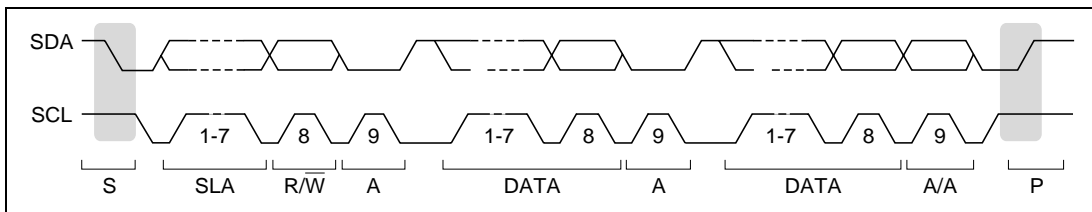


Figure 25.5 I²C Bus Timing

Table 25.4 I²C Bus Data Format Symbols

| | |
|-------------------|---|
| S | Start condition. The master device drives SDA from high to low while SCL is high |
| SLA | Slave address, by which the master device selects a slave device |
| R/ \overline{W} | Indicates the direction of data transfer: from the slave device to the master device when R/ \overline{W} is 1, or from the master device to the slave device when R/ \overline{W} is 0 |
| A | Acknowledge. The receiving device (the slave in master transmit mode, or the master in master receive mode) drives SDA low to acknowledge a transfer |
| DATA | Transferred data. The bit length is set by bits BC2 to BC0 in ICMR. The MSB-first or LSB-first format is selected by bit MLS in ICMR |
| P | Stop condition. The master device drives SDA from low to high while SCL is high |

25.3.2 Master Transmit Operation

In I²C bus format master transmit mode, the master device outputs the transmit clock and transmit data, and the slave device returns an acknowledge signal. The transmission procedure and operations synchronize with the ICDR writing are described below.

- [1] Set bit ICE in ICCR to 1. Set bits MLS, WAIT, and CKS2 to CKS0 in ICMR, and bit IICX in STCR, according to the operating mode.
- [2] Read the BBSY flag in ICCR to confirm that the bus is free.
- [3] Set bits MST and TRS to 1 in ICCR to select master transmit mode.
- [4] Write 1 to BBSY and 0 to SCP. This changes SDA from high to low when SCL is high, and generates the start condition.
- [5] Then IRIC and IRTR flags are set to 1. If the IEIC bit in ICCR has been set to 1, an interrupt request is sent to the CPU.
- [6] Write the data (slave address + R/ \overline{W}) to ICDR. After the start condition instruction has been issued and the start condition has been generated, write data to ICDR. If this procedure is not followed, data may not be output correctly. With the I²C bus format (when the FS bit in SAR or the FSX bit in SARX is 0), the first frame data following the start condition indicates the 7-bit slave address and transmit/receive direction. As indicating the end of the transfer, and so the IRIC flag is cleared to 0. After writing ICDR, clear IRIC immediately not to execute other interrupt handling routine. If one frame of data has been transmitted before the IRIC

clearing, it can not be determine the end of transmission. The master device sequentially sends the transmission clock and the data written to ICDR using the timing shown in figure 25.6. The selected slave device (i.e. the slave device with the matching slave address) drives SDA low at the 9th transmit clock pulse and returns an acknowledge signal.

- [7] When one frame of data has been transmitted, the IRIC flag is set to 1 at the rise of the 9th transmit clock pulse. After one frame has been transmitted SCL is automatically fixed low in synchronization with the internal clock until the next transmit data is written.
- [8] Read the ACKB bit in ICSR to confirm that ACKB is cleared to 0. When the slave device has not acknowledged (ACKB bit is 1), operate the step [12] to end transmission, and retry the transmit operation.
- [9] Write the transmit data to ICDR. As indicating the end of the transfer, and so the IRIC flag is cleared to 0. After writing ICDR, clear IRIC immediately not to execute other interrupt handling routine. The master device sequentially sends the transmission clock and the data written to ICDR. Transmission of the next frame is performed in synchronization with the internal clock.
- [10] When one frame of data has been transmitted, the IRIC flag is set to 1 at the rise of the 9th transmit clock pulse. After one frame has been transmitted SCL is automatically fixed low in synchronization with the internal clock until the next transmit data is written.
- [11] Read the ACKB bit in ICSR and confirm ACKB is cleared to 0. When there is data to be transmitted, go to the step [6] to continue next transmission. When the slave device has not acknowledged (ACKB bit is set to 1), operate the step [12] to end transmission.
- [12] Clear the IRIC flag to 0. And write 0 to BBSY and SCP in ICCR. This changes SDA from low to high when SCL is high, and generates the stop condition.

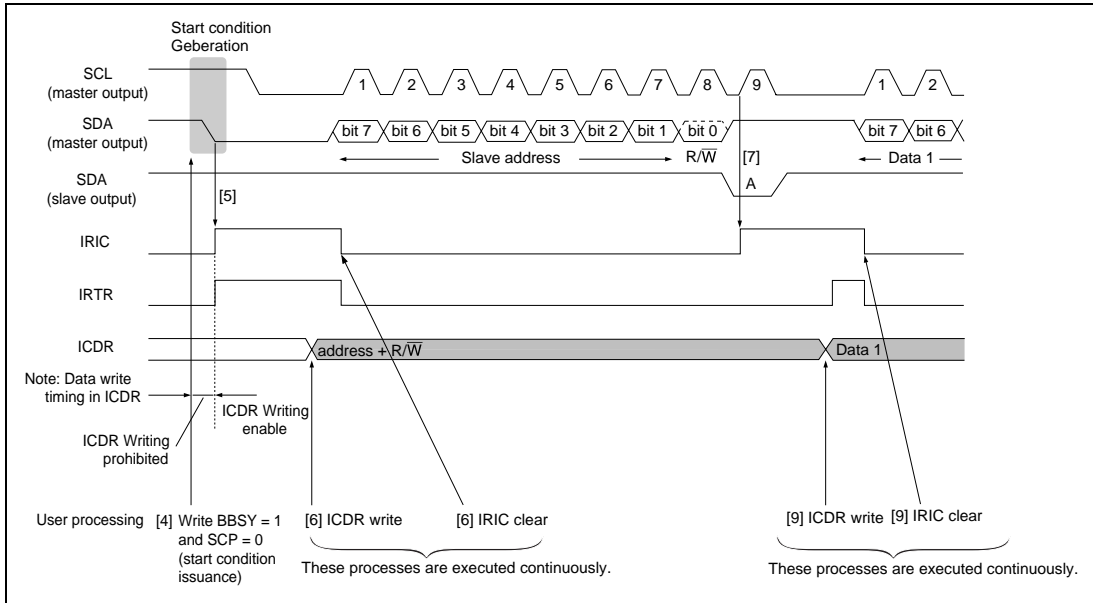


Figure 25.6 Example of Master Transmit Mode Operation Timing (MLS = WAIT = 0)

25.3.3 Master Receive Operation

In master receive mode, the master device outputs the receive clock, receives data, and returns an acknowledge signal. The slave device transmits data. I²C bus interface module consists of the data buffers of ICDRR and ICDRS, so data can be received continuously in master receive mode. For this construction, when stop condition issuing timing delayed, it may occurs the internal contention between stop condition issuance and SCL clock output for next data receiving, and then the extra SCL clock would be outputted automatically or the SCL line would be held to low. And for I²C bus interface system, the acknowledge bit must be set to 1 at the last data receiving, so the change timing of ACKB bit in ICSR should be controlled by software. To take measures against these problems, the wait function should be used in master receive mode. The reception procedure and operations with the wait function in master receive mode are described below.

- [1] Clear the TRS bit in ICCR to 0 to switch from transmit mode to receive mode, and set the WAIT bit in ICMR to 1. Also clear the ACKB bit in ICSR to 0 (acknowledge data setting).
- [2] When ICDR is read (dummy data read), reception is started, and the receive clock is output, and data received, in synchronization with the internal clock. In order to detect wait operation, set the IRIC flag in ICCR must be cleared to 0. After reading ICDR, clear IRIC immediately not to execute other interrupt handling routine. If one frame of data has been received before the IRIC clearing, it can not be determine the end of reception.
- [3] The IRIC flag is set to 1 at the fall of the 8th receive clock pulse. If the IEIC bit in ICCR has been set to 1, an interrupt request is sent to the CPU. SCL is automatically fixed low in synchronization with the internal clock until the IRIC flag clearing. If the first frame is the last receive data, execute step [10] to halt reception.
- [4] Clear the IRIC flag to release from the Wait State. The master device outputs the 9th clock and drives SDA at the 9th receive clock pulse to return an acknowledge signal.
- [5] When one frame of data has been received, the IRIC flag in ICCR and the IRTR flag in ICSR are set to 1 at the rise of the 9th receive clock pulse. The master device outputs SCL clock to receive next data.
- [6] Read ICDR.
- [7] Clear the IRIC flag to detect next wait operation. From clearing of the IRIC flag to negation of a wait as described in step [4] (and [9]) to clearing of the IRIC flag as described in steps [5], [6], and [7], must be performed within the time taken to transfer one byte.
- [8] The IRIC flags set to 1 at the fall of the 8th receive clock pulse. SCL is automatically fixed low in synchronization with the internal clock until the IRIC flag clearing. If this frame is the last receive data, execute step [10] to halt reception.
- [9] Clear the IRIC flag in ICCR to cancel wait operation. The master device outputs the 9th clock and drives SDA at the 9th receive clock pulse to return an acknowledge signal. Data can be received continuously by repeating step [5] to [9].

- [10] Set the ACKB bit in ICSR to 1 so as to return “No acknowledge” data. Also set the TRS bit to 1 to switch from receive mode to transmit mode.
- [11] Clear IRIC flag to 0 to release from the Wait State.
- [12] When one frame of data has been received, the IRIC flag is set to 1 at the rise of the 9th receive clock pulse.
- [13] Clear the WAIT bit to 0 to switch from wait mode to no wait mode. Read ICDR and the IRIC flag to 0. Clearing of the IRIC flag should be after the WAIT = 0.
- [14] Clear the BBSY bit and SCP bit to 0. This changes SDA from low to high when SCL is high, and generates the stop condition.

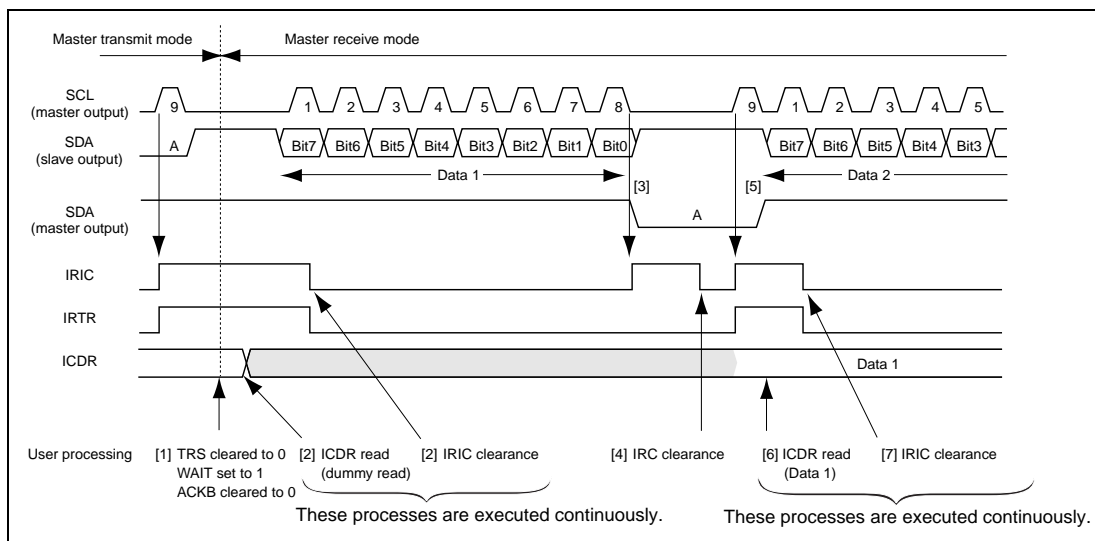


Figure 25.7 Example of Master Receive Mode Operation Timing
(MLS = ACKB = 0, WAIT = 1)

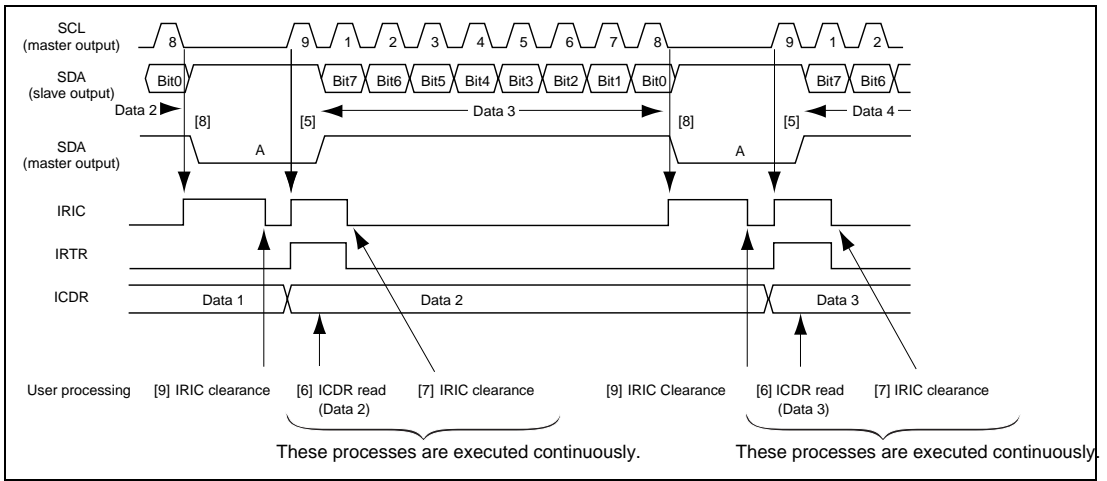


Figure 25.8 Example of Master Receive Mode Operation Timing
(MLS = ACKB = 0, WAIT = 1) continued

25.3.4 Slave Receive Operation

In slave receive mode, the master device outputs the transmit clock and transmit data, and the slave device returns an acknowledge signal. The receive procedure and operations in slave receive mode are described below.

- [1] Set bit ICE in ICCR to 1. Set bits MLS in ICMR and bits MST and TRS in ICCR according to the operating mode.
- [2] A start condition output by the master device sets the BBSY flag to 1 in ICCR.
- [3] After the slave device detects the start condition, if the first frame matches its slave address, it functions as the slave device designated as the master device. If the 8th bit data (R/\overline{W}) is 0, TRS bit in ICCR remains 0 and executes slave receive operation.
- [4] At the ninth clock pulse of the receive frame, the slave device drives SDA low to acknowledge the transfer. At the same time, the IRIC flag is set to 1 in ICCR. If IEIC is 1 in ICCR, a CPU interrupt is requested. If the RDRF internal flag is 0, it is set to 1 and continuous reception is performed. If the RDRF internal flag is 1, the slave device holds SCL low from the fall of the receive clock until it has read the data in ICDR.
- [5] Read ICDR and clear IRIC to 0 in ICCR. At this time, the RDRF flag is cleared to 0.

Steps [4] and [5] can be repeated to receive data continuously. When a stop condition is detected (a low-to-high transition of SDA while SCL is high), the BBSY flag is cleared to 0 in ICCR.

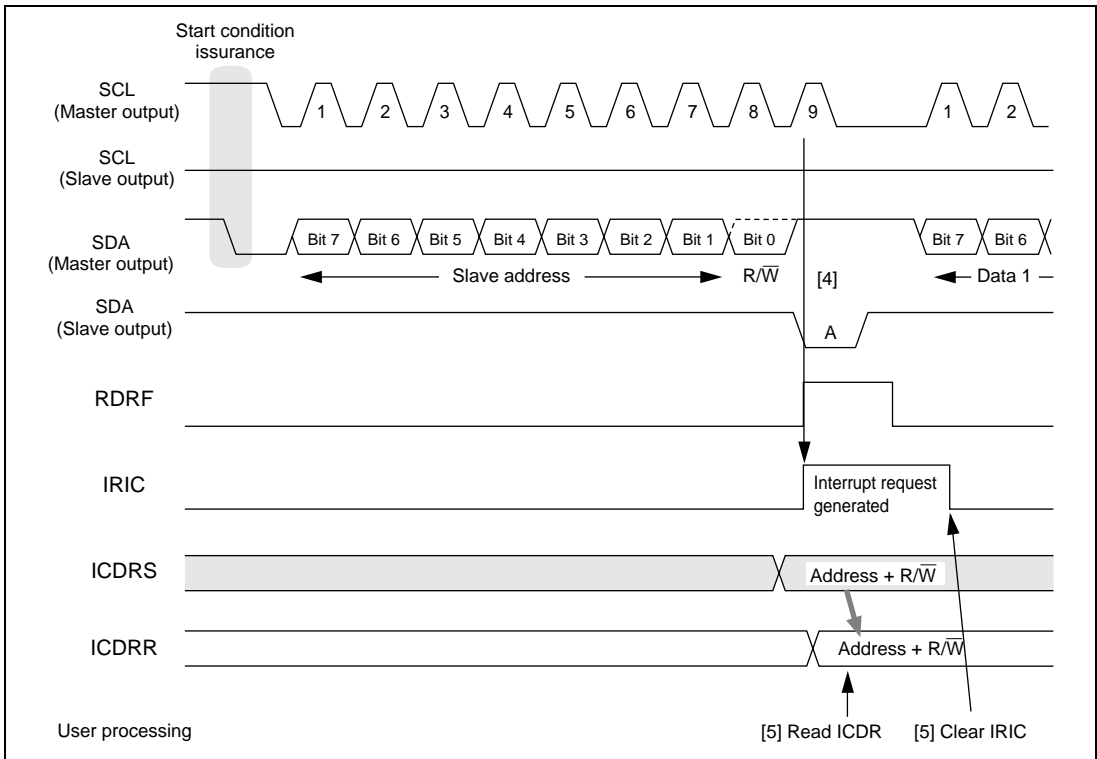


Figure 25.9 Example of Timing in Slave Receive Mode (MLS = ACKB = 0) (1)

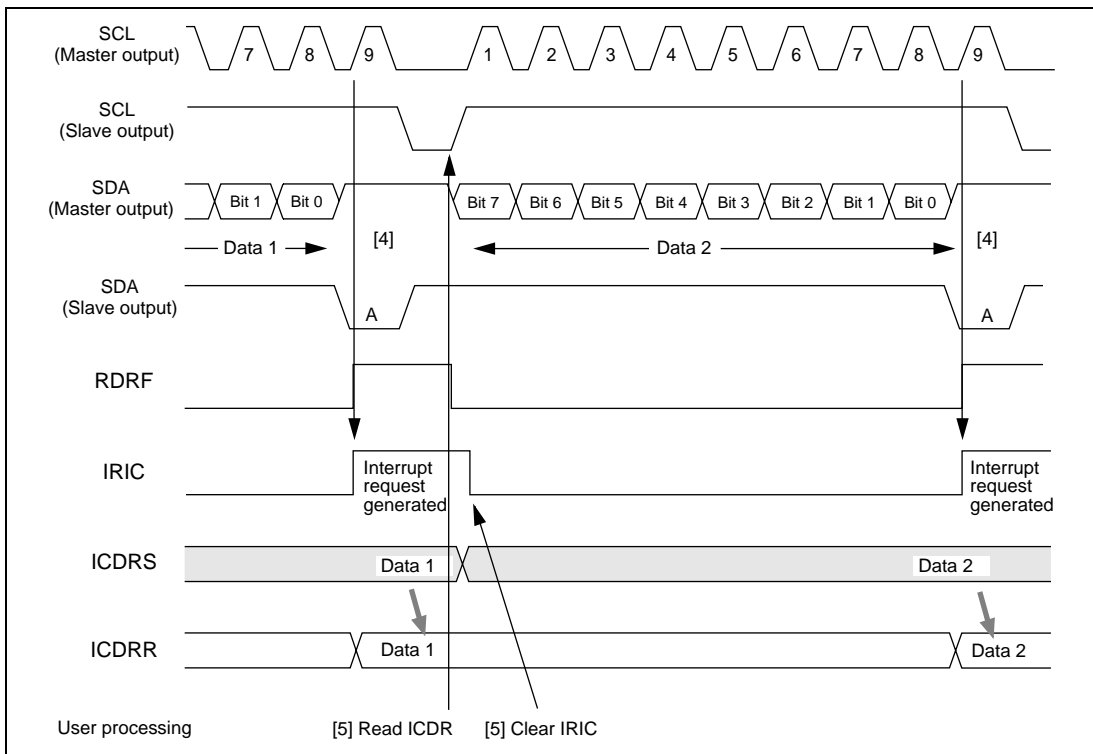


Figure 25.10 Example of Timing in Slave Receive Mode (MLS = ACKB = 0) (2)

25.3.5 Slave Transmit Operation

In slave transmit mode, the slave device outputs the transmit data, and the master device outputs the transmit clock and returns an acknowledge signal. The transmit procedure and operations in slave transmit mode are described below.

- [1] Set bit ICE in ICCR to 1. Set bits MLS in ICMR and bits MST and TRS in ICCR according to the operating mode.
- [2] After the slave device detects a start condition, if the first frame matches its slave address, at the ninth clock pulse the slave device drives SDA low to acknowledge the transfer. At the same time, the IRIC flag is set to 1 in ICCR, and if the IEIC bit in ICCR is set to 1 at this time, an interrupt request is sent to the CPU. If the eighth data bit (R/\bar{W}) is 1, the TRS bit is set to 1 in ICCR, automatically causing a transition to slave transmit mode. The slave device holds SCL low from the fall of the transmit clock until data is written in ICDR.
- [3] Clear the IRIC flag to 0, then write data in ICDR. The written data is transferred to ICDRS, and the TDRE internal flag and the IRIC and IRTR flags are set to 1 again. Clear IRIC to 0, then write the next data in ICDR. The slave device outputs the written data serially in step with the clock output by the master device, with the timing shown in figure 25.11.
- [4] When one frame of data has been transmitted, at the rise of the ninth transmit clock pulse IRIC is set to 1 in ICCR. If the TDRE internal flag is 1, the slave device holds SCL low from the fall of the transmit clock until data is written in ICDR. The master device drives SDA low at the ninth clock pulse to acknowledge the data. The acknowledge signal is stored in the ACKB bit in ICSR, and can be used to check whether the transfer was carried out normally. If TDRE internal flag is set to 0, the data written in ICDR is transferred to ICDRS, then transmission starts and TDRE internal flag and IRIC and IRTR flags are all set to 1 again.
- [5] To continue transmitting, clear IRIC to 0, then write the next transmit data in ICDR.

Steps [4] and [5] can be repeated to transmit continuously. To end the transmission, write H'FF in ICDR so that the SDA may be freed on the slave side. When a stop condition is detected (a low-to-high transition of SDA while SCL is high), the BBSY flag will be cleared to 0 in ICCR.

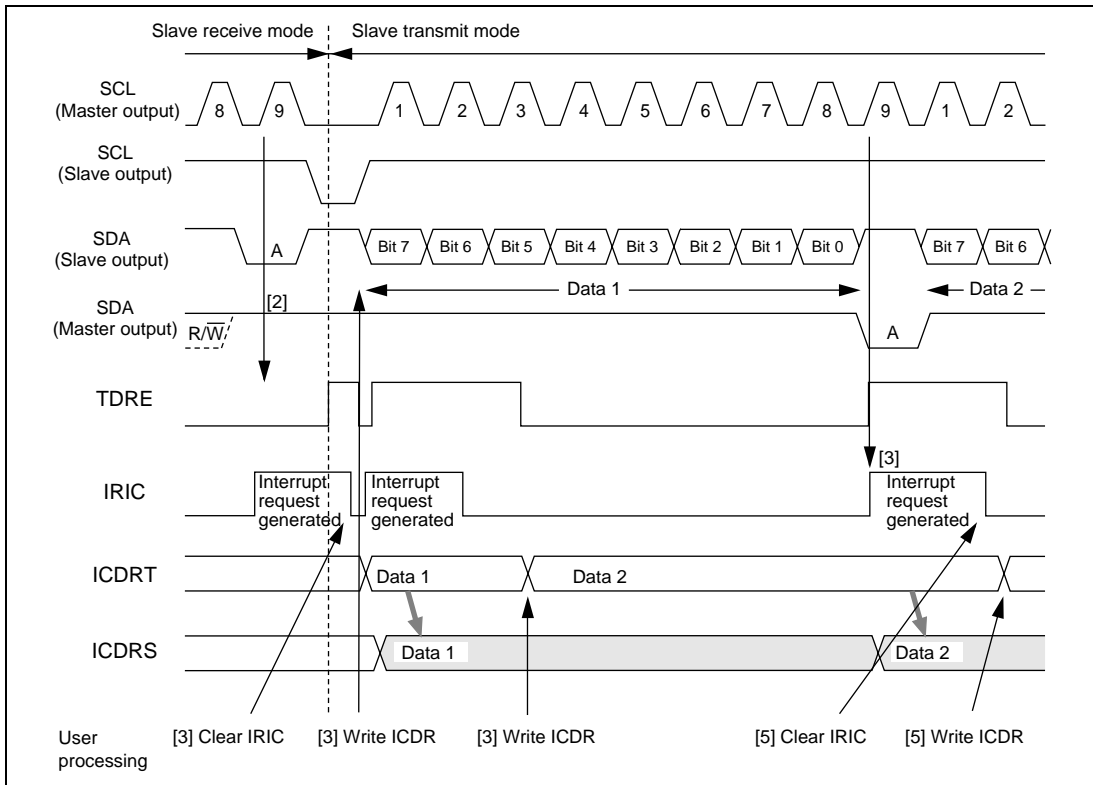
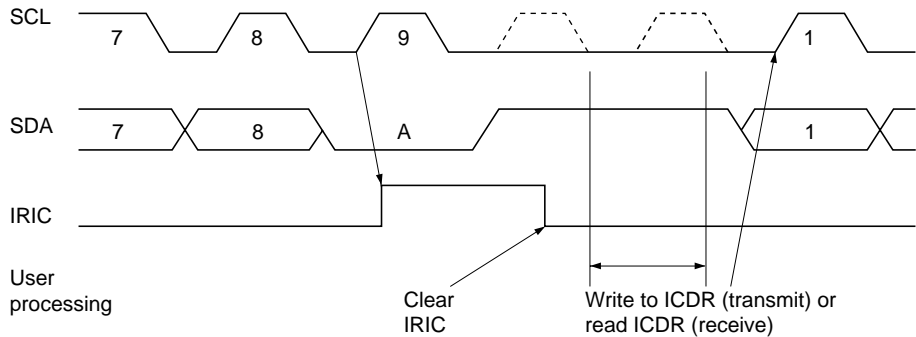


Figure 25.11 Example of Timing in Slave Transmit Mode (MLS = 0)

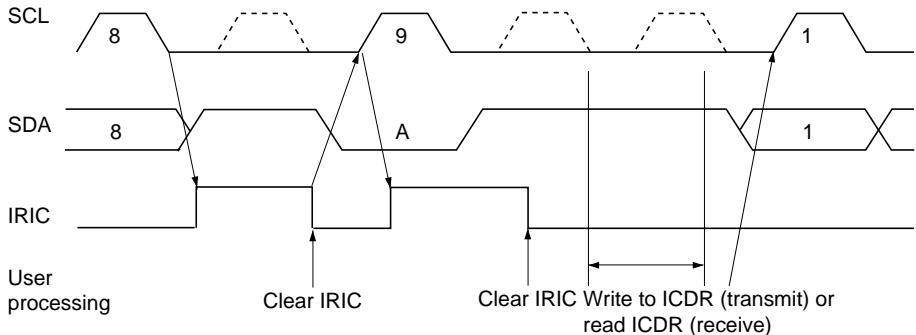
25.3.6 IRIC Setting Timing and SCL Control

The interrupt request flag (IRIC) is set at different times depending on the WAIT bit in ICMR, the FS bit in SAR, and the FSX bit in SARX. If the TDRE or RDRF internal flag is set to 1, SCL is automatically held low after one frame has been transferred; this timing is synchronized with the internal clock. Figure 25.12 shows the IRIC set timing and SCL control.

(a) When WAIT = 0, and FS = 0 or FSX = 0 (I²C bus format, no wait)



(b) When WAIT = 1, and FS = 0 or FSX = 0 (I²C bus format, wait inserted)



(c) When FS = 1 and FSX = 1 (synchronous serial format)

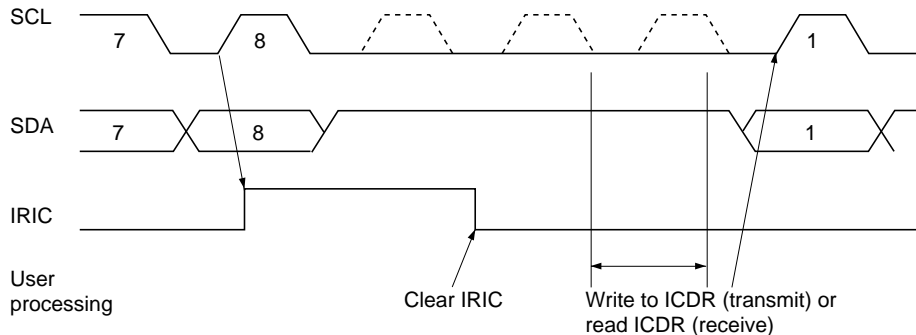


Figure 25.12 IRIC Setting Timing and SCL Control

25.3.7 Noise Canceler

The logic levels at the SCL and SDA pins are routed through noise cancelers before being latched internally. Figure 25.13 shows a block diagram of the noise canceler circuit.

The noise canceler consists of two cascaded latches and a match detector. The SCL (or SDA) input signal is sampled on the system clock, but is not passed forward to the next circuit unless the outputs of both latches agree. If they do not agree, the previous value is held.

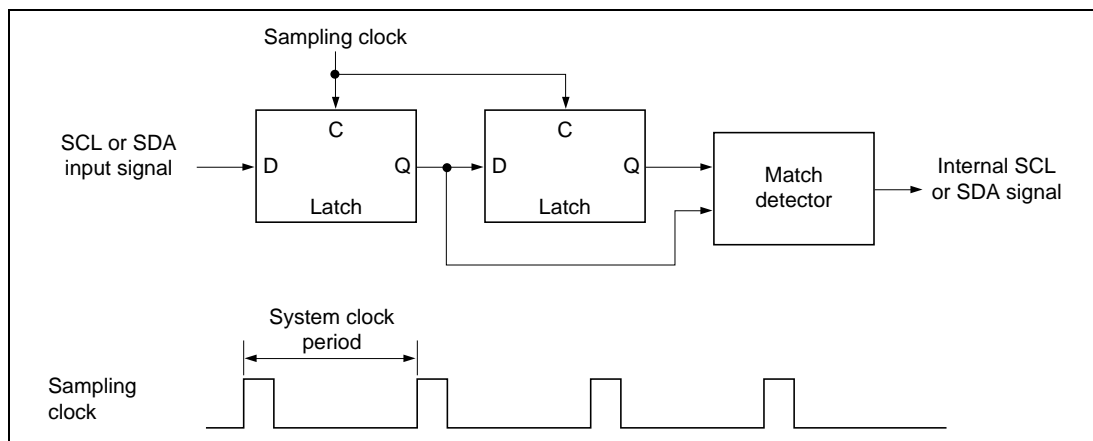


Figure 25.13 Block Diagram of Noise Canceler

25.3.8 Sample Flowcharts

Figures 25.14 to 25.17 show sample flowcharts for using the I²C bus interface in each mode.

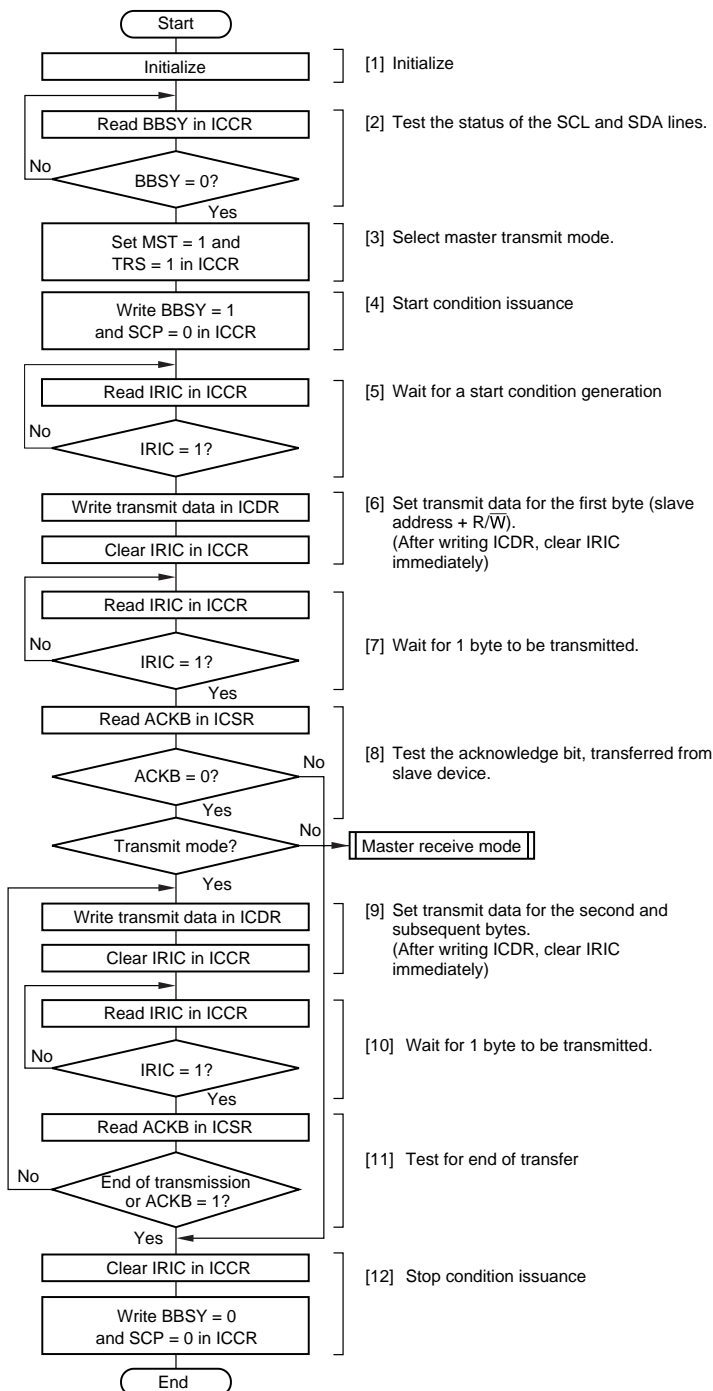


Figure 25.14 Flowchart for Master Transmit Mode (Example)

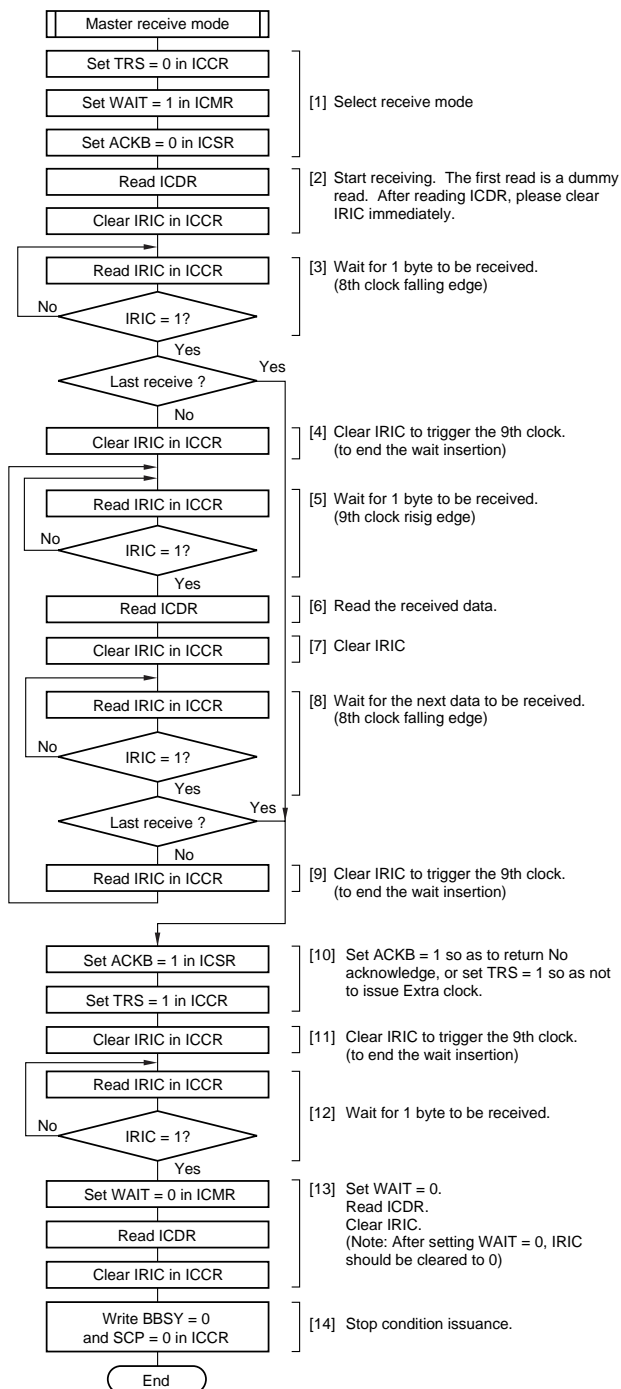


Figure 25.15 Flowchart for Master Receive Mode (Example)

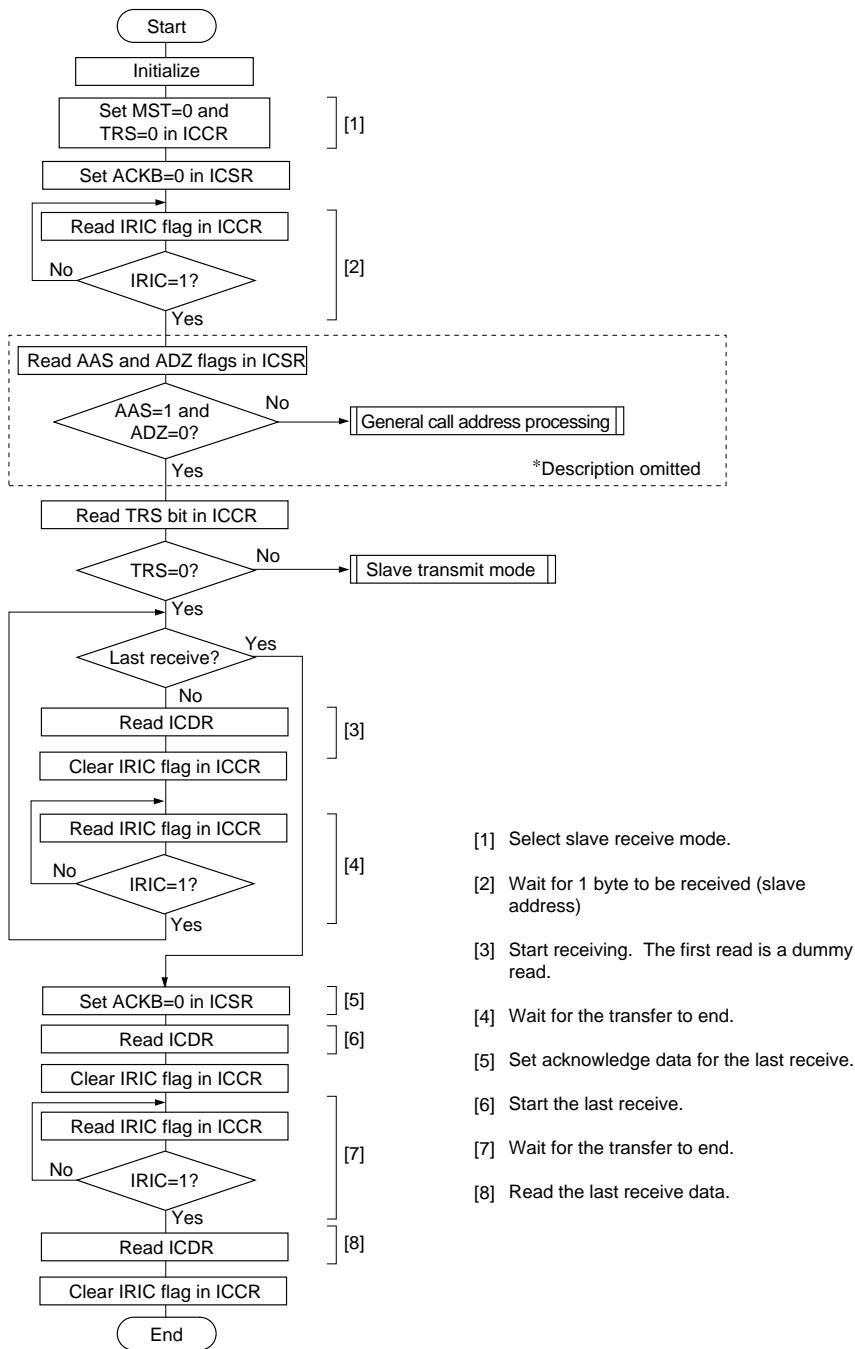
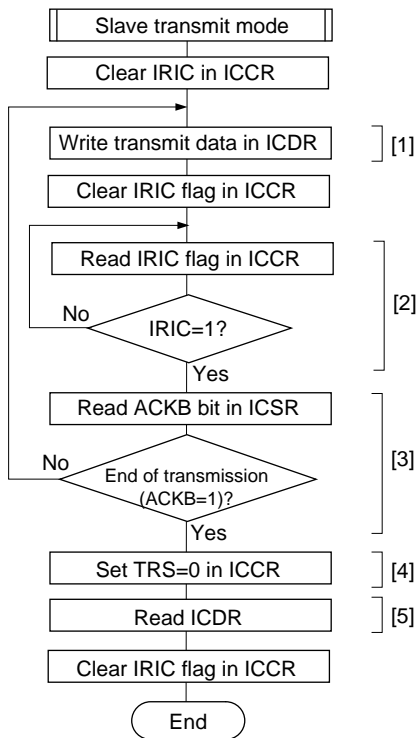


Figure 25.16 Flowchart for Slave Transmit Mode (Example)



- [1] Set transmit data for the second and subsequent bytes.
- [2] Wait for 1 byte to be transmitted.
- [3] Test for end of transfer.
- [4] Select slave receive mode.
- [5] Dummy read (to release the SCL line).

Figure 25.17 Flowchart for Slave Receive Mode (Example)

25.3.9 Initialization of Internal State

This I²C is capable of forcibly initializing internal state of I²C if deadlock develops during communication.

The initialization is done by setting IICRST bit in STCR register, or clearing ICE bit.

For details, see section 25.2.7, Serial/Time control Register (STCR).

(1) Range of Initialization

The following is initialized by this function:

- Internal flags of TDRE and RDRF
- Programmable logic controller for signal receiving and sending.
- Internal latches used for holding outputs from SCL and SDA pins (wait, clock, data output, etc.).

The following is not initialized by this function:

- Register values (ICDR, SAR, SARX, ICMR, ICCR, ICSR, and STCR).
- Internal latches employed for maintaining data read from the registers which is used for setting or clearing flags on ICMR, ICCR, and ICSR registers.
- Values on the ICMR register bit counters (BC2 to BC0).
- Interrupt factors currently generated (interrupt factors transferred to the interrupt controller).

(2) Precautions on Initialization

- Interrupt flags and interrupt factors are not cleared by this function. Thus, you need to clear them own as needed.
- Other register flags are not basically cleared, too. Thus, you need to clear them as needed.
- When this I²C is initialized with IICRST bit, write data specified by IICRST bit is maintained. When clearing I²C, set IICRST bit once, then clear it using the MOV instruction. The I²C cannot operate with the IICRST bit set to 1. Don't try to use bit operation instructions such as BCLR.
- If you try to clear a flag while data sending or receiving is taking place, I²C module stops sending or receiving at that moment and frees the SCL and SDA pins. When resuming the communication, initialize registers as needed so that the system communication capability may function as intended.

Clear function of this module does not directly rewrite value of BBSY bit. However, depending on state of SCL and SDA pins and the timing in which they are made free, BBSY bit can be cleared. Other bits and flags can also be affected by status change.

In order to avoid these troubles, the following procedures must be observed in initialization of I²C.

- (1) Implement initialization of internal state by setting IICRST bit or ICE bit.
- (2) Execute the stop condition issue instruction (setting BBSY = 0 and SCP = 0 to write) and wait for a duration equivalent to 2 clocks of the transfer rate.
- (3) Execute initialization of internal state again by setting IICRST bit or ICE bit.
- (4) Initialize each I²C register (re-setting).

25.4 Usage Notes

- (1) In master mode, if an instruction to generate a start condition is immediately followed by an instruction to generate a stop condition, neither condition will be output correctly. To output consecutive start and stop conditions, after issuing the instruction that generates the start condition, read the relevant ports, check that SCL and SDA are both low, then issue the instruction that generates the stop condition.
- (2) Either of the following two conditions will start the next transfer. Pay attention to these conditions when reading or writing to ICDR.
 - (a) Write access to ICDR when ICE = 1 and TRS = 1 (including automatic transfer from ICDR to ICDRT)
 - (b) Read access to ICDR when ICE = 1 and TRS = 0 (including automatic transfer from ICDR to ICDRR)
- (3) Table 25.5 shows the timing of SCL and SDA output in synchronization with the internal clock. Timings on the bus are determined by the rise and fall times of signals affected by the bus load capacitance, series resistance, and parallel resistance.

Table 25.5 I²C Bus Timing (SCL and SDA Output)

| Item | Symbol | Output Timing | Unit | Notes |
|--|--------------------|--|------|--------------------------|
| SCL output cycle time | t_{SCLC} | $28t_{\text{cyc}}$ to $256t_{\text{cyc}}$ | ns | Figure 28.10 (reference) |
| SCL output high pulse width | t_{SCLH} | $0.5t_{\text{SCLC}}$ | ns | |
| SCL output low pulse width | t_{SCLL} | $0.5t_{\text{SCLC}}$ | ns | |
| SDA output bus free time | t_{BUFO} | $0.5t_{\text{SCLC}} - 1t_{\text{cyc}}$ | ns | |
| Start condition output hold time | t_{STAHO} | $0.5t_{\text{SCLC}} - 1t_{\text{cyc}}$ | ns | |
| Retransmission start condition output setup time | t_{STASO} | $1t_{\text{SCLC}}$ | ns | |
| Stop condition output setup time | t_{STOSO} | $0.5t_{\text{SCLC}} + 2t_{\text{cyc}}$ | ns | |
| Data output setup time (master) | t_{SDASO} | $1t_{\text{SCLL}} - 3t_{\text{cyc}}$ | ns | |
| Data output setup time (slave) | | $1t_{\text{SCLL}} - (6t_{\text{cyc}} \text{ or } 12t_{\text{cyc}}^{*1})$ | ns | |
| Data output hold time | t_{SDAHO} | $3t_{\text{cyc}}$ | ns | |

Note: 1. $6t_{\text{cyc}}$ when IICX is 0, $12t_{\text{cyc}}$ when 1.

- (4) SCL and SDA input is sampled in synchronization with the internal clock. The AC timing therefore depends on the system clock cycle t_{cyc} , as shown in table 29.6 in section 29, Electrical Characteristics. Note that the I²C bus interface AC timing specifications will not be met with a system clock frequency of less than 5 MHz.

(5) The I²C bus interface specification for the SCL rise time t_{sr} is under 1000 ns (300 ns for high-speed mode). In master mode, the I²C bus interface monitors the SCL line and synchronizes one bit at a time during communication. If t_{sr} (the time for SCL to go from low to V_{IH}) exceeds the time determined by the input clock of the I²C bus interface, the high period of SCL is extended. The SCL rise time is determined by the pull-up resistance and load capacitance of the SCL line. To insure proper operation at the set transfer rate, adjust the pull-up resistance and load capacitance so that the SCL rise time does not exceed the values given in table 25.6.

Table 25.6 Permissible SCL Rise Time (t_{sr}) Values

| IICX | t_{cyc} Indication | Time Indication [ns] | | | | |
|------|-------------------------|----------------------|---|------------------------|------------------------|-------------------------|
| | | | I ² C Bus Specification (Max.) | $\phi = 5 \text{ MHz}$ | $\phi = 8 \text{ MHz}$ | $\phi = 10 \text{ MHz}$ |
| 0 | $7.5t_{cyc}$ | Normal mode | 1000 | ← | 937 | 750 |
| | | High-speed mode | 300 | ← | ← | ← |
| 1 | $17.5t_{cyc}$ | Normal mode | 1000 | ← | ← | ← |
| | | High-speed mode | 300 | ← | ← | ← |

(6) The I²C bus interface specifications for the SCL and SDA rise and fall times are under 1000 ns and 300 ns. The I²C bus interface SCL and SDA output timing is prescribed by t_{scyc} and t_{cyc} , as shown in table 25.5. However, because of the rise and fall times, the I²C bus interface specifications may not be satisfied at the maximum transfer rate. Table 25.7 shows output timing calculations for different operating frequencies, including the worst-case influence of rise and fall times.

t_{BUFO} fails to meet the I²C bus interface specifications at any frequency. The solution is either (a) to provide coding to secure the necessary interval (approximately 1 μ s) between issuance of a stop condition and issuance of a start condition, or (b) to select devices whose input timing permits this output timing for use as slave devices connected to the I²C bus.

t_{SCLLO} in high-speed mode and t_{STASO} in standard mode fail to satisfy the I²C bus interface specifications for worst-case calculations of t_{sr}/t_{sf} . Possible solutions that should be investigated include (a) adjusting the rise and fall times by means of a pull-up resistor and capacitive load, (b) reducing the transfer rate to meet the specifications, or (c) selecting devices whose input timing permits this output timing for use as slave devices connected to the I²C bus.

Table 25.7 I²C Bus Timing (with Maximum Influence of t_{sr}/t_{st})

| Item | t_{cyc} Indication | Time Indication (at Maximum Transfer Rate) [ns] | | | | | |
|-------------------------|--|---|--|---|------------------------|------------------------|-------------------------|
| | | | t_{sr}/t_{st} Influence (Max.) | I ² C Bus Specification (Min.) | $\phi = 5 \text{ MHz}$ | $\phi = 8 \text{ MHz}$ | $\phi = 10 \text{ MHz}$ |
| t_{SCLHO} | $0.5t_{SCLO}$ ($-t_{sr}$) | Normal mode | -1000 | 4000 | 4000 | ← | ← |
| | | High-speed mode | -300 | 600 | 950 | ← | ← |
| t_{SCLLO} | $0.5t_{SCLO}$ ($-t_{st}$) | Normal mode | -250 | 4700 | 4750 | ← | ← |
| | | High-speed mode | -250 | 1300 | 1000 ^{*1} | ← | ← |
| t_{BUFO} | $0.5t_{SCLO}-1t_{cyc}$ ($-t_{sr}$) | Normal mode | -1000 | 4700 | 3800 ^{*1} | 3875 ^{*1} | 3900 ^{*1} |
| | | High-speed mode | -300 | 1300 | 750 ^{*1} | 825 ^{*1} | 850 ^{*1} |
| t_{STAHO} | $0.5t_{SCLO}-1t_{cyc}$ ($-t_{st}$) | Normal mode | -250 | 4000 | 4550 | 4625 | 4650 |
| | | High-speed mode | -250 | 600 | 800 | 875 | 900 |
| t_{STASO} | $1t_{SCLO}$ ($-t_{sr}$) | Normal mode | -1000 | 4700 | 9000 | 9000 | 9000 |
| | | High-speed mode | -300 | 600 | 2200 | 2200 | 2200 |
| t_{STOSO} | $0.5t_{SCLO}+2t_{cyc}$ ($-t_{st}$) | Normal mode | -1000 | 4000 | 4400 | 4250 | 4200 |
| | | High-speed mode | -300 | 600 | 1350 | 1200 | 1150 |
| t_{SDASO} (master) | $1t_{SCLLO}^{*3}-3t_{cyc}$ ($-t_{sr}$) | Normal mode | -1000 | 250 | 3100 | 3325 | 3400 |
| | | High-speed mode | -300 | 100 | 400 | 625 | 700 |
| t_{SDASO} (slave) | $1t_{SCLL}^{*3}-12t_{cyc}^{*2}$ ($-t_{sr}$) | Normal mode | -1000 | 250 | 1300 | 2200 | 2500 |
| | | High-speed mode | -300 | 100 | -1400 ^{*1} | -500 ^{*1} | -200 ^{*1} |
| t_{SDAHO} | $3t_{cyc}$ | Normal mode | 0 | 0 | 600 | 375 | 300 |
| | | High-speed mode | 0 | 0 | ↑ | ↑ | ↑ |

Notes: 1. Does not meet the I²C bus interface specification. Remedial action such as the following is necessary: (a) secure a start/stop condition issuance interval; (b) adjust the rise and fall times by means of a pull-up resistor and capacitive load; (c) reduce the transfer rate; (d) select slave devices whose input timing permits this output timing.

The values in the above table will vary depending on the settings of the IICX bit and bits CKS0 to CKS2. Depending on the frequency it may not be possible to achieve the maximum transfer rate; therefore, whether or not the I²C bus interface specifications are met must be determined in accordance with the actual setting conditions.

- Value when the IICX bit is set to 1. When the IICX bit is cleared to 0, the value is ($t_{SCLL} - 6t_{cyc}$).
- Calculated using the I²C bus specification values (standard mode: 4700 ns min.; high-speed mode: 1300 ns min.).

(7) Precautions on reading ICDR at the end of master receive mode

When terminating the master receive mode, set TRS bit to 1, and select "write" for ICCR BBSY = 0 and SCP = 0. This forces to move SDA from low to high level when SCL is at high level, thereby generating the stop condition.

Now you can read received data from ICDR. If, however, any data is remaining on the buffer, received data on ICDRS is not transferred to ICDR, thus you won't be able to read the second byte data.

When it is required to read the second byte data, issue the stop condition from the master receive state (TRS bit is 0).

Before reading data from ICDR register, make sure that BBSY bit on ICCR register is 0, stop condition is generated and bus is made free.

If you try to read received data after the stop condition issue instruction (setting ICCR's BBSY = 0 and SCP = 0 to write) has been executed but before the actual stop condition is generated, clock may not be appropriately signaled when the next master sending mode is turned on. Thus, reasonable care is needed for determining when to read the received data.

After the master receive is complete, if you want to re-write I²C control bit (such as clearing MST bit) for switching the sending/receiving mode or modifying settings, it must be done during period (a) indicated in figure 25.18 (after making sure ICCR register BBSY bit is cleared to 0).

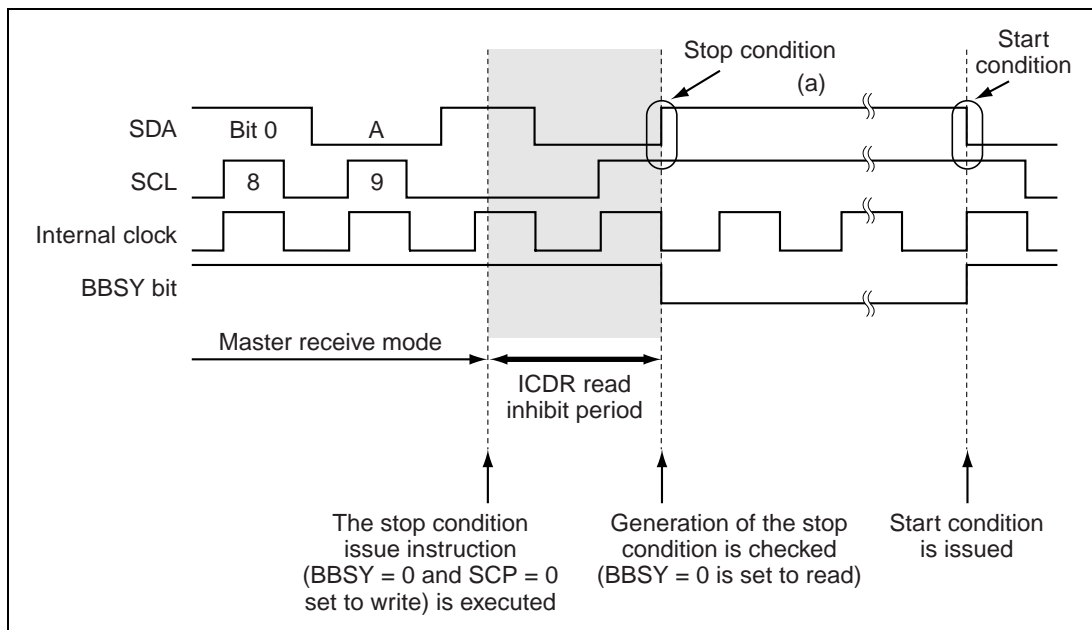


Figure 25.18 Precautions on Reading the Master Receive Data

(8) Notes on Start Condition Issuance for Retransmission

Figure 25.19 shows the timing of start condition issuance for retransmission, and the timing for subsequently writing data to ICDR, together with the corresponding flowchart. After start condition issuance is done and determined the start condition, write the transmit data to ICDR.

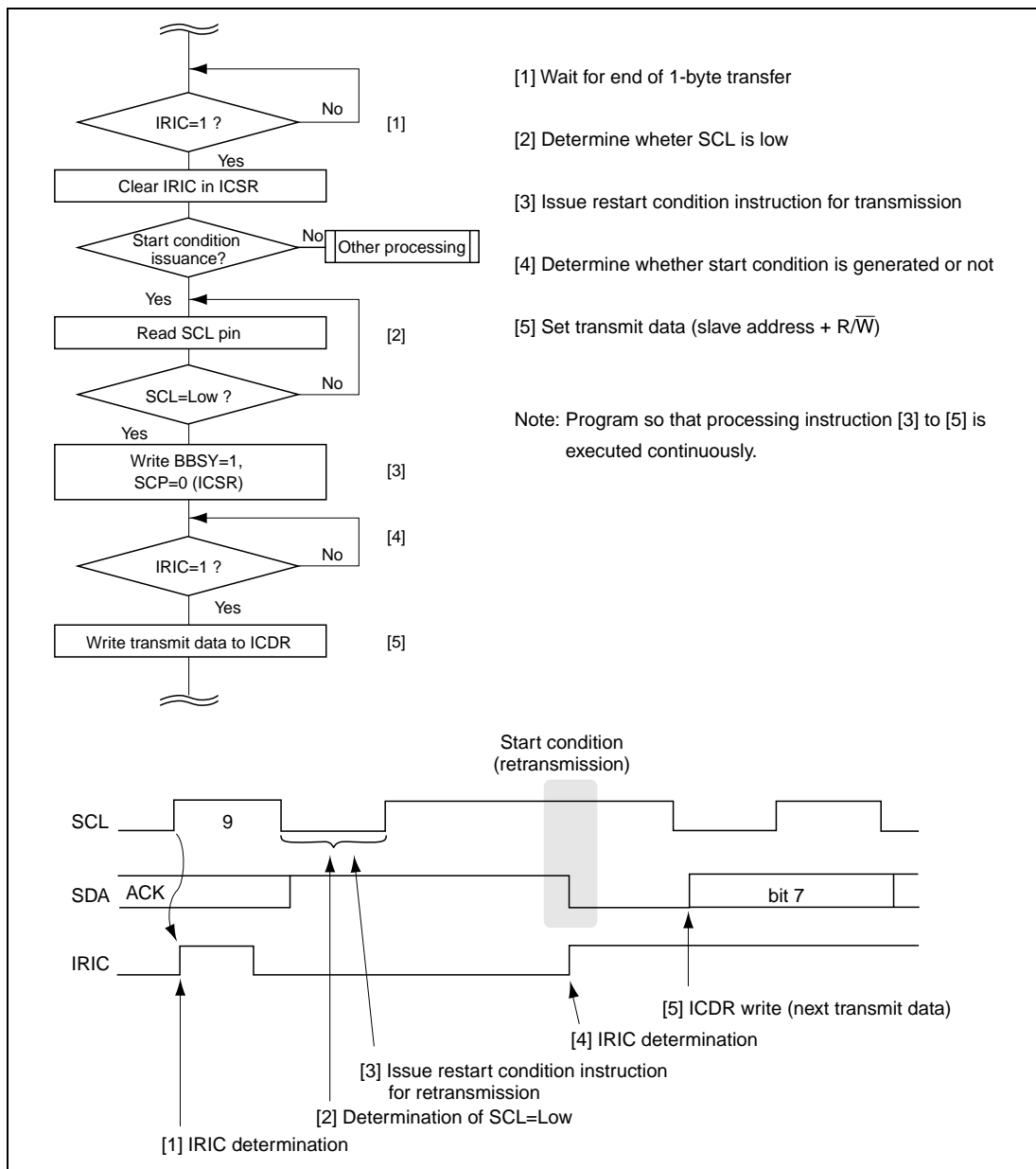


Figure 25.19 Flowchart and Timing of Start Condition Instruction Issuance for Retransmission

(9) Notes on I²C Bus Interface Stop Condition Instruction Issuance

If the rise time of the 9th SCL acknowledge exceeds the specification because the bus load capacitance is large, or if there is a slave device of the type that drives SCL low to effect a wait, issue the stop condition instruction after reading SCL and determining it to be low, as shown below.

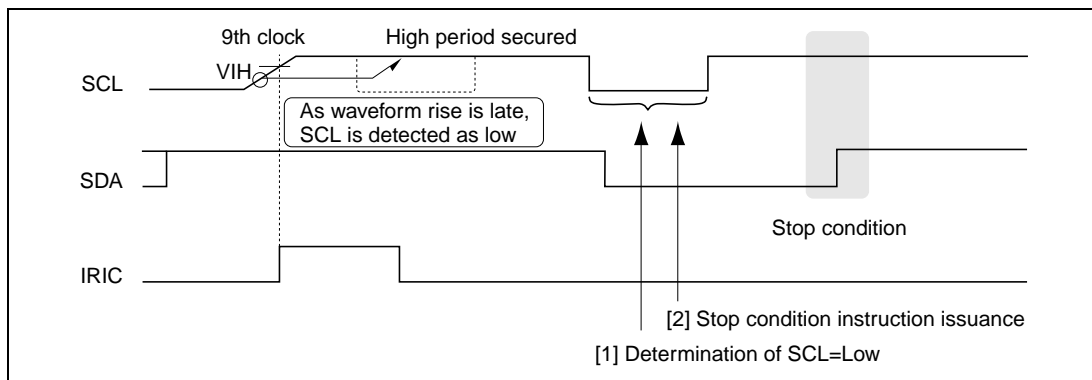


Figure 25.20 Timing of Stop Condition Issuance

Section 26 A/D Converter

26.1 Overview

This LSI incorporates a 10-bit successive-approximations A/D converter that allows up to 12 analog input channels to be selected.

26.1.1 Features

A/D converter features are listed below.

- 10-bit resolution
- 12 input channels
- Sample and hold function
- Choice of software, hardware (internal signal) triggering or external triggering for A/D conversion start.
- A/D conversion end interrupt request generation

26.1.2 Block Diagram

Figure 26.1 shows a block diagram of the A/D converter.

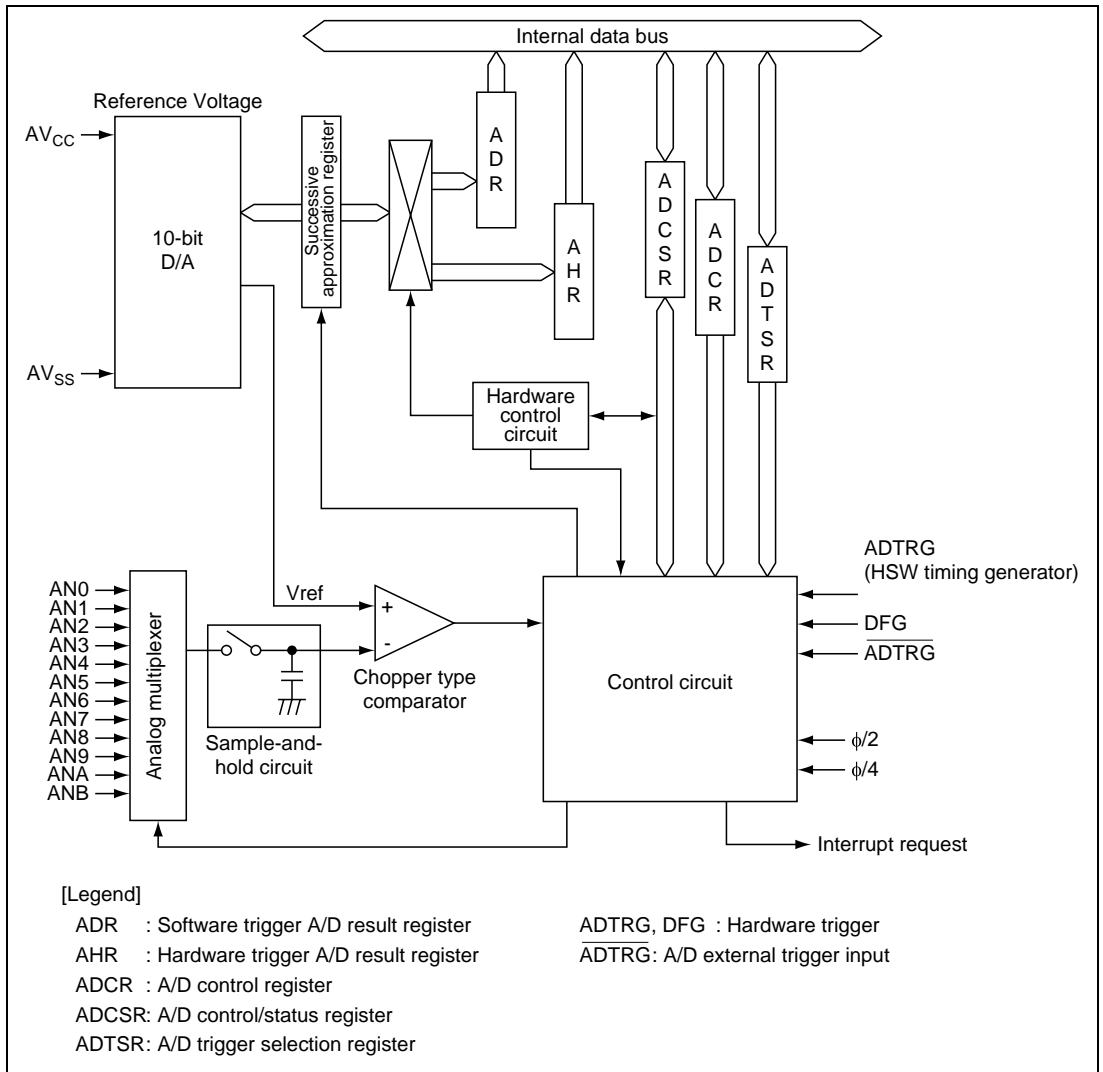


Figure 26.1 Block Diagram of A/D Converter

26.1.3 Pin Configuration

Table 26.1 summarizes the input pins used by the A/D converter.

Table 26.1 A/D Converter Pins

| Name | Abbrev. | I/O | Function |
|--------------------------------|--------------------|-------|--|
| Analog power supply pin | AV_{cc} | Input | Analog block power supply |
| Analog ground pin | AV_{ss} | Input | Analog block ground and A/D conversion reference voltage |
| Analog input pin 0 | AN0 | Input | Analog input channel 0 |
| Analog input pin 1 | AN1 | Input | Analog input channel 1 |
| Analog input pin 2 | AN2 | Input | Analog input channel 2 |
| Analog input pin 3 | AN3 | Input | Analog input channel 3 |
| Analog input pin 4 | AN4 | Input | Analog input channel 4 |
| Analog input pin 5 | AN5 | Input | Analog input channel 5 |
| Analog input pin 6 | AN6 | Input | Analog input channel 6 |
| Analog input pin 7 | AN7 | Input | Analog input channel 7 |
| Analog input pin 8 | AN8 | Input | Analog input channel 8 |
| Analog input pin 9 | AN9 | Input | Analog input channel 9 |
| Analog input pin A | ANA | Input | Analog input channel A |
| Analog input pin B | ANB | Input | Analog input channel B |
| A/D external trigger input pin | \overline{ADTRG} | Input | External trigger input for starting A/D conversion |

26.1.4 Register Configuration

Table 26.2 summarizes the registers of the A/D converter.

Table 26.2 A/D Converter Registers

| Name | Abbrev. | R/W | Size | Initial Value | Address^{*2} |
|--|----------------|---------------------|-------------|----------------------|-----------------------------|
| Software trigger A/D result register H | ADRH | R | Byte | H'00 | H'D130 |
| Software trigger A/D result register L | ADRL | R | Byte | H'00 | H'D131 |
| Hardware trigger A/D result register H | AHRH | R | Byte | H'00 | H'D132 |
| Hardware trigger A/D result register L | AHRL | R | Byte | H'00 | H'D133 |
| A/D control register | ADCR | R/W | Byte | H'40 | H'D134 |
| A/D control/status register | ADCSR | R (W) ^{*1} | Byte | H'01 | H'D135 |
| A/D trigger selection register | ADTSR | R/W | Byte | H'FC | H'D136 |
| Port mode register 0 | PMR0 | R/W | Byte | H'00 | H'FFCD |

Notes: 1. Only 0 can be written in bits 7 and 6, to clear the flag. Bits 3 to 1 are read-only.

2. Lower 16 bits of the address.

26.2 Register Descriptions

26.2.1 Software-Triggered A/D Result Register (ADR)

| ADRH | | | | | | | | ADRL | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ADR9 | ADR8 | ADR7 | ADR6 | ADR5 | ADR4 | ADR3 | ADR2 | ADR1 | ADR0 | — | — | — | — | — | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R | R | R | — | — | — | — | — | — |

The software-triggered A/D result register (ADR) is a register that stores the result of an A/D conversion started by software.

The A/D-converted data is 10-bit data. Upon completion of software-triggered A/D conversion, the 10-bit result data is transferred to ADR and the data is retained until the next software-triggered A/D conversion completion. The upper 8 bits of the data are stored in the upper bytes (bits 15 to 8) of ADR, and the lower 2 bits are stored in the lower bytes (bits 7 and 6). Bits 5 to 0 are always read as 0.

ADR can be read by the CPU at any time, but the ADR value during A/D conversion is not fixed. The upper bytes can always be read directly, but the data in the lower bytes is transferred via a temporary register (TEMP). For details, see section 26.3, Interface to Bus Master.

ADR is a 16-bit read-only register which is initialized to H'0000 at a reset, and in module stop mode, standby mode, watch mode, subactive mode and subsleep mode.

26.2.2 Hardware-Triggered A/D Result Register (AHR)

| AHRH | | | | | | | | AHL | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | AHR9 | AHR8 | AHR7 | AHR6 | AHR5 | AHR4 | AHR3 | AHR2 | AHR1 | AHR0 | — | — | — | — | — | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R | R | R | — | — | — | — | — | — |

The hardware-triggered A/D result register (AHR) is a register that stores the result of an A/D conversion started by hardware (internal signal: ADTRG and DFG) or by external trigger input ($\overline{\text{ADTRG}}$).

The A/D-converted data is 10-bit data. Upon completion of hardware- or external-triggered A/D conversion, the 10-bit result data is transferred to AHR and the data is retained until the next hardware- or external- triggered A/D conversion completion. The upper 8 bits of the data are stored in the upper bytes (bits 15 to 8) of AHR, and the lower 2 bits are stored in the lower bytes (bits 7 and 6). Bits 5 to 0 are always read as 0.

AHR can be read by the CPU at any time, but the AHR value during A/D conversion is not fixed. The upper bytes can always be read directly, but the data in the lower bytes is transferred via a temporary register (TEMP). For details, see section 26.3, Interface to Bus Master. AHR is a 16-bit read-only register which is initialized to H'0000 at a reset, and in module stop mode, standby mode, watch mode, subactive mode and subsleep mode.

26.2.3 A/D Control Register (ADCR)

| | | | | | | | | |
|-----------------|-----|---|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CK | — | HCH1 | HCH0 | SCH3 | SCH2 | SCH1 | SCH0 |
| Initial value : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | — | R/W | R/W | R/W | R/W | R/W | R/W |

ADCR is a register that sets A/D conversion speed and selects analog input channel. When executing ADCR setting, make sure that the SST and HST flags in ADCSR is set to 0. ADCR is an 8-bit readable/writable register that is initialized to H'40 by a reset, and in module stop mode, standby mode, watch mode, subactive mode and subsleep mode.

Bit 7: Clock Select (CK)
Sets A/D conversion speed.

Bit 7

| CK | Description |
|----|--|
| 0 | Conversion frequency is 266 states (Initial value) |
| 1 | Conversion frequency is 134 states |

Note: A/D conversion starts when 1 is written in SST, or when HST is set to 1. The conversion period is the time from when this start flag is set until the flag is cleared at the end of conversion. Actual sample-and-hold takes place (repeatedly) during the conversion frequency shown in figure 26.2.

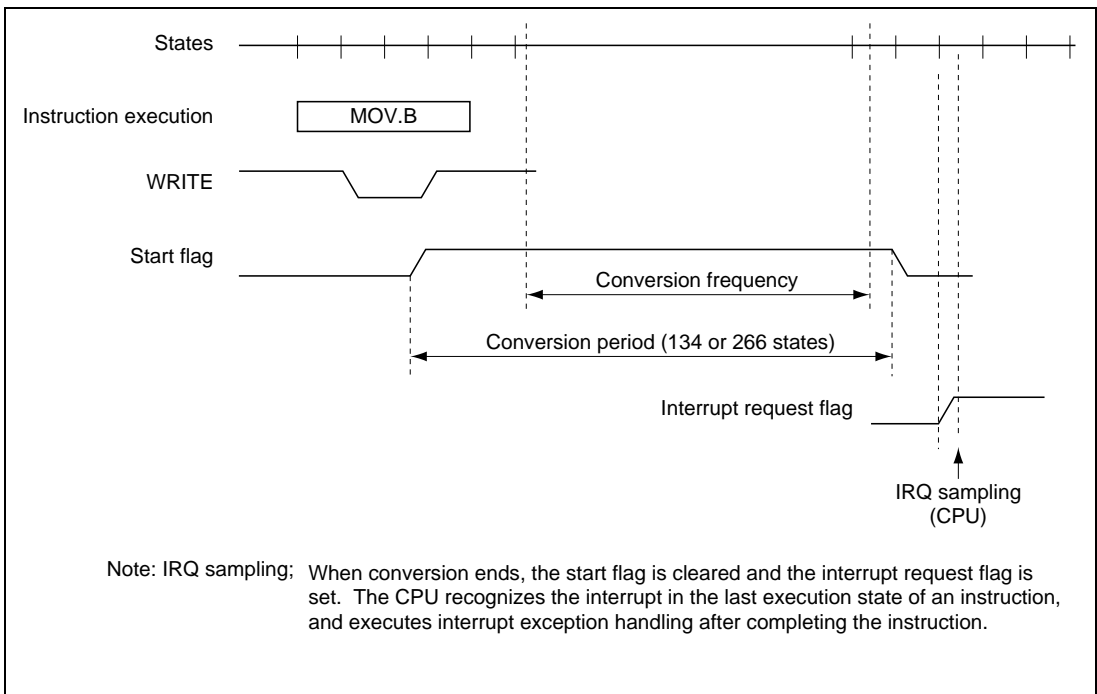


Figure 26.2 Internal Operation of A/D Converter

Bit 6: Reserved

This bit cannot be modified and always reads 1. Writes are disabled.

Bits 5 and 4: Hardware Channel Select (HCH1, HCH0)

These bits select the analog input channel that is converted by hardware triggering or triggering by an external input. Only channels AN8 to ANB are available for hardware- or external-triggered conversion.

| Bit 5 | Bit 4 | Analog Input Channel |
|-------|-------|----------------------|
| HCH1 | HCH0 | |
| 0 | 0 | AN8 (Initial value) |
| | 1 | AN9 |
| 1 | 0 | ANA |
| | 1 | ANB |

Bits 3 to 0: Software Channel Select (SCH3 to SCH0)

These bits select the analog input channel that is converted by software triggering.

When channels AN0 to AN7 are used, appropriate pin settings must be made in port mode register 0 (PMR0). For pin settings, see section 26.2.6, Port Mode Register 0 (PMR0).

| Bit 3 | Bit 2 | Bit 1 | Bit 0 | Analog Input Channel | |
|-------|-------|-------|-------|---|-----------------|
| SCH3 | SCH2 | SCH1 | SCH0 | | |
| 0 | 0 | 0 | 0 | AN0 | (Initial value) |
| | | | 1 | AN1 | |
| | | 1 | 0 | AN2 | |
| | | | 1 | AN3 | |
| | 1 | 0 | 0 | AN4 | |
| | | | 1 | AN5 | |
| | | 1 | 0 | AN6 | |
| | | | 1 | AN7 | |
| 1 | 0 | 0 | 0 | AN8 | |
| | | | 1 | AN9 | |
| | | 1 | 0 | ANA | |
| | | | 1 | ANB | |
| | 1 | * | * | No channel selected for software-triggered conversion | |

Notes: 1. If conversion is started by software when SCH3 to SCH0 are set to 11**, the conversion result is undetermined. Hardware- or external-triggered conversion, however, will be performed on the channel selected by HCH1 and HCH0.

2. *: Don't care.

26.2.4 A/D Control/Status Register (ADCSR)

| | | | | | | | | |
|-----------------|--------|--------|------|-----|-----|------|------|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | SEND | HEND | ADIE | SST | HST | BUSY | SCNL | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W : | R/(W)* | R/(W)* | R/W | R/W | R | R | R | — |

Note: * Only 0 can be written to bits 7 and 6, to clear the flag.

The A/D status register (ADCSR) is an 8-bit register that can be used to start or stop A/D conversion, or check the status of the A/D converter.

A/D conversion starts when 1 is written in SST flag. A/D conversion can also start by setting HST flag to 1 by hardware- or external-triggering.

For ADTRG start by HSW timing generator in hardware triggering, see section 28.4, HSW Timing Generator.

When conversion ends, the converted data is stored in the software-triggered A/D result register (ADR) or hardware-triggered A/D result register (AHR), and the SST or HST bit is cleared to 0.

If software-triggering and hardware- or external-triggering are generated at the same time, priority is given to hardware- or external-triggering.

ADCSR is an 8-bit register which is initialized to H'01 by a reset, and in module stop mode, standby mode, watch mode, subactive mode and subsleep mode.

Bit 7: Software A/D End Flag (SEND)

Indicates the end of A/D conversion.

Bit 7

| SEND | Description |
|------|---|
| 0 | [Clearing Conditions] 0 is written after reading 1 (Initial value) |
| 1 | [Setting Conditions] Software-triggered A/D conversion has ended |

Bit 6: Hardware A/D End Flag (HEND)

Indicates that hardware- or external-triggered A/D conversion has ended.

Bit 6

| HEND | Description |
|------|--|
| 0 | [Clearing Conditions] 0 is written after reading (Initial value) |
| 1 | [Setting Conditions] Hardware- or external-triggered A/D conversion has ended |

Bit 5: A/D Interrupt Enable (ADIE)

Selects enable or disable of interrupt (ADI) generation upon A/D conversion end.

Bit 5

| ADIE | Description |
|------|---|
| 0 | Interrupt (ADI) upon A/D conversion end is disabled (Initial value) |
| 1 | Interrupt (ADI) upon A/D conversion end is enabled |

Bit 4: Software A/D Start Flag (SST)

Starts software-triggered A/D conversion and indicates or controls the end of conversion. This bit remains 1 during software-triggered A/D conversion.

When 0 is written in this bit, software-triggered A/D conversion operation can forcibly be aborted.

Bit 4

| SST | Description |
|-----|--|
| 0 | Read: Indicates that software-triggered A/D conversion has ended or been stopped (Initial value) |
| | Write: Software-triggered A/D conversion is aborted |
| 1 | Read: Indicates that software-triggered A/D conversion is in progress |
| | Write: Starts software-triggered A/D conversion |

Bit 3: Hardware A/D Status Flag (HST)

Indicates the status of hardware- or external-triggered A/D conversion. When 0 is written in this bit, A/D conversion is aborted regardless of whether it was hardware-triggered or external-triggered.

Bit 3

| HST | Description |
|-----|--|
| 0 | Read: Hardware- or external-triggered A/D conversion is not in progress(Initial value) |
| | Write: Hardware- or external-triggered A/D conversion is aborted. |
| 1 | Hardware- or external-triggered A/D conversion is in progress. |

Bit 2: Busy Flag (BUSY)

During hardware- or external-triggered A/D conversion, if software attempts to start A/D conversion by writing to the SST bit, the SST bit is not modified and instead the BUSY flag is set to 1.

This flag is cleared when the hardware-triggered A/D result register (AHR) is read.

Bit 2

| BUSY | Description |
|------|--|
| 0 | No contention for A/D conversion (Initial value) |
| 1 | Indicates an attempt to execute software-triggered A/D conversion while hardware- or external-triggered A/D conversion was in progress |

Bit 1: Software-Triggered Conversion Cancel Flag (SCNL)

Indicates that software-triggered A/D conversion was canceled by the start of hardware-triggered A/D conversion.

This flag is cleared when A/D conversion is started by software.

Bit 1

| SCNL | Description |
|------|---|
| 0 | No contention for A/D conversion (Initial value) |
| 1 | Indicates that software-triggered A/D conversion was canceled by the start of hardware-triggered A/D conversion |

Bit 0: Reserved

This bit cannot be modified and always reads 1. Writes are disabled.

26.2.5 Trigger Select Register (ADTSR)

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | TRGS1 | TRGS0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W : | — | — | — | — | — | — | R/W | R/W |

The trigger select register (ADTSR) selects hardware- or external-triggered A/D conversion start factor.

ADTSR is an 8-bit readable/writable register that is initialized to H'FC by a reset, and in module stop mode, standby mode, watch mode, subactive mode and subsleep mode.

Bits 7 to 2: Reserved

These bits are reserved and are always read as 1. Writes are disabled.

Bits 1 and 0: Trigger Select

These bits select hardware- or external-triggered A/D conversion start factor. Set these bits when A/D conversion is not in progress.

| Bit 1 | Bit 0 | Description |
|-------|-------|---|
| TRGS1 | TRGS0 | |
| 0 | 0 | Hardware- or external-triggered A/D conversion is disabled (Initial value) |
| | 1 | Hardware-triggered (ADTRG) A/D conversion is selected |
| 1 | 0 | Hardware-triggered (DFG) A/D conversion is selected |
| | 1 | External-triggered (ADTRG) A/D conversion is selected |

26.2.6 Port Mode Register 0 (PMR0)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR07 | PMR06 | PMR05 | PMR04 | PMR03 | PMR02 | PMR01 | PMR00 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Port mode register 0 (PMR0) controls switching of each pin function of port 0. Switching is specified for each bit.

PMR0 is an 8-bit readable/writable register and is initialized to H'00 by a reset.

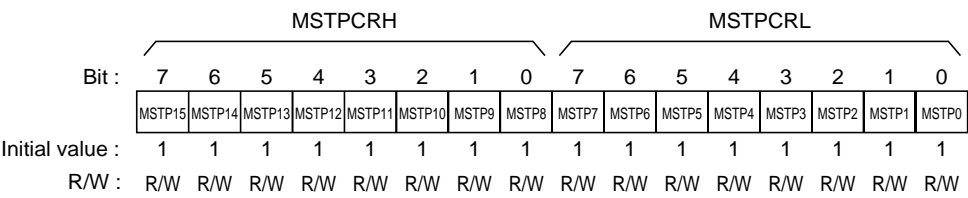
Bit 7 to 0: P07/AN7 to P00/AN0 pin switching (PMR07 to PMR00)

These bits set the P0n/ANn pin as the input pin for P0n or as the ANn pin for A/D conversion analog input channel.

| Bit n | |
|-------|---|
| PMR0n | Description |
| 0 | P0n/ANn functions as a general-purpose input port (Initial value) |
| 1 | P0n/ANn functions as an analog input channel |

(n = 7 to 0)

26.2.7 Module Stop Control Register (MSTPCR)



MSTPCR consists of 8-bit readable/writable registers and performs module stop mode control. When the MSTP2 bit in MSTPCR is set to 1, A/D converter operation stops at the end of the bus cycle and a transition is made to module stop mode. For details, see section 4.5, Module Stop Mode.

MSTPCR is initialized to H'FFFF by a reset

Bit 2: Module Stop (MSTP2)
 Specifies the A/D converter module stop mode.

| MSTPCRL | |
|---------|---|
| Bit 2 | |
| MSTP2 | Description |
| 0 | A/D converter module stop mode is cleared |
| 1 | A/D converter module stop mode is set (Initial value) |

26.3 Interface to Bus Master

ADR and AHR are 16-bit registers, but the data bus to the bus master is only 8 bits wide. Therefore, in accesses by the bus master, the upper byte is accessed directly, but the lower byte is accessed via a temporary register (TEMP).

A data reading from ADR and AHR is performed as follows. When the upper byte is read, the upper byte value is transferred to the CPU and the lower byte value is transferred to TEMP. Next, when the lower byte is read, the TEMP contents are transferred to the CPU. When reading ADR and AHR, always read the upper byte before the lower byte. It is possible to read only the upper byte, but if only the lower byte is read, incorrect data may be obtained. Figure 26.3 shows the data flow for ADR access. The data flow for AHR access is the same.

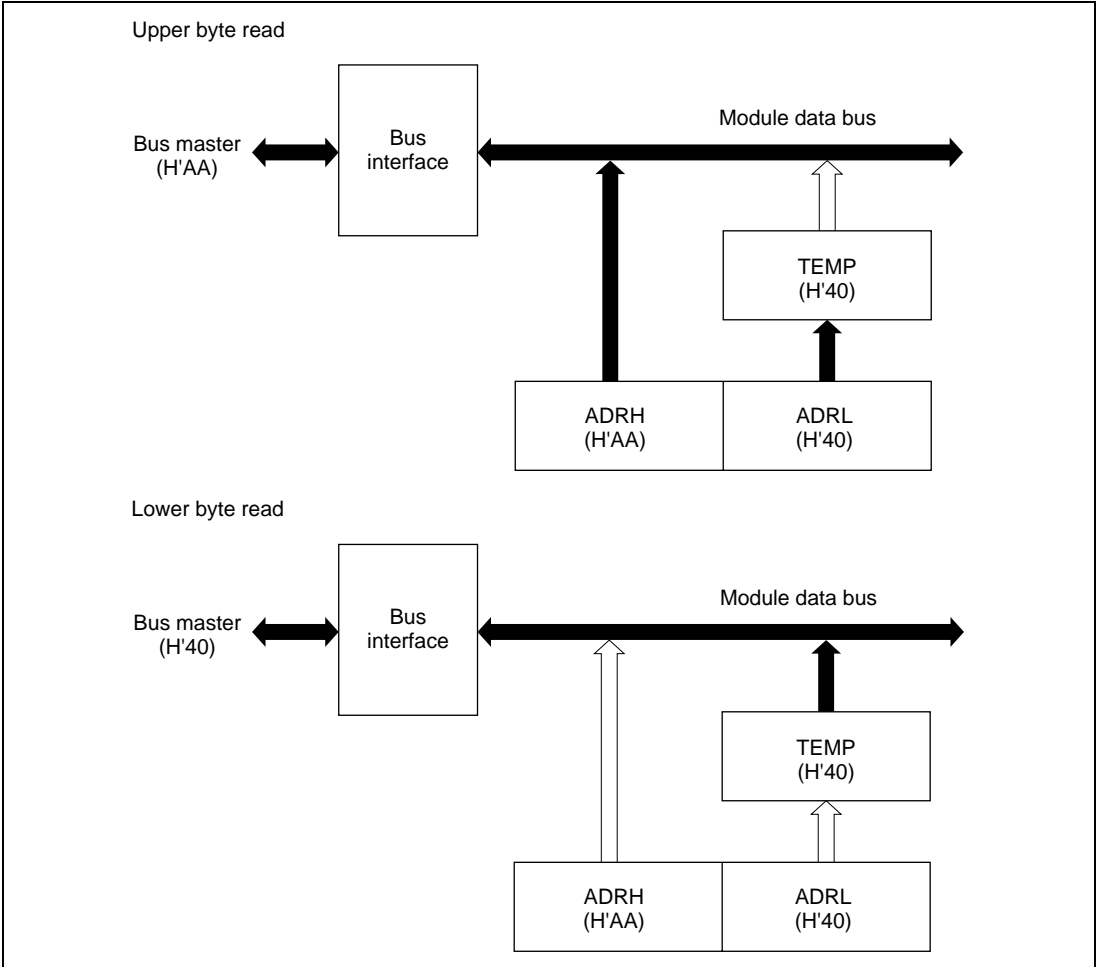


Figure 26.3 ADR Access Operation (Reading H'AA40)

26.4 Operation

The A/D converter operates by successive approximations with 10-bit resolution.

26.4.1 Software-Triggered A/D Conversion

A/D conversion starts when software sets the software A/D start flag (SST bit) to 1. The SST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when conversion ends. Conversion can be software-triggered on any of the 12 channels provided by analog input pins AN0 to ANB. Bits SCH3 to SCH0 in ADCR select the analog input pin used for software-triggered A/D conversion. Pins AN8 to ANB are also available for hardware- or external-triggered conversion.

When conversion ends, SEND flag in ADCSR bit is set to 1. If ADIE bit in ADCSR is also set to 1, an A/D conversion end interrupt occurs.

If the conversion time or input channel selection in ADCR needs to be changed during A/D conversion, to avoid malfunctions, first clear the SST bit to 0 to halt A/D conversion.

If software writes 1 in the SST bit to start software-triggered conversion while hardware- or external-triggered conversion is in progress, the hardware- or external-triggered conversion has priority and the software-triggered conversion is not executed. At this time, BUSY flag in ADCSR is set to 1. The BUSY flag is cleared to 0 when the hardware-triggered A/D result register (AHR) is read. If conversion is triggered by hardware while software-triggered conversion is in progress, the software-triggered conversion is immediately canceled and the SST flag is cleared to 0, and SCNL flag in ADCSR is set to 1. The SCNL flag is cleared when software writes 1 in the SST bit to start conversion after the hardware-triggered conversion ends.

26.4.2 Hardware- or External-Triggered A/D Conversion

The system contains the hardware trigger function that allows to turn on A/D conversion at a specified timing by use of the hardware trigger (internal signals: ADTRG and DFG) and the incoming external trigger ($\overline{\text{ADTRG}}$). This function can be used to measure an analog signal that varies in synchronization with an external signal at a fixed timing.

To execute hardware- or external-triggered A/D conversion, select appropriate start factor in TRGS1 and TRGS0 bits in ADTSR. When the selected triggering occurs, HST flag in ADCSR is set to 1 and A/D conversion starts. The HST flag remains 1 during A/D conversion, and is automatically cleared to 0 when conversion ends. For ADTRG start by HSW timing generator in hardware triggering, see section 28.4, HSW Timing Generator. Setting of the analog input pins on four channels from AN8 to ANB can be modified with the hardware trigger or the incoming external trigger. Setting is done from HCH1 and HCH0 bits on ADCR. Pins AN8 to ANB are also available for software-triggered conversion.

When conversion ends, HEND flag in ADCSR is set to 1. If ADIE bit in ADCSR is also set to 1, an A/D conversion end interrupt occurs.

If the conversion time or input channel selection in ADCR needs to be changed during A/D conversion, to avoid malfunctions, first clear the HST flag to 0 to halt A/D conversion.

If software writes 1 in the SST bit to start software-triggered conversion while hardware- or external-triggered conversion is in progress, the hardware- or external-triggered conversion has priority and the software-triggered conversion is not executed. At this time, BUSY flag in ADCSR is set to 1. The BUSY flag is cleared to 0 when the hardware-triggered A/D result register (AHR) is read.

If conversion is triggered by hardware while software-triggered conversion is in progress, the software-triggered conversion is immediately canceled and the SST flag is cleared to 0, and SCNL flag in ADCSR is set to 1 (the SCNL flag is cleared when software writes 1 in the SST bit to start conversion after the hardware-triggered conversion ends). The analog input channel changes automatically from the channel that was undergoing software-triggered conversion (selected by bits SCH3 to SCH0 in ADCR) to the channel selected by bits HCH1 and HCH0 in ADCR for hardware- or external-triggered conversion. After the hardware- or external-triggered conversion ends, the channel reverts to the channel selected by the software-triggered conversion channel select bits in ADCR.

Hardware- or external-triggered conversion has priority over software-triggered conversion, so the A/D interrupt-handling routine should check the SCNL and BUSY flags when it processes the converted data.

26.5 Interrupt Sources

When A/D conversion ends, SEND or HEND flag in ADCSR is set to 1. The A/D conversion end interrupt can be enabled or disabled by ADIE bit in ADCSR.

Figure 26.4 shows the block diagram of A/D conversion end interrupt.

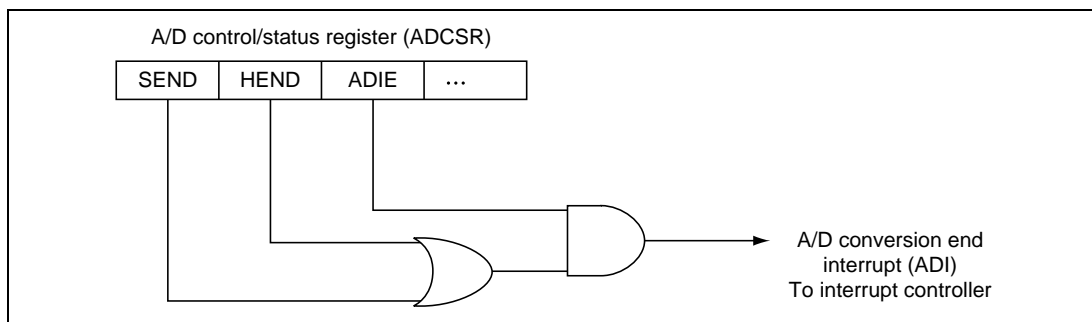


Figure 26.4 Block Diagram of A/D Conversion End Interrupt

Section 27 Address Trap Controller (ATC)

27.1 Overview

The address trap controller (ATC) is capable of generating interrupt by setting an address to trap, when the address set appears during bus cycle.

27.1.1 Features

Address to trap can be set independently at three points.

27.1.2 Block Diagram

Figure 27.1 shows a block diagram of the address trap controller.

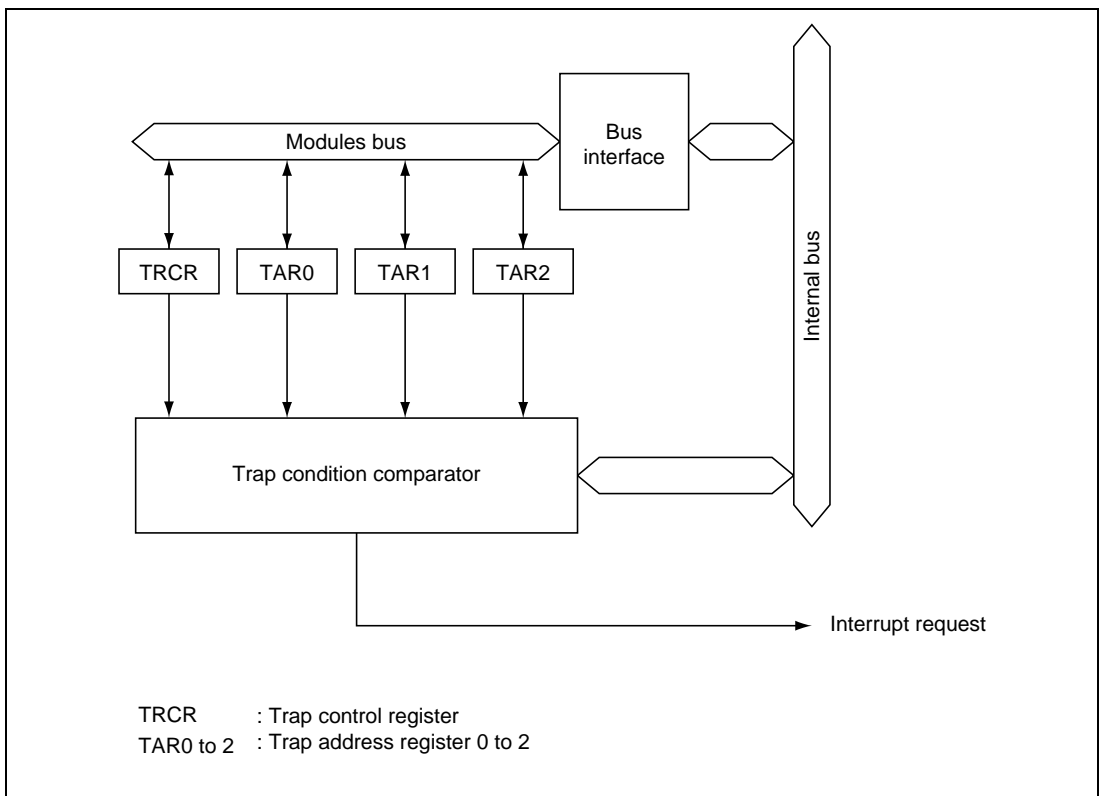


Figure 27.1 Block Diagram of ATC

27.1.3 Register Configuration

Table 27.1 Register List

| Name | Abbrev. | R/W | Initial Value | Address * |
|-------------------------------|---------|-----|---------------|------------------|
| Address trap control register | ATCR | R/W | H'F8 | H'FFB9 |
| Trap address register 0 | TAR0 | R/W | H'F00000 | H'FFB0 to H'FFB2 |
| Trap address register 1 | TAR1 | R/W | H'F00000 | H'FFB3 to H'FFB5 |
| Trap address register 2 | TAR2 | R/W | H'F00000 | H'FFB6 to H'FFB8 |

Note: * Lower 16 bits of the address.

27.2 Register Descriptions

27.2.1 Address Trap Control Register (ATCR)

| | | | | | | | | |
|-----------------|---|---|---|---|---|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | TRC2 | TRC1 | TRC0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| R/W : | — | — | — | — | — | R/W | R/W | R/W |

Bits 7 to 3: Reserved

When read, 1 is read at all times. Writes are disabled.

Bit 2: Trap Control 2 (TRC2)

Sets ON/OFF operation of the address trap function 2.

Bit 2

| TRC2 | Description |
|------|--|
| 0 | Address trap function 2 disabled (Initial value) |
| 1 | Address trap function 2 enabled |

Bit 1: Trap Control 1 (TRC1)

Sets ON/OFF operation of the address trap function 1.

Bit 1

| TRC1 | Description |
|------|--|
| 0 | Address trap function 1 disabled (Initial value) |
| 1 | Address trap function 1 enabled |

Bit 0: Trap Control 0 (TRC0)

Sets ON/OFF operation of the address trap function 0.

Bit 0

| TRC0 | Description |
|------|--|
| 0 | Address trap function 0 disabled (Initial value) |
| 1 | Address trap function 0 enabled |

27.2.2 Trap Address Register 2 to 0 (TAR2 to TAR0)

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | A23 | A22 | A21 | A20 | A19 | A18 | A17 | A16 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | A7 | A6 | A5 | A4 | A3 | A2 | A1 | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | — |

The TAR is composed of three 8-bit readable/writable registers (TARnA, B, and C)(n = 2 to 0). The TAR sets the address to trap. The function of the TAR2 to TAR0 is the same. The TAR is initialized to H'00 by a reset.

TARA bits 7 to 0: Addresses 23 to 16 (A23 to A16)

TARB bits 7 to 0: Addresses 15 to 8 (A15 to A8)

TARC bits 7 to 0: Addresses 7 to 1 (A7 to A1)

If the value installed in this register and internal address buses A23 to A1 match as a result of comparison, an interruption occurs.

For the address to trap, set to the address where the first byte of an instruction exists. In the case of other addresses, it may not be considered that the condition has been satisfied.

Bit 0 of this register is fixed at 0. The address to trap becomes an even address.

The range where comparison is made is H'000000 to H'FFFFFFE.

27.3 Precautions in Usage

Address trap interrupt arises 2 states after prefetching the trap address. Trap interrupt may occur after the trap instruction has been executed, depending on a combination of instructions immediately preceding the setting up of the address trap.

If the instruction to trap immediately follows the branch instruction or the conditional branch instruction, operation may differ, depending on whether the condition was satisfied or not, or the address to be stacked may be located at the branch. Figures 27.2 to 27.22 show specific operations.

For information as to where the next instruction prefetch occurs during the execution cycle of the instruction, see appendix A.5 of this manual or section 2.7 Bus State during Execution of Instruction of the H8S/2600, H8S/2000 Series Programming Manual. (R:W NEXT is the next instruction prefetch.)

27.3.1 Basic Operations

After terminating the execution of the instruction being executed in the second state from the trap address prefetch, the address trap interrupt exception handling is started.

- (1) Figure 27.2 shows the operation when the instruction immediately preceding the trap address is that of 3 states or more of the execution cycle and the next instruction prefetch occurs in the state before the last 2 states. The address to be stacked is 0260.

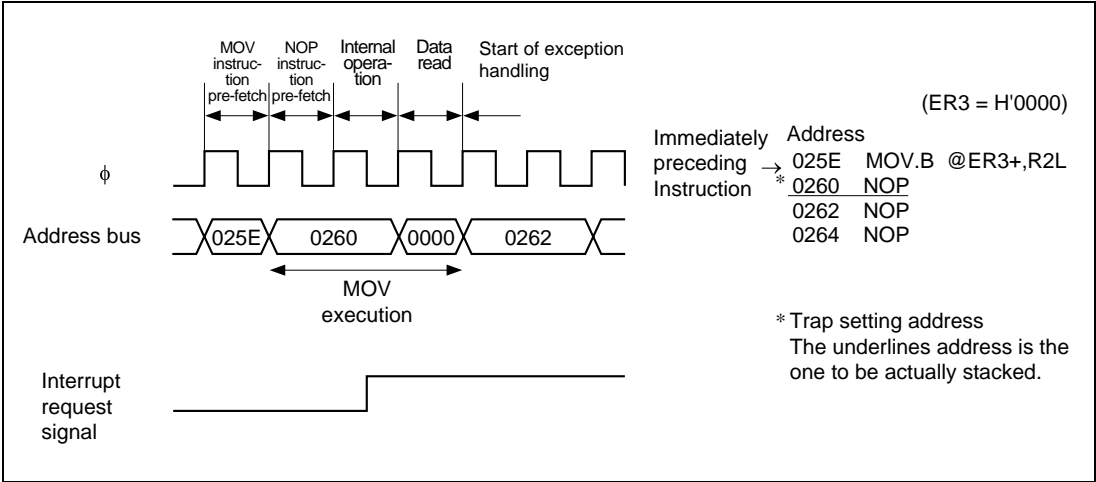


Figure 27.2 Basic Operations (1)

Note: In the figure above, the NOP instruction is used as the typical example of instruction with execution cycle of 1 state. Other instructions with the execution cycle of 1 state also apply (Ex. MOV.B, Rs, Rd).

- (2) Figure 27.3 shows the operation when the instruction immediately preceding the trap address is that of 2 states or more of the execution cycle and the next instruction prefetch occurs in the second state from the last. The address to be stacked is 0268.

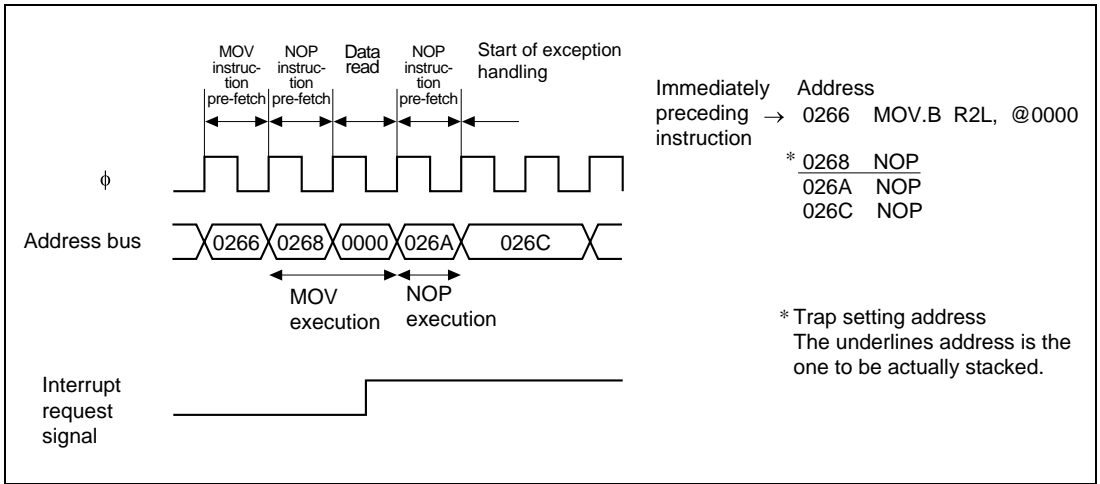


Figure 27.3 Basic Operations (2)

- (3) Figure 27.4 shows the operation when the instruction immediately preceding the trap address is that of 1 state or 2 states or more and the prefetch occurs in the last state. The address to be stacked is 025C.

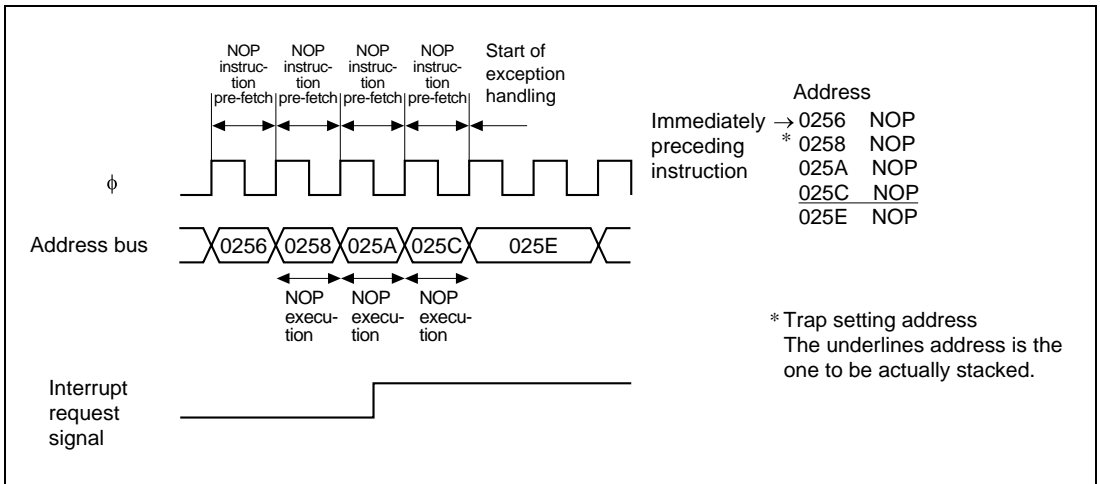
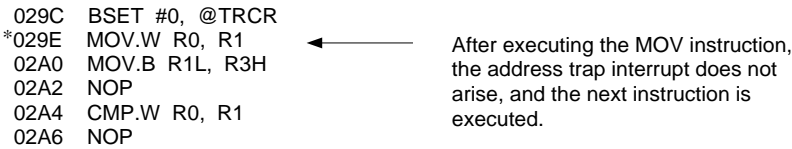


Figure 27.4 Basic Operations (3)

27.3.2 Enable

The address trap function becomes valid after executing one instruction following the setting of the enable bit of the address trap control register (TRCR) to 1.



* Trap setting address

Figure 27.5 Enable

27.3.3 Bcc Instruction

(1) When the condition is satisfied by Bcc instruction (8-bit displacement)

If the trap address is the next instruction to the Bcc instruction and the condition is satisfied by the Bcc instruction and then branched, transition is made to the address trap interrupt after executing the instruction at the branch. The address to be stacked is 02A8.

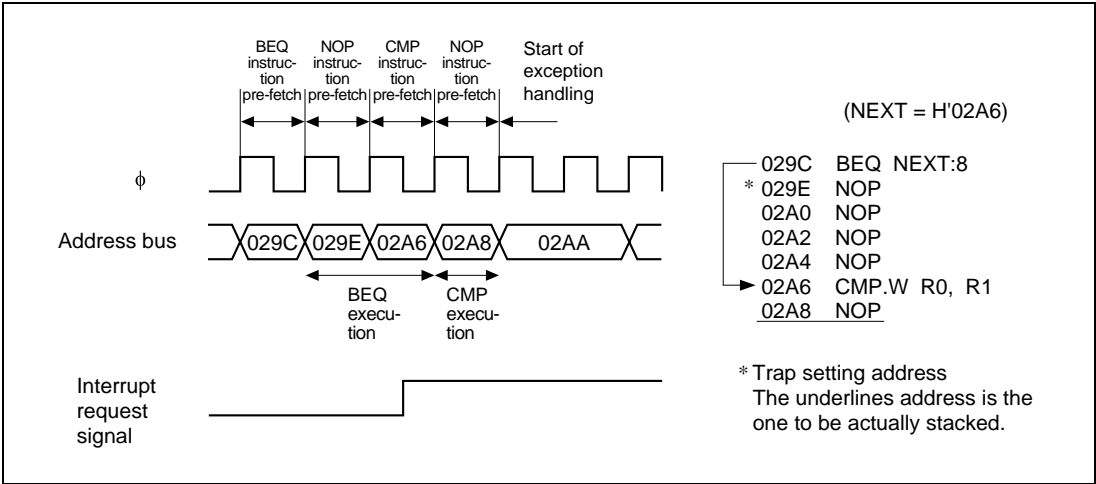


Figure 27.6 When the Condition is Satisfied by Bcc Instruction (8-bit Displacement)

(2) When the condition is not satisfied by Bcc instruction (8-bit displacement)

If the trap address is the next instruction to the Bcc instruction and the condition is not satisfied by the Bcc instruction and thus it fails to branch, transition is made to the address trap interrupt after executing the trap address instruction and prefetching the next instruction. The address to be stacked is 02A2.

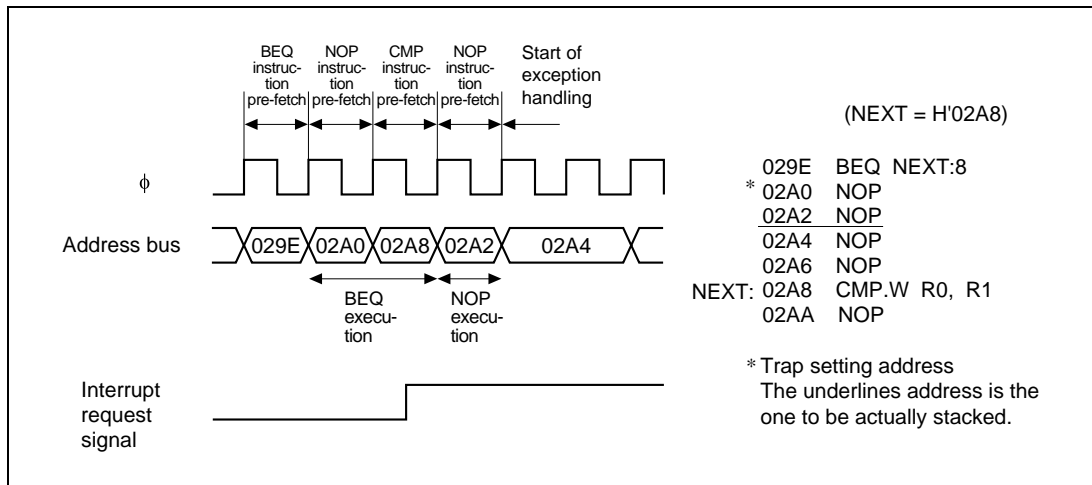


Figure 27.7 When the Condition is Not Satisfied by Bcc Instruction (8-bit Displacement)

(3) When condition is not satisfied by Bcc instruction (16-bit displacement)

If the trap address is the next instruction to the Bcc instruction and the condition is not satisfied by the Bcc instruction and thus it fails to branch, transition is made to the address trap interrupt after executing the trap address instruction (if the trap address instruction is that of 2 states or more. If the instruction is that of 1 state, after executing two instructions). The address to be stacked is 02C0.

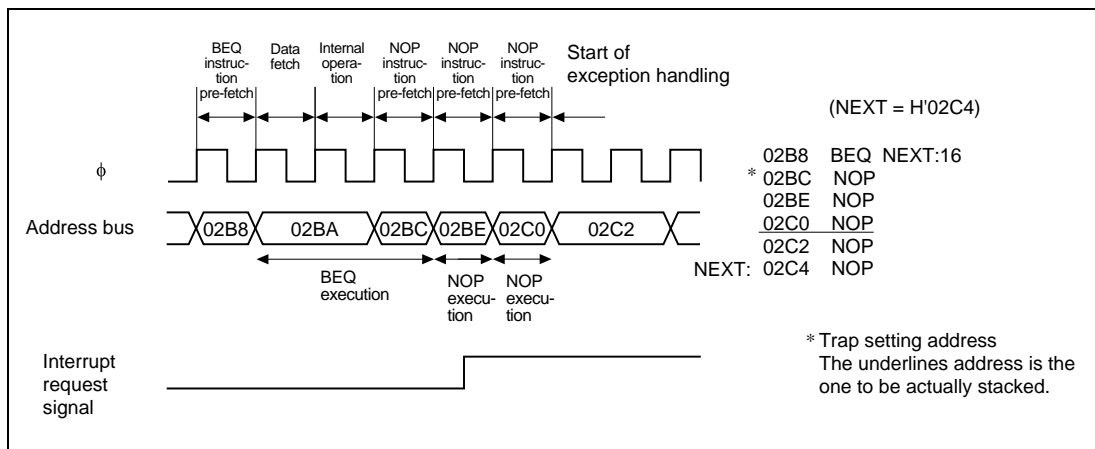
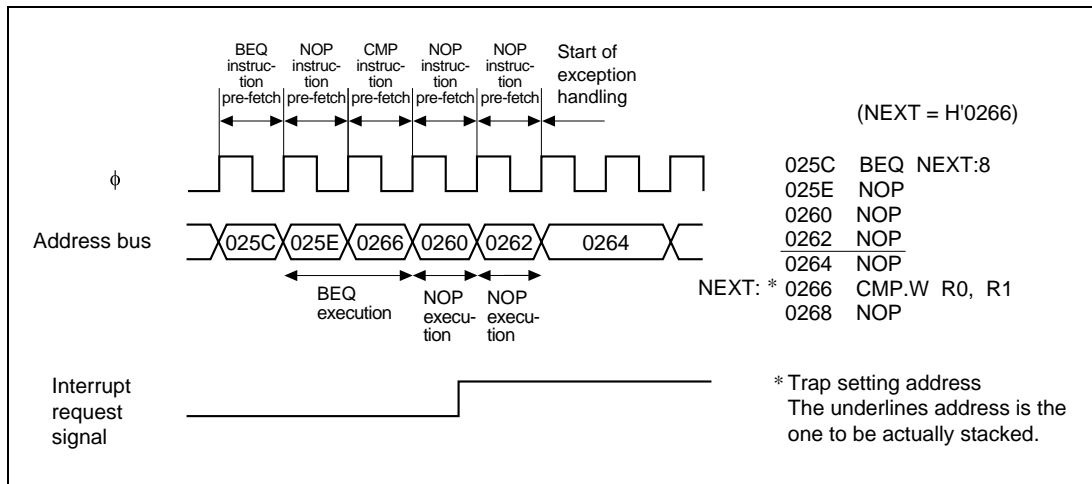


Figure 27.8 When the Condition is Not Satisfied by Bcc Instruction (16-bit Displacement)

(4) When the condition is not satisfied by Bcc instruction (Trap address at branch)

When the trap address is at the branch of the Bcc instruction and the condition is not satisfied by the Bcc instruction and thus it fails to branch, transition is made into the address trap interrupt after executing the next instruction (if the next instruction is that of 2 states or more. If the next instruction is that of 1 state, after executing two instructions). The address to be stacked is 0262.



**Figure 27.9 When the Condition is Not Satisfied by Bcc Instruction
(Trap Address at Branch)**

27.3.4 BSR Instruction

(1) BSR Instruction (8-bit displacement)

When the trap address is the next instruction to the BSR instruction and the addressing mode is an 8-bit displacement, transition is made to the address trap interrupt after prefetching the instruction at the branch. The address to be stacked is 02C2.

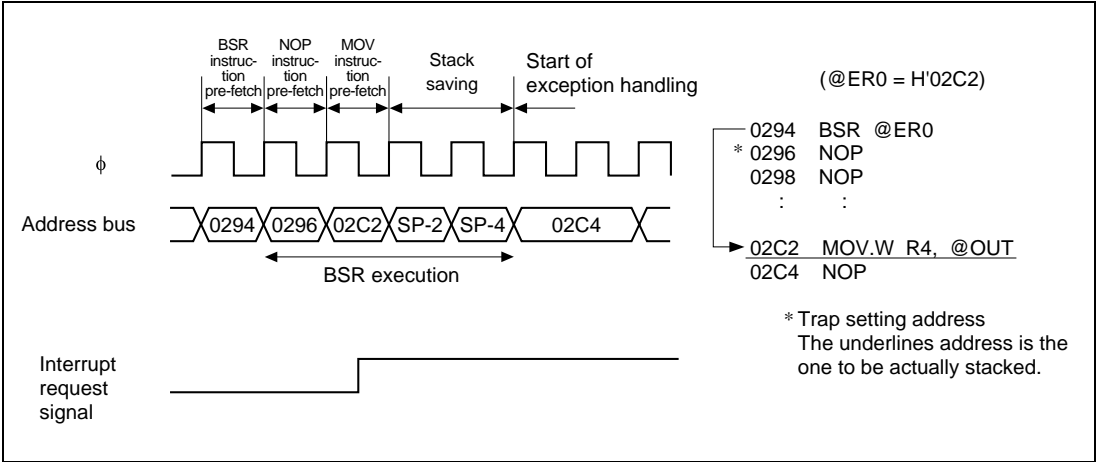


Figure 27.10 BSR Instruction (8-bit Displacement)

27.3.5 JSR Instruction

(1) JSR Instruction (Register indirect)

When the trap address is the next instruction to the JSR instruction and the addressing mode is a register indirect, transition is made to the address trap interrupt after prefetching the instruction at the branch. The address to be stacked is 02C8.

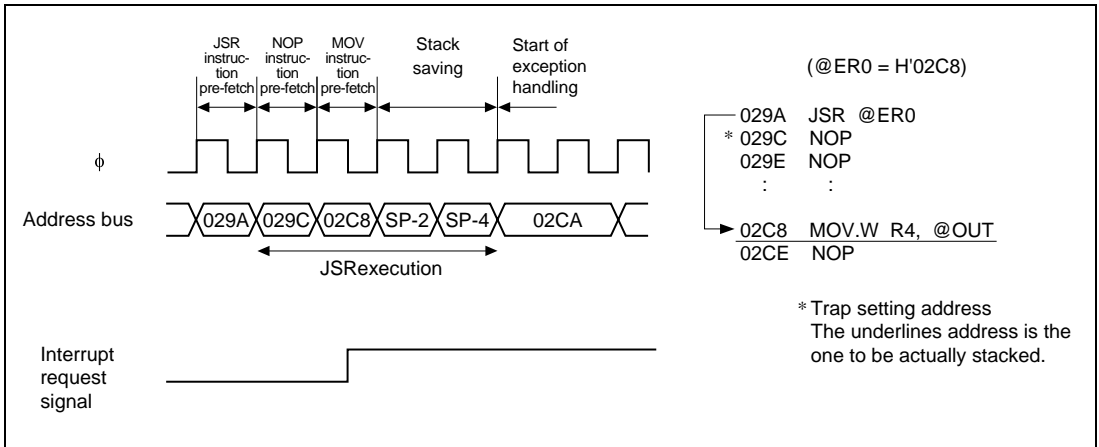


Figure 27.11 JSR Instruction (Register indirect)

(2) JSR Instruction (Memory indirect)

When the trap address is the next instruction to the JSR instruction and the addressing mode is memory indirect, transition is made to the address trap interrupt after prefetching the instruction at the branch. The address to be stacked is 02EA.

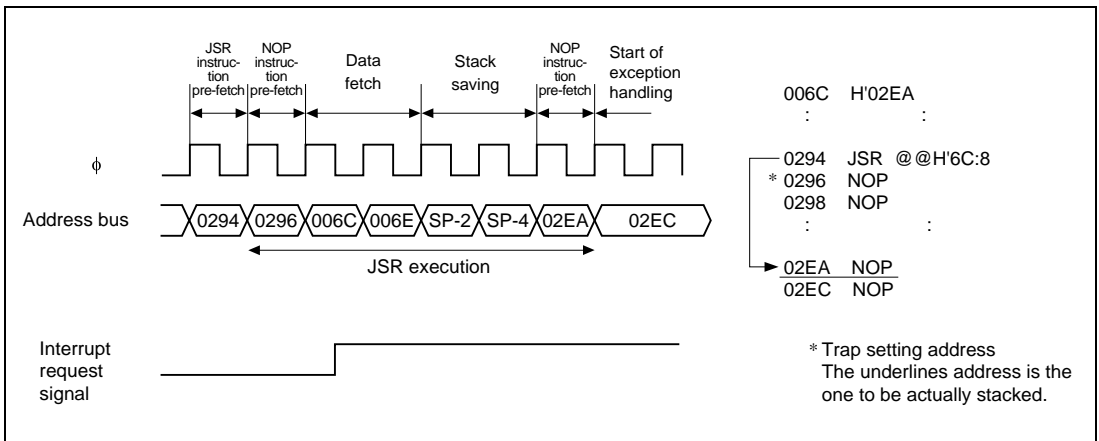


Figure 27.12 JSR Instruction (Memory Indirect)

27.3.6 JMP Instruction

(1) JMP Instruction (Register indirect)

When the trap address is the next instruction to the JMP instruction and the addressing mode is a register indirect, transition is made to the address trap interrupt after prefetching the instruction at the branch. The address to be stacked is 02AA.

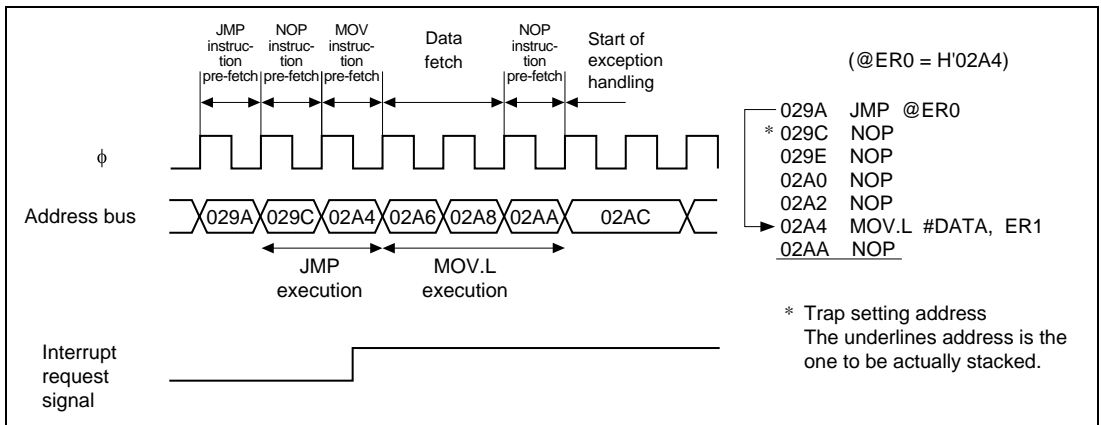


Figure 27.13 JMP Instruction (Register Indirect)

(2) JMP Instruction (Memory indirect)

When the trap address is the next instruction to the JMP instruction and the addressing mode is memory indirect, transition is made to the address trap interrupt after prefetching the instruction at the branch. The address to be stacked is 02E4.

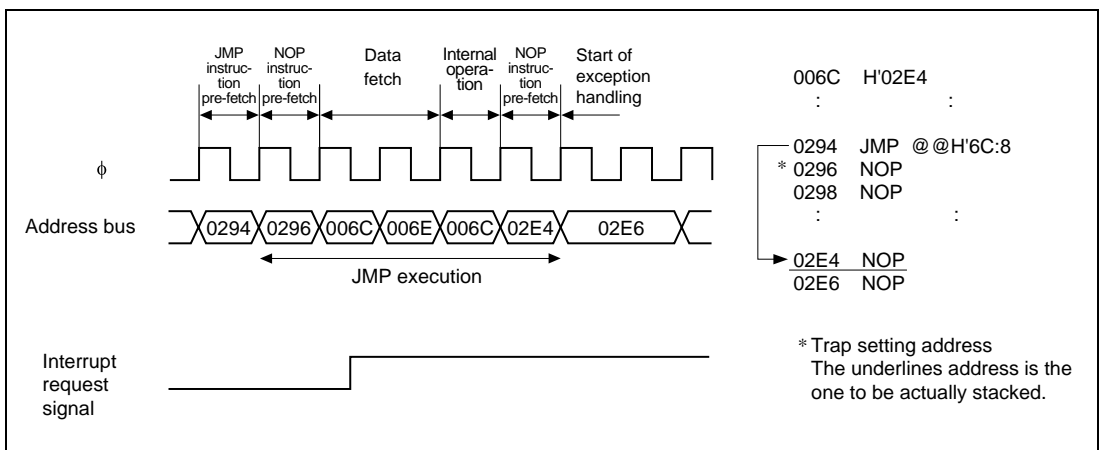


Figure 27.14 JMP Instruction (Memory Indirect)

27.3.7 **RTS Instruction**

When the trap address is the next instruction to the RTS instruction, transition is made to the address trap interrupt after reading the CCR and PC from the stack and prefetching the instruction at the return location. The address to be stacked is 0298.

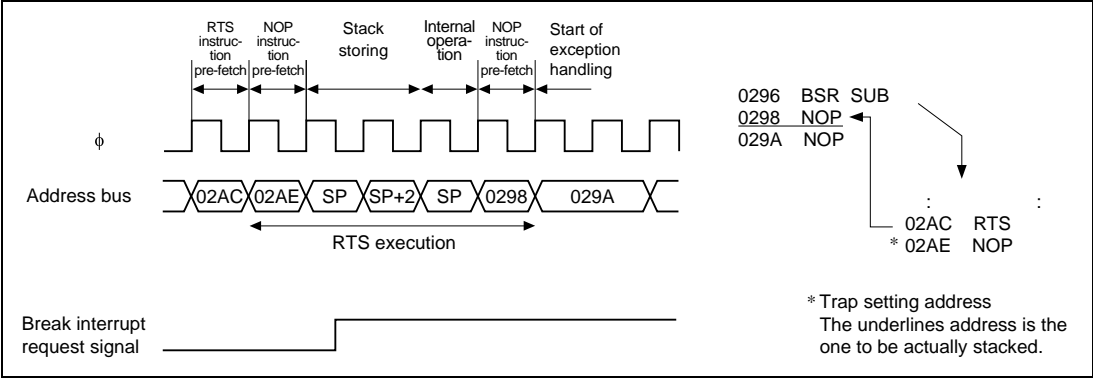


Figure 27.15 **RTS Instruction**

27.3.8 **SLEEP Instruction**

(1) SLEEP Instruction 1

When the trap address is the SLEEP instruction and the instruction execution cycle immediately preceding the SLEEP instruction is that of 2 states or more and prefetch does not occur in the last state, the SLEEP instruction is not executed and transition is made to the address trap interrupt without going into SLEEP mode. The address to be stacked is 0274.

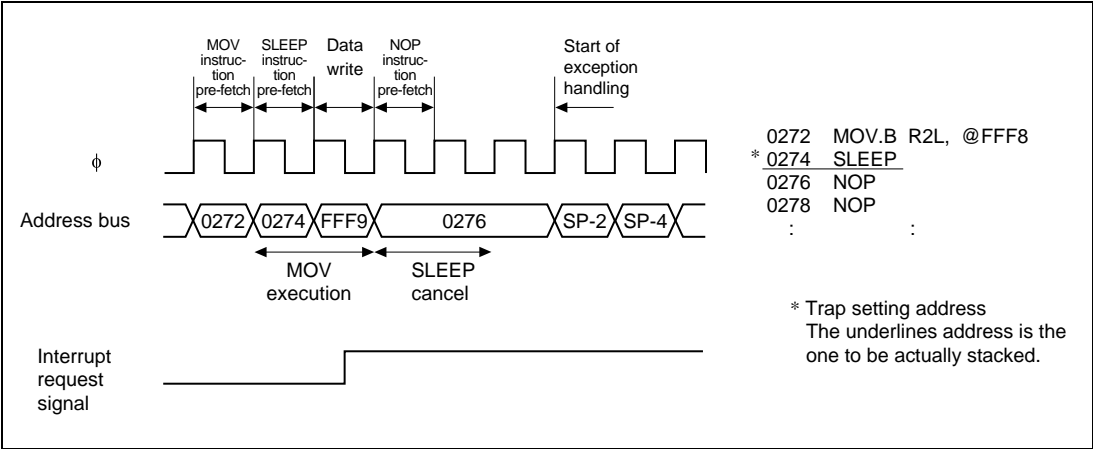


Figure 27.16 **SLEEP Instruction (1)**

(2) SLEEP Instruction 2

When the trap address is the SLEEP instruction and the instruction execution cycle immediately preceding the SLEEP instruction is that of 1 state 2 states or more and prefetch occurs in the last state, this puts in the SLEEP mode after execution of the SLEEP instruction, and the SLEEP mode is cancelled by the address trap interrupt and transition is made to the exception handling. The address to be stacked is 0264.

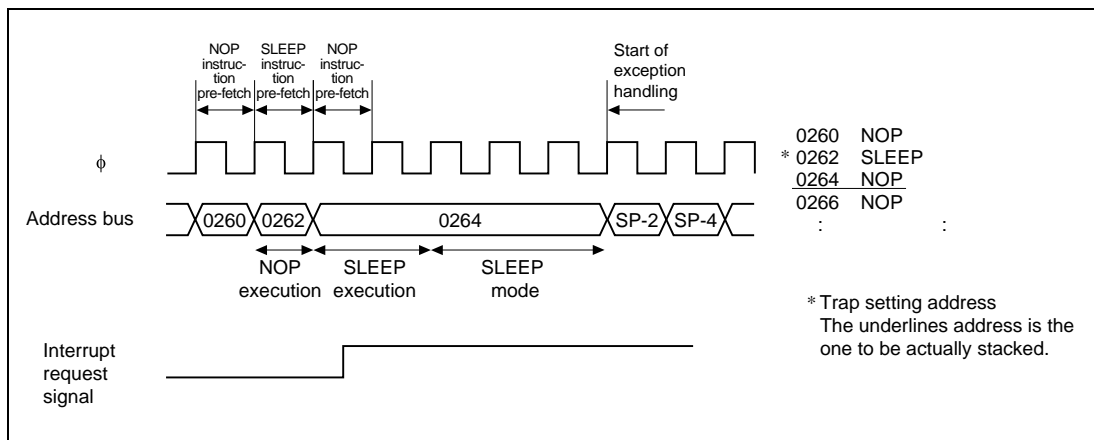


Figure 27.17 SLEEP Instruction (2)

(3) SLEEP Instruction 3

When the trap address is the next instruction to the SLEEP instruction, this puts in the SLEEP mode after execution of the SLEEP instruction, and the SLEEP mode is cancelled by the address trap interrupt and transition is made to the exception handling. The address to be stacked is 0282.

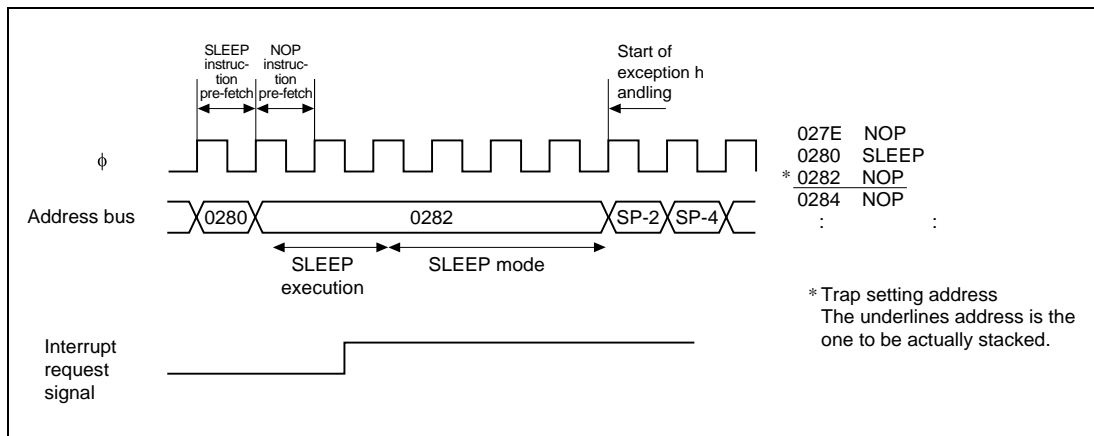


Figure 27.18 SLEEP Instruction (3)

(4) SLEEP Instruction 4 (Standby or Watch Mode Setting)

When the trap address is the SLEEP instruction and the instruction immediately preceding the SLEEP instruction is that of 1 state or 2 states or more and prefetch occurs in the last state, this puts in the standby (watch) mode after execution of the SLEEP instruction. After that, if the standby (watch) mode is cancelled by the NMI interrupt, transition is made to NMI interrupt following the CCR and PC (at the address of 0266) stack saving and vector reading. However, if the address trap interrupt arises before starting execution of the NMI interrupt processing, transition is made to the address trap exception handling. The address to be stacked is the starting address of the NMI interrupt processing.

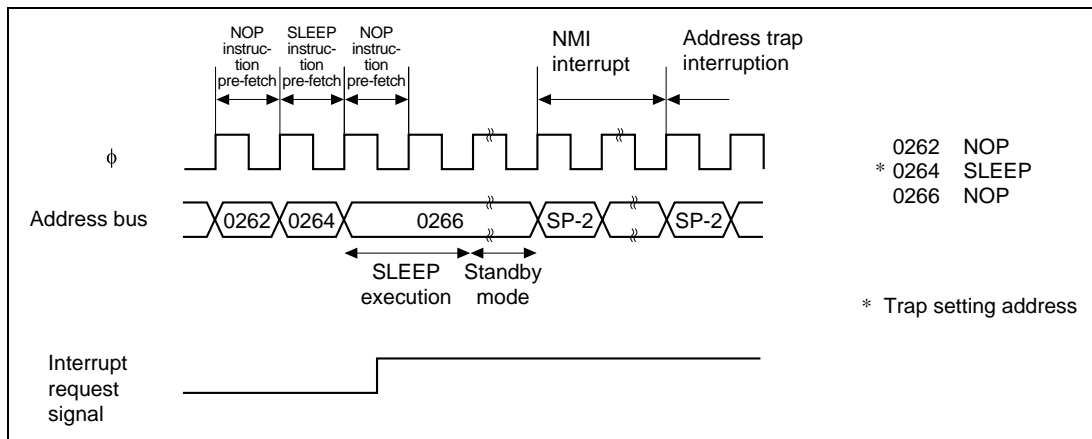


Figure 27.19 SLEEP Instruction (4) (Standby or Watch Mode Setting)

(5) SLEEP Instruction 5 (Standby or Watch Mode Setting)

When the trap address is the next instruction to the SLEEP instruction, this puts in the standby (watch) mode after execution of the SLEEP instruction. After that, if the standby (watch) mode is cancelled by the NMI interruption, transition is made to the NMI interrupt following the CCR and PC (at the address of 0266) stack saving and vector reading.

However, if the address trap interrupt arises before starting execution of the NMI interrupt processing, transition is made to the address trap exception handling. The address to be stacked is the starting address of the NMI interrupt processing.

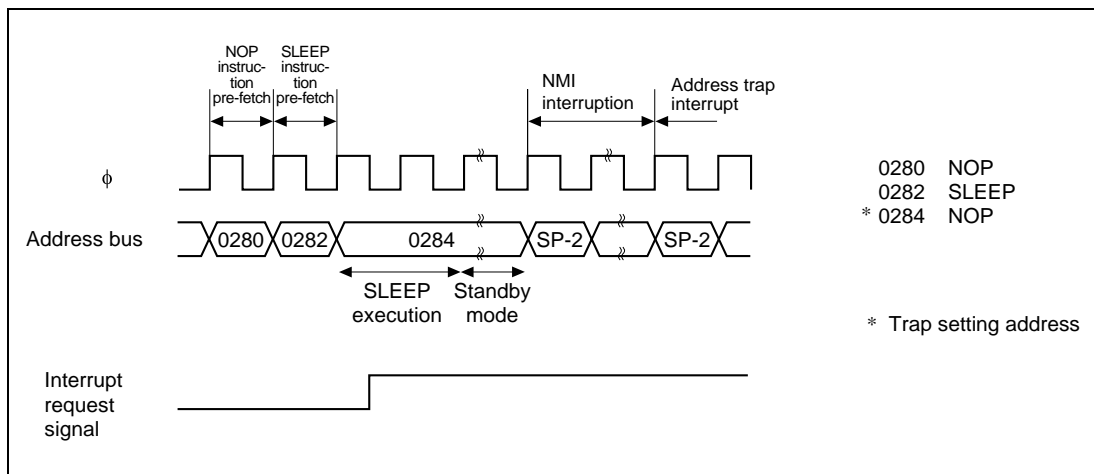


Figure 27.20 SLEEP Instruction (5) (Standby or Watch Mode Setting)

27.3.9 Competing Interrupt

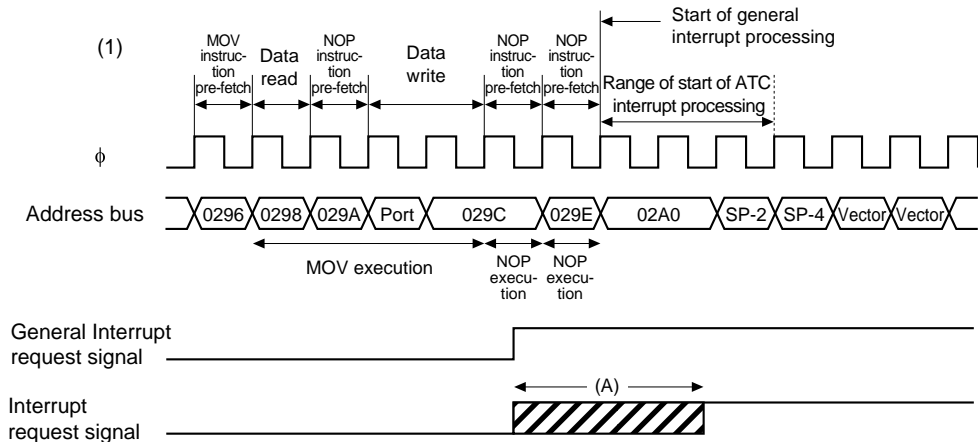
(1) General Interrupt (Interrupt other than NMI)

When the ATC interrupt request is made at the timing in (1) (A) against the general interrupt request, the interruption appears to take place in the ATC at the timing earlier than usual, because higher priority is assigned to the ATC interrupt processing (Simultaneous interrupt with the general interrupt has no effect on processing). The address to be stacked is 029E. For comparison, the case where the trap address is set at 02A0 if no general interrupt request was made is shown in (2). The address to be stacked is 02A4.


```

0296 MOV.B R2L, @Port
029A NOP
029C NOP } Set one of these to the
029E NOP } trap address
02A0 NOP
02A2 NOP
02A4 NOP

```



```

0296 MOV.B R2L, @Port
029A NOP
029C NOP
029E NOP
02A0 NOP Trap address
02A2 NOP
02A4 NOP

```

Address to be stacked → 02A4 NOP

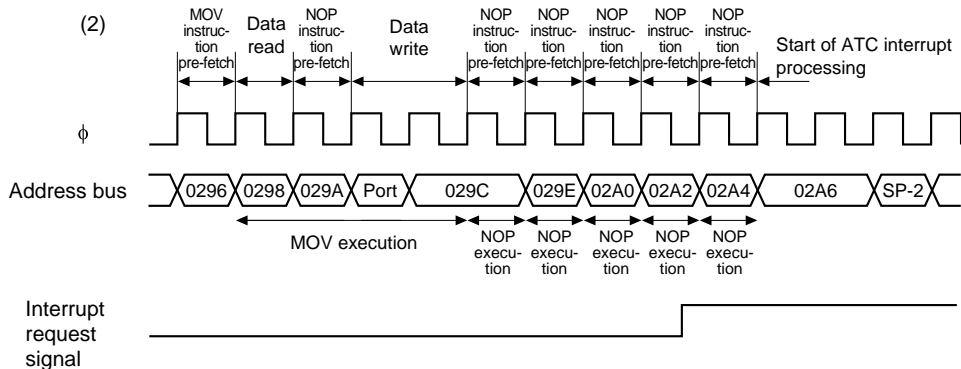


Figure 27.21 Competing Interrupt (General Interrupt)

(2) In case of NMI

When the NMI interruption request is made at the timing in (1) (A) against the ATC interrupt request, the interrupt appears to take place in NMI at the timing earlier than usual, because higher priority is assigned to the NMI interrupt processing. The ATC interrupt processing starts after fetching the instruction at the starting address of the NMI interrupt processing. The address to be stacked is 02E0 for the NMI and 340 for the ATC.

When the ATC interrupt request is made at the timing in (2) (B) against the NMI interrupt request, the ATC interrupt processing starts after fetching the instruction at the starting address of the NMI interrupt processing. The address to be stacked is 02E6 for the NMI and 0340 for the ATC.

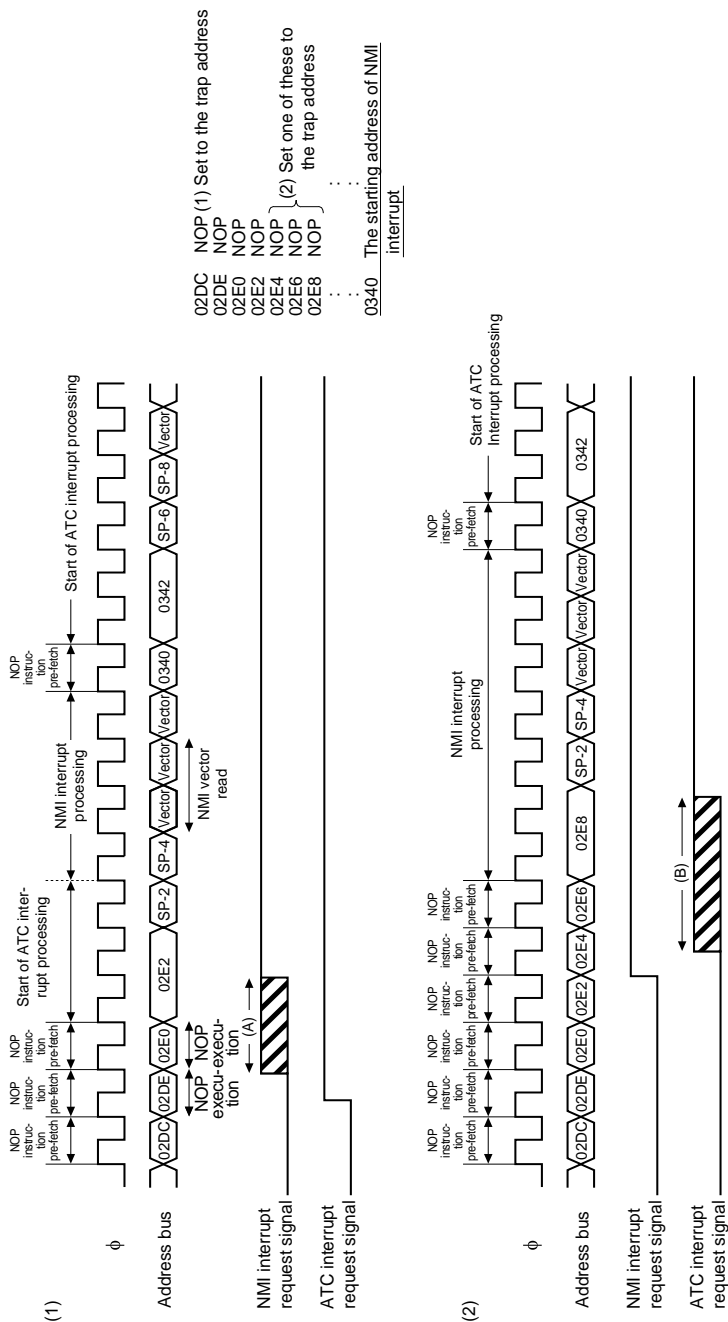


Figure 27.22 Competing Interrupt (In Case of NMI)

Section 28 Servo Circuits

28.1 Overview

28.1.1 Functions

Servo circuits for a video cassette recorder are included on-chip.

The functions of the servo circuits can be divided into four groups, as listed in table 28.1.

Table 28.1 Servo Circuit Functions

| Group | Function | Description |
|---------------------------------|---|---|
| (1) Input and output circuits | CTL I/O amplifier | Gain variable input amplifier Output amplifier with rewrite mode |
| | CFGDuty compensation input | Duty accuracy: 50±2% (Zero cross type comparator) |
| | DFG, DPG separation/overlap input | Overlap input available: Three-level input method, DFG noise mask function |
| | Reference signal generators | V compensation, field detection, external signal sync, V sync when in REC mode, REF30 signal output to outside |
| | HSW timing generator | Head-switching signals, FIFO 20 stages Compatible with DFG counter soft-reset |
| | Four-head high-speed switching circuit for special playback | Chroma-rotary/head-amplifier switching output |
| | 12-bit PWM | Improved speed of carrier frequency |
| | Frequency division circuit | With CFG mask, no CFG for phase or CTL mask |
| | Sync detection circuit | Noise count, field discrimination, Hsync compensation, Hsync detection noise mask |
| (2) Error detectors | Drum speed error detector | Lock detector function, pause at the counter overflow, R/W error latch register, limiter function |
| | Drum phase error detector | Latch signal selectable, R/W error latch register |
| | Capstan speed error detector | Lock detector function, pause at the counter overflow, R/W error latch register, limiter function |
| | Capstan phase error detector | R/W error latch register |
| | X-value adjustment and tracking adjustment circuit | (Separate setting available) |
| (3) Phase and gain compensation | Digital filter computation circuit | Computations performed automatically by hardware Output gain variable: $\times 2$ to $\times 64$ (exponents of 2) (Partial write in Z^{-1} (high-order 8 bits) available) |
| (4) Other circuits | Additional V signal circuit | Valid when in special playback |
| | CTL circuit | Duty discrimination circuit, CTL head R/W control, compatible with wide aspect |

28.1.2 Block Diagram

Figure 28.1 shows a block diagram of the servo circuits.

28.2 Servo Port

28.2.1 Overview

This LSI is equipped with seventeen pins dedicated to servo module and twenty-five dual-purpose pins used also for general-purpose port. It has also built-in input amplifier to amplify CTL signals, CTL output amplifier, CTL Schmitt comparator, and CFG zero cross type comparator. The CTL input amplifier allows gain adjustment by software. DFG and DPG signals, which are the signals to control the drum, allow selection between separate or overlap input.

SV1 and SV2 pins allow to output to monitor the inside signals of the servo section. The signals to be output can be selected out of eight kinds of signals. See section 28.2.5 (4), Servo Monitor Control Register (SVMCR).

28.2.2 Block Diagram

(1) DFG and DPG input circuits

The DFG and DPG input pins have on-chip Schmitt circuits. Figure 28.2 shows the input circuits of DFG and DPG.

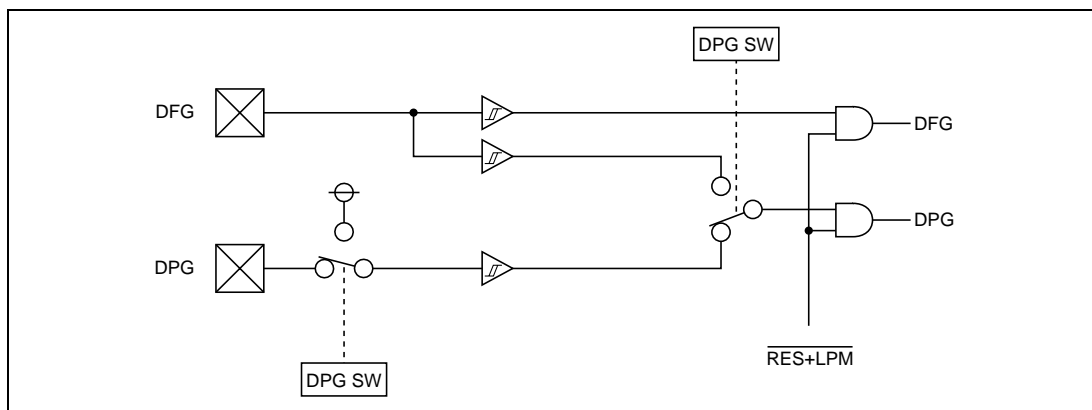


Figure 28.2 Input Circuits of DFG and DPG

(2) CFG Input Circuit

The CFG input pin has built-in an amplifier and a zero cross type comparator. Figure 28.3 shows the input circuit of CFG.

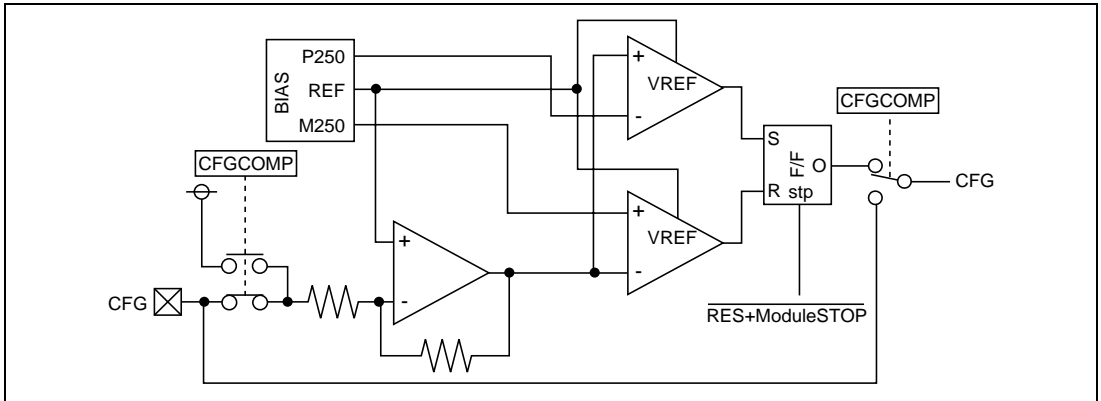


Figure 28.3 CFG Input Circuit

(3) CTL Input Circuit

The CTL input pin has built-in an amplifier. Figure 28.4 shows the input circuit of CTL.

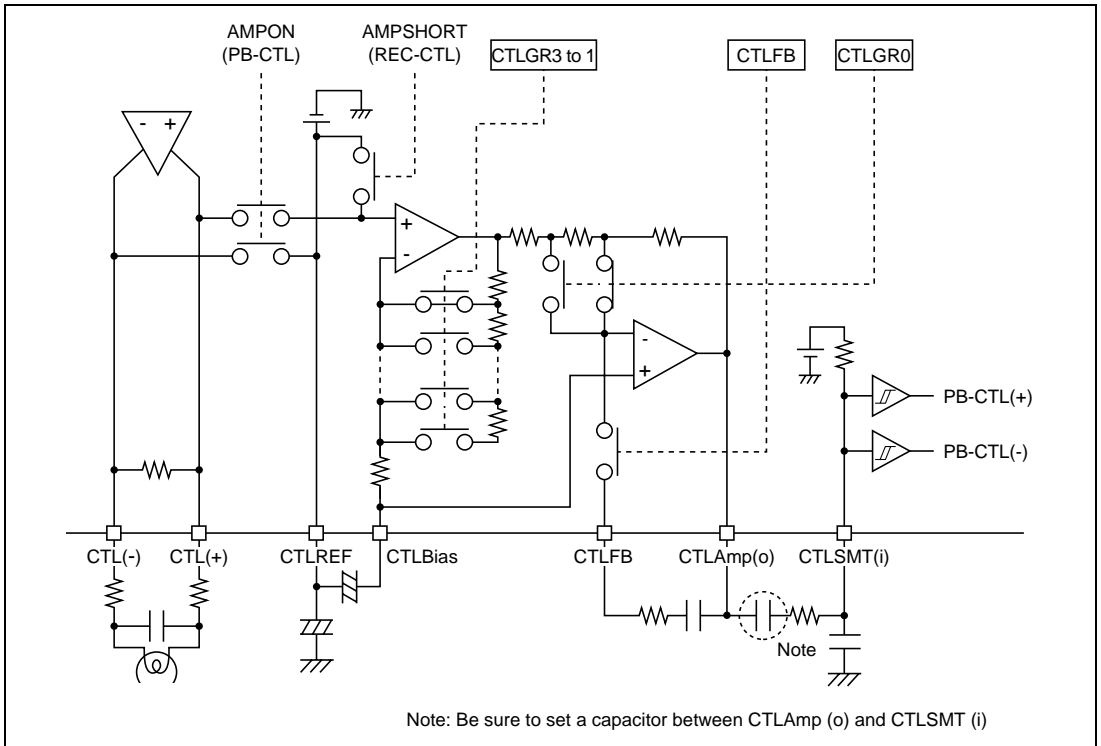


Figure 28.4 CTL Input Circuit

28.2.3 Pin Configuration

Table 28.2 shows the pin configuration of the servo section. P6n, P7n, P80 to P38, and PS1 to PS4 are general-purpose ports. As for P6, P7, and P8, see section 10, I/O Port.

Table 28.2 Pin Configuration

| Name | Abbrev. | I/O | Function |
|-------------------------------------|------------------|-------------|--|
| Servo V _{cc} pin | SV _{cc} | Input | Power source pin for servo section |
| Servo V _{ss} pin | SV _{ss} | Input | Power source pin for servo section |
| Audio head switching pin | Audio FF | Output | Audio head switching signal output |
| Video head switching pin | Video FF | Output | Video head switching signal output |
| Capstan mix pin | CAPPWM | Output | 12-bit PWM square wave output |
| Drum mix pin | DRMPWM | Output | 12-bit PWM square wave output |
| Additional V pulse pin | Vpulse | Output | Additional V signal output |
| Color rotary signal output pin | C.Rotary/PS0 | Output, I/O | Control signal output port for processing color signals/general-purpose port |
| Head amplifier switching pin | H.Amp. SW/PS1 | Output, I/O | Pre-amplifier output selection signal output/general-purpose port |
| Compare signal input pin | COMP/PS2 | Input, I/O | Pre-amplifier output result signal input/general-purpose port |
| CTL (+) I/O pin | CTL (+) | I/O | CTL signal input/output |
| CTL (-) I/O pin | CTL (-) | I/O | CTL signal input/output |
| CTL Bias input pin | CTLBias | Input | CTL primary amplifier bias supply |
| CTL Amp (O) output pin | CTLAMP (O) | Output | CTL amplifier output |
| CTL SMT (i) input pin | CTLSMT (I) | Input | CTL Schmitt amplifier input |
| CTL FB input pin | CTLFB | Input | CTL amplifier high-range characteristics control |
| CTL REF output pin | CTLREF | Output | CTL amplifier reference voltage output |
| Capstan FG amplifier input pin | CFG | Input | CFG signal amplifier input |
| Drum FG input pin | DFG | Input | DFG signal input |
| Drum PG input pin | DPG/PS3 | Input, I/O | DPG signal input/general-purpose port |
| External CTL signal input pin | EXCTL/PS4 | Input, I/O | External CTL signal input/general-purpose port |
| Complex sync signal input pin | Csync | Input | Complex sync signal input |
| External reference signal input pin | P80/EXTTRG | I/O, input | General-purpose port/external reference signal input |
| External capstan signal input pin | P81/EXCAP | I/O, input | General-purpose port/external capstan signal input |
| Servo monitor signal output pin 1 | P82/SV1 | I/O, output | General-purpose port/servo monitor signal output |
| Servo monitor signal output pin 2 | P83/SV2 | I/O, output | General-purpose port/servo monitor signal output |
| PPG output pin | P7n/PPGn | I/O, output | General-purpose port/PPG output |
| RTP output pin | P6n/RPn | I/O, output | General-purpose port/RTP output |

28.2.4 Register Configuration

Table 28.3 shows the register configuration of the servo port section.

Table 28.3 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|--------------------------------|---------|-----|------|---------------|---------|
| Servo port mode register | SPMR | R/W | Byte | H'40 | H'FD0A0 |
| Servo control register | SPCR | W | Byte | H'E0 | H'FD0A1 |
| Servo data register | SPDR | R/W | Byte | H'E0 | H'FD0A2 |
| Servo monitor control register | SVMCR | R/W | Byte | H'C0 | H'FD0A3 |
| CTL gain control register | CTLGR | R/W | Byte | H'C0 | H'FD0A4 |

28.2.5 Register Descriptions

(1) Servo Port Mode Register (SPMR)

| | | | | | | | | |
|-----------------|---------|---|---------|---------|-------|------|----------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CTLSTOP | — | CFGCOMP | EXCTLON | DPGSW | COMP | H.Amp.SW | C.Rot |
| Initial value : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | — | R/W | R/W | R/W | R/W | R/W | R/W |

A register to switch the servo port/general-purpose port, and the CFG input system. SPMR is an 8-bit read/write register. Bit 6 is reserved; writing in it is invalid. If read is attempted, an undetermined value is read out. It is initialized to H'40 by a reset or stand-by.

Bit 7: CTLSTOP Bit (CTLSTOP)

Controls whether the CTL circuits are operated or stopped.

Bit 7

| CTLSTOP | Description |
|---------|--------------------------------------|
| 0 | CTL circuits operate (Initial value) |
| 1 | CTL circuits stop operation |

Bit 6: Reserved

This bit is reserved. It cannot be written or read. If read is attempted, an undetermined value is read out.

Bit 5: CFG Input System Switching Bit (CFGCOMP)

Selects whether the CFG input signal system is set to the zero cross type comparator system or digital signal input system.

Bit 5

| CFGCOMP | Description |
|---------|--|
| 0 | CFG signal input system is set to the zero cross type comparator system (Initial value) |
| 1 | CFG signal input system is set to the digital signal input system |

Bit 4: EXCTL Pin Switching Bit (EXCTLON)

Selects whether the EXCTL/PS4 pin is used as the EXCTL input pin or PS4 (general-purpose I/O pin).

Bit 4

| EXCTLON | Description |
|---------|---|
| 0 | EXCTL/PS4 pin functions as EXCTL input pin (Initial value) |
| 1 | EXCTL/PS4 pin functions as PS4 I/O |

Bit 3: DPG Pin Switching Bit (DPGSW)

Selects the drum control system input signals (DFG, DPG) as separate or overlapped inputs.

Bit 3

| DPGSW | Description |
|-------|---|
| 0 | Drum control system inputs are separate inputs (DPG/PS3 pin functions as DPG input pin) (Initial value) |
| 1 | Drum control system inputs are overlapped inputs (DPG/PS3 pin functions as PS3 I/O pin) |

Bit 2: COMP Pin Switching Pin (COMP)

Selects whether the COMP/PS2 pin is used as the COMP input pin or PS2 (general-purpose I/O pin).

Bit 2

| COMP | Description |
|------|---|
| 0 | COMP/PS2 pin functions as COMP input pin (Initial value) |
| 1 | COMP/PS2 pin functions as PS2 I/O pin |

Bit 1: H.Amp SW Pin Switching Bit (H.Amp.SW)

Selects whether the H.Amp SW/PS1 pin is used as the H.Amp SW output pin or PS1 (general-purpose I/O pin).

Bit 1

| H.Amp.SW | Description |
|----------|---|
| 0 | H.Amp SW/PS1 pin functions as H.Amp SW output pin (Initial value) |
| 1 | H.Amp SW/PS1 pin functions as PS1 I/O pin |

Bit 0: C.Rotary Pin Switching Bit (C.Rot)

Selects whether the C.Rotary/PS0 pin is used as the C.Rotary output pin or PS0 (general-purpose I/O pin).

Bit 0

| C.Rot | Description |
|-------|---|
| 0 | C.Rotary/PS0 pin functions as C.Rotary output pin (Initial value) |
| 1 | C.Rotary/PS0 pin functions as PS0 I/O pin |

(2) Servo Control Register (SPCR)

| | | | | | | | | |
|-----------------|---|---|---|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | SPCR4 | SPCR3 | SPCR2 | SPCR1 | SPCR0 |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | W | W | W | W | W |

Controls input and output of each pin (PS4 to PS0) for each bit when the servo port/general-purpose port dual-purpose pin is used as the general-purpose port. If SPCR is set to 1, the corresponding PS4 to PS0 pins function as output pins; if cleared to 0, they function as input pins. Settings of SPCR and SPDR are valid if the corresponding pins are set to general-purpose I/O by SPMR.

SPCR is an 8-bit write-only register. If read is attempted, an undetermined value is read out.

Bits 7 to 5 are reserved bits. Writes are disabled.

SPCR is initialized to H'E0 by a reset or stand-by.

Bit n

| SPCRn | Description |
|-------|--|
| 0 | PSn pin functions as input (Initial value) |
| 1 | PSn pin functions as output |

(3) Servo Data Register (SPDR)

| | | | | | | | | |
|-----------------|---|---|---|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | SPDR4 | SPDR3 | SPDR2 | SPDR1 | SPDR0 |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | R/W | R/W | R/W | R/W | R/W |

Stores the data of each pin (PS4 to PS0) when the servo port/general-purpose dual-purpose pin is used as general-purpose port. If the port is accessed for read when SPCR is 1 (output), the SPDRn value is read directly. Accordingly, this register is not affected by the state of the pin. If the port is accessed for read when SPCR is 0 (input), the state of the pin is read out. SPDR is an 8-bit read/write register. Bits 7-5 are reserved. No write in it is valid. If read is attempted, an undetermined value is read out. SPCR is initialized to H'E0 by reset or stand-by.

(4) Servo Monitor Control Register (SVMCR)

| | | | | | | | | |
|-----------------|---|---|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | SVMCR5 | SVMCR4 | SVMCR3 | SVMCR2 | SVMCR1 | SVMCR0 |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Selects the monitor signal output to the SV1 and SV2 pins when the P82/SV1 pin is used as the SV1 monitor output pin or when the P83/SV2 pin is used as the SV2 monitor output pin.

SVMCR is an 8-bit read/write register. Bits 7 and 6 are reserved. Writes are disabled. If read is attempted, an undetermined value is read out. It is initialized to H'C0 by a reset or stand-by.

| Bit 5 | Bit 4 | Bit 3 | Description |
|--------|--------|--------|--|
| SVMCR5 | SVMCR4 | SVMCR3 | |
| 0 | 0 | 0 | Outputs REF30 signal to SV2 output pin (Initial value) |
| | | 1 | Outputs CAPREF30 signal to SV2 output pin |
| | 1 | 0 | Outputs CREF signal to SV2 output pin |
| | | 1 | Outputs CTLMONI signal to SV2 output pin |
| 1 | 0 | 0 | Outputs DVCFG signal to SV2 output pin |
| | | 1 | Outputs CFG signal to SV2 output pin |
| | 1 | 0 | Outputs DFG signal to SV2 output pin |
| | | 1 | Outputs DPG signal to SV2 output pin |

| Bit 2 | Bit 1 | Bit 0 | Description |
|--------|--------|--------|--|
| SVMCR2 | SVMCR1 | SVMCR0 | |
| 0 | 0 | 0 | Outputs REF30 signal to SV1 output pin (Initial value) |
| | | 1 | Outputs CAPREF30 signal to SV1 output pin |
| | 1 | 0 | Outputs CREF signal to SV1 output pin |
| | | 1 | Outputs CTLMONI signal to SV1 output pin |
| 1 | 0 | 0 | Outputs DVCFG signal to SV1 output pin |
| | | 1 | Outputs CFG signal to SV1 output pin |
| | 1 | 0 | Outputs DFG signal to SV1 output pin |
| | | 1 | Outputs DPG signal to SV1 output pin |

(5) CTL Gain Control Register (CTLGR)

| | | | | | | | | |
|-----------------|---|---|-----------------|-------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | CTLE/ \bar{A} | CTLFB | CTLGR3 | CTLGR2 | CTLGR1 | CTLGR0 |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

Sets the CTLFB switch in the CTL amplifier circuit to on/off and CTL amplifier gain.

CTLGR is an 8-bit read/write register. Bits 7 and 6 are reserved. No write in it is valid. If read is attempted, an undetermined value is read out. It is initialized to H'C0 by a reset or stand-by.

Bit 7 to 6: Reserved

Reserved bits; writes are disabled. If read was attempted, an undetermined value is read out.

Bit 5: CTL Selection Bit (CTLE/ \bar{A})

Controls whether the amplifier output or EXCTL is used as the CTLP signal supplied to the CTL circuit.

Bit 5

| CTLE/ \bar{A} | Description |
|-----------------|----------------------------|
| 0 | AMP output (Initial value) |
| 1 | EXCTL |

Bit 4: SW Bit of the Feedback Section of CTL Amplifier (CTLFB)

Turning on/off the SW of the feedback section allows adjustment of gain.

See figure 28.4, CTL Input Circuit.

Bit 4

| CTLFB | Description |
|-------|------------------------------------|
| 0 | Turns off CTLFB SW (Initial value) |
| 1 | Turns on CTLFB SW |

Bits 3 to 0: CTL Amplifier Gain Setting Bits (CTLGR3 to 0)

Set the output gain of the CTL amplifier.

| Bit 3 | Bit 2 | Bit 1 | Bit 0 | CTL Output Gain | |
|--------|--------|--------|--------|-----------------|-----------------|
| CTLGR3 | CTLGR2 | CTLGR1 | CTLGR0 | | |
| 0 | 0 | 0 | 0 | 34.0 dB | (Initial value) |
| | | | 1 | 36.5 dB | |
| | | 1 | 0 | 39.0 dB | |
| | | | 1 | 41.5 dB | |
| | 1 | 0 | 0 | 44.0 dB | |
| | | | 1 | 46.5 dB | |
| | | 1 | 0 | 49.0 dB | |
| | | | 1 | 51.5 dB | |
| 1 | 0 | 0 | 0 | 54.0 dB | |
| | | | 1 | 56.5 dB | |
| | | 1 | 0 | 59.0 dB | |
| | | | 1 | 61.5 dB | |
| | 1 | 0 | 0 | 64.0 dB* | |
| | | | 1 | 66.5 dB* | |
| | | 1 | 0 | 69.0 dB* | |
| | | | 1 | 71.5 dB* | |

Note: * With a setting of 64.0 dB or more, the CTLAMP is in a very sensitive status. When configuring the set board, be concerned about countermeasure against noise around the control head signal input port. Also, thoroughly set the filter between the CTLAMP and CTLSMT.

28.2.6 DFG/DPG Input Signals

DFG and DPG signals allow either separate or overlapped input. If the latter was selected (DPGSW = 1), take care in the input levels of DFG and DPG. Figure 28.5 shows DFG/DPG input signals.

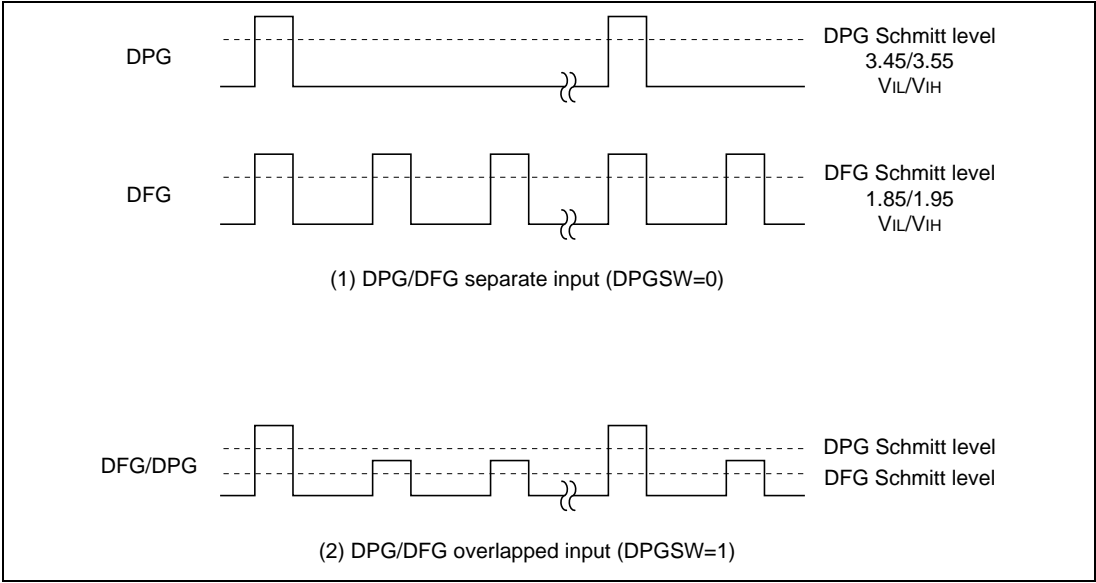


Figure 28.5 DFG/DPG Input Signals

28.3 Reference Signal Generators

28.3.1 Overview

The reference signal generators consist of REF30 signal generator and CREF signal generator, and they create the reference signals (REF30 and CREF signals) used in phase comparison, etc. The REF30 signal is used to control the phase of the drum and capstan. The CREF signal is used if the reference signal to control the phase of the capstan cannot be shared with the REF30 signal in REC mode. Each signal generator consists of a 16-bit counter which has the servo clock $\phi s/2$ (or $\phi s/4$) as its clock source, a reference period register and a comparator. The value set in the reference period register should be 1/2 of the desired reference signal period.

28.3.2 Block Diagram

Figure 28.6 shows the block diagram of the REF30 signal generator. Figure 28.7 shows that of the CREF signal generator.

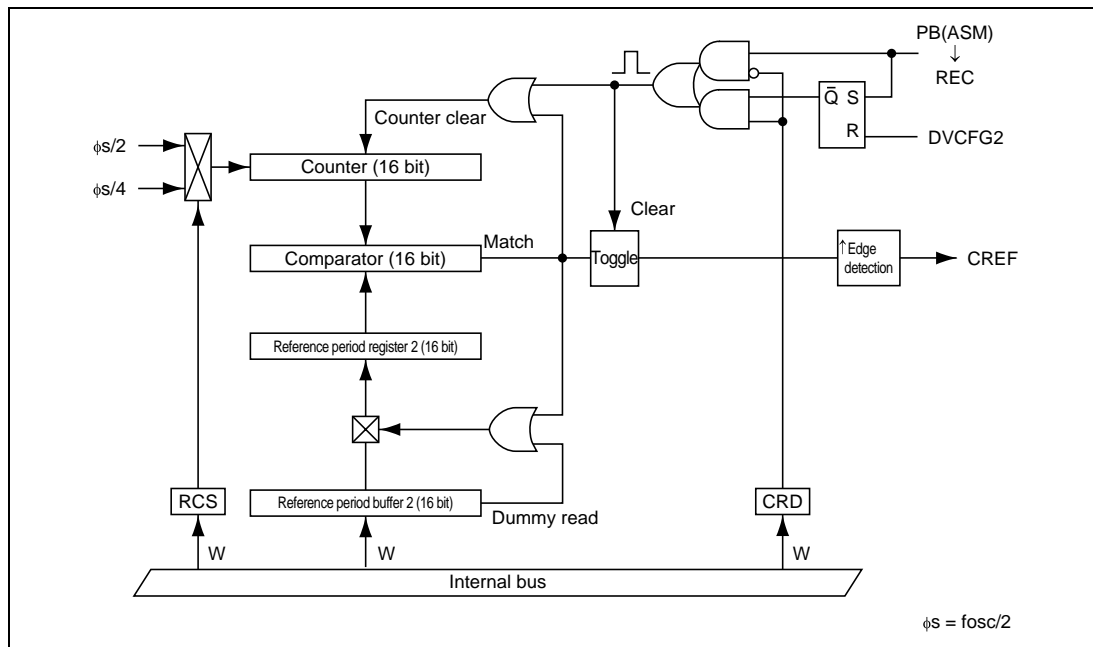


Figure 28.7 Block Diagram of CREF Signal Generator

28.3.3 Register Configuration

Table 28.4 shows the register configuration of the reference signal generators.

Table 28.4 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|----------------------------------|---------|-----|------|---------------|---------|
| Reference period mode register | RFM | W | Byte | H'00 | H'FD096 |
| Reference period register 1 | RFD | W | Word | H'FFFF | H'FD090 |
| Reference period register 2 | CRF | W | Word | H'FFFF | H'FD092 |
| REF30 counter register | RFC | R/W | Word | H'0000 | H'FD094 |
| Reference period mode register 2 | RFM2 | R/W | Byte | H'FE | H'FD097 |

28.3.4 Register Descriptions

(1) Reference Period Mode Register (RFM)

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-------|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | RCS | VNA | CVS | REX | CRD | OD/EV | VST | VEG |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

RFM is an 8-bit write-only register which determines the operational state of the reference signal generators. If a read is attempted, an undetermined value is read out.

It is initialized to H'00 by a reset, stand-by or module stop.

RFM is accessible by byte access only. If accessed by a word, its operation is not assured.

Bit 7: Clock Source Selection Bit (RCS)

Selects the clock source supplied to the counter. ($\phi_s = f_{osc}/2$)

Bit 7

| RCS | Description |
|-----|----------------------------|
| 0 | $\phi_s/2$ (Initial value) |
| 1 | $\phi_s/4$ |

Bit 6: Mode Selection Bit (VNA)

Selects whether the transition to free-run operation when the REF30 signals are being generated in sync with the VD signals in REC mode is controlled automatically by the V noise detection signal, which has been detected by the sync signal detection circuit, or is controlled manually by software.

Bit 6

| VNA | Description |
|-----|-----------------------------|
| 0 | Manual mode (Initial value) |
| 1 | Auto mode |

Bit 5: Manual Selection Bit (CVS)

Selects whether the REF30 signals are generated in sync with VD or operated free-run in manual mode (VNA = 0). (No selection is reflected in PB mode, except in TBC mode.)

Bit 5

| CVS | Description | |
|-----|--------------------|-----------------|
| 0 | Sync with VD | (Initial value) |
| 1 | Free-run operation | |

Bit 4: External Signals Sync Selection Bit (REX)

Selects whether the REF30 signals are generated in sync with VD or in free-run or in sync with the external signals. (Valid in both PB and REC modes.)

Bit 4

| REX | Description | |
|-----|----------------------------|-----------------|
| 0 | VD signals or free-run | (Initial value) |
| 1 | Sync with external signals | |

Bit 3: DVCFG2 Sync Selection Bit (CRD)

Selects whether the reset timing in the CREF signals generation is immediately after switching from PB (ASM) mode to REC mode or is in sync with the DVCFG2 signals immediately after the switching.

Bit 3

| CRD | Description | |
|-----|-----------------------------|-----------------|
| 0 | On switching modes | (Initial value) |
| 1 | In sync with DVCFG2 signals | |

Bit 2: ODD/EVEN Edge Switching Selection Bit (OD/EV)

Selects whether REF30P signals are generated by ODD of the field signals or EVEN when in REC mode.

Bit 2

| OD/EV | Description | |
|-------|--|-----------------|
| 0 | Generated at the rising edge (EVEN) of the field signals | (Initial value) |
| 1 | Generated at the falling edge (ODD) of the field signals | |

Bit 1: Video FF Counter Set (VST)

Selects whether the REF30 counter register value is set on or off by the Video FF signal when the drum phase is in FIX on in PB mode.

Bit 1

| VST | Description |
|------------|--|
| 0 | Counter set off by Video FF signal (Initial value) |
| 1 | Counter set on by Video FF signal |

Bit 0: Video FF Edge Selection Bit (VEG)

Selects the edge at which the REF30 counter is set ($VST = 1$) by the Video FF signal.

Bit 0

| VEG | Description |
|------------|---|
| 0 | Set at the rising edge of Video FF signal (Initial value) |
| 1 | Set at the falling edge of Video FF signal |

(2) Reference Period Register 1 (RFD)

| | | | | | | | | | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | REF15 | REF14 | REF13 | REF12 | REF11 | REF10 | REF9 | REF8 | REF7 | REF6 | REF5 | REF4 | REF3 | REF2 | REF1 | REF0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

The reference period register 1 (RFD) is a buffer register which generates the reference signals for playback (REF30), VD compensation for recording and the reference signals for free-running. It is a 16-bit write-only register accessible by a word only. If a read is attempted, an undetermined value is read out.

The value set in RFD should be 1/2 of the desired reference signal period. Care is required when VD is unstable, such as when the field is weak (Synchronization with VD cannot be acquired if a value less than 1/2 is set when in REC). When data is written in RFD, it is stored in the buffer once, and then fetched into RFD by a match signal of the comparator. (The data which generates the reference signal is updated from time to time by the match signal.) An enforced write, such as initial setting, etc., should be done by a dummy read of RFD.

If a byte-write in RFD is attempted, no operation is assured. RFD is initialized to H'FFFF by a reset, stand-by, or module stop.

Use bit 7 (ASM) and bit 6 (REC/PB) in the CTL mode register (CTLM) in the CTL circuit to switch between record and playback modes. Use bit 4 (CR/RF bit) in the capstan phase error detection control register (CPGCR) to switch between REF30 and CREF for capstan phase control.

(3) Reference Period Register 2 (CRF)

| | | | | | | | | | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CRF15 | CRF14 | CRF13 | CRF12 | CRF11 | CRF10 | CRF9 | CRF8 | CRF7 | CRF6 | CRF5 | CRF4 | CRF3 | CRF2 | CRF1 | CRF0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

The reference period register 2 (CRF) is a 16-bit write-only buffer register which generates the reference signals to control the capstan phase (CREF). CRF is accessibly by a word only. If a read is attempted, an undetermined value is read out. The value set in CRF should be 1/2 of the desired reference signal period.

When data is written in CRF, it is stored in the buffer once, and then fetched into CRF by a match signal of the comparator. (The data which generates the reference signal is updated from time to time by the match signal.) An enforced write, such as initial setting, etc., should be done by a dummy read of CRF.

If a byte-write in CRF is attempted, no operation is assured. CRF is initialized to H'FFFF by a reset, stand-by, or module stop.

Use bit 4 (CR/RF bit) in the capstan phase error detection control register (CPGCR) to switch between REF30 and CREF for capstan phase control. (See section 28.9, Capstan Phase Error Detector)

(4) REF30 Counter Register (RFC)

| | | | | | | | | | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | RFC15 | RFC14 | RFC13 | RFC12 | RFC11 | RFC10 | RFC9 | RFC8 | RFC7 | RFC6 | RFC5 | RFC4 | RFC3 | RFC2 | RFC1 | RFC0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

The REF30 counter register (RFC) is a register which determines the initial value of the free-run counter when it generates REF30 signals when in playback. When data is written in RFC, its value is written in the counter by a match signal of the comparator. If bit 1 (VST) in RFM is set to 1, the counter is set by the Video FF signal when the drum phase is in FIX ON. The counter setting by the Video FF signal should be done by setting RFM's bit 1 (VST) and bit 0 (VEG). Don't set the RFC value at a value greater than 1/2 of the reference period register 1 (RFD). RFC is a read/write register. If a read is attempted, the value of the counter is read out. If a byte-access is attempted, no operation is assured. RFC is initialized to H'0000 by a reset, stand-by, or module stop.

(5) Reference Period Mode Register 2 (RFM2)

| | | | | | | | | |
|-----------------|--------|---|---|---|---|---|---|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | (TBC) | — | — | — | — | — | — | FDS |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| R/W : | (R/W)* | — | — | — | — | — | — | R/W |

Note:* Writable only in the H8S/2194C Series.

RFM2 is an 8-bit read/write register which determines the operational state of the reference signal generators. Bits 7 to 1 are reserved. If a read is attempted, an undetermined value is read out.

It is initialized to H'FE by a reset, stand-by or module stop. RFM2 is a byte access-only register; if accessed by a word, no operation is assured.

Bits 7: TBC Selection Bit (TBC)

Selects whether the reference signals are generated by VD or in free-run in PB mode.

Bit 7

| TBC | Description |
|-----|---|
| 0 | Reference signals are generated by VD (This function is effective only in the H8S/2194C series) |
| 1 | Reference signals are generated in free-run (Initial value) |

Bits 6 to 1: Reserved

No write is valid. If a read is attempted, an undetermined value is read out.

Bit 0: Field Selection Bit (FDS)

Determines whether selection between ODD or EVEN is made for the field signals when PB mode was switched over to REC mode, or these signals are synchronized with VD signals within phase error of 90° immediately after the switching over.

Bit 0

| FDS | Description |
|-----|--|
| 0 | Generated by the VD signal of ODD or EVEN selected (Initial value) |
| 1 | Generated by the VD signal within mode transition phase error of 90° |

28.3.5 Description of Operation

(1) Operation of REF30 Signal Generators

The REF30 signal generators generate the reference signals required to control the phase of the drum and capstan.

To generate REF30 signals, set the half-period value to the reference period register 1 (RFD) corresponding to the 50% duty cycle. When in playback, REF30 signals are generated by operating REF30 signal generator in free-run. The generator has the external signal synchronization function built-in, and if bit 4 (REX) of the reference period mode register (RFM) is set to 1, it generates REF30 signals from external signals (EXTTRG).

In record mode, the reference signals are generated from the VD signal generated in the sync signal detection circuit. Any VD drop-out caused by weak field intensity, etc., is compensated by a set value of RFD. To cope with the VD noises, the generator performs automatically the VD masking for a time period about 75% of the RFD setting after REF30 signal was changed due to VD. In record mode, the generation of the reference signals either by VD or free-run operation can be controlled automatically or by software, using the V noise detection signal detected in the sync signal detection circuit. Select which is used by setting bit 6 (VNA) or 5 (CVS) of RFM.

The phase of the toggle output of the REF30 signal is cleared to L level when the signal mode transits from PB to REC (ASM). Also the frame servo function can be set, allowing to control the phase of REF30 signals with the field signal detected in the sync signal detection circuit. Use bit 2 (OD/EV) of RFM for such control.

See section 28.13.5(2), CTL Mode Register (CTLM) as for switching over between PB, ASM and REC.

(2) Operation of the Mask Circuit

The REF30 signal generators have a toggle mask circuit and counter mask (counter set signal mask) circuit built-in. Each mask circuit masks irregular VD signals which may occur when the VD signal is unstable because of weak field intensity, etc., in record mode.

The toggle mask and counter mask circuits mask the VD automatically for about 75% of double the time period set in the reference period register 1 (RFD) after a VD signal was detected (see figure 28.9). If a VD signal dropped out and V was compensated, the toggle mask circuit begins masking. The counter mask circuit does not do so for about 25% of the time period. If a VD signal was detected during such time period, it does masking for about 75% of the time period. If not detected, it does for the same time period after V was compensated (see figures 28.10 and 28.11).

(3) Timing of the REF30 Signal Generation

Figures 28.8, 28.9, 28.10, 28.11 and 28.12 show the timing of the generation of REF30 and REF30P signals.

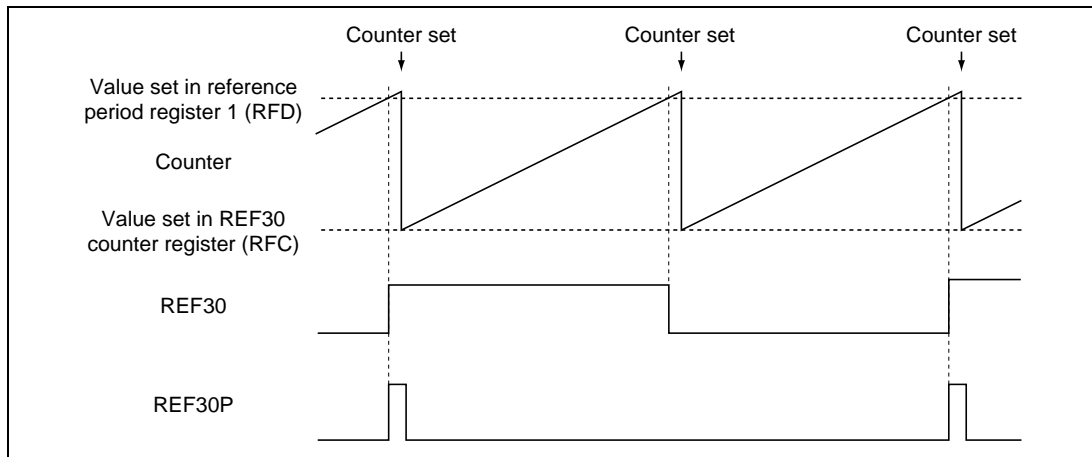


Figure 28.8 REF30 Signals in Playback Mode

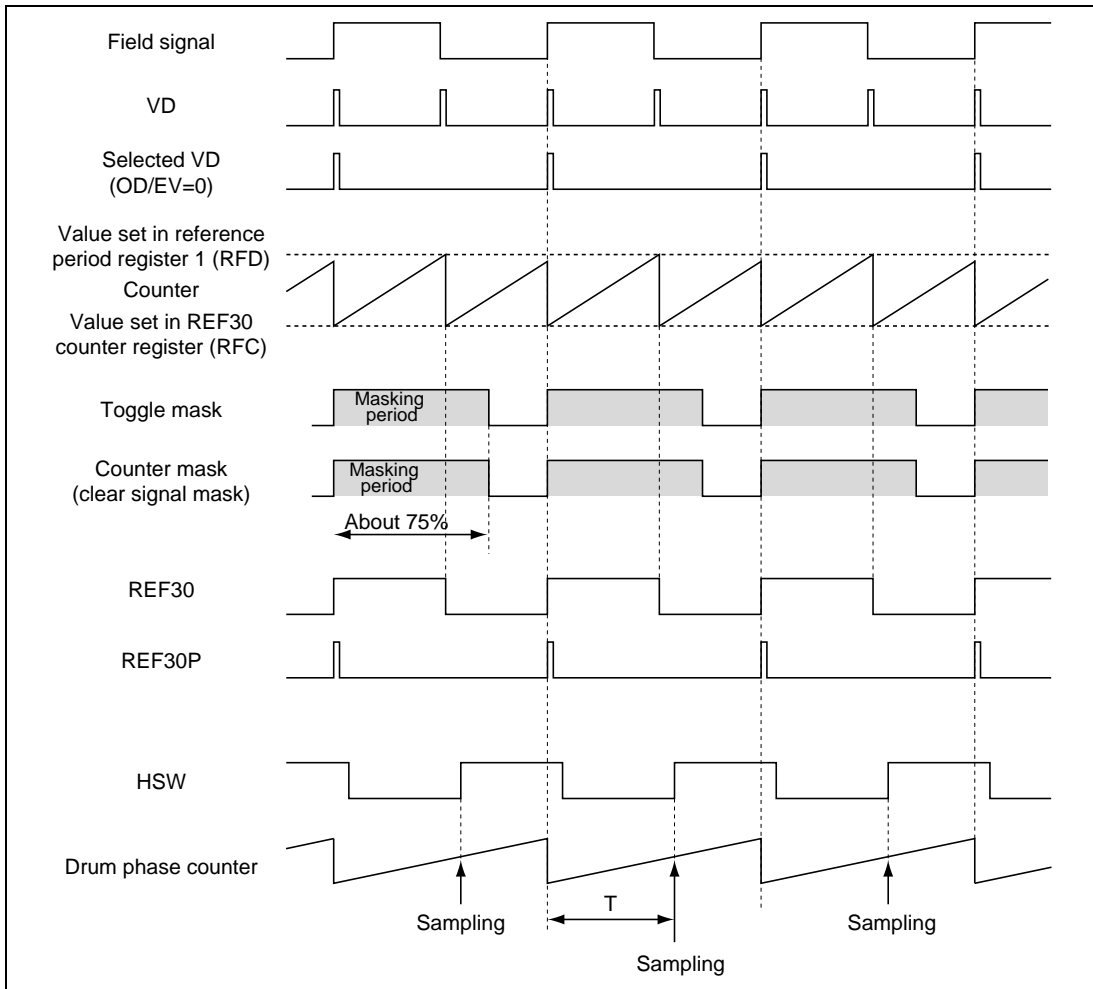


Figure 28.9 Generation of Reference Signal in Record Mode (Normal Operation)

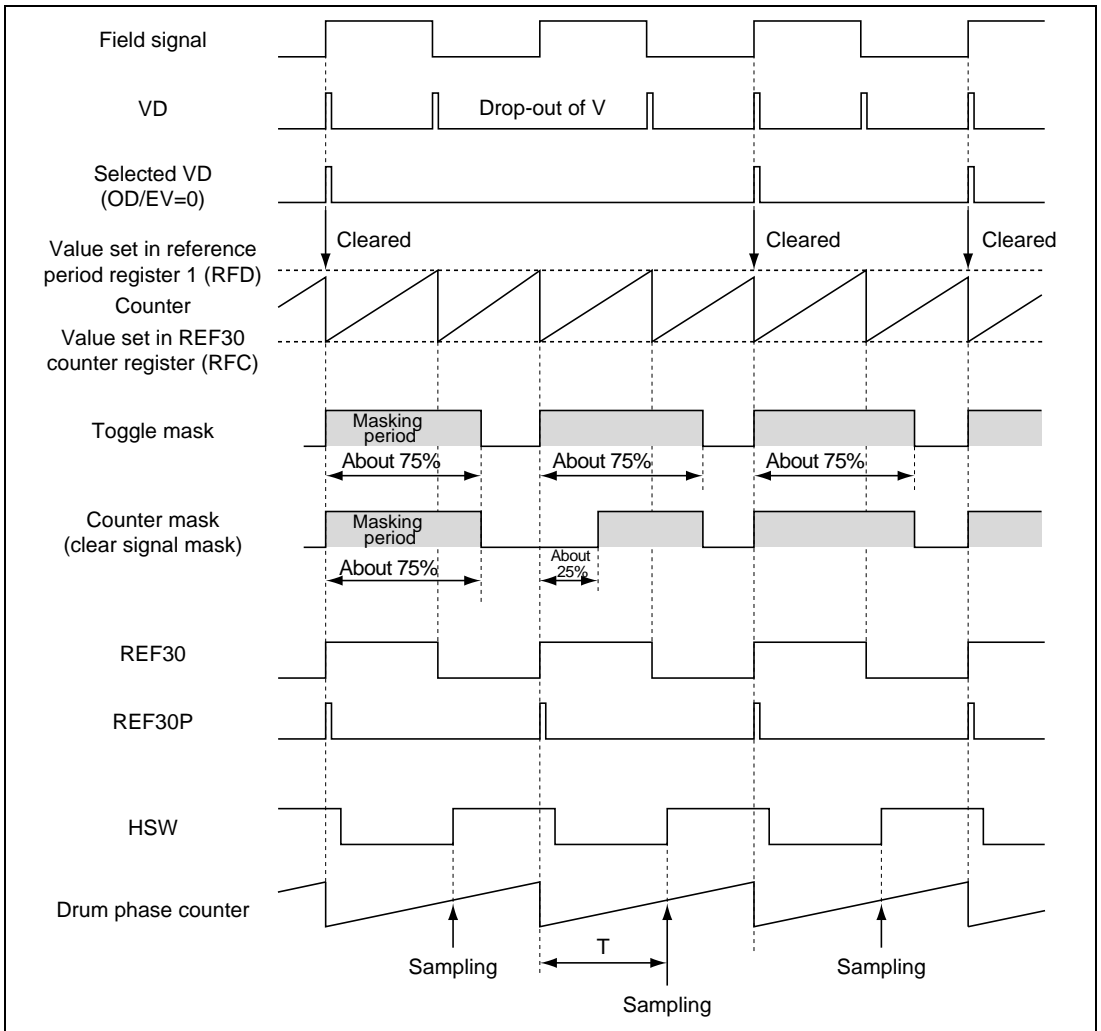


Figure 28.10 Generation of Reference Signal in Record Mode (V Dropped Out)

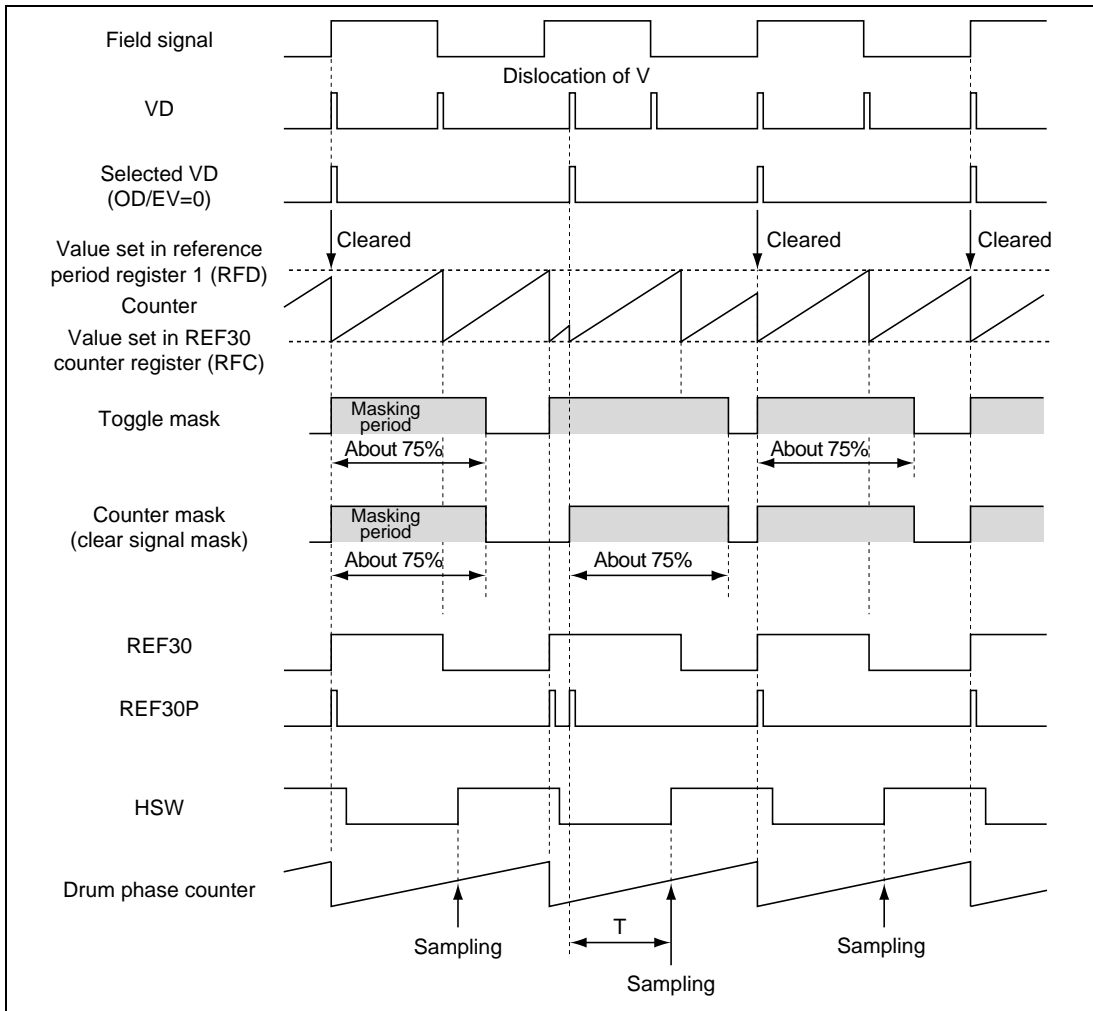


Figure 28.11 Generation of Reference Signal in Record Mode (V Dislocated)

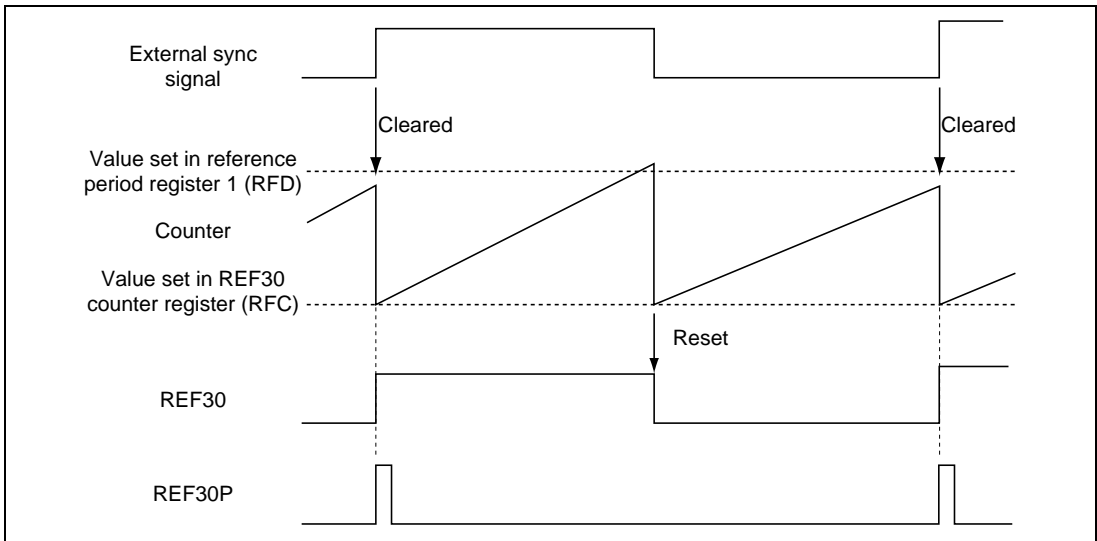


Figure 28.12 Generation of REF30 Signal by External Sync Signal

(4) CREF Signal Generator

The CREF signal generator generates a CREF signal which is the reference signal to control the phase of the capstan.

To generate CREF signals, set the half-period value to the reference period register 2 (CRF). If the set value matches the counter value, a toggle waveform is generated corresponding to the 50% duty cycle, and a one-shot pulse signal is output at the rising edge of the waveform. The counter of the CREF signal generator is initialized to H'0000 and the phase of the toggle is cleared to L level at the mode transition of PB (ASM) to REC. The timing of clearing is selectable between immediately after the transition from PB (ASM) to REC and the timing of DVCFG2 after the transition. Use bit 3 (CRD) of the reference period mode register (RFM) for the selection.

In the capstan phase error detection circuit, either the REF30 signal or CREF signal can be selected for the reference signal. Use either of them according to the use of the system.

Use the CREF signal to control the phase of the capstan at a period which is different from the period used to control the phase of the drum. As for the switching between REF30 and CREF in the capstan phase control, see section 28.9.4(3) Capstan Phase Error Detection Control Register (CPGCR).

(5) Timing Chart of the CREF Signal Generation

Figures 28.13, 28.14 and 28.15 show the generation of the CREF signal.

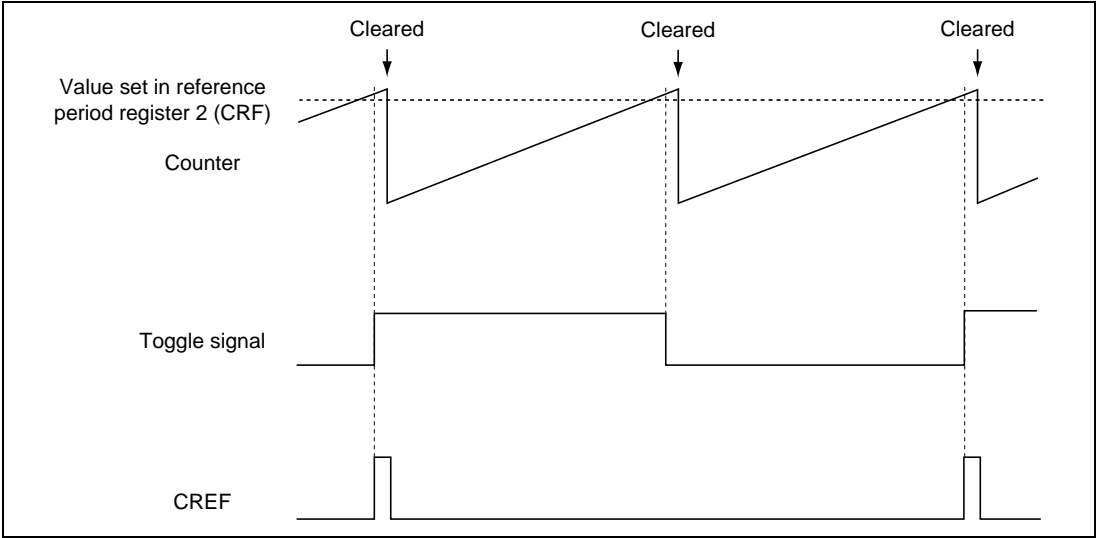


Figure 28.13 Generation of CREF Signal

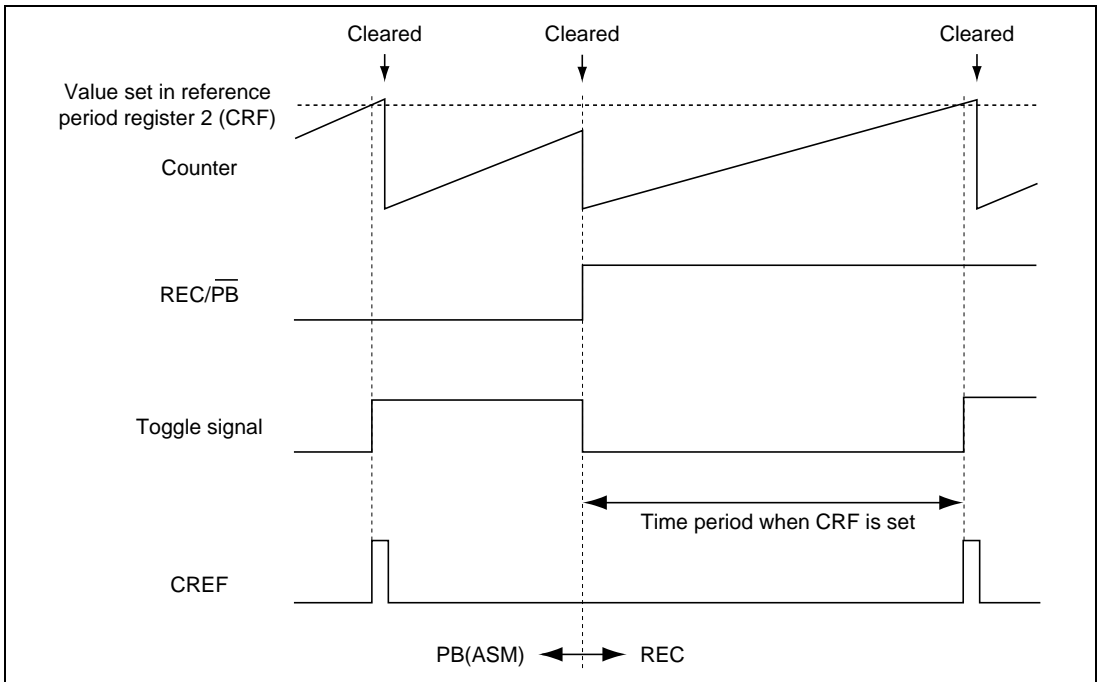


Figure 28.14 CREF Signal when PB is Switched to REC (when CRD Bit = 0)

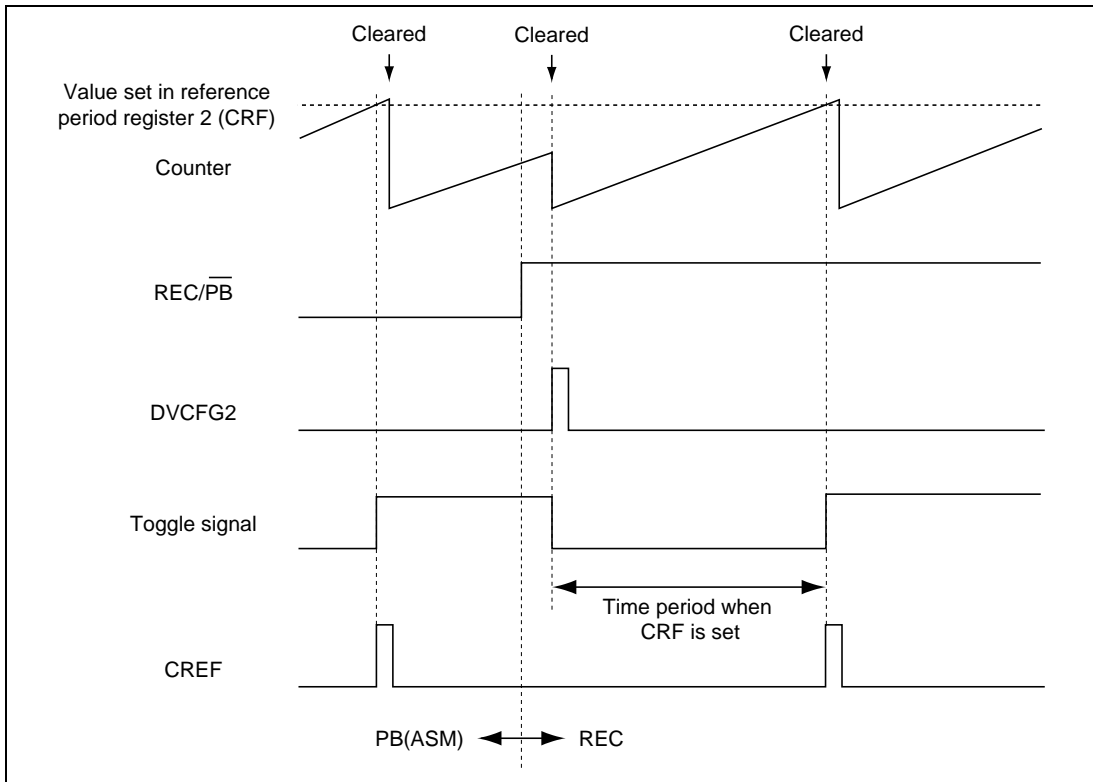


Figure 28.15 CREF Signal when PB is Switched to REC (when CRD Bit = 1)

Figures 28.16 and 28.17 show REF30 (REF30P) when PB is switched to REC.

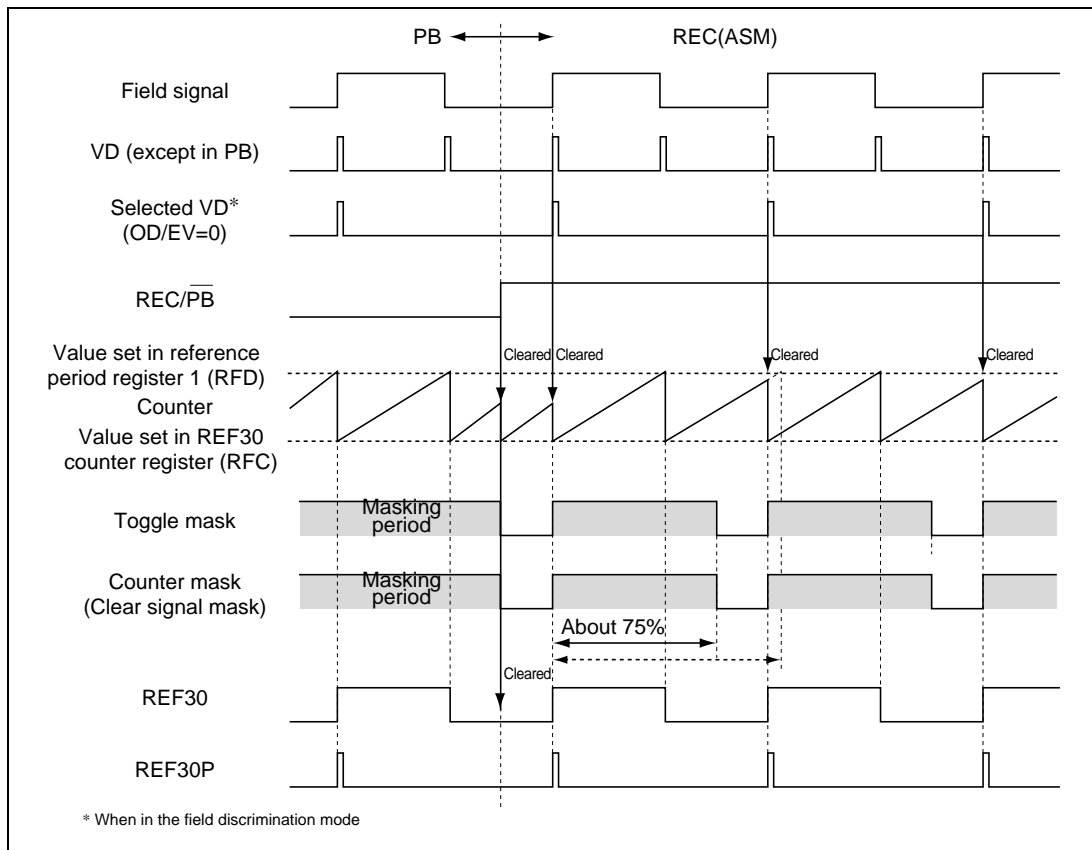


Figure 28.16 Generation of the Reference Signal when PB is Switched to REC (1)

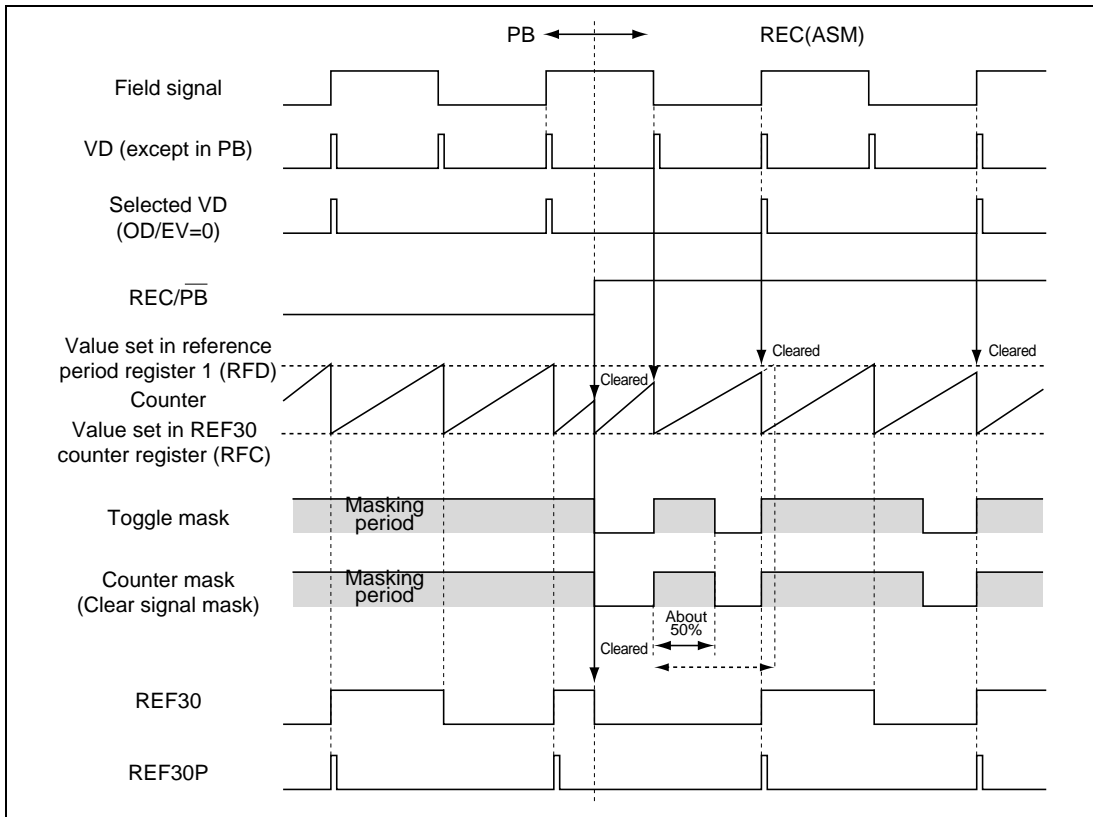


Figure 28.17 Generation of the Reference Signal when PB is Switched to REC (2)

Figures 28.18, 28.19, 28.20 and 28.21 show REF30 (REF30P) when PB is switched to REC (where FDS bit = 1).

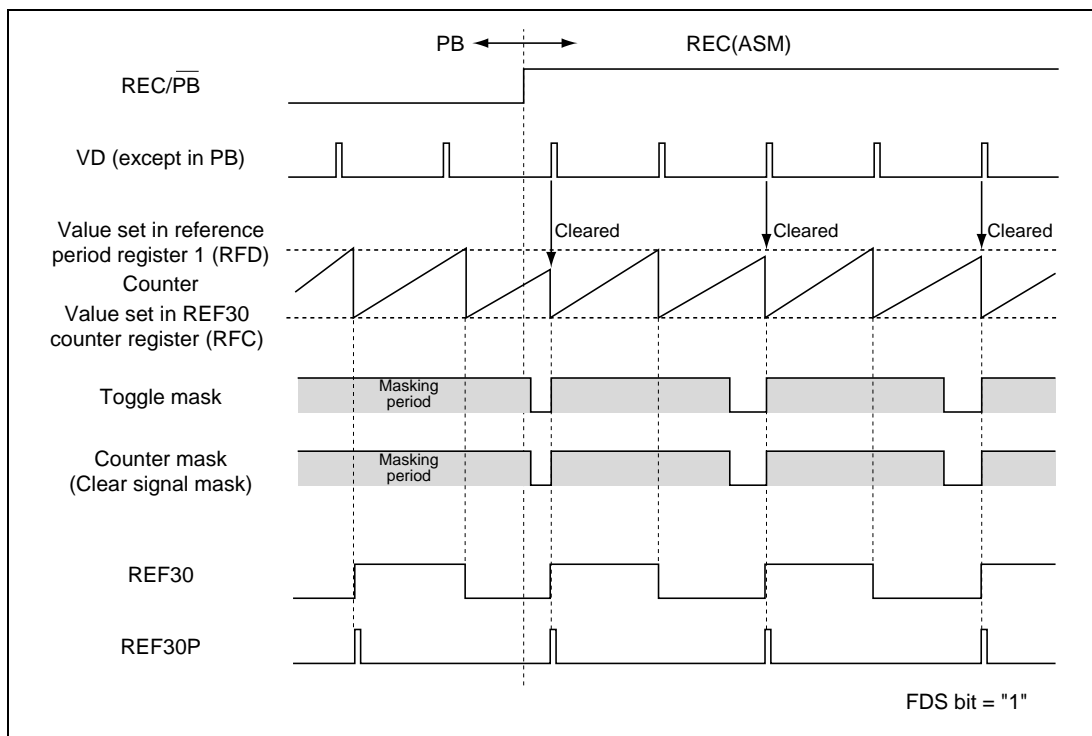


Figure 28.18 Generation of the Reference Signal when PB is Switched to REC where RFD Bit is 1 (1)

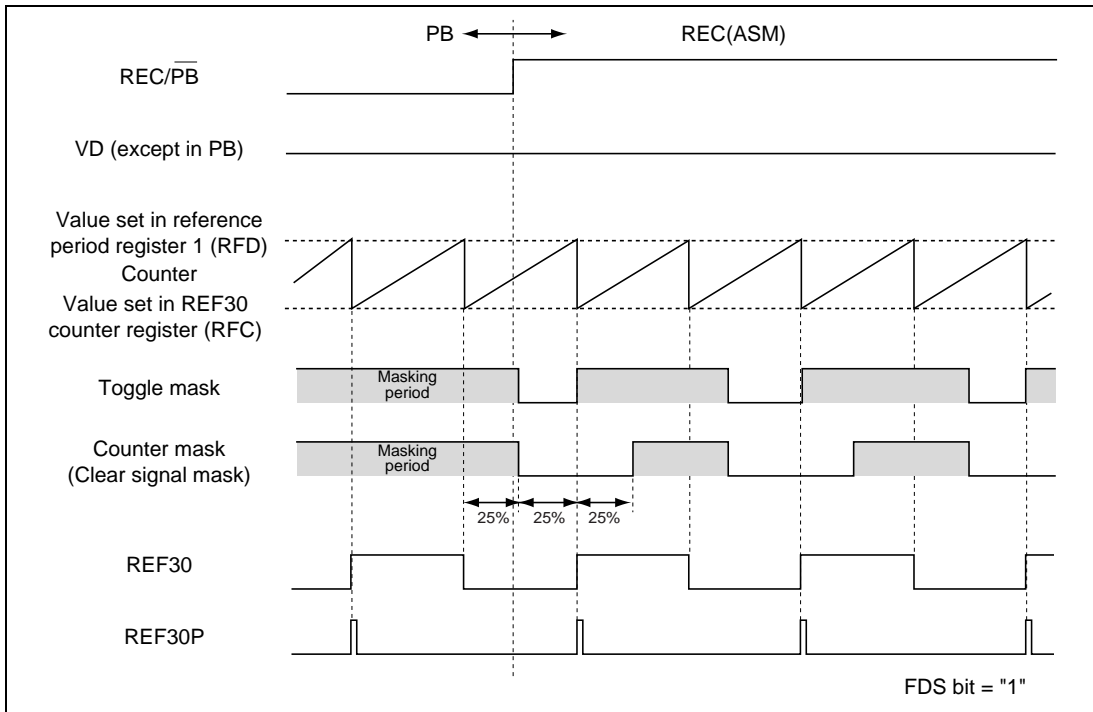


Figure 28.19 Generation of the Reference Signal when PB is Switched to REC where RFD Bit is 1 (when VD Signal is Not Detected) (2)

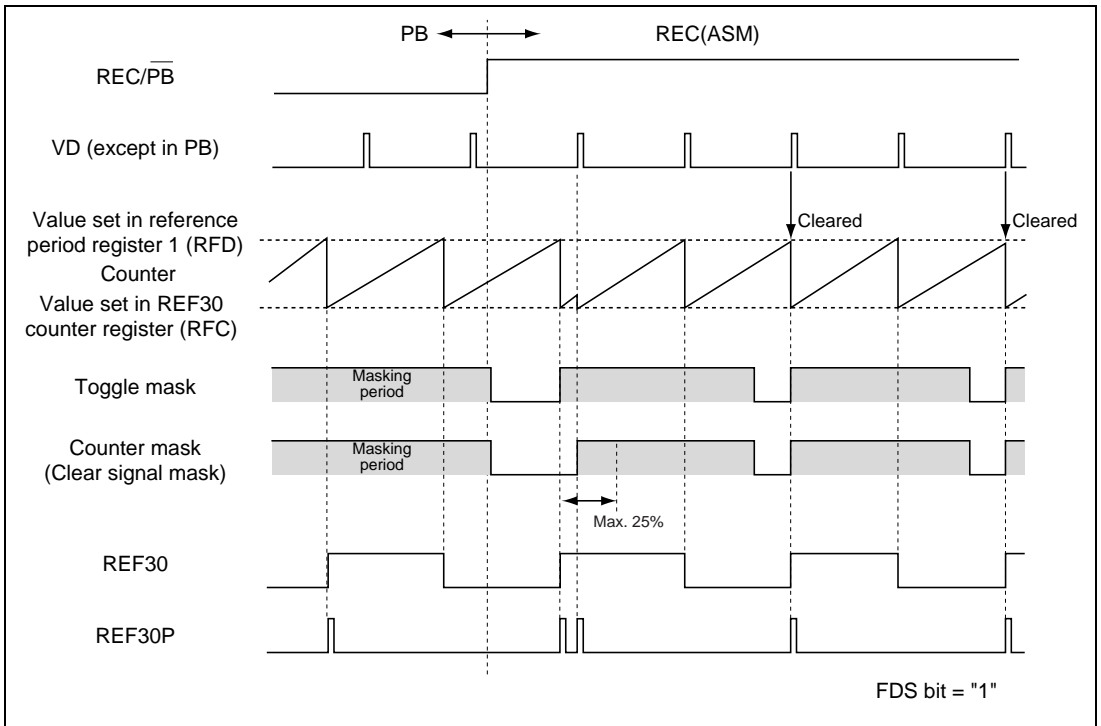


Figure 28.20 Generation of the Reference Signal when PB is Switched to REC where RFD Bit is 1 (3)

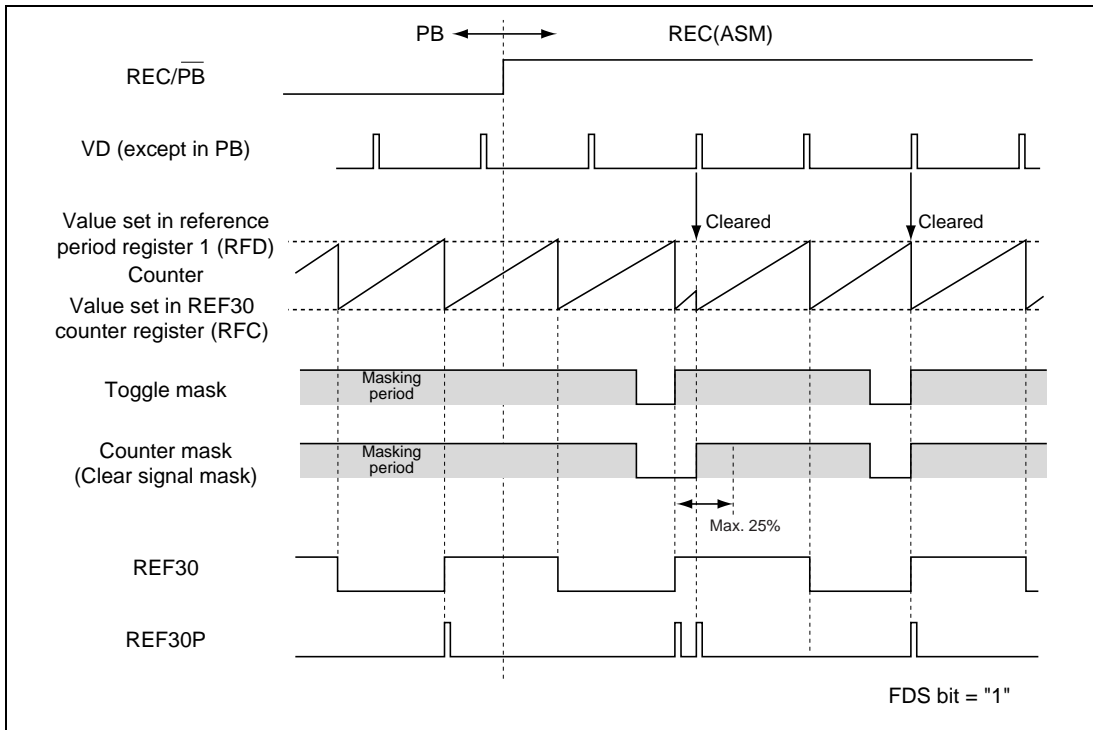


Figure 28.21 Generation of the Reference Signal when PB is Switched to REC where RFD Bit is 1 (4)

28.4 HSW (Head-switch) Timing Generator

28.4.1 Overview

The HSW timing generator consists of one 5-bit counter and one 16-bit counter, matching circuit, and two 31-bit 10-stage FIFOs.

The 5-bit counter counts the DFG pulses following a DPG pulse. Each of them determines the timing to reset the 16-bit timer for each field. The matching circuit compares the timing data in the most significant 16 bits of FIFO with the 16-bit timer, and controls the output of pattern data set in the least significant 15 bits of FIFO. The 16-bit timer is a timer clocked by a ϕ s/4 clock source, and can be used as a PPG (Programmable Pattern Generator) as well as a free-running counter (FRC). If used as a free-running counter, it is cleared by overflow (FRCOVF) of the Prescaler unit. Accordingly, two free-running counter operate in sync.

28.4.2 Block Diagram

Figure 28.22 shows a block diagram of the HSW timing generator.

28.4.3 Composition

The HSW timing generator is composed of the elements shown in table 28.5.

Table 28.5 Composition of the HSW Timing Generator

| Element | Function |
|---|---|
| HSW mode register 1 (HSM1) | Confirmation/determination of this circuits' operating status |
| HSW mode register 2 (HSM2) | Confirmation/determination of this circuits' operating status |
| HSW loop stage number setting register (HSLP) | Setting of number of loop stages in loop mode |
| FIFO output pattern register 1 (FPDRA) | Output pattern data register of FIFO1 |
| FIFO output pattern register 2 (FPDRB) | Output pattern data register of FIFO2 |
| FIFO timing pattern register 1 (FTPRA) | Output timing register of FIFO1 |
| FIFO timing pattern register 2 (FTPRB) | Output timing register of FIFO2 |
| DFG reference register 1 (DFCRA) | Setting of reference DFG edge for FIFO1 |
| DFG reference register 2 (DFCRB) | Setting of reference DFG edge for FIFO2 |
| FIFO timer capture register (FTCTR) | Capture register of timer counter |
| DFG reference count register (DFCTR) | DFG edge count |
| FIFO control circuit | Controls FIFO status |
| DFG count compare circuit (×2) | Detection of match between DFCR and DFG counters |
| 16-bit timer counter | 16-bit free-run timer counter |
| 31-bit x 20 stage FIFO | First In First Out data buffer |
| 31-bit FIFO data buffer | Data storing buffer for the first stage of FIFO |
| 16-bit compare circuit | Detection of match between timer counter and FIFO data buffer |

FPDRA and FPDRB are intermediate buffers; an FTPRA and FTPRB write results in simultaneous writing of all 31 bits to the FIFO. The FIFO has two 31-bit x 10-stage data buffers, its operating status being controlled by HSM1 and HSM2. Data is stored in the 31-bit data buffer. The values of FTPRA, FTPRB and the timer counter are compared, and if they match, the 15-bit pattern data is output to each function. AudioFF, VideoFF and PPG (P70 to P77) are pin outputs, ADTRG is the A/D converter hardware start signal, Vpulse and Mlevel signals are the signals to generate the additional V pulses, and HSW and NHSW signals are the same with VideoFF signals used for the phase control of the drum. In free-run mode (when FRT bit of HSM2 = 1), the 16-bit timer counter is initialized when the prescaler unit overflows, or by a signal indicating a match between DFCRA, DFCRB and the DFG counter in DFG reference mode.

28.4.4 Register Configuration

Table 28.6 shows the register configuration of the HSW timing generator.

Table 28.6 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|--|---------|-----|------|---------------|---------|
| HSW mode register 1 | HSM1 | R/W | Byte | H'30 | H'FD060 |
| HSW mode register 2 | HSM2 | R/W | Byte | H'00 | H'FD061 |
| HSW loop stage number setting register | HSLP | R/W | Byte | Undetermined | H'FD062 |
| FIFO output pattern register 1 | FPDRA | W | Word | Undetermined | H'FD064 |
| FIFO timing pattern register 1* | FTPRA | W | Word | Undetermined | H'FD066 |
| FIFO output pattern register 2 | FPDRB | W | Word | Undetermined | H'FD068 |
| FIFO timing pattern register 2 | FTP RB | W | Word | Undetermined | H'FD06A |
| DFG reference register 1* | DFCRA | W | Byte | Undetermined | H'FD06C |
| DFG reference register 2 | DFCRB | W | Byte | Undetermined | H'FD06D |
| FIFO timer capture register* | FTCTR | R | Word | H'0000 | H'FD066 |
| DFG reference count register* | DFCTR | R | Byte | H'E0 | H'FD06C |

Note: * FTPRA and FTCTR, as well as DFCRA and DFCTR, are allocated to the same addresses.

28.4.5 Register Descriptions

(1) HSW Mode Register 1 (HSM1)

| | | | | | | | | |
|-----------------|-----|-----|------|------|--------|--------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FLB | FLA | EMPB | EMPA | OVWB | OVWA | CLRB | CLRA |
| Initial value : | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R/(W)* | R/(W)* | R/W | R/W |

Note: * Only 0 can be written

HSM1 is a register which confirms and determines the operational state of the HSW timing generator.

HSM1 is an 8-bit register. Bits 7 to 4 are read-only bits, and write is disabled. All the other bits accept both read and write. It is initialized to H'30 by a reset or stand-by.

Bit 7: FIFO2 Full Flag (FLB)

When the FLB bit is 1, it indicates that the timing pattern data and the output pattern data of FIFO2 are full. If a write is attempted in this state, the write operation becomes invalid, an interrupt is generated, the OVWB flag (bit 3) is set to 1, and the write data is lost. Wait until space becomes available in the FIFO2, then write again.

Bit 7

| FLB | Description |
|-----|--|
| 0 | FIFO2 is not full, and can accept data input (Initial value) |
| 1 | FIFO2 is full |

Bit 6: FIFO1 Full Flag (FLA)

When the FLA bit is 1, it indicates that the timing pattern data and the output pattern data of FIFO1 are full. If a write is attempted in this state, the write operation becomes invalid, an interrupt is generated, the OVWA flag (bit 2) is set to 1, and the write data is lost. Wait until space becomes available in the FIFO1, then write again.

Bit 6

| FLA | Description |
|-----|--|
| 0 | FIFO1 is not full, and can accept data input (Initial value) |
| 1 | FIFO1 is full |

Bit 5: FIFO2 Empty Flag (EMPB)

Indicates that FIFO2 has no data, or that all the data has been output in single mode.

Bit 5

| EMPB | Description |
|------|--|
| 0 | FIFO2 contains data |
| 1 | FIFO2 contains no data (Initial value) |

Bit 4: FIFO1 Empty Flag (EMPA)

Indicates that FIFO1 has no data, or that all the data has been output in single mode.

Bit 4

| EMPA | Description |
|------|--|
| 0 | FIFO1 contains data |
| 1 | FIFO1 contains no data (Initial value) |

Bit 3: FIFO2 Overwrite Flag (OVWB)

If a write is attempted when the timing pattern data and the output pattern data of FIFO2 are full (FLB bit = 1), the write operation becomes invalid, an interrupt is generated, the OVWB flag is set to 1, and the write data is lost. Wait until space becomes available in the FIFO2, then write again.

Write 0 to clear the OVWB flag, because it is not cleared automatically.

Bit 3

| OVWB | Description |
|------|---|
| 0 | Normal operation (Initial value) |
| 1 | Indicates that a write in FIFO2 was attempted when FIFO2 was full. Clear this flag by 0 writing |

Bit 2: FIFO1 Overwrite Flag (OVWA)

If a write is attempted when the timing pattern data and the output pattern data of FIFO1 are full (FLA bit = 1), the write operation becomes invalid, an interrupt is generated, the OVWA flag is set to 1, and the write data is lost. Wait until space becomes available in the FIFO1, then write again.

Write 0 to clear the OVWA flag, because it is not cleared automatically.

Bit 2

| OVWA | Description |
|------|---|
| 0 | Normal operation (Initial value) |
| 1 | Indicates that a write in FIFO1 was attempted when FIFO1 was full. Clear this flag by 0 writing |

Bit 1: FIFO2 Pointer Clear (CLRB)

Clears the FIFO2 write position pointer. After 1 is written, the bit immediately reverts to 0. Writing 0 in this bit has no effect.

Bit 1

| CLRB | Description |
|------|----------------------------------|
| 0 | Normal operation (Initial value) |
| 1 | Clears the FIFO2 pointer |

Bit 0: FIFO1 Pointer Clear (CLRA)

Clears the FIFO1 write position pointer. After 1 is written, the bit immediately reverts to 0. Writing 0 in this bit has no effect.

Bit 0

| CLRA | Description |
|-------------|----------------------------------|
| 0 | Normal operation (Initial value) |
| 1 | Clears the FIFO1 pointer |

(2) HSW Mode Register 2 (HSM2)

| | | | | | | | | |
|-----------------|-----|---------|-----|-----|-------|------|-----|---------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FRT | FGR2OFF | LOP | EDG | ISEL1 | SOFG | OFG | VFF/NFF |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R | R/W | R/W | R/W | R/W | R | W |

HSM2 is a register which confirms and determines the operational state of the HSW timing generator.

HSM2 is an 8-bit register. Bits 6 and 1 are read-only bits, and write is disabled. Bit 0 is a write-only bit, and if a read is attempted, an undetermined value is read out. All the other bits accept both read and write. It is initialized to H'00 by a reset or stand-by.

Bit 7: Free-run Bit (FRT)

Selects whether timing is matched to the DPG counter and timer, or to free-running counter.

Bit 7

| FRT | Description |
|-----|--|
| 0 | 5-bit DFG counter + 16-bit timer counter (Initial value) |
| 1 | 16-bit FRC |

Bit 6: FRG2 Clear Stop Bit (FGR2OFF)

Nullifies the clearing of the counter by the DFG reference register 2. The FIFO group, including both FIFO1 and FIFO2, is available.

Bit 6

| FGR2OFF | Description |
|---------|--|
| 0 | Validates the clearing of the 16-bit timer counter by DFG reference register 2 (Initial value) |
| 1 | Nullifies the clearing of the 16-bit timer counter by DFG reference register 2 |

Bit 5: Mode Selection Bit (LOP)

Selects the output mode of FIFO. If the loop mode is selected, LOB3 to LOB0 bits and LOA3 to LOA0 bits become valid. If the LOP bit is rewritten, the pointer which counts the writing position of FIFO is cleared. In this case, the ultimate output date is kept.

Bit 5

| LOP | Description |
|-----|-----------------------------|
| 0 | Single mode (Initial value) |
| 1 | Loop mode |

Bit 4: DFG Edge Selection Bit (EDG)

Selects the edge by which to count DFG.

Bit 4

| EDG | Description |
|-----|--|
| 0 | Counts by the rising edge of DFG (Initial value) |
| 1 | Counts by the falling edge of DFG |

Bit 3: Interrupt Selection Bit (ISEL1)

Selects the factor which causes an interrupt. (IRRHSW1)

Bit 3

| ISEL1 | Description |
|-------|---|
| 0 | Generates an interrupt request by the rising edge of the STRIG signal of FIFO (Initial value) |
| 1 | Generates an interrupt request by the matching signal of FIFO |

Bit 2: FIFO Output Group Selection Bit (SOFG)

Selects whether 20 stages of FIFO1 + FIFO2 or only 10 stages of FIFO1 are used.

If 20-stage output mode is used in single mode, data write in FIFO1 and FIFO2 is required.

Monitor the output FIFO group flag (OFG) and control it by software. Output all the data of FIFO2 after all the data of FIFO1 was output. Repeat this step again. If 10-stage output mode is used, the data of FIFO2 is not reflected.

Rewriting the SOFG bit 0 → 1 → 0 initializes the control signal of the FIFO output stage to the FIFO1 side.

Bit 2

| SOFG | Description |
|------|--|
| 0 | 20-stage output of FIFO1 + FIFO2 (Initial value) |
| 1 | 10-stage output of FIFO1 only |

Bit 1: Output FIFO Group Flag (OFG)

Indicates the FIFO group which is outputting.

Bit 1

| OFG | Description |
|-----|--|
| 0 | Pattern is being output by FIFO1 (Initial value) |
| 1 | Pattern is being output by FIFO2 |

Bit 0: Output Switching Bit Between VideoFF and NarrowFF (VFF/NFF)

Switches the signal output to the VideoFF pin.

Bit 0

| VFF/NFF | Description |
|---------|--------------------------------|
| 0 | VideoFF output (Initial value) |
| 1 | NarrowFF output |

(3) HSW Loop Stage Number Setting Register (HSLP)

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | LOB3 | LOB2 | LOB1 | LOB0 | LOA3 | LOA2 | LOA1 | LOA0 |
| Initial value : | * | * | * | * | * | * | * | * |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Note: * Undetermined

HSLP sets the number of the loop stages when the HSW timing generator is in loop mode. It is valid if bit 5 (LOP) of HSM2 is 1. Bits 7 to 4 set the number of FIFO2 stages. Bits 3 to 0 set the number of FIFO1 stages.

HSLP is an 8-bit read/write register. It is not initialized by a reset, stand-by or module stop, accordingly be sure to set the number of the stages when the loop mode is used.

Bits 7 to 4: FIFO2 Stage Number Setting Bits (LOB3 to LOB0)

Set the number of FIFO2's stages in loop mode. They are valid only if the loop mode is set (LOP bit of HSM2 is 1).

| HSM2 | | HSLP | | | Description | |
|-------|-------|-------|-------|-------|--|---------------------------------------|
| Bit 5 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | | |
| LOP | LOB3 | LOB2 | LOB1 | LOB0 | | |
| 0 | * | * | * | * | Single mode (Initial value) | |
| 1 | 0 | 0 | 0 | 0 | Only 0th stage of FIFO2 is output | |
| | | | | 1 | 0th and 1st stages of FIFO2 are output | |
| | | | | 1 | 0 | 0th to 2nd stages of FIFO2 are output |
| | | | | 1 | 0th to 3rd stages of FIFO2 are output | |
| | | 1 | 0 | 0 | 0th to 4th stages of FIFO2 are output | |
| | | | | 1 | 0th to 5th stages of FIFO2 are output | |
| | | | | 1 | 0 | 0th to 6th stages of FIFO2 are output |
| | | | | 1 | 0th to 7th stages of FIFO2 are output | |
| | 1 | 0 | 0 | 0 | 0th to 8th stages of FIFO2 are output | |
| | | | | 1 | 0th to 9th stages of FIFO2 are output | |
| | | | | 1 | 0 | Setting prohibited |
| | | | | 1 | 1 | |
| | | 1 | 0 | 0 | | |
| | | | | 1 | | |
| | | | | 1 | 0 | |
| | | | | 1 | 1 | |

Note: * Don't care.

Bits 3 to 0: FIFO1 Stage Number Setting Bits (LOA3 to LOA0)

Set the number of FIFO1's stages in loop mode. They are valid only if the loop mode is set (LOP bit of HSM2 is 1).

| HSM2 | HSLP | | | | |
|-------|-------|-------|-------|-------|--|
| Bit 5 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| LOP | LOA3 | LOA2 | LOA1 | LOA0 | Description |
| 0 | * | * | * | * | Single mode (Initial value) |
| 1 | 0 | 0 | 0 | 0 | Only 0th stage of FIFO1 is output |
| | | | | 1 | 0th and 1st stages of FIFO1 are output |
| | | | 1 | 0 | 0th to 2nd stages of FIFO1 are output |
| | | | | 1 | 0th to 3rd stages of FIFO1 are output |
| | | 1 | 0 | 0 | 0th to 4th stages of FIFO1 are output |
| | | | | 1 | 0th to 5th stages of FIFO1 are output |
| | | | 1 | 0 | 0th to 6th stages of FIFO1 are output |
| | | | | 1 | 0th to 7th stages of FIFO1 are output |
| | 1 | 0 | 0 | 0 | 0th to 8th stages of FIFO1 are output |
| | | | | 1 | 0th to 9th stages of FIFO1 are output |
| | | | 1 | 0 | Setting prohibited |
| | | | | 1 | |
| | | 1 | 0 | 0 | |
| | | | | 1 | |
| | | | 1 | 0 | |
| | | | | 1 | |
| | | | | 1 | |
| | | | | 1 | |

Note: * Don't care.

(4) FIFO Output Pattern Register 1 (FPDRA)

| | | | | | | | | |
|-----------------|----|--------|--------|-----------|------|------|---------|---------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | ADTRGA | STRIGA | NarrowFFA | VFFA | AFFA | VpulseA | MlevelA |
| Initial value : | 1 | * | * | * | * | * | * | * |
| R/W : | — | W | W | W | W | W | W | W |

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PPGA7 | PPGA6 | PPGA5 | PPGA4 | PPGA3 | PPGA2 | PPGA1 | PPGA0 |
| Initial value : | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W |

Note: * Undetermined

FPDRA is a buffer register for the output pattern register of FIFO1. When the timing pattern is written in FTPRA the output pattern data written in FPDRA is written at the same time to the position pointed by the buffer pointer of FIFO1. Be sure to write the output pattern data in FPDRA before writing it in FTPRA.

FPDRA is a 16-bit write-only register. Only a word access is valid. If a byte access is attempted, resulting operation is not assured. No read is valid. If a read is attempted, an undetermined value is read out. It is not initialized by a reset, stand-by or module stop, accordingly be sure to write data before use.

Bit 15: Reserved

It cannot be written in or read out.

Bit 14: A/D Trigger A Bit (ADTRGA)

A signal for starting the A/D converter hardware.

Bit 13: S-TRIGA Bit (STRIGA)

A signal for generating an interrupt by pattern data. When STRIGA is selected by ISEL, pattern data changes from 0 to 1, and thus generates an interrupt.

Bit 12: NarrowFFA Bit (NarrowFFA)

Controls the Narrow Video Head.

Bit 11: VideoFFA Bit (VFFA)

Controls the Video Head.

Bit 10: AudioFFA Bit (AFFA)

Controls the Audio Head.

Bit 9: VpulseA Bit (VpulseA)

Used for generating an additional V signal. See section 28.12, Additional V Signal Generator, for more information.

Bit 8: MlevelA Bit (MlevelA)

Used for generating an additional V signal. See section 28.12, Additional V Signal Generator, for more information.

Bits 7 to 0: PPG Output Signal A Bits (PPGA7 to PPGA0)

Used for timing control output of port 7 (PPG).

(5) FIFO Output Pattern Register 2 (FPDRB)

| | | | | | | | | |
|-----------------|----|--------|--------|-----------|------|------|---------|---------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | ADTRGB | STRIGB | NarrowFFB | VFFB | AFFB | VpulseB | MlevelB |
| Initial value : | 1 | * | * | * | * | * | * | * |
| R/W : | — | W | W | W | W | W | W | W |

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PPGB7 | PPGB6 | PPGB5 | PPGB4 | PPGB3 | PPGB2 | PPGB1 | PPGB0 |
| Initial value : | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W |

Note: * Undetermined

FPDRB is a buffer register for the output pattern register of FIFO2. When the timing pattern is written in FTPRB the output pattern data written in FPDRB is written at the same time to the position pointed by the buffer pointer of FIFO2. Be sure to write the output pattern data in FPDRB before writing it in FTPRB.

FPDRB is a 16-bit write-only register. Only a word access is valid. If a byte access is attempted, resulting operation is not assured. No read is valid. If a read is attempted, an undetermined value is read out. It is not initialized by a reset, stand-by or module stop, accordingly be sure to write data before use.

Bit 15: Reserved

It cannot be written in or read out.

Bit 14: A/D Trigger B Bit (ADTRGB)

A signal for starting the A/D converter hardware.

Bit 13: S-TRIGB Bit (STRIGB)

A signal for generating an interrupt by pattern data. When STRIGB is selected by ISEL, pattern data changes from 0 to 1, and thus generates an interrupt.

Bit 12: NarrowFFB Bit (NarrowFFB)

Controls the Narrow Video Head.

Bit 11: VideoFFB Bit (VFFB)

Controls the Video Head.

Bit 10: AudioFFB Bit (AFFB)

Controls the Audio Head.

Bit 9: VpulseB Bit (VpulseB)

Used for generating an additional V signal. See section 28.12, Additional V Signal Generator, for more information.

Bit 8: MlevelB Bit (MlevelB)

Used for generating an additional V signal. See section 28.12, Additional V Signal Generator, for more information.

Bits 7 to 0: PPG Output Signal B Bits (PPGB7 to PPGB0)

Used for timing control output of port 7 (PPG).

(6) FIFO Timing Pattern Register 1 (FTPRA)

| | | | | | | | | | | | | | | | | |
|-----------------|---------|---------|---------|---------|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FTPRA15 | FTPRA14 | FTPRA13 | FTPRA12 | FTPRA11 | FTPRA10 | FTPRA9 | FTPRA8 | FTPRA7 | FTPRA6 | FTPRA5 | FTPRA4 | FTPRA3 | FTPRA2 | FTPRA1 | FTPRA0 |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

Note: * Undetermined

FTPRA is a register to write the timing pattern data of FIFO1. The timing data written in FTPRA is written at the same time to the position pointed by the buffer pointer of FIFO1 together with the buffer data of FPDRA.

FTPRA is a 16-bit write-only register. Only a word access is valid. If a byte access is attempted, resulting operation is not assured. It is not initialized by a reset, stand-by or module stop, accordingly be sure to write data before use.

Note: Its address is shared with the FIFO timer capture register (FTCTR). Accordingly, the value of FTCTR is read out if a read is attempted.

(7) FIFO Timing Pattern Register 2 (FTPRB)

| | | | | | | | | | | | | | | | | |
|-----------------|---------|---------|---------|---------|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FTPRB15 | FTPRB14 | FTPRB13 | FTPRB12 | FTPRB11 | FTPRB10 | FTPRB9 | FTPRB8 | FTPRB7 | FTPRB6 | FTPRB5 | FTPRB4 | FTPRB3 | FTPRB2 | FTPRB1 | FTPRB0 |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

Note: * Undetermined

FTPRB is a register to write the timing pattern data of FIFO2. The timing data written in FTPRB is written at the same time to the position pointed by the buffer pointer of FIFO2 together with the buffer data of FPDRB.

FTPRB is a 16-bit write-only register. Only a word access is valid. If a byte access is attempted, resulting operation is not assured. It is not initialized by a reset, stand-by or module stop, accordingly be sure to write data before use.

(8) DFG Reference Register 1 (DFCRA)

| | | | | | | | | |
|-----------------|-------|------|------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ISEL2 | CCLR | CKSL | DFCRA4 | DFCRA3 | DFCRA2 | DFCRA1 | DFCRA0 |
| Initial value : | 0 | 0 | 0 | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W |

Note: * Undetermined

DFCRA is a register which determines the operation of the HSW timing generator as well as the starting point of the timing of FIFO1.

DFCRA is an 8-bit write-only register. It is not initialized by a reset, stand-by or module stop, accordingly be sure to write data before use.

Note: Its address is shared with the DFG reference counter register (DFCTR). Accordingly, the value of DFCTR is read out in the low-order five bits if a read is attempted.

Bit 7: Interrupt Selection Bit (ISEL2)

Selects the factor which causes an interrupt. (IRRHSW2)

Bit 7

| ISEL2 | Description |
|-------|---|
| 0 | Generates an interrupt request by the clear signal of the 16-bit timer counter (Initial value) |
| 1 | Generates an interrupt request by the VD signal in PB mode |

Bit 6: DFG Counter Clear Bit (CCLR)

Enforces clearing of the 5-bit counter which counts DFG by software. Writing 1 returns 0 immediately. Writing 0 causes no effect on operation.

Bit 6

| CCLR | Description |
|------|----------------------------------|
| 0 | Normal operation (Initial value) |
| 1 | Clears the 5-bit DFG counter |

Bit 5: 16-bit Timer Counter Clock Source Selection Bit (CKSL)

Selects the clock source of the 16-bit timer counter.

Bit 5

| CKSL | Description |
|------|----------------------------|
| 0 | $\phi_s/4$ (Initial value) |
| 1 | $\phi_s/8$ |

Bits 4 to 0: FIFO1 Output Timing Setting Bits (DFCRA4 to DFCRA0)

Determine the starting point of the timing of FIFO1. The initial value is undetermined. Be sure to set a value after a reset or stand-by. It is valid only if bit 7 (FRT bit) of HSM2 is 0.

(9) DFG Reference Register 2 (DFCRB)

| | | | | | | | | |
|-----------------|---|---|---|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | DFCRB4 | DFCRB3 | DFCRB2 | DFCRB1 | DFCRB0 |
| Initial value : | 1 | 1 | 1 | * | * | * | * | * |
| R/W : | — | — | — | W | W | W | W | W |

Note: * Undetermined

DFCRB is a register which determines the starting point of the timing of FIFO2.

DFCRB is an 8-bit write-only register. If a read is attempted, an undetermined value is read out. Bits 7 to 5 are reserved. No write is valid. If a read is attempted, 1 is read out. It is not initialized by a reset or stand-by, accordingly be sure to write data before use.

Bits 4 to 0: FIFO2 Output Timing Setting Bits (DFCRB4 to DFCRB0)

Determine the starting of the FIFO2 output. The initial values are undetermined, accordingly be sure to write values in the bits after a reset or stand-by.

It is valid only if bit 7 (FRT bit) of HSM2 is 0.

(10) FIFO Timer Capture Register (FTCTR)

| | | | | | | | | | | | | | | | | |
|-----------------|---------|---------|---------|---------|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FTCTR15 | FTCTR14 | FTCTR13 | FTCTR12 | FTCTR11 | FTCTR10 | FTCTR9 | FTCTR8 | FTCTR7 | FTCTR6 | FTCTR5 | FTCTR4 | FTCTR3 | FTCTR2 | FTCTR1 | FTCTR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

FTCTR is a register to display the count of the 16-bit timer counter.

FTCTR is a 16-bit read-only register. It stores the counter value if a VD signal was detected in PB mode. Only a word access is accepted. If a byte access is attempted, resulting operation is not assured. It is initialized to H'0000 by a reset or stand-by.

Note: Its address is shared with the FIFO timing pattern register 1 (FTPRA). Accordingly, if a write is attempted, the value is written in FTPRA.

(11) DFG Reference Count Register (DFCTR)

| | | | | | | | | |
|-----------------|---|---|---|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | DFCTR4 | DFCTR3 | DFCTR2 | DFCTR1 | DFCTR0 |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | R | R | R | R | R |

DFCTR is a register to count the DFG pulses.

DFCTR is an 8-bit read-only register. Bits 7 to 5 are reserved. If a read is attempted, 1 is read out. It is initialized to H'E0 by a reset or stand-by.

Note: Its address is shared with the DFG reference register 1 (DFCRA). Accordingly, if a write is attempted, the value is written in DFCRA.

Bits 4 to 0: DFG Pulse Count Bits (DFCTR4 to DFCTR0)

Count the number of pulses of DFG.

28.4.6 Description of Operation

(a) 5-bit DFG counter

The 5-bit DFG counter takes counts by the edges of DFG selected by the EDG bit of HSW mode register 2. The 5-bit DFG counter is cleared by the DPG's rise or when 1 was written in the CCLR bit of DFG reference register 1.

(b) 16-bit Timer Counter

DFG reference mode or free-run mode can be selected for the 16-bit timer counter.

- DFG Reference Mode

DFG reference mode is based on the DFG signal. When the DFG reference registers 1 and 2 and the 5-bit DFG counter value match, the 16-bit timer counter is initialized, and that point becomes the starting of the FIFO output.

In DFG reference mode, the FGR2OFF bit of the HSW mode register 2 can be used to select between using only the DFG reference register 1 to set the starting of the FIFO output or using both DFG reference registers 1 and 2 to set the starting of the FIFO1 and FIFO2 outputs, respectively. When using only the DFG reference register 1 to set the starting, continuous values should be set as the timing patterns for FIFO1 and FIFO2.

- Free-run Mode

Free-run mode is to operate together with the prescaler unit. An overflow of the 18-bit free-running counter in the prescaler unit initializes the 16-bit timer counter, and that point becomes the starting of the FIFO output.

(c) Matching Circuit

The matching circuit compares the timing pattern value of FIFO with the 16-bit timer counter value, and if they match, it generates a trigger signal to output the pattern data for the FIFO's next stage.

(d) FIFO

FIFO generates the head-switching signal used in the VCR and the pattern data necessary for servo control. Data is set in FIFO by the FIFO timing pattern registers 1 and 2 and the FIFO output pattern registers 1 and 2.

FIFO has two modes, i.e. single mode and loop mode. In either mode, output of 20 stages of FIFO1 + FIFO2 or output of only 10 stages of FIFO1 can be selected.

- Single Mode

In single mode, the output pattern data is output each time the timing data matches. The data, once output, is lost, and the internal pointer is decremented by 1. When the last data was output, it stops operation until data is written again. When it is used in the 20-stage output mode, writing in FIFO1 and FIFO2 has to be controlled by software.

- Loop Mode

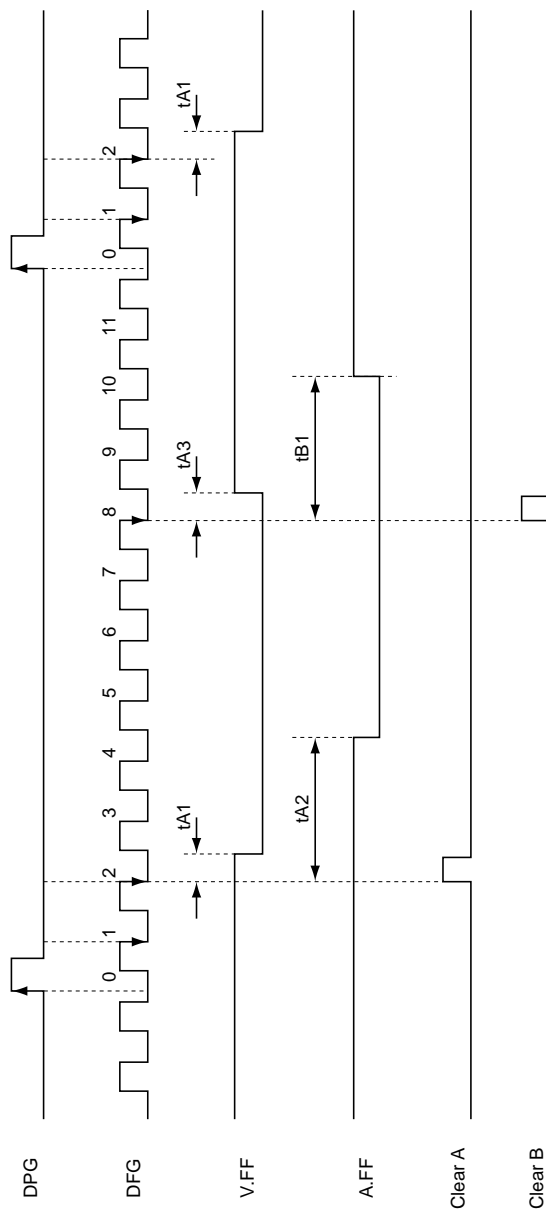
In loop mode, the output pattern cycles repeatedly from stage 0 through the final stage selected in the HSW loop number setting register. As in single mode, the output pattern data is output each time the timing data matches. In loop mode, the FIFO data is retained.

When loop mode is active, data can be rewritten for each FIFO group.

After confirming with the OFG bit of the HSW mode register 2 which FIFO group is outputting, clear the FIFO group which is not outputting and write data for the entire FIFO group. Writing has to be completed before the rewritten FIFO group starts operation.

Partial rewriting in the FIFO is not possible, because the write pointer is outside the loop stages.

Figures 28.23 and 28.24 show examples of the timing waveform and its operation of the HSW timing generator.



Example of setting: DFCRA=H'02, DFCRB=H'08, HSLP=H'21, DFG falling edge

Figure 28.23 Example of Timing Waveform of HSW (when DFG is 12 Shots)

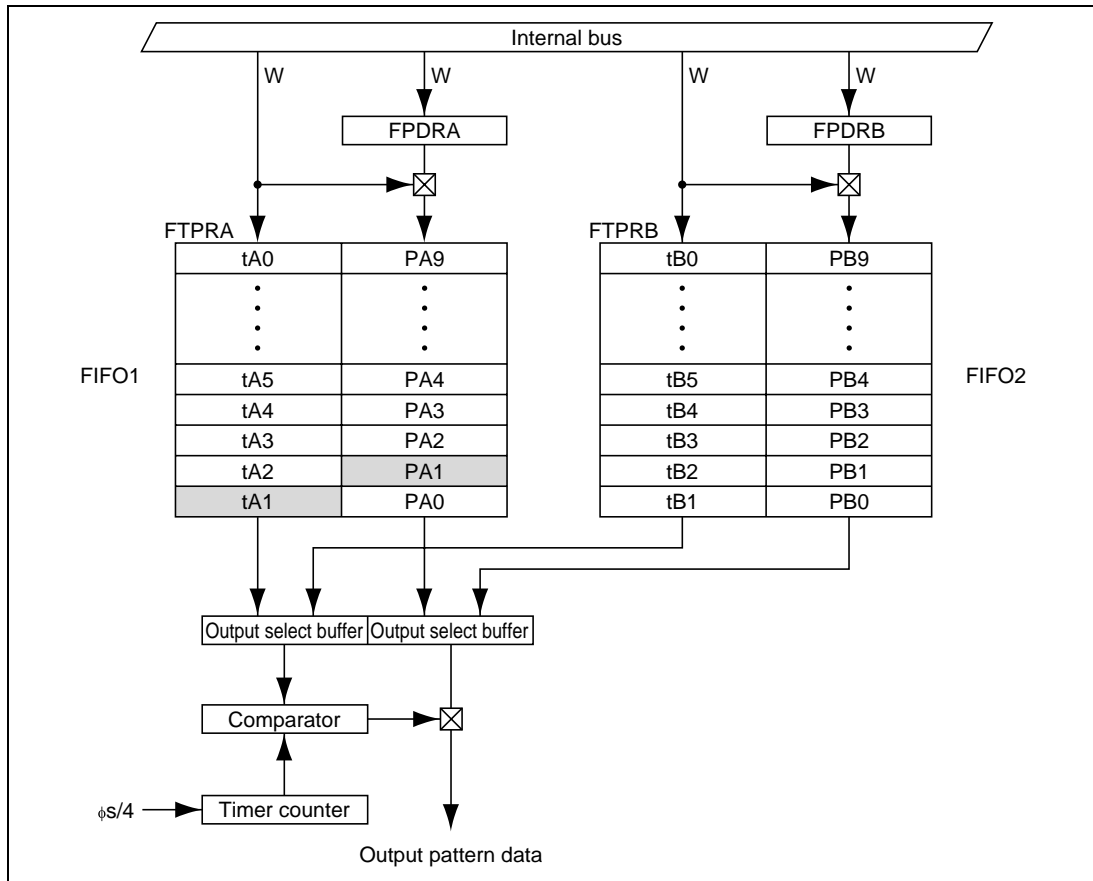


Figure 28.24 Example of Operation of the HSW Timing Generator

- (1) Example of operation in single mode (20 stages of FIFO used)
- (a) Set to single mode ($LOP = 0$)
 - (b) Write the output pattern data (PA0) to FPDRA.
 - (c) Write the output timing (t_{A1}) to FTPRA. t_{A1} is written in FIFO1 together with PA0. This initializes the output pattern data to PA0.
 - (d) Repeat the steps in the same way, until PA1, PA2, etc., are set.
 - (e) Write the output pattern data (PB0) to FPDRB.
 - (f) Write the output timing (t_{B1}) to FTPRB. t_{B1} is written in FIFO2 together with PB0. This initializes the output pattern data to PB0.
 - (g) Repeat these steps in the same way, until PB1, PB2, etc., are set.

From (c), the pattern data of PA0 is output.

If t_{A1} matches with the timer counter, the pattern data of PA1 is output.

If t_{A2} matches with the timer counter, the pattern data of PA2 is output.

.
.
.

After this sequence is repeated and all the pattern data set in FIFO1 is output, the pattern data of FIFO2 is output. After the pattern data is output, the pointer is decremented by 1. Care is required, however, because matching of t_{A0} is not detected until data is written in FIFO2. Matching of t_{B0} also is not detected until data is written in FIFO1 again.

(2) Example of operation in loop mode

- (a) Set the number of loop stages in the HSLP register (e.g. $HSLP = H'44$)
- (b) Write the output pattern data (PA0) to FPDRA.
- (c) Write the output timing (t_{A1}) to FTPRA. t_{A1} is written in FIFO1 together with PA0. This initializes the output pattern data to PA0.
- (d) Repeat the steps in the same way, until PA1, PA2, etc., are set.
- (e) Write the output pattern data (PB0) to FPDRE.
- (f) Write the output timing (t_{B1}) to FTRE. t_{B1} is written in FIFO2 together with PB0. This initializes the output pattern data to PB0.
- (g) Repeat the steps in the same way, until PB1, PB2, etc., are set.

From (c), the pattern data PA0 is output.

If t_{A1} matches the timer counter, the pattern data PA1 is output.

If t_{A2} matches the timer counter, the pattern data PA2 is output.

.

.

.

If t_{A4} matches the timer counter, the pattern data PA4 is output.

If t_{A5} matches the timer counter, the pattern data PB0 is output.

If t_{B1} matches the timer counter, the pattern data PB1 is output.

.

.

.

If t_{B4} matches the timer counter, the pattern data PB4 is output.

If t_{B5} matches the timer counter, the pattern data PA0 is output.

.

.

.

28.4.7 Interrupt

The HSW timing generator generates an interrupt under the following conditions.

- (1) IRRHSW1 occurred when pattern data was written (OVWA, OVWB = 1) and FIFO was full (FULL).
 - (2) IRRHSW1 occurred when matching was detected and the STRIG bit of FIFO was 1.
 - (3) IRRHSW1 occurred when the values of the 16-bit timer counter and 16-bit timing pattern register matched.
 - (4) IRRHSW2 occurred when the 16-bit timer counter was cleared.
 - (5) IRRHSW2 occurred when a VD signal (capture signal of the timer capture register) was received in PB mode.
- (2) and (3), as well as (4) and (5), are switched over by ISEL1 and ISEL2.

28.4.8 Cautions

- (1) When both the 5-bit DFG counter and 16-bit timer counter are operating, the latter is not cleared if input of DPG and DFG signals is stopped. This leads to free-running of the 16-bit timer counter, and periodical detection of matching by the 16-bit timer counter. In such a case, the period of the output from the HSW timing generator is independent from DPG or DFG.
- (2) Specify the mode setting bit (LOP) of the HSW mode register 2 (HSM2) immediately before writing the FIFO data.
- (3) Input the rising edge of DPG and DFG count edge at different timings. If the same timing was input, counting up of DFG and clearing of the 5-bit DFG counter occurs simultaneously. In this case, the latter will take precedence. This leads to the 5-bit DFG counter's lag by 1. Figure 28.25 shows the input timing of DPG and DFG.
- (4) If stop of the drum system is required when FIFO output is being used in the 20-stage output mode, rewrite the SOFG bit of HSM2 register 0 $\rightarrow 1 \rightarrow 0$ by software, and initialize the FIFO output stage to the FIFO1 side without fail. Also clear and rewrite the data of FIFO1 and FIFO2.

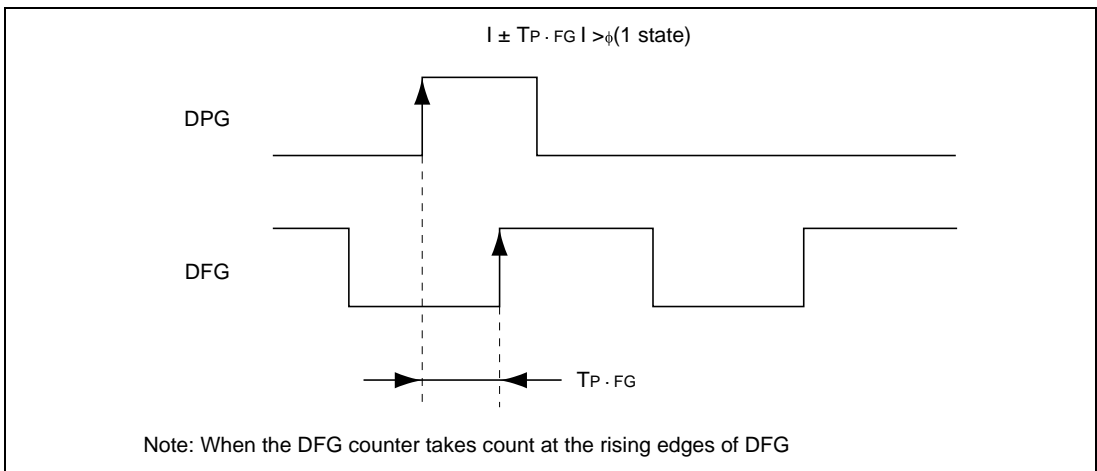


Figure 28.25 Input Timing of DPG and DFG

28.5 Four-head High-speed Switching Circuit for Special Playback

28.5.1 Overview

This four-head high-speed switching circuit generates a color rotary signal (C.Rotary) and head-amplifier switching signal (H.Amp SW) for use in four-head special playback. A pre-amplifier output comparison result signal is input from the COMP pin. The signal output at the C.Rotary pin is a Chroma signal processing control signal. The signal output at the H.Amp SW pin is a pre-amplifier output select signal. To reduce the width of noise bars, the C.Rotary and H.Amp SW signals are synchronized to the horizontal sync signal (OSCH). OSCH is made by adding supplemented H, which has been separated from the Csync signal in the sync signal detector circuit. For more details of OSCH, see section 28.15, Sync Signal Detector. If the C.Rotary, H.Amp SW or COMP pin does not require this circuit to configure a VCR system, it can be used as an I/O port.

28.5.2 Block Diagram

Figure 28.26 shows the block diagram of this circuit.

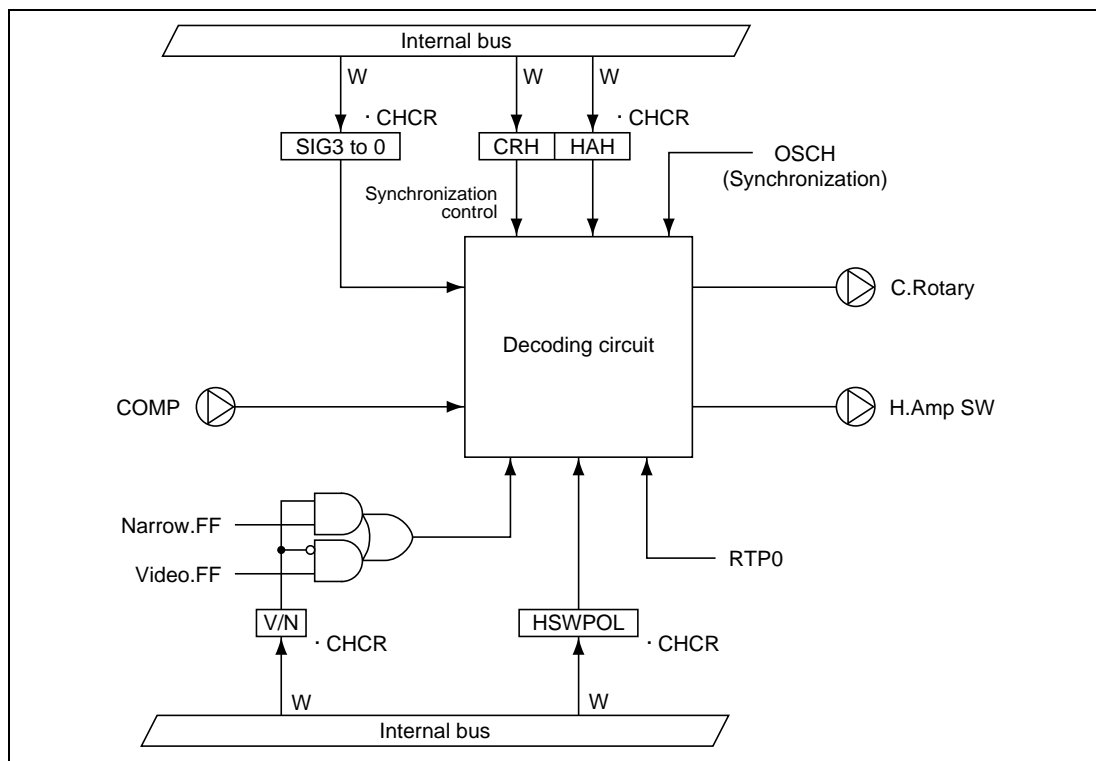


Figure 28.26 Four-Head High-Speed Switching Circuit for Special Playback

28.5.3 Pin Configuration

Table 28.7 summarizes the pin configuration of the high-speed switching circuit used in four-head special playback. They can also be used as I/O ports when not in use. See section 28.2, Servo Port.

Table 28.7 Pin Configuration

| Name | Abbrev. | I/O | Function |
|--------------------------------|----------|--------|--|
| Compare input pin | COMP | Input | Input of pre-amplifier output result signal |
| Color rotary signal output pin | C.Rotary | Output | Output of chroma processing control signal |
| Head-amplifier switching pin | H.Amp SW | Output | Output of pre-amplifier output select signal |

28.5.4 Register Description

(1) Register Configuration

Table 28.8 shows the register configuration of the high-speed switching circuit used in four-head special playback.

Table 28.8 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|-----------------------------------|---------|-----|------|---------------|---------|
| Special playback control register | CHCR | W | Byte | H'00 | H'FD06E |

(2) Special Playback Control Register (CHCR)

| | | | | | | | | |
|-----------------|-----|--------|-----|-----|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | V/N | HSWPOL | CRH | HAH | SIG3 | SIG2 | SIG1 | SIG0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

The special playback control register (CHCR) is an 8-bit write-only register. It cannot be read. If a read is attempted, an undetermined value is read out. It is initialized to H'00 by a reset, stand-by or module stop.

Bit 7: HSW Signal Selection Bit (V/N)

Selects the HSW signal to be used at special playback.

Bit 7

| V/N | Description |
|-----|--|
| 0 | Video FF signal output (Initial value) |
| 1 | Narrow FF signal output |

Bit 6: COMP Polarity Selection Bit (HSWPOL)

Selects the polarity of the COMP signal.

Bit 6

| HSWPOL | Description |
|--------|--------------------------|
| 0 | Positive (Initial value) |
| 1 | Negative |

Bit 5: C.Rotary Synchronization Control Bit (CRH)

Synchronizes the C.Rotary signal with the OSCH signal.

Bit 5

| CRH | Description |
|-----|-----------------------------|
| 0 | Synchronous (Initial value) |
| 1 | Asynchronous |

Bit 4: H.Amp SW Synchronization Control Bit (HAH)

Synchronizes the H.Amp SW signal with the OSCH signal.

Bit 4

| HAH | Description |
|-----|-----------------------------|
| 0 | Synchronous (Initial value) |
| 1 | Asynchronous |

Bits 3 to 0: Signal Control Bits (SIG3 to SIG0)

These bits, combined with the state of the COMP input pin, control the outputs at the C.Rotary and H.Amp SW pins.

| Bit 3 | Bit 2 | Bit 1 | Bit 0 | Output Pins | |
|-------|-------|-------|-------|-------------------------|-------------------------|
| SIG3 | SIG2 | SIG1 | SIG0 | C.Rotary | H.Amp SW |
| 0 | 0 | * | * | L | L (Initial value) |
| | | 0 | 0 | HSW | L |
| | | | 1 | $\overline{\text{HSW}}$ | H |
| | | 1 | 0 | L | HSW |
| | | | 1 | H | $\overline{\text{HSW}}$ |
| 1 | 0 | 0 | * | HSW EX-OR COMP | COMP |
| | | 1 | | HSW EX-NOR COMP | COMP |
| | | 0 | | HSW E-OR RTP0 | RTP0 |
| | | 1 | | HSW EX-NOR RTP0 | RTP0 |

Note: * Don't care.

28.6 Drum Speed Error Detector

28.6.1 Overview

Drum speed error control operates so as to hold the drum at a constant revolution speed by measuring the period of the DFG signal. A digital counter detects the speed deviation from a preset value. The speed error data is processed and added to phase error data in a digital filter. This filter controls a pulse-width modulated (PWM) output, which controls the revolution speed and phase of the drum.

The DFG input signal is reshaped into a square wave by a reshaping circuit, and sent to the speed error detector as the DFG signal.

The speed error detector uses the system clock to measure the period of the DFG signal, and detects the deviation from a preset data value. The preset data is the value that would result from measuring the DFG signal period with the clock signal if the drum motor was running at the correct speed.

The error detector operates by latching a counter value when it detects an edge of the DFG signal. The latched count provides 16 bits of speed error data for the digital filter to operate on. The digital filter processes and adds the speed error data to phase error data from the drum phase control system, then sends the result to the pulse-width modulator as drum error data.

28.6.2 Block Diagram

Figure 28.27 shows a block diagram of the drum speed error detector.

28.6.3 Register Configuration

Table 28.9 shows the register configuration of the drum speed error detector.

Table 28.9 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|---|----------------|------------|-------------|----------------------|----------------|
| Specified DFG speed preset data register | DFPR | W | Word | H'0000 | H'FD030 |
| DFG speed error data register | DFER | R/W | Word | H'0000 | H'FD032 |
| DFG lock UPPER data register | DFRUDR | W | Word | H'7FFF | H'FD034 |
| DFG lock LOWER data register | DFRLDR | W | Word | H'8000 | H'FD036 |
| Drum speed error detection control register | DFVCR | R/W | Byte | H'00 | H'FD038 |

28.6.4 Register Descriptions

(1) Specified DFG Speed Preset Data Register (DFPR)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

The specified DFG speed preset data is set in DFPR. When the data is written, a 16-bit preset data is sent to the preset circuit. The preset data is referenced to H'8000*, and can be calculated from the following equation.

$$\text{Specified DFG speed preset data} = \text{H}'8000 - \left(\frac{\phi s/n}{\text{DFG frequency}} - 2 \right)$$

ϕ s: Servo clock frequency (fosc/2) in Hz

DFG frequency: In Hz

The constant 2 is the presetting interval (see Figure 28.28).

ϕ s/n Clock source of selected counter

DFPR is a 16-bit write-only register, and is accessible by word access only. Byte access gives unassured results. Reads are disabled. DFPR is initialized to H'0000 by a reset, and in standby mode and module stop mode.

Note: * The preset data value is calculated so that the counter will reach H'8000 when the error is zero. When the counter value is latched as error data in the DFG speed error data register (DFER), however, it is converted to a value referenced to H'0000.

(2) DFG Speed Error Data Register (DFER)

| | | | | | | | | | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

Note: * Note that only detected error data can be read.

DFER is a register that stores 16-bit DFG speed error data. When the drum motor speed is correct, the data latched in DFER is H'0000. Negative data will be latched if the speed is too fast, and positive data if the speed is too slow. The DFER value is sent to the digital filter either automatically or by software.

DFER is a 16-bit readable/writable register. DFER is accessible by word access only. Byte access gives unassured results. DFER is initialized to H'0000 by a reset, and in standby mode and module stop mode.

Refer to the Note in 28.6.4 (1) Specified DFG Speed Preset Data Register (DFPR).

(3) DFG Lock UPPER Data Register (DFRUDR)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

DFRUDR is a register used to set the lock range on the UPPER side when drum speed lock is detected, and to set the limit value on the UPPER side when the limiter function is in use. Set a signed data to DFRUDR (bit 15 is a sign-setting bit).

When lock is being detected, if the drum speed is detected within the lock range, the lock counter which has been set by DFRCS 1 and 0 bits of the DFVCR register counts down. If the set value of DFRCS 1 and 0 matches the number of times of occurrence of locking, the computation of the digital filter in the drum phase system can be controlled automatically. Also, if the DFG speed error data is beyond the DFRUDR value while the limiter function is in use, the DFRUDR value can be used as the data for computation by the digital filter.

DFRUDR is a 16-bit write-only register. Only a word access is valid. If a byte access is attempted, operation is not assured. No read is valid. If a read is attempted, an undetermined value is read out. It is initialized to H'7FFF by a reset, stand-by or module-stop.

(4) DFG Lock LOWER Data Register (DFRLDR)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

DFRLDR is a register used to set the lock range on the LOWER side when drum speed lock is detected, and to set the limit value on LOWER side when the limiter function is in use. Set a signed data to DFRLDR (bit 15 is a sign-setting bit).

When lock is being detected, if the drum speed is detected within the lock range, the lock counter which has been set by the DFRCS1 and DFRCS0 bits of the DFVCR register counts down. If the set value of DFRCS1 and DFRCS0 matches the number of times of occurrence of locking, the computation of the digital filter in the drum phase system can be controlled automatically. Also, if the DFG speed error data is under the DFRLDR value when the limiter function is in use, the DFRLDR value can be used as the data for computation by the digital filter.

DFRLDR is a 16-bit write-only register. Only a word access is valid. If a byte access is attempted, operation is not assured. No read is valid. If a read is attempted, an undetermined value is read out. It is initialized to H'8000 by a reset, stand-by or module-stop.

(5) Drum Speed Error Detection Control Register (DFVCR)

| | | | | | | | | |
|-----------------|-------|-------|---------|--------|----------|-------|---------|---------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | DFCS1 | DFCS0 | DFOVF | DFRFON | DF-R/UNR | DPCNT | DFRCS1 | DFRCS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/(W)*1 | R/W | R | R/W | (R)*2/W | (R)*2/W |

Notes: 1. Only 0 can be written.

2. If read-accessed, the counter value is read out.

DFVCR controls the operation of drum speed error detection.

DFVCR is an 8-bit readable/writable register. Bit 3 accepts only read, and bit 5 accepts only read and 0 write. It is initialized to H'00 by a reset, stand-by or module-stop.

Bits 7 and 6: Clock Source Selection Bits (DFCS1, DFCS0)

DFCS1 and DFCS0 select the clock to be supplied to the counter. ($\phi s = fosc/2$)

| Bit 7 | | Bit 6 | |
|-------|-------|--------------------------|--|
| DFCS1 | DFCS0 | Description | |
| 0 | 0 | ϕs (Initial value) | |
| | 1 | $\phi s/2$ | |
| 1 | 0 | $\phi s/4$ | |
| | 1 | $\phi s/8$ | |

Bit 5: Counter Overflow Flag (DFOVF)

The DFOVF flag indicates the overflow of the 16-bit counter. It is cleared by writing 0. Write 0 after reading 1. Also, setting has the highest priority in this flag. If a flag set and 0 write occurs simultaneously, the latter is nullified.

Bit 5

| DFOVF | Description | |
|-------|---|--|
| 0 | Normal state (Initial value) | |
| 1 | Indicates that overflow has occurred in the counter | |

Bit 4: Error Data Limit Function Selection Bit (DFRFON)

Makes the error data limit function valid. (Limit values are the values set in the lock range data register (DFRUDR, DFRLDR)).

Bit 4

| DFRFON | Description | |
|--------|------------------------------------|--|
| 0 | Limit function off (Initial value) | |
| 1 | Limit function on | |

Bit 3: Drum Lock Flag (DF-R/UNR)

Sets a flag if an underflow occurred in the drum lock counter.

Bit 3

| DF-R/UNR | Description | |
|----------|--|--|
| 0 | Indicates that the drum speed system is not locked (Initial value) | |
| 1 | Indicates that the drum speed system is locked | |

Bit 2: Drum Phase System Filter Computation Automatic Start Bit (DPCNT)

Sets on the filter computation of the phase system if an underflow occurred in the drum lock counter.

Bit 2

| DPCNT | Description |
|--------------|---|
| 0 | Does not perform the filter computation by detection of the drum lock (Initial value) |
| 1 | Sets on the filter computation of the phase system when drum lock is detected |

Bits 1 and 0: Drum Lock Counter Setting Bits (DFRCS1, DFRCS0)

Set the number of times where drum lock has been determined (DFG has been detected in the range set by the lock range data register). It sets the drum lock flag if it detected the set number of times of occurrence of drum lock. If an NCDFG signal is detected outside the lock range after data is written in DFRCS1 and 0, data is stored in the lock counter.

Note: If DFRCS1 or DFRCS0 is read-accessed, the counter value is read out. If bit 3 (drum lock flag) is 1 and the drum lock counter's value is 3, it indicates that the drum speed system is locked. The drum lock counter stops until lock is released after underflow.

| Bit 1 | Bit 0 | Description |
|---------------|---------------|--|
| DFRCS1 | DFRCS0 | |
| 0 | 0 | Underflow after lock was detected once (Initial value) |
| | 1 | Underflow after lock was detected twice |
| 1 | 0 | Underflow after lock was detected three times |
| | 1 | Underflow after lock was detected four times |

28.6.5 Description of Operation

The drum speed error detector detects the speed error based on the reference value set in the DFG specified speed preset register (DFPR). The reference value set in DFPR is preset in the counter by the NCDFG signal, and counts down by the selected clock. The timing of the counter presetting and the error data latching can be selected between the rising or falling edge of the NCDFG signal. See section 28.14.4, DFG Noise Removal Circuit. The error data detected is sent to the digital filter circuit. The error data is signed binaries. It takes a positive number (+) if the speed is slower than the specified speed, a negative number (-) if the speed is faster, or 0 if it correct (revolving at the specified speed). Figure 28.28 shows an example of operation to detect the drum speed.

(a) Setting the error data limit

A limit can be set to the error data sent to the digital filter circuit using the DFG lock data register (DFRUDR, DFRLDR). Set the upper limit of the error data in DFRUDR and the lower limit in DFRLDR, and write 1 in the DFRFON bit. If the error data is beyond the limit range, the DFRLDR value is sent if a negative number is latched, or the DFRUDR value is sent if a positive one is latched, as a limit value. Be sure to turn off the limit setting (DFRFON = 0) when you set the limit value. If the limit was set with the limit setting on (DFRFON = 1), result of computation is not assured.

(b) Lock detection

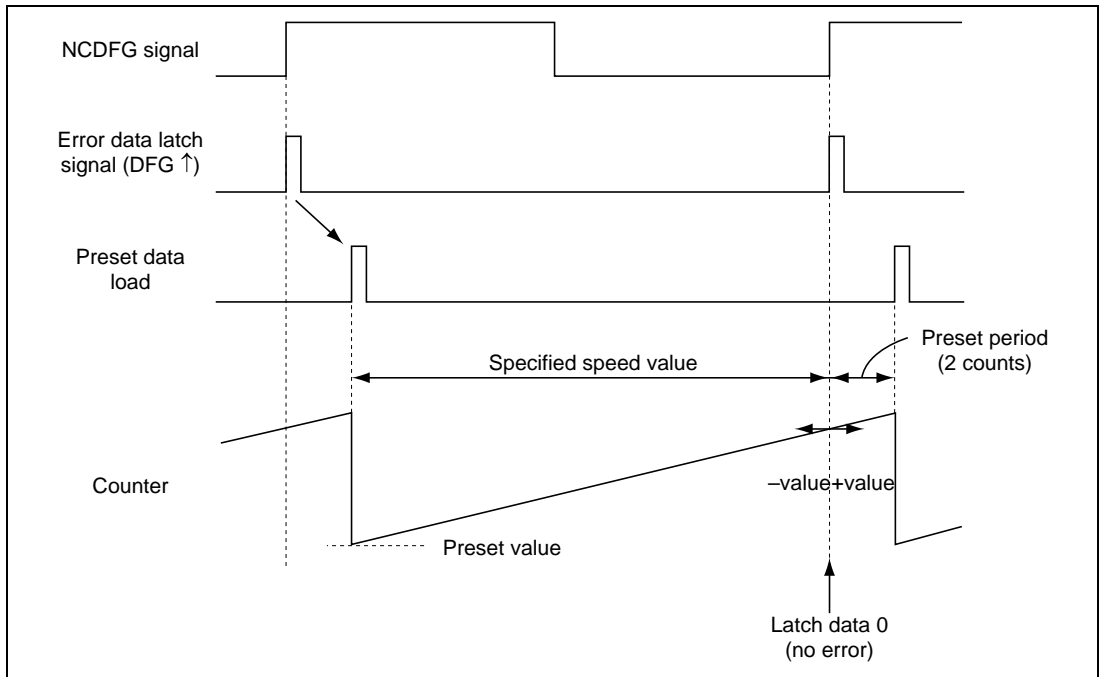
If an error data was detected within the lock range set in the lock data register, the drum lock flag (DF-R/UNR) is set by the number of the times of occurrence of locking set by the DFRCS1 and DFRCS0 bits, and an interrupt is requested (IRRDRM2) at the same time. The number of the occurrence of locking (once to 4 times) can be specified when setting the flag. Use the DFRCS1 and DFRCS0 bits for this purpose. Also, if bit 5 (DPHA bit) of the drum system digital filter control register (DFIC) is 0 (phased system digital filter computation off) and the DPCNT bit is 1, turning on/off of the phase system digital filter computation can be controlled automatically by the status of lock detection.

(c) Drum system speed error detection counter

The drum system speed error detection counter stops the counter and sets the overflow flag (DFOVF) when the overflow occurred. At the same time, it generates an interrupt request (IRRDRM1). Clear DFOVF by writing 0 after reading 1. If setting the flag and writing 0 take place simultaneously, the latter is nullified.

(d) Interrupt request

IRRDRM1 is generated by the NCDFG signal latch and the overflow of the error detection counter. IRRDRM2 is generated by detection of lock (after the detection of the number of times of setting).



**Figure 28.28 Example of the Operation of the Drum Speed Error Detection
(Selection of the Rising Edge of DFG)**

28.6.6 f_H Correction in Trick Play Mode

In trick play mode, the tape speed changes relative to the video head. This change alters the horizontal sync signal (f_H), causing skew. To correct the skew, the drum motor speed must be shifted to a different speed in each trick play mode, so as to obtain the normal horizontal sync frequency. To shift the drum motor speed, software should modify the value written in the DFG preset data register in the speed error detector.

This f_H correction can be expressed in terms of the basic frequency f_F of the drum as follows.

$$f_F = \frac{N_0}{N_0 + \alpha_H (1-n)} \times f_{F0}$$

Legend:

n : Speed multiplier (FWD = positive, REV = negative)

α_H : H alignment (1.5H in standard mode, 0.75H in 2x mode, and 0.5H in 3x mode for VHS and β systems; 1H for an 8-mm VCR)

N_0 : Standard H numbers within field

f_{F0} : Field frequency

NTSC: $N_0 = 262.5$, $f_{F0} = 59.94$

PAL: $N_0 = 312.5$, $f_{F0} = 50.00$

28.7 Drum Phase Error Detector

28.7.1 Overview

Drum phase control must start operating after the drum motor is brought to the correct revolution speed by the speed control system. Drum phase control works as follows in record and playback.

Record: Phase is controlled so that the vertical blanking intervals of the recorded video signal will line up along the bottom edge of the tape.

Playback: Phase is controlled so as to trace the recorded tracks accurately.

A digital counter detects the phase deviation from a preset value. The phase error data is processed and added to speed error data in a digital filter. This filter controls a pulse-width modulated (PWM) output, which controls the rotational phase and speed of the drum.

The DPG signal from the drum motor is reshaped into a rectangular pulse waveform by a reshaping circuit, and sent to the phase error detector.

The phase error detector compares the phase of the DPG pulse (tackle pulse), which contains video head phase information, with a reference signal. In the actual circuit, the comparison is carried out by comparing the head-switching (HSW) signal, which is delayed by a counter that is reset by DPG, with a reference signal value. The reference signal is the REF30 signal, which differs between record and playback as follows.

Record: Vsync signal extracted from the video signal to be recorded (frame rate signal, actually 1/2 Vsync)

Playback: 30 Hz or 25 Hz signal divided from the system clock

28.7.2 Block Diagram

Figure 28.29 shows a block diagram of the drum phase error detector.

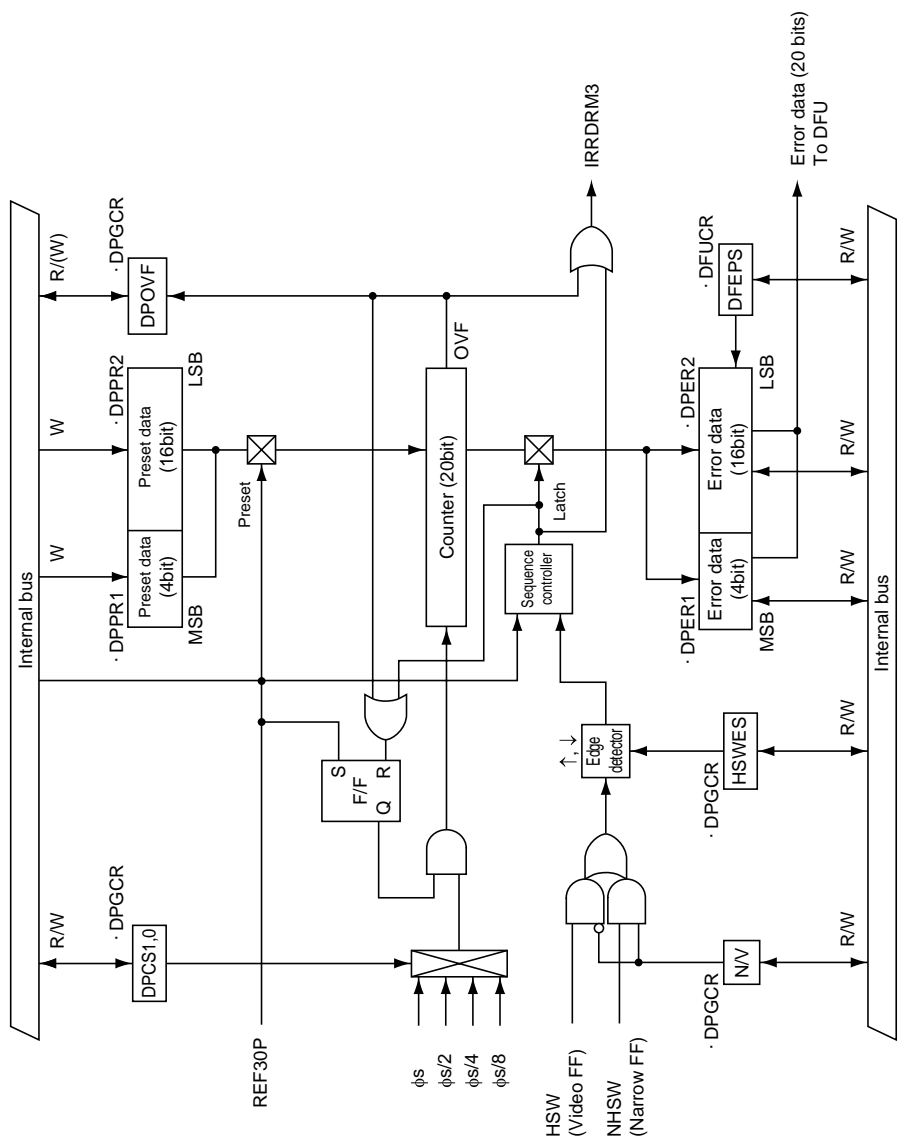


Figure 28.29 Block Diagram of Drum Phase Error Detector

28.7.3 Register Configuration

Table 28.10 shows the register configuration of the drum phase error detector.

Table 28.10 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|---|----------------|------------|-------------|----------------------|----------------|
| Drum phase preset data register 1 | DPPR1 | W | Byte | H'F0 | H'FD03C |
| Drum phase preset data register 2 | DPPR2 | W | Word | H'0000 | H'FD03A |
| Drum phase error data register 1 | DPER1 | R/W | Byte | H'F0 | H'FD03D |
| Drum phase error data register 2 | DPER2 | R/W | Word | H'0000 | H'FD03E |
| Drum phase error detection control register | DPGCR | R/W | Byte | H'07 | H'FD039 |

28.7.4 Register Descriptions

(1) Drum Phase Preset Data Registers (DPPR1, DPPR2)

DPPR1

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | | | | |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W |

DPPR2

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

The 20-bit preset data that defines the specified drum phase is set in DPPR1 and DPPR2. The 20 bits are weighted as follows. Bit 3 of DPPR1 is the MSB, and bit 0 of DPPR2 is the LSB. When data is written to DPPR2, the 20-bit preset data, including DPPR1, is loaded into the preset circuit. Write to DPPR1 first, and DPPR2 next. The preset data is referenced to H'80000*, and can be calculated from the following equation.

Target phase difference = (reference signal frequency/2) – 6.5H

Drum phase preset data = H'80000 - ($\phi_s/n \times$ target phase difference)

ϕ_s : Servo clock frequency in Hz ($f_{osc}/2$)

ϕ_s/n : Clock source of selected counter

DPPR2 is accessible by word access only. Byte access gives unassured results. Reads are disabled. DPPR1 and DPPR2 are initialized to H'F0 and H'0000 by a reset, and in standby mode.

Note: * The preset data value is calculated so that the counter will reach H'80000 when the error value is zero. When the counter value is latched as error data in the drum phase error data registers (DPER1 and DPER2), however, it is converted to a value referenced to H'00000.

(2) Drum Phase Error Data Registers (DPER1, DPER2)

DPER1

| | | | | | | | | |
|-----------------|---|---|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | | | | |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | R*/W | R*/W | R*/W | R*/W |

DPER2

| | | | | | | | | | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

Note: * Note that only detected error data can be read.

DPER1 and DPER2 consist of a 20-bit DPG phase error data register. The 20 bits are weighted as follows. Bit 3 of DPER1 is the MSB, and bit 0 of DPER2 is the LSB. When the rotational phase is correct, the data H'00000 is latched. Negative data will be latched if the drum leads the correct phase, and positive data if it lags. Values in DPER1 and DPER 2 are transferred to the digital filter circuit.

DPER1 and DPER2 are 20-bit readable/writable registers. When writing data to DPER1 and DPER2, write to DPER1 first, and then write to DPER2. DPER2 is accessible by word access only. Byte access gives unassured results. DPER1 and DPER2 are initialized to H'F0 and H'0000 by a reset, and in standby mode.

See the note on the drum phase preset data registers (DPPR1 and DPPR2) in section 28.7.4 (1).

(3) Drum Phase Error Detection Control Register (DPGCR)

| | | | | | | | | |
|-----------------|-------|-------|--------|-----|-------|---|---|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | DPCS1 | DPCS0 | DPOVF | N/V | HSWES | — | — | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| R/W : | R/W | R/W | R/(W)* | R/W | R/W | — | — | — |

Note: * Only 0 can be written

DPGCR controls the operation of drum phase error detection.

DPGCR is an 8-bit readable/writable register. Bits 2 to 0 are reserved, bit 5 accepts only read and 0 write.

It is initialized to H'07 by a reset or stand-by.

Bits 7 and 6: Clock Source Selection Bits (DPCS1, DPCS0)

Select the clock supplied to the counter. ($\phi_s = f_{osc}/2$)

| Bit 7 | Bit 6 | Description |
|-------|-------|--------------------------|
| DPCS1 | DPCS0 | |
| 0 | 0 | ϕ_s (Initial value) |
| | 1 | $\phi_s/2$ |
| 1 | 0 | $\phi_s/3$ |
| | 1 | $\phi_s/4$ |

Bit 5: Counter Overflow Flag (DPOVF)

The DPOVF flag indicates the overflow of the 20-bit counter. It is cleared by writing 0. Write 0 after reading 1. Also, setting has the highest priority in this flag. If a flag set and 0 write occurs simultaneously, the latter is nullified.

| Bit 5 | Description |
|-------|--|
| DPOVF | |
| 0 | Normal state (Initial value) |
| 1 | Indicates that an overflow has occurred in the counter |

Bit 4: Error Data Latch Signal Selection Bit (N/V)

Selects the latch signal of error data.

Bit 4

| N/V | Description |
|-----|--------------------------------------|
| 0 | HSW (VideoFF) signal (Initial value) |
| 1 | NHSW (NarrowFF) signal |

Bit 3: Edge Selection Bit (HSWES)

Selects the edge of the error data latch signal (HSW or NHSW).

Bit 3

| HSWES | Description |
|-------|--|
| 0 | Latches at the rising edge (Initial value) |
| 1 | Latches at the falling edge |

Bits 2 to 0: Reserved

No read or write is valid.

28.7.5 Description of Operation

The drum phase error detector detects the phase error based on the reference value set in the drum phase preset data register 1 and 2 (DPPR1, DPPR2). The reference values set in DPPR1 and DPPR2 are preset in the counter by the REF30P signal, and counted up by the clock selected. The latch of the error data can be selected between the rising or falling edge of HSW (NHSW). The error data detected in the error data automatic transmission mode (DFEPS bit of DFUCR = 0) is sent to the digital filter circuit automatically. In software transmission mode (DFEPS bit of DFUCR = 1), the data written in DPER1 and DPER2 is sent to the digital filter circuit. The error data is signed binary. It takes a positive number (+) if the phase is behind the specified phase, a negative number (-) if in advance of the specified phase, or 0 if it had no phase error (revolving at the specified phase). Figures 28.30 and 28.31 show examples of operation to detect a drum phase error.

(a) Drum phase error detection counter

The drum phase error detection counter stops the counter when overflow or latch occurred. At the same time, it generates an interrupt request (IRRDRM3), setting the overflow flag (DPOVF) if overflow occurred. Clear DPOVF by writing 0 after reading 1. If setting the flag and writing 0 take place simultaneously, the latter is nullified.

(b) Interrupt request

IRRDRM3 is generated by the HSW (NHSW) signal latch and the overflow of the error detection counter.

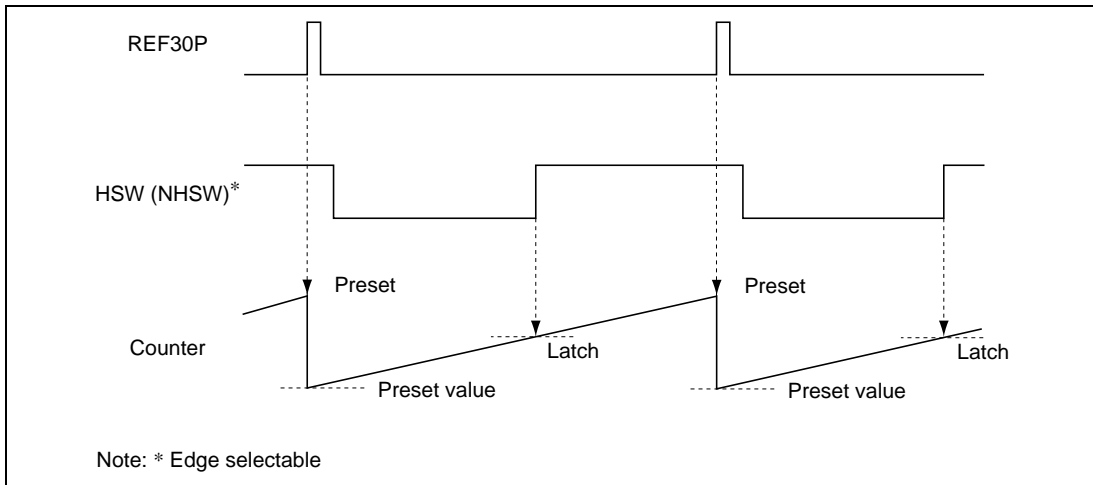


Figure 28.30 Drum Phase Control in Playback Mode (HSW Rising Edge Selected)

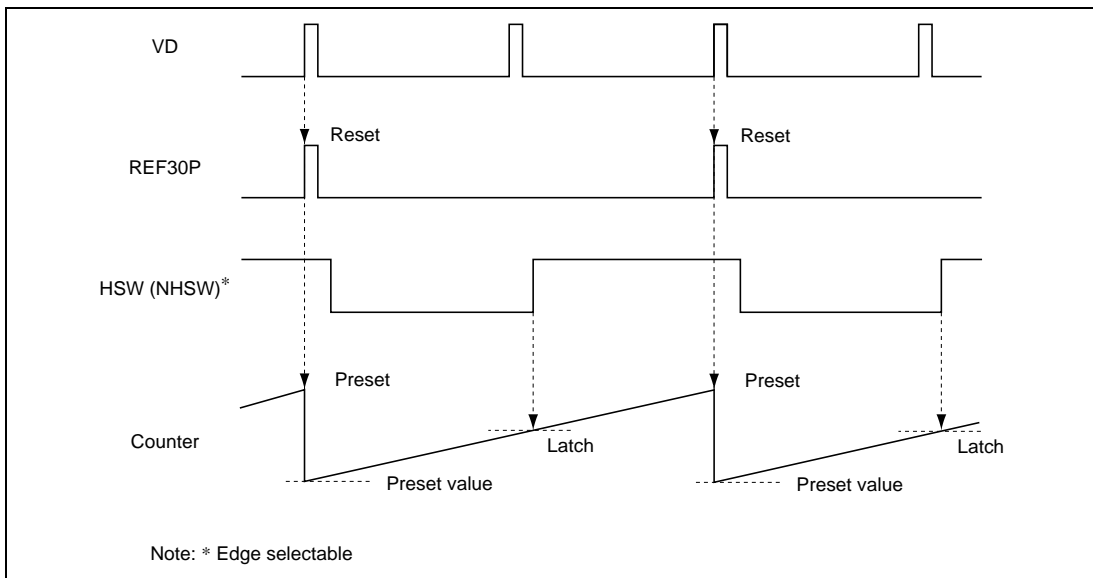


Figure 28.31 Drum Phase Control in Record Mode (HSW Rising Edge Selected)

28.7.6 Phase Comparison

The phase comparison circuit takes measures of the difference of time between the reference signal and the comparing signal with a digital counter. The REF30 signal is used for the reference signal, and the HSW signal (VideoFF) or HHSW signal (NarrowFF) from the HSW timing generator is used for the comparing signal. In record mode, however, the phase of the REF30 signal is the same as that of the vertical sync signal (Vsync) because the reference signal generator (REF30 generator) is reset by the vertical sync signal (Vsync) in the video signals. The error detection counter performs the data latching operation at the rising or falling edge of the HSW signal. The digital filter circuit performs computation using this data as 20-bit phase error data. After processing and adding the phase error data and the speed error data from the drum speed control system, the digital filter circuit sends the data as the error data of the drum system to the PWM modulation circuit.

28.8 Capstan Speed Error Detector

28.8.1 Overview

Capstan speed control operates so as to hold the capstan motor at a constant revolution speed, by measuring the period of the CFG signal. A digital counter detects the speed deviation from a preset value. The speed error data is added to phase error data in a digital filter. This filter controls a pulse-width modulated (PWM) output, which controls the revolution speed and phase of the capstan motor.

The CFG input signal is downloaded by the comparator circuit, then reshaped into a square wave by a reshaping circuit, divided by the CFG divider, and sent to the speed error detector as the DVCFG signal.

The speed error detector uses the system clock to measure the period of the DVCFG signal, and detects the deviation from a preset data value. The preset data is the value that would result from measuring the DVCFG signal period with the clock signal if the capstan motor was running at the correct speed.

The error detector operates by latching a counter value when it detects an edge of the DVCFG signal. The latched count provides 16 bits of speed error data for the digital filter to operate on. The digital filter adds the speed error data to phase error data from the capstan phase control system, then sends the result to the pulse-width modulator as capstan error data.

28.8.2 Block Diagram

Figure 28.32 shows a block diagram of the capstan speed error detector.

28.8.3 Register Configuration

Table 28.11 shows the register configuration of the capstan speed error detector.

Table 28.11 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|--|----------------|------------|-------------|----------------------|----------------|
| CFG speed preset data register | CFPR | W | Word | H'0000 | H'FD050 |
| CFG speed error data register | CFER | R/W | Word | H'0000 | H'FD052 |
| CFG lock UPPER data register | CFRUDR | W | Word | H'7FFF | H'FD054 |
| CFG lock LOWER data register | CFRLDR | W | Word | H'8000 | H'FD056 |
| Capstan speed error detection control register | CFVCR | R/W | Byte | H'00 | H'FD058 |

28.8.4 Register Descriptions

(1) CFG Speed Preset Data Register (CFPR)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

The 16-bit preset data that defines the specified CFG speed is set in CFPR. The preset data is referenced to H'8000*, and can be calculated from the following equation.

CFG speed preset data =H'8000 – ($\frac{\phi s/n}{\text{DVCFG frequency}}$ – 2)

ϕs : Servo clock frequency in Hz ($f_{osc}/2$)

DVCFG frequency: In Hz

The constant 2 is the preset interval (see figure 28.33).

$\phi s/n$: Clock source of the selected counter

CFPR is a 16-bit write-only register. CFPR is accessible by word access only. Byte access gives unassured results. No read is valid. If a read is attempted, an undetermined value is read out. CFPR is initialized to H'0000 by a reset, stand-by or module stop.

Note: * The preset data value is calculated so that the counter will reach H'8000 when the error is zero. When the counter value is latched as error data in the CFG speed error data register (CFER), however, it is converted to a value referenced to H'0000.

(2) CFG Speed Error Data Register (CFER)

| | | | | | | | | | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

Note: * Note that only detected error data can be read.

CFER is a 16-bit data register. When the speed of the capstan motor is correct, the data latched in CFER is H'0000. Negative data will be latched if the speed is too fast, and positive data if the speed is too slow. The CFER value is sent to the digital filter either automatically or by software.

CFER is a 16-bit readable/writable register. CFER is accessible by word access only. Byte access gives unassured results. CFER is initialized to H'0000 by a reset, and in module stop mode and standby mode.

See the note on the CFG speed preset data register (CFPR) in section 28.8.4 (1).

(3) CFG Lock UPPER Data Register (CFRUDR)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

CFRUDR is a register used to set the lock range on the UPPER side when capstan speed lock is detected, and to set the limit value on the UPPER side when the limiter function is in use.

When lock is being detected, if the capstan speed is detected within the lock range, the lock counter which has been set by the CFRCS1 and CFRCS0 bits of the CFVCR register counts down. If the set value of CFRCS1 and CFRCS0 matches the number of times of occurrence of locking, the computation of the digital filter in the capstan phase system can be controlled automatically. Also, if the CFG speed error data is beyond the CFRUDR value when the limiter function is in use, the CFRUDR value can be used as the data for computation by the digital filter.

CFRUDR is a 16-bit write-only register. Only a word access is valid. If a byte access is attempted, operation is not assured. A read is invalid. If a read is attempted, an undetermined value is read out. It is initialized to H'7FFF by a reset, stand-by or module-stop.

(4) CFG Lock LOWER Data Register (CFRLDR)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

CFRLDR is a register used to set the lock range on the LOWER side when capstan speed lock is detected, and to set the limit value on LOWER side when limiter function is in use. When lock is being detected, if the drum speed is detected within the lock range, the lock counter which has been set by the CFRCS1 and CFRCS0 bits of the CFVCR register counts down. If the set value of CFRCS1 and CFRCS0 matches the number of times of occurrence of locking, the computation of the digital filter in the drum phase system can be controlled automatically. Also, if the CFG speed error data is under the CFRLDR value when the limiter function is in use, the CFRLDR value can be used as the data for computation by the digital filter.

CFRLDR is a 16-bit write-only register. Only a word access is valid. If a byte access is attempted, operation is not assured. No read is valid. If a read is attempted, an undetermined value is read out. It is initialized to H'8000 by a reset, stand-by or module-stop.

(5) Capstan Speed Error Detection Control Register (CFVCR)

| | | | | | | | | |
|-----------------|-------|-------|---------------------|--------|----------|-------|----------------------|----------------------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CFCS1 | CFCS0 | CFOVF | CFRFON | CF-R/UNR | CPCNT | CFRCS1 | CFRCS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/(W) ^{*1} | R/W | R | R/W | (R) ^{*2} /W | (R) ^{*2} /W |

- Notes: 1. Only 0 can be written.
2. If read-accessed, the counter value is read out.

CFVCR controls the operation of capstan speed error detection. CFVCR is an 8-bit readable/writable register. Bit 3 accepts only read, and bit 5 accepts only read and 0 write. It is initialized to H'00 by a reset, stand-by or module-stop.

Bits 7 and 6: Clock Source Selection Bits (CFCS1, CFCS0)

CFCS1 and CFCS0 select the clock to be supplied to the counter. ($\phi_s = f_{osc}/2$)

| Bit 7 | Bit 6 | Description |
|-------|-------|--------------------------|
| CFCS1 | CFCS0 | |
| 0 | 0 | ϕ_s (Initial value) |
| | 1 | $\phi_s/2$ |
| 1 | 0 | $\phi_s/4$ |
| | 1 | $\phi_s/8$ |

Bit 5: Counter Overflow Flag (CFOVF)

The CFOVF flag indicates overflow of the 16-bit counter. It is cleared by writing 0. Write 0 after reading 1. Also, setting has the highest priority in this flag. If a flag set and 0 write occurs simultaneously, the latter is nullified.

Bit 5

| CFOVF | Description |
|-------|--|
| 0 | Normal state (Initial value) |
| 1 | Indicates that an overflow has occurred in the counter |

Bit 4: Error Data Limit Function Selection Bit (CFRFON)

Makes the error data limit function valid. (Limit values are the values set in the lock range data register (CFRUDR, CFRLLDR)).

Bit 4

| CFRFON | Description |
|--------|------------------------------------|
| 0 | Limit function off (Initial value) |
| 1 | Limit function on |

Bit 3: Capstan Lock Flag (CF-R/UNR)

Sets a flag if an underflow occurred in the capstan lock counter.

Bit 3

| CF-R/UNR | Description |
|----------|---|
| 0 | Indicates that the capstan speed system is not locked (Initial value) |
| 1 | Indicates that the capstan speed system is locked |

Bit 2: Capstan Phase System Filter Computation Automatic Start Bit (CPCNT)

Sets on the filter computation of the phase system if an underflow occurred in the capstan lock counter.

Bit 2

| CPCNT | Description |
|-------|--|
| 0 | Does not perform the filter computation by detection of the capstan lock (Initial value) |
| 1 | Set on the filter computation of the phase system when capstan lock is detected |

Bits 1 and 0: Capstan Lock Counter Setting Bits (CFRCS1, CFRCS0)

Sets the number of times where drum lock has been determined (DVCFG has been detected in the range set by the lock range data register). It sets the capstan lock flag if it detected the set number of times of occurrence of capstan lock. If a DVCFG signal is detected outside the lock range after data is written in CFRCS1 and CFRCS0, data is stored in the lock counter.

Note: If CFRCS1 or CFRCS0 is read-accessed, the counter value is read out. If bit 3 (capstan lock flag) is 1 and the capstan lock counter's value is 3, it indicates that the capstan speed system is locked. The capstan lock counter stops until lock is released after underflow.

| Bit 1 | Bit 0 | Description |
|--------|--------|--|
| CFRCS1 | CFRCS0 | |
| 0 | 0 | Underflow after lock was detected once (Initial value) |
| | 1 | Underflow after lock was detected twice |
| 1 | 0 | Underflow after lock was detected three times |
| | 1 | Underflow after lock was detected four times |

28.8.5 Description of Operation

The capstan speed error detector detects the speed error based on the reference value set in the CFG speed preset register (CFPR). The reference value set in CFPR is preset in the counter by the DVCFG signal, and counts down by the selected clock. The timing of the counter presetting and the error data latching can be selected between the rising or falling edge of the DVCFG signal. See section 28.14.3, CFG Frequency Divider. The error data detected is sent to the digital filter circuit. The error data is signed binaries. It takes a positive number (+) if the speed is slower than the specified speed, a negative number (-) if the speed is faster, or 0 if it had no error (revolving at the specified speed). Figure 28.33 shows an example of operation to detect the capstan speed.

(a) Setting the error data limit

A limit can be set to the error data sent to the digital filter circuit using the CFG lock data register (CFRUDR, CFRLDR). Set the upper limit of the error data in CFRUDR and the lower limit in CFRLDR, and write 1 in the CFRFON bit. If the error data is beyond the limit range, the CFRLDR value is sent if a negative number is latched, or the CFRUDR value is sent if a positive one is latched, as a limit value. Be sure to turn off the limit setting (CFRFON = 0) when you set the limit value. If the limit was set with the limit setting on (CFRFON = 1), result of computation is not assured.

(b) Lock detection

If error data was detected within the lock range set in the lock data register, the capstan lock flag (CF-R/UNR) is set by the number of the times of occurrence of locking set by the CFRCS1 and CFRCS0 bits, and an interrupt is requested (IRRCAP2) at the same time. The number of the occurrence of locking (once to 4 times) can be specified when setting the flag. Use the CFRCS1 and CFRCS0 bits for this purpose. Also, if bit 5 (CPHA bit) of the capstan system digital filter control register (CFIC) is 0 (phased system digital filter computation off) and the DPCNT bit is 1, turning on/off of the phase system digital filter computation can be controlled automatically by the status of lock detection.

(c) Capstan system speed error detection counter

The capstan system speed error detection counter stops the counter and sets the overflow flag (CFOVF) when overflow occurred. At the same time, it generates an interrupt request (IRRCAP1). Clear CFOVF by writing 0 after reading 1. If setting the flag and writing 0 take place simultaneously, the latter is nullified.

(d) Interrupt request

IRRCAP1 is generated by the DVCFG signal latch and the overflow of the error detection counter. IRRCAP2 is generated by detection of lock (after the detection of the number of times of setting).

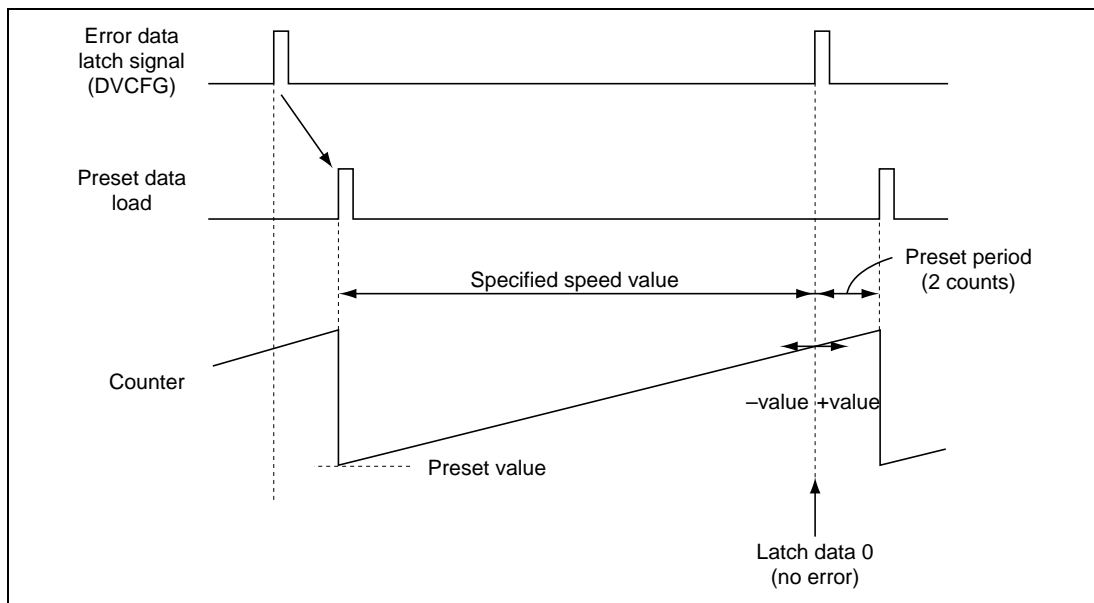


Figure 28.33 Example of the Operation of the Capstan Speed Error Detection

28.9 Capstan Phase Error Detector

28.9.1 Overview

The capstan phase control system is required to start operation after the capstan motor has arrived at the specified speed under the control of the speed control system. The capstan phase control system operates in the following way in record/playback mode.

In record mode: Controls the tape running so that it may run at a specified speed together with the speed control system.

In playback mode: Controls the tape running so that the recorded track may be traced correctly. Any error deviated from the reference phase is detected by the digital counter. This phase error data and the speed error data is processed and added by the digital filter circuit to control the PWM output. The phase and speed of the capstan, in turn, is controlled by this PWM output. The control signal of the capstan phase control in REC mode differs from that in PB mode. In REC mode, the control is performed by the DVCFG2 signal which is generated by dividing the frequencies of the reference signal (REF30P or CREF) and the CFG signal. In PB mode, it is performed by divided rising signal (DVCTL) of the reference signal (CAPREF30) and the playback control pulse (PB-CTL).

The reference signal in record and playback modes are as follows.

In record mode: 1/2 Vsync signal extracted from the video signal to be recorded

In playback mode: Signal generated by dividing the PB-CTL signal (DVCTL) at its rising edge

28.9.2 Block Diagram

Figure 28.34 shows the block diagram of the capstan phase error detector.

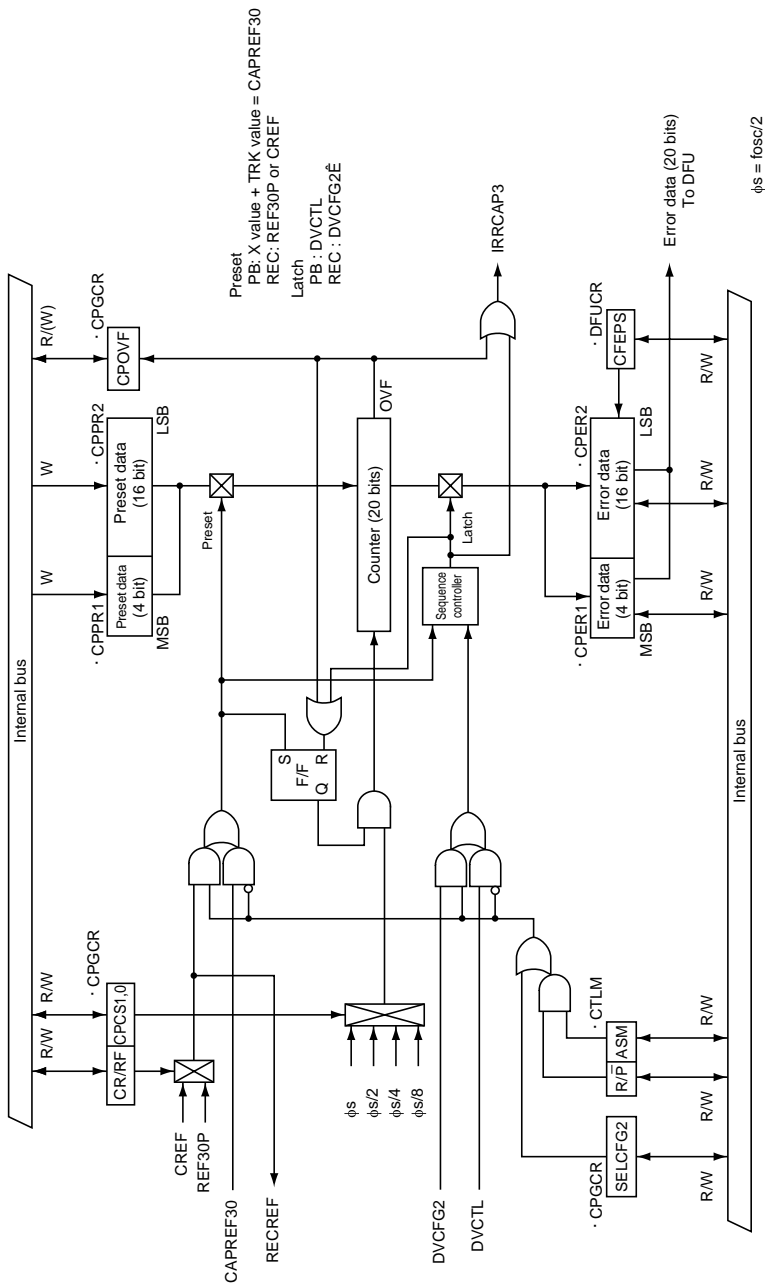


Figure 28.34 Block Diagram of Capstan Phase Error Detector

28.9.3 Register Configuration

Table 28.12 shows the register configuration of the capstan phase error detector.

Table 28.12 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|--|----------------|------------|-------------|----------------------|-----------------|
| Capstan phase preset data register 1 | CPPR1 | W | Byte | H'F0 | H'FD05C |
| Capstan phase preset data register 2 | CPPR2 | W | Word | H'0000 | H'FD05A |
| Capstan phase error data register 1 | CPER1 | R/W | Byte | H'F0 | H'FD05D |
| Capstan phase error data register 2 | CPER2 | R/W | Word | H'0000 | H'FD05E |
| Capstan phase error detection control register | CPGCR | R/W | Byte | H'07 | H'FD059 |

28.9.4 Register Descriptions

(1) Capstan Phase Preset Data Registers (CPPR1, CPPR2)

CPPR1

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | | | | |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W |

CPPR2

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

The 20-bit preset data that defines the specified capstan phase is set in CPPR1 and CPPR2. The 20 bits are weighted as follows. Bit 3 of CPPR1 is the MSB. Bit 0 of CPPR2 is the LSB. When CPPR2 is written to, the 20-bit preset data, including CPPR1, is loaded into the preset circuit. Write to CPPR1 first, and CPPR2 next. The preset data is referenced to H'80000*, and can be calculated from the following equation.

Target phase difference = Rreference signal frequency/2
Capstan phase preset data = H'80000 – (φs/n × target phase difference)

φs: Servo clock frequency in Hz (fosc/2)
φs/n: Clock source of selected counter

CPPR2 is accessible by word access only. Byte access gives unassured results. Reads are disabled. If read is attempted to CPPR1 or CPPR2, an undetermined value is read out. CPPR1 and CPPR2 are initialized to H'F0 and H'0000 by a reset, and in standby mode.

Note: * The preset data value is calculated so that the counter will reach H'80000 when the error is zero. When the counter value is latched as error data in the capstan phase error data registers (CPER1 and CPER2), however, it is converted to a value referenced to H'00000.

(2) Capstan Phase Error Data Registers (CPER1, CPER2)

| | | | | | | | | |
|-----------------|---|---|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | | | | |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | R*/W | R*/W | R*/W | R*/W |

| | | | | | | | | | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

Note: * Note that only detected error data can be read.

CPER1 and CPER2 constitute a 20-bit capstan phase error data register. The 20 bits are weighted as follows. Bit 3 of CPER1 is the MSB. Bit 0 of CPER2 is the LSB. When the rotational phase is correct, the data H'00000 is latched. Negative data will be latched if the phase leads the correct phase, and positive data if it lags. Values in CPER1 and CPER 2 are transferred to the digital filter circuit.

CPER1 and CPER are 20-bit readable/writable registers. When writing data to CPER1 and CPER2, write to CPER1 first, and then write to CPER2. CPER2 is accessible by word access only. Byte access gives unassured results. CPER1 and CPER2 are initialized to H'F0 and H'0000 by a reset, and in standby mode.

See the note on the capstan phase preset data registers (CPPR1 and CPPR2) in section 28.9.4 (1).

(3) Capstan Phase Error Detection Control Register (CPGCR)

| | | | | | | | | |
|-----------------|-------|-------|--------|-------|---------|---|---|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CPCS1 | CPCS0 | CPOVF | CR/RF | SELCFG2 | — | — | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| R/W : | R/W | R/W | R/(W)* | R/W | R/W | — | — | — |

Note: * Only 0 can be written

CPGCR controls the operation of capstan phase error detection.

CPGCR is an 8-bit readable/writable register. Bits 2 to 0 are reserved, bit 5 accepts only read and 0 write.

It is initialized to H'07 by a reset or stand-by.

Bits 7 and 6: Clock Source Selection Bits (CPCS1, CPCS0)

Select the clock supplied to the counter. ($\phi_s = f_{osc}/2$)

| Bit 7 | Bit 6 | Description |
|-------|-------|--------------------------|
| CPCS1 | CPCS0 | |
| 0 | 0 | ϕ_s (Initial value) |
| | 1 | $\phi_s/2$ |
| 1 | 0 | $\phi_s/4$ |
| | 1 | $\phi_s/8$ |

Bit 5: Counter Overflow Flag (CPOVF)

CPOVF flag indicates the overflow of the 20-bit counter. It is cleared by writing 0. Write 0 after reading 1. Also, setting has the highest priority in this flag. If a flag set and 0 write occurs simultaneously, the latter is nullified.

Bit 5

| CPOVF | Description |
|-------|--|
| 0 | Normal state (Initial value) |
| 1 | Indicates that an overflow has occurred in the counter |

Bit 4: Preset Signal Selection Bit (CR/RF)

Selects the preset signal.

Bit 4

| CR/RF | Description |
|-------|---------------------------------------|
| 0 | Presets REF30P signal (Initial value) |
| 1 | Presets CREF signal |

Bit 3: Preset and Latch Signal Selection Bit (SELCFG2)

Selects the counter preset signal and the error data latch signal data in PB (ASM) mode.

Bit 3

| SELCFG2 | Description |
|---------|---|
| 0 | Presets CAPREF30 signal; latches DVCTL signal (Initial value) |
| 1 | Presets REF30P (CREF) signal; latches DVCFG2 signal |

Bits 2 to 0: Reserved

No read or write is valid.

28.9.5 Description of Operation

The capstan phase error detector detects the phase error based on the reference value set in the capstan specified phase preset data register 1 and 2 (CPPR1, CPPR2). The reference values set in CPPR1 and CPPR2 are preset in the counter by the REF30P (CREF) signal or CAPREF30 signal, and counted up by the clock selected. The latching of the error data is performed by DVCTL or DVCFG2.

The error data detected in the error data automatic transmission mode (CFEPS bit of DFUCR = 0) is sent to the digital filter circuit automatically. In software transmission mode (CFEPS bit of DFUCR = 1), the data written in CPER1 and CPER2 is sent to the digital filter circuit. The error data is signed binary. It takes a positive number (+) if the phase is behind the specified phase, a negative number (-) if in advance of the specified phase, or 0 if it had no phase error (revolving at the specified phase). Figures 28.35 and 28.36 show examples of operation to detect a capstan phase error.

(a) Capstan phase error detection counter

The capstan phase error detection counter stops the counter when overflow or latch occurred. At the same time, it generates an interrupt request (IRRCAP3), setting the overflow flag (CPOVF) if overflow occurred. Clear CPOVF by writing 0 after reading 1. If setting the flag and writing 0 take place simultaneously, the latter is nullified.

(b) Interrupt request

IRRCAP3 is generated by the DVCTL or DVCFG2 signal latch and the overflow of the error detection counter.

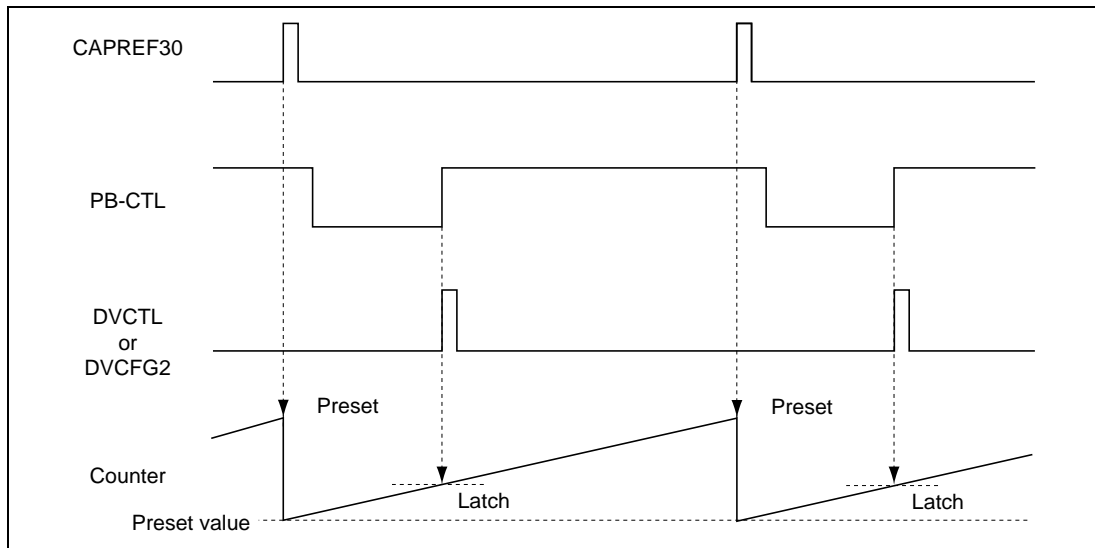


Figure 28.35 Capstan Phase Control in Playback Mode

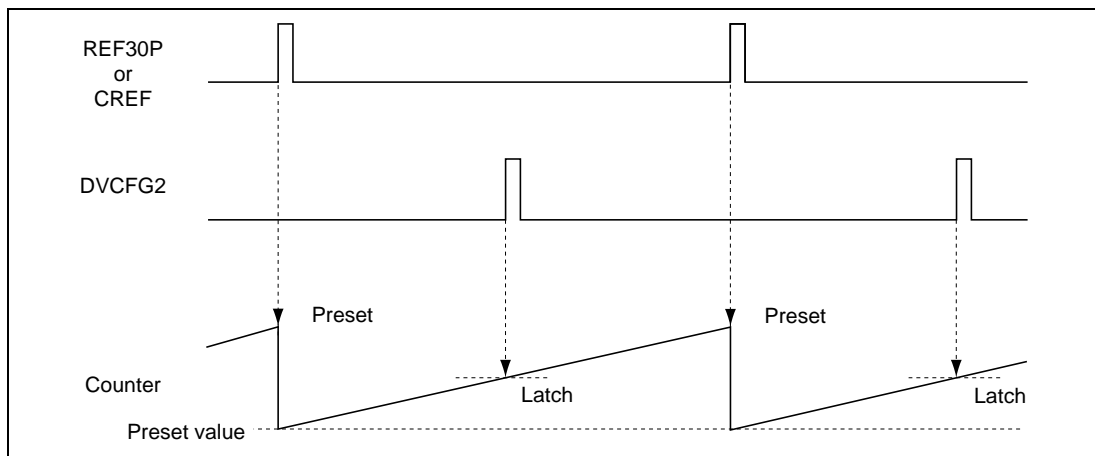


Figure 28.36 Capstan Phase Control in Record Mode

28.10 X-Value and Tracking Adjustment Circuit

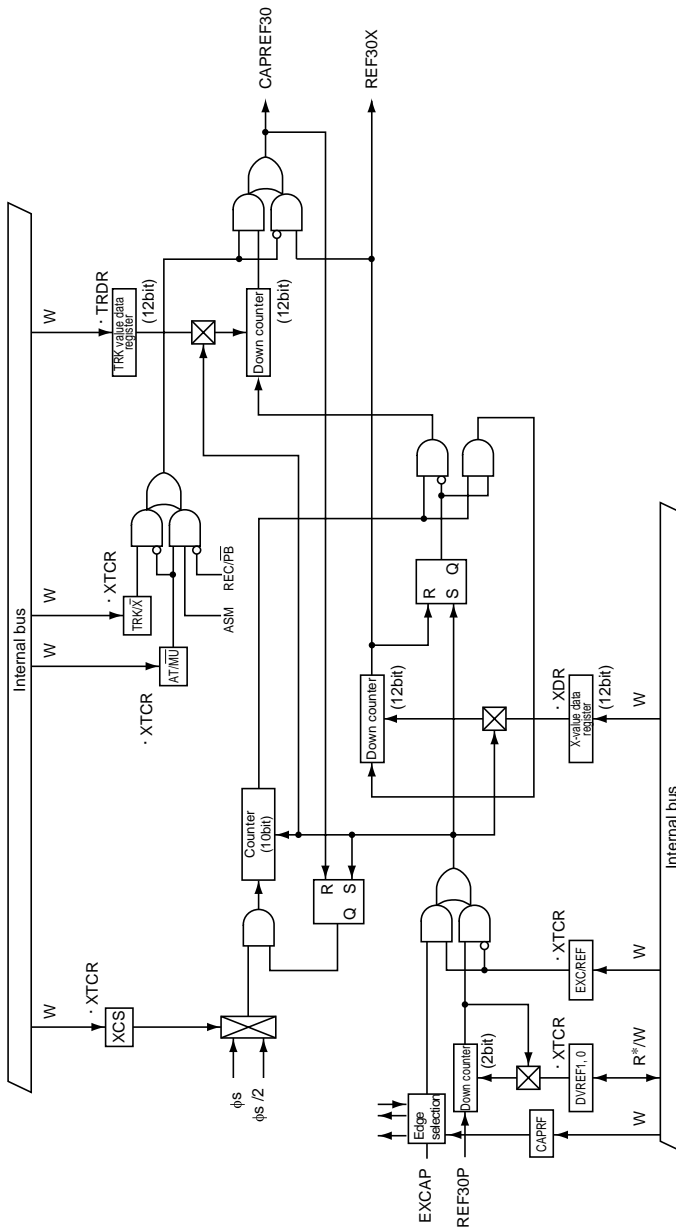
28.10.1 Overview

To maintain compatibility with other VCRs, an on-chip adjustment circuit adjusts the phase of the reference signal (internal reference signal (REF30) or external reference signal (EXCAP)) during playback. Because of manufacturing tolerances, the physical distance between the video head and control head (the X-value: 79.244 mm) may vary from set to set, so when a tape that was recorded on a different set is played back, the phase of the reference signal may need to be adjusted. The adjustment can be made by a register setting. The same setting can adjust the rotational phase of the capstan motor to maintain positional alignment (tracking alignment) of the video head with the recorded tracks in autotracking, or when tracks that were recorded with an EP head are traced by a wider head. These tracking adjustments can be made by acquisition of the envelope signal by the A/D converter.

28.10.2 Block Diagram

The adjustment circuit consists of a 10-bit counter clocked by the servo clock (ϕ s or ϕ s/2), and two down-counters with load register. Individual setting of X-value adjustment can be made by the X-value data register (XDR) and tracking adjustment by the TRK data register (TRDR). The reference signal clears the 10-bit counter and sets the load register value in the down-counter with two load registers. After the adjusted reference signal is generated, clock supply stops and the circuit halts until the next reference signal is input. The REF30 signal can be divided (by 2 to 4) as necessary.

Figure 28.37 shows a block diagram.



Note: * When DVREF1 and DVREF0 are read, values in the down counter (2 bits) are read out.
 $\phi s = f_{osc}/2$

Figure 28.37 Block Diagram of X-Value Adjustment Circuit

28.10.3 Register Descriptions

(1) Register Configuration

Table 28.13 shows the register configuration of X-value adjustment and tracking adjustment circuits.

Table 28.13 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address* |
|--|---------|-----|------|---------------|----------|
| X-value and TRK-value control register | XTCR | R/W | Byte | H'80 | H'FD074 |
| X-value data register | XDR | W | Word | H'F000 | H'FD070 |
| TRK-value data register | TRDR | W | Word | H'F000 | H'FD072 |

(2) X-value and TRK-value Control Register (XTCR)

| | | | | | | | | |
|-----------------|---|-------|-------|----------------|---------|-----|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | CAPRF | AT/MU | TRK/ \bar{X} | EXC/REF | XCS | DVREF1 | DVREF0 |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | W | W | W | W | W | R/W | R/W |

XTCR is an 8-bit register to determine the X-value and TRK-value adjustment circuits. Bits 6 to 2 are write-only bits. No read is valid. If a read is attempted, an undetermined value is read out. Bits 1 and 0 are readable/writable bits. XTCR accepts only a byte access. If a word access is attempted, operation is unassured.

It is initialized to H'80 by a reset, stand-by or module stop.

Bit 7: Reserved

No write is valid. If a read is attempted, an undetermined value is read out.

Bit 6: External Sync Signal Edge Selection Bit (CAPRF)

Selects the EXCAP edge when a selection is made to generate external sync signals.

Bit 6

| CAPRF | Description |
|-------|--|
| 0 | Signal generated at the rising edge of EXCAP (Initial value) |
| 1 | Signal generated at both edges of EXCAP |

Bit 5: Capstan Phase Correction Auto/Manual Selection Bit (AT/ $\overline{\text{MU}}$)

Selects whether the generation of the correction reference signal (CAPREF30) for capstan phase control is controlled automatically or manually depending on the status of the ASM and REC/ $\overline{\text{PB}}$ bits of the CTL mode register.

Bit 5

| AT/$\overline{\text{MU}}$ | Description | |
|---|--------------------|-----------------|
| 0 | Manual mode | (Initial value) |
| 1 | Auto mode | |

Bit 4: Capstan Phase Correction Register Selection Bit (TRK/ $\overline{\text{X}}$)

Determines the method to generate the CAPREF30 signal when the AT/ $\overline{\text{MU}}$ bit is 0.

Bit 4

| TRK/$\overline{\text{X}}$ | Description | |
|---|--|-----------------|
| 0 | Generates CAPREF30 only by the set value of XDR | (Initial value) |
| 1 | Generates CAPREF30 by the set values of XDR and TRDR | |

Bit 3: Reference Signal Selection Bit (EXC/REF)

Selects the reference signal to generate the correction reference signal (CAPREF30).

Bit 3

| EXC/REF | Description | |
|----------------|---|-----------------|
| 0 | Generates the signal based on REF30P | (Initial value) |
| 1 | Generates the signal based on the external reference signal | |

Bit 2: Clock Source Selection Bit (XCS)

Selects the clock source to be supplied to the 10-bit counter.

Bit 2

| XCS | Description | |
|------------|--------------------|-----------------|
| 0 | ϕS | (Initial value) |
| 1 | $\phi\text{S}/2$ | |

Bits 1 and 0: REF30P Division Ratio Selection Bits (DVREF1, DVREF0)

Select the division value of REF30P. If it is read-accessed, the counter value is read out. (The selected division value is set by the UDF of the counter.)

| Bit 1 | Bit 0 | Description |
|--------|--------|-------------------------------|
| DVREF1 | DVREF0 | |
| 0 | 0 | Division in 1 (Initial value) |
| | 1 | Division in 2 |
| 1 | 0 | Division in 3 |
| | 1 | Division in 4 |

(3) X-value Data Register (XDR)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | XD11 | XD10 | XD9 | XD8 | XD7 | XD6 | XD5 | XD4 | XD3 | XD2 | XD1 | XD0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

The X-value data register (XDR) is a 16-bit write-only register. No read is valid. If a read is attempted, an undefined value is read out. XDR accepts only a word-access. If a byte access is attempted, operation is not assured.

Set X-value correction data to XDR, except a value which is beyond the cycle of the CTL pulse.

If $AT/\overline{MU} = 0$, $TRK/\overline{X} = 0$ was set, CAPREF30 can be generated only by the setting of XDR.

Set an X-value and TRK correction value in PB mode, and an X-value in REC mode.

It is initialized to H'F000 by a reset, stand-by or module stop.

(4) TRK-value Data Register (TRDR)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | TRD11 | TRD10 | TRD9 | TRD8 | TRD7 | TRD6 | TRD5 | TRD4 | TRD3 | TRD2 | TRD1 | TRD0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

The TRK-value data register (TRDR) is a 16-bit write-only register. No read is valid. If a read is attempted, an undefined value is read out. TRDR accepts only a word-access. If a byte access is attempted, operation is not assured.

Set a TRK-value correction data to TRDR, except a value which is beyond the cycle of the CTL pulse. It is initialized to H'F000 by a reset, stand-by or module stop.

28.11 Digital Filters

28.11.1 Overview

The digital filters required in servo control make extensive use of multiply-accumulate operations on signed integers (error data) and coefficients. A filter computation circuit (digital filter computation circuit) is provided in on-chip hardware to reduce the load on software, and to improve processing efficiency. Figure 28.38 shows a block diagram of the digital filter computation circuit configuration.

The filter computation circuit includes a high-speed 24-bit \times 16-bit multiplier-accumulator, an arithmetic buffer, and an I/O processor. The digital filter computations are carried out by the high-speed multiplier-accumulator. The arithmetic buffer stores coefficients and gain constants needed in the filter computations, which are referenced by the high-speed multiplier-accumulator.

The I/O processor is activated by a frequency generator signal, and determines what operation is carried out. When activated, it reads the speed error and phase error from the speed and phase error detectors and sends them to the accumulator.

When the filter computation is completed, the I/O processor reads the result from the accumulator and sends it to a 12-bit PWM. At this time, the accumulation result gain can be controlled.

28.11.2 Block Diagram

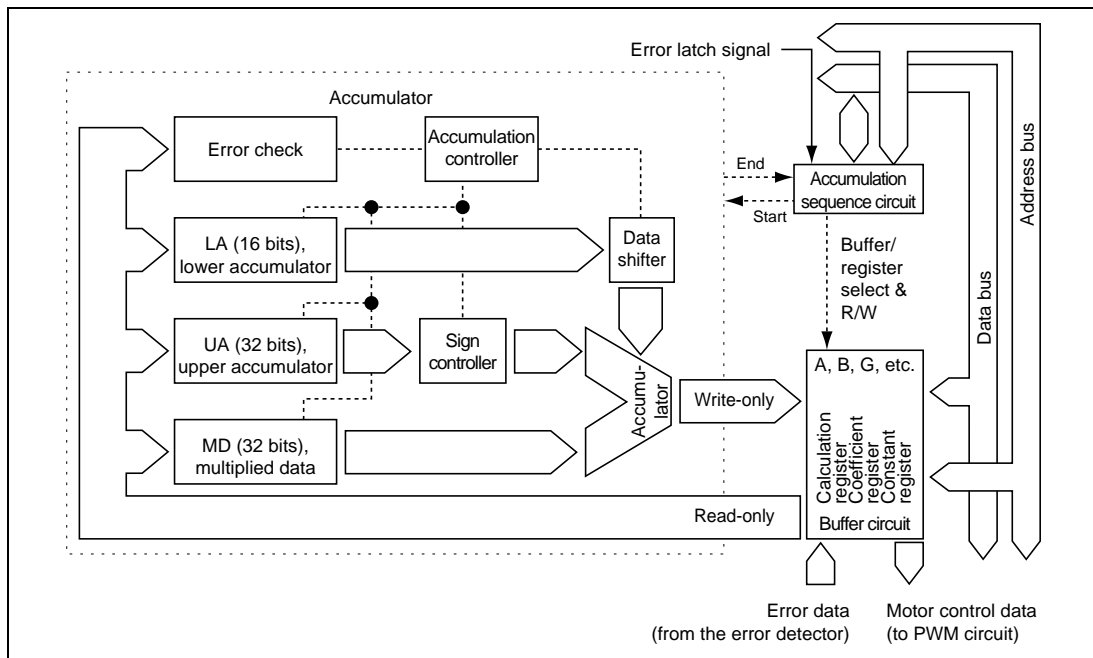


Figure 28.38 Block Diagram of Digital Filter Circuit

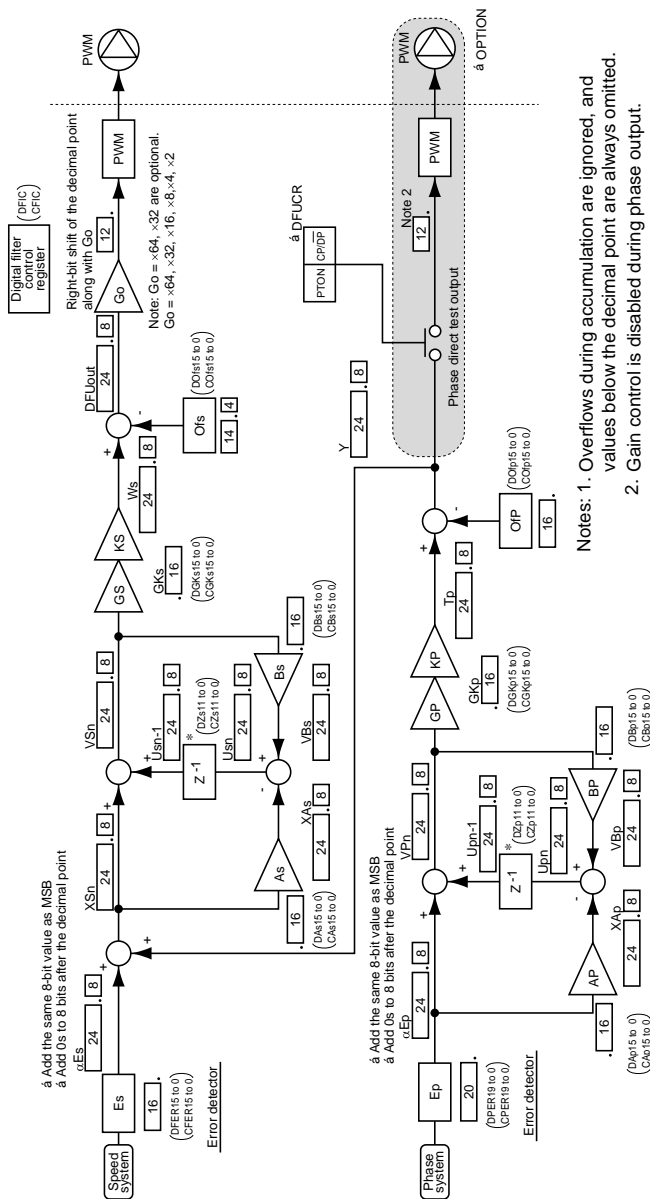


Figure 28.39 Digital Filter Representation

28.11.3 Arithmetic Buffer

This buffer stores computational data used in the digital filters. See table 28.14. Write access is limited to the gain and coefficient data (Z^{-1}). Other data is used by hardware. None of the data can be read.

Table 28.14 Arithmetic Buffer Register Configuration

| | | Buffer Data Length | | |
|--------------|---------------------|---------------------|-----------------|---------|
| | | 16 bits | 16 bits | 16 bits |
| | Arithmetic Data | Gain or Coefficient | Processing Data | |
| Phase system | Ep | | | |
| | Upn | | | |
| | Upn-1 (Zp^{-1}) | | | |
| | Vpn | | | |
| | Tp | | | |
| | Y | | | |
| | Ap | | | |
| | Bp | | | |
| | GKp | | | |
| | Ofp | | | |
| Speed system | | | Ap × Epn | |
| | | | Bp × Vpn | |
| | Es | | | |
| | Xsn | | | |
| | Usn | | | |
| | Usn-1 ($Z^{-1}s$) | | | |
| | Vsn | | | |
| | Ws | | | |
| | As | | | |
| | Bs | | | |
| Error output | | | As × Xsn | |
| | | | Bs × Vsn | |
| | PWM | | | |
| | | | | |

Legend:

Valid bits

Non-existent bits

↑

Decimal point

28.11.4 Register Configuration

Table 28.15 shows the register configuration of the digital filter computation circuit.

Table 28.15 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|--|----------------|------------|-------------|----------------------|----------------|
| Capstan phase gain constant | CGKp | W | Word | Undetermined | H'FD010 |
| Capstan speed gain constant | CGKs | W | Word | Undetermined | H'FD012 |
| Capstan phase coefficient A | CAP | W | Word | Undetermined | H'FD014 |
| Capstan phase coefficient B | CBp | W | Word | Undetermined | H'FD016 |
| Capstan speed coefficient A | CAs | W | Word | Undetermined | H'FD018 |
| Capstan speed coefficient B | CBs | W | Word | Undetermined | H'FD01A |
| Capstan phase offset | COfp | W | Word | Undetermined | H'FD01C |
| Capstan speed offset | COfs | W | Word | Undetermined | H'FD01E |
| Drum phase gain constant | DGKp | W | Word | Undetermined | H'FD000 |
| Drum speed gain constant | DGKs | W | Word | Undetermined | H'FD002 |
| Drum phase coefficient A | DAP | W | Word | Undetermined | H'FD004 |
| Drum phase coefficient B | DBp | W | Word | Undetermined | H'FD006 |
| Drum speed coefficient A | DAs | W | Word | Undetermined | H'FD008 |
| Drum speed coefficient B | DBs | W | Word | Undetermined | H'FD00A |
| Drum phase offset | DOfp | W | Word | Undetermined | H'FD00C |
| Drum speed offset | DOfs | W | Word | Undetermined | H'FD00E |
| Drum system speed delay initialization register | DZs | W | Word | H'F000 | H'FD020 |
| Drum system phase delay initialization register | DZp | W | Word | H'F000 | H'FD022 |
| Capstan system speed delay initialization register | CZs | W | Word | H'F000 | H'FD024 |
| Capstan system phase delay initialization register | CZp | W | Word | H'F000 | H'FD026 |
| Drum system digital filter control register | DFIC | R/W | Byte | H'80 | H'FD028 |
| Capstan system digital filter control register | CFIC | R/W | Byte | H'80 | H'FD029 |
| Digital filter control register | DFUCR | R/W | Byte | H'C0 | H'FD02A |

28.11.5 Register Descriptions

(1) Gain Constants (DGKp, DGKs, CGKp, CGKs)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

Note: * Initial value is uncertain.

These registers are 16-bit write-only buffers that set accumulation gain of the digital filter. They cannot be read. They can be accessed by word access only. Accumulation gain can be set to gain 1 value as the maximum value. Byte access gives unassured results. If read is attempted, an undetermined value is read out.

These registers are not initialized by a reset or in standby mode. Be sure to write data in them before processing starts.

In the digital filter, output gain and accumulation gain can be adjusted separately. Take output gain into account when setting accumulation gain.

(2) Coefficients (DAp, DBp, DAs, DBs, CAp, CBp, CAs, CBs)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

Note: * Initial value is uncertain.

These registers are 16-bit write-only buffers that determine the cutoff frequency f1 and f2. They cannot be read. They can be accessed by word access only. Byte access gives unassured results. If read is attempted, an undetermined value is read out.

These registers are not initialized by a reset or in standby mode. Be sure to write data in them before processing starts.

In the digital filter, output gain and accumulation gain can be adjusted separately. Take output gain into account when setting accumulation gain.

(3) Offsets (DOfp, DOfs, COfp, COfs)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

Note: * Initial value is uncertain.

These registers are 16-bit write-only buffers that set the offset level of digital filter output. They cannot be read. They can be accessed by word access only. Byte access gives unassured results. If read is attempted, an undetermined value is read out.

These registers are not initialized by a reset or in standby mode. Be sure to write data in them before processing starts.

In this digital filter, output gain adjustment ($\times 1, 2, 4, 8, 16, 32, 64$) after offset adding is enabled. Take output gain into account when setting accumulation gain.

(4) Delay Initialization Registers (CZp, CZs, DZp, DZs)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

The delay initialization register is a 16-bit write-only register. It accepts only a word-access. If a byte access is attempted, operation is not assured. If a read is attempted, an undefined value is read out. Bits 12 to 15 are reserved, and no write in them is valid.

It is initialized to H'F000 by a reset, stand-by or module stop. The MSB of 12-bit data (bit 11) is a sign bit.

Loading to Z^{-1} is performed automatically by bits 4 and 3 of CFIC and DFIC (CZPON, CZSON, DZPON, DZSON). Writing in register is always available, but loading in Z^{-1} is not possible when the digital filter is performing calculation processing in relation to such register. In such a case, loading to Z^{-1} will be done the next time computation begins.

(5) Drum System Digital Filter Control Register (DFIC)

| | | | | | | | | |
|-----------------|---|--------|-------|-------|-------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | DROV | DPHA | DZPON | DZSON | DSG2 | DSG1 | DSG0 |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | R/(W)* | R/(W) | R/W | R/W | R/W | R/W | R/W |

Note: * Only 0 can be written

DFIC is an 8-bit readable/writable register that controls the status of the drum system digital filter and operating mode. It can be accessed by byte access only. Word access gives unassured results.

Bit 7 is a reserved bit. Writes are disabled. If read is attempted, an undetermined value is read out. DFIC is initialized to H'80 by a reset, and in standby mode and module stop mode.

Bit 7: Reserved

Reads and writes are both disabled.

Bit 6: Drum System Range Over Flag (DROV)

This flag is set to 1 when the result of a drum system filter computation exceeds 12 bits in width. To clear this flag, write 0.

Bit 6

| DROV | Description |
|------|---|
| 0 | Indicates that the filter computation result did not exceed 12 bits (Initial value) |
| 1 | Indicates that the filter computation result exceeded 12 bits |

Bit 5: Drum Phase System Filter Computation Start Bit (DPHA)

Starts or stops filter processing for the drum phase system.

Bit 5

| DPHA | Description |
|------|---|
| 0 | Phase system filter computations are disabled Phase computation result (Y) is not added to Es (see figure 28.39) (Initial value) |
| 1 | Phase system filter computations are enabled |

Bit 4: Drum Phase System Z^{-1} Initialization Bit (DZPON)

Reflects the DZp value on Z^{-1} of the phase system when computation processing of the drum phase system begins. If 1 was written, it is reflected on the computation, and then cleared to 0. Set this bit after writing data to DZp.

Bit 4

| DZPON | Description |
|-------|--|
| 0 | DZp value is not reflected on Z^{-1} of the phase system (Initial value) |
| 1 | DZp value is reflected on Z^{-1} of the phase system |

Bit 3: Drum Speed System Z^{-1} Initialization Bit (DZSON)

Reflects the DZs value on Z^{-1} of the speed system when computation processing of the drum speed system begins. If 1 was written, it is reflected on the computation, and then cleared to 0. Set this bit after writing data to DZs.

Bit 3

| DZSON | Description |
|-------|--|
| 0 | DZs value is not reflected on Z^{-1} of the speed system (Initial value) |
| 1 | DZs value is reflected on Z^{-1} of the speed system |

Bits 2 to 0: Drum System Output Gain Control Bits (DSG2, DSG1, DSG0)

Control the gain output to DRMPWM.

| Bit 2 | Bit 1 | Bit 0 | Description |
|-------|-------|-------|----------------------------|
| DSG2 | DSG1 | DSG0 | |
| 0 | 0 | 0 | $\times 1$ (Initial value) |
| | | 1 | $\times 2$ |
| | 1 | 0 | $\times 4$ |
| | | 1 | $\times 8$ |
| 1 | 0 | 0 | $\times 16$ |
| | | 1 | $(\times 32)^*$ |
| | 1 | 0 | $(\times 64)^*$ |
| | | 1 | Invalid (Do not set) |

Note: * Setting optional

(6) Capstan System Digital Filter Control Register (CFIC)

| | | | | | | | | |
|-----------------|---|--------|-------|-------|-------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | DROV | DPHA | DZPON | DZSON | DSG2 | DSG1 | DSG0 |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | R/(W)* | R/(W) | R/W | R/W | R/W | R/W | R/W |

Note: * Only 0 can be written

CFIC is an 8-bit readable/writable register that controls the status of the capstan system digital filter and operating mode. It can be accessed by byte access only. Word access gives unassured results.

Bit 7 is a reserved bit. Writes are disabled. If read is attempted, an undetermined value is read out. CFIC is initialized to H'80 by a reset, and in standby mode and module stop mode.

Bit 7: Reserved

Reads and writes are both disabled.

Bit 6: Capstan System Range Over Flag (CROV)

This flag is set to 1 when the result of a capstan system filter computation exceeds 12 bits in width. To clear this flag, write 0.

Bit 6

| CROV | Description |
|------|---|
| 0 | Indicates that the filter computation result did not exceed 12 bits (Initial value) |
| 1 | Indicates that the filter computation result exceeded 12 bits |

Bit 5: Capstan Phase System Filter Start Bit (CPHA)

Starts or stops filter processing for the capstan phase system.

Bit 5

| CPHA | Description |
|------|--|
| 0 | Phase filter computations are disabled Phase computation result (Y) is not added to Es (see figure 28.39) (Initial value) |
| 1 | Phase filter computations are enabled |

Bit 4: Capstan Phase System Z^{-1} Initialization Bit (CZPON)

Reflects the CZp value on Z^{-1} of the capstan phase system when computation processing of the phase system begins. If 1 was written, it is reflected on the computation, and then cleared to 0. Set this bit after writing data to CZp.

Bit 4

| CZPON | Description |
|--------------|--|
| 0 | CZp value is not reflected on Z^{-1} of the phase system (Initial value) |
| 1 | CZp value is reflected on Z^{-1} of the phase system |

Bit 3: Capstan Speed System Z^{-1} Initialization Bit (CZSON)

Reflects the CZs value on Z^{-1} of the capstan speed system when computation processing of the speed system begins. If 1 was written, it is reflected on the computation, and then cleared to 0. Set this bit after writing data to CZs.

Bit 3

| CZSON | Description |
|--------------|--|
| 0 | CZs value is not reflected on Z^{-1} of the speed system (Initial value) |
| 1 | CZs value is reflected on Z^{-1} of the speed system |

Bits 2 to 0: Capstan System Gain Control Bits (CSG2, CSG1, CSG0)

Control the gain output to CAPPWM.

| Bit 1 | Bit 2 | Bit 0 | Description |
|--------------|--------------|--------------|----------------------------|
| CSG2 | CSG1 | CSG0 | |
| 0 | 0 | 0 | $\times 1$ (Initial value) |
| | | 1 | $\times 2$ |
| | 1 | 0 | $\times 4$ |
| | | 1 | $\times 8$ |
| 1 | 0 | 0 | $\times 16$ |
| | | 1 | $(\times 32)^*$ |
| | 1 | 0 | $(\times 64)^*$ |
| | | 1 | Invalid (Do not set) |

Note: * Setting optional

(7) Digital Filter Control Register (DFUCR)

| | | | | | | | | |
|-----------------|---|---|------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | PTON | CP/DP | CFEPS | DFEPS | CFESS | DFESS |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

DFUCR is an 8-bit readable/writable register which controls the operation of the digital filter. It accepts a byte-access only. If it was word-accessed, operation is not assured.

Bits 7 and 6 are reserved. No write in them is valid. It is initialized to H'00 by a reset, stand-by or module stop.

Bits 7 and 6: Reserved

No read or write is valid. If a read is attempted, an undefined value is read out.

Bit 5: Phase System Computation Result PWM Output Bit (PTON)

Outputs the computation results of only the phase system to PWM. (The computation results of the drum phase system is output to the CAPPWM pin, and that of the capstan phase system is output to the DRMPWM pin.)

Bit 5

| PTON | Description |
|------|--|
| 0 | Outputs the results of ordinary computation of the filter to PWM pin (Initial value) |
| 1 | Outputs the computation results of only the phase system to PWM pin |

Bit 4: PWM Output Selection Bit (CP/DP)

Selects whether the phase system computation results when PTON was set to 1 is output to the drum or capstan. The PWM of the selected side outputs ordinary filter computation results (speed system of MIX).

Bit 4

| CP/DP | Description |
|-------|--|
| 0 | Outputs the drum phase system computation results (CAPPWM) (Initial value) |
| 1 | Outputs the capstan phase system computation results (DRMPWM) |

Bit 3: Capstan Phase System Error Data Transfer Bit (CFEPS)

Transfers the capstan phase system error data to the digital filter when the data write is enforced.

Bit 3

| CFEPS | Description |
|--------------|---|
| 0 | Error data is transferred by DVCFG2 signal latching (Initial value) |
| 1 | Error data is transferred when the data is written |

Bit 2: Drum Phase System Error Data Transfer Bit (DFEPS)

Transfers the drum phase system error data to the digital filter when the data write is enforced.

Bit 2

| DFEPS | Description |
|--------------|---|
| 0 | Error data is transferred by HSW (NHSW) signal latching (Initial value) |
| 1 | Error data is transferred when the data is written |

Bit 1: Capstan Speed System Error Data Transfer Bit (CFESS)

Transfers the capstan speed system error data to the digital filter when the data write is enforced.

Bit 1

| CFESS | Description |
|--------------|--|
| 0 | Error data is transferred by DVCFG signal latching (Initial value) |
| 1 | Error data is transferred when the data is written |

Bit 0: Drum Speed System Error Data Transfer Bit (DFESS)

Transfers the drum speed system error data to the digital filter when the data write is enforced.

Bit 0

| DFESS | Description |
|--------------|--|
| 0 | Error data is transferred by NCDFG signal latching (Initial value) |
| 1 | Error data is transferred when the data is written |

28.11.6 Filter Characteristics

(1) Lag-Lead Filter

A filter required for a servo loop is built in the hardware. This filter uses an IIR (Infinite Impulse Response) type digital filter (another type of the digital filter is FIR, i.e. Finite Impulse Response type). This digital filter circuit implements a lag-lead filter, as shown in figure 28.40.

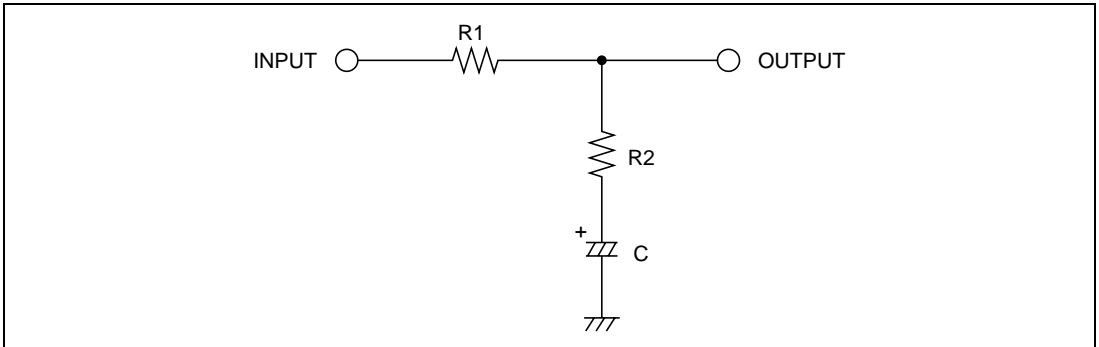


Figure 28.40 Lag-lead Filter

The transfer function $G(S)$ is expressed by the following equation.

$$\text{Transfer function } G(S) = \frac{S}{1 + \frac{S}{2\pi f_2}} \cdot \frac{1}{1 + \frac{S}{2\pi f_1}}$$

$$f_1 = 1/2\pi C (R_1 + R_2)$$

$$f_2 = 1/2\pi C R_2$$

(2) Frequency Characteristics

The computation circuit repeats computation of the function, which is obtained by s-z conversion according to bi-linear approximation of the transfer function on the s-plane.

Figure 28.41 shows the frequency characteristics of the lag-lead filter.

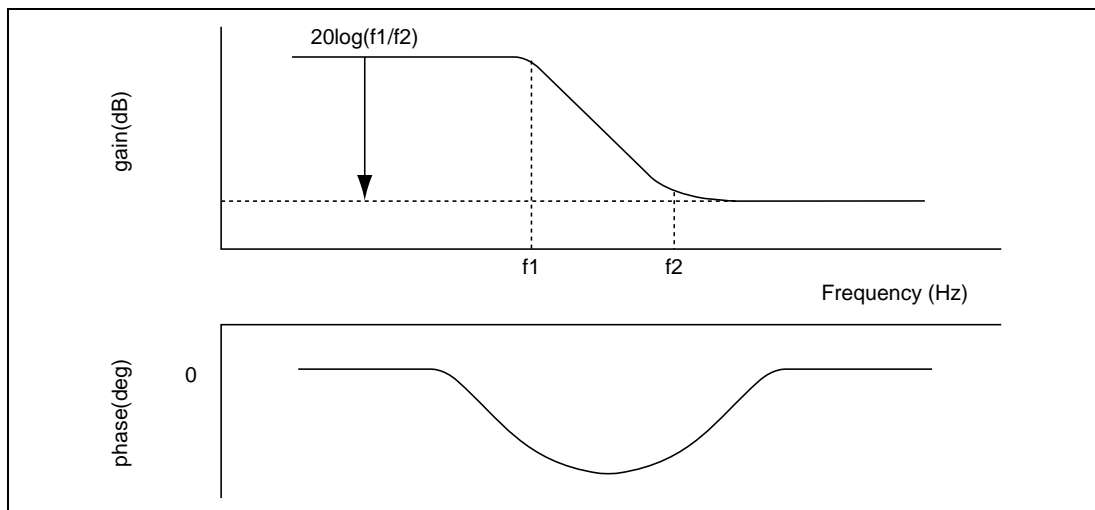


Figure 28.41 Frequency Characteristics of the Lag-Lead Filter

The pulse transfer function $G(Z)$ is obtained by the bi-linear approximation of the transfer function $G(S)$.

In the transfer $G(S)$,

$$S = \frac{2}{T_s} \cdot \frac{1-Z^{-1}}{1+Z^{-1}}$$

Where, assumed that $Z^{-1} = e^{-j\omega T_s}$,

$$G(Z) = G \cdot \frac{2}{T_s} \cdot \frac{1+AZ^{-1}}{1+BZ^{-1}}$$

$$G(Z) = \frac{T_s + \frac{1}{\pi f_2}}{T_s + \frac{1}{\pi f_1}} \quad A = \frac{T_s - \frac{1}{\pi f_2}}{T_s + \frac{1}{\pi f_2}} \quad B = \frac{T_s - \frac{1}{\pi f_1}}{T_s + \frac{1}{\pi f_1}}$$

T_s : Sampling cycle (sec)

28.11.7 Operations in Case of Transient Response

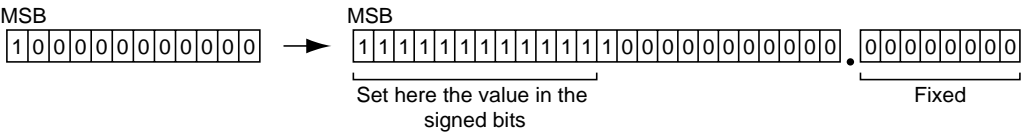
In case of transient response when the motor is activated, the digital filter computation circuit must prevent computation due to a large error. The convergence of the computations becomes slow and servo retraction becomes deteriorating if a large error is input to the filter circuit when it is performing repeated computations. To prevent them from occurring, operate the filter (set constants A and B) after pulling in the speed and phase within a certain range of error, initialize Z^{-1} (set initial values in CZp, CZs, DZp, DZs)(see section 28.11.8, Initialization of Z^{-1}), or use the error data limit function (see section regarding the error detector).

28.11.8 Initialization of Z^{-1}

Z^{-1} can be initialized by its delay initialization register (CZp, CZs, DZp, DZs). Loading to Z^{-1} is performed automatically by bits 4 and 3 of CFIC and DFIC (CZPON, CZSON, DZPON, DZSON). Writing in register is always available, but loading in Z^{-1} is not possible when the digital filter is performing calculation processing in relation to such register. In such a case, loading to Z^{-1} will be done the next time computation begins. Figure 28.42 shows the initialization circuit of Z^{-1} .

The delay initialization register sets 12-bit data. The MSB (bit 11) is a signed bit. Z^{-1} has 24 bits for integers and 8 bits for decimals. Accordingly, the same value as the signed bits should be set in the 13 bits on the MSB side of Z^{-1} , and 0 in the entire decimal section.

Example: Value set for the delay initialization register Value set for Z^{-1}



28.12 Additional V Signal Generator

28.12.1 Overview

The circuit described in this section outputs an additional vertical sync signal to take the place of Vsync in special playback. It is activated at both edges of the HSW signal output by the head-switch timing generator. The head-switch timing generator also outputs a Vpulse signal containing the additional vertical sync pulse itself, and an Mlevel signal that defines the width of the additional vertical sync signal including the equalizing pulses.

The additional V signal is output at a three-level output pin (Vpulse).

Figure 28.43 shows the additional V signal control circuit.

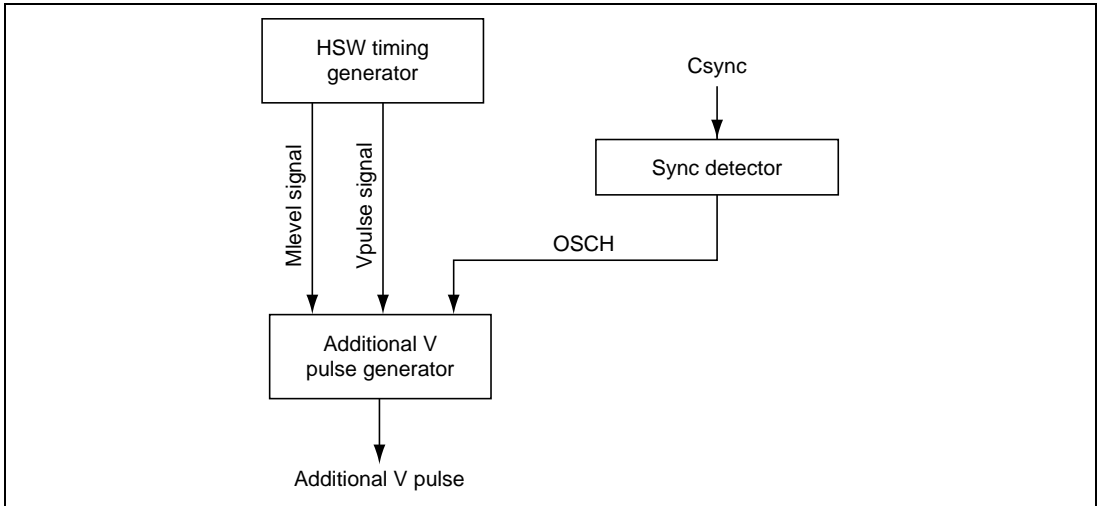


Figure 28.43 Additional V Pulse Control Circuit

(a) HSW timing generator

This circuit generates signals that are synchronized with head switching. It should be programmed to generate the Mlevel and Vpulse signals at edges of the HSW signal (VideoFF). For details, see section 28.4, HSW (Head-switch) Timing Generator.

(b) Sync detector

This circuit detects pulses of the width specified by VTR or HTR from the signal input at the Csyntax pin and generates an internal horizontal sync signal (OSCH). The sync detector has an interpolation function, so OSCH has a regular period even if there are horizontal sync dropouts in the signal received at the pin. For details, see section 28.15, Sync Signal Detector.

28.12.2 Pin Configuration

Table 28.16 summarizes the pin configuration of the additional V signal.

Table 28.16 Pin Configuration

| Name | Abbrev. | I/O | Function |
|------------------------|---------|--------|---|
| Additional V pulse pin | Vpulse | Output | Output of additional V signal synchronized to VideoFF |

28.12.3 Register Configuration

Table 28.17 summarizes the register that controls the additional V signal.

Table 28.17 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|-------------------------------|---------|-----|------|---------------|---------|
| Additional V control register | ADDVR | R/W | Byte | H'E0 | H'FD06F |

28.12.4 Register Description

Additional V Control Register (ADDVR)

| | | | | | | | | |
|-----------------|---|---|---|------|-----|-----|------|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | HMSK | HiZ | CUT | VPON | POL |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | R/W | R/W | R/W | R/W | R/W |

ADDVR is an 8-bit readable/writable register. It is initialized to H'E0 by a reset, and in standby mode.

Bits 7 to 5: Reserved

Writes are disabled. If a read is attempted, an undefined value is read out.

Bit 4: OSCH Mask Bit (HMSK)

Masks the OSCH signal in the additional V pulse.

Bit 4

| HMSK | Description |
|------|----------------------------------|
| 0 | OSCH is added in (Initial value) |
| 1 | OSCH is not added in |

Bit 3: High Impedance Bit (HiZ)

Set to 1 when the intermediate level is generated by an external circuit.

Bit 3

| HiZ | Description |
|-----|---|
| 0 | Vpulse is a three-level output pin (Initial value) |
| 1 | Vpulse is a three-state output pin (high, low, or high-impedance) |

Bits 2 to 0: Additional V Output Control Bit (CUT, VPON, POL)

These bits control the output at the additional V pin.

| Bit 2 | Bit 1 | Bit 0 | Description |
|-------|-------|-------|--|
| CUT | VPON | POL | |
| 0 | 0 | * | Low level (Initial value) |
| | 1 | 0 | Negative polarity (see figure 28.46) |
| | | 1 | Positive polarity (see figure 28.45) |
| 1 | * | 0 | Intermediate level (high impedance if HiZ bit = 1) |
| | | 1 | High level |

Note: * Don't care.

28.12.5 Additional V Pulse Signal

Figure 28.44 shows the additional V pulse signal. The Mlevel and Vpulse signals are generated by the head-switch timing generator. The OSCH signal is combined with these to produce equalizing pulses. The polarity can be selected by the POL bit in the additional V register (ADDVR). The Vpulse pin outputs a low level by a reset, and in standby mode and module stop mode.

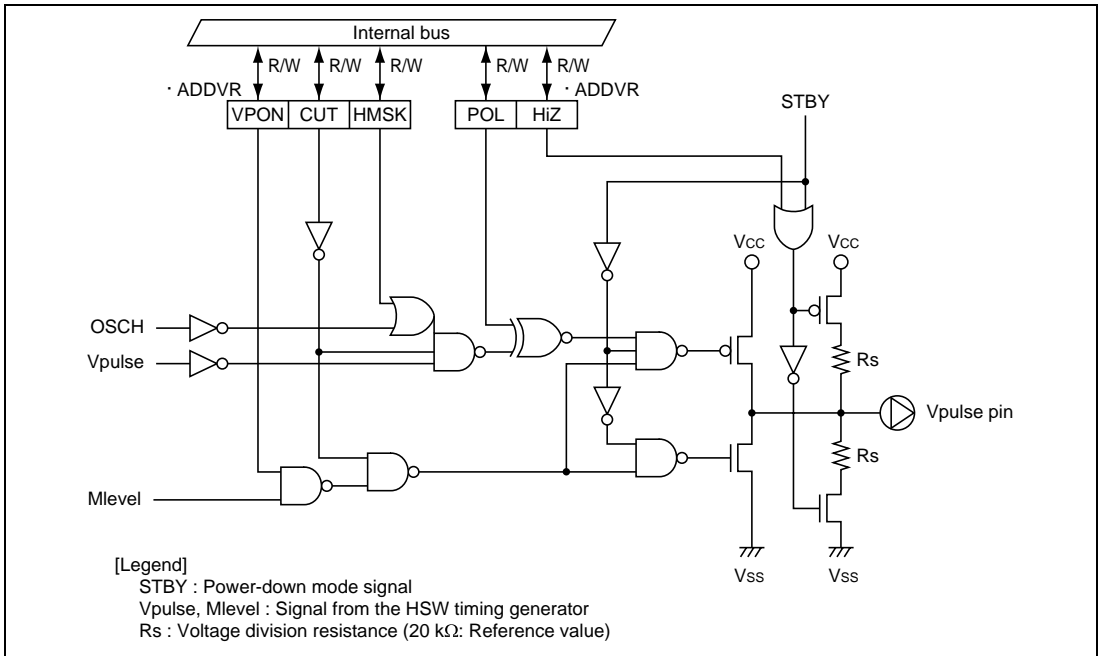


Figure 28.44 Additional V Pin

(a) Additional V pulses when sync signal is not detected

With additional V pulses, the pulse signal (OSCH) detected by the sync detector is superimposed on the Vpulse and Mlevel signals generated by the head-switch timing generator. If there is a lot of noise in the input sync signal (Csync), or a pulse is missing, OSCH will be a complementary pulse, and therefore an H pulse of the period set in HRTR and HPWR will be superimposed. In this case, there may be slight timing drift compared with the normal sync signal, depending on the HRTR and FPWR setting, with resultant discontinuity.

If no sync signal is input, the additional V pulse is generated as a complementary pulse. Set the sync detector registers and activate the sync detector by manipulating the SYCT bit in the sync signal control register (SYNCR). See section 28.15.7, Sync Signal Detector Activation. Figures 28.45 and 28.46 show the additional V pulse timing charts.

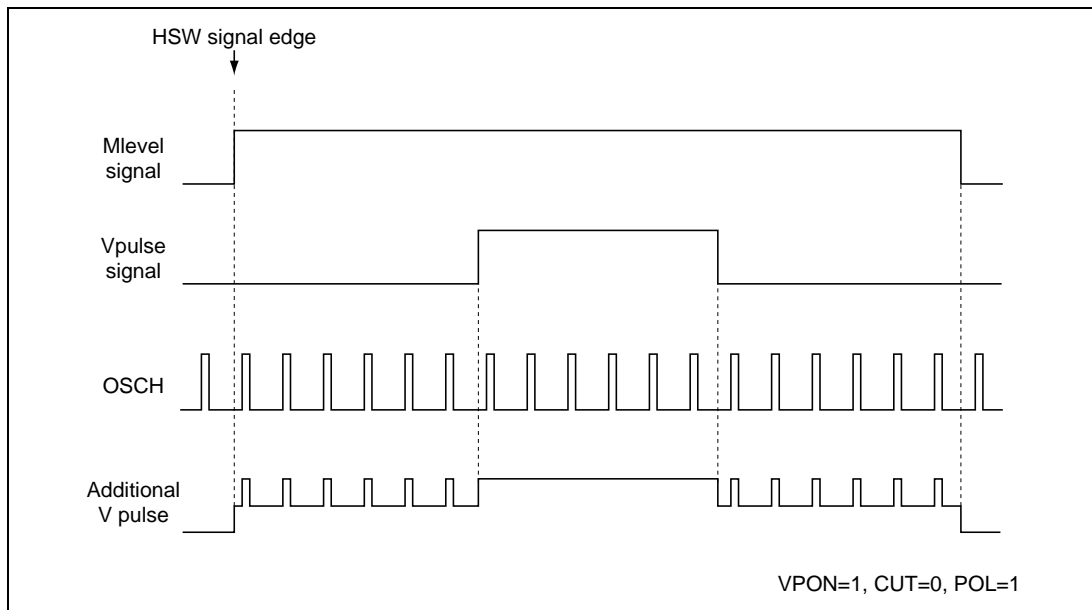


Figure 28.45 Additional V Pulse When Positive Polarity is Specified

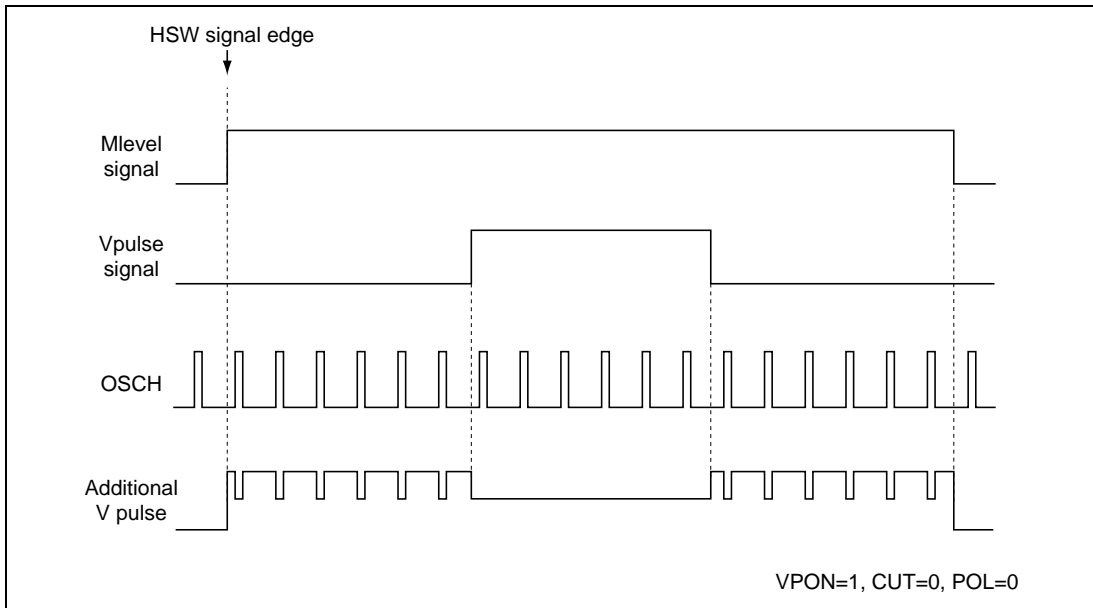


Figure 28.46 Additional V Pulse When Negative Polarity is Specified

28.13 CTL Circuit

28.13.1 Overview

The CTL circuit includes a Schmitt amplifier that amplifies and reshapes the CTL input, then outputs it as the PB-CTL signal to the servo, linear time counter, and other circuits.

The PB-CTL signal is also sent to a duty discriminator in the CTL circuit that detects and records VISS, ASM, and VASS marks. A REC-CTL amplifier is included in the record circuits. Detection and recording whether the CTL pulse pattern is long or short can also be enabled to correspond to the wide-aspect.

The following operating modes can be selected by settings in the CTL mode register:

- Duty discrimination
VISS detect, ASM detect, VASS detect, L/S bit pattern detect
- CTL record
VISS record, ASM record, VASS record, L/S bit pattern detect
- Rewrite
Trapezoid waveform generator

28.13.2 Block Diagram

Figure 28.47 shows a block diagram of the CTL circuit.

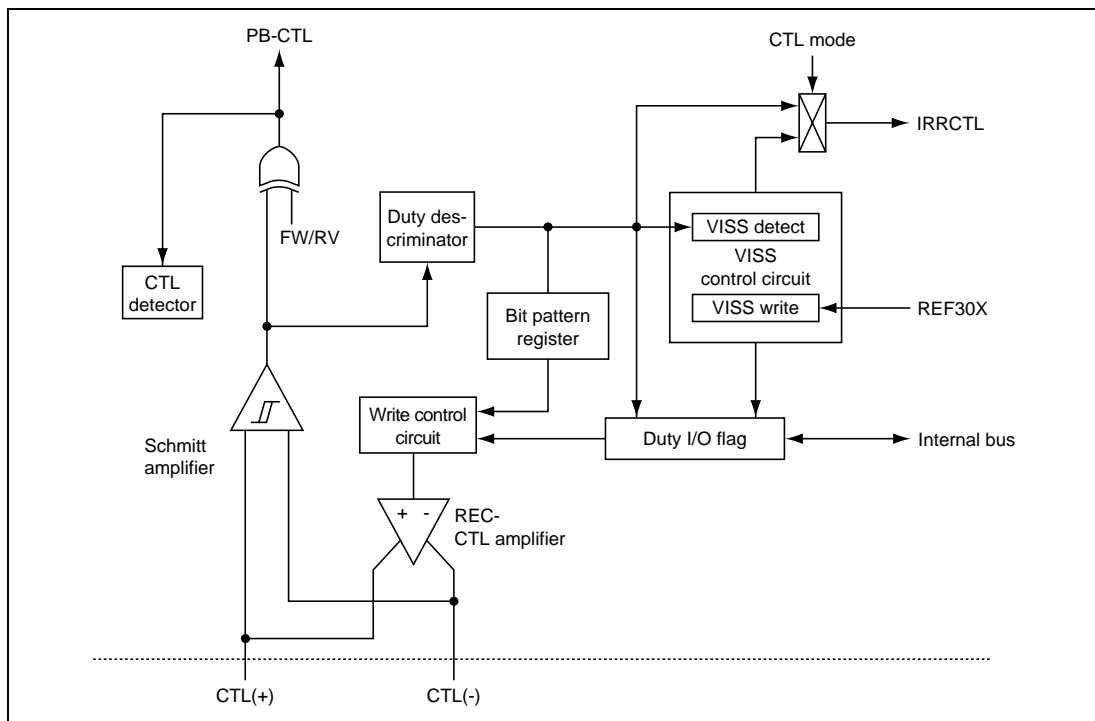


Figure 28.47 Block Diagram of CTL Circuit

28.13.3 Pin Configuration

Table 28.18 summarizes the pin configuration of the CTL circuit.

Table 28.18 Pin Configuration

| Name | Abbrev. | I/O | Function |
|------------------------|------------|--------|--|
| CTL (+) I/O pin | CTL (+) | I/O | CTL signal input/output |
| CTL (–) I/O pin | CTL (–) | I/O | CTL signal input/output |
| CTL Bias input pin | CTL Bias | Input | CTL primary amplifier bias supply |
| CTL Amp (O) output pin | CTLamp (O) | Output | CTL amplifier output |
| CTL SMT (i) input pin | CTLSMT (i) | Input | CTL Schmitt amplifier input |
| CTL FB input pin | CTL FB | Input | CTL amplifier high-range characteristics control |
| CTL REF output pin | CTL REF | Output | CTL amplifier reference voltage output |

28.13.4 Register Configuration

Table 28.19 shows the register configuration of the CTL circuit.

Table 28.19 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|------------------------------|---------|-----|------|---------------|---------|
| CTL control register | CTCR | R/W | Byte | H'30 | H'FD080 |
| CTL mode register | CTLM | R/W | Byte | H'00 | H'FD081 |
| REC-CTL duty data register 1 | RCDR1 | W | Word | H'F000 | H'FD082 |
| REC-CTL duty data register 2 | RCDR2 | W | Word | H'F000 | H'FD084 |
| REC-CTL duty data register 3 | RCDR3 | W | Word | H'F000 | H'FD086 |
| REC-CTL duty data register 4 | RCDR4 | W | Word | H'F000 | H'FD088 |
| REC-CTL duty data register 5 | RCDR5 | W | Word | H'F000 | H'FD08A |
| Duty I/O register | DI/O | R/W | Byte | H'F1 | H'FD08C |
| Bit pattern register | BTPR | R/W | Byte | H'FF | H'FD08D |

28.13.5 Register Descriptions

(1) CTL Control Register (CTCR)

| | | | | | | | | |
|-----------------|-------|------|------|------|-----|------|-------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | NT/PL | FSLC | FSLB | FSLA | CCS | LCTL | UNCTL | SLWM |
| Initial value : | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | R | W |

The CTL control register (CTCR) controls PB-CTL rewrite and sets the slow mode. When a CTL pulse cannot be detected with the input amplifier gain set at the CTL gain control register (CTLGR) in the PB-CTL circuit, bit 1 (UNCTL) of CTCR is set to 1. It is automatically cleared to 0 when a CTL pulse is detected.

CTCR is an 8-bit readable/writable register. However, bit 1 is read-only, and the rest is write-only.

CTCR is initialized to H'30 by a reset, and in standby and module stop mode.

Bit 7: NTSC/PAL Selection Bit (NT/PL)

Selects the period of the rewrite circuit.

Bit 7

| NT/PL | Description |
|-------|---|
| 0 | NTSC mode (frame rate: 30 Hz) (Initial value) |
| 1 | PAL mode (frame rate: 25 Hz) |

Bits 6 to 4: Frequency Selection Bits (FSLA, FSLB, FSLC)

These bits select the operating frequency of the CTL rewrite circuit. They should be set according to fosc.

| Bit 6 | Bit 5 | Bit 4 | Description |
|-------|-------|-------|-------------------------------|
| FSLC | FSLB | FSLA | |
| 0 | 0 | 0 | Reserved (do not set) |
| | | 1 | Reserved (do not set) |
| | 1 | 0 | fosc = 8 MHz |
| | | 1 | fosc = 10 MHz (Initial value) |
| 1 | * | * | Reserved (do not set) |

Note: * Don't care.

Bits 3: Clock Source Selection Bit (CCS)

Selects clock source of CTL.

Bit 3

| CCS | Description |
|-----|--------------------------|
| 0 | ϕ_S (Initial value) |
| 1 | $\phi_S/2$ |

Bit 2: Long CTL Bit (LCTL)

Sets the long CTL detection mode.

Bit 2

| LCTL | Description |
|------|---|
| 0 | Clock source (CCS) operates at the setting value (Initial value) |
| 1 | Clock source (CCS) operates for further 8-division after operating at the setting value |

Bit 1: CTL Undetected Bit (UNCTL)

Indicates the CTL pulse detection status at the CTL input amplifier sensitivity set at the CTL gain control register (CTLGR).

Bit 1

| UNCTL | Description |
|-------|--------------------------|
| 0 | Detected (Initial value) |
| 1 | Undetected |

Bit 0: Mode Selection Bit (SLWM)

Selects CTL mode.

Bit 0

| SLWM | Description |
|------|-----------------------------|
| 0 | Normal mode (Initial value) |
| 1 | Slow mode |

(2) CTL Mode Register (CTLM)

| | | | | | | | | |
|-----------------|-----|-----------------------------|-------|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ASM | REC/ $\overline{\text{PB}}$ | FW/RV | MD4 | MD3 | MD2 | MD1 | MD0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

The CTL mode register (CTLM) is an 8-bit readable/writable register that controls the operating state of the CTL circuit. If 1 is written in bits MD3 and MD2, they will be cleared to 0 one cycle (ϕ) later.

CTLM is initialized to H'00 by a reset, and in standby mode and module stop mode. When CTL is being stopped, only bits 7, 6 and 5 operate.

Note: Do not set any value other than the setting value for each mode (see table 28.20, CTL Mode Functions).

Bits 7 and 6: Record/Playback Mode Bits (ASM, REC/PB)

These bits switch between record and playback. Combined with bits 4 to 0 (MD4 to MD0), they support the VISS, VASS, and ASM mark functions.

| Bit 7 | Bit 6 | Description |
|-------|-----------------------------|-------------------------------|
| ASM | REC/ $\overline{\text{PB}}$ | |
| 0 | 0 | Playback mode (Initial value) |
| | 1 | Record mode |
| 1 | 0 | Assemble mode |
| | 1 | Invalid (do not set) |

Bit 5: Direction Bit (FW/RV)

Selects the direction in playback. Clear this bit to 0 during record. Figure 28.48 shows the PB-CTL signal in forward and reverse.

| Bit 5 | Description |
|-------|-------------------------|
| FW/RV | |
| 0 | FORWARD (Initial value) |
| 1 | REVERSE |

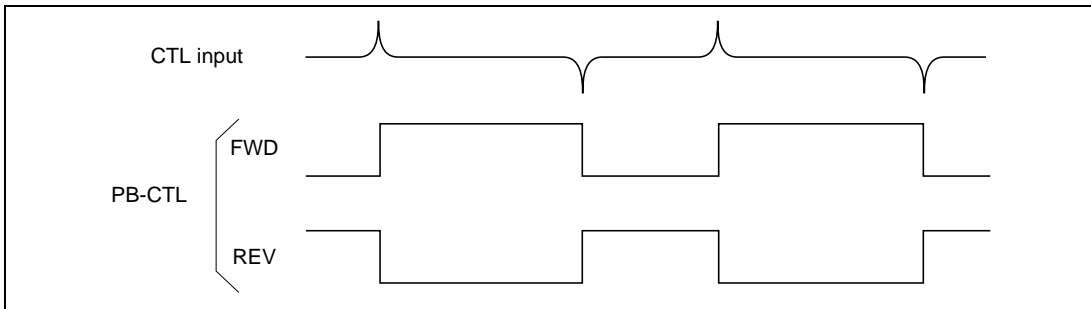


Figure 28.48 Internal PB-CTL Signal in Forward and Reverse

Bits 4 to 0: CTL Mode Selection Bits (MD4 to MD0)

These bits select the detect, record, and rewrite modes for VISS, VASS, and ASM marks. If 1 is written in bits MD3 and MD2, they will be cleared to 0 one cycle (ϕ) later.

The 5 bits from MD4 to MD0 are used in combination with bits 7 and 6 (ASM and REC/ $\overline{\text{PB}}$).

Table 28.20 describes the modes.

Table 28.20 CTL Mode Functions

| Bit | | | | | | | | | Mode | Description |
|-----|------------------------------------|-----------|-----|-----|-----|-----|-----|------------------------------------|---|-------------|
| ASM | REC / $\overline{\text{PB}}$ | FW/ RV | MD4 | MD3 | MD2 | MD1 | MD0 | | | |
| 0 | 0 | 0/1 | 0 | 0 | 0 | 0 | 0 | VASS detect (duty detect) | PB-CTL duty discrimination (Initial value) <ul style="list-style-type: none">Duty I/O flag is set to 1 if duty \geq 44% is detectedDuty I/O flag is cleared to 0 if duty $<$ 44% is detectedInterrupt request is generated when one CTL pulse has been detected | |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | VASS record | <ul style="list-style-type: none">If 0 is written in the duty I/O flag, REC-CTL is generated and recorded with the duty cycle set by register RCDR2 or RCDR3If 1 is written in the duty I/O flag, REC-CTL is generated and recorded with the duty cycle set by register RCDR4 or RCDR5 | |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | VASS rewrite | Same as above (VASS record; trapezoid waveform circuit operation) | |

| Bit | | | | | | | | Mode | Description |
|-----|---------------------------------|-----------|-----|-----|-----|-----|-----|-------------------------------------|---|
| ASM | REC / $\overline{\text{PB}}$ | FW/ RV | MD4 | MD3 | MD2 | MD1 | MD0 | | |
| 0 | 0 | 0/1 | 0 | 1 | 0 | 0 | 1 | VISS detect (index detect) | <ul style="list-style-type: none">The duty I/O flag is set to 1 at the point of write access to register CTLMThe 1 pulses recognized by the duty discrimination circuit are counted in the VISS control circuitThe duty I/O flag is cleared to 0, indicating VISS detection, when the value set at VCTR register is repeatedly detectedAn interrupt request is generated when VISS is detected |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | VISS record (index record) | <ul style="list-style-type: none">64 pulse data with 0 pulse data at both edges are written (index record)The index bit string is written through the duty I/O flagAn interrupt request is generated at the end of VISS recording |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | VISS rewrite | Same as above (VISS record; trapezoid waveform circuit operation) |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | VISS initialize | VISS write is forcibly aborted |
| 1 | 0 | 0/1 | 0 | 0 | 0 | 0 | 0 | ASM mark detect | <p>ASM mark detection</p> <ul style="list-style-type: none">The duty I/O flag is cleared to 0 when PB-CTL duty $\geq 66\%$ is detectedAn interrupt request is generated when an ASM mark is detected |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | ASM mark record | <ul style="list-style-type: none">An ASM mark is recorded by writing 0 in the duty I/O flagAn interrupt is requested for every one CTL pulseREC-CTL is generated and recorded with the duty cycle set by register RCDR3 |

(3) REC-CTL Duty Data Register 1 (RCDR1)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMT1B | CMT1A | CMT19 | CMT18 | CMT17 | CMT16 | CMT15 | CMT14 | CMT13 | CMT12 | CMT11 | CMT10 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

RCDR1 is a register that sets the REC-CTL rising timing. This setting is valid only for recording and rewriting, and is not used in detection.

RCDR1 is a 12-bit write-only register, and can be accessed by word access only. Byte access gives unassured results. If read is attempted, an undetermined value is read out. Bits 15 to 12 are reserved and are not affected by write access.

RCDR1 is initialized to H'F000 by a reset, and in standby mode, module stop mode and CTL stop mode.

The value to set in RCDR1 can be calculated from the transition timing T1 and the servo clock frequency ϕ_s by the equation given below. See figure 28.60, REC-CTL Signal Generation Timing. Any transition timing can be set. However, the timing should be selected with attention to playback tracking compensation and the latch timing for phase control.

$$RCDR1 = T1 \times \phi_s / 64$$

ϕ_s is the servo clock frequency ($= f_{osc}/2$) in Hz, and T1 is the set timing (s).

Note: 0 cannot be set to RCDR1. Set a value 1 or above.

(4) REC-CTL Duty Data Register 2 (RCDR2)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMT2B | CMT2A | CMT29 | CMT28 | CMT27 | CMT26 | CMT25 | CMT24 | CMT23 | CMT22 | CMT21 | CMT20 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

RCDR2 is a register that sets 1 pulse (short) falling timing of REC-CTL at recording and rewriting, and detects long/short pulses at detecting.

RCDR2 is a 12-bit write-only register, and can be accessed by word access only. Byte access gives unassured results. If read is attempted, an undetermined value is read out. Bits 15 to 12 are reserved and are not affected by write access.

RCDR2 is initialized to H'F000 by a reset, and in standby mode, module stop mode, and CTL stop mode.

At recording, the value to set in RCDR2 can be calculated from the transition timing T2 and the servo clock frequency ϕ_s by the equation given below, and the set value should be 25% of the duty obtained by the equation. See figure 28.60, REC-CTL Signal Generation Timing.

$$\text{RCDR2} = T2 \times \phi_s / 64$$

ϕ_s is the servo clock frequency ($= f_{\text{osc}}/2$) in Hz, and T2 is the set timing (s).

At bit pattern detection, set the 1 pulse long/short threshold value at FWD. See figure 28.56, Duty Discriminator.

$$\text{RCDR2} = T2' \times \phi_s / 64$$

ϕ_s is the servo clock frequency ($= f_{\text{osc}}/2$) in Hz, and T2' is the 1 pulse long/short threshold value at FWD (s).

(5) REC-CTL Duty Data Register 3 (RCDR3)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMT3B | CMT3A | CMT39 | CMT38 | CMT37 | CMT36 | CMT35 | CMT34 | CMT33 | CMT32 | CMT31 | CMT30 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

RCDR3 is a register that sets 1 pulse (long) and assemble mark falling timing of REC-CTL at recording and rewriting, and detects long/short pulses at detecting.

RCDR3 is a 12-bit write-only register, and can be accessed by word access only. Byte access gives unassured results. If read is attempted, an undetermined value is read out. Bits 15 to 12 are reserved and are not affected by write access.

RCDR3 is initialized to H'F000 by a reset, and in standby mode, module stop mode, and CTL stop mode.

At recording, the value to set in RCDR3 can be calculated from the transition timing T_3 and the servo clock frequency ϕ_s by the equation given below. The set value should be 30% of the duty when the RCDR3 is used for REC-CTL 1 pulse (long), and 67 to 70% when used for assemble mark. The set value must not exceed the value of REF30X. See figure 28.60, REC-CTL Signal Generation Timing.

$$\text{RCDR3} = T_3 \times \phi_s / 64$$

ϕ_s is the servo clock frequency ($= f_{\text{osc}}/2$) in Hz, and T_3 is the set timing (s).

At bit pattern detection, set the 0 pulse long/short threshold value at FWD. See figure 28.56, Duty Discriminator.

$$\text{RCDR3} = T_3' \times \phi_s / 64$$

ϕ_s is the servo clock frequency ($= f_{\text{osc}}/2$) in Hz, and T_3' is the 0 pulse long/short threshold value at FWD (s).

(6) REC-CTL Duty Data Register 4 (RCDR4)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMT4B | CMT4A | CMT49 | CMT48 | CMT47 | CMT46 | CMT45 | CMT44 | CMT43 | CMT42 | CMT41 | CMT40 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

RCDR4 sets the timing of falling edge of the 0 pulse (short) of REC-CTL in record or rewrite mode. In detection mode, it is used to detect the long/short pulse.

RCDR4 is a 12-bit write-only register. It accepts only a word-access. If a byte access is attempted, operation is not assured. If a read is attempted, an undefined value is read out. Bits 15 to 12 are reserved, and no write in them is valid.

It is initialized to H'F000 by a reset, stand-by, module stop or CTL stop.

In record mode, set a value with the 57.5% duty cycle obtained from the transition timing T4 corresponding to the servo clock frequency ϕ s according to the following equation. See figure 28.60, REC-CTL Signal Generation Timing.

$$RCDR4 = T4 \times \phi / 64$$

ϕ is the servo clock frequency ($= f_{osc}/2$) in Hz, and T4 is the set timing (s).

At bit pattern detection, set the 0 pulse long/short threshold value at REV. See figure 28.56, Duty Discriminator.

$$RCDR4 = H'FFF - (T4' \times \phi / 80)$$

ϕ s is the servo clock frequency ($= f_{osc}/2$) in Hz, and T4' is the 0 pulse long/short threshold value at REV (s).

(7) REC-CTL Duty Data Register 5 (RCDR5)

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMT5B | CMT5A | CMT59 | CMT58 | CMT57 | CMT56 | CMT55 | CMT54 | CMT53 | CMT52 | CMT51 | CMT50 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

RCDR5 sets the timing of falling edge of the 0 pulse (long) of REC-CTL in record or rewrite mode. In detection mode, it is used to detect the long/short pulse.

RCDR5 is a 12-bit write-only register. It accepts only a word-access. If a byte access is attempted, operation is not assured. If a read is attempted, an undefined value is read out. Bits 15 to 12 are reserved, and no write in them is valid.

It is initialized to H'F000 by a reset, stand-by, module stop or CTL stop.

In record mode, set a value with the 62.5% duty cycle obtained from the transition timing T5 corresponding to the servo clock frequency ϕ s according to the following equation. See figure 28.60, REC-CTL Signal Generation Timing.

$$\text{RCDR5} = T5 \times \phi \text{ s}/64$$

ϕ is the servo clock frequency ($= f_{\text{osc}}/2$) in Hz, and T5 is the set timing (s).

At bit pattern detection, set the 1 pulse long/short threshold value at REV. See figure 28.56, Duty Discriminator.

$$\text{RCDR5} = \text{H'FFF} - (T5' \times \phi \text{ s}/80)$$

ϕ s is the servo clock frequency ($= f_{\text{osc}}/2$) in Hz, and T5' is the 1 pulse long/short threshold value at REV (s).

(8) Duty I/O Register (DI/O)

| | | | | | | | | |
|-----------------|-------|-------|-------|---|------|-----|--------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | VCTR2 | VCTR1 | VCTR0 | — | BPON | BPS | BPF | DI/O |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| R/W : | W | W | W | — | W | W | R/(W)* | R/W |

Note: * Only 0 can be written.

The duty I/O register is an 8-bit register that confirms and determines the operating status of the CTL circuit.

It is initialized to H'F1 by a reset, and in standby mode, module stop mode, and CTL stop mode.

Bits 7, 6, and 5: VISS Interrupt Setting Bits (VCTR2, VCTR1, VCTR0)

Combination of VCTR2, VCTR1 and VCTR0 sets number of 1 pulse detection in VISS detection mode. Detecting the set number of pulse detection is considered as VISS detection, and an interrupt request is generated.

Note: When changing the detection pulse number during VISS detection, initialize VISS first, then resume the VISS detection setting.

| Bit 7 | Bit 6 | Bit 5 | |
|-------|-------|-------|---------------------------------|
| VCTR2 | VCTR1 | VCTR0 | Number of 1-pulse for Detection |
| 0 | 0 | 0 | 2 |
| | | 1 | 4 (SYNC mark) |
| | 1 | 0 | 6 |
| | | 1 | 8 (mark A, short) |
| 1 | 0 | 0 | 12 (mark A, long) |
| | | 1 | 16 |
| | 1 | 0 | 24 (mark B) |
| | | 1 | 32 |

Bit 4: Reserved

Writes are disabled. When read, undefined values are obtained.

Bit 3: Bit Pattern Detection ON/OFF Bit (BPON)

Determines ON or OFF of bit pattern detection.

Note: When writing 1 to the BPON bit, be sure to set appropriate data to RCDR2 to RCDR5 beforehand.

Bit 3

| BPON | Description |
|------|---|
| 0 | Bit pattern detection OFF (Initial value) |
| 1 | Bit pattern detection ON |

Bit 2: Bit Pattern Detection Start Bit (BPS)

Starts 8-bit bit pattern detection. When 1 is written to this bit, it returns to 0 after one cycle. Writing 0 to this bit does not affect operation.

Bit 2

| BPS | Description |
|-----|------------------------------------|
| 0 | Normal status (Initial value) |
| 1 | Starts 8-bit bit pattern detection |

Bit 1: Bit Pattern Detection Flag (BPF)

Sets the flag every time 8-bit PB-CTL is detected in PB or ASM mode. To clear the flag, write 0 after reading 1.

Bit 1

| BPF | Description |
|-----|---|
| 0 | Bit pattern (8-bit) is not detected (Initial value) |
| 1 | Bit pattern (8-bit) is detected |

Bit 0: Duty I/O Register (DI/O)

This flag has different functions for record and playback.

In VISS detect mode, VASS detect mode, and ASM mark detect mode, this flag indicates the detection result.

In VISS record or rewrite mode, this flag controls the write control circuit so as to write an index code, operating according to a control signal from the VISS control circuit.

In VASS record or rewrite mode and ASM mark record mode, this flag is used for write control, one CTL pulse at a time.

This bit can always be written to, but this does not affect the write control circuit in modes other than VISS record or rewrite, and ASM record.

VISS Detect Mode and VASS Detect Mode:

The duty I/O flag indicates the result of duty discrimination. The duty I/O flag is 1 when the duty cycle of the PB-CTL signal is equal to or above 44% (a 0 pulse in the CTL signal). The duty I/O flag is 0 when the duty cycle of the PB-CTL signal is below 43% (a 1 pulse in the CTL signal).

ASM Mark Detect Mode:

The duty I/O flag indicates the result of duty discrimination. The duty I/O flag is 0 when the duty cycle of the PB-CTL signal is equal to or above 66% (when an ASM mark is detected).

The duty I/O flag is 1 when the duty cycle of the PB-CTL signal is below 65% (when an ASM mark is not detected).

VISS Record Mode and VISS Rewrite Mode:

The duty I/O flag operates according to a control signal from the VISS control circuit, and controls the write control circuit so as to write an index code. The write timing is set in the REC-CTL duty data registers (RCDR1 to RCDR5). For VISS recording, registers RCDR1 to RCDR5 are set with reference to REF30X. For VISS rewrite, registers RCDR2 to RCDR5 are set with reference to the low-to-high transition of the previously recorded CTL signal, and the write is carried out through the trapezoid waveform generator.

Set the duty timing for a 1 pulse (short) in RCDR2, for a 1 pulse (long) in RCDR3, for a 0 pulse (short) in RCDR4, and for a 0 pulse (long) in RCDR5.

While an index code is being written, the value of the bit being written can be read by reading the duty I/O flag. If the CTL signal currently being written is a 0 pulse, the duty I/O flag will read 1. If the CTL signal currently being written is a 1 pulse, the duty I/O flag will read 0.

VASS Record Mode and VASS Rewrite Mode:

The duty I/O flag is used for write control, one CTL pulse at a time. The write timing is set in the REC-CTL duty data registers (RCDR1 to RCDR5). For VASS recording, registers RCDR1 to RCDR5 are set with reference to REF30X. For VASS rewrite, registers RCDR2 to RCDR5 are set with reference to the low-to-high transition of the previously recorded CTL signal, and the write is carried out through the trapezoid waveform generator.

Set the duty timing for a 1 pulse (short) in RCDR2, for a 1 pulse (long) in RCDR3, for a 0 pulse (short) in RCDR4, and for a 0 pulse (long) in RCDR5.

If 0 is written in the duty I/O flag, a CTL pulse will be written with a duty cycle set in RCDR2 and RCDR3, referenced to the immediately following REF30X. If 1 is written in the duty I/O flag, a CTL pulse will be written with a duty cycle set in RCDR4 and RCDR5, referenced to the immediately following REF30X.

ASM Record Mode:

The duty I/O flag is used for write control, one CTL pulse at a time. The write timing is set in the REC-CTL duty data registers (RCDR1 and RCDR3). If 0 is written in the duty I/O flag, a CTL pulse will be written with a duty cycle of 67% to 70% as set in RCDR3, referenced to the immediately following REF30X.

(9) Bit Pattern Register (BTPR)

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | LSP7 | LSP6 | LSP5 | LSP4 | LSP3 | LSP2 | LSP1 | LSP0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R/W* | R/W* | R/W* | R/W* | R/W* | R/W* | R/W* | R/W* |

Note: * Write is prohibited when bit pattern detection is selected.

The bit pattern register (BTPR) is an 8-bit shift register which detects and records the bit pattern of the CTL pulses. If a CTL pulse is detected in PB or ASM mode, the register is shifted leftward at the rising edge of PB-CTL, and reflects the determined result of long/short on bit 0 (long pulse = 1, short pulse = 0).

If the BPON bit is set to 1 in PB mode, the register starts detection of bit pattern immediately after the CTL pulse. To exit the bit pattern detection, set the BPON bit at 0.

If 1 was written in the BPS bit when the bit pattern is being detected, the BPF bit is set at 1 when an 8-bit bit pattern was detected. If continuous detection of 8-bits is required, write 0 in the BPF bit, and then write 1 in the BPS bit.

At the time of VISS detection, the bit pattern detection is disabled. Set the BPON bit to 0 at the time of VISS detection.

In REC mode, the register records the long/short in the bit pattern set in BTPR. The pulse in record mode is determined always by bit 7 (LSP7) of BTPR. BTPR records one pulse, shifts leftward, and stores the data of bit 7 to bit 0.

BTPR is initialized to H'FF by a reset, stand-by, module stop, or CTL stop.

28.13.6 Operation

(a) CTL circuit operation

As shown in figure 28.49, the CTL discrimination/record circuit is composed of a 16-bit up/down counter and 12-bit registers ($\times 5$).

In playback (PB) mode, the 16-bit up/down counter counts on a $\phi s/4$ clock when the PB-CTL pulse is high, and on a $\phi s/5$ clock when low. In record (REC) or slow mode, this counter counts up on a $\phi s/8$ clock when the pulse is high, and on a $\phi s/4$ clock when low.

This counter always counts up in record and slow modes.

In playback or slow mode, it is cleared on the rise of a PB-CTL signal. In record mode, it is cleared on the rise of an REF30X signal.

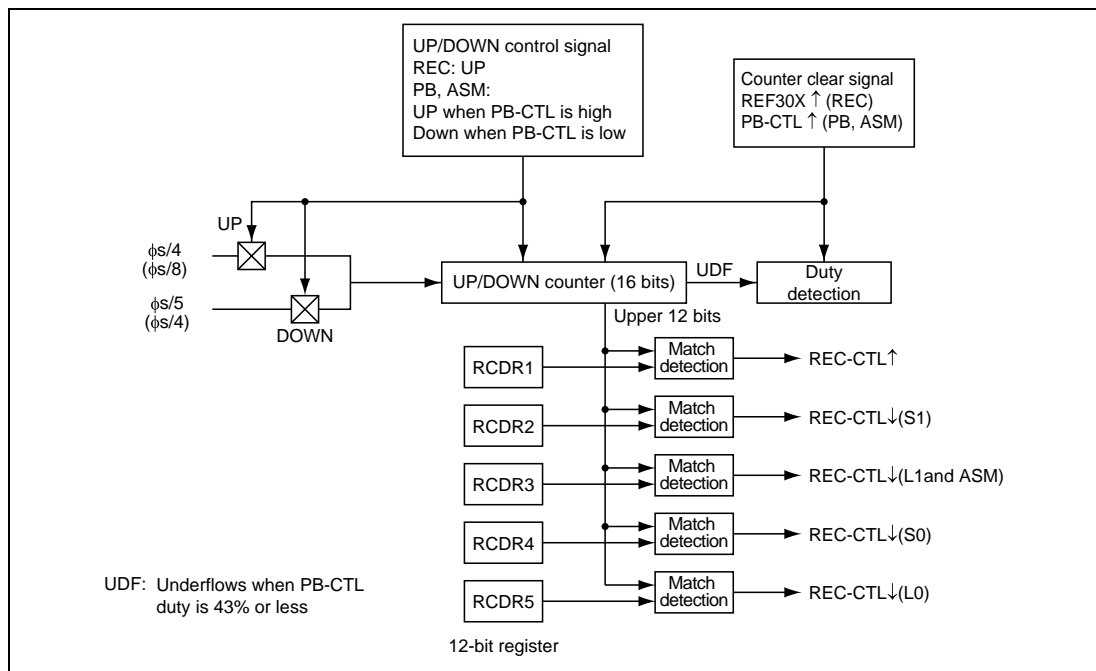


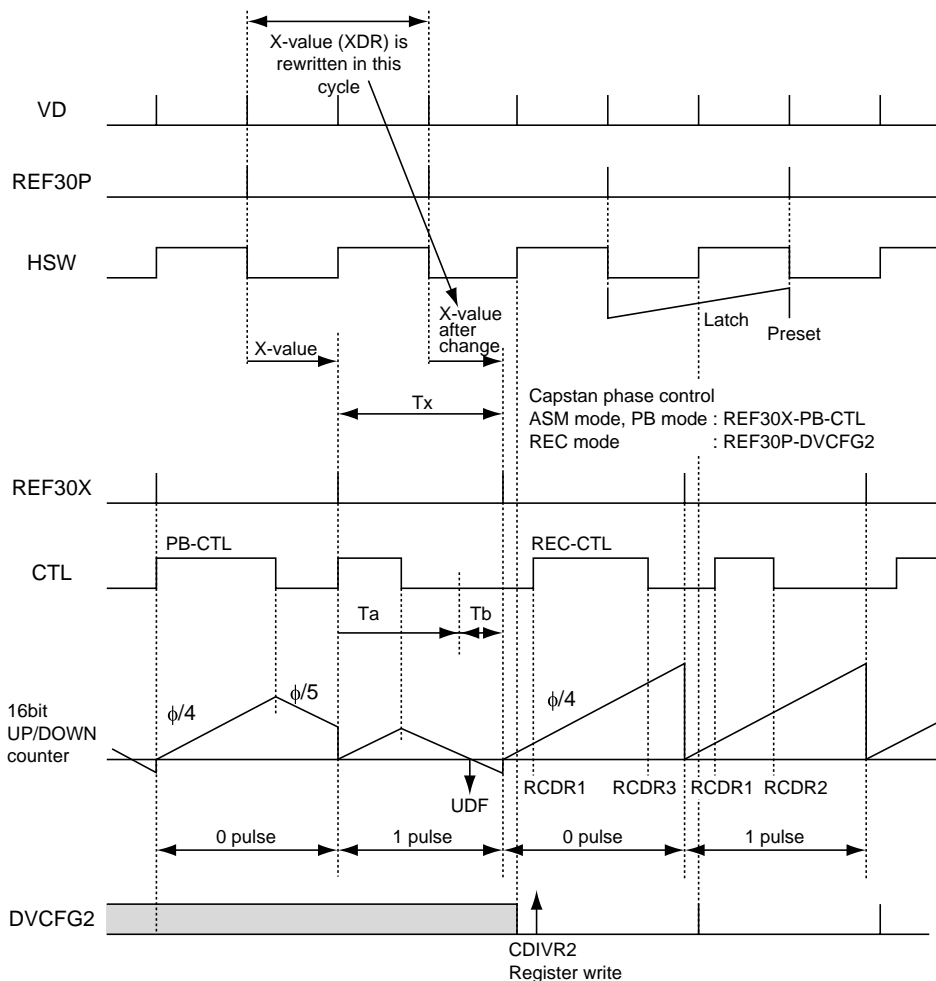
Figure 28.49 CTL Discrimination/Record Circuit

(b) CTL mode register (CTLM) switchover timing

CTLM is enabled immediately after data is written to the register. Care must be taken with changes in the operating state.

Capstan phase control is performed by the VD sync REF30X (X-value + tracking value) and PB-CTL in ASM mode, and by the REF30X or CREF and CFG division signal (DVCFG2) in REC mode. If the CAPREF30 signal to be used for capstan phase control is always generated by XDR, the value of XDR must be overwritten when switching between PB and REC modes. Figures 28.50 and 28.51 show examples of switchover timing of CTLM and XDR.

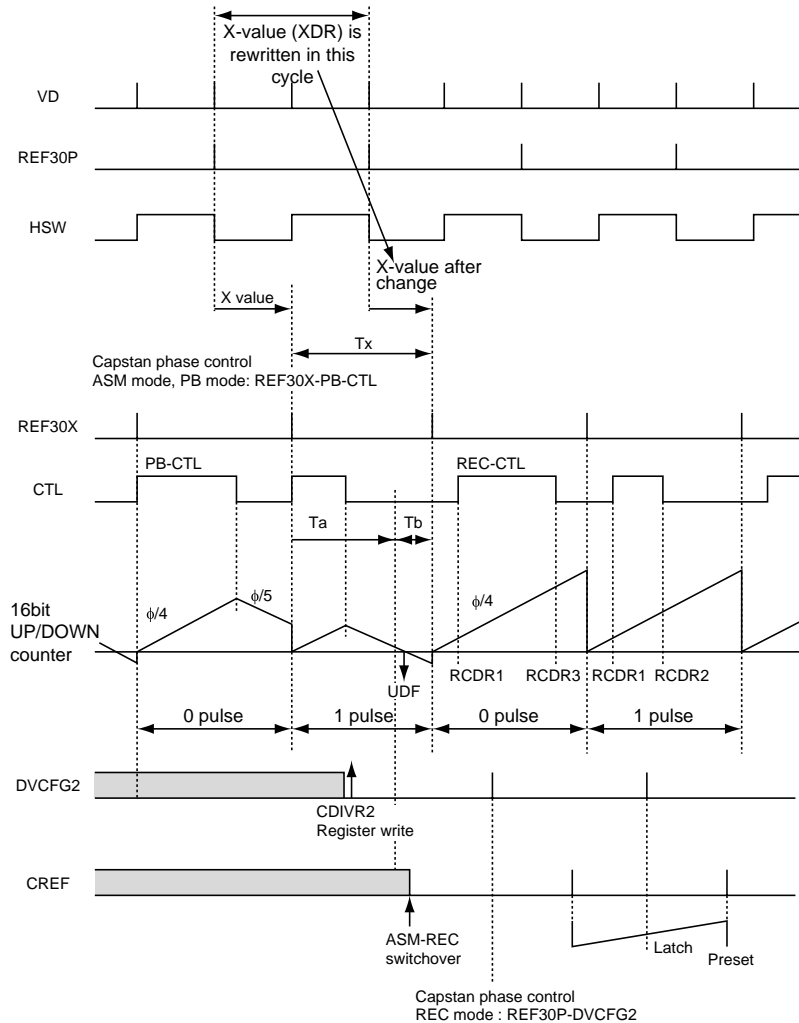
The X-value is updated by REF30P. Modification of XDR must be performed before REF30P in the cycle in which the X-value is changed.



T_a is the interval calculated from RDCR3.
 T_b is the interval in which switchover is performed from ASM mode to REC mode.
 T_x is the cycle in which the REF30X period is shortened due to the change of XDR.

Figure 28.50 Example of CTLM Switchover Timing
 (When Phase Control is Performed by REF30P and DVCFG2 in REC Mode)

The X-value is updated by REF30P. Modification of XDR must be performed before REF30P in the cycle in which the X-value is changed.



T_a is the interval calculated from RCDR3.
 T_b is the interval in which switchover is performed from ASM mode to REC mode.
 T_x is the cycle in which the REF30X period is shortened due to the change of XDR.

With CREF and DVCFG2 phase alignment, the frequency need not be 25 Hz or 30 Hz.

Figure 28.51 Example of CTLM Switchover Timing
(When Phase Control is Performed by CREF and DVCFG2 in REC Mode)

28.13.7 CTL Input Section

The CTL input section consists of an input amplifier whose gain can be controlled by register setting and a Schmitt amplifier. Figure 28.52 shows a block diagram of the CTL input section. A trivial CTL pulse signal is received from the CTL head, amplified by the input amplifier, reshaped into a square wave by the Schmitt amplifier, and sent to the servo circuits and timer L as the PB-CTL signal. Control the CTL input amplifier gain by bits 3 to 0 in the CTL gain control register (CTLGR) of the servo port.

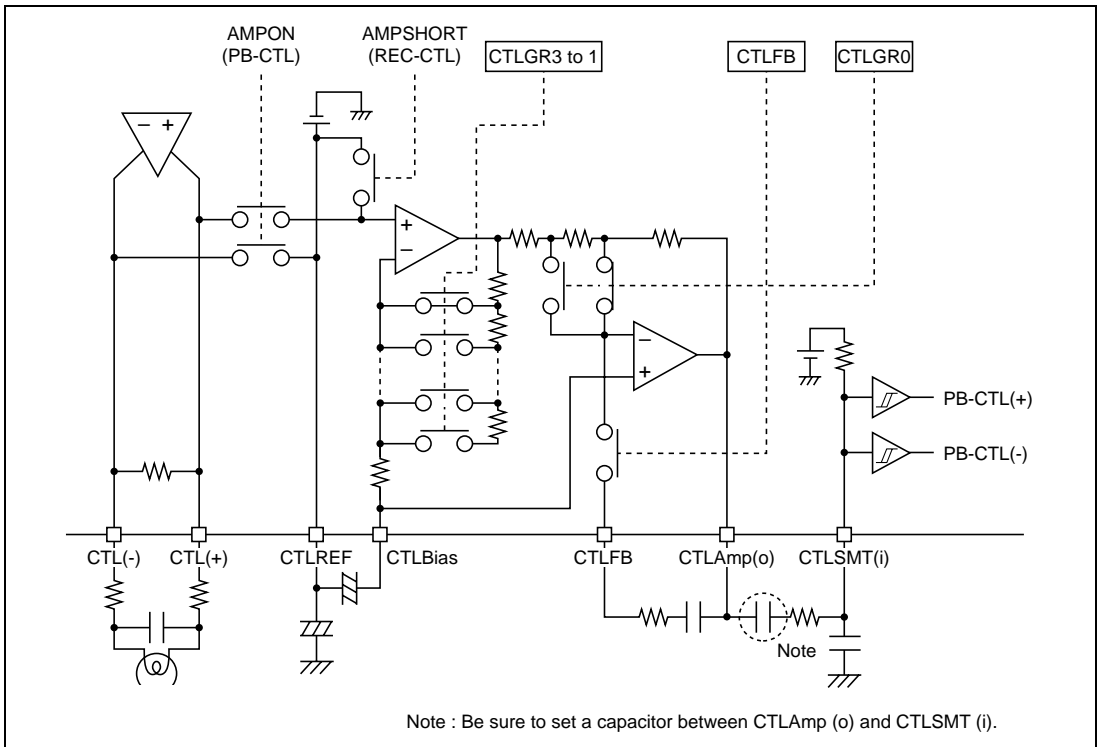


Figure 28.52 Block Diagram of CTL Input Section

(1) CTL Detector

If the CTL detector fails to detect a CTL pulse, it sets bit 1 of the CTL control register (CTCR) to high indicating that the pulse has not been detected. If a CTL pulse is detected after that, the bit is automatically cleared to 0. Duration used for determining detection or non-detection of the pulse depends on magnitude of phase shift of the last detected pulse from the reference phase (phase difference between REF30 and CTL signal). Typically, detection or non-detection is determined within 3 to 4 cycles of the reference period.

If settings of the CTL gain control register are maintained in a table format, you can refer to it when the CTL detector failed to detect CTL pulses. From the table, you can control input sensitivity of the CTL according to the state of the UNCTL bit, thereby selecting an optimum CTL amplifier gain depending on the state of the pulse recorded.

Figure 28.53 illustrates the concept of gain control for detecting the CTL input pulse.

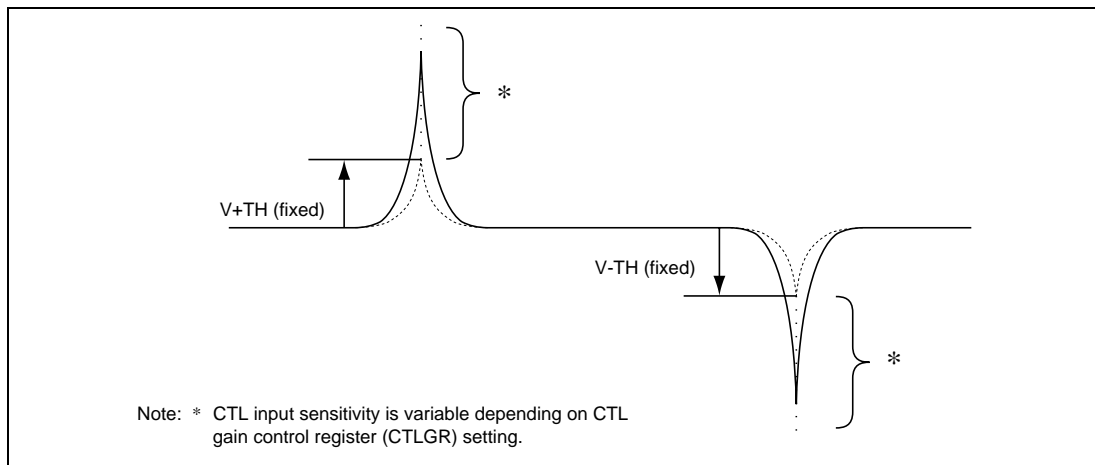


Figure 28.53 CTL Input Pulse Gain Control

(2) PB-CTL Waveform Shaper in Slow Mode Operation

If bit 0 in the CTL control register (CTCR) is set to slow mode, slow reset function is activated. In slow mode, if the falling edge is not detected within the specified time from rising edge detection, PB-CTL is forcibly shut down (slow reset).

The time T_{FS} (s) until the signal falls is the following interval after the rising edge of the internal CTL signal is detected:

$$T_{FS} = 16384 \times 4\phi s \quad (\phi s = f_{OSC}/2)$$

When $f_{OSC} = 10 \text{ MHz}$, $T_{FS} = 13.1 \text{ ms}$.

Figure 28.54 shows the PB-CTL waveform in slow mode.

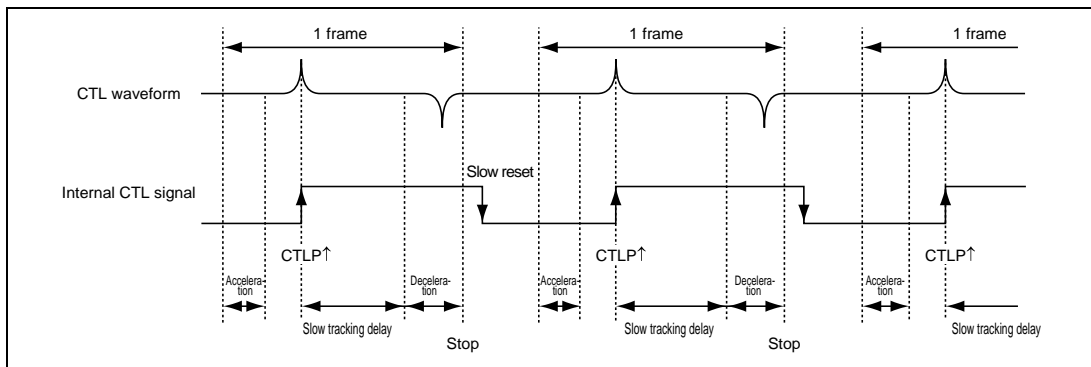


Figure 28.54 PB-CTL Waveform in Slow Mode Operation

28.13.8 Duty Discriminator

The duty discriminator circuit measures the period of the control signal recorded on the tape (PB-CTL signal) and discriminates its duty cycle. In VISS or VASS detection, the duty I/O flag is set or cleared according to the result of duty discrimination. The duty I/O flag is set to 1 when the duty cycle of the PB-CTL signal is equal to or above 44%, and is cleared to 0 when the duty cycle is below 43%.

In ASM detection, an ASM mark is recognized (and the duty I/O flag is cleared to 0) when the duty cycle is equal to or above 66%. When the duty cycle is below 65%, no ASM mark is recognized and the duty I/O flag is set to 1.

The detection direction can be switched between forward and reverse by bit 5 (FW/RV) in the CTL mode register.

A long or short pulse can be detected by comparing the REC-CTL duty data register (RCDR2 to RCDR5) and UP/DOWN counter. Long or short pulse is discriminated at PB-CTL signal falling. Discrimination result is stored in bit 0 (LSP0) of the bit pattern register (BTPR). At the same time, BTPR is shifted to the left. LSP0 indicates 0 when a short pulse is detected, and 1 when a long pulse is detected.

Set the threshold value of a long/short pulse in RCDR2 to RCDR5. See (4), Detection of the Long/Short Pulse.

Figure 28.55 shows the duty cycle of the PB-CTL signal.

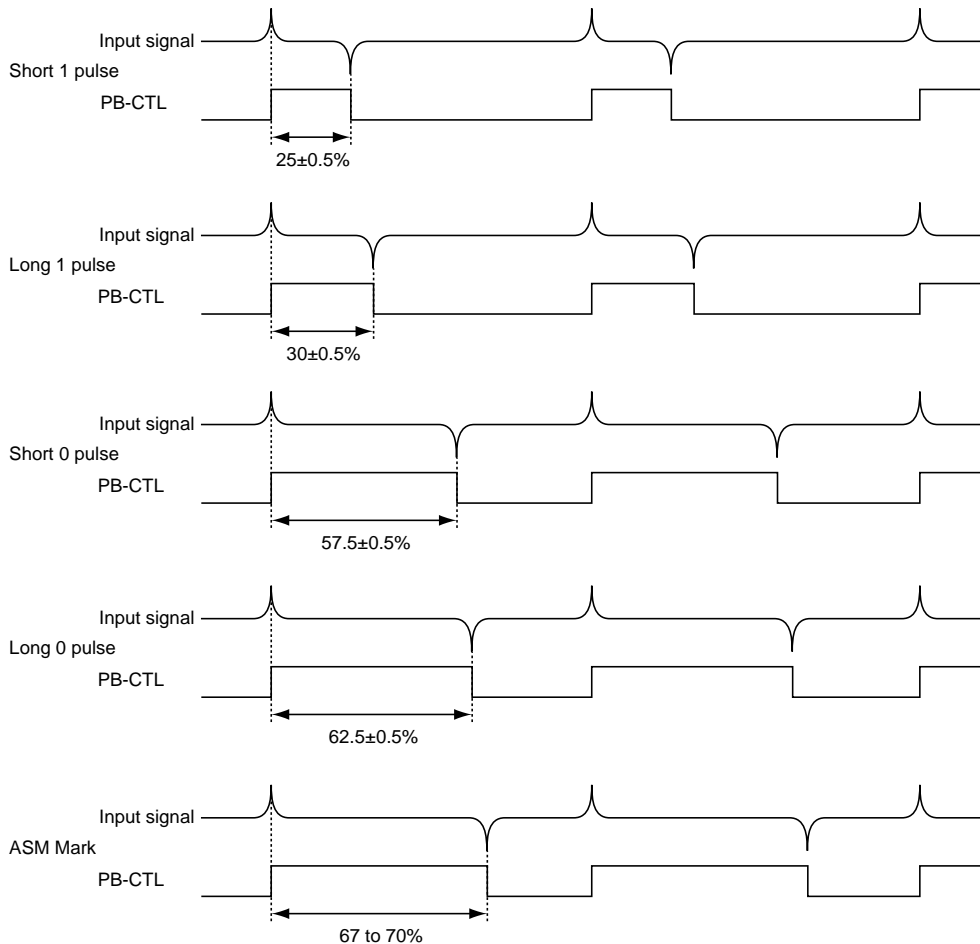


Figure 28.55 PB-CTL Signal Duty Cycle

Figure 28.56 shows the duty discrimination circuit. A 44% duty cycle is discriminated by counting with the 16-bit up/down counter, using a $\phi s/4$ clock for the up-count and a $\phi s/5$ clock for the down-count. An up-count is performed when the PB-CTL signal is high, and a down-count when low. Long or short pulse is discriminated by comparing with RCDR2 to RCDR5.

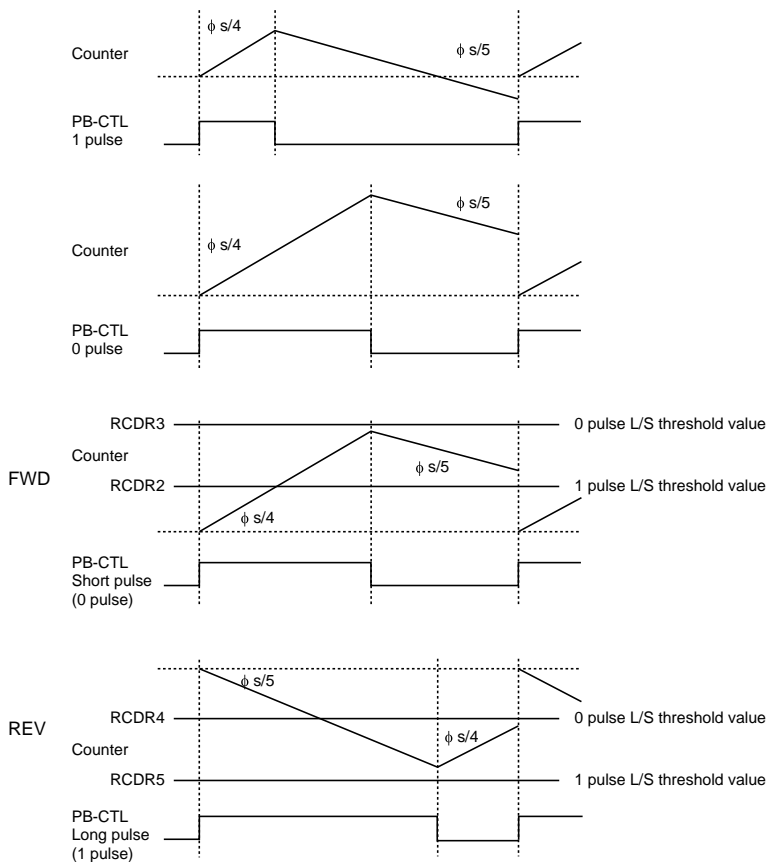
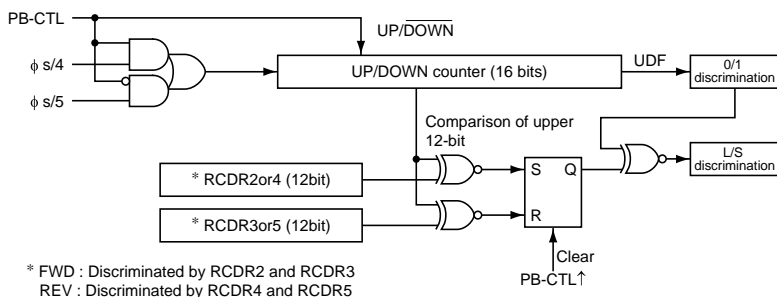


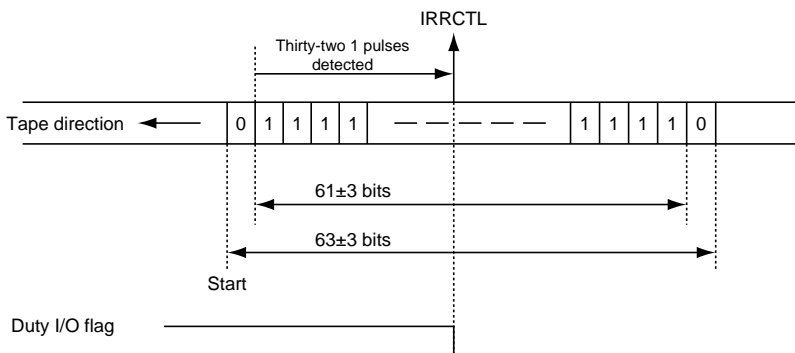
Figure 28.56 Duty Discriminator

(1) VISS (Index) Detect/Record Mode

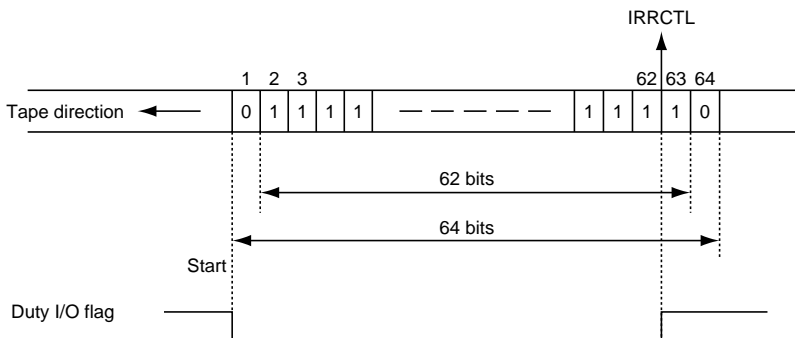
VISS detection is carried out by the VISS control circuit, which counts 1 pulses in the PB-CTL signal. If the pulse count detects any value set in the VISS interrupt setting bits (bits 5, 6 and 7 in the duty I/O register), an interrupt request is generated and the duty I/O flag is cleared to 0.

At VISS record or rewrite, INDEX code is automatically written. INDEX code is composed of continuous 62-bit data with 0 pulse data at both edges.

Examples of bit strings and the duty I/O flag at VISS detection/record are illustrated in figure 28.57.



(a) VISS detection (INDEX: Thirty-two 1 pulse setting)



(b) VISS record

Figure 28.57 Examples of VISS Bit Strings and Duty I/O Flag

(2) VASS Detect Mode

VASS detection is carried out by the duty discriminator. Software can detect index sequences by reading the duty I/O flag at each CTL pulse.

At each CTL pulse, the duty discriminator sends the result of duty discrimination to the duty I/O flag, and simultaneously generates an interrupt request. The duty I/O flag is cleared to 0 if the CTL pulse is a 1 (duty cycle below 43%), and is set to 1 if the CTL pulse is a 0 (duty cycle equal to or above 44%).

The duty I/O flag is modified at each CTL pulse. It should be read by the interrupt-handling routine within the period of the PB-CTL signal. The VASS detection format is illustrated in figure 28.58.

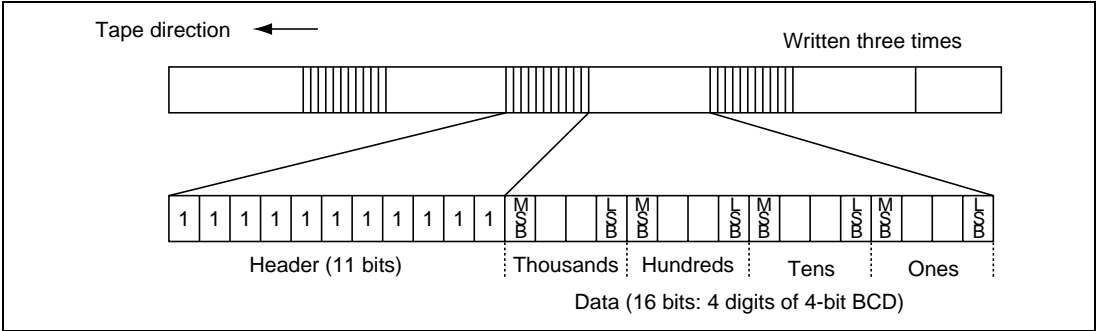


Figure 28.58 VASS (Index) Format

(3) Assemble (ASM) Mark Detect Mode

ASM mark detection is carried out by the duty discriminator. If the duty discriminator detects that the duty cycle of the PB-CTL signal is 66% or higher, it generates an interrupt request, and simultaneously clears the duty I/O flag to 0.

The duty I/O flag is updated at every CTL pulse. It should be read by the interrupt-handling routine within the period of the PB-CTL signal.

(4) Detection of the Long/Short Pulse

The Long/Short pulse is detected in PB mode by the L/S determination based on the comparison of the REC-CTL duty registers (RCDR2 to RCDR5) with the up/down counter and the results of the duty I/O flag. The results of the determination are stored in bit 0 (LSP0) of the bit pattern register (BTPR) at the rising edge of PB-CTL, shifting BTPR leftward at the same time.

RCDR2 to RCDR5 set the L/S thresholds for each of FWD/REV. Set to RCDR2 a threshold of 1 pulse L/S for FWD, to RCDR3 a threshold of 0 pulse L/S for FWD, to RCDR4 a threshold of 0 pulse L/S for REV, and to RCDR5 a threshold of 1 pulse L/S for REV. Figure 28.59 shows the detection of the Long/Short pulse.

Also, the bit pattern of eight bits can be detected by BTPR. Check that an 8-bit detection has been done by bit 1 (BPF bit) of the duty I/O register, and then read BTPR.

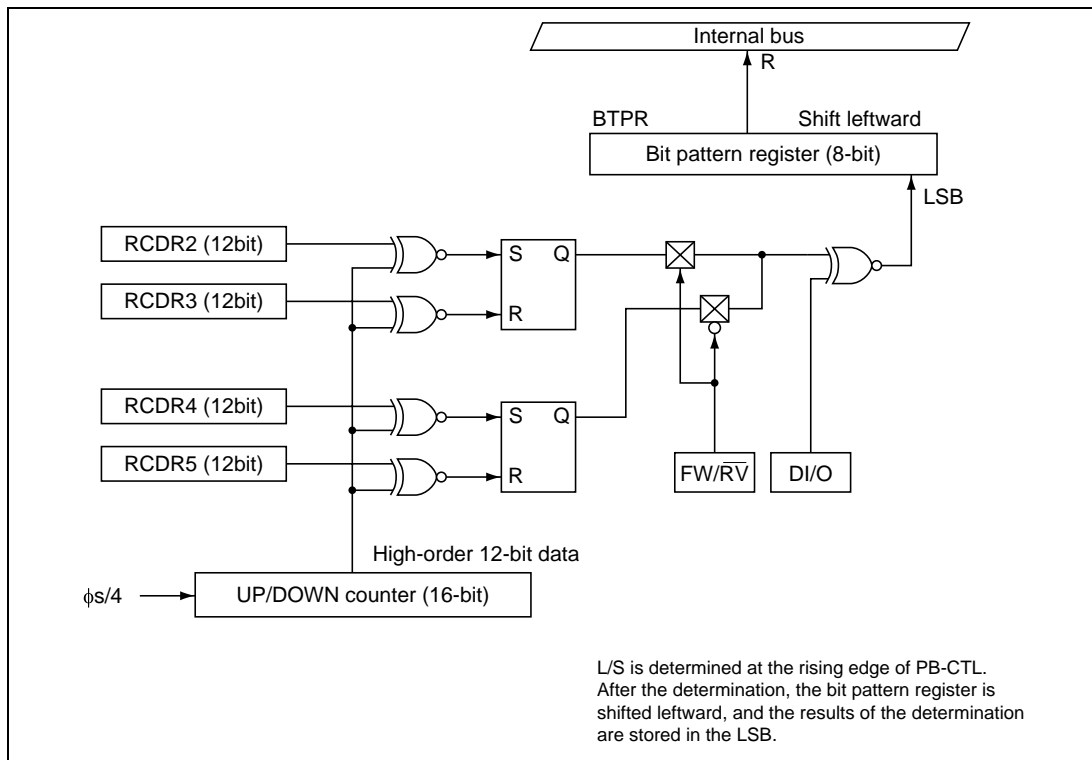


Figure 28.59 Detection of Long/Short Pulse

28.13.9 CTL Output Section

An on-chip control head amplifier is provided for writing the REC-CTL signal generated by the write control circuit onto the tape.

The write control circuit controls the duty cycle of the REC-CTL signal in the writing of VISS and VASS sequences and ASM marks and the rewriting of VISS and VASS sequences. The duty cycle of the REC-CTL signal is set in REC-CTL duty data registers 1 to 5 (RCDR1 to RCDR5). Times calculated in terms of $\phi_s (= f_{osc}/2)$ should be converted to appropriate data to be set in these registers. In VISS or VASS mode, set RCDR2 for a duty cycle of $25\pm0.5\%$, RCDR3 for a duty cycle of $30\pm0.5\%$, RCDR4 for a duty cycle of $57.5\pm0.5\%$, and RCDR5 for a duty cycle of $62.5\pm0.5\%$. When 1 is written in the duty I/O flag, the REC-CTL signal will be written on the tape with a $25\pm0.5\%$ duty cycle when 0 is written in bit 7 (LSP7) in the bit pattern register (BTPR) and with a $30\pm0.5\%$ duty cycle when 1 is written. Table 28.21 shows the relationship between the REC-CTL duty register and CTL outputs.

In ASM mark write mode, set RCDR3 for a duty cycle of 67% to 70%. An ASM mark will be written when 0 is written in the duty I/O flag.

An interrupt request is generated at the rise of the reference signal after one CTL pulse has been written. The reference signal is derived from the output signal (REF30X) of the X-value adjustment circuit, and has a period of one frame.

Figure 28.60 shows the timings that generate the REC-CTL signal.

Table 28.21 REC-CTL Duty Register and CTL Outputs

| MODE | D/I/O | LSP7 | Pulse | RCDR | Duty |
|---------------------|-------|------|-------|-------|----------------|
| VISS and VASS modes | 0 | 0 | S1 | RCDR2 | $25\pm0.5\%$ |
| | | 1 | L1 | RCDR3 | $30\pm0.5\%$ |
| | 1 | 0 | S0 | RCDR4 | $57.5\pm0.5\%$ |
| | | 1 | L0 | RCDR5 | $65.5\pm0.5\%$ |
| ASM mode | 0 | * | — | RCDR3 | 67 to 70% |

Note: * Don't care.

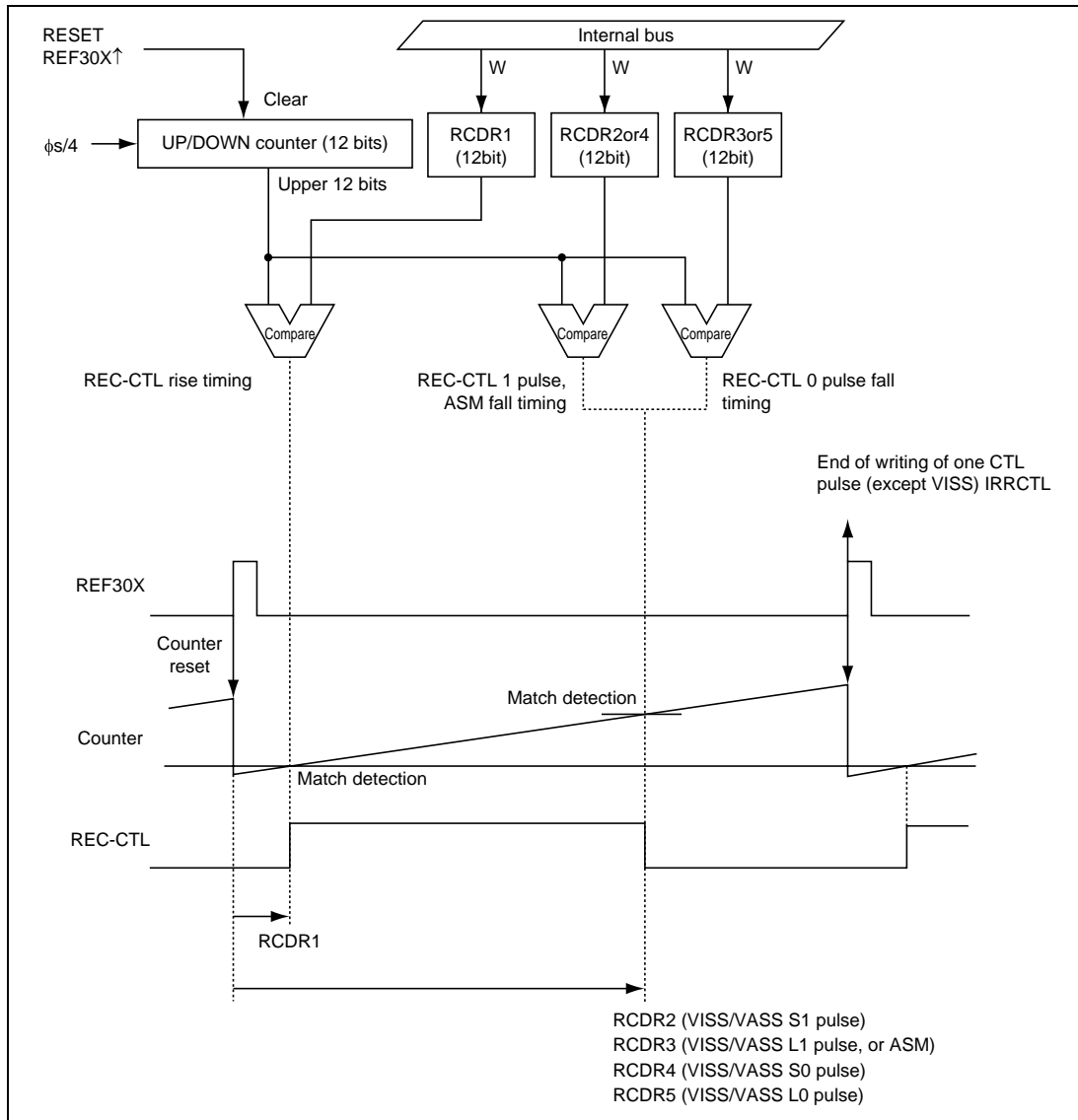


Figure 28.60 REC-CTL Signal Generation Timing

The 16-bit counter in the REC-CTL circuit continues counting on a clock derived by dividing the system clock $\phi_s (= f_{osc}/2)$ by 4. The counter is cleared on the rise of REF30X in record mode, and on the rise of PB-CTL in rewrite mode. The REC-CTL match detection is carried out by comparing the counter value with each RCDR value.

RCDR1 to RCDR5 can be written to by software at all times. If RCDR is changed before the respective match detection is performed, match detection is performed using the new value. The value changed after match detection becomes valid on the rise of REF30X following the change. Figure 28.61 shows an example of RCDR change timing.

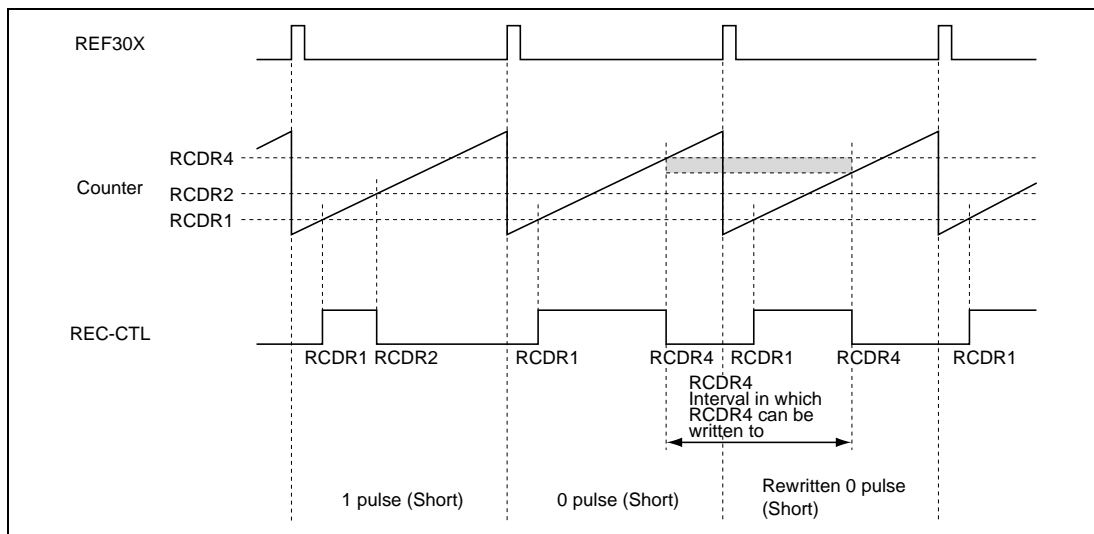


Figure 28.61 Example of RCDR Change Timing (Example Showing RCDR4)

28.13.10 Trapezoid Waveform Circuit

In rewriting, the trapezoid waveform circuit leaves the rising edge of the already-recorded PB-CTL signal intact, but changes the duty cycle.

In rewriting, the CTL pulse is written with reference to the rise of PB-CTL. The CTL duty cycle for a rewrite is set in the REC-CTL duty data registers (RCDR2 to RCDR5). Time values T2 to T5 are referenced to the rise of PB-CTL.

Figure 28.62 shows the rewrite waveform.

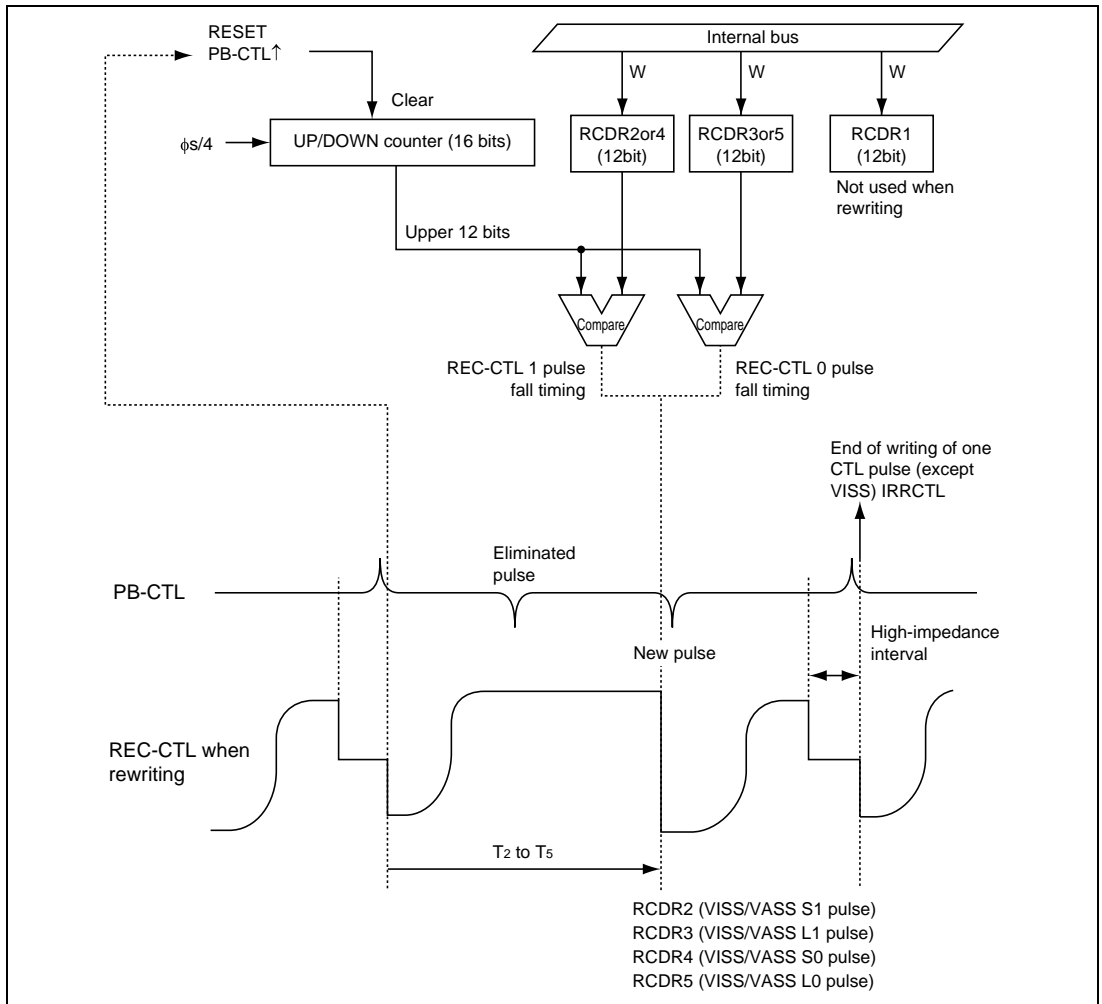


Figure 28.62 Relationship between REC-CTL and RCDR2 to RCDR5 when Rewriting

28.13.11 Note on CTL Interrupt

Following a reset, the CTL circuit is in the VASS detect (duty detect) mode.

Depending on the CTL pin states, a false PB-CTL input pulse may be recognized and an interrupt request generated. If the interrupt request will be enabled, first clear the CTL interrupt request flag.

28.14 Frequency Dividers

28.14.1 Overview

On-chip frequency dividers are provided for the pulse signal picked up from the control track during playback (PB-CTL signal), and the pulse signal received from the capstan motor (CFG signal). An on-chip noise canceller is provided for the drum motor pulse signal (DFG signal). The CTL frequency divider generates a CTL divided control signal (DVCTL) from the PB-CTL signal, for use in capstan phase control during high-speed search, for example. The CFG frequency divider generates two divided signals (DVCFG for speed control and DVCFG2 for phase control) from the CFG signal. The DFG noise canceller is a circuit which considers a signal less than 2ϕ as noise and masks it.

28.14.2 CTL Frequency Divider

(1) Block Diagram

Figure 28.63 shows a block diagram of the CTL frequency divider.

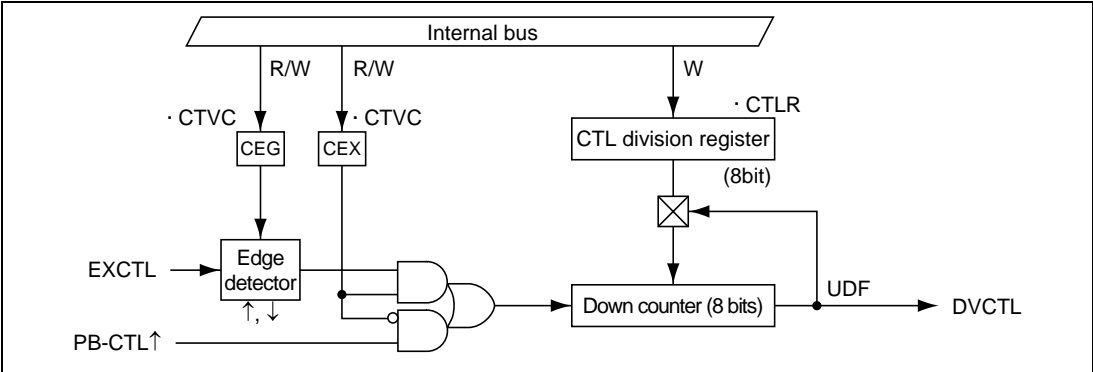


Figure 28.63 CTL Frequency Divider

(2) Register Configuration

- Register configuration

Table 28.22 shows the register configuration of the CTL dividers.

Table 28.22 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|------------------------|---------|-----|------|---------------|---------|
| DVCTL control register | CTVC | R/W | Byte | Undefined | H'FD098 |
| CTL division register | CTLR | W | Byte | H'00 | H'FD099 |

- **DVCTL control register (CTVC)**

| | | | | | | | | |
|-----------------|-----|-----|---|---|---|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CEX | CEG | — | — | — | CFG | HSW | CTL |
| Initial value : | 0 | 0 | 1 | 1 | 1 | * | * | * |
| R/W : | W | W | — | — | — | R | R | R |

Note: * Initial value is uncertain.

The DVCTL control register (CTVC) is a register consisting of the external input signal selection bit and the flags which show the CFG, HSW and CTL levels.

Note: It has an undetermined value by a reset or stand-by.

Bit 7: DVCTL Signal Generation Selection Bit (CEX)

Selects whether the PB-CTL signal or the external input signal is used to generate the DVCTL signal.

Bit 7

| CEX | Description |
|-----|---|
| 0 | Generates DVCTL signal with PB-CTL signal (Initial value) |
| 1 | Generates DVCTL signal with external input signal |

Bit 6: External Sync Signal Edge Selection Bit (CEG)

Selects the edge of the external signal at which the frequency division is made when the external signal was selected to generate the DVCTL signal.

Bit 6

| CEG | Description |
|-----|-----------------------------|
| 0 | Rising edge (Initial value) |
| 1 | Falling edge |

Bits 5 to 3: Reserved

No write in them is valid. If a read is attempted, an undetermined value is read out.

Bit 2: CFG Flag (CFG)

Shows the CFG level.

Bit 2

| CFG | Description |
|-----|-------------------------------------|
| 0 | CFG is at Low level (Initial value) |
| 1 | CFG is at High level |

Bit 1: HSW Flag (HSW)

Shows the level of the HSW signal selected by the VFF/NFF bit of the HSW mode register 2 (HSM2).

Bit 1

| HSW | Description |
|-----|-------------------------------------|
| 0 | HSW is at Low level (Initial value) |
| 1 | HSW is at High level |

Bit 0: CTL Flag (CTL)

Shows the CTL level.

Bit 0

| CTL | Description |
|-----|---|
| 0 | REC or PB-CTL is at Low level (Initial value) |
| 1 | REC or PB-CTL is at High level |

- CTL frequency division register (CTLR)**

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CTL7 | CTL6 | CTL5 | CTL4 | CTL3 | CTL2 | CTL1 | CTL0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

The CTL frequency division register (CTLR) is an 8-bit write-only register to set the frequency dividing value (N-1 if divided by N) for PB-CTL. If a read is attempted, an undetermined value is read out.

PB-CTL is divided by N at its rising edge. If the register value was 0, no division operation is performed, and the DVCTL signal with the same cycle with PB-CTL is output. It is initialized to H'00 by a reset or stand-by.

(3) Operation

During playback, control pulses recorded on the tape are picked up by the control head and input to the CTL pin. The control pulse signal is amplified by a Schmitt amplifier, reshaped, then input to the CTL frequency divider as the PB-CTL signal.

This circuit is employed when the control pulse (PB-CTL signal) is used for phase control of the capstan motor. The divided signal is sent as the DVCTL signal to the capstan phase system in the servo circuits, and the Timer R.

The CTL frequency divider is an 8-bit reload timer consisting of a reload register and a down-counter. Frequency division is obtained by setting frequency-division data in bits 7 to 0 in the CTL frequency division register (CTLR), which is the reload register. When a frequency-division value is written in this reload register, it is also written into the down-counter. The down-counter is decremented on rising edges of the PB-CTL signal.

Figure 28.64 shows examples of the PB-CTL signal and DVCTL waveforms.

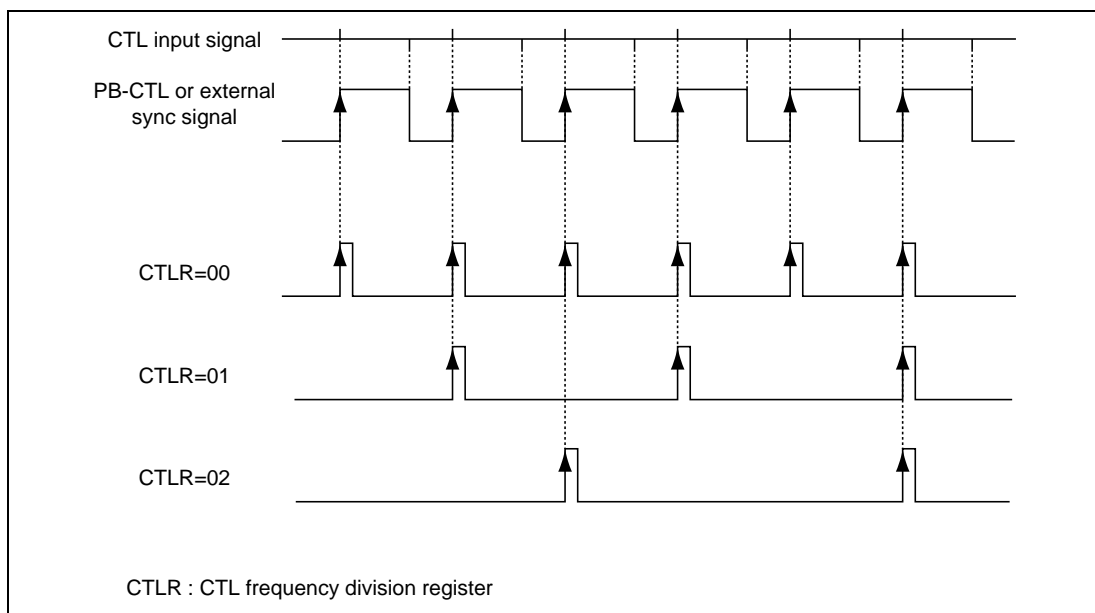


Figure 28.64 CTL Frequency Division Waveforms

28.14.3 CFG Frequency Divider

(1) Block Diagram

Figure 28.65 shows a block diagram of the 7-bit CFG frequency divider and its mask timer.

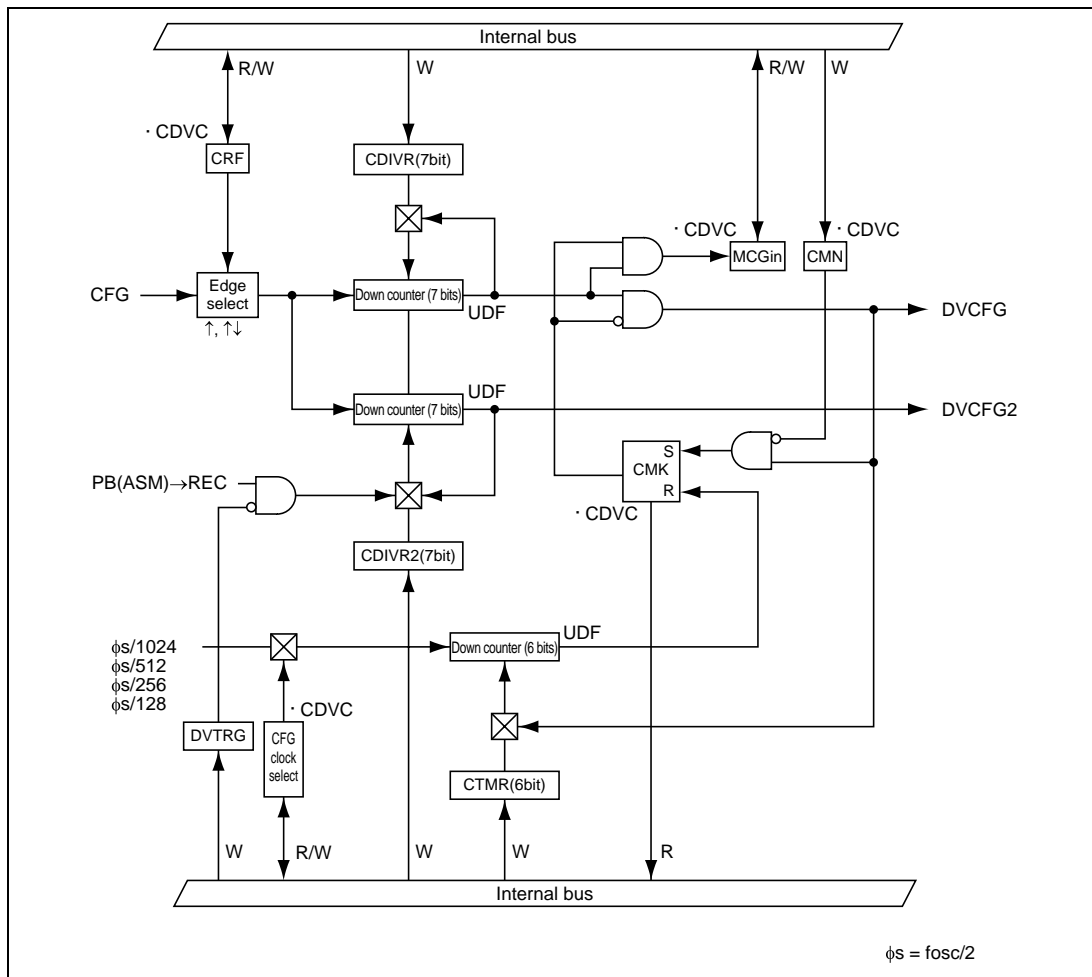


Figure 28.65 CFG Frequency Divider

(2) Register Descriptions

- Register configuration

Table 28.23 shows the register configuration of the CFG frequency divider.

Table 28.23 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|-----------------------------------|---------|-----|------|---------------|---------|
| DVCFG control register | CDVC | R/W | Byte | H'60 | F'FD09A |
| CFG frequency division register 1 | CDIVR1 | W | Byte | H'80 | H'FD09B |
| CFG frequency division register 2 | CDIVR2 | W | Byte | H'80 | H'FD09C |
| DVCFG mask period register | CTMR | W | Byte | H'FF | H'FD09D |

- DVCFG control register (CDVC)**

| | | | | | | | | |
|-----------------|-------|---|-----|-----|-------|-----|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | MCGin | — | CMK | CMN | DVTRG | CRF | CPS1 | CPS0 |
| Initial value : | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W* | — | R | W | W | W | W | W |

Note: * Only 0 can be written.

CDVC is an 8-bit register to control the capstan frequency division circuit.
It is initialized to H'60 by a reset, stand-by or module stop.

Bit 7: Mask CFG Flag (MCGin)

MCGin is a flag to indicate occurrence of a frequency division signal during the mask timer's mask period. To clear it, write 0. To clear it by software, write 0 after reading 1. Also, setting has the highest priority in this flag. If a condition setting the flag and 0 write occurs simultaneously, the latter is nullified.

Bit 7

| MCGin | Description |
|-------|---|
| 0 | CFG is in normal operation (Initial value) |
| 1 | Shows that DVCFG was detected during masking (runaway detected) |

Bit 6: Reserved

No write in it is valid. If a read is attempted, 1 is read out.

Bit 5: CFG Mask Status Bit (CMK)

Indicates the status of the mask. It is initialized to 1 by a reset, stand-by or module stop.

Bit 5

| CMK | Description |
|-----|--|
| 0 | Indicates that the capstan mask timer has released masking |
| 1 | Indicates that the capstan mask timer is currently masking (Initial value) |

Bit 4: CFG Mask Selection Bit (CMN)

Selects the turning ON/OFF of the mask function.

Bit 4

| CMN | Description |
|-----|--|
| 0 | Capstan mask timer function ON (Initial value) |
| 1 | Capstan mask timer function OFF |

Bit 3: PB (ASM) → REC Transition Timing Sync ON/OFF Selection Bit (DVTRG)

Selects the ON/OFF of the timing sync of the transition from PB (ASM) to REC when the DVCFG2 signal is generated.

Bit 3

| DVTRG | Description |
|-------|--|
| 0 | PB (ASM) → REC transition timing sync ON (Initial value) |
| 1 | PB (ASM) → REC transition timing sync OFF |

Bit 2: CFG Frequency Division Edge Selection Bit (CRF)

Selects the edge of the CFG signal to be divided.

Bit 2

| CRF | Description |
|-----|---|
| 0 | Performs frequency division at the rising edge of CFG (Initial value) |
| 1 | Performs frequency division at both edges of CFG |

Bits 1 and 0: CFG Mask Timer Clock Selection Bits (CPS1, CPS0)

Selects the clock source for the CFG mask timer. ($\phi s = fosc/2$)

| Bit 1 | Bit 0 | Description |
|-------|-------|-------------------------------|
| CPS1 | CPS0 | |
| 0 | 0 | $\phi s/1024$ (Initial value) |
| | 1 | $\phi s/512$ |
| 1 | 0 | $\phi s/256$ |
| | 1 | $\phi s/128$ |

- **CFG frequency division register 1 (CDIVR1)**

| | | | | | | | | |
|-----------------|---|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | CDV16 | CDV15 | CDV14 | CDV13 | CDV12 | CDV11 | CDV10 |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | W | W | W | W | W | W | W |

The CFG frequency division register 1 (CDIVR1) is an 8-bit write-only register to set the CFG division value (N-1 for N division). If a read is attempted, an undetermined value is read out. Bit 7 is reserved.

The frequency division value is written in the reload register and the down counter at the same time.

CFG's frequency is divided by N at its rising edge or both edges. If the register value was 0, no division operation is performed, and the DVCFG signal with the same input cycle with the CFG signal is output. The DVCFG signal is sent to the capstan speed error detector. It is initialized to H'80 by a reset or stand-by together with the capstan frequency division register and the down counter.

- **CFG frequency division register 2 (CDIVR2)**

| | | | | | | | | |
|-----------------|---|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | CDV26 | CDV25 | CDV24 | CDV23 | CDV22 | CDV21 | CDV20 |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | W | W | W | W | W | W | W |

The CFG frequency division register 2 (CDIVR2) is an 8-bit write-only register to set the CFG division value (N-1 for N division). If a read is attempted, an undetermined value is read out. Bit 7 is reserved.

The frequency division value is written in the reload register and the down counter at the same time.

CFG's frequency is divided by N at its rising edge or both edges. If the register value was 0, no division operation is performed, and the DVCFG2 signal with the same input cycle with the CFG signal is output. The DVCFG2 signal is sent to the capstan speed error detector and the Timer L.

The DVCFG2 circuit has no mask timer function.

The frequency division counter for the DVCFG2 signal starts its division operation at the point data was written in CDIVR2. If synchronization is required for phase matching, for example, do it by writing in CDIVR2. If the DVTRG bit of the CDVC register was 0, the register synchronizes with the switching timing from PB (ASM) to REC.

It is initialized to H'80 by a reset or stand-by together with the capstan frequency division register and the down counter.

- **DVCFG mask period register (CTMR)**

| | | | | | | | | |
|-----------------|---|---|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | CPM5 | CPM4 | CPM3 | CPM2 | CPM1 | CPM0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | — | — | W | W | W | W | W | W |

The DVCFG mask period register (CTMR) is an 8-bit write-only register. If a read is attempted, an undetermined value is read out. CTMR is a reload register for the mask timer (down counter). Set in it the mask period of CFG. The mask period is determined by the clock specified by bits 1 and 0 of CDVC and the set value (N-1). If data is written in CTMR, it is written also in the mask timer at the same time.

It is initialized to H'FF by a reset, stand-by or module stop.

Mask period = $N \times \text{clock cycle}$

(3) Operation

- Frequency divider

The CFG pulses output from the capstan motor are sent to internal circuitry as the CFG signal via the zero-cross type comparator. The CFG signal, shaped into a rectangular waveform by a reshaping circuit, is divided by the CFG frequency dividers, and used in servo control. The rising edge or both edges of the CFG signal can be selected for the frequency divider.

The CFG frequency dividers comprises a 7-bit frequency divider with a mask timer for capstan speed control (DVCFG signal generator) and a 7-bit frequency divider for capstan phase control (DVCFG2 signal generator).

The DVCFG signal generator consists of a 7-bit reload register (CFG frequency division register1: CDIVR1), a 7-bit down-counter, and a 6-bit mask timer (with settable mask interval). Frequency division is performed by setting the frequency-division value in 7-bit CDIVR1. When the frequency-division value is written in CDIVR1, it is also written in the down-counter. After frequency division of a CFG signal for which the edge has been selected, the signal is sent via the mask timer to the capstan speed error detector as the DVCFG signal.

The DVCFG2 signal generator consists of a 7-bit reload register (CFG frequency division register 2: CDIVR2) and a 7-bit down-counter. The 7-bit frequency divider does not have a mask timer. Frequency division is performed by setting the frequency-division value in CDIVR2. When the frequency-division value is written in CDIVR2, it is also written in the down-counter. After frequency division of a CFG signal for which the edge has been selected, the signal is sent to the capstan speed error detector and the Timer L as the DVCFG2 signal. Frequency division starts when the frequency-division value is written.

When the DVTRG bit in the CDVC register is set to 0, reloading is executed with the switchover timing from PB (ASM) mode to REC mode. To switch from REF30 to CREF, change the settings of bit 4 (CR/RF bit) in the capstan phase error detection control register (CPGCR). If synchronization is necessary for phase control, this can be provided by writing the frequency-division value in CDIVR2.

The down-counters are decremented on rising edges of the CFG signal when the CRF bit is 0 in the DVCFG control register (CDVC), and on both edges when the CRF bit is 1.

Figure 28.66 shows examples of CFG frequency division waveforms.

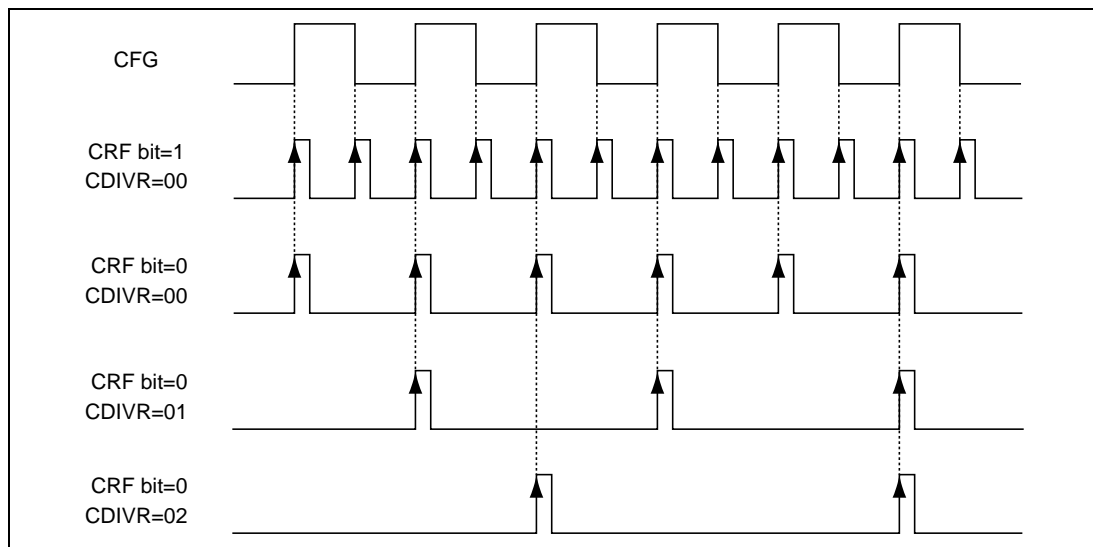


Figure 28.66 Frequency Division Waveforms

- Mask timer

The capstan mask timer is a 6-bit reload timer that uses a prescaled clock as a clock source.

The mask timer is used for masking the DVCFG signal intended for controlling the capstan speeds.

The capstan mask timer prevents edge detection to be carried out for an unnecessarily long duration by masking the edge detection for a certain period. The above trouble can result from abnormal revolution (runout) of the capstan motor because its revolution has to cover a wide range speeds from the slow/still up to the high speed search.

The capstan mask timer is started by output of a pulse edge in the divided CFG signal (DVCFG). While the timer is running, a mask signal disables the output of further DVCFG pulses. The mask signal is shown in Figure 28.67.

The mask timer status can be recognized by reading the CMK flag in the DVCFG control register (CDVC).

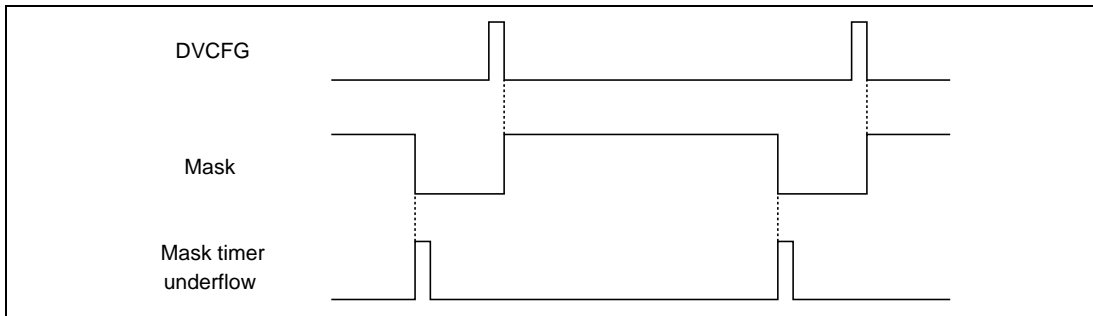


Figure 28.67 Mask Signal

Figures 28.68 and 28.69 show examples of CFG mask timer operations.

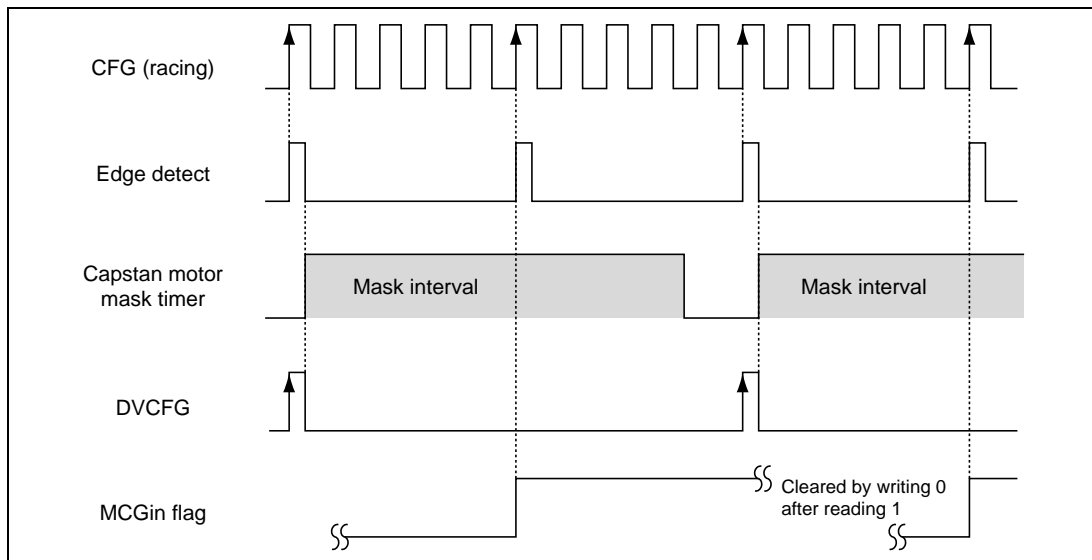


Figure 28.68 CFG Mask Timer Operation (When Capstan Motor is Racing)

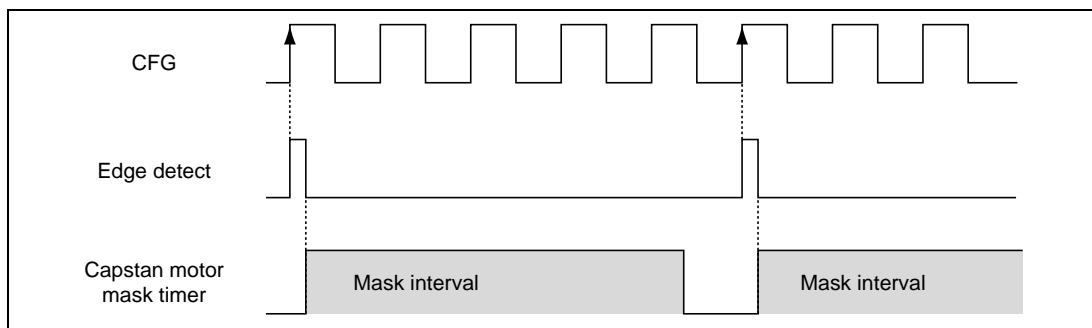


Figure 28.69 CFG Mask Timer Operation (When Capstan Motor is Operating Normally)

28.14.4 DFG Noise Removal Circuit

(1) Block Diagram

Figure 28.70 shows the block diagram of the DFG noise removal circuit.

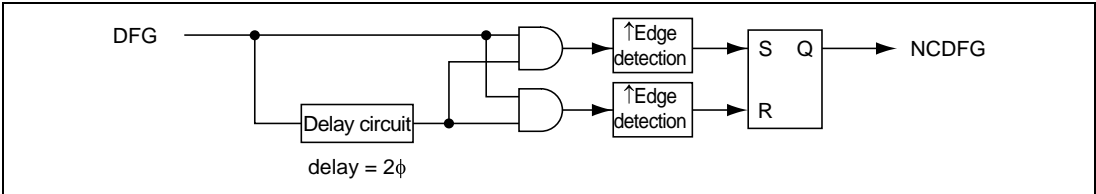


Figure 28.70 DFG Noise Removal Circuit

(2) Register Descriptions

- Register configuration

Table 28.24 shows the register configuration of the DFG mask circuit.

Table 28.24 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|---------------------|---------|-----|------|---------------|---------|
| FG control register | FGCR | W | Byte | H'FE | H'FD09E |

- FG Control Register (FGCR)

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | DRF |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| R/W : | — | — | — | — | — | — | — | W |

Selects the edge of the DFG noise removal signal (NCD FG) to be sent to the drum speed error detector. If a read is attempted, an undetermined value is read out. Bits 7 to 1 are reserved. No write in them is valid.

It is initialized to H'FE by a reset, stand-by or module stop.

The edge selection circuit is located in the drum speed error detector, and outputs the register output to the drum speed error detector.

Bits 7 to 1: Reserved

No write in them is valid. If a read is attempted, an undetermined value is read out.

Bit 0: DFG Edge Selection Bit (DRF)

Selects the edge of the NCDFG signal used in the drum speed error detector.

Bit 0

| DRF | Description |
|-----|---|
| 0 | Selects the rising edge of NCDFG signal (Initial value) |
| 1 | Selects the falling edge of NCDFG signal |

(3) Description of Operation

The DFG noise removal circuits generates a signal (NCDFG signal) with a delay circuit as a result of removing noise (signal fluctuation smaller than 2ϕ) from the DFG signal. The resulted NCDFG signal is behind the time when the DFG signal was detected by 2ϕ . Figure 28.71 shows the NCDFG signal.

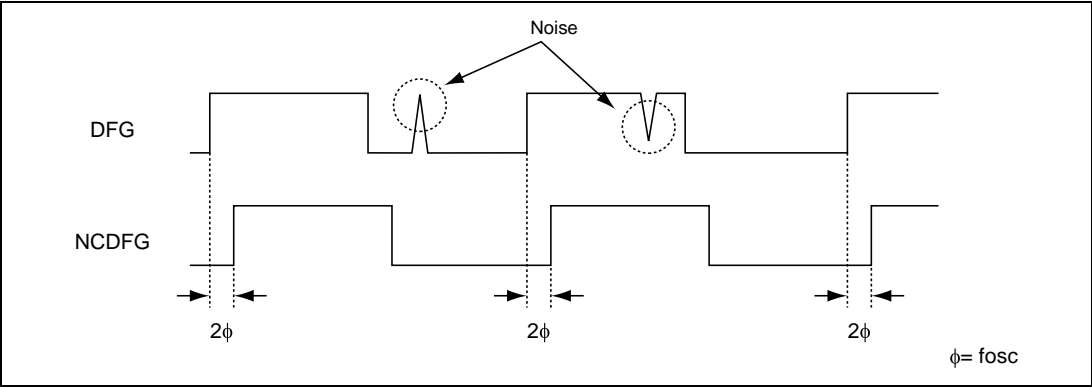


Figure 28.71 NCDFG signal

28.15 Sync Signal Detector

28.15.1 Overview

This block performs detection of the horizontal sync signal (Hsync) and vertical sync signal (Vsync) from the composite sync signal (Csync), noise counting, and field detection. It detects the horizontal and vertical sync signals by setting threshold in the register and based on the servo clock ($\phi_s = f_{osc}/2$). Noise masking is possible during the detection of the horizontal sync signals, and if any Hsync is missing, it can be supplemented. Also, if total volume of the noise detected in one frame of Csync amounted over a specified volume, the detector generates a noise detection interrupt.

Note: This circuit detects a pulse with a specific width set by the threshold register. It does not classify or restore the sync signal to a formal one.

28.15.2 Block Diagram

Figure 28.72 shows the block diagram of the sync signal detector.

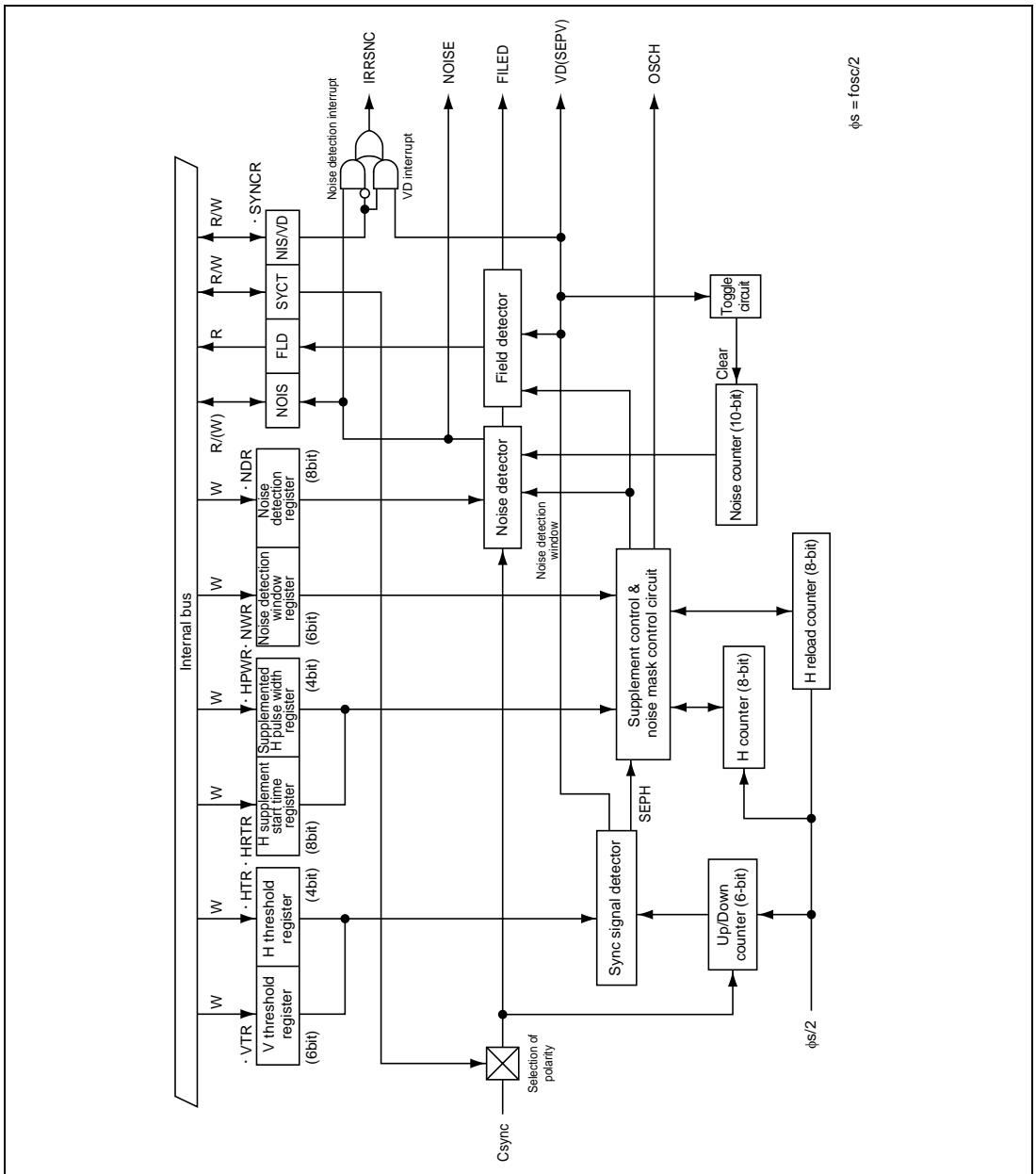


Figure 28.72 Block Diagram of the Sync Signal Detector

28.15.3 Pin Configuration

Table 28.25 shows the pin configuration of the sync signal detector.

Table 28.25 Pin Configuration

| Name | Abbrev. | I/O | Function |
|---------------------------------|---------|-------|-----------------------------|
| Composite sync signal input pin | Csync | Input | Composite sync signal input |

28.15.4 Register Configuration

Table 28.26 shows the register configuration of the sync signal detector.

Table 28.26 Register Configuration

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|---|---------|-----|------|---------------|---------|
| Vertical sync signal threshold register | VTR | W | Byte | H'C0 | H'FD0B0 |
| Horizontal sync signal threshold register | HTR | W | Byte | H'F0 | H'FD0B1 |
| H supplement start time setting register | HRTR | W | Byte | H'00 | H'FD0B2 |
| Supplemented H pulse width setting register | HPWR | W | Byte | H'F0 | H'FD0B3 |
| Noise detection window setting register | NWR | W | Byte | H'C0 | H'FD0B4 |
| Noise detector register | NDR | W | Byte | H'00 | H'FD0B5 |
| Sync signal control register | SYNCR | R/W | Byte | H'F8 | H'FD0B6 |

28.15.5 Register Descriptions

(1) Vertical Sync Signal Threshold Register (VTR)

| | | | | | | | | |
|-----------------|---|---|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | VTR5 | VTR4 | VTR3 | VTR2 | VTR1 | VTR0 |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/ W : | — | — | W | W | W | W | W | W |

Sets the threshold for the vertical sync signal when the signal is detected from the composite sync signal. The threshold is set by bits 5 to 0 (VTR5 to VTR0). Bits 7 and 6 are reserved.

VTR is an 8-bit write-only register. If a read is attempted, an undetermined value is read out. It is initialized to H'C0 by a reset, stand-by or module stop.

(2) Horizontal Sync Signal Threshold Register (HTR)

| | | | | | | | | |
|-----------------|---|---|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | HTR3 | HTR2 | HTR1 | HTR0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W |

Sets the threshold for the horizontal sync signal when the signal is detected from the composite sync signal. The threshold is set by bits 3 to 0 (HTR3 to HTR0). Bits 7 and 4 are reserved.

HTR is an 8-bit write-only register. If a read is attempted, an undetermined value is read out. It is initialized to H'F0 by a reset, stand-by or module stop.

Figure 28.73 shows threshold and separated sync signals.

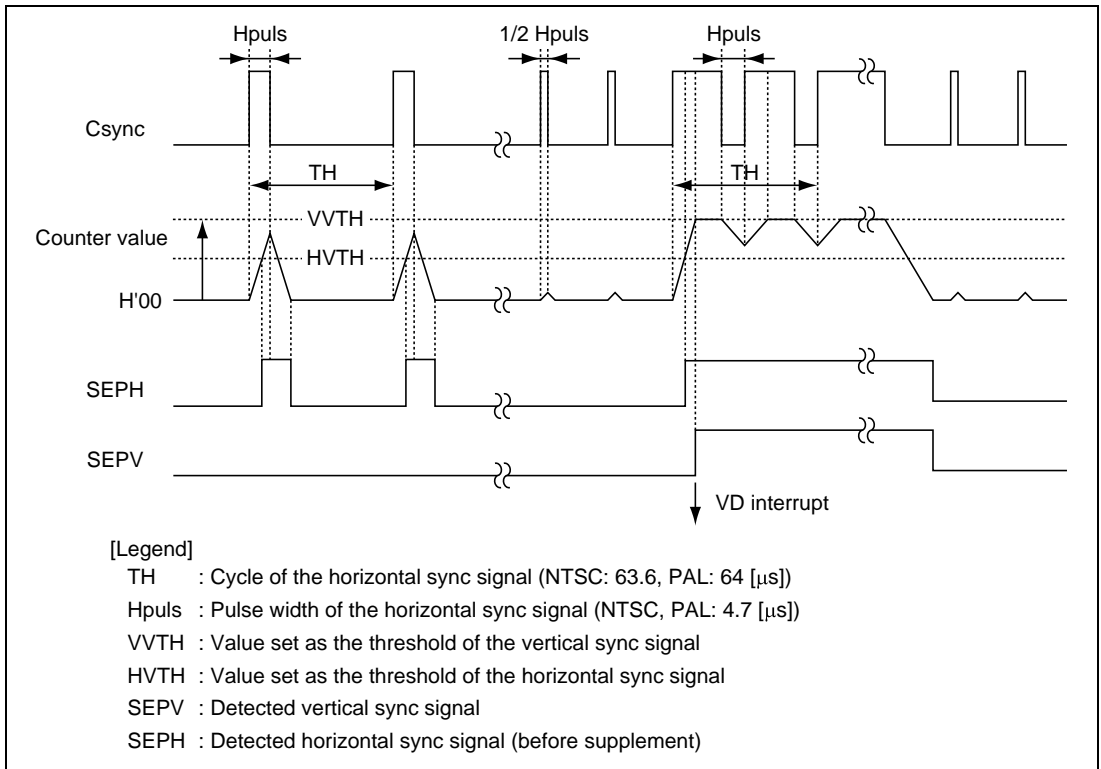


Figure 28.73 Threshold and Separated Sync Signals

- Example

The set values to detect the vertical and horizontal sync signals (SEPV and SEPH) from Csync are required to meet the following conditions. Assumed that the set values in the VTHR register were VVTH and HVTH,

$$(VVTH-1) \times 2/\phi_s > H_{puls}$$

$$(HVTH-2) \times 2/\phi_s \leq H_{puls}/2 < (HVTH-1) \times 2/\phi_s$$

Where, H_{puls} is pulse width (μs) of the horizontal sync signal, and ϕ_s is servo clock ($f_{osc}/2$).

Thus, if $\phi_s = 5 \text{ MHz}$, NTSC system is used,

$$(VVTH-1) \times 0.4\mu s > 4.7\mu s$$

$$\therefore VVTH \geq H'D$$

$$(HVTH-2) \times 0.4\mu s \leq 2.35\mu s < (HVTH-1) \times 0.4\mu s$$

$$\therefore HVTH \geq H'7$$

Note: This circuit detects the pulse with the width set in the VTHR register. If a noise pulse with the width greater than the set value was input, the circuit regards that it detected a sync signal.

(3) H Supplement Start Time Setting Register (HRTR)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | HRTR7 | HRTR6 | HRTR5 | HRTR4 | HRTR3 | HRTR2 | HRTR1 | HRTR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

Sets the timing to generate a supplementary pulse if a drop-out of a pulse of the horizontal sync signal occurred.

HRTR is an 8-bit write-only register. If a read is attempted, an undetermined value is read out. It is initialized to H'00 by a reset, stand-by or module stop.

$$((\text{Value of HRTR7 to HRTR0}) + 1) \times 2/\phi_s = \text{TH}$$

where, TH is the cycle of the horizontal sync signal (μs), and ϕ_s is the servo clock ($f_{\text{osc}}/2$).

Whether the horizontal sync signal exists or not is determined one clock before the supplementary pulse is generated. Accordingly, set to HRTR7 to HRTR0 a value obtained from the equation shown above plus one.

Also, HRTR7 to HRTR0 set the noise mask period. If the horizontal sync signal had the normal pulses, it is masked in the mask period.

The start and end of the mask period are computed from the rising edge of OSCH and SEPH, respectively. See figure 28.75.

(4) Supplemented H Pulse Width Setting Register (HPWR)

| | | | | | | | | |
|-----------------|---|---|---|---|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | HPWR3 | HPWR2 | HPWR1 | HPWR0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W |

HPWR sets the pulse width of the supplemented pulse which is generated if a drop-out of a pulse of the horizontal sync signal occurs. Bits 7 to 4 are reserved.

HPWR is an 8-bit write-only register. If a read is attempted, an undetermined value is read out. It is initialized to H'F0 by a reset or stand-by.

$$((\text{Value of HPWR3 to HPWR0}) + 1) \times 2/\phi_s = \text{Hpulse}$$

Where, Hpuls is the pulse width of the horizontal sync signal (μs), and ϕ_s is the servo clock ($f_{\text{osc}}/2$).

(5) Noise Detection Window Setting Register (NWR)

| | | | | | | | | |
|-----------------|---|---|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | NWR5 | NWR4 | NWR3 | NWR2 | NWR1 | NWR0 |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | W | W | W | W | W | W |

NWR sets the period (window) when the drop-out of the pulse of the horizontal sync signal is detected and the noise is counted. Set the timing of the noise detection window in bits 5 to 0. Bits 7 and 6 are reserved.

NWR is an 8-bit write-only register. If a read is attempted, an undetermined value is read out. It is initialized to H'C0 by a reset, stand-by or module stop.

Set the value of the noise detection window timing according to the following equation.

$$((\text{Value of NWR5 to NWR0}) + 1) \times 2/\phi_s = 1/4 \times \text{TH}$$

Where, TH is the pulse width of the horizontal sync signal (μs), and ϕ_s is the servo clock ($f_{\text{osc}}/2$).

It is recommended that this timing value is set at about 1/4 of the cycle of the horizontal sync signal.

(6) Noise Detection Register (NDR)

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | NDR7 | NDR6 | NDR5 | NDR4 | NDR3 | NDR2 | NDR1 | NDR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

NDR sets the noise detection level when the noise of the horizontal sync signal is detected (when NWR is set). Set the noise detection level in bits 7 to 0.

NDR is an 8-bit write-only register. No read is valid. If a read is attempted, an undetermined value is read out. It is initialized to H'00 by a reset, stand-by or module stop.

The noise detector takes counts of the drop-outs of the horizontal sync signals and the noises within the pulses, and if they amount to a count greater than four times of the value set in NDR7 to NDR0, the detector sets the NOIS flag in the sync signal control register (SYNCR). Set the noise detection level at 1/4 of the noise counts in one frame.

The noise counter is cleared whenever Vsync was detected twice.

See section 28.15.6, Noise Detection, for the details of the noise detection window and the noise detection level.

(7) Sync Signal Control Register (SYNCR)

| | | | | | | | | |
|-----------------|---|---|---|---|--------|--------|-----|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | NIS/VD | NOIS | FLD | SYCT |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| R/W : | — | — | — | — | R/W | R/(W)* | R | R/W |

Note: * Only 0 can be written

SYNCR controls the noise detection, field detection, polarity of the sync signal input, etc.

SYNCR is an 8-bit register. It is initialized to H'F8 by a reset, stand-by or module stop. Bits 7 to 4 are reserved. No write is valid. Bit 1 is valid for read only.

Bits 7 to 4: Reserved

Writes are disabled. If a read is attempted, an undetermined value is read out.

Bit 3: Interrupt Selection Bit (NIS/VD)

Selects whether an interrupt request is generated when a noise level was detected or when the VD signal was detected.

Bit 3

| NIS/VD | Description |
|--------|---------------------------------|
| 0 | Interrupt at the noise level |
| 1 | Interrupt at VD (Initial value) |

Bit 2: Noise Detection Flag (NOIS)

NOIS is a status flag indicating that the noise counts reached at more than four times of the value set in NDR. The flag is cleared only by writing 0 after reading 1. Care is required because it is not cleared automatically.

Bit 2

| NOIS | Description |
|------|--|
| 0 | Noise count is smaller than four times of the value set in NDR (Initial value) |
| 1 | Noise count is equal to or greater than four times of the value set in NDR |

Bit 1: Field Detection Flag (FLD)

Indicates whether the field currently being scanned is even or odd. See figure 28.74.

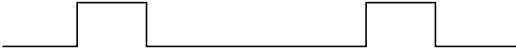
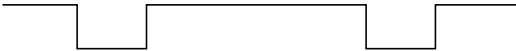
Bit 1

| FLD | Description | |
|-----|-------------|-----------------|
| 0 | Odd field | (Initial value) |
| 1 | Even field | |

Bit 0: Sync Signal Polarity Selection Bit (SYCT)

Selects the polarity of the sync signal (Csync) to be input.

Bit 0

| SYCT | Description | Polarity |
|------|---|-----------------------------|
| 0 |  | Positive (Initial value) |
| 1 |  | Negative |

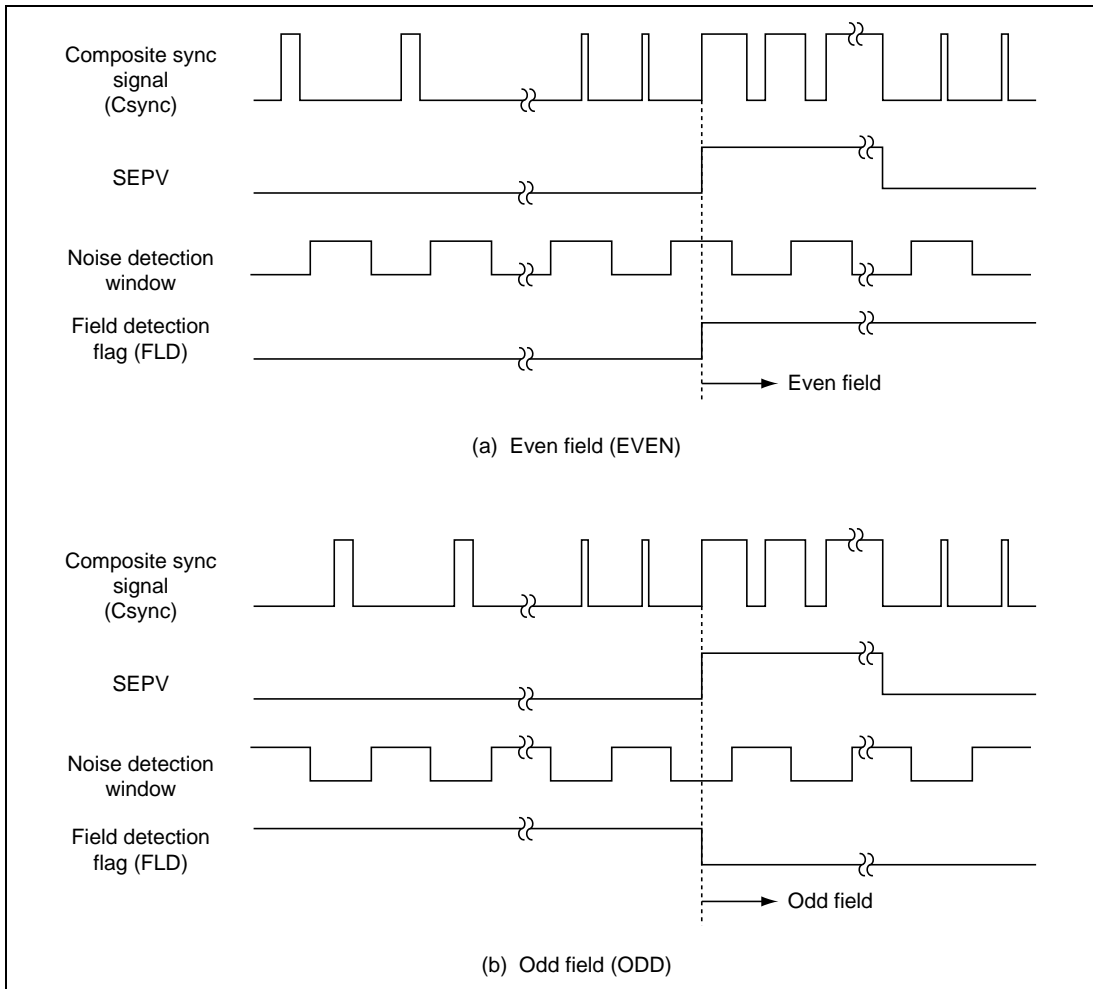


Figure 28.74 Field Detection

28.15.6 Noise Detection

If drop-out of a pulse of the horizontal sync signal occurred, set a supplemented pulse at the timing set in HPWR and with the set pulse width.

Set the noise detection window with HWR of about 1/4 of the horizontal sync signal, and the pulse with equal High and Low periods will be obtained.

(1) Example of Setting

Assumed that a supplemented pulse is set when $f_{osc} = 10\text{MHz}$ under the conditions $\phi_s = 5\text{MHz}$, NTSC:TH = 63.6 (μs) and Hpuls = 4.7 (μs), the set values of the supplemented pulse timing (HRTR7-0), supplemented pulse width (HPWR3-0) and noise detection window timing (NWR5-0) are expressed by the following equations.

$$(\text{Value of HRTR7-0}) \times 2/\phi_s = \text{TH}$$

$$((\text{Value of HPWR3-0}) + 1) \times 2/\phi_s = \text{Hpuls}$$

$$((\text{Value of NWR5-0}) + 1) \times 2/\phi_s = 1/4 \times \text{TH}$$

Where, TH is the cycle of the horizontal sync signal (μs), Hpuls is the pulse width of the horizontal sync signal (μs) and ϕ_s is the servo clock (Hz) ($f_{osc}/2$).

Accordingly,

$$(\text{Value of HRTR7 to HRTR0}) \times 0.4 (\mu\text{s}) = 63.6 (\mu\text{s})$$

$$\therefore \text{HRTR7-0} = \text{H'9F}$$

$$((\text{Value of HPWR3 to HPWR0}) + 1) \times 0.4 (\mu\text{s}) = 4.7 (\mu\text{s})$$

$$\therefore \text{HPWR3-0} = \text{H'B}$$

$$((\text{Value of NWR5 to NWR0}) + 1) \times 0.4 (\mu\text{s}) = 16 (\mu\text{s})$$

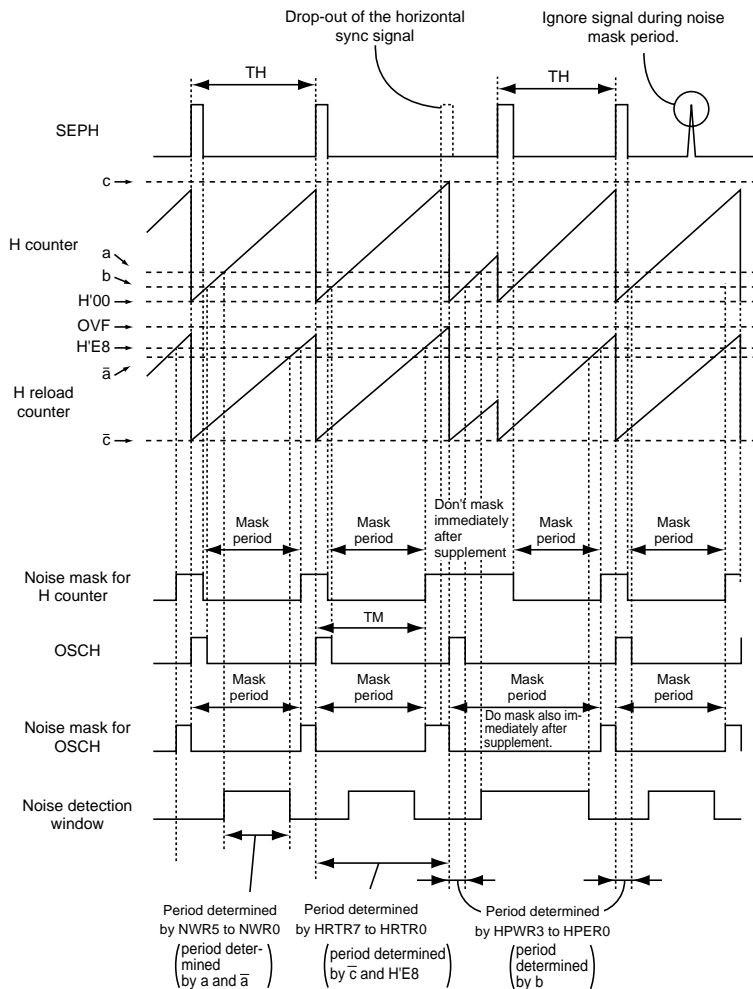
$$\therefore \text{NWR5-0} = \text{H'27}$$

Also, the noise mask period is computed as follows.

$$((\text{Value of HRTR7 to HRTR0}) + 1) - 24 \times 2/\phi_s = 54 (\mu\text{s})$$

Where, 24 is a constant required for a structural reason.

Figure 28.75 shows the set period for HRTR, HPWR and NWR.



[Legend]

SEPH : Horizontal sync signal after detection

OSCH : Horizontal sync signal after supplement

a : Value set for the noise detection window (NWR5 to NWR0)

b : Value set for the pulse width of the horizontal sync signal (NPWR3 to NPWR0)

c : Value set for supplement timing (HRTR7 to HRTR0)

a, b, c : Complements of 1 of a,b,c, respectively

H'E8 : Complement of 2 of multiplier 24 in the equation for the noise mask period

(The noise mask period ends 24 counts before the overflow of H reload counter.)

TH : Cycle of the horizontal sync signal

(NTSC:63.6 [ms], PAL:64[ms])

TM : Timing at which the noise mask period ends.

Figure 28.75 Set Period for HRTR, HPWR and NWR

(2) Operation to Detect Noise

The noise detector considers an irregular pulse of the composite sync signal (Csync) and a chip of a pulse of the horizontal sync signal within a frame as noise. The noise counter takes counts of the irregular pulses during the High period of the noise detection window and the chips and drop-outs of the horizontal sync signal pulses during the Low period. Also, it counts more than one irregular pulses as one. The noise counter is cleared at every frame (Vsync is detected twice).

The equivalent pulse contained in 9H of the vertical sync signal is counted also as an irregular pulse.

It sets the noise detection flag (NOIS) in the sync signal control register (SYNCR) at 1 if the count of the irregular pulses + the count of the pulse chips and drop-outs of the horizontal sync signal $> 4 \times (\text{value of NDR7 to NDR0})$.

See section 28.15.5 (7), Sync Signal Control Register (SYNCR) for the NOIS bit.

Figure 28.76 shows the operation of the noise detection.

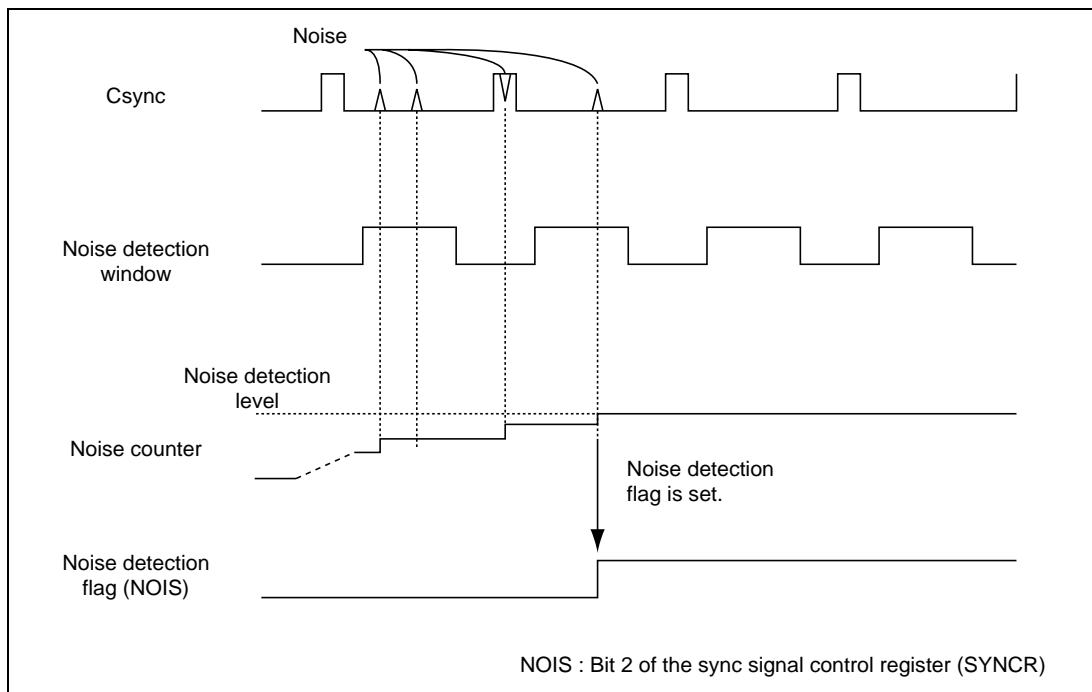


Figure 28.76 Operation of the Noise Detection

28.15.7 Sync Signal Detector Activation

The sync signal detector starts operation by release of reset, or by accepting input of a sync signal after its transition from power-down mode to active mode and release of module stop. The signal given to the detector is the polarity pulse assigned by the SYCT bit of the sync signal control register (SYNCR). The detector starts operation even if this pulse was a noise pulse with a width short of the regular width. The minimum pulse width which can activate the detector is not constant depending on the internal operation of the input circuit. Accordingly, if the assured activation of the detector is required, input a pulse with a width greater than $4/\phi_s$ ($\phi_s = f_{osc}/2$ (Hz)). In such a case, care is required to noise, etc., because even a pulse with a width smaller than $4\phi/s$ may cause activation.

28.16 Servo Interrupt

28.16.1 Overview

The interrupt exception processing of the servo module is started by one of ten factors, i.e. the drum speed error detector (×2), drum phase error detector, capstan speed error detector (×2), capstan phase error detector, HSW timing generator (×2), sync detector and CTL circuit. For these interrupt factors, see each of their circuit sections in this manual. Also, see section 5, Exception Handling.

28.16.2 Register Configuration

Table 28.27 shows the list of the registers which control the interrupt of the servo section.

Table 28.27 Registers which Control the Interrupt of the Servo Section

| Name | Abbrev. | R/W | Size | Initial Value | Address |
|---------------------------------------|---------|-----|------|---------------|---------|
| Servo interrupt permission register 1 | SIENR1 | R/W | Byte | H'00 | H'FD0B8 |
| Servo interrupt permission register 2 | SIENR2 | R/W | Byte | H'FC | H'FD0B9 |
| Servo interrupt request register 1 | SIRQR1 | R/W | Byte | H'00 | H'FD0BA |
| Servo interrupt request register 2 | SIRQR2 | R/W | Byte | H'FC | H'FD0BB |

28.16.3 Register Description

(1) Servo Interrupt Permission Register 1 (SIENR1)

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | IEDRM3 | IEDRM2 | IEDRM1 | IECAP3 | IECAP2 | IECAP1 | IEHSW2 | IEHSW1 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

SIENR1 controls the permission and prohibition of the interrupt of the servo section. SIENR1 is an 8-bit readable/writable register. It is initialized to H'00 by a reset, stand-by or module stop.

Bit 7: Drum Phase Error Detection Interrupt Permission Bit (IEDRM3)

Bit 7

| IEDRM3 | Description |
|--------|---|
| 0 | Prohibits the interrupt request through IRRDRM3 (Initial value) |
| 1 | Permits the interrupt request through IRRDRM3 |

Bit 6: Drum Speed Error Detection (lock detection) Interrupt Permission Bit (IEDRM2)

Bit 6

| IEDRM2 | Description |
|--------|---|
| 0 | Prohibits the interrupt request through IRRDRM2 (Initial value) |
| 1 | Permits the interrupt request through IRRDRM2 |

Bit 5: Drum Speed Error Detection (OVF, latch) Interrupt Permission Bit (IEDRM1)

Bit 5

| IEDRM1 | Description |
|--------|---|
| 0 | Prohibits the interrupt request through IRRDRM1 (Initial value) |
| 1 | Permits the interrupt request through IRRDRM1 |

Bit 4: Capstan Phase Error Detection Interrupt Permission Bit (IECAP3)

Bit 4

| IECAP3 | Description |
|--------|---|
| 0 | Prohibits the interrupt request through IRRCAP3 (Initial value) |
| 1 | Permits the interrupt request through IRRCAP3 |

Bit 3: Capstan Speed Error Detection (lock detection) Interrupt Permission Bit (IECAP2)

Bit 3

| IECAP2 | Description |
|--------|---|
| 0 | Prohibits the interrupt request through IRRCAP2 (Initial value) |
| 1 | Permits the interrupt request through IRRCAP2 |

Bit 2: Capstan Speed Error Detection (OVF, latch) Interrupt Permission Bit (IECAP1)

Bit 2

| IECAP1 | Description |
|--------|---|
| 0 | Prohibits the interrupt request through IRRCAP1 (Initial value) |
| 1 | Permits the interrupt request through IRRCAP1 |

Bit 1: HSW Timing Generation (counter clear, capture) Interrupt Permission bit (IEHSW2)

Bit 1

| IEHSW2 | Description |
|--------|---|
| 0 | Prohibits the interrupt request through IRRHSW2 (Initial value) |
| 1 | Permits the interrupt request through IRRHSW2 |

Bit 0: HSW Timing Generation (OVW, matching, STRIG) Interrupt Permission bit (IEHSW1)

Bit 0

| IEHSW1 | Description |
|--------|---|
| 0 | Prohibits the interrupt request through IRRHSW1 (Initial value) |
| 1 | Permits the interrupt request through IRRHSW1 |

(2) Servo Interrupt Permission Register 2 (SIENR2)

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | IESNC | IECTL |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W : | — | — | — | — | — | — | R/W | R/W |

SIENR2 controls the permission and prohibition of the interrupt of the servo section. SIENR2 is an 8-bit readable/writable register. It is initialized to H'FC by a reset, stand-by or module stop.

Bits 7 to 2: Reserved

No read or write is valid. If a read is attempted, an undetermined value is read out.

Bit 1: Vertical Sync Signal Interrupt Permission Bit (IESNC)

Bit 1

| IESNC | Description |
|-------|---|
| 0 | Prohibits the interrupt (interrupt to the vertical sync signal) request through IRRSNC (Initial value) |
| 1 | Permits the interrupt request through IRRSNC |

Bit 0: CTL Interrupt Permission Bit (IECTL)

Bit 0

| IECTL | Description |
|-------|---|
| 0 | Prohibits the interrupt request through IRRCTL (Initial value) |
| 1 | Permits the interrupt request through IRRCTL |

(3) Servo Interrupt Request Register 1 (SIRQR1)

| | | | | | | | | |
|-----------------|---------|---------|---------|---------|---------|---------|----------|----------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | IRRDRM3 | IRRDRM2 | IRRDRM1 | IRRCAP3 | IRRCAP2 | IRRCAP1 | IRRHWSW2 | IRRHWSW1 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Note: * Only 0 can be written to clear the flag.

SIRQR1 displays an occurrence of an interrupt request of the servo section. If the interrupt request occurred, the corresponding bit is set to 1.

SIRQR1 is an 8-bit readable/writable register. Writing is allowed only in the case of writing 0 to clear the flag. It is initialized to H'00 by a reset, stand-by or module stop.

Bit 7: Drum Phase Error Detector Interrupt Request Bit (IRRDRM3)

Bit 7

| IRRDRM3 | Description |
|---------|---|
| 0 | No interrupt request from the drum phase error detector (Initial value) |
| 1 | Interrupt requested from the drum phase error detector |

Bit 6: Drum Speed Error Detector (lock detection) Interrupt Request Bit (IRRDRM2)

Bit 6

| IRRDRM2 | Description |
|---------|--|
| 0 | No interrupt request from the drum speed error detector (lock detection) (Initial value) |
| 1 | Interrupt requested from the drum speed error detector (lock detection) |

Bit 5: Drum Speed Error Detector (OVF, latch) Interrupt Request Bit (IRRDRM1)

Bit 5

| IRRDRM1 | Description |
|---------|--|
| 0 | No interrupt request from the drum speed error detector (OVF, latch) (Initial value) |
| 1 | Interrupt requested from the drum speed error detector (OVF, latch) |

Bit 4: Capstan Phase Error Detector Interrupt Request Bit (IRRCAP3)

Bit 4

| IRRCAP3 | Description |
|---------|--|
| 0 | No interrupt request from the capstan phase error detector (Initial value) |
| 1 | Interrupt requested from the capstan phase error detector |

Bit 3: Capstan Speed Error Detector (lock detection) Interrupt Request Bit (IRRCAP2)

Bit 3

| IRRCAP2 | Description |
|---------|---|
| 0 | No interrupt request from the capstan speed error detector (lock detection) (Initial value) |
| 1 | Interrupt requested from the drum speed error detector (lock detection) |

Bit 2: Capstan Speed Error Detector (OVF, latch) Interrupt Request Bit (IRRCAP1)

Bit 2

| IRRCAP1 | Description |
|---------|--|
| 0 | No interrupt request from the capstan speed error detector (OVF, latch)(Initial value) |
| 1 | Interrupt requested from the capstan speed error detector (OVF, latch) |

Bit 1: HSW Timing Generator (counter clear, capture) Interrupt Permission Bit (IRRHSW2)

Bit 1

| IRRHSW2 | Description |
|---------|---|
| 0 | No interrupt request from the HSW timing generator (counter clear, capture) (Initial value) |
| 1 | Interrupt requested from the HSW timing generator (counter clear, capture) |

Bit 0: HSW Timing Generator (OVW, matching, STRIG) Interrupt Permission Bit (IRRHSW1)

Bit 0

| IRRHSW1 | Description |
|---------|---|
| 0 | No interrupt request from the HSW timing generator (OVW, matching, STRIG) (Initial value) |
| 1 | Interrupt requested from the HSW timing generator (OVW, matching, STRIG) |

(4) Servo Interrupt Request Register 2 (SIRQR2)

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | IRRSNC | IRRCTL |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W : | — | — | — | — | — | — | R/(W)* | R/(W)* |

Note: * Only 0 can be written to clear the flag.

SIRQR2 displays an occurrence of an interrupt request of the servo section. If the interrupt request occurred, the corresponding bit is set to 1.
SIRQR2 is an 8-bit readable/writable register. Writing 0 after reading 1 is allowed; no other writing is allowed. It is initialized to H'FC by a reset, stand-by or module stop.

Bits 7 to 2: Reserved

No read or write is valid. If a read is attempted, an undetermined value is read out.

Bit 1: Vertical Sync Signal Interrupt Request Bit (IRRSNC)

Bit 1

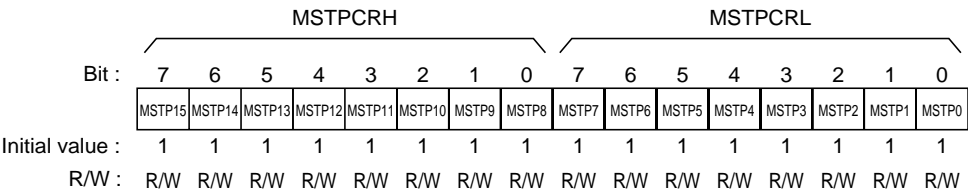
| IRRSNC | Description |
|--------|--|
| 0 | No interrupt request from the sync signal detector (VD, noise) (Initial value) |
| 1 | Interrupt requested from the sync signal detector (VD, noise) |

Bit 0: CTL Signal Interrupt Request Bit (IRRCTL)

Bit 0

| IRRCTL | Description |
|--------|---|
| 0 | No interrupt request from CTL (Initial value) |
| 1 | Interrupt requested from CTL |

28.17 Module Stop Control Reigster (MSTPCR)



MSTPCR comprises two 8-bit readable/writable registers, that perform module stop mode control. When the MSTP1 bit is set to 1, servo circuit and 12-bit PWM stop the operation at the end of the bus cycle and enter to the module stop mode. For details, see 4.5 Module stop mode. MSTPCR is initialized to H'FFFF by a reset.

Bit 1: Module Stop 1 (MSTP1)

This bit specifies module stop mode of the servo circuit and 12-bit PWM.

MSTPCRL

Bit 1

| MSTP1 | Description |
|-------|--|
| 0 | Clear the module stop mode of the Servo Circuit and 12-bit PWM |
| 1 | Set the module stop mode of the Servo Circuit and 12-bit PWM (Initial value) |

Section 29 Electrical Characteristics

29.1 Absolute Maximum Ratings

Table 29.1 lists the absolute maximum ratings.

Table 29.1 Absolute Maximum Ratings

| Item | Symbol | Value | Unit |
|--|------------------|-----------------------------------|------|
| Power supply voltage | V _{cc} | −0.3 to +7.0 | V |
| Input voltage (ports other than port 0) | V _{in} | −0.3 to V _{cc} +0.3 | V |
| Input voltage (port 0) | V _{in} | −0.3 to AV _{cc} +0.3 | V |
| A/D converter power supply voltage | AV _{cc} | −0.3 to +7.0 | V |
| A/D converter input voltage | AV _{in} | −0.3 to AV _{cc} +0.3 | V |
| Servo power supply voltage | SV _{cc} | −0.3 to +7.0 | V |
| Servo amplifier input voltage | V _{in} | −0.3 to SV _{cc} + 0.3 | V |
| Operating temperature | T _{opr} | −20 to +75 | °C |
| Operating temperature (At Flash memory program/erase) | T _{opr} | 0 to +75 | °C |
| Storage temperature | T _{str} | −55 to +125 | °C |

- Notes: 1. Permanent damage may occur to the chip if absolute maximum ratings are exceeded. Normal operation should be under the conditions specified in Electrical Characteristics. Exceeding these values can result in incorrect operation and reduced reliability.
2. All voltages are relative to V_{ss} = SV_{ss} = AV_{ss} = 0.0 V.

29.2 Electrical Characteristics of HD64F2194

29.2.1 DC Characteristics of HD64F2194

Table 29.2 DC Characteristics of HD64F2194, HD64F2194C

(Conditions: $V_{cc} = AV_{cc} = 4.0$ to 5.5 V, $V_{ss} = 0.0$ V, $T_a = -20$ to $+75^\circ\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|--------------------|----------|---|-------------------------------|--------------|-----|--------------|------|-------|
| | | | | Min | Typ | Max | | |
| Input high voltage | V_{IH} | MD0 | $V_{cc}=2.7$ to 5.5V | $0.9 V_{cc}$ | — | $V_{cc}+0.3$ | V | |
| | | \overline{RES} , \overline{NMI} , FWE, \overline{IC} , $\overline{IRQ0}$ to $\overline{IRQ5}$ | | $0.8 V_{cc}$ | — | $V_{cc}+0.3$ | | |
| | | | $V_{cc}=2.7$ to 5.5V | $0.9 V_{cc}$ | — | $V_{cc}+0.3$ | | |
| | | SCK1, SCK2, SI1, SI2, \overline{CS} , FTIA, FTIB, FTIC, FTID, TRIG, TMBI, \overline{ADTRG} | | $0.8 V_{cc}$ | — | $V_{cc}+0.3$ | | |
| | | OSC1, X1 | | $V_{cc}-0.5$ | — | $V_{cc}+0.3$ | | |
| | | | $V_{cc}=2.7$ to 5.5V | $V_{cc}-0.3$ | — | $V_{cc}+0.3$ | | |
| | | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, PS0 to PS4 | | $0.7 V_{cc}$ | — | $V_{cc}+0.3$ | | |
| | | | $V_{cc}=2.7$ to 5.5V | $0.8 V_{cc}$ | — | $V_{cc}+0.3$ | | |
| | | Csync | | $0.7 V_{cc}$ | — | $V_{cc}+0.3$ | | |

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|---------------------|----------|---|------------------------|--------------|--------------|--------------|------|-------------------------|
| | | | | Min | Typ | Max | | |
| Input low voltage | V_{IL} | MD0 | $V_{CC}=2.7$ to $5.5V$ | -0.3 | — | $0.1 V_{CC}$ | V | |
| | | \overline{RES} , \overline{NMI} , \overline{FWE} , \overline{IC} , $\overline{IRQ0}$ to $\overline{IRQ5}$ | $V_{CC}=2.7$ to $5.5V$ | -0.3 | — | $0.2 V_{CC}$ | | |
| | | SCK1, SCK2, SI1, SI2, \overline{CS} , FTIA, FTIB, FTIC, FTID, TRIG, TMBI, \overline{ADTRG} | | -0.3 | — | $0.2 V_{CC}$ | | |
| | | OSC1, X1 | | -0.3 | — | 0.5 | | |
| | | | $V_{CC}=2.7$ to $5.5V$ | -0.3 | — | 0.3 | | |
| | | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, PS0 to PS4 | $V_{CC}=2.7$ to $5.5V$ | -0.3 | — | $0.3 V_{CC}$ | | |
| | | | | -0.3 | — | $0.2 V_{CC}$ | | |
| Output high voltage | V_{OH} | SO1, SO2, SCK1, SCK2, PWM1, PWM2, PWM3, PWM4, PWM14, STRB, BUZZ, TMO, TMOW, FTOA, FTOB, PPG70 to PPG77, RP0 to RP7, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, PS0 to PS4 | $-I_{OH}=1.0mA$ | $V_{CC}-1.0$ | — | — | V | |
| | | | $-I_{OH}=0.5mA$ | — | $V_{CC}-0.5$ | — | V | Refer- ence value |
| | | | $-I_{OH}=0.1mA$ | $V_{CC}-0.5$ | — | — | V | |
| | | | $V_{CC}=2.7$ to $5.5V$ | | | | | |

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|-------------------------------|------------|--|--|--------|-----|-----|---------------|-------|
| | | | | Min | Typ | Max | | |
| Output low voltage | V_{OL} | SO1, SO2, SCK1, SCK2, PWM1, PWM2, PWM3, PWM4, PWM14, STRB, BUZZ, TMO, TMOW, FTOA, FTOB, PPG70 to PPG77, RP0 to RP7, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, PS0 to PS4 | $I_{OL}=1.6\text{mA}$ | — | — | 0.6 | V | |
| | | | $I_{OL}=0.4\text{mA}$ $V_{CC}=2.7$ to 5.5V | — | — | 0.4 | V | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | P80 to P87, | $I_{OL}=20\text{mA}$ | — | — | 1.5 | V | |
| | | | $I_{OL}=1.6\text{mA}$ | — | — | 0.6 | V | |
| | | | | | | | | |
| | | | | | | | | |
| | | | $I_{OL}=0.4\text{mA}$ $V_{CC}=2.7$ to 5.5V | — | — | 0.4 | V | |
| Input /output leakage current | $ I_{IL} $ | MD0, OSC1 | $V_{in}=0.5$ to $V_{CC}-0.5\text{V}$ | — | — | 1.0 | μA | |
| | | $\overline{\text{RES}}$, $\overline{\text{NMI}}$, FWE, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ5}}$, $\overline{\text{IC}}$ | | | | | | |
| | | SCK1, SCK2, SI1, SI2, $\overline{\text{CS}}$, FTIA, FTIB, FTIC, FTID, TRIG, TMBI, ADTRG | $V_{in}=0.5$ to $V_{CC}-0.5\text{V}$ | — | — | 1.0 | | |
| | | P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P53, P60 to P67, P70 to P77, P80 to P87, PS0 to PS4 | $V_{in}=0.5$ to $V_{CC}-0.5\text{V}$ | — | — | 1.0 | | |
| | | P00 to P07, AN8 to ANB | $V_{in}=0.5$ to $A V_{CC}-0.5\text{V}$ | — | — | 1.0 | | |

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|---|---------------------|--|--|--------|-----|-----|------|-------------------------|
| | | | | Min | Typ | Max | | |
| Pull-up MOS current | -I _p | P10 to P17, P20 to P27, P30 to P37, | V _{cc} =5.0V, V _{in} =0V | 50 | — | 300 | μA | Note 1 |
| Input capacitance | C _{in} | All input pins except power supply, P13, P23, P24 and analog system pins | f _{in} =1 MHz, V _{in} =0V, T _a =25°C | — | — | 15 | pF | |
| | | P13, P23, P24 | f _{in} =1 MHz, V _{in} =0V, T _a =25°C | — | — | 20 | pF | |
| Active mode current dissipation (CPU operating) | I _{OPE} | V _{cc} | V _{cc} =5V, f _{osc} =10 MHz, High-speed mode | — | 50 | 70 | mA | Note 2 |
| | | | V _{cc} =5V, f _{osc} =10 MHz, Medium-speed mode (1/64) | — | 35 | — | mA | Reference value |
| Active mode current dissipation (reset) | I _{RES} | V _{cc} | V _{cc} =5V, f _{osc} =10 MHz | — | 30 | 45 | mA | Note 2 |
| Sleep mode current dissipation | I _{SLEEP} | V _{cc} | V _{cc} =5V, f _{osc} =10 MHz High-speed mode | — | 20 | 30 | mA | Note 2 |
| Subactive mode current dissipation | I _{SUB} | V _{cc} | V _{cc} =2.7V, With 32kHz crystal oscillator (φ _{sub} =φ _w /2) | — | 90 | 150 | μA | Note 2 |
| | | | V _{cc} =2.7V, With 32kHz crystal oscillator (φ _{sub} =φ _w /8) | — | 40 | — | | Reference value, Note 2 |
| Subsleep mode current dissipation | I _{SUBSLP} | V _{cc} | V _{cc} =2.7V, With 32kHz crystal oscillator (φ _{sub} =φ _w /2) | — | 15 | 30 | μA | Note 2 |
| | | | V _{cc} =2.7V, With 32kHz crystal oscillator (φ _{sub} =φ _w /8) | — | 10 | — | | Reference value, Note 2 |

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|--|--------------------|-----------------|--|--------|-----|-----|------|---------------------------|
| | | | | Min | Typ | Max | | |
| Watch mode current dissipation | I _{WATCH} | Vcc | Vcc=2.7V, With 32kHz crystal oscillator | — | 5 | 10 | μA | Note 2 |
| | | | Vcc=5.0V, With 32kHz crystal oscillator | — | 10 | — | μA | Reference value Note 2 |
| Standby mode current dissipation | I _{STBY} | Vcc | X1=Vcc, Without 32kHz crystal oscillator | — | — | 5 | μA | Note 2 |
| RAM data retaining voltage in standby mode | V _{STBY} | | | 2.0 | — | — | V | |

Notes: Do not open the AVcc and AVss pin even when the A/D converter is not in use.

1. Current value when the relevant bit of the pull-up MOS select register (PUR1 to PUR3) is set to 1.
2. The current on the pull-up MOS or the output buffer excluded.

Table 29.3 Pin Status at Current Dissipation Measurement

| Mode | RES pin | Internal State | Pin | Oscillator Pin |
|---|---------|---------------------------------|-----|---|
| Active mode High-speed, medium-speed | Vcc | Operating | Vcc | Main clock: Crystal oscillator Sub clock: X1 pin = Vcc |
| Sleep mode High-speed, medium-speed | Vcc | CPU and servo circuits stopped. | Vcc | |
| Reset | Vss | Reset | Vcc | |
| Standby mode | Vcc | All stopped | Vcc | |
| Subactive mode | Vcc | CPU and timer A operating | Vcc | Main clock: Crystal oscillator Sub clock: Crystal oscillator |
| Subsleep mode | Vcc | Timer A operating | Vcc | |
| Watch mode | Vcc | Timer A operating | Vcc | |

Table 29.4 Bus Drive Characteristics of HD64F2194, HD64F2194C

(Conditions: $V_{cc} = AV_{cc} = 4.0$ to 5.5 V, $V_{ss} = 0.0$ V, $T_a = -20$ to $+75^\circ\text{C}$ unless otherwise specified.) Applicable pin: SCL, SDA

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|-------------------------------|---------------------|-----------------|---------------------|---------------------|-----|----------------------|------|-------|
| | | | | Min | Typ | Max | | |
| Schmidt trigger input voltage | V_T^- | SCL, SDA | | 0.2V _{cc} | — | — | V | |
| | V_T^+ | | | — | — | 0.7V _{cc} | V | |
| | V_T^+ $-V_T^-$ | | | 0.05V _{cc} | — | — | V | |
| Input High voltage | V_{IH} | SCL, SDA | | 0.7V _{cc} | — | V _{cc} +0.5 | V | |
| Input Low voltage | V_{IL} | SCL, SDA | | −0.5 | — | 0.2V _{cc} | V | |
| Output Low voltage | V_{OL} | SCL, SDA | $I_{OL}=8\text{mA}$ | — | — | 0.5 | V | |
| | | | $I_{OL}=3\text{mA}$ | — | — | 0.4 | | |
| SCL and SDA output fall time | t_{of} | SCL, SDA | | 20+ 0.1Cb | — | 250 | ns | |

29.2.2 Allowable Output Currents of HD64F2194, HD64F2194C

The specifications for the digital pins are shown below.

Table 29.5 Allowable Output Currents

(Conditions: $V_{cc} = 2.7$ to 5.5 V, $V_{ss} = 0.0$ V, $T_a = -20$ to $+75^\circ\text{C}$)

| Item | Symbol | Value | Unit | Notes |
|--|---------------|-------|------|-------|
| Allowable input current (to chip) | I_o | 2 | mA | 1 |
| Allowable input current (to chip) | I_o | 22 | mA | 2 |
| Allowable input current (to chip) | I_o | 10 | mA | 3 |
| Allowable output current (from chip) | $-I_o$ | 2 | mA | 4 |
| Total allowable input current (to chip) | ΣI_o | 80 | mA | 5 |
| Total allowable output current (from chip) | $-\Sigma I_o$ | 50 | mA | 6 |

- Notes:
1. The allowable input current is the maximum value of the current flowing from each I/O pin to V_{ss} (except for port 8, SCL and SDA).
 2. The allowable input current is the maximum value of the current flowing from each I/O pin to V_{ss} . This applies to port 8.
 3. The allowable input current is the maximum value of the current flowing from each I/O pin to V_{ss} . This applies to SCL and SDA.
 4. The allowable output current is the maximum value of the current flowing from V_{cc} to each I/O pin.
 5. The total allowable input current is the sum of the currents flowing from all I/O pins to V_{ss} simultaneously.
 6. The total allowable output current is the sum of the currents flowing from V_{cc} to all I/O pins.

29.2.3 AC Characteristics of HD64F2194, HD64F2194C

Table 29.6 AC Characteristics of HD64F2194, HD64F2194C

(Conditions: $V_{CC} = AV_{CC} = 4.0$ to 5.5 V, $V_{SS} = AV_{SS} = 0.0$ V, $T_a = -20$ to $+75^\circ\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|---|--------------|-----------------|---|--------|--------|-----|---------------|-------------|
| | | | | Min | Typ | Max | | |
| Clock oscillation frequency | f_{OSC} | OSC1, OSC2 | | 8 | — | 10 | MHz | |
| Clock cycle time | t_{cyc} | OSC1, OSC2 | | 100 | — | 125 | ns | Figure 29.1 |
| Subclock oscillation frequency | f_x | X1, X2 | $V_{CC}=2.7$ to 5.5 V | — | 32.768 | — | kHz | |
| Subclock cycle time | t_{subcyc} | X1, X2 | $V_{CC}=2.7$ to 5.5 V | — | 30.518 | — | μs | Figure 29.2 |
| Oscillation stabilization time | t_{rc} | OSC1, OSC2 | Crystal oscillator | — | — | 10 | ms | |
| | | X1, X2 | 32kHz crystal oscillator $V_{CC}=2.7$ to 5.5 V | — | — | 2 | s | |
| External clock high width | t_{CPH} | OSC1 | | 40 | — | — | ns | Figure 29.1 |
| External clock low width | t_{CPL} | OSC1 | | 40 | — | — | ns | |
| External clock rise time | t_{CPr} | OSC1 | | — | — | 10 | ns | |
| External clock fall time | t_{CPl} | OSC1 | | — | — | 10 | ns | |
| External clock stabilization delay time | t_{DEXT} | OSC1 | | 500 | — | — | μs | Figure 29.3 |
| Subclock input low level pulse width | t_{EXCLL} | X1 | $V_{CC}=2.7$ to 5.5 V | — | 15.26 | — | μs | Figure 29.2 |
| Subclock input high level pulse width | t_{EXCLH} | X1 | $V_{CC}=2.7$ to 5.5 V | — | 15.26 | — | μs | |
| Subclock input rise time | t_{EXCLr} | X1 | $V_{CC}=2.7$ to 5.5 V | — | — | 10 | ns | |
| Subclock input fall time | t_{EXCLf} | X1 | $V_{CC}=2.7$ to 5.5 V | — | — | 10 | ns | |

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Figure |
|---|------------------|---|--|--------|-----|-----|---|-------------|
| | | | | Min | Typ | Max | | |
| $\overline{\text{RES}}$ pin low level width | t_{REL} | RES | $V_{\text{CC}}=2.7\text{V}$ to 5.5V | 20 | — | — | t_{cyc} | Figure 29.4 |
| Input pin high level width | t_{IH} | $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ5}}$, $\overline{\text{NMI}}$, $\overline{\text{IC}}$, $\overline{\text{ADTRG}}$, TMBI, FTIA, FTIB, FTIC, FTID, TRIG | $V_{\text{CC}}=2.7\text{V}$ to 5.5V | 2 | — | — | t_{cyc} t_{subcyc} | Figure 29.5 |
| Input pin low level width | t_{IL} | $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ5}}$, $\overline{\text{NMI}}$, $\overline{\text{IC}}$, $\overline{\text{ADTRG}}$, TMBI, FTIA, FTIB, FTIC, FTID, TRIG | $V_{\text{CC}}=2.7\text{V}$ to 5.5V | 2 | — | — | t_{cyc} t_{subcyc} | |

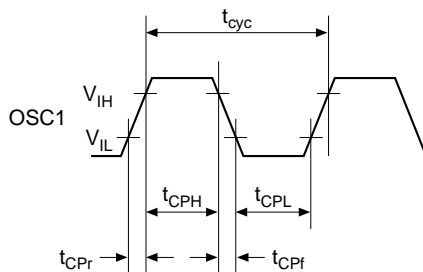


Figure 29.1 System Clock Timing

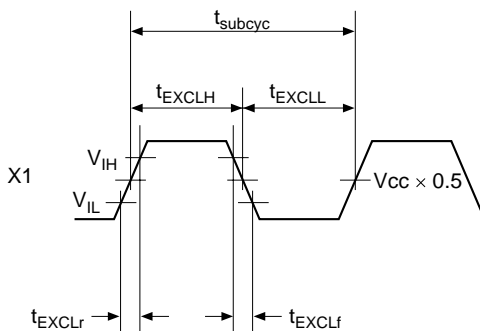


Figure 29.2 Subclock Input Timing

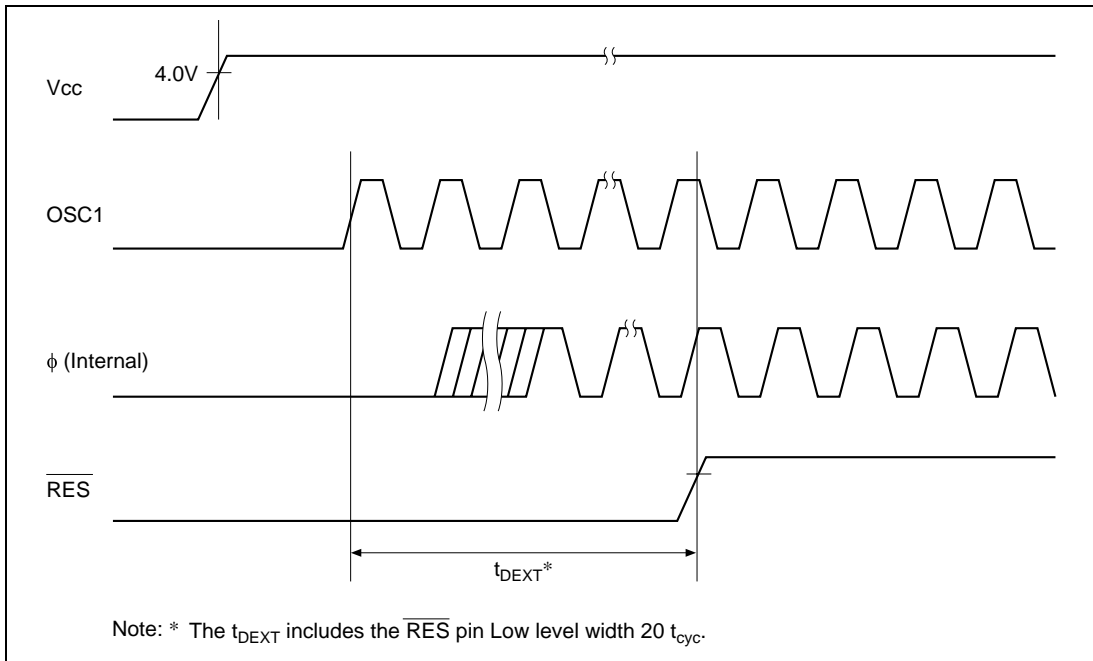


Figure 29.3 External Clock Stabilization Delay Timing

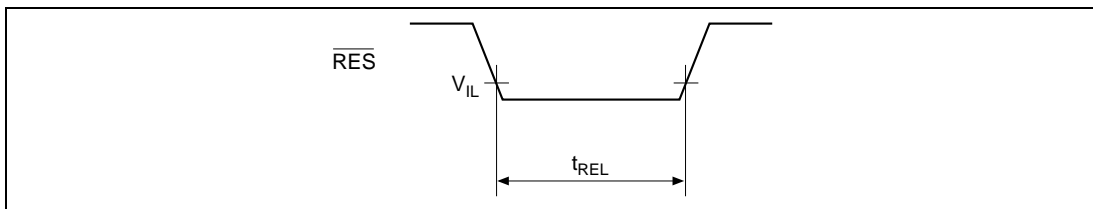


Figure 29.4 Reset Input Timing

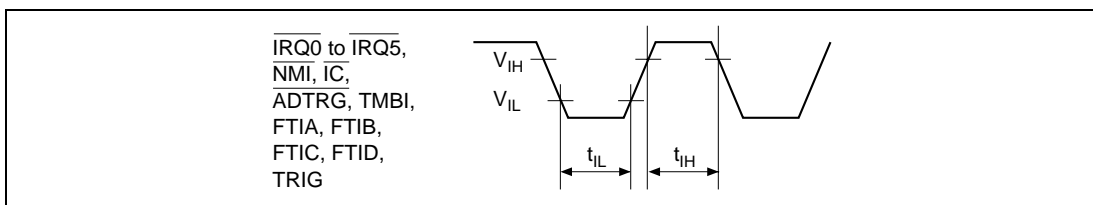


Figure 29.5 Input Timing

29.2.4 Serial Interface Timing of HD64F2194, HD64F2194C

Table 29.7 Serial Interface Timing of HD64F2194, HD64F2194C

(Conditions: $V_{cc} = AV_{cc} = 4.0$ to 5.5 V, $V_{ss} = AV_{ss} = 0.0$ V, $T_a = -20$ to $+75^{\circ}\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Figure |
|--|-------------------|------------------------|--------------------------|--------|-----|-----|-------------------|----------------|
| | | | | Min | Typ | Max | | |
| Input clock cycle | t_{scyc} | SCK1 | Asynchroniza- tion | 4 | — | — | t_{cyc} | Figure 29.6 |
| | | | Clock synchronization | 6 | — | — | | |
| | | SCK2 | | 2 | — | — | | |
| Input clock pulse width | t_{SCKW} | SCK1, SCK2 | | 0.4 | — | 0.6 | t_{scyc} | |
| Input clock rise time | t_{SCKr} | SCK1 | | — | — | 1.5 | t_{cyc} | |
| | | SCK2 | | — | — | 60 | ns | |
| Input clock fall time | t_{SCKf} | SCK1 | | — | — | 1.5 | t_{cyc} | |
| | | SCK2 | | — | — | 60 | ns | |
| Transmit data delay time (clock sync) | t_{TXD} | SO1 | | — | — | 100 | ns | Figure 29.7 |
| Receive data setup time (clock sync) | t_{RXS} | SI1 | | 100 | — | — | ns | |
| Receive data hold time (clock sync) | t_{RXH} | SI1 | | 100 | — | — | ns | |
| Transmit data output delay time | t_{TXD} | SO2 | | — | — | 200 | ns | Figure 29.7 |
| Receive data setup time (clock sync) | t_{RXS} | SI2 | | 180 | — | — | ns | |
| Receive data hold time (clock sync) | t_{RXH} | SI2 | | 180 | — | — | ns | |
| $\overline{\text{CS}}$ setup time | t_{CSS} | $\overline{\text{CS}}$ | | 1 | — | — | t_{scyc} | Figure 29.8 |
| $\overline{\text{CS}}$ hold time | t_{CSH} | $\overline{\text{CS}}$ | | 1 | — | — | t_{scyc} | |

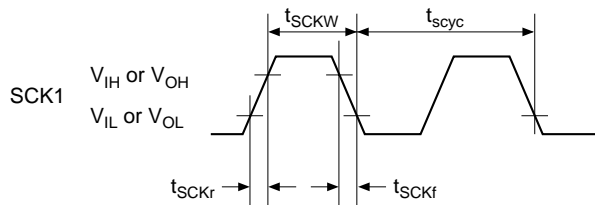


Figure 29.6 SCK1 Clock Timing

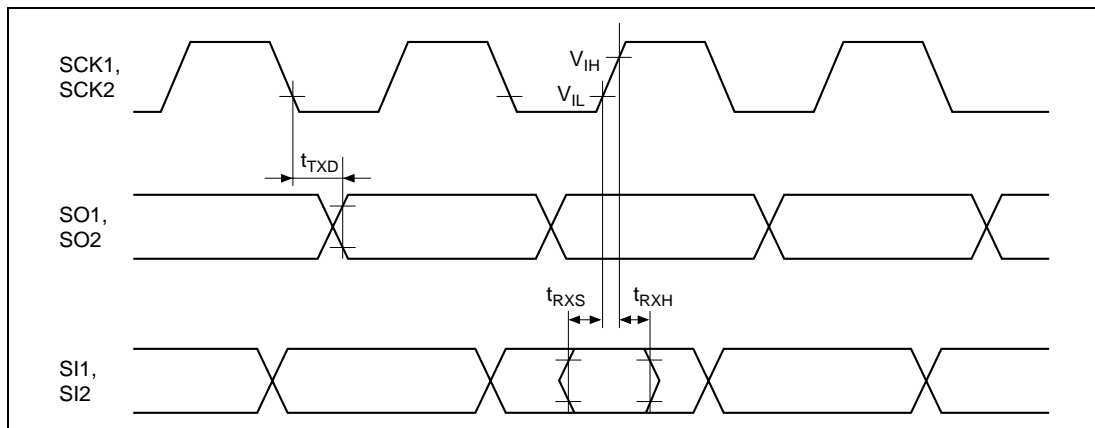


Figure 29.7 SCI I/O Timing/Clock Synchronization Mode

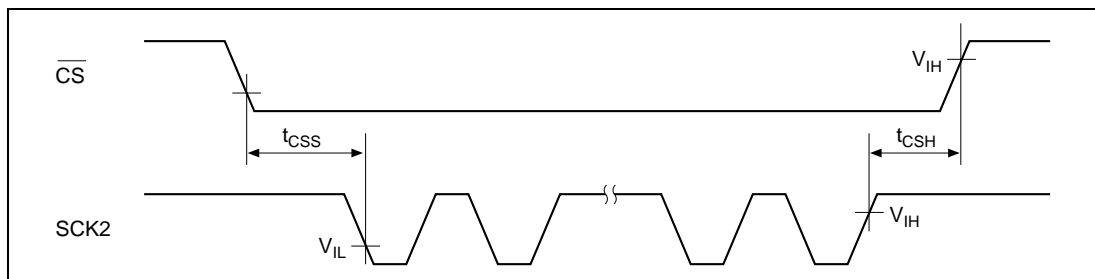


Figure 29.8 SCI2 Chip Select Timing

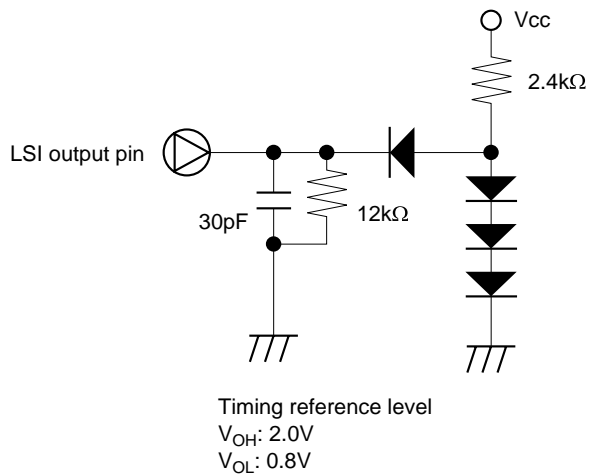


Figure 29.9 Output Load Conditions

Table 29.8 I²C Bus Interface Timing of HD64F2194, HD64F2194C

(Conditions: $V_{CC} = AV_{CC} = 4.0$ to 5.5 V, $V_{SS} = AV_{SS} = 0.0$ V, $T_a = -20$ to $+75^{\circ}\text{C}$ unless otherwise specified.)

| Item | Symbol | Test Conditions | Values | | | Unit | Figure |
|--|------------|-----------------|--------|-----|------------|-----------|--------------|
| | | | Min | Typ | Max | | |
| SCL input cycle time | t_{SCL} | | 12 | — | — | t_{cyc} | Figure 29.10 |
| SCL input high pulse width | t_{SCLH} | | 3 | — | — | t_{cyc} | |
| SCL input low pulse width | t_{SCLL} | | 5 | — | — | t_{cyc} | |
| SCL, SDA input rise time | t_{sr} | | — | — | 7.5^{*1} | t_{cyc} | |
| SCL, SDA input fall time | t_{sf} | | — | — | 300 | ns | |
| SCL, SDA input spike pulse removal time | t_{sp} | | — | — | 1 | t_{cyc} | |
| SDA input bus free time | t_{BUF} | | 5 | — | — | t_{cyc} | |
| Start condition input hold time | t_{STAH} | | 3 | — | — | t_{cyc} | |
| Re-transmit start condition input setup time | t_{STAS} | | 3 | — | — | t_{cyc} | |
| Stop condition input setup time | t_{STOS} | | 3 | — | — | t_{cyc} | |
| Data input setup time | t_{SDAS} | | 0.5 | — | — | t_{cyc} | |
| Data input hold time | t_{SDAH} | | 0 | — | — | ns | |
| SCL, SDA capacity load | C_b | | — | — | 400 | pF | |

Note: 1. Can also be set to $17.5 t_{cyc}$ depending on the selection of clock to be used by the I²C module.

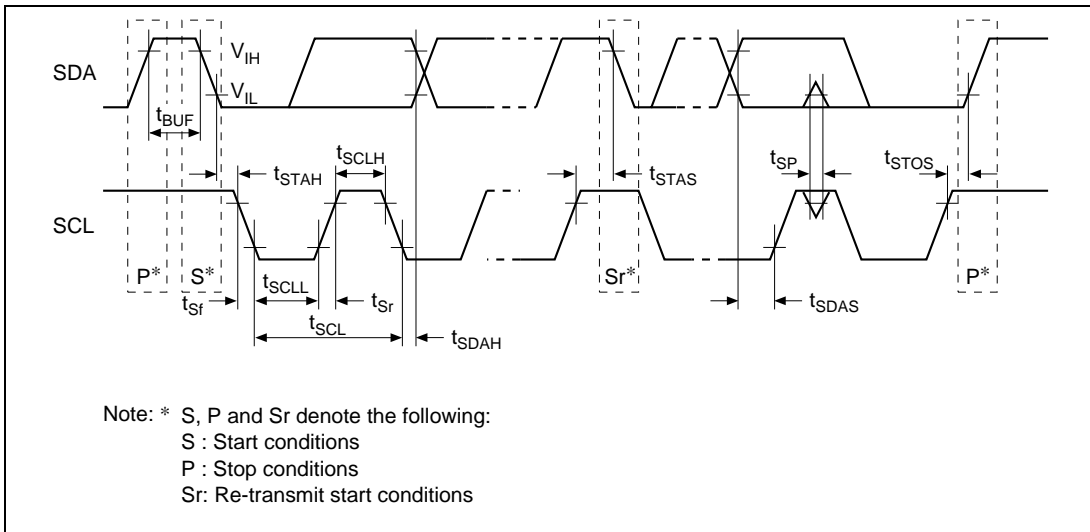


Figure 29.10 I²C Bus Interface I/O Timing

29.2.5 A/D Converter Characteristics of HD64F2194, HD64F2194C

Table 29.9 A/D Converter Characteristics of HD64F2194, HD64F2194C

(Conditions: $V_{CC} = AV_{CC} = 4.0$ to 5.5 V, $V_{SS} = AV_{SS} = 0.0$ V, $T_a = -20$ to $+75^\circ\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Note |
|-----------------------------------|-------------|------------------------|--|--------------|----------|--------------|------------------|------|
| | | | | Min | Typ | Max | | |
| Analog power supply voltage | AV_{CC} | AV_{CC} | | $V_{CC}-0.3$ | V_{CC} | $V_{CC}+0.3$ | V | |
| Analog input voltage | A_{VIN} | AN0 to AN7, AN8 to ANB | | AV_{SS} | — | AV_{CC} | V | |
| Analog power supply current | A_{ICC} | AV_{CC} | $AV_{CC}=5.0\text{V}$ | — | — | 2.0 | mA | |
| | A_{ISTOP} | AV_{CC} | $V_{CC}=2.7$ to 5.5V At reset and in power-down mode | — | — | 10 | μA | |
| Analog input capacitance | C_{AIN} | AN0 to AN7, AN8 to ANB | | — | — | 30 | pF | |
| Allowable signal source impedance | R_{AIN} | AN0 to AN7, AN8 to ANB | | — | — | 10 | $\text{k}\Omega$ | |
| Resolution | | | | — | — | 10 | Bit | |
| Absolute accuracy | | | $AV_{CC}=5.0\text{V}$ | — | — | ± 4 | LSB | |
| Conversion time | | | | 13.4 | — | 26.6 | μs | |

Note: Do not open the AV_{CC} and AV_{SS} pin even when the A/D converter is not in use. Set $AV_{CC} = V_{CC}$ and $AV_{SS} = V_{SS}$.

29.2.6 Servo Section Electrical Characteristics of HD64F2194, HD64F2194C

Table 29.10 Servo Section Electrical Characteristics of HD64F2194, HD64F2194C
(reference values)

(Conditions: $V_{cc} = SV_{cc} = 5.0\text{ V}$, $V_{ss} = SV_{ss} = 0.0\text{ V}$, $T_a = 25^\circ\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Reference Values | | | Unit | Note |
|-------------------------------------|--------|-----------------|--|------------------|---------------|------|------------|------|
| | | | | Min | Typ | Max | | |
| PB-CTL input amplifier voltage gain | | CTL (+) | CTLGR3=0, CTLGR2=0, CTLGR1=0, CTLGR0=0, $f=10\text{kHz}$ | 32.0 | 34.0 | 36.0 | dB | |
| | | | CTLGR3=0, CTLGR2=0, CTLGR1=0, CTLGR0=1, $f=10\text{kHz}$ | 34.5 | 36.5 | 38.5 | | |
| | | | CTLGR3=0, CTLGR2=0, CTLGR1=1, CTLGR0=0, $f=10\text{kHz}$ | 37.0 | 39.0 | 41.0 | | |
| | | | CTLGR3=0, CTLGR2=0, CTLGR1=1, CTLGR0=1, $f=10\text{kHz}$ | 39.5 | 41.5 | 43.5 | | |
| | | | CTLGR3=0, CTLGR2=1, CTLGR1=0, CTLGR0=0, $f=10\text{kHz}$ | 42.0 | 44.0 | 46.0 | | |
| | | | CTLGR3=0, CTLGR2=1, CTLGR1=0, CTLGR0=1, $f=10\text{kHz}$ | 44.5 | 46.5 | 48.5 | | |
| | | | CTLGR3=0, CTLGR2=1, CTLGR1=1, CTLGR0=0, $f=10\text{kHz}$ | 47.0 | 49.0 | 51.0 | | |
| | | | CTLGR3=0, CTLGR2=1, CTLGR1=1, CTLGR0=1, $f=10\text{kHz}$ | 49.5 | 51.5 | 53.5 | | |
| | | | CTLGR3=1, CTLGR2=0, CTLGR1=0, CTLGR0=0, $f=10\text{kHz}$ | 52.0 | 54.0 | 56.0 | | |
| | | | CTLGR3=1, CTLGR2=0, CTLGR1=0, CTLGR0=1, $f=10\text{kHz}$ | 54.5 | 56.5 | 58.5 | | |
| | | | CTLGR3=1, CTLGR2=0, CTLGR1=1, CTLGR0=0, $f=10\text{kHz}$ | 57.0 | 59.0 | 61.0 | | |
| | | | CTLGR3=1, CTLGR2=0, CTLGR1=1, CTLGR0=1, $f=10\text{kHz}$ | 59.5 | 61.5 | 63.5 | | |
| | | | CTLGR3=1, CTLGR2=1, CTLGR1=0, CTLGR0=0, $f=10\text{kHz}$ | 62.0 | 64.0 | 66.0 | | |
| | | | CTLGR3=1, CTLGR2=1, CTLGR1=0, CTLGR0=1, $f=10\text{kHz}$ | 64.5 | 66.5 | 68.5 | | |
| | | | CTLGR3=1, CTLGR2=1, CTLGR1=1, CTLGR0=0, $f=10\text{kHz}$ | 67.0 | 69.0 | 71.0 | | |
| | | | CTLGR3=1, CTLGR2=1, CTLGR1=1, CTLGR0=1, $f=10\text{kHz}$ | 69.5 | 71.5 | 73.5 | | |
| PB-CTL Schmidt input | V+TH | CTLSMT (i) | AC coupling, $C=0.1\mu\text{F}$ Typ (non pol) | — | 250 | — | mVp | |
| | V-TH | | AC coupling, $C=0.1\mu\text{F}$ Typ (non pol) | — | -250 | — | | |
| Analog switch ON resistance | REB | | | — | 150 | — | Ω | |
| REC-CTL output current | ICTL | CTL (+) | Series resistance = $0\ \Omega$ | — | 8 | — | mA | |
| | | CTL (-) | | — | 8 | — | | |
| REC-CTL inter-pin resistance | RCTL | | | — | 10 | — | k Ω | |
| CTL reference output voltage | | CTLREF | | — | 1/2 SV_{cc} | — | V | |

| Item | Symbol | Applicable Pins | Test Conditions | Reference Values | | | Unit | Note |
|---|-----------------|-----------------|--|------------------|------------------|-----|-----------------|------|
| | | | | Min | Typ | Max | | |
| CFG pin bias voltage | | CFG | | — | 1/2 SV_{CC} | — | V | |
| CFG input level | | CFG | AC coupling, $C=1\mu F$ Typ, $f=1kHz$ | 1.0 | — | — | V _{pp} | |
| CFG input impedance | | CFG | | — | 10 | — | k Ω | |
| CFG input threshold voltage | V+THCF | CFG | Rise threshold level | — | 2.25 | — | V | |
| | V−THCF | | Fall threshold level | — | 2.75 | — | | |
| DFG Schmidt input | V+THDF | DFG | Rising edge Schmidt level | — | 1.95 | — | V | |
| | V−THDF | | Falling edge Schmidt level | — | 1.85 | — | | |
| DPG Schmidt input | V+THDP | DPG | Rising edge Schmidt level | — | 3.55 | — | V | |
| | V−THDP | | Falling edge Schmidt level | — | 3.45 | — | | |
| 3-level output voltage | V _{OH} | Vpulse | −I _{OH} =0.1mA | 4.0 | — | — | V | |
| | V _{OM} | | No load, H _{iz} =1 | — | 2.5 | — | | |
| | V _{OL} | | I _{OL} =0.1mA | — | — | 1.0 | | |
| 3-level output pin divided voltage resistance | | Vpulse | | — | 15 | — | k Ω | |
| CFG Duty | | CFG | AC coupling, $C=1\mu F$ Typ, $f=1kHz$ | 48 | — | 52 | % | |

Table 29.11 Servo Section Electrical Characteristics of HD64F2194, HD64F2194C(Conditions: $V_{CC} = SV_{CC} = 5.0\text{ V}$, $V_{SS} = SV_{SS} = 0.0\text{ V}$, $T_a = 25^\circ\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Note |
|-----------------------------|------------|---|-------------------------|------------------|-----|----------------|------|------|
| | | | | Min | Typ | Max | | |
| Digital input high voltage | V_{IH} | COMP, EXCTL, EXCAP, EXTTRG | | 0.8 V_{CC} | — | $V_{CC} + 0.3$ | V | |
| Digital input low voltage | V_{IL} | | | −0.3 | — | $0.2 V_{CC}$ | | |
| Digital output high voltage | V_{OH} | H.AmpSW, C.Rotary, VIDEOFF, AUDIOFF, DRMPWM, CAPPWM, SV1, SV2 | $-I_{OH} = 1\text{mA}$ | V_{CC} −1.0 | — | — | V | |
| Digital output low voltage | V_{OL} | | $I_{OL} = 1.6\text{mA}$ | — | — | 0.6 | | |
| Current dissipation | I_{CCSV} | SVcc | At no load | — | 5 | 10 | mA | |

29.2.7 FLASH Memory Characteristics

Table 29.12 shows the flash memory characteristics.

Table 29.12 Flash Memory Characteristics (Preliminary)

Conditions: $V_{CC} = 5.0 \text{ V} \pm 10\%$, $AV_{CC} = 5.0 \text{ V} \pm 10\%$, $V_{SS} = AV_{SS} = 0 \text{ V}$, $T_a = 0 \text{ to } +75^\circ\text{C}$
(operating temperature range at programming/erasing)

| Item | | Symbol | Test conditions | Min | Typ | Max | Unit |
|------------------------------------|--|---------------|----------------------------|-----|-----|------|-----------------|
| Programming time ^{*1+2+4} | | t_p | | — | 10 | 200 | ms/ 32 bytes |
| Erasing time ^{*1+3+5} | | t_E | | — | 100 | 1200 | ms/ block |
| No. of reprogramming | | N_{WEC} | | — | — | 100 | Times |
| At programming | Wait time after SWE-bit setting ^{*1} | x | | 10 | — | — | μs |
| | Wait time after PSU-bit setting ^{*1} | y | | 50 | — | — | μs |
| | Wait time after P-bit setting ^{*1+4} | z | | 150 | — | 200 | μs |
| | Wait time after P-bit clearing ^{*1} | α | | 10 | — | — | μs |
| | Wait time after PSU-bit clearing ^{*1} | β | | 10 | — | — | μs |
| | Wait time after PV-bit setting ^{*1} | γ | | 4 | — | — | μs |
| | Wait time after dummy write ^{*1} | ε | | 2 | — | — | μs |
| | Wait time after PV-bit clearing ^{*1} | η | | 4 | — | — | μs |
| | Maximum No. of programmings ^{*1+4+5} | N | When z = 200 μs | — | — | 1000 | Times |
| At erasing | Wait time after SWE-bit setting ^{*1} | x | | 10 | — | — | μs |
| | Wait time after ESU-bit setting ^{*1} | y | | 200 | — | — | μs |
| | Wait time after E-bit setting ^{*1+6} | z | | 5 | — | 10 | ms |
| | Wait time after E-bit clearing ^{*1} | α | | 10 | — | — | μs |
| | Wait time after ESU-bit clearing ^{*1} | β | | 10 | — | — | μs |
| | Wait time after EV-bit setting ^{*1} | γ | | 20 | — | — | μs |
| | Wait time after dummy write ^{*1} | ε | | 2 | — | — | μs |
| | Wait time after EV-bit clearing ^{*1} | η | | 5 | — | — | μs |
| | Maximum No. of erasings ^{*1+6} | N | | 120 | — | 240 | Times |

- Notes: 1. Perform each time setting according to the programming/erasing algorithm.
2. Programming time per 32 bytes (total time of setting P-bit of the flash memory control register. Programming verify time is not included).

3. Time to erase 1 block (total time of setting E-bit of the flash memory control register. Erasing verify time is not included).
4. Maximum programming time (t_p (max.)) = Wait time after P-bit setting (z) \times Maximum No. of programming (N)
5. No. of times when wait time after P-bit setting (z) = 200 μ s. Set maximum No. of programming shall be set less than maximum programming time (t_p (max.)) according to the actual setting (z).
6. Relationship between wait time after E-bit setting (z) and maximum No. of erasing (N) for maximum erasing time (t_e (max.)) is as follows:
 t_e (max.) = Wait time after E-bit setting \times Maximum No. of erasing (N)
 Set the (z) and (N) values so that they satisfy the above equation.
 (Ex.) When z = 5 [ms], N = 240 times
 (Ex.) When z = 10 [ms], N = 120 times

29.2.8 Usage Note

The F-ZTAT version and the Mask ROM version satisfy the electrical characteristics indicated in this manual, but the actual power value, operating margin, and noise margin may differ from those in this manual, due to the difference of production process, on-chip ROM, layout pattern, etc.

When executing the system examination using the F-ZTAT version, be sure to execute the same system examination using the Mask ROM version when changing to the Mask ROM version.

29.3 Electrical Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

29.3.1 DC Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

Table 29.13 DC Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

(Conditions: $V_{cc} = AV_{cc} = 4.0$ to 5.5 V, $V_{ss} = AV_{ss} = 0.0$ V, $T_a = -20$ to $+75^\circ\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|--------------------|----------|--|-------------------------|--------------|-----|--------------|------|-------|
| | | | | Min | Typ | Max | | |
| Input high voltage | V_{IH} | MD0 | $V_{cc}=2.5$ to 5.5 V | $0.9 V_{cc}$ | — | $V_{cc}+0.3$ | V | |
| | | \overline{RES} , \overline{NMI} , \overline{IC} , $\overline{IRQ0}$ to $\overline{IRQ5}$ | | $0.8 V_{cc}$ | — | $V_{cc}+0.3$ | | |
| | | | $V_{cc}=2.5$ to 5.5 V | $0.9 V_{cc}$ | — | $V_{cc}+0.3$ | | |
| | | SCK1, SCK2, SI1, SI2, \overline{CS} , FTIA, FTIB, FTIC, FTID, TRIG, TMBI, \overline{ADTRG} | | $0.8 V_{cc}$ | — | $V_{cc}+0.3$ | | |
| | | OSC1, X1 | | $V_{cc}-0.5$ | — | $V_{cc}+0.3$ | | |
| | | | $V_{cc}=2.5$ to 5.5 V | $V_{cc}-0.3$ | — | $V_{cc}+0.3$ | | |
| | | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, PS0 to PS4 | $V_{cc}=2.5$ to 5.5 V | $0.7 V_{cc}$ | — | $V_{cc}+0.3$ | | |
| | | | | $0.8 V_{cc}$ | — | $V_{cc}+0.3$ | | |
| | | Csync | | $0.7 V_{cc}$ | — | $V_{cc}+0.3$ | | |

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|---------------------|----------|--|------------------------|--------------|--------------|--------------|------|-----------------|
| | | | | Min | Typ | Max | | |
| Input low voltage | V_{IL} | MD0 | $V_{CC}=2.5$ to $5.5V$ | -0.3 | — | $0.1 V_{CC}$ | V | |
| | | \overline{RES} , \overline{NMI} , \overline{IC} , $\overline{IRQ0}$ to $\overline{IRQ5}$ | | -0.3 | — | $0.2 V_{CC}$ | | |
| | | | $V_{CC}=2.5$ to $5.5V$ | -0.3 | — | $0.1 V_{CC}$ | | |
| | | SCK1, SCK2, SI1, SI2, \overline{CS} , FTIA, FTIB, FTIC, FTID, TRIG, TMBI, \overline{ADTRG} | | -0.3 | — | $0.2 V_{CC}$ | | |
| | | OSC1, X1 | | -0.3 | — | 0.5 | | |
| | | | $V_{CC}=2.5$ to $5.5V$ | -0.3 | — | 0.3 | | |
| | | P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, PS0 to PS4 | $V_{CC}=2.5$ to $5.5V$ | -0.3 | — | $0.3 V_{CC}$ | | |
| | | Csync | | -0.3 | — | $0.2 V_{CC}$ | | |
| Output high voltage | V_{OH} | SO1, SO2, SCK1, SCK2, PWM1, PWM2, PWM3, PWM4, PWM14, STRB, BUZZ, TMO, TMOW, FTOA, FTOB, PPG70 to PPG77, RP0 to RP7, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, PS0 to PS4 | $-I_{OH}=1.0mA$ | $V_{CC}-1.0$ | — | — | V | |
| | | | $-I_{OH}=0.5mA$ | — | $V_{CC}-0.5$ | — | V | Reference value |
| | | | $-I_{OH}=0.1mA$ | $V_{CC}-0.5$ | — | — | V | |
| | | | $V_{CC}=2.5$ to $5.5V$ | | | | | |

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|--|------------|---|--|--------|-----|-----|---------------|-------|
| | | | | Min | Typ | Max | | |
| Output low voltage | V_{OL} | SO1, SO2, SCK1, SCK2, PWM1, PWM2, PWM3, PWM4, PWM14, STRB, BUZZ, TMO, TMOW, FTOA, FTOB, PPG70 to PPG77, RP0 to RP7, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, PS0 to PS4 | $I_{OL}=1.6\text{mA}$ | — | — | 0.6 | V | |
| | | | $I_{OL}=0.4\text{mA}$ $V_{CC}=2.5$ to 5.5V | — | — | 0.4 | V | |
| | | P80 to P87 | $I_{OL}=20\text{mA}$ | — | — | 1.5 | V | |
| | | | $I_{OL}=1.6\text{mA}$ | — | — | 0.6 | V | |
| | | | $I_{OL}=0.4\text{mA}$ $V_{CC}=2.5$ to 5.5V | — | — | 0.4 | V | |
| Input /output leakage current | $ I_{IL} $ | MD0, OSC1 $\overline{\text{RES}}$, $\overline{\text{NMI}}$, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ5}}$, $\overline{\text{IC}}$ | $V_{in}=0.5$ to $V_{CC}-0.5\text{V}$ | — | — | 1.0 | μA | |
| | | SCK1, SCK2, SI1, SI2, $\overline{\text{CS}}$, FTIA, FTIB, FTIC, FTID, TRIG, TMBI, $\overline{\text{ADTRG}}$ | $V_{in}=0.5$ to $V_{CC}-0.5\text{V}$ | — | — | 1.0 | | |
| | | P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87 PS0 to PS4 | $V_{in}=0.5$ to $V_{CC}-0.5\text{V}$ | — | — | 1.0 | | |
| | | P00 to P07, AN8 to ANB | $V_{in}=0.5$ to $AV_{CC}-0.5\text{V}$ | — | — | 1.0 | | |

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|---|---------------------|---|--|--------|-----|-----|------|-------------------------|
| | | | | Min | Typ | Max | | |
| Pull-up MOS current | -I _p | P10 to P17, P20 to P27, P30 to P37 | V _{cc} =5.0V, V _{in} =0V | 50 | — | 300 | μA | Note 1 |
| Input capacity | C _{in} | All input pins except power supply, P23, P24 and analog system pins | f _{in} =1 MHz, V _{in} =0V, T _a =25°C | — | — | 15 | pF | |
| | | P23, P24 | f _{in} =1 MHz, V _{in} =0V, T _a =25°C | — | — | 20 | pF | |
| Active mode current dissipation (CPU operating) | I _{OPe} | V _{cc} | V _{cc} =5V, f _{osc} =10 MHz High-speed mode | — | 50 | 70 | mA | Note 2 |
| | | | V _{cc} =5V, f _{osc} =10 MHz Medium-speed mode (1/64) | — | 35 | — | mA | Reference value |
| Active mode current dissipation (reset) | I _{RES} | V _{cc} | V _{cc} =5V, f _{osc} =10 MHz | — | 25 | 45 | mA | Note 2 |
| Sleep mode current dissipation | I _{SLEEP} | V _{cc} | V _{cc} =5V, f _{osc} =10 MHz High-speed mode | — | 20 | 30 | mA | Note 2 |
| Subactive mode current dissipation | I _{SUB} | V _{cc} | V _{cc} =2.5V, With 32kHz crystal oscillator (φ _{sub} =φ _w /2) | — | 40 | 100 | μA | Note 2 |
| | | | V _{cc} =2.5V, With 32kHz crystal oscillator (φ _{sub} =φ _w /8) | — | 20 | — | | Reference value, Note 2 |
| Subsleep mode current dissipation | I _{SUBSLP} | V _{cc} | V _{cc} =2.5V, With 32kHz crystal oscillator (φ _{sub} =φ _w /2) | — | 15 | 30 | μA | Note 2 |
| | | | V _{cc} =2.5V, With 32kHz crystal oscillator (φ _{sub} =φ _w /8) | — | 10 | — | | Reference value, Note 2 |

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|--|--------------------|-----------------|--|--------|-----|-----|------|---------------------------|
| | | | | Min | Typ | Max | | |
| Watch mode current dissipation | I _{WATCH} | Vcc | Vcc=2.5V, With 32kHz crystal oscillator | — | 5 | 10 | μA | Note 2 |
| | | | Vcc=5.0V, With 32kHz crystal oscillator | — | 10 | — | μA | Reference value Note 2 |
| Standby mode current dissipation | I _{STBY} | Vcc | X1=Vcc, Without 32kHz crystal oscillator | — | — | 5 | μA | Note 2 |
| RAM data retaining voltage in standby mode | V _{STBY} | Vcc | | 2.0 | — | — | V | |

Notes: Do not open the AVcc and AVss pin even when the A/D converter is not in use.

1. Current value when the relevant bit of the pull-up MOS select register (PUR1 to PUR3) is set to 1.
2. The current on the pull-up MOS or the output buffer excluded.

Table 29.14 Pin Status at Current Dissipation Measurement

| Mode | RES Pin | Internal State | Pin | Oscillator Pin |
|---|---------|--------------------------------|-----|---|
| Active mode High-speed, medium-speed | Vcc | Operating | Vcc | Main clock: Crystal oscillator Sub clock: X1 pin = Vcc |
| Sleep mode High-speed, medium-speed | Vcc | CPU and servo circuits stopped | Vcc | |
| Reset | Vss | Reset | Vcc | |
| Standby mode | Vcc | All stopped | Vcc | |
| Subactive mode | Vcc | CPU and timer A operating | Vcc | Main clock: Crystal oscillator Sub clock: Crystal oscillator |
| Subsleep mode | Vcc | Timer A operating | Vcc | |
| Watch mode | Vcc | Timer A operating | Vcc | |

Table 29.15 Bus Drive Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

(Conditions: $V_{CC} = AV_{CC} = 4.0$ to 5.5 V, $V_{SS} = 0.0$ V, $T_a = -20$ to $+75^\circ\text{C}$ unless otherwise specified.) Applicable pin: SCL, SDA

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|-------------------------------|---------------------|-----------------|---------------------|---------------|-----|--------------|------|-------|
| | | | | Min | Typ | Max | | |
| Schmidt trigger input voltage | V_T^- | SCL, SDA | | 0.2 V_{CC} | — | — | V | |
| | V_T^+ | | | — | — | 0.7 V_{CC} | V | |
| | V_T^+ $-V_T^-$ | | | 0.05 V_{CC} | — | — | V | |
| Input High voltage | V_{IH} | SCL, SDA | | 0.7 V_{CC} | — | $V_{CC}+0.5$ | V | |
| Input Low voltage | V_{IL} | SCL, SDA | | -0.5 | — | 0.2 V_{CC} | V | |
| Output Low voltage | V_{OL} | SCL, SDA | $I_{OL}=8\text{mA}$ | — | — | 0.5 | V | |
| | | | $I_{OL}=3\text{mA}$ | — | — | 0.4 | | |
| SCL and SDA output fall time | t_{of} | SCL, SDA | | 20+ 0.1Cb | — | 250 | ns | |

29.3.2 Allowable Output Currents of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

The specifications for the digital pins are shown below.

Table 29.16 Allowable Output Currents of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

(Conditions: $V_{cc} = 2.5$ to 5.5 V, $V_{ss} = 0.0$ V, $T_a = -20$ to $+75^{\circ}\text{C}$)

| Item | Symbol | Value | Unit | Notes |
|--|---------------|-------|------|-------|
| Allowable input current (to chip) | I_o | 2 | mA | 1 |
| Allowable input current (to chip) | I_o | 22 | mA | 2 |
| Allowable input current (to chip) | I_o | 10 | mA | 3 |
| Allowable output current (from chip) | $-I_o$ | 2 | mA | 4 |
| Total allowable input current (to chip) | ΣI_o | 80 | mA | 5 |
| Total allowable output current (from chip) | $-\Sigma I_o$ | 50 | mA | 6 |

- Notes:
1. The allowable input current is the maximum value of the current flowing from each I/O pin to V_{ss} (except for port 8, SCL and SDA).
 2. The allowable input current is the maximum value of the current flowing from each I/O pin to V_{ss} . This applies to port 8.
 3. The allowable input current is the maximum value of the current flowing from each I/O pin to V_{ss} . This applies to SCL and SDA.
 4. The allowable output current is the maximum value of the current flowing from V_{cc} to each I/O pin.
 5. The total allowable input current is the sum of the currents flowing from all I/O pins to V_{ss} simultaneously.
 6. The total allowable output current is the sum of the currents flowing from V_{cc} to all I/O pins.

29.3.3 AC Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

Table 29.17 AC Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

(Conditions: $V_{CC} = AV_{CC} = 4.0$ to 5.5 V, $V_{SS} = AV_{SS} = 0.0$ V, $T_a = -20$ to $+75^{\circ}\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Notes |
|---|--------------|-----------------|---|--------|--------|-----|---------------|--------------|
| | | | | Min | Typ | Max | | |
| Clock oscillation frequency | f_{OSC} | OSC1, OSC2 | | 8 | — | 10 | MHz | |
| Clock cycle time | t_{cyc} | OSC1, OSC2 | | 100 | — | 125 | ns | Figure 29.11 |
| Subclock oscillation frequency | f_X | X1, X2 | $V_{CC}=2.5$ to 5.5V | — | 32.768 | — | kHz | |
| Subclock cycle time | t_{subcyc} | X1, X2 | $V_{CC}=2.5$ to 5.5V | — | 30.518 | — | μs | Figure 29.12 |
| Oscillation stabilization time | t_{tc} | OSC1, OSC2 | Crystal oscillator | — | — | 10 | ms | |
| | | X1, X2 | 32kHz crystal oscillator $V_{CC}=2.5$ to 5.5V | — | — | 2 | s | |
| External clock high width | t_{CPH} | OSC1 | | 40 | — | — | ns | Figure 29.11 |
| External clock low width | t_{CPL} | OSC1 | | 40 | — | — | ns | |
| External clock rise time | t_{CPr} | OSC1 | | — | — | 10 | ns | |
| External clock fall time | t_{CPf} | OSC1 | | — | — | 10 | ns | |
| External clock stabilization delay time | t_{DEXT} | OSC1 | | 500 | — | — | μs | Figure 29.13 |
| Subclock input low level pulse width | t_{EXCLL} | X1 | $V_{CC}=2.5$ to 5.5V | — | 15.26 | — | μs | Figure 29.12 |
| Subclock input high level pulse width | t_{EXCLH} | X1 | $V_{CC}=2.5$ to 5.5V | — | 15.26 | — | μs | |

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Figure |
|--------------------------------------|-------------|---|-------------------------|--------|-----|-----|---------------------------|--------------|
| | | | | Min | Typ | Max | | |
| Subclock input rise time | t_{EXCLr} | X1 | $V_{CC}=2.5$ to $5.5V$ | — | — | 10 | ns | Figure 29.12 |
| Subclock input fall time | t_{EXCLf} | X1 | $V_{CC}=2.5$ to $5.5V$ | — | — | 10 | ns | |
| \overline{RES} pin low level width | t_{REL} | \overline{RES} | $V_{CC}=2.5V$ to $5.5V$ | 20 | — | — | t_{cyc} | Figure 29.14 |
| Input pin high level width | t_{IH} | $\overline{IRQ0}$ to $\overline{IRQ5}$, \overline{NMI} , \overline{IC} , \overline{ADTRG} , \overline{TMBI} , \overline{FTIA} , \overline{FTIB} , \overline{FTIC} , \overline{FTID} , \overline{TRIG} | $V_{CC}=2.5V$ to $5.5V$ | 2 | — | — | t_{cyc} t_{subcyc} | Figure 29.15 |
| Input pin low level width | t_{IL} | $\overline{IRQ0}$ to $\overline{IRQ5}$, \overline{NMI} , \overline{IC} , \overline{ADTRG} , \overline{TMBI} , \overline{FTIA} , \overline{FTIB} , \overline{FTIC} , \overline{FTID} , \overline{TRIG} | $V_{CC}=2.5V$ to $5.5V$ | 2 | — | — | t_{cyc} t_{subcyc} | |

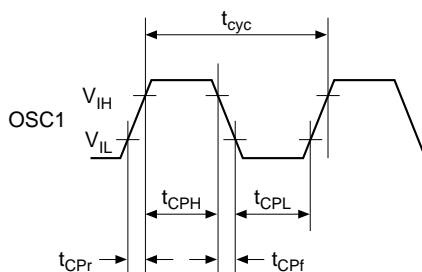


Figure 29.11 System Clock Timing

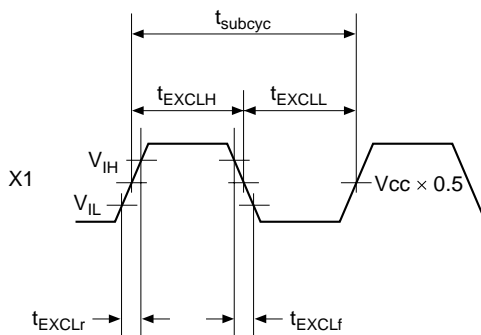
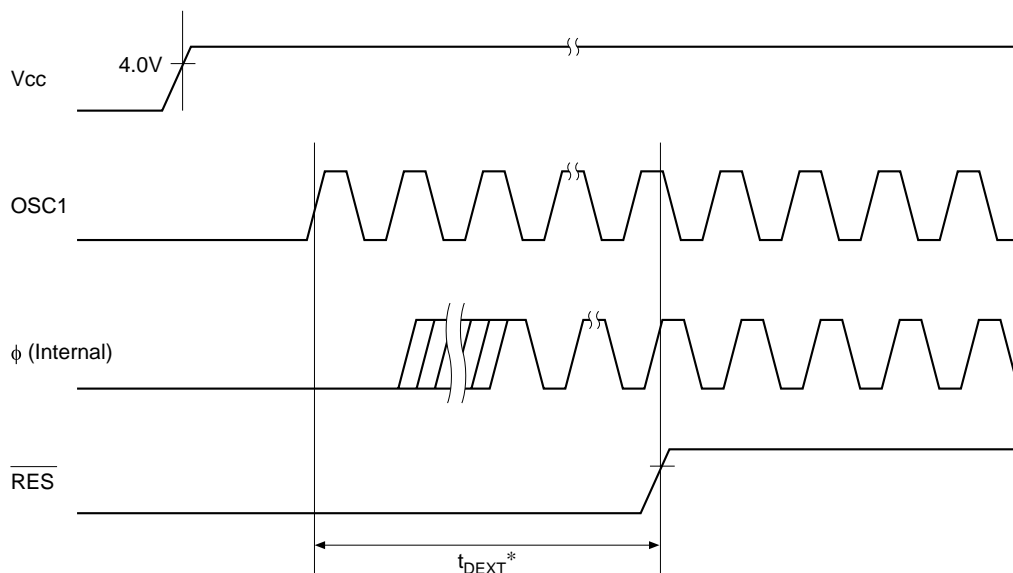


Figure 29.12 Subclock Input Timing



Note: * The t_{DEXT} includes the \overline{RES} pin Low level width $20 t_{cyc}$.

Figure 29.13 External Clock Stabilization Delay Timing

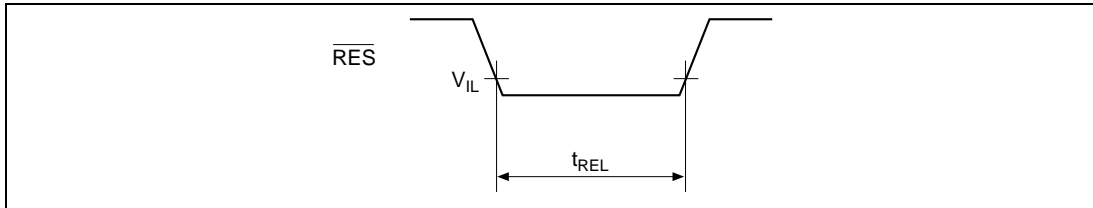


Figure 29.14 Reset Input Timing

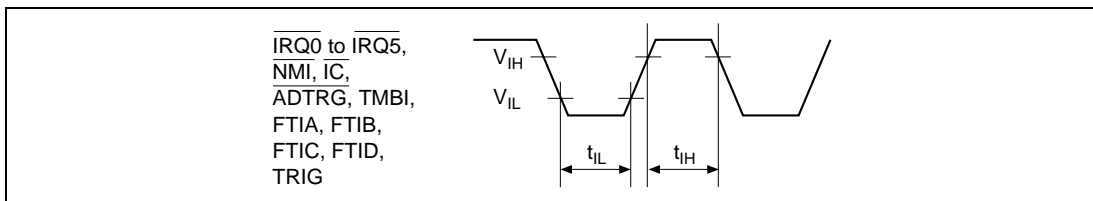


Figure 29.15 Input Timing

29.3.4 Serial Interface Timing of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

Table 29.18 Serial Interface Timing of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

(Conditions: $V_{CC} = AV_{CC} = 4.0$ to 5.5 V, $V_{SS} = AV_{SS} = 0.0$ V, $T_a = -20$ to $+75^{\circ}\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Figure |
|---------------------------------------|-------------------|------------------------|-----------------------|--------|-----|-----|-------------------|--------------|
| | | | | Min | Typ | Max | | |
| Input clock cycle | t_{cyc} | SCK1 | Asynchronization | 4 | — | — | t_{cyc} | Figure 29.16 |
| | | | Clock synchronization | 6 | — | — | | |
| | | SCK2 | | 2 | — | — | | |
| Input clock pulse width | t_{SCKW} | SCK1, SCK2 | | 0.4 | — | 0.6 | t_{scyc} | |
| Input clock rise time | t_{SCKr} | SCK1 | | — | — | 1.5 | t_{cyc} | |
| | | SCK2 | | — | — | 60 | ns | |
| Input clock fall time | t_{SCKf} | SCK1 | | — | — | 1.5 | t_{cyc} | |
| | | SCK2 | | — | — | 60 | ns | |
| Transmit data delay time (clock sync) | t_{TXD} | SO1 | | — | — | 100 | ns | Figure 29.17 |
| Receive data setup time (clock sync) | t_{RXS} | SI1 | | 100 | — | — | ns | |
| Receive data hold time (clock sync) | t_{RXH} | SI1 | | 100 | — | — | ns | |
| Transmit data output delay time | t_{TXD} | SO2 | | — | — | 200 | ns | Figure 29.17 |
| Receive data setup time (clock sync) | t_{RXS} | SI2 | | 180 | — | — | ns | |
| Receive data hold time (clock sync) | t_{RXH} | SI2 | | 180 | — | — | ns | |
| $\overline{\text{CS}}$ setup time | t_{CSS} | $\overline{\text{CS}}$ | | 1 | — | — | t_{scyc} | Figure 29.18 |
| $\overline{\text{CS}}$ hold time | t_{CSH} | $\overline{\text{CS}}$ | | 1 | — | — | t_{scyc} | |

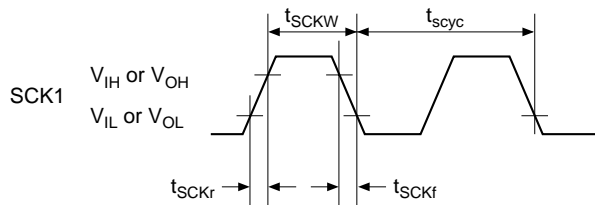


Figure 29.16 SCK1 Clock Timing

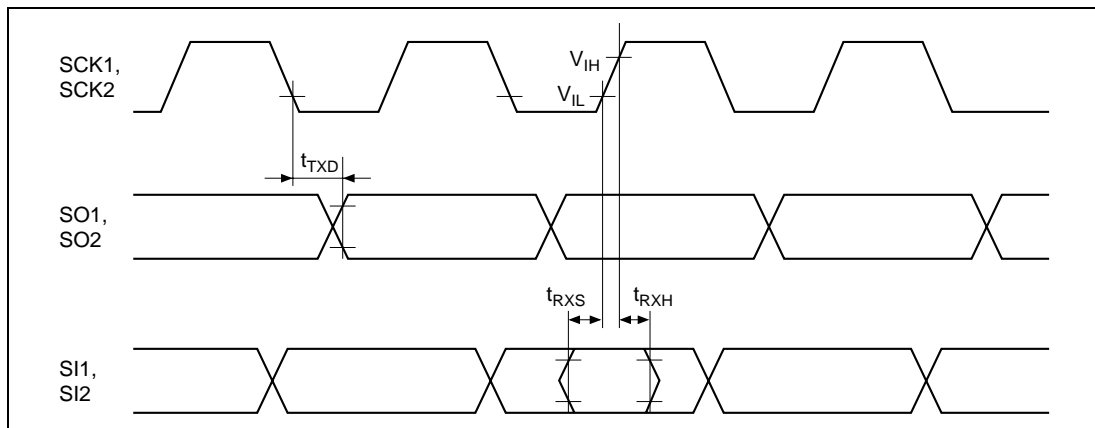


Figure 29.17 SCI I/O Timing/Clock Synchronization Mode

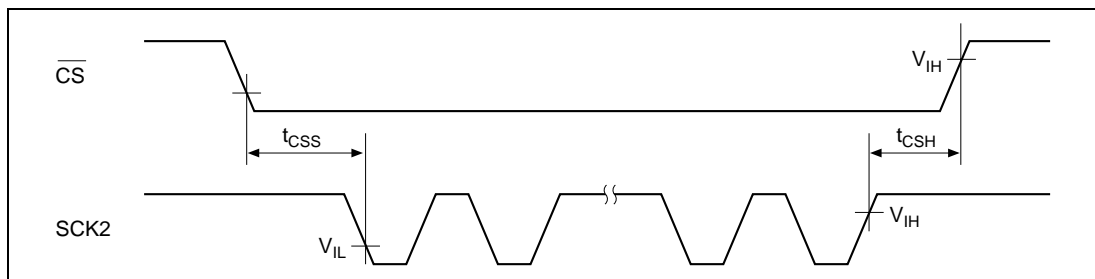


Figure 29.18 SCI2 Chip Select Timing

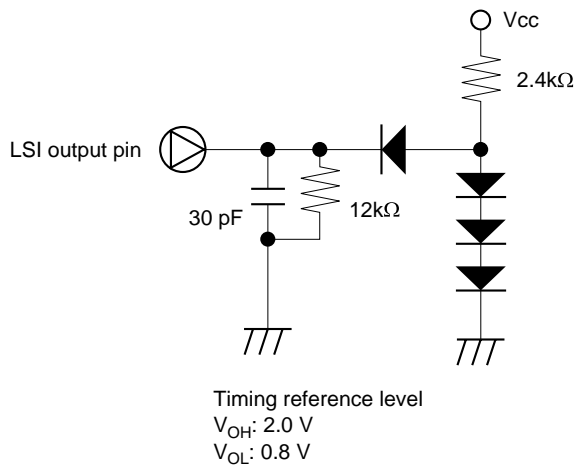


Figure 29.19 Output Load Conditions

Table 29.19 I²C Bus Interface Timing of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

(Conditions: V_{CC} = AV_{CC} = 4.0 to 5.5 V, V_{SS} = AV_{SS} = 0.0 V, Ta = -20 to +75°C unless otherwise specified.)

| Item | Symbol | Test Conditions | Values | | | Unit | Figure |
|--|-------------------|-----------------|--------|-----|------|------------------|--------------|
| | | | Min | Typ | Max | | |
| SCL input cycle time | t _{SCL} | | 12 | — | — | t _{cyc} | Figure 29.10 |
| SCL input high pulse width | t _{SCLH} | | 3 | — | — | t _{cyc} | |
| SCL input low pulse width | t _{SCLL} | | 5 | — | — | t _{cyc} | |
| SCL, SDA input rise time | t _{sr} | | — | — | 7.5* | t _{cyc} | |
| SCL, SDA input fall time | t _{sf} | | — | — | 300 | ns | |
| SCL, SDA input spike pulse removal time | t _{sp} | | — | — | 1 | t _{cyc} | |
| SDA input bus free time | t _{BUF} | | 5 | — | — | t _{cyc} | |
| Start condition input hold time | t _{STAH} | | 3 | — | — | t _{cyc} | |
| Re-transmit start condition input setup time | t _{STAS} | | 3 | — | — | t _{cyc} | |
| Stop condition input setup time | t _{STOS} | | 3 | — | — | t _{cyc} | |
| Data input setup time | t _{SDAS} | | 0.5 | — | — | t _{cyc} | |
| Data input hold time | t _{SDAH} | | 0 | — | — | ns | |
| SCL, SDA capacity load | C _b | | — | — | 400 | pF | |

Note: * Can also be set to 17.5 t_{cyc} depending on the selection of clock to be used by the I²C module.

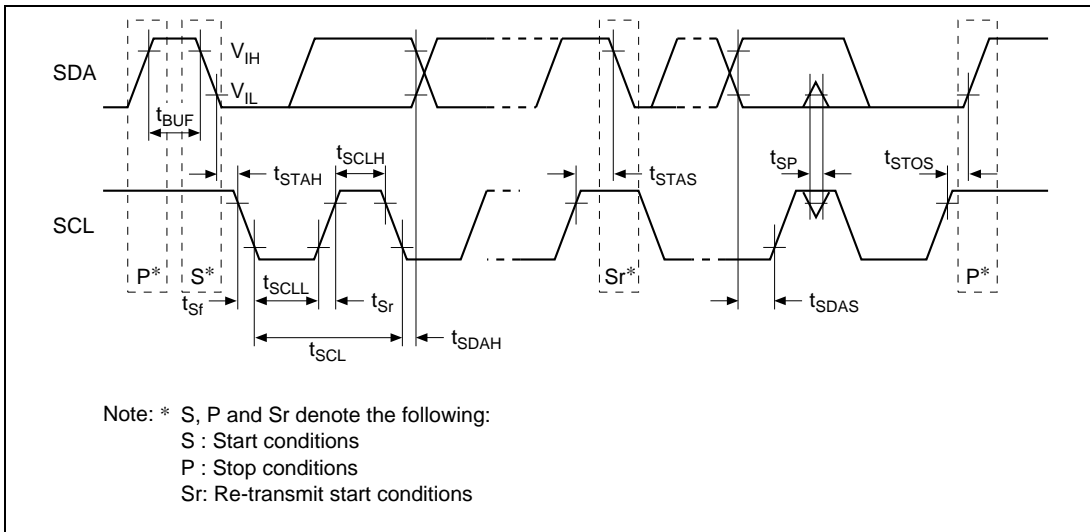


Figure 29.20 I²C Bus Interface I/O Timing

29.3.5 A/D Converter Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

Table 29.20 A/D Converter Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

(Conditions: $V_{CC} = AV_{CC} = 4.0$ to 5.5 V, $V_{SS} = AV_{SS} = 0.0$ V, $T_a = -20$ to $+75^{\circ}\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Note |
|-----------------------------------|-------------|------------------------|--|--------------|----------|--------------|------------------|------|
| | | | | Min | Typ | Max | | |
| Analog power supply voltage | AV_{CC} | AV_{CC} | | $V_{CC}-0.3$ | V_{CC} | $V_{CC}+0.3$ | V | |
| Analog input voltage | A_{VIN} | AN0 to AN7, AN8 to ANB | | AV_{SS} | — | AV_{CC} | V | |
| Analog power supply current | A_{ICC} | AV_{CC} | $AV_{CC}=5.0\text{V}$ | — | — | 2.0 | mA | |
| | A_{ISTOP} | AV_{CC} | $V_{CC}=2.5$ to 5.5V At reset and in power-down mode | — | — | 10 | μA | |
| Analog input capacitance | C_{AIN} | AN0 to AN7, AN8 to ANB | | — | — | 30 | pF | |
| Allowable signal source impedance | R_{AIN} | AN0 to AN7, AN8 to ANB | | — | — | 10 | $\text{k}\Omega$ | |
| Resolution | | | | — | — | 10 | Bit | |
| Absolute accuracy | | | $AV_{CC}=5.0\text{V}$ | — | — | ± 4 | LSB | |
| Conversion time | | | | 13.4 | — | 26.6 | μs | |

Note: Do not open the AV_{CC} and AV_{SS} pin even when the A/D converter is not in use. Set $AV_{CC} = V_{CC}$ and $AV_{SS} = V_{SS}$.

29.3.6 Servo Section Electrical Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

Table 29.21 Servo Section Electrical Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A (reference values)

(Conditions: $V_{cc} = SV_{cc} = 5.0\text{ V}$, $V_{ss} = SV_{ss} = 0.0\text{ V}$, $T_a = 25^\circ\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Reference Values | | | Unit | Note |
|-------------------------------------|--------|-----------------|--|------------------|------|------|------------|------|
| | | | | Min | Typ | Max | | |
| PB-CTL input amplifier voltage gain | | CTL (+) | CTLGR3=0, CTLGR2=0, CTLGR1=0, CTLGR0=0, $f=10\text{kHz}$ | 32.0 | 34.0 | 36.0 | dB | |
| | | | CTLGR3=0, CTLGR2=0, CTLGR1=0, CTLGR0=1, $f=10\text{kHz}$ | 34.5 | 36.5 | 38.5 | | |
| | | | CTLGR3=0, CTLGR2=0, CTLGR1=1, CTLGR0=0, $f=10\text{kHz}$ | 37.0 | 39.0 | 41.0 | | |
| | | | CTLGR3=0, CTLGR2=0, CTLGR1=1, CTLGR0=1, $f=10\text{kHz}$ | 39.5 | 41.5 | 43.5 | | |
| | | | CTLGR3=0, CTLGR2=1, CTLGR1=0, CTLGR0=0, $f=10\text{kHz}$ | 42.0 | 44.0 | 46.0 | | |
| | | | CTLGR3=0, CTLGR2=1, CTLGR1=0, CTLGR0=1, $f=10\text{kHz}$ | 44.5 | 46.5 | 48.5 | | |
| | | | CTLGR3=0, CTLGR2=1, CTLGR1=1, CTLGR0=0, $f=10\text{kHz}$ | 47.0 | 49.0 | 51.0 | | |
| | | | CTLGR3=0, CTLGR2=1, CTLGR1=1, CTLGR0=1, $f=10\text{kHz}$ | 49.5 | 51.5 | 53.5 | | |
| | | | CTLGR3=1, CTLGR2=0, CTLGR1=0, CTLGR0=0, $f=10\text{kHz}$ | 52.0 | 54.0 | 56.0 | | |
| | | | CTLGR3=1, CTLGR2=0, CTLGR1=0, CTLGR0=1, $f=10\text{kHz}$ | 54.5 | 56.5 | 58.5 | | |
| | | | CTLGR3=1, CTLGR2=0, CTLGR1=1, CTLGR0=0, $f=10\text{kHz}$ | 57.0 | 59.0 | 61.0 | | |
| | | | CTLGR3=1, CTLGR2=0, CTLGR1=1, CTLGR0=1, $f=10\text{kHz}$ | 59.5 | 61.5 | 63.5 | | |
| | | | CTLGR3=1, CTLGR2=1, CTLGR1=0, CTLGR0=0, $f=10\text{kHz}$ | 62.0 | 64.0 | 66.0 | | |
| | | | CTLGR3=1, CTLGR2=1, CTLGR1=0, CTLGR0=1, $f=10\text{kHz}$ | 64.5 | 66.5 | 68.5 | | |
| | | | CTLGR3=1, CTLGR2=1, CTLGR1=1, CTLGR0=0, $f=10\text{kHz}$ | 67.0 | 69.0 | 71.0 | | |
| | | | CTLGR3=1, CTLGR2=1, CTLGR1=1, CTLGR0=1, $f=10\text{kHz}$ | 69.5 | 71.5 | 73.5 | | |
| PB-CTL Schmitt input | V+TH | CTLSMT (i) | AC coupling, $C=0.1\mu\text{F}$ Typ (non pol) | — | 250 | — | mVp | |
| | V-TH | | AC coupling, $C=0.1\mu\text{F}$ Typ (non pol) | — | -250 | — | | |
| Analog switch ON resistance | REB | | | — | 150 | — | Ω | |
| REC-CTL output current | ICTL | CTL (+) | Series resistance = $0\ \Omega$ | — | 8 | — | mA | |
| | | CTL (-) | | — | 8 | — | | |
| REC-CTL inter-pin resistance | RCTL | | | — | 10 | — | k Ω | |

| Item | Symbol | Applicable Pins | Test Conditions | Reference Values | | | Unit | Note |
|---|----------|-----------------|--|------------------|------------------|-----|------------|------|
| | | | | Min | Typ | Max | | |
| CTL reference output voltage | | CTLREF | | — | 1/2 SV_{CC} | — | V | |
| CFG pin bias voltage | | CFG | | — | 1/2 SV_{CC} | — | V | |
| CFG input level | | CFG | AC coupling, $C=1\mu F$ Typ, $f=1kHz$ | 1.0 | — | — | Vpp | |
| CFG input impedance | | CFG | | — | 10 | — | k Ω | |
| CFG input threshold value | V+THCF | CFG | Rise threshold level | — | 2.25 | — | V | |
| | V−THCF | | Fall threshold level | — | 2.75 | — | | |
| DFG Schmidt input | V+THDF | DFG | Rising edge Schmidt level | — | 1.95 | — | V | |
| | V−THDF | | Falling edge Schmidt level | — | 1.85 | — | | |
| DPG Schmidt input | V+THDP | DPG | Rising edge Schmidt level | — | 3.55 | — | V | |
| | V−THDP | | Falling edge Schmidt level | — | 3.45 | — | | |
| 3-level output voltage | V_{OH} | Vpulse | $-I_{OH}=0.1mA$ | 4.0 | — | — | V | |
| | V_{OM} | | No load, $Hiz=1$ | — | 2.5 | — | | |
| | V_{OL} | | $I_{OL}=0.1mA$ | — | — | 1.0 | | |
| 3-level output pin divided voltage resistance | | Vpulse | | — | 15 | — | k Ω | |
| CFG Duty | | CFG | AC coupling, $C=1\mu F$ Typ, $f=1kHz$ | 48 | — | 52 | % | |

Table 29.22 Servo Section Electrical Characteristics of HD6432194, HD6432193, HD6432192, HD6432191, HD6432194C, HD6432194B, and HD6432194A

(Conditions: $V_{CC} = SV_{CC} = 5.0\text{ V}$, $V_{SS} = SV_{SS} = 0.0\text{ V}$, $T_a = 25^\circ\text{C}$ unless otherwise specified.)

| Item | Symbol | Applicable Pins | Test Conditions | Values | | | Unit | Note |
|-----------------------------|------------|--|-------------------------|----------------|-----|----------------|------|------|
| | | | | Min | Typ | Max | | |
| Digital input high voltage | V_{IH} | COMP, EXCTL, | | 0.8 | — | $V_{CC} + 0.3$ | V | |
| Digital input low voltage | V_{IL} | EXCAP, EXTTRG | | -0.3 | — | $0.2 V_{CC}$ | | |
| Digital output high voltage | V_{OH} | H.AmpSW, C.Rotary, | $-I_{OH} = 1\text{mA}$ | $V_{CC} - 1.0$ | — | — | V | |
| Digital output low voltage | V_{OL} | VIDEOFF, AUDIOFF, DRMPWM, CAPPWM, SV1, SV2 | $I_{OL} = 1.6\text{mA}$ | — | — | 0.6 | | |
| Current dissipation | I_{CCSV} | SVcc | At no load | — | 5 | 10 | mA | |

Appendix A Instruction Set

A.1 Instructions

[Operation Notation]

| | |
|----------------|--|
| Rd | General register (destination) ^{*1} |
| Rs | General register (source) ^{*1} |
| Rn | General register ^{*1} |
| ERn | General register (32-bit register) |
| MAC | Multiplication-Addition register (32-bit register) ^{*2} |
| (EAd) | Destination operand |
| (EAs) | Source operand |
| EXR | Extend register |
| CCR | Condition code register |
| N | N (negative flag) in CCR |
| Z | Z (zero) flag in CCR |
| V | V (overflow) flag in CCR |
| C | C (carry) flag in CCR |
| PC | Program counter |
| SP | Stack pointer |
| #IMM | Immediate data |
| disp | Displacement |
| + | Addition |
| – | Subtraction |
| × | Multiplication |
| ÷ | Division |
| ^ | Logical AND |
| ∨ | Logical OR |
| ⊕ | Exclusive logical OR |
| → | Move from the left to the right |
| ~ | Logical complement |
| () <> | Contents of operand |
| :8/:16/:24/:32 | 8/16/24/32 bit length |

- Notes: 1. General register is 8-bit (R0H to R7H, R0L to R7L), 16-bit (R0 to R7) or 32-bit (ER0 to ER7).
2. MAC register cannot be used in this LSI.

[Condition Code Notation]

| Symbol | Description |
|--------|--|
| ↓ | Modified according to the instruction result |
| * | Not fixed (value not guaranteed) |
| 0 | Always cleared to 0 |
| 1 | Always set to 1 |
| — | Not affected by the instruction execution result |

Table A.1 List of Instruction Set
(1) Data Transfer Instruction

| Mnemonic | | Size | Addressing Mode and Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | No of Execution States *1 | |
|-----------------------|-----------------------|----------------------------|--|----|------|----------|-------------|-----|---------|--------------------------|---|----------------|---|---|---|---|---------------------------|------------|
| | | | #xx | Rn | @ERn | @(d,ERn) | @-ERn/@ERn+ | @aa | @(d,PC) | @aa | | I | H | N | Z | V | | C |
| | | | | | | | | | | | | | | | | | Advanced Mode | |
| MOV | MOV.B #xx:8,Rd | B | 2 | | | | | | | | #xx:8→Rd8 | — | — | ↑ | ↓ | 0 | — | 1 |
| | MOV.B Rs,Rd | B | | 2 | | | | | | | Rs8→Rd8 | — | — | ↑ | ↓ | 0 | — | 1 |
| | MOV.B @ERs,Rd | B | | | 2 | | | | | | @ERs→Rd8 | — | — | ↑ | ↓ | 0 | — | 2 |
| | MOV.B @(d:16,ERs),Rd | B | | | | 4 | | | | | @(d:16,ERs)→Rd8 | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.B @(d:32,ERs),Rd | B | | | | | 8 | | | | @(d:32,ERs)→Rd8 | — | — | ↑ | ↓ | 0 | — | 5 |
| | MOV.B @ERs+,Rd | B | | | | | 2 | | | | @ERs→Rd8,ERs32+1→ERs32 | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.B @aa:8,Rd | B | | | | | | 2 | | | @aa:8→Rd8 | — | — | ↑ | ↓ | 0 | — | 2 |
| | MOV.B @aa:16,Rd | B | | | | | | | 4 | | @aa:16→Rd8 | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.B @aa:32,Rd | B | | | | | | | | 6 | @aa:32→Rd8 | — | — | ↑ | ↓ | 0 | — | 4 |
| | MOV.B Rs,@ERd | B | | | 2 | | | | | | Rs8→@ERd | — | — | ↑ | ↓ | 0 | — | 2 |
| | MOV.B Rs,@(d:16,ERd) | B | | | | 4 | | | | | Rs8→@(d:16,ERd) | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.B Rs,@(d:32,ERd) | B | | | | | 8 | | | | Rs8→@(d:32,ERd) | — | — | ↑ | ↓ | 0 | — | 5 |
| | MOV.B Rs,@-ERd | B | | | | | 2 | | | | ERd32-1→ERd32,Rs8→@ERd | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.B Rs,@aa:8 | B | | | | | | 2 | | | Rs8→@aa:8 | — | — | ↑ | ↓ | 0 | — | 2 |
| | MOV.B Rs,@aa:16 | B | | | | | | | 4 | | Rs8→@aa:16 | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.B Rs,@aa:32 | B | | | | | | | | 6 | Rs8→@aa:32 | — | — | ↑ | ↓ | 0 | — | 4 |
| | MOV.W #xx:16,Rd | W | 4 | | | | | | | | #xx:16→Rd16 | — | — | ↑ | ↓ | 0 | — | 2 |
| | MOV.W Rs,Rd | W | | | 2 | | | | | | Rs16→Rd16 | — | — | ↑ | ↓ | 0 | — | 1 |
| | MOV.W @ERs,Rd | W | | | | 2 | | | | | @ERs→Rd16 | — | — | ↑ | ↓ | 0 | — | 2 |
| | MOV.W @(d:16,ERs),Rd | W | | | | | 4 | | | | @(d:16,ERs)→Rd16 | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.W @(d:32,ERs),Rd | W | | | | | | 8 | | | @(d:32,ERs)→Rd16 | — | — | ↑ | ↓ | 0 | — | 5 |
| | MOV.W @ERs+,Rd | W | | | | | | 2 | | | @ERs→Rd16,ERs32+2→ERs32 | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.W @aa:16,Rd | W | | | | | | | 4 | | @aa:16→Rd16 | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.W @aa:32,Rd | W | | | | | | | | 6 | @aa:32→Rd16 | — | — | ↑ | ↓ | 0 | — | 4 |
| | MOV.W Rs,@ERd | W | | | 2 | | | | | | Rs16→@ERd | — | — | ↑ | ↓ | 0 | — | 2 |
| | MOV.W Rs,@(d:16,ERd) | W | | | | 4 | | | | | Rs16→@(d:16,ERd) | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.W Rs,@(d:32,ERd) | W | | | | | 8 | | | | Rs16→@(d:32,ERd) | — | — | ↑ | ↓ | 0 | — | 5 |
| | MOV.W Rs,@-ERd | W | | | | | 2 | | | | ERd32-2→ERd32,Rs16→@ERd | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.W Rs,@aa:16 | W | | | | | | | 4 | | Rs16→@aa:16 | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.W Rs,@aa:32 | W | | | | | | | | 6 | Rs16→@aa:32 | — | — | ↑ | ↓ | 0 | — | 4 |
| | MOV.L #xx:32,ERd | L | 6 | | | | | | | | #xx:32→ERd32 | — | — | ↑ | ↓ | 0 | — | 3 |
| | MOV.L ERs,ERd | L | | | 2 | | | | | | ERs32→ERd32 | — | — | ↑ | ↓ | 0 | — | 1 |
| | MOV.L @ERs,ERd | L | | | | 4 | | | | | @ERs→ERd32 | — | — | ↑ | ↓ | 0 | — | 4 |
| | MOV.L @(d:16,ERs),ERd | L | | | | | 6 | | | | @(d:16,ERs)→ERd32 | — | — | ↑ | ↓ | 0 | — | 5 |
| MOV.L @(d:32,ERs),ERd | L | | | | | | 10 | | | @(d:32,ERs)→ERd32 | — | — | ↑ | ↓ | 0 | — | 7 | |
| MOV.L @ERs+,ERd | L | | | | | | 4 | | | @ERs→ERd32,ERs32+4→ERs32 | — | — | ↑ | ↓ | 0 | — | 5 | |
| MOV.L @aa:16,ERd | L | | | | | | | 6 | | @aa:16→ERd32 | — | — | ↑ | ↓ | 0 | — | 5 | |
| MOV.L @aa:32,ERd | L | | | | | | | | 8 | @aa:32→ERd32 | — | — | ↑ | ↓ | 0 | — | 6 | |
| MOV.L ERs,@ERd | L | | | | 4 | | | | | ERs32→@ERd | — | — | ↑ | ↓ | 0 | — | 4 | |
| MOV.L ERs,@(d:16,ERd) | L | | | | | 6 | | | | ERs32→@(d:16,ERd) | — | — | ↑ | ↓ | 0 | — | 5 | |
| MOV.L ERs,@(d:32,ERd) | L | | | | | | 10 | | | ERs32→@(d:32,ERd) | — | — | ↑ | ↓ | 0 | — | 7 | |
| MOV.L ERs,@-ERd | L | | | | | | 4 | | | ERd32-4→ERd32,ERs32→@ERd | — | — | ↑ | ↓ | 0 | — | 5 | |
| MOV.L ERs,@aa:16 | L | | | | | | | 6 | | ERs32→@aa:16 | — | — | ↑ | ↓ | 0 | — | 5 | |
| MOV.L ERs,@aa:32 | L | | | | | | | | 8 | ERs32→@aa:32 | — | — | ↑ | ↓ | 0 | — | 6 | |
| POP | POP.W Rn | W | | | | | | | | 2 | @SP→Rn16,SP+2→SP | — | — | ↑ | ↓ | 0 | — | 3 |
| | POP.L ERn | L | | | | | | | | 4 | @SP→ERn32,SP+4→SP | — | — | ↑ | ↓ | 0 | — | 5 |
| PUSH | PUSH.W Rn | W | | | | | | | | 2 | SP-2→SP,Rn16→@SP | — | — | ↑ | ↓ | 0 | — | 3 |
| | PUSH.L ERn | L | | | | | | | | 4 | SP-4→SP,ERn32→@SP | — | — | ↑ | ↓ | 0 | — | 5 |
| LDM | LDM @SP+,(ERm-ERn) | L | | | | | | | | 4 | (@SP→ERn32,SP+4→SP) Repeat for the number of returns | — | — | — | — | — | — | 7/9/11 [1] |
| STM | STM (ERm-ERn),@-SP | L | | | | | | | | 4 | (SP-4→SP,ERn32→@SP) Repeat for the number of returns | — | — | — | — | — | — | 7/9/11 [1] |
| MOVFPE | MOVFPE @aa:16,Rd | Cannot be used in this LSI | | | | | | | | | | | | | | | | [2] |
| MOVTPE | MOVTPE Rs,@aa:16 | | | | | | | | | | | | | | | | | [2] |

(2) Arithmetic Instructions

| Mnemonic | Size | Addressing Mode and Instruction Length (Bytes) | | | | | | | | | | Operation | Condition Code | | | | | No of Execution States *1 | |
|-------------|------------------|--|----|------|---------|-------------|-----|--------|------|---|---|--|----------------|-----|-----|-----|---|---------------------------|-----|
| | | #xx | Rn | @ERn | @(dERn) | @-ERn/@ERn+ | @aa | @(dPC) | @@aa | I | I | | H | N | Z | V | C | | |
| | | | | | | | | | | | | | | | | | | Advanced Mode | |
| ADD | ADD.B #xx:8,Rd | B | 2 | | | | | | | | | Rd8+xx:8→Rd8 | — | ↑ | ↑ | ↑ | ↑ | ↑ | 1 |
| | ADD.B Rs,Rd | B | 2 | | | | | | | | | Rd8+Rs8→Rd8 | — | ↑ | ↑ | ↑ | ↑ | ↑ | 1 |
| | ADD.W #xx:16,Rd | W | 4 | | | | | | | | | Rd16+xx:16→Rd16 | — | [3] | ↑ | ↑ | ↑ | ↑ | 2 |
| | ADD.W Rs,Rd | W | 2 | | | | | | | | | Rd16+Rs16→Rd16 | — | [3] | ↑ | ↑ | ↑ | ↑ | 1 |
| | ADD.L #xx:32,ERd | L | 6 | | | | | | | | | ERd32+xx:32→ERd32 | — | [4] | ↑ | ↑ | ↑ | ↑ | 3 |
| ADDX | ADD.L ERs,ERd | L | 2 | | | | | | | | | ERd32+ERs32→ERd32 | — | [4] | ↑ | ↑ | ↑ | ↑ | 1 |
| | ADDX #xx:8,Rd | B | 2 | | | | | | | | | Rd8+xx:8+C→Rd8 | — | ↑ | ↑ | [5] | ↑ | ↑ | 1 |
| ADDS | ADDX Rs,Rd | B | 2 | | | | | | | | | Rd8+Rs8+C→Rd8 | — | ↑ | ↑ | [5] | ↑ | ↑ | 1 |
| | ADDS #1,ERd | L | 2 | | | | | | | | | ERd32+1→ERd32 | — | — | — | — | — | — | 1 |
| | ADDS #2,ERd | L | 2 | | | | | | | | | ERd32+2→ERd32 | — | — | — | — | — | — | 1 |
| INC | ADDS #4,ERd | L | 2 | | | | | | | | | ERd32+4→ERd32 | — | — | — | — | — | — | 1 |
| | INC.B Rd | B | 2 | | | | | | | | | Rd8+1→Rd8 | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | INC.W #1,Rd | W | 2 | | | | | | | | | Rd16+1→Rd16 | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| DAA | INC.W #2,Rd | W | 2 | | | | | | | | | Rd16+2→Rd16 | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | INC.L #1,ERd | L | 2 | | | | | | | | | ERd32+1→ERd32 | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | INC.L #2,ERd | L | 2 | | | | | | | | | ERd32+2→ERd32 | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | DAA Rd | B | 2 | | | | | | | | | Rd8 10 Decimal adjust →Rd8 | — | * | ↑ | ↑ | * | ↑ | 1 |
| | SUB.B Rs,Rd | B | 2 | | | | | | | | | Rd8-Rs8→Rd8 | — | ↑ | ↑ | ↑ | ↑ | ↑ | 1 |
| SUB | SUB.W #xx:16,Rd | W | 4 | | | | | | | | | Rd16-xx:16→Rd16 | — | [3] | ↑ | ↑ | ↑ | ↑ | 2 |
| | SUB.W Rs,Rd | W | 2 | | | | | | | | | Rd16-Rs16→Rd16 | — | [3] | ↑ | ↑ | ↑ | ↑ | 1 |
| | SUB.L #xx:32,ERd | L | 6 | | | | | | | | | ERd32-xx:32→ERd32 | — | [4] | ↑ | ↑ | ↑ | ↑ | 3 |
| | SUB.L ERs,ERd | L | 2 | | | | | | | | | ERd32-ERs32→ERd32 | — | [4] | ↑ | ↑ | ↑ | ↑ | 1 |
| | SUBX #xx:8,Rd | B | 2 | | | | | | | | | Rd8-xx:8-C→Rd8 | — | ↑ | ↑ | [5] | ↑ | ↑ | 1 |
| SUBS | SUBX Rs,Rd | B | 2 | | | | | | | | | Rd8-Rs8-C→Rd8 | — | ↑ | ↑ | [5] | ↑ | ↑ | 1 |
| | SUBS #1,ERd | L | 2 | | | | | | | | | ERd32-1→ERd32 | — | — | — | — | — | — | 1 |
| | SUBS #2,ERd | L | 2 | | | | | | | | | ERd32-2→ERd32 | — | — | — | — | — | — | 1 |
| DEC | SUBS #4,ERd | L | 2 | | | | | | | | | ERd32-4→ERd32 | — | — | — | — | — | — | 1 |
| | DEC.B Rd | B | 2 | | | | | | | | | Rd8-1→Rd8 | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | DEC.W #1,Rd | W | 2 | | | | | | | | | Rd16-1→Rd16 | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | DEC.W #2,Rd | W | 2 | | | | | | | | | Rd16-2→Rd16 | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | DEC.L #1,ERd | L | 2 | | | | | | | | | ERd32-1→ERd32 | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| DAS | DEC.L #2,ERd | L | 2 | | | | | | | | | ERd32-2→ERd32 | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | DAS Rd | B | 2 | | | | | | | | | Rd8 10 Decimal adjust →Rd8 | — | * | ↑ | ↑ | * | ↑ | 1 |
| | MULXU.B Rs,Rd | B | 2 | | | | | | | | | Rd8-Rs8→Rd16(Multiplication w/o sign) | — | — | — | — | — | — | 12 |
| | MULXU.W Rs,ERd | W | 2 | | | | | | | | | Rd16-Rs16→ERd32(Multiplication w/o sign) | — | — | — | — | — | — | 20 |
| | MULXS.B Rs,Rd | B | 4 | | | | | | | | | Rd8-Rs8→Rd16(Multiplication w/o sign) | — | — | ↑ | ↑ | — | — | 13 |
| DIVXU | MULXS.W Rs,ERd | W | 4 | | | | | | | | | Rd16-Rs16→ERd32(Multiplication w/o sign) | — | — | ↑ | ↑ | — | — | 21 |
| | DIVXU.B Rs,Rd | B | 2 | | | | | | | | | Rd16-Rs8→Rd16 (RdH: Rmainder, RdL: Quatent)(Division w/o sign) | — | — | [6] | [7] | — | — | 12 |
| DIVXS | DIVXU.W Rs,ERd | W | 2 | | | | | | | | | ERd32-Rs16→ERd32 (Ed:Remainder, Rd: Quatent)(Division with sign) | — | — | [6] | [7] | — | — | 20 |
| | DIVXS.B Rs,Rd | B | 4 | | | | | | | | | Rd16-Rs8→Rd16(RdH: Rmainder, RdL: Quatent)(Division w/o sign) | — | — | [8] | [7] | — | — | 13 |
| CMP | DIVXS.W Rs,ERd | W | 4 | | | | | | | | | ERd32-Rs16→ERd32 (Ed:Remainder, Rd: Quatent)(Division with sign) | — | — | [8] | [7] | — | — | 21 |
| | CMP.B #xx:8,Rd | B | 2 | | | | | | | | | Rd8-xx:8 | — | ↑ | ↑ | ↑ | ↑ | ↑ | 1 |
| | CMP.B Rs,Rd | B | 2 | | | | | | | | | Rd8-Rs8 | — | ↑ | ↑ | ↑ | ↑ | ↑ | 1 |
| | CMP.W #xx:16,Rd | W | 4 | | | | | | | | | Rd16-xx:16 | — | [3] | ↑ | ↑ | ↑ | ↑ | 2 |
| | CMP.W Rs,Rd | W | 2 | | | | | | | | | Rd16-Rs16 | — | [3] | ↑ | ↑ | ↑ | ↑ | 1 |
| NEG | CMP.L #xx:32,ERd | L | 6 | | | | | | | | | ERd32-xx:32 | — | [4] | ↑ | ↑ | ↑ | ↑ | 3 |
| | CMP.L ERs,ERd | L | 2 | | | | | | | | | ERd32-ERs32 | — | [4] | ↑ | ↑ | ↑ | ↑ | 1 |
| | NEG.B Rd | B | 2 | | | | | | | | | 0-Rd8→Rd8 | — | ↑ | ↑ | ↑ | ↑ | ↑ | 1 |
| EXTU | NEG.W Rd | W | 2 | | | | | | | | | 0-Rd16→Rd16 | — | ↑ | ↑ | ↑ | ↑ | ↑ | 1 |
| | NEG.L ERd | L | 2 | | | | | | | | | 0-ERd32→ERd32 | — | ↑ | ↑ | ↑ | ↑ | ↑ | 1 |
| EXTS | EXTU.W Rd | W | 2 | | | | | | | | | 0→(<Bits 15 to 8> of Rd16) | — | — | 0 | ↑ | 0 | — | 1 |
| | EXTU.L ERd | L | 2 | | | | | | | | | 0→(<Bits 31 to 16> of ERd32) | — | — | 0 | ↑ | 0 | — | 1 |
| TAS | EXTS.W Rd | W | 2 | | | | | | | | | (<Bit7> of Rd16)→ (<Bits 15 to 8> of Rd16) | — | — | ↑ | ↑ | 0 | — | 1 |
| | EXTS.L ERd | L | 2 | | | | | | | | | (<Bit15> of ERd32)→ (<Bits31 to 16> of ERd32) | — | — | ↑ | ↑ | 0 | — | 1 |
| TAS @ERd *3 | | B | | 4 | | | | | | | | @ERd-0→CCR set, (1)→ (<Bit7> of @ERd) | — | — | ↑ | ↑ | 0 | — | 4 |
| MAC | MAC @ERn+,@ERm+ | Cannot be used in this LSI | | | | | | | | | | | | | | | | | [2] |
| CLRMAC | CLRMAC | | | | | | | | | | | | | | | | | | |
| LDMAC | LDMAC ERs,MACH | | | | | | | | | | | | | | | | | | |
| | LDMAC ERs,MACL | | | | | | | | | | | | | | | | | | |
| STMAC | STMAC MACH,ERd | | | | | | | | | | | | | | | | | | |
| | STMAC MACL,ERd | | | | | | | | | | | | | | | | | | |

(3) Logic Operations Instructions

| Mnemonic | | Size | Addressing Mode and Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | | No of Execution States *1 | |
|----------|------------------|------|--|----|------|-----------|-------------|-----|----------|--------|--------------------|----------------|---------------|---|---|---|---|---------------------------|---|
| | | | #xx | Rn | @ERn | @ (d,ERn) | @-ERn/@ERn+ | @aa | @ (d,PC) | @ @ aa | | | Advanced Mode | | | | | | |
| | | | | | | | | | | | | | I | H | N | Z | V | | C |
| AND | AND.B #xx:8,Rd | B | 2 | | | | | | | | Rd8^#xx:8→Rd8 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | AND.B Rs,Rd | B | | 2 | | | | | | | Rd8^Rs8→Rd8 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | AND.W #xx:16,Rd | W | 4 | | | | | | | | Rd16^#xx:16→Rd16 | — | — | ↑ | ↑ | 0 | — | 2 | |
| | AND.W Rs,Rd | W | | 2 | | | | | | | Rd16^Rs16→Rd16 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | AND.L #xx:32,ERd | L | 6 | | | | | | | | ERd32^#xx:32→ERd32 | — | — | ↑ | ↑ | 0 | — | 3 | |
| | AND.L ERs,ERd | L | | 4 | | | | | | | ERd32^ERs32→ERd32 | — | — | ↑ | ↑ | 0 | — | 2 | |
| OR | OR.B #xx:8,Rd | B | 2 | | | | | | | | Rd8v#xx:8→Rd8 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | OR.B Rs,Rd | B | | 2 | | | | | | | Rd8vRs8→Rd8 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | OR.W #xx:16,Rd | W | 4 | | | | | | | | Rd16v#xx:16→Rd16 | — | — | ↑ | ↑ | 0 | — | 2 | |
| | OR.W Rs,Rd | W | | 2 | | | | | | | Rd16vRs16→Rd16 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | OR.L #xx:32,ERd | L | 6 | | | | | | | | ERd32v#xx:32→ERd32 | — | — | ↑ | ↑ | 0 | — | 3 | |
| | OR.L ERs,ERd | L | | 4 | | | | | | | ERd32vERs32→ERd32 | — | — | ↑ | ↑ | 0 | — | 2 | |
| XOR | XOR.B #xx:8,Rd | B | 2 | | | | | | | | Rd8@#xx:8→Rd8 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | XOR.B Rs,Rd | B | | 2 | | | | | | | Rd8@Rs8→Rd8 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | XOR.W #xx:16,Rd | W | 4 | | | | | | | | Rd16@#xx:16→Rd16 | — | — | ↑ | ↑ | 0 | — | 2 | |
| | XOR.W Rs,Rd | W | | 2 | | | | | | | Rd16@Rs16→Rd16 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | XOR.L #xx:32,ERd | L | 6 | | | | | | | | ERd32@#xx:32→ERd32 | — | — | ↑ | ↑ | 0 | — | 3 | |
| | XOR.L ERs,ERd | L | | 4 | | | | | | | ERd32@ERs32→ERd32 | — | — | ↑ | ↑ | 0 | — | 2 | |
| NOT | NOT.B Rd | B | | 2 | | | | | | | ~Rd8→Rd8 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | NOT.W Rd | W | | 2 | | | | | | | ~Rd16→Rd16 | — | — | ↑ | ↑ | 0 | — | 1 | |
| | NOT.L ERd | L | | 2 | | | | | | | ~ERd32→ERd32 | — | — | ↑ | ↑ | 0 | — | 1 | |

(4) Shift Instructions

| Mnemonic | | Size | Addressing Mode and Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No of Execution States *1 |
|----------|----------------|------|--|----|------|-----------|-------------|-----|----------|-------|--|-----------|----------------|---|---|---|---|---|---------------------------|
| | | | #xx | Rn | @ERn | @ (d,ERn) | @-ERn/@ERn+ | @aa | @ (d,PC) | @ @aa | | | I | H | N | Z | V | C | |
| | | | | | | | | | | | | | | | | | | | Advanced Mode |
| SHAL | SHAL.B Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | SHAL.B #2,Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | SHAL.W Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | SHAL.W #2,Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | SHAL.L ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| | SHAL.L #2,ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | ↑ | ↑ | 1 |
| SHAR | SHAR.B Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | SHAR.B #2,Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | SHAR.W Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | SHAR.W #2,Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | SHAR.L ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | SHAR.L #2,ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| SHLL | SHLL.B Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | SHLL.B #2,Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | SHLL.W Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | SHLL.W #2,Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | SHLL.L ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | SHLL.L #2,ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| SHLR | SHLR.B Rd | B | 2 | | | | | | | | | | — | — | 0 | ↑ | 0 | ↑ | 1 |
| | SHLR.B #2,Rd | B | 2 | | | | | | | | | | — | — | 0 | ↑ | 0 | ↑ | 1 |
| | SHLR.W Rd | W | 2 | | | | | | | | | | — | — | 0 | ↑ | 0 | ↑ | 1 |
| | SHLR.W #2,Rd | W | 2 | | | | | | | | | | — | — | 0 | ↑ | 0 | ↑ | 1 |
| | SHLR.L ERd | L | 2 | | | | | | | | | | — | — | 0 | ↑ | 0 | ↑ | 1 |
| | SHLR.L #2,ERd | L | 2 | | | | | | | | | | — | — | 0 | ↑ | 0 | ↑ | 1 |
| ROTXL | ROTXL.B Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTXL.B #2,Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTXL.W Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTXL.W #2,Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTXL.L ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTXL.L #2,ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| ROTXR | ROTXR.B Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTXR.B #2,Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTXR.W Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTXR.W #2,Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTXR.L ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTXR.L #2,ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| ROTL | ROTL.B Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTL.B #2,Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTL.W Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTL.W #2,Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTL.L ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTL.L #2,ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| ROTR | ROTR.B Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTR.B #2,Rd | B | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTR.W Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTR.W #2,Rd | W | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTR.L ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |
| | ROTR.L #2,ERd | L | 2 | | | | | | | | | | — | — | ↑ | ↑ | 0 | ↑ | 1 |

(5) Bit Manipulation Instructions

| Mnemonic | Size | Addressing Mode and Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | No of Execution States *1 |
|----------|-------------------|--|----|------|-----------|-------------|-----|----------|-------|--------------------------------------|-----------|----------------|---|---|---|---|---------------------------|
| | | #xx | Rn | @ERn | @ (d,ERn) | @-ERn @ERn+ | @aa | @ (d,PC) | @ @aa | | | I | H | N | Z | V | |
| | | | | | | | | | | | | | | | | | Advanced Mode |
| BSET | BSET #xx:3,Rd | B | 2 | | | | | | | (#xx:3 of Rd8)←1 | — | — | — | — | — | — | 1 |
| | BSET #xx:3,@ERd | B | | 4 | | | | | | (#xx:3 of @ERd)←1 | — | — | — | — | — | — | 4 |
| | BSET #xx:3,@aa:8 | B | | | | | 4 | | | (#xx:3 of @aa:8)←1 | — | — | — | — | — | — | 4 |
| | BSET #xx:3,@aa:16 | B | | | | | 6 | | | (#xx:3 of @aa:16)←1 | — | — | — | — | — | — | 5 |
| | BSET #xx:3,@aa:32 | B | | | | | 8 | | | (#xx:3 of @aa:32)←1 | — | — | — | — | — | — | 6 |
| | BSET Rn,Rd | B | 2 | | | | | | | (Rn8 of Rd8)←1 | — | — | — | — | — | — | 1 |
| | BSET Rn,@ERd | B | | 4 | | | | | | (Rn8 of @ERd)←1 | — | — | — | — | — | — | 4 |
| | BSET Rn,@aa:8 | B | | | | | 4 | | | (Rn8 of @aa:8)←1 | — | — | — | — | — | — | 4 |
| BCLR | BSET Rn,@aa:16 | B | | | | | 6 | | | (Rn8 of @aa:16)←1 | — | — | — | — | — | — | 5 |
| | BSET Rn,@aa:32 | B | | | | | 8 | | | (Rn8 of @aa:32)←1 | — | — | — | — | — | — | 6 |
| | BCLR #xx:3,Rd | B | 2 | | | | | | | (#xx:3 of Rd8)←0 | — | — | — | — | — | — | 1 |
| | BCLR #xx:3,@ERd | B | | 4 | | | | | | (#xx:3 of @ERd)←0 | — | — | — | — | — | — | 4 |
| | BCLR #xx:3,@aa:8 | B | | | | | 4 | | | (#xx:3 of @aa:8)←0 | — | — | — | — | — | — | 4 |
| | BCLR #xx:3,@aa:16 | B | | | | | 6 | | | (#xx:3 of @aa:16)←0 | — | — | — | — | — | — | 5 |
| | BCLR #xx:3,@aa:32 | B | | | | | 8 | | | (#xx:3 of @aa:32)←0 | — | — | — | — | — | — | 6 |
| | BCLR Rn,Rd | B | 2 | | | | | | | (Rn8 of Rd8)←0 | — | — | — | — | — | — | 1 |
| BNOT | BCLR Rn,@ERd | B | | 4 | | | | | | (Rn8 of @ERd)←0 | — | — | — | — | — | — | 4 |
| | BCLR Rn,@aa:8 | B | | | | | 4 | | | (Rn8 of @aa:8)←0 | — | — | — | — | — | — | 4 |
| | BCLR Rn,@aa:16 | B | | | | | 6 | | | (Rn8 of @aa:16)←0 | — | — | — | — | — | — | 5 |
| | BCLR Rn,@aa:32 | B | | | | | 8 | | | (Rn8 of @aa:32)←0 | — | — | — | — | — | — | 6 |
| | BNOT #xx:3,Rd | B | 2 | | | | | | | (#xx:3 of Rd8)←¬(#xx:3 of Rd8) | — | — | — | — | — | — | 1 |
| | BNOT #xx:3,@ERd | B | | 4 | | | | | | (#xx:3 of @ERd)←¬(#xx:3 of @ERd) | — | — | — | — | — | — | 4 |
| | BNOT #xx:3,@aa:8 | B | | | | | 4 | | | (#xx:3 of @aa:8)←¬(#xx:3 of @aa:8) | — | — | — | — | — | — | 4 |
| | BNOT #xx:3,@aa:16 | B | | | | | 6 | | | (#xx:3 of @aa:16)←¬(#xx:3 of @aa:16) | — | — | — | — | — | — | 5 |
| BTST | BNOT #xx:3,@aa:32 | B | | | | | 8 | | | (#xx:3 of @aa:32)←¬(#xx:3 of @aa:32) | — | — | — | — | — | — | 6 |
| | BNOT Rn,Rd | B | 2 | | | | | | | (Rn8 of Rd8)←¬(Rn8 of Rd8) | — | — | — | — | — | — | 1 |
| | BNOT Rn,@ERd | B | | 4 | | | | | | (Rn8 of @ERd)←¬(Rn8 of @ERd) | — | — | — | — | — | — | 4 |
| | BNOT Rn,@aa:8 | B | | | | | 4 | | | (Rn8 of @aa:8)←¬(Rn8 of @aa:8) | — | — | — | — | — | — | 4 |
| | BNOT Rn,@aa:16 | B | | | | | 6 | | | (Rn8 of @aa:16)←¬(Rn8 of @aa:16) | — | — | — | — | — | — | 5 |
| | BNOT Rn,@aa:32 | B | | | | | 8 | | | (Rn8 of @aa:32)←¬(Rn8 of @aa:32) | — | — | — | — | — | — | 6 |
| | BTST #xx:3,Rd | B | 2 | | | | | | | ¬(#xx:3 of Rd8)→Z | — | — | ↑ | — | — | — | 1 |
| | BTST #xx:3,@ERd | B | | 4 | | | | | | ¬(#xx:3 of @ERd)→Z | — | — | ↑ | — | — | — | 3 |
| BLD | BTST #xx:3,@aa:8 | B | | | | | 4 | | | ¬(#xx:3 of @aa:8)→Z | — | — | ↑ | — | — | — | 3 |
| | BTST #xx:3,@aa:16 | B | | | | | 6 | | | ¬(#xx:3 of @aa:16)→Z | — | — | ↑ | — | — | — | 4 |
| | BTST #xx:3,@aa:32 | B | | | | | 8 | | | ¬(#xx:3 of @aa:32)→Z | — | — | ↑ | — | — | — | 5 |
| | BTST Rn,Rd | B | 2 | | | | | | | ¬(Rn8 of Rd8)→Z | — | — | ↑ | — | — | — | 1 |
| | BTST Rn,@ERd | B | | 4 | | | | | | ¬(Rn8 of @ERd)→Z | — | — | ↑ | — | — | — | 3 |
| | BTST Rn,@aa:8 | B | | | | | 4 | | | ¬(Rn8 of @aa:8)→Z | — | — | ↑ | — | — | — | 3 |
| | BTST Rn,@aa:16 | B | | | | | 6 | | | ¬(Rn8 of @aa:16)→Z | — | — | ↑ | — | — | — | 4 |
| | BTST Rn,@aa:32 | B | | | | | 8 | | | ¬(Rn8 of @aa:32)→Z | — | — | ↑ | — | — | — | 5 |
| BILD | BLD #xx:3,Rd | B | 2 | | | | | | | (#xx:3 of Rd8)→C | — | — | — | ↑ | — | — | 1 |
| | BLD #xx:3,@ERd | B | | 4 | | | | | | (#xx:3 of @ERd)→C | — | — | — | ↑ | — | — | 3 |
| | BLD #xx:3,@aa:8 | B | | | | | 4 | | | (#xx:3 of @aa:8)→C | — | — | — | ↑ | — | — | 3 |
| | BLD #xx:3,@aa:16 | B | | | | | 6 | | | (#xx:3 of @aa:16)→C | — | — | — | ↑ | — | — | 4 |
| | BLD #xx:3,@aa:32 | B | | | | | 8 | | | (#xx:3 of @aa:32)→C | — | — | — | ↑ | — | — | 5 |
| BST | BILD #xx:3,Rd | B | 2 | | | | | | | ¬(#xx:3 of Rd8)→C | — | — | — | ↑ | — | — | 1 |
| | BILD #xx:3,@ERd | B | | 4 | | | | | | ¬(#xx:3 of @ERd)→C | — | — | — | ↑ | — | — | 3 |
| | BILD #xx:3,@aa:8 | B | | | | | 4 | | | ¬(#xx:3 of @aa:8)→C | — | — | — | ↑ | — | — | 3 |
| | BILD #xx:3,@aa:16 | B | | | | | 6 | | | ¬(#xx:3 of @aa:16)→C | — | — | — | ↑ | — | — | 4 |
| | BILD #xx:3,@aa:32 | B | | | | | 8 | | | ¬(#xx:3 of @aa:32)→C | — | — | — | ↑ | — | — | 5 |
| BIST | BST #xx:3,Rd | B | 2 | | | | | | | C→(#xx:3 of Rd8) | — | — | — | — | — | — | 1 |
| | BST #xx:3,@ERd | B | | 4 | | | | | | C→(#xx:3 of @ERd) | — | — | — | — | — | — | 4 |
| | BST #xx:3,@aa:8 | B | | | | | 4 | | | C→(#xx:3 of @aa:8) | — | — | — | — | — | — | 4 |
| | BST #xx:3,@aa:16 | B | | | | | 6 | | | C→(#xx:3 of @aa:16) | — | — | — | — | — | — | 5 |
| | BST #xx:3,@aa:32 | B | | | | | 8 | | | C→(#xx:3 of @aa:32) | — | — | — | — | — | — | 6 |
| BAND | BIST #xx:3,Rd | B | 2 | | | | | | | ¬C→(#xx:3 of Rd8) | — | — | — | — | — | — | 1 |
| | BIST #xx:3,@ERd | B | | 4 | | | | | | ¬C→(#xx:3 of @ERd) | — | — | — | — | — | — | 4 |
| | BIST #xx:3,@aa:8 | B | | | | | 4 | | | ¬C→(#xx:3 of @aa:8) | — | — | — | — | — | — | 4 |
| | BIST #xx:3,@aa:16 | B | | | | | 6 | | | ¬C→(#xx:3 of @aa:16) | — | — | — | — | — | — | 5 |
| | BIST #xx:3,@aa:32 | B | | | | | 8 | | | ¬C→(#xx:3 of @aa:32) | — | — | — | — | — | — | 6 |
| BAND | BAND #xx:3,Rd | B | 2 | | | | | | | C∧(#xx:3 of Rd8)→C | — | — | — | — | ↑ | — | 1 |
| | BAND #xx:3,@ERd | B | | 4 | | | | | | C∧(#xx:3 of @ERd)→C | — | — | — | — | ↑ | — | 3 |
| | BAND #xx:3,@aa:8 | B | | | | | 4 | | | C∧(#xx:3 of @aa:8)→C | — | — | — | — | ↑ | — | 3 |
| | BAND #xx:3,@aa:16 | B | | | | | 6 | | | C∧(#xx:3 of @aa:16)→C | — | — | — | — | ↑ | — | 4 |
| | BAND #xx:3,@aa:32 | B | | | | | 8 | | | C∧(#xx:3 of @aa:32)→C | — | — | — | — | ↑ | — | 5 |

| Mnemonic | | Size | Addressing Mode and Instruction Length (Bytes) | | | | | | | | Operation | Condition Code | | | | | No of Execution States *1 | |
|----------|--------------------|------|--|----|------|----------|-------------|-----|---------------|------|-----------|---------------------------|--|--|--|--|---------------------------|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @-ERn/@ERn+ | @aa | @(d,PC) | @@aa | | | | | | | | |
| | | | I | H | N | Z | V | C | Advanced Mode | | | | | | | | | |
| BIAND | BIAND #xx:3,Rd | B | | 2 | | | | | | | | C^ [~(#xx:3 of Rd8)]→C | | | | | | 1 |
| | BIAND #xx:3,@ERd | B | | | 4 | | | | | | | C^ [~(#xx:3 of @ERd)]→C | | | | | 3 | |
| | BIAND #xx:3,@aa:8 | B | | | | | | 4 | | | | C^ [~(#xx:3 of @aa:8)]→C | | | | | 3 | |
| | BIAND #xx:3,@aa:16 | B | | | | | | 6 | | | | C^ [~(#xx:3 of @aa:16)]→C | | | | | 4 | |
| | BIAND #xx:3,@aa:32 | B | | | | | | 8 | | | | C^ [~(#xx:3 of @aa:32)]→C | | | | | 5 | |
| BOR | BOR #xx:3,Rd | B | | 2 | | | | | | | | Cv(#xx:3 of Rd8)→C | | | | | 1 | |
| | BOR #xx:3,@ERd | B | | | 4 | | | | | | | Cv(#xx:3 of @ERd)→C | | | | | 3 | |
| | BOR #xx:3,@aa:8 | B | | | | | | 4 | | | | Cv(#xx:3 of @aa:8)→C | | | | | 3 | |
| | BOR #xx:3,@aa:16 | B | | | | | | 6 | | | | Cv(#xx:3 of @aa:16)→C | | | | | 4 | |
| | BOR #xx:3,@aa:32 | B | | | | | | 8 | | | | Cv(#xx:3 of @aa:32)→C | | | | | 5 | |
| BIOR | BIOR #xx:3,Rd | B | | 2 | | | | | | | | Cv [~(#xx:3 of Rd8)]→C | | | | | 1 | |
| | BIOR #xx:3,@ERd | B | | | 4 | | | | | | | Cv [~(#xx:3 of @ERd)]→C | | | | | 3 | |
| | BIOR #xx:3,@aa:8 | B | | | | | | 4 | | | | Cv [~(#xx:3 of @aa:8)]→C | | | | | 3 | |
| | BIOR #xx:3,@aa:16 | B | | | | | | 6 | | | | Cv [~(#xx:3 of @aa:16)]→C | | | | | 4 | |
| | BIOR #xx:3,@aa:32 | B | | | | | | 8 | | | | Cv [~(#xx:3 of @aa:32)]→C | | | | | 5 | |
| BXOR | BXOR #xx:3,Rd | B | | 2 | | | | | | | | C@ (#xx:3 of Rd8)→C | | | | | 1 | |
| | BXOR #xx:3,@ERd | B | | | 4 | | | | | | | C@ (#xx:3 of @ERd)→C | | | | | 3 | |
| | BXOR #xx:3,@aa:8 | B | | | | | | 4 | | | | C@ (#xx:3 of @aa:8)→C | | | | | 3 | |
| | BXOR #xx:3,@aa:16 | B | | | | | | 6 | | | | C@ (#xx:3 of @aa:16)→C | | | | | 4 | |
| | BXOR #xx:3,@aa:32 | B | | | | | | 8 | | | | C@ (#xx:3 of @aa:32)→C | | | | | 5 | |
| BIXOR | BIXOR #xx:3,Rd | B | | 2 | | | | | | | | C@ [~(#xx:3 of Rd8)]→C | | | | | 1 | |
| | BIXOR #xx:3,@ERd | B | | | 4 | | | | | | | C@ [~(#xx:3 of @ERd)]→C | | | | | 3 | |
| | BIXOR #xx:3,@aa:8 | B | | | | | | 4 | | | | C@ [~(#xx:3 of @aa:8)]→C | | | | | 3 | |
| | BIXOR #xx:3,@aa:16 | B | | | | | | 6 | | | | C@ [~(#xx:3 of @aa:16)]→C | | | | | 4 | |
| | BIXOR #xx:3,@aa:32 | B | | | | | | 8 | | | | C@ [~(#xx:3 of @aa:32)]→C | | | | | 5 | |

(6) Branch Instructions

| Mnemonic | | Size | Addressing Mode and Instruction Length (Bytes) | | | | | | | | Operation | Operation Code | No of Execution States *1 | | | | | | |
|----------|--------------------|------|--|----|------|----------|-------------|-----|---------|--|------------------|----------------|---------------------------|---|---|---|---|---------------|---|
| | | | #xx | Rn | @ERn | @(d,ERn) | @-ERn/@ERn+ | @aa | @(d,PC) | @@aa | | | | I | | | | | |
| Bcc | BRA d:8(BT d:8) | — | | | | | | | 2 | if condition is true then PC←PC+d else next; | Branch Condition | I | H | N | Z | V | C | Advanced Mode | |
| | BRA d:16(BT d:16) | — | | | | | | | 4 | | Always | — | — | — | — | — | — | — | 2 |
| | BRN d:8(BF d:8) | — | | | | | | | 2 | | Never | — | — | — | — | — | — | — | 3 |
| | BRN d:16(BF d:16) | — | | | | | | | 4 | | CvZ=0 | — | — | — | — | — | — | — | 2 |
| | BHI d:8 | — | | | | | | | 2 | | CvZ=1 | — | — | — | — | — | — | — | 3 |
| | BHI d:16 | — | | | | | | | 4 | | C=0 | — | — | — | — | — | — | — | 2 |
| | BLS d:8 | — | | | | | | | 2 | | C=1 | — | — | — | — | — | — | — | 3 |
| | BLS d:16 | — | | | | | | | 4 | | Z=0 | — | — | — | — | — | — | — | 2 |
| | BCC d:8(BHS d:8) | — | | | | | | | 2 | | Z=1 | — | — | — | — | — | — | — | 3 |
| | BCC d:16(BHS d:16) | — | | | | | | | 4 | | V=0 | — | — | — | — | — | — | — | 2 |
| | BCS d:8(BLO d:8) | — | | | | | | | 2 | | V=1 | — | — | — | — | — | — | — | 3 |
| | BCS d:16(BLO d:16) | — | | | | | | | 4 | | N=0 | — | — | — | — | — | — | — | 2 |
| | BNE d:8 | — | | | | | | | 2 | | N=1 | — | — | — | — | — | — | — | 3 |
| | BNE d:16 | — | | | | | | | 4 | | N⊕V=0 | — | — | — | — | — | — | — | 2 |
| | BEQ d:8 | — | | | | | | | 2 | | N⊕V=1 | — | — | — | — | — | — | — | 3 |
| | BEQ d:16 | — | | | | | | | 4 | | Z∨(N⊕V)=0 | — | — | — | — | — | — | — | 2 |
| | BVC d:8 | — | | | | | | | 2 | | Z∨(N⊕V)=1 | — | — | — | — | — | — | — | 3 |
| | BVC d:16 | — | | | | | | | 4 | | | | | | | | | | |
| | BVS d:8 | — | | | | | | | 2 | | | | | | | | | | |
| | BVS d:16 | — | | | | | | | 4 | | | | | | | | | | |
| | BPL d:8 | — | | | | | | | 2 | | | | | | | | | | |
| | BPL d:16 | — | | | | | | | 4 | | | | | | | | | | |
| | BMI d:8 | — | | | | | | | 2 | | | | | | | | | | |
| | BMI d:16 | — | | | | | | | 4 | | | | | | | | | | |
| | BGE d:8 | — | | | | | | | 2 | | | | | | | | | | |
| | BGE d:16 | — | | | | | | | 4 | | | | | | | | | | |
| | BLT d:8 | — | | | | | | | 2 | | | | | | | | | | |
| | BLT d:16 | — | | | | | | | 4 | | | | | | | | | | |
| | BGT d:8 | — | | | | | | | 2 | | | | | | | | | | |
| | BGT d:16 | — | | | | | | | 4 | | | | | | | | | | |
| | BLE d:8 | — | | | | | | | 2 | | | | | | | | | | |
| | BLE d:16 | — | | | | | | | 4 | | | | | | | | | | |
| JMP | JMP @ERn | — | | | 2 | | | | | PC←ERn | | | | | | | | 2 | |
| | JMP @aa:24 | — | | | | | | 4 | | PC←aa:24 | | | | | | | | 3 | |
| | JMP @@aa:8 | — | | | | | | | 2 | PC←@aa:8 | | | | | | | | 5 | |
| BSR | BSR d:8 | — | | | | | | | 2 | PC→@-SP,PC←PC+d:8 | | | | | | | | 4 | |
| | BSR d:16 | — | | | | | | | 4 | PC→@-SP,PC←PC+d:16 | | | | | | | | 5 | |
| JSR | JSR @ERn | — | | | 2 | | | | | PC→@-SP,PC←ERn | | | | | | | | 4 | |
| | JSR @aa:24 | — | | | | | | 4 | | PC→@-SP,PC←aa:24 | | | | | | | | 5 | |
| | JSR @@aa:8 | — | | | | | | | 2 | PC→@-SP,PC←@aa:8 | | | | | | | | 6 | |
| RTS | RTS | — | | | | | | | 2 | PC←@SP+ | | | | | | | | 5 | |

(7) System Control Instructions

| Mnemonic | | Size | Addressing Mode and Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | No of Execution States *1 | |
|----------|---------------------|------|--|----|------|----------|------------|-----|---------|-----|---|---|----------------|---|---|---|---|---------------------------|---------------|
| | | | #xx | Rn | @ERn | @(d,ERn) | @-ERn@ERn+ | @aa | @(d,PC) | @aa | | | I | H | N | Z | V | C | Advanced Mode |
| TRAPA | TRAPA #xx:2 | — | | | | | | | | | | PC→@-SP,CCR→@-SP, EXR→@-SP,<Vector>→PC | 1 | — | — | — | — | — | 8 [9] |
| RTE | RTE | — | | | | | | | | | | EXR←@SP+,CCR←@SP+, PC←@SP+ | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 5 [9] 2 |
| SLEEP | SLEEP | — | | | | | | | | | | Transition to power-down state | — | — | — | — | — | — | 1 |
| LDC | LDC #xx:8,CCR | B | 2 | | | | | | | | | #xx:8→CCR | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 2 |
| | LDC #xx:8,EXR | B | 4 | | | | | | | | | #xx:8→EXR | — | — | — | — | — | — | 1 |
| | LDC Rs,CCR | B | | 2 | | | | | | | | Rs8→CCR | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 1 |
| | LDC Rs,EXR | B | | 2 | | | | | | | | Rs8→EXR | — | — | — | — | — | — | 3 |
| | LDC @ERs,CCR | W | | | 4 | | | | | | | @ERs→CCR | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 3 |
| | LDC @ERs,EXR | W | | | 4 | | | | | | | @ERs→EXR | — | — | — | — | — | — | 4 |
| | LDC @(d:16,ERs),CCR | W | | | | 6 | | | | | | @(d:16,ERs)→CCR | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 4 |
| | LDC @(d:16,ERs),EXR | W | | | | 6 | | | | | | @(d:16,ERs)→EXR | — | — | — | — | — | — | 6 |
| | LDC @(d:32,ERs),CCR | W | | | | 10 | | | | | | @(d:32,ERs)→CCR | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 6 |
| | LDC @(d:32,ERs),EXR | W | | | | 10 | | | | | | @(d:32,ERs)→EXR | — | — | — | — | — | — | 4 |
| | LDC @ERs+,CCR | W | | | | | 4 | | | | | @ERs→CCR,ERs32+2→ERs32 | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 4 |
| | LDC @ERs+,EXR | W | | | | | 4 | | | | | @ERs→EXR,ERs32+2→ERs32 | — | — | — | — | — | — | 4 |
| | LDC @aa:16,CCR | W | | | | | | 6 | | | | @aa:16→CCR | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 4 |
| | LDC @aa:16,EXR | W | | | | | | 6 | | | | @aa:16→EXR | — | — | — | — | — | — | 5 |
| | LDC @aa:32,CCR | W | | | | | | 8 | | | | @aa:32→CCR | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 5 |
| | LDC @aa:32,EXR | W | | | | | | 8 | | | | @aa:32→EXR | — | — | — | — | — | — | 1 |
| STC | STC CCR,Rd | B | | 2 | | | | | | | | CCR→Rd8 | — | — | — | — | — | — | 1 |
| | STC EXR,Rd | B | | 2 | | | | | | | | EXR→Rd8 | — | — | — | — | — | — | 3 |
| | STC CCR,@ERd | W | | | 4 | | | | | | | CCR→@ERd | — | — | — | — | — | — | 3 |
| | STC EXR,@ERd | W | | | 4 | | | | | | | EXR→@ERd | — | — | — | — | — | — | 4 |
| | STC CCR,@(d:16,ERd) | W | | | | 6 | | | | | | CCR→@(d:16,ERd) | — | — | — | — | — | — | 4 |
| | STC EXR,@(d:16,ERd) | W | | | | 6 | | | | | | EXR→@(d:16,ERd) | — | — | — | — | — | — | 6 |
| | STC CCR,@(d:32,ERd) | W | | | | 10 | | | | | | CCR→@(d:32,ERd) | — | — | — | — | — | — | 6 |
| | STC EXR,@(d:32,ERd) | W | | | | 10 | | | | | | EXR→@(d:32,ERd) | — | — | — | — | — | — | 4 |
| | STC CCR,@-ERd | W | | | | | 4 | | | | | ERd32-2→ERd32,CCR→@ERd | — | — | — | — | — | — | 4 |
| | STC EXR,@-ERd | W | | | | | 4 | | | | | ERd32-2→ERd32,EXR→@ERd | — | — | — | — | — | — | 4 |
| | STC CCR,@aa:16 | W | | | | | | 6 | | | | CCR→@aa:16 | — | — | — | — | — | — | 4 |
| | STC EXR,@aa:16 | W | | | | | | 6 | | | | EXR→@aa:16 | — | — | — | — | — | — | 5 |
| | STC CCR,@aa:32 | W | | | | | | 8 | | | | CCR→@aa:32 | — | — | — | — | — | — | 5 |
| | STC EXR,@aa:32 | W | | | | | | 8 | | | | EXR→@aa:32 | — | — | — | — | — | — | 1 |
| ANDC | ANDC #xx:8,CCR | B | 2 | | | | | | | | | CCR∧#xx:8→CCR | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 2 |
| | ANDC #xx:8,EXR | B | 4 | | | | | | | | | EXR∧#xx:8→EXR | — | — | — | — | — | — | 1 |
| ORC | ORC #xx:8,CCR | B | 2 | | | | | | | | | CCR∨#xx:8→CCR | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 2 |
| | ORC #xx:8,EXR | B | 4 | | | | | | | | | EXR∨#xx:8→EXR | — | — | — | — | — | — | 1 |
| XORC | XORC #xx:8,CCR | B | 2 | | | | | | | | | CCR⊕#xx:8→CCR | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | 2 |
| | XORC #xx:8,EXR | B | 4 | | | | | | | | | EXR⊕#xx:8→EXR | — | — | — | — | — | — | 1 |
| NOP | NOP | — | | | | | | | | | 2 | PC←PC+2 | — | — | — | — | — | — | |

(8) Block Transfer Instructions

| Mnemonic | Size | Addressing Mode and Instruction Length (Bytes) | | | | | | | | | Operation | Condition Code | | | | | | No of Execution States ^{#1} |
|----------|----------|--|----|------|----------|-------------|-----|---------|------|---|--|----------------|---|---|---|---|---|--------------------------------------|
| | | #xx | Rn | @ERn | @(r,ERn) | @-ERn/@ERn+ | @aa | @(r,PC) | @@aa | | | I | H | N | Z | V | C | |
| | | | | | | | | | | | | | | | | | | Advanced Mode |
| EEPMOV | EEPMOV.B | — | | | | | | | | 4 | if R4L≠0 Repeat @ER5→@ER6 ER5+1→ER5 ER6+1→ER6 R4L-1→R4L Until R4L=0 else next; | — | — | — | — | — | — | 4+2n ^{#2} |
| | EEPMOV.W | — | | | | | | | | 4 | if R4≠0 Repeat @ER5→@ER6 ER5+1→ER5 ER6+1→ER6 R4-1→R4 Until R4=0 else next; | — | — | — | — | — | — | 4+2n ^{#2} |

- Notes: 1. The values indicated in the column of number of execution states apply when instruction code and operand exist in the on-chip memory.
2. n is the initial setting value of R4L or R4.
3. Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.
- [1] 7 states when the number of return/retract registers is 2, 9 states when the number of registers is 3, and 11 states when the number of registers is 4.
- [2] Cannot be used in this LSI.
- [3] Set to 1 when a carry or borrow occurs at bit 11, otherwise cleared to 0.
- [4] Set to 1 when a carry or borrow occurs at bit 27, otherwise cleared to 0.
- [5] Retains the value before computation when the computation result is 0, otherwise cleared to 0.
- [6] Set to 1 when the divisor is negative, otherwise cleared to 0.
- [7] Set to 1 when the divisor is 0, otherwise cleared to 0.
- [8] Set to 1 when the quotient is negative, otherwise cleared to 0.
- [9] 1 is added to the number of execution states when EXR is valid.

A.2 Instruction Codes

A.2 Instruction Codes

| Instruction | Mnemonic | Op/Size | Instruction Format | | | | | | | | | |
|-------------|---------------------|---------|--------------------|----------|----------|----------|----------|----------|----------|----------|----------|-----------|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| ADD | ADD.B #xx:8,Rd | B | 8 | rd | IMM | | | | | | | |
| | ADD.B Rs,Rd | B | 0 | 8 | rs | rd | | | | | | |
| | ADD.W #xx:16,Rd | W | 7 | 9 | 1 | rd | | | | | | |
| | ADD.W Rs,Rd | W | 0 | 9 | rs | rd | | | | | | |
| | ADD.L #xx:32,Rd | L | 7 | A | 1 | 0:erd | IMM | | | | | |
| ADDS | ADD.L ERs,ERd | L | 0 | A | 1:ers | 0:erd | | | | | | |
| | ADDS #1,ERd | L | 0 | B | 0 | 0:erd | | | | | | |
| | ADDS #2,ERd | L | 0 | B | 8 | 0:erd | | | | | | |
| | ADDS #4,ERd | L | 0 | B | 9 | 0:erd | | | | | | |
| | ADDS #x:8,ERd | L | 0 | B | 9 | 0:erd | | | | | | |
| ADDX | ADDX #xx:8,Rd | B | 9 | rd | IMM | | | | | | | |
| | ADDX Rs,Rd | B | 0 | E | rs | rd | | | | | | |
| | AND.B #xx:8,Rd | B | E | rd | IMM | | | | | | | |
| | AND.B Rs,Rd | B | 1 | 6 | rs | rd | | | | | | |
| | AND.W #xx:16,Rd | W | 7 | 9 | 6 | rd | IMM | | | | | |
| AND | AND.W Rs,Rd | W | 6 | 6 | rs | rd | | | | | | |
| | AND.L #xx:32,Rd | L | 7 | A | 6 | 0:erd | IMM | | | | | |
| | AND.L ERs,ERd | L | 0 | 1 | F | 0 | 6:ers | 0:erd | | | | |
| | ANDC #xx:8,CCR | B | 0 | 6 | IMM | | | | | | | |
| | ANDC #xx:8,EXR | B | 0 | 1 | 4 | 1 | 0:6 | IMM | | | | |
| BAND | BAND #xx:3,Rd | B | 7 | 6 | 0:IMM | rd | | | | | | |
| | BAND #xx:3,@ERd | B | 7 | C | 0:erd | 0 | 7:6 | 0:IMM | 0 | | | |
| | BAND #xx:3,@aa:8 | B | 7 | E | abs | 0 | 7:6 | 0:IMM | 0 | | | |
| | BAND #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | 0:IMM | 0 | | | |
| | BAND #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | 0:IMM | 0 | | | |
| Bcc | BRA d:8 (BT d:8) | — | 4 | 0 | disp | | | | 7 | 6 | 0:IMM | 0 |
| | BRA d:16 (BT d:16) | — | 5 | 8 | 0 | 0 | disp | | | | | |
| | BRN d:8 (BF d:8) | — | 4 | 1 | disp | | | | | | | |
| | BRN d:16 (BF d:16) | — | 5 | 8 | 1 | 0 | disp | | | | | |
| | BHI d:8 | — | 4 | 2 | disp | | | | | | | |
| | BHI d:16 | — | 5 | 8 | 2 | 0 | disp | | | | | |
| | BLS d:8 | — | 4 | 3 | disp | | | | | | | |
| | BLS d:16 | — | 5 | 8 | 3 | 0 | disp | | | | | |
| | BCC d:8 (BHS d:8) | — | 4 | 4 | disp | | | | | | | |
| | BCC d:16 (BHS d:16) | — | 5 | 8 | 4 | 0 | disp | | | | | |
| | BCS d:8 (BLO d:8) | — | 4 | 5 | disp | | | | | | | |
| | BCS d:16 (BLO d:16) | — | 5 | 8 | 5 | 0 | disp | | | | | |
| | BNE d:8 | — | 4 | 6 | disp | | | | | | | |
| | BNE d:16 | — | 5 | 8 | 6 | 0 | disp | | | | | |
| | BEQ d:8 | — | 4 | 7 | disp | | | | | | | |
| | BEQ d:16 | — | 5 | 8 | 7 | 0 | disp | | | | | |
| | BVC d:8 | — | 4 | 8 | disp | | | | | | | |
| | BVC d:16 | — | 5 | 8 | 8 | 0 | disp | | | | | |
| | BVS d:8 | — | 4 | 9 | disp | | | | | | | |
| | BVS d:16 | — | 5 | 8 | 9 | 0 | disp | | | | | |

| Instruction | Mnemonic | Size | Instruction Format | | | | | | | | | |
|----------------|--------------------|------|--------------------|----------|----------|----------|----------|----------|----------|----------|----------|-----------|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| Bcc (Cont.) | BPL d:8 | — | 4 | A | | | | | | | | |
| | BPL d:16 | — | 5 | 8 | A | 0 | disp | | | | | |
| | BMI d:8 | — | 4 | B | | | | | | | | |
| | BMI d:16 | — | 5 | 8 | B | 0 | disp | | | | | |
| | BGE d:8 | — | 4 | C | | | | | | | | |
| | BGE d:16 | — | 5 | 8 | C | 0 | disp | | | | | |
| | BLT d:8 | — | 4 | D | | | | | | | | |
| | BLT d:16 | — | 5 | 8 | D | 0 | disp | | | | | |
| | BGT d:8 | — | 4 | E | | | | | | | | |
| | BGT d:16 | — | 5 | 8 | E | 0 | disp | | | | | |
| | BLE d:8 | — | 4 | F | | | | | | | | |
| | BLE d:16 | — | 5 | 8 | F | 0 | disp | | | | | |
| BCLR | BCLR #xx:3,Rd | B | 7 | 2 | 0:IMM | rd | | | | | | |
| | BCLR #xx:3,@ERd | B | 7 | D | 0:erd | 0 | 7 | 2 | 0:IMM | 0 | | |
| | BCLR #xx:3,@aa:8 | B | 7 | F | | | 7 | 2 | 0:IMM | 0 | | |
| | BCLR #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | |
| | BCLR #xx:3,@aa:32 | B | 6 | A | 3 | 8 | | | 7 | 2 | 0:IMM | 0 |
| | BCLR Rn,Rd | B | 6 | 2 | m | rd | | | | | | |
| | BCLR Rn,@ERd | B | 7 | D | 0:erd | 0 | 6 | 2 | m | 0 | | |
| | BCLR Rn,@aa:8 | B | 7 | F | | | 6 | 2 | m | 0 | | |
| | BCLR Rn,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | |
| | BCLR Rn,@aa:32 | B | 6 | A | 3 | 8 | | | 6 | 2 | m | 0 |
| | BIAND #xx:3,Rd | B | 7 | 6 | 1:IMM | rd | | | | | | |
| | BIAND #xx:3,@ERd | B | 7 | C | 0:erd | 0 | 7 | 6 | 1:IMM | 0 | | |
| BIAND | BIAND #xx:3,@aa:8 | B | 7 | E | | | 7 | 6 | 1:IMM | 0 | | |
| | BIAND #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | |
| | BIAND #xx:3,@aa:32 | B | 6 | A | 3 | 0 | | | 7 | 6 | 1:IMM | 0 |
| | BILD #xx:3,Rd | B | 7 | 7 | 1:IMM | rd | | | | | | |
| | BILD #xx:3,@ERd | B | 7 | C | 0:erd | 0 | 7 | 7 | 1:IMM | 0 | | |
| | BILD #xx:3,@aa:8 | B | 7 | E | | | 7 | 7 | 1:IMM | 0 | | |
| | BILD #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | |
| | BILD #xx:3,@aa:32 | B | 6 | A | 3 | 0 | | | 7 | 7 | 1:IMM | 0 |
| | BIOR #xx:3,Rd | B | 7 | 4 | 1:IMM | rd | | | | | | |
| | BIOR #xx:3,@ERd | B | 7 | C | 0:erd | 0 | 7 | 4 | 1:IMM | 0 | | |
| | BIOR #xx:3,@aa:8 | B | 7 | E | | | 7 | 4 | 1:IMM | 0 | | |
| | BIOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | |
| BIST | BIOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | | | 7 | 7 | 1:IMM | 0 |
| | BIST #xx:3,Rd | B | 6 | 7 | 1:IMM | rd | | | | | | |
| | BIST #xx:3,@ERd | B | 7 | D | 0:erd | 0 | 6 | 7 | 1:IMM | 0 | | |
| | BIST #xx:3,@aa:8 | B | 7 | F | | | 6 | 7 | 1:IMM | 0 | | |
| | BIST #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | |
| | BIST #xx:3,@aa:32 | B | 6 | A | 3 | 8 | | | 6 | 7 | 1:IMM | 0 |

| Instruction | Mnemonic | Op Size | Instruction Format | | | | | | | | | |
|-------------|--------------------|------------|--------------------|----------|----------|----------|----------|----------|----------|----------|----------|-----------|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| BIXOR | BIXOR #xx:3,Rd | B | 7 | 5 | 1:IMM rd | | | | | | | |
| | BIXOR #xx:3,@ERd | B | 7 | C | 0:erd | 0 | 7 | 5 | 1:IMM 0 | | | |
| | BIXOR #xx:3,@aa:8 | B | 7 | E | abs | | | | | | | |
| | BIXOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | |
| | BIXOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | |
| BLD | BLD #xx:3,Rd | B | 7 | 7 | 0:IMM rd | | | | | | | |
| | BLD #xx:3,@ERd | B | 7 | C | 0:erd | 0 | 7 | 7 | 0:IMM 0 | | | |
| | BLD #xx:3,@aa:8 | B | 7 | E | abs | | | | | | | |
| | BLD #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | |
| | BLD #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | |
| BNOT | BNOT #xx:3,Rd | B | 7 | 1 | 0:IMM rd | | | | | | | |
| | BNOT #xx:3,@ERd | B | 7 | D | 0:erd | 0 | 7 | 1 | 0:IMM 0 | | | |
| | BNOT #xx:3,@aa:8 | B | 7 | F | abs | | | | | | | |
| | BNOT #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | |
| | BNOT #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | |
| BNOT Rn,Rd | BNOT Rn,Rd | B | 6 | A | 1 | rd | | | | | | |
| | BNOT Rn,@ERd | B | 7 | D | 0:erd | 0 | 6 | 1 | m | 0 | | |
| | BNOT Rn,@aa:8 | B | 7 | F | abs | | 6 | 1 | m | 0 | | |
| | BNOT Rn,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | |
| | BNOT Rn,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | |
| BOR | BNOT Rn,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | |
| | BOR #xx:3,Rd | B | 7 | 4 | 0:IMM rd | | | | | | | |
| | BOR #xx:3,@ERd | B | 7 | C | 0:erd | 0 | 7 | 4 | 0:IMM 0 | | | |
| | BOR #xx:3,@aa:8 | B | 7 | E | abs | | 7 | 4 | 0:IMM 0 | | | |
| | BOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | |
| BSET | BOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | |
| | BSET #xx:3,Rd | B | 7 | 0 | 0:IMM rd | | | | | | | |
| | BSET #xx:3,@ERd | B | 7 | D | 0:erd | 0 | 7 | 0 | 0:IMM 0 | | | |
| | BSET #xx:3,@aa:8 | B | 7 | F | abs | | 7 | 0 | 0:IMM 0 | | | |
| | BSET #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | |
| BSET Rn,Rd | BSET #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | |
| | BSET Rn,Rd | B | 6 | 0 | m | rd | | | | | | |
| | BSET Rn,@ERd | B | 7 | D | 0:erd | 0 | 6 | 0 | m | 0 | | |
| | BSET Rn,@aa:8 | B | 7 | F | abs | | 6 | 0 | m | 0 | | |
| | BSET Rn,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | |
| BSR | BSET Rn,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | |
| | BSR d:8 | — | 5 | 5 | disp | | 6 | 0 | m | 0 | | |
| | BSR d:16 | — | 5 | C | 0 | 0 | | | | | | |
| | BST #xx:3,Rd | B | 6 | 7 | 0:IMM rd | | | | | | | |
| | BST #xx:3,@ERd | B | 7 | D | 0:erd | 0 | 6 | 7 | 0:IMM 0 | | | |
| BST | BST #xx:3,@aa:8 | B | 7 | F | abs | | 6 | 7 | 0:IMM 0 | | | |
| | BST #xx:3,@aa:16 | B | 6 | A | 1 | 8 | abs | | | | | |
| | BST #xx:3,@aa:32 | B | 6 | A | 3 | 8 | abs | | | | | |
| | | | | | | | 6 | 7 | 0:IMM 0 | | | |
| | | | | | | | abs | | | | | |

| Instruction | Mnemonic | Op 0 | Instruction Format | | | | | | | | | |
|-------------|-------------------|---------|----------------------------|-------------------|----------|--------------------|----------|--------------------|----------|--------------------|----------|-----------|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| | | | 7 3 | 0:IMM rd 0:erd | 7 3 | 0:IMM 0 0:IMM 0 | 7 3 | 0:IMM 0 0:IMM 0 | 7 3 | 0:IMM 0 0:IMM 0 | | |
| BTST | BTST #xx:3,Rd | B | 7 | 3 | | | | | | | | |
| | BTST #xx:3,@ERd | B | 7 | C | 0 | 7 | 3 | 0:IMM 0 | | | | |
| | BTST #xx:3,@aa:8 | B | 7 | E | | 7 | 3 | 0:IMM 0 | | | | |
| | BTST #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | |
| | BTST #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | |
| | BTST Rn,Rd | B | 6 | 3 | rn | rd | | | | | | |
| | BTST Rn,@ERd | B | 7 | C | 0:erd | 0 | 6 | 3 | rn | 0 | | |
| | BTST Rn,@aa:8 | B | 7 | E | abs | 6 | 3 | rn | 0 | | | |
| BXOR | BTST Rn,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | |
| | BTST Rn,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | |
| | BXOR #xx:3,Rd | B | 7 | 5 | 0:IMM rd | | | | | | | |
| | BXOR #xx:3,@ERd | B | 7 | C | 0:erd | 0 | 7 | 5 | 0:IMM 0 | | | |
| | BXOR #xx:3,@aa:8 | B | 7 | E | abs | 7 | 5 | 0:IMM 0 | | | | |
| | BXOR #xx:3,@aa:16 | B | 6 | A | 1 | 0 | abs | | | | | |
| | BXOR #xx:3,@aa:32 | B | 6 | A | 3 | 0 | abs | | | | | |
| | CLRMAC | — | Cannot be used in this LSI | | | | | | | | | |
| CMP | CMP.B #xx:3,Rd | B | A | rd | IMM | | | | | | | |
| | CMP.B Rs,Rd | B | 1 | C | rs | rd | | | | | | |
| | CMP.W #xx:16,Rd | W | 7 | 9 | 2 | rd | IMM | | | | | |
| | CMP.W Rs,Rd | W | 1 | D | rs | rd | | | | | | |
| | CMP.L #xx:32,ERd | L | 7 | A | 2 | 0:erd | | | | | | |
| | CMP.L ERs,ERd | L | 1 | F | 1:ers | 0:erd | | | | | | |
| | DAA Rd | B | 0 | F | 0 | rd | | | | | | |
| | DAS Rd | B | 1 | F | 0 | rd | | | | | | |
| DEC | DEC.B Rd | B | 1 | A | 0 | rd | | | | | | |
| | DEC.W #1,Rd | W | 1 | B | 5 | rd | | | | | | |
| | DEC.W #2,Rd | W | 1 | B | D | rd | | | | | | |
| | DEC.L #1,ERd | L | 1 | B | 7 | 0:erd | | | | | | |
| DIVXS | DEC.L #2,ERd | L | 1 | B | F | 0:erd | | | | | | |
| | DIVXS.B Rs,Rd | B | 0 | 1 | D | 0 | 5 | 1 | rs | rd | | |
| | DIVXS.W Rs,ERd | W | 0 | 1 | D | 0 | 5 | 3 | rs | 0:erd | | |
| | DIVXU.B Rs,Rd | B | 5 | 1 | rs | rd | | | | | | |
| DIVXU | DIVXU.W Rs,ERd | W | 5 | 3 | rs | 0:erd | | | | | | |
| | EEPMOV.B | — | 7 | B | 5 | C | 5 | 9 | 8 | F | | |
| | EEPMOV.W | — | 7 | B | D | 4 | 5 | 9 | 8 | F | | |
| | EXTS.W Rd | W | 1 | 7 | D | rd | | | | | | |
| EXTU | EXTS.L ERd | L | 1 | 7 | F | 0:erd | | | | | | |
| | EXTU.W Rd | W | 1 | 7 | 5 | rd | | | | | | |
| | EXTU.L ERd | L | 1 | 7 | 7 | 0:erd | | | | | | |
| | EXTU.L ERd | L | 1 | 7 | 7 | 0:erd | | | | | | |

| Instruction | Mnemonic | Size (B) | Instruction Format | | | | | | | | | |
|-------------|------------------------|----------|--------------------|--------------|------------------|----------------|----------|----------|----------|----------|----------|---------|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10 byte |
| INC | INC.B Rd | B | 0 : A | 0 : rd | | | | | | | | |
| | INC.W #1,Rd | W | 0 : B | 5 : rd | | | | | | | | |
| | INC.W #2,Rd | W | 0 : B | D : rd | | | | | | | | |
| | INC.L #1,ERd | L | 0 : B | 7 : 0 : :erd | | | | | | | | |
| | INC.L #2,ERd | L | 0 : B | F : 0 : :erd | | | | | | | | |
| JMP | JMP @ERn | — | 5 : 9 : 0 : :ern | 0 | | | | | | | | |
| | JMP @aa:24 | — | 5 : A | | | | | | | | | |
| | JMP @aa:8 | — | 5 : B | abs | | | | | | | | |
| | JMP @aa:24 | — | 5 : D | 0 : :ern | 0 | | | | | | | |
| JSR | JSR @ERn | — | 5 : E | | | | | | | | | |
| | JSR @aa:24 | — | 5 : F | abs | | | | | | | | |
| | JSR @aa:8 | — | 5 : 7 | IMM | | | | | | | | |
| | JSR @aa:24 | — | 5 : 7 | IMM | | | | | | | | |
| LDC | LDC #x:8,CCR | B | 0 : 1 | 4 : 1 | 0 : 7 | | | | | | | |
| | LDC #x:8,EXR | B | 0 : 1 | 4 : 1 | 0 : 7 | | | | | | | |
| | LDC Rs,CCR | B | 0 : 3 | 0 : rs | | | | | | | | |
| | LDC Rs,EXR | B | 0 : 3 | 1 : rs | | | | | | | | |
| | LDC @ERs,CCR | W | 0 : 1 | 4 : 0 | 6 : 9 : 0 : :ers | 0 | | | | | | |
| | LDC @ERs,EXR | W | 0 : 1 | 4 : 1 | 6 : 9 : 0 : :ers | 0 | | | | | | |
| | LDC @d:16,ERs),CCR | W | 0 : 1 | 4 : 0 | 6 : F | 0 : :ers | 0 | | | | | |
| | LDC @d:16,ERs),EXR | W | 0 : 1 | 4 : 1 | 6 : F | 0 : :ers | 0 | | | | | |
| | LDC @d:32,ERs),CCR | W | 0 : 1 | 4 : 0 | 7 : 8 | 0 : :ers | 0 | | | | | |
| | LDC @d:32,ERs),EXR | W | 0 : 1 | 4 : 1 | 7 : 8 | 0 : :ers | 0 | | | | | |
| | LDC @ERs+,CCR | W | 0 : 1 | 4 : 0 | 6 : D | 0 : :ers | 0 | | | | | |
| | LDC @ERs+,EXR | W | 0 : 1 | 4 : 1 | 6 : D | 0 : :ers | 0 | | | | | |
| LDM | LDC @aa:16,CCR | W | 0 : 1 | 4 : 0 | 6 : B | 0 : 0 | | | | | | |
| | LDC @aa:16,EXR | W | 0 : 1 | 4 : 1 | 6 : B | 0 : 0 | | | | | | |
| | LDC @aa:32,CCR | W | 0 : 1 | 4 : 0 | 6 : B | 2 : 0 | | | | | | |
| | LDC @aa:32,EXR | W | 0 : 1 | 4 : 1 | 6 : B | 2 : 0 | | | | | | |
| | LDML @SP+, (ERn-ERn+1) | L | 0 : 1 | 1 : 0 | 6 : D | 7 : 0:ern+1 | | | | | | |
| | LDML @SP+, (ERn-ERn+2) | L | 0 : 1 | 2 : 0 | 6 : D | 7 : 0:ern+2 | | | | | | |
| LDMAC | LDML @SP+, (ERn-ERn+3) | L | 0 : 1 | 3 : 0 | 6 : D | 7 : 0:ern+3 | | | | | | |
| | LDMAC ERs, MACH | L | | | | | | | | | | |
| MAC | LDMAC ERs, MACL | L | | | | | | | | | | |
| | MAC @ERn+, @ERm+ | — | | | | | | | | | | |
| MOV | MOV.B #x:8,Rd | B | F : rd | IMM | | | | | | | | |
| | MOV.B Rs,Rd | B | 0 : C | rs | rd | | | | | | | |
| | MOV.B @ERs,Rd | B | 6 : 8 | 0 : :ers | rd | | | | | | | |
| | MOV.B @d:16,ERs),Rd | B | 6 : E | 0 : :ers | rd | | | | | | | |
| | MOV.B @d:32,ERs),Rd | B | 7 : 8 | 0 : :ers | 0 | disp | | | | | | |
| | MOV.B @ERs+,Rd | B | 6 : C | 0 : :ers | rd | | | | | | | |
| | MOV.B @aa:8,Rd | B | 2 : rd | abs | | | | | | | | |
| | MOV.B @aa:16,Rd | B | 6 : A | 0 : rd | abs | | | | | | | |
| | MOV.B @aa:32,Rd | B | 6 : A | 2 : rd | abs | | | | | | | |
| | MOV.B Rs,@ERd | B | 6 : 8 | 1 : :erd | rs | | | | | | | |
| | MOV.B Rs,@d:16,ERd) | B | 6 : E | 1 : :erd | rs | disp | | | | | | |
| | MOV.B Rs,@d:32,ERd) | B | 7 : 8 | 0 : :erd | 0 | 6 : A : A : rs | | | | | | |

| Instruction | Mnemonic | Op code | Instruction Format | | | | | | | | | |
|----------------|-------------------------|------------|----------------------------|----------|----------|----------|----------|----------|----------|----------|----------|-----------|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| MOV (Cont.) | MOV.B Rs,@ERd | B | 6 | C | 1:erd | rs | | | | | | |
| | MOV.B Rs,aa:8 | B | 3 | rs | abs | | | | | | | |
| | MOV.B Rs,aa:16 | B | 6 | A | 8 | rs | abs | | | | | |
| | MOV.B Rs,aa:32 | B | 6 | A | A | rs | | | | | | |
| | MOV.W #xx:16,Rd | W | 7 | 9 | 0 | rd | | | | | | |
| | MOV.W Rs,Rd | W | 0 | D | rs | rd | | | | | | |
| | MOV.W @ERs,Rd | W | 6 | 9 | 0:ers | rd | | | | | | |
| | MOV.W @(d:16,ERs),Rd | W | 6 | F | 0:ers | rd | | | | | | |
| | MOV.W @(d:32,ERs),Rd | W | 7 | 8 | 0:ers | 0 | 6 | B | 2 | rd | disp | |
| | MOV.W @ERs+,Rd | W | 6 | D | 0:ers | rd | | | | | | |
| | MOV.W @aa:16,Rd | W | 6 | B | 0 | rd | abs | | | | | |
| | MOV.W @aa:32,Rd | W | 6 | B | 2 | rd | | | | | | |
| | MOV.W Rs,@ERd | W | 6 | 9 | 1:erd | rs | | | | | | |
| | MOV.W Rs,@(d:16,ERd) | W | 6 | F | 1:erd | rs | | | | | | |
| | MOV.W Rs,@(d:32,ERd) | W | 7 | 8 | 0:erd | 0 | 6 | B | A | rs | disp | |
| | MOV.W Rs,@-ERd | W | 6 | D | 1:erd | rs | | | | | | |
| | MOV.W Rs,@aa:16 | W | 6 | B | 8 | rs | abs | | | | | |
| | MOV.W Rs,@aa:32 | W | 6 | B | A | rs | | | | | | |
| | MOV.L #xx:32,Rd | W | 7 | A | 0 | 0:erd | | | | | | |
| | MOV.L ERs,ERd | L | 0 | F | 1:ers | 0:erd | | | | | | |
| | MOV.L @ERs,ERd | L | 0 | 1 | 0 | 0 | 6 | 9 | 0:ers | 0 | erd | |
| | MOV.L @(d:16,ERs),ERd | L | 0 | 1 | 0 | 0 | 6 | F | 0:ers | 0 | erd | |
| | MOV.L @(d:32,ERs),ERd | L | 0 | 1 | 0 | 0 | 7 | 8 | 0:ers | 0 | | disp |
| | MOV.L @ERs+,ERd | L | 0 | 1 | 0 | 0 | 6 | D | 0:ers | 0 | erd | |
| | MOV.L @aa:16,ERd | L | 0 | 1 | 0 | 0 | 6 | B | 0 | 0 | erd | |
| | MOV.L @aa:32,ERd | L | 0 | 1 | 0 | 0 | 6 | B | 2 | 0 | erd | |
| | MOV.L ERs,@ERd | L | 0 | 1 | 0 | 0 | 6 | 9 | 1:erd | 0 | ers | |
| | MOV.L ERs,@(d:16,ERd) | L | 0 | 1 | 0 | 0 | 6 | F | 1:erd | 0 | ers | |
| | MOV.L ERs,@(d:32,ERd)*1 | L | 0 | 1 | 0 | 0 | 7 | 8 | 0:erd | 0 | | |
| | MOV.L ERs,@-ERd | L | 0 | 1 | 0 | 0 | 6 | D | 1:erd | 0 | ers | disp |
| | MOV.L ERs,@aa:16 | L | 0 | 1 | 0 | 0 | 6 | B | 8 | 0 | ers | |
| | MOV.L ERs,@aa:32 | L | 0 | 1 | 0 | 0 | 6 | B | A | 0 | ers | |
| MOV.FE | MOV.FE @aa:16,Rd | B | Cannot be used in this LSI | | | | | | | | | |
| MOV.TPE | MOV.TPE Rs,@aa:16 | B | | | | | | | | | | |
| MUL.XS | MUL.XS.B Rs,Rd | B | 0 | 1 | C | 0 | 5 | 0 | rs | rd | | |
| MUL.XU | MUL.XS.W Rs,ERd | W | 0 | 1 | C | 0 | 5 | 2 | rs | 0:erd | | |
| | MUL.XU.B Rs,Rd | B | 5 | 0 | rs | rd | | | | | | |
| NEG | MUL.XU.W Rs,ERd | W | 5 | 2 | rs | 0:erd | | | | | | |
| | NEG.B Rd | B | 1 | 7 | 8 | rd | | | | | | |
| | NEG.W Rd | W | 1 | 7 | 9 | rd | | | | | | |
| NOP | NEGL ERd | L | 1 | 7 | B | 0:erd | | | | | | |
| | NOP | — | 0 | 0 | 0 | 0 | | | | | | |

| Instruction | Mnemonic | Size [B] | Instruction Format | | | | | | | | | |
|-------------|-----------------|-------------|--------------------|-----------|----------|----------------|----------|----------|----------|----------|----------|-----------|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| NOT | NOT.B Rd | B | 1 7 | 0 rd | | | | | | | | |
| | NOT.W Rd | W | 1 7 | 1 rd | | | | | | | | |
| | NOT.L ERd | L | 1 7 | 3 :0 :erd | | | | | | | | |
| OR | OR.B #xx:8.Rd | B | C rd | IMM | | | | | | | | |
| | OR.B Rs.Rd | B | 1 4 | rs rd | | | | | | | | |
| | OR.W #xx:16.Rd | W | 7 9 | 4 rs rd | | | | | | | | |
| | OR.W Rs.Rd | W | 6 4 | rs rd | | IMM | | | | | | |
| | OR.L #xx:32.ERd | L | 7 A | 4 :0 :erd | | | | | | | | |
| | OR.L ERs.ERd | L | 0 4 | F 0 | 6 4 | 0 :ers :0 :erd | | | | | | |
| ORC | ORC #xx:8.CCR | B | 0 4 | IMM | | | | | | | | |
| | ORC #xx:8.EXR | B | 0 1 | 4 1 | 0 4 | IMM | | | | | | |
| POP | POP.W Rn | W | 6 D | 7 rm | | | | | | | | |
| | POP.L ERn | L | 0 1 | 0 0 | 6 D | 7 :0 :ern | | | | | | |
| PUSH | PUSH.W Rn | W | 6 D | F rm | | | | | | | | |
| | PUSH.L ERn | L | 0 1 | 0 0 | 6 D | F :0 :ern | | | | | | |
| ROTL | ROTL.B Rd | B | 1 2 | 8 rd | | | | | | | | |
| | ROTL.W Rd | W | 1 2 | 9 rd | | | | | | | | |
| | ROTL.W #2. Rd | W | 1 2 | D rd | | | | | | | | |
| | ROTL.L ERd | L | 1 2 | B :0 :erd | | | | | | | | |
| | ROTL.L #2. ERd | L | 1 2 | F :0 :erd | | | | | | | | |
| | ROTL.B Rd | B | 1 3 | 8 rd | | | | | | | | |
| ROTR | ROTR.B #2. Rd | B | 1 3 | C rd | | | | | | | | |
| | ROTR.W Rd | W | 1 3 | 9 rd | | | | | | | | |
| | ROTR.W #2. Rd | W | 1 3 | D rd | | | | | | | | |
| | ROTR.L ERd | L | 1 3 | B :0 :erd | | | | | | | | |
| | ROTR.L #2. ERd | L | 1 3 | F :0 :erd | | | | | | | | |
| | ROTXL.B Rd | B | 1 2 | 0 rd | | | | | | | | |
| ROTXL | ROTXL.B #2. Rd | B | 1 2 | 4 rd | | | | | | | | |
| | ROTXL.W Rd | W | 1 2 | 1 rd | | | | | | | | |
| | ROTXL.W #2. Rd | W | 1 2 | 5 rd | | | | | | | | |
| | ROTXL.L ERd | L | 1 2 | 3 :0 :erd | | | | | | | | |
| | ROTXL.L #2. ERd | L | 1 2 | 7 :0 :erd | | | | | | | | |
| | ROTXR.B Rd | B | 1 3 | 0 rd | | | | | | | | |
| ROTXR | ROTXR.B #2. Rd | B | 1 3 | 4 rd | | | | | | | | |
| | ROTXR.W Rd | W | 1 3 | 1 rd | | | | | | | | |
| | ROTXR.W #2. Rd | W | 1 3 | 5 rd | | | | | | | | |
| | ROTXR.L ERd | L | 1 3 | 3 :0 :erd | | | | | | | | |
| | ROTXR.L #2. ERd | L | 1 3 | 7 :0 :erd | | | | | | | | |
| | RTE | — | 5 6 | 7 0 | | | | | | | | |
| RTS | RTS | — | 5 4 | 7 0 | | | | | | | | |

| Instruction | Mnemonic | 9 10 | Instruction Format | | | | | | | | | |
|-------------|-----------------------|---------|--------------------|----------|----------|----------|----------|----------|----------|----------|----------|-----------|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| SHAL | SHAL.B Rd | B | 1 | 0 | C | rd | | | | | | |
| | SHAL.B #2, Rd | B | 1 | 0 | 8 | rd | | | | | | |
| | SHAL.W Rd | W | 1 | 0 | 9 | rd | | | | | | |
| | SHAL.W #2, Rd | W | 1 | 0 | D | rd | | | | | | |
| | SHAL.L ERd | L | 1 | 0 | B | 0 : erd | | | | | | |
| | SHAL.L #2, ERd | L | 1 | 0 | F | 0 : erd | | | | | | |
| SHAR | SHAR.B Rd | B | 1 | 1 | 8 | rd | | | | | | |
| | SHAR.B #2, Rd | B | 1 | 1 | C | rd | | | | | | |
| | SHAR.W Rd | W | 1 | 1 | 9 | rd | | | | | | |
| | SHAR.W #2, Rd | W | 1 | 1 | D | rd | | | | | | |
| | SHAR.L ERd | L | 1 | 1 | B | 0 : erd | | | | | | |
| | SHAR.L #2, ERd | L | 1 | 1 | F | 0 : erd | | | | | | |
| SHLL | SHLL.B Rd | B | 1 | 0 | 0 | rd | | | | | | |
| | SHLL.B #2, Rd | B | 1 | 0 | 4 | rd | | | | | | |
| | SHLL.W Rd | W | 1 | 0 | 1 | rd | | | | | | |
| | SHLL.W #2, Rd | W | 1 | 0 | 5 | rd | | | | | | |
| | SHLL.L ERd | L | 1 | 0 | 3 | 0 : erd | | | | | | |
| | SHLL.L #2, ERd | L | 1 | 0 | 7 | 0 : erd | | | | | | |
| SHLR | SHLR.B Rd | B | 1 | 1 | 0 | rd | | | | | | |
| | SHLR.B #2, Rd | B | 1 | 1 | 4 | rd | | | | | | |
| | SHLR.W Rd | W | 1 | 1 | 1 | rd | | | | | | |
| | SHLR.W #2, Rd | W | 1 | 1 | 5 | rd | | | | | | |
| | SHLR.L ERd | L | 1 | 1 | 3 | 0 : erd | | | | | | |
| | SHLR.L #2, ERd | L | 1 | 1 | 7 | 0 : erd | | | | | | |
| SLEEP | SLEEP | — | 0 | 1 | 8 | 0 | | | | | | |
| | STC.B CCR.Rd | B | 0 | 2 | 0 | rd | | | | | | |
| STC | STC.B EXR.Rd | B | 0 | 2 | 1 | rd | | | | | | |
| | STC.W CCR.@ERd | W | 0 | 1 | 4 | 0 | 6 | 9 | 1 : erd | 0 | | |
| | STC.W EXR.@ERd | W | 0 | 1 | 4 | 1 | 6 | 9 | 1 : erd | 0 | | |
| | STC.W CCR.@(d:16)ERd) | W | 0 | 1 | 4 | 0 | 6 | F | 1 : erd | 0 | | |
| | STC.W EXR.@(d:16)ERd) | W | 0 | 1 | 4 | 1 | 6 | F | 1 : erd | 0 | | |
| | STC.W CCR.@(d:32)ERd) | W | 0 | 1 | 4 | 0 | 7 | 8 | 0 : erd | 0 | disp | |
| | STC.W EXR.@(d:32)ERd) | W | 0 | 1 | 4 | 1 | 7 | 8 | 0 : erd | 0 | disp | |
| | STC.W CCR.@-ERd | W | 0 | 1 | 4 | 0 | 6 | D | 1 : erd | 0 | | |
| | STC.W EXR.@-ERd | W | 0 | 1 | 4 | 1 | 6 | D | 1 : erd | 0 | | |
| | STC.W CCR.@aa:16 | W | 0 | 1 | 4 | 0 | 6 | B | 8 | 0 | abs | |
| | STC.W EXR.@aa:16 | W | 0 | 1 | 4 | 1 | 6 | B | 8 | 0 | abs | |
| | STC.W CCR.@aa:32 | W | 0 | 1 | 4 | 0 | 6 | B | A | 0 | abs | |
| STM | STC.W EXR.@aa:32 | W | 0 | 1 | 4 | 1 | 6 | B | A | 0 | abs | |
| | STML(ERn-ERn+1), @-SP | L | 0 | 1 | 1 | 0 | 6 | D | F | 0 : ern | | |
| | STML(ERn-ERn+2), @-SP | L | 0 | 1 | 2 | 0 | 6 | D | F | 0 : ern | | |
| | STML(ERn-ERn+3), @-SP | L | 0 | 1 | 3 | 0 | 6 | D | F | 0 : ern | | |
| STMAC | STMAC MACL.ERd | L | | | | | | | | | | |
| | STMAC MACL.ERd | L | | | | | | | | | | |

Cannot be used in this LSI

| Instruction | Mnemonic | Size [B] | Instruction Format | | | | | | | | | |
|-------------|------------------------|-------------|--------------------|----------|----------|-------------|----------|----------|----------|----------|----------|-----------|
| | | | 1st byte | 2nd byte | 3rd byte | 4th byte | 5th byte | 6th byte | 7th byte | 8th byte | 9th byte | 10th byte |
| SUB | SUB B Rs,Rd | B | 1 8 | rs rd | | | | | | | | |
| | SUB.W #xx:16,Rd | W | 7 9 | 3 rd | | | | | | | | |
| | SUB.W Rs,Rd | W | 1 9 | rs rd | | IMM | | | | | | |
| | SUB.L #xx:32,ERd | L | 7 A | 3 0:erd | | | | | | | | |
| | SUB.L ERs,ERd | L | 1 A | 1 0:erd | | | | | | | | |
| SUBS | SUBS #1,ERd | L | 1 B | 0 0:erd | | | | | | | | |
| | SUBS #2,ERd | L | 1 B | 8 0:erd | | | | | | | | |
| | SUBS #4,ERd | L | 1 B | 9 0:erd | | | | | | | | |
| | SUBX #xx:8,Rd | B | B rd | IMM | | | | | | | | |
| | SUBX Rs,Rd | B | 1 E | rs rd | | | | | | | | |
| TAS | TAS @ERd ^{#2} | B | 0 1 | E 0 | 7 B | 0:erd C | | | | | | |
| TRAPA | TRAPA #x:2 | — | 5 7 | 00:IMM | 0 | | | | | | | |
| XOR | XOR B #xx:8,Rd | B | D rd | IMM | | | | | | | | |
| | XOR B Rs,Rd | B | 1 5 | rs rd | | | | | | | | |
| | XOR.W #xx:16,Rd | W | 7 9 | 5 rd | | | | | | | | |
| | XOR.W Rs,Rd | W | 6 5 | rs rd | | IMM | | | | | | |
| | XOR.L #xx:32,ERd | L | 7 A | 5 0:erd | | | | | | | | |
| XORC | XOR.L ERs,ERd | L | 0 1 | F 0 | 6 5 | 0:ers 0:erd | | | | | | |
| | XORC #xx:8,CCR | B | 0 5 | IMM | | | | | | | | |
| | XORC #xx:8,EXR | B | 0 1 | 4 1 | 0 5 | IMM | | | | | | |

Notes: *1 Either 1 or 0 can be set to bit 7 in 4th byte of MOV.L ERs, @(d: 32, ERd) instruction.

*2 Only register ER0,ER1,ER4, or ER5 should be used when using the TAS instruction.

[Legend]

IMM: Immediate data (2, 3, 8, 16, 32 bits)

abs: Absolute address (8, 16, 24, 32 bits)

disp: Displacement (8, 16, 32 bits)

rs, rd, rn: Register fields (8-bit register or 16-bit register is selected in 4 bits. rs, rd and rn correspond to the operand type Rs, Rd, and Rn respectively.)

ers, erd, ern, erm: Register fields (address register or 32-bit register is selected in 3 bits. ers, erd, ern and erm correspond to the operand type ERs, ERd, ERn and Rm respectively.)

The following table shows the correspondence between the register field and the general register.

| Address Register, 32-bit Register | | 16-bit Register | | 8-bit Register | |
|-----------------------------------|------------------|-----------------|------------------|----------------|------------------|
| Register Field | General Register | Register Field | General Register | Register Field | General Register |
| 000 | ER0 | 0000 | R0 | 0000 | R0H |
| 001 | ER1 | 0001 | R1 | 0001 | R1H |
| : | : | : | : | : | : |
| : | : | : | : | : | : |
| : | : | : | : | : | : |
| : | : | : | : | : | : |
| 111 | ER7 | 0111 | R7 | 0111 | R7H |
| | | 1000 | E0 | 1000 | R0L |
| | | 1001 | E1 | 1001 | R1L |
| | | : | : | : | : |
| | | : | : | : | : |
| | | : | : | : | : |
| | | : | : | : | : |
| | | 1111 | E7 | 1111 | R7L |

A.3 Operation Code Map

Table A.3 shows an operation code map.

Table A.3 Operation Code Map (1)

Instruction code:

1st byte2nd byte

AH ALBH BL

BH highest bit is set to 0.

BH highest bit is set to 1

| AL/AH | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|-------|-------|--------------|-------|-------|------|-------|-------|--------------|--------------|--------------|--------------|--------------|-----|-----|------|--------------|
| 0 | NOP | Table A.3(2) | STC | LDC | ORC | XORC | ANDC | LDC | ADD | | Table A.3(2) | Table A.3(2) | MOV | | ADDX | Table A.3(2) |
| 1 | | Table A.3(2) | STMAC | LDMAC | OR | XOR | AND | Table A.3(2) | | SUB | Table A.3(2) | Table A.3(2) | CMP | | SUBX | Table A.3(2) |
| 2 | MOV.B | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | |
| 4 | BRA | BRN | BHI | BLS | BCC | BCS | BNE | BEQ | BVC | BVS | BPL | BMI | BGE | BLT | BGT | BLE |
| 5 | MULXU | DIVXU | MULXU | DIVXU | RTS | BSR | RTE | TRAPA | Table A.3(2) | | JMP | | BSR | | JSR | |
| 6 | | | | | OR | XOR | AND | BST | MOV | | Table A.3(2) | | | MOV | | |
| 7 | BSET | BNOT | BCLR | BTST | BOR | BXOR | BAND | BLD | MOV | Table A.3(2) | EEP | MOV | | | | |
| | | | | | BIOR | BIXOR | BIAND | BILD | | | Table A.3(2) | | | | | |
| 8 | ADD | | | | | | | | | | | | | | | |
| 9 | ADDX | | | | | | | | | | | | | | | |
| A | CMP | | | | | | | | | | | | | | | |
| B | SUBX | | | | | | | | | | | | | | | |
| C | OR | | | | | | | | | | | | | | | |
| D | XOR | | | | | | | | | | | | | | | |
| E | AND | | | | | | | | | | | | | | | |
| F | MOV | | | | | | | | | | | | | | | |

Note: * Cannot be used in this LSI

Table A.3 Operation Code Map (2)

| Instruction code: | | 1st byte | | 2nd byte | |
|-------------------|----|----------|----|----------|----|
| BH AH | AL | 1 | 2 | 3 | 4 |
| | | AH | AL | BH | BL |

| BH AH AL | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F | | | |
|-------------|------|-----------------|-----|-----------------|--------|------|-----|-------|-------|------|---------|-----|-----------------|-----------------|-----|-----------------|-----|--|--|
| 01 | MOV | LDM | STM | | LDC | STC | | | SLEEP | | CLRMAC* | | Table A.3(3) | Table A.3(3) | TAS | Table A.3(3) | | | |
| 0A | INC | ADD | | | | | | | | | | | | | | | | | |
| 0B | ADDS | | | | | | INC | | INC | ADDS | | | | | INC | | INC | | |
| 0F | DAA | MOV | | | | | | | | | | | | | | | | | |
| 10 | | SHLL | | | SHLL | | | SHLL | SHAL | | | | SHAL | | | SHAL | | | |
| 11 | | SHLR | | | SHLR | | | SHLR | SHAR | | | | SHAR | | | SHAR | | | |
| 12 | | ROTXL | | | ROTXL | | | ROTXL | ROTL | | | | ROTL | | | ROTL | | | |
| 13 | | ROTXR | | | ROTXR | | | ROTXR | ROTR | | | | ROTR | | | ROTR | | | |
| 17 | | NOT | | | NOT | EXTU | | EXTU | NEG | | | NEG | | EXTS | | EXTS | | | |
| 1A | DEC | SUB | | | | | | | | | | | | | | | | | |
| 1B | SUBS | | | | | DEC | | DEC | SUBS | | | | | DEC | | DEC | | | |
| 1F | DAS | CMP | | | | | | | | | | | | | | | | | |
| 58 | BRA | BRN | BHI | BLS | BCC | BCS | BNE | BEQ | BVC | BVS | BPL | BMI | BGE | BLT | BGT | BLE | | | |
| 6A | MOV | Table A.3(4) | MOV | Table A.3(4) | MOVFP* | | | | MOV | | MOV | | | MOVTP* | | | | | |
| 79 | MOV | ADD | CMP | SUB | OR | XOR | AND | | | | | | | | | | | | |
| 7A | MOV | ADD | CMP | SUB | OR | XOR | AND | | | | | | | | | | | | |

Note: * Cannot be used in this LSI

Table A.3 Operation Code Map (3)

Instruction code:

| 1st byte | | 2nd byte | | 3rd byte | | 4th byte | |
|----------|----|----------|----|----------|----|----------|----|
| AH | AL | BH | BL | CH | CL | DH | DL |



| CL | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|-------------|-------|-------|-------|-------|-------------|---------------|---------------|--------------------|---|---|---|---|---|---|---|---|
| AH/AL/BH/CH | MULXS | | MULXS | | | | | | | | | | | | | |
| 01C05 | | DIVXS | | DIVXS | | | | | | | | | | | | |
| 01D05 | | | | | OR | XOR | AND | | | | | | | | | |
| 01F06 | | | | | | | | | | | | | | | | |
| 7C06 *1 | | | | BTST | | | | | | | | | | | | |
| 7C07 *1 | | | | BTST | BOR BIOR | BXOR BIXOR | BAND BIAND | BLD BILD BST | | | | | | | | |
| 7D06 *1 | BSET | BNOT | BCLR | | | | | | | | | | | | | |
| 7D07 *1 | BSET | BNOT | BCLR | | | | | | | | | | | | | |
| 7Eaa6 *2 | | | | BTST | | | | | | | | | | | | |
| 7Eaa7 *2 | | | | BTST | | | | | | | | | | | | |
| 7Faa6 *2 | BSET | BNOT | BCLR | | BOR BIOR | BXOR BIXOR | BAND BIAND | BLD BILD BST | | | | | | | | |
| 7Faa7 *2 | BSET | BNOT | BCLR | | | | | | | | | | | | | |

Notes: 1. r is the register specification section.
2. Absolute address is set at aa.

Table A.3 Operation Code Map (4)

| Instruction code: | | | | | | | | | | | | | | | | |
|--|------|----------|------|----------|------|----------|-------|----------|----|----------|----|---|---|---|---|---|
| 1st byte | | 2nd byte | | 3th byte | | 4th byte | | 5th byte | | 6th byte | | | | | | |
| AH | AL | BH | BL | CH | CL | DH | DL | EH | EL | FH | FL | | | | | |
| AHLBHLCHDLHLER 6A10aaaa6* 6A10aaaa7* 6A18aaaa6* 6A18aaaa7* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | | | | BTST | BOR | BXOR | BAND | BLD | | | | | | | | |
| | | | | | BIOR | BIXOR | BIAND | BILD | | | | | | | | |
| | | | | | | | | BST | | | | | | | | |
| | BSET | | BNOT | BCLR | | | | | | | | | | | | |

FH highest bit is set to 0.
FH highest bit is set to 1.

| Instruction code: | | | | | | | | | | | | | | | | |
|--|------|----------|------|----------|------|----------|-------|----------|----|----------|----|----------|----|----------|----|---|
| 1st byte | | 2nd byte | | 3rd byte | | 4th byte | | 5th byte | | 6th byte | | 7th byte | | 8th byte | | |
| AH | AL | BH | BL | CH | CL | DH | DL | EH | EL | FH | FL | GH | GL | HH | HL | |
| GL AHLBHL...RFLGH 6A30aaaaaaa6* 6A30aaaaaaa7* 6A38aaaaaaa6* 6A38aaaaaaa7* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| | | | | BTST | BOR | BXOR | BAND | BLD | | | | | | | | |
| | | | | | BIOR | BIXOR | BIAND | BILD | | | | | | | | |
| | | | | | | | | BST | | | | | | | | |
| | BSET | | BNOT | BCLR | | | | | | | | | | | | |

HH highest bit is set to 0.
HH highest bit is set to 1.

Note: * Absolute address is set at aa.

A.4 Number of Execution States

This section explains execution state and how to calculate the number of execution states for each instruction of the H8S/2194 CPU.

Table A.5 indicates number of cycles of instruction fetch and data read/write during instruction execution, and table A.4 indicates number of states required for each instruction size.

The number of execution states can be obtained from the equation below.

$$\text{Number of execution states} = I \cdot S_I + J \cdot S_J + K \cdot S_K + L \cdot S_L + M \cdot S_M + N \cdot S_N$$

(1) Examples of execution state number calculation

The conditions are as follows: In advanced mode, program and stack areas are set in the on-chip memory, a wait is inserted every 2 states in the on-chip supporting module access with 8-bit bus width.

1. BSET #0, @FFFFC7:8

From Table A.5,

$$I = L = 2, J = K = M = N = 0$$

From Table A.4,

$$S_I = 1, S_L = 2$$

$$\text{Number of execution states} = 2 \times 1 + 2 \times 2 = 6$$

2. JSR @@30

From Table A.5,

$$I = J = K = 2, L = M = N = 0$$

From Table A.4,

$$S_I = S_J = S_K = 1$$

$$\text{Number of execution states} = 2 \times 1 + 2 \times 1 + 2 \times 1 = 6$$

Table A.4 Number of States Required for Each Execution Status (Cycle)

| Execution Status (Cycle) | Target of Access | | |
|---------------------------|------------------|---------------------------|------------|
| | On-Chip Memory | On-Chip Supporting Module | |
| | | 8-bit bus | 16-bit bus |
| Instruction fetch S_i | 1 | — | — |
| Branch address read S_j | | | |
| Stack operation S_k | | | |
| Byte data access S_L | | 2 | 2 |
| Word data access S_M | | 4 | |
| Internal operation S_N | 1 | | |

Table A.5 Instruction Execution Status (No. of Cycles)

| Instruction | Mnemonic | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|-------------|---------------------|----------------------|---------------------------|--------------------|---------------------|---------------------|-----------------------|
| | | I | J | K | L | M | N |
| ADD | ADD.B #xx:8,Rd | 1 | | | | | |
| | ADD.B Rs, Rd | 1 | | | | | |
| | ADD.W #xx:16,Rd | 2 | | | | | |
| | ADD.W Rs,Rd | 1 | | | | | |
| | ADD.L #xx:32,ERd | 3 | | | | | |
| | ADD.L ERs,ERd | 1 | | | | | |
| ADDS | ADDS #1/2/4,ERd | 1 | | | | | |
| ADDX | ADDX #xx:8,Rd | 1 | | | | | |
| | ADDX Rs,Rd | 1 | | | | | |
| AND | AND.B #xx:8,Rd | 1 | | | | | |
| | AND.B Rs,Rd | 1 | | | | | |
| | AND.W #xx:16,Rd | 2 | | | | | |
| | AND.W Rs,Rd | 1 | | | | | |
| | AND.L #xx:32,ERd | 3 | | | | | |
| | AND.L ERs,ERd | 2 | | | | | |
| ANDC | ANDC #xx:8,CCR | 1 | | | | | |
| | ANDC #xx:8,EXR | 2 | | | | | |
| BAND | BAND #xx:3,Rd | 1 | | | | | |
| | BAND #xx:3,@ERd | 2 | | | 1 | | |
| | BAND #xx:3@aa:8 | 2 | | | 1 | | |
| | BAND #xx:3@aa:16 | 3 | | | 1 | | |
| | BAND #xx:3@aa:32 | 4 | | | 1 | | |
| Bcc | BRA d:8 (BT d:8) | 2 | | | | | |
| | BRN d:8 (BF d:8) | 2 | | | | | |
| | BHI d:8 | 2 | | | | | |
| | BLS d:8 | 2 | | | | | |
| | BCC d:8 (BHS d:8) | 2 | | | | | |
| | BCS d:8 (BLO d:8) | 2 | | | | | |
| | BNE d:8 | 2 | | | | | |
| | BEQ d:8 | 2 | | | | | |
| | BVC d:8 | 2 | | | | | |
| | BVS d:8 | 2 | | | | | |
| | BPL d:8 | 2 | | | | | |
| | BMI d:8 | 2 | | | | | |
| | BGE d:8 | 2 | | | | | |
| | BLT d:8 | 2 | | | | | |
| | BGT d:8 | 2 | | | | | |
| | BLE d:8 | 2 | | | | | |
| | BRA d:16 (BT d:16) | 2 | | | | | 1 |
| | BRN d:16 (BF d:16) | 2 | | | | | 1 |
| | BHI d:16 | 2 | | | | | 1 |
| | BLS d:16 | 2 | | | | | 1 |
| | BCC d:16 (BHS d:16) | 2 | | | | | 1 |
| | BCS d:16 (BLO d:16) | 2 | | | | | 1 |
| | BNE d:16 | 2 | | | | | 1 |
| | BEQ d:16 | 2 | | | | | 1 |
| | BVC d:16 | 2 | | | | | 1 |
| | BVS d:16 | 2 | | | | | 1 |
| | BPL d:16 | 2 | | | | | 1 |
| | BMI d:16 | 2 | | | | | 1 |
| | BGE d:16 | 2 | | | | | 1 |
| | BLT d:16 | 2 | | | | | 1 |

| Instruction | Mnemonic | Instruction | Branch | Stack | Byte Data | Word Data | Internal |
|-------------|--------------------|-------------|---------|-------|-----------|-----------|----------|
| | | Fetch | Address | | | | |
| | | I | J | K | L | M | N |
| Bcc | BGT d:16 | 2 | | | | | 1 |
| | BLE d:16 | 2 | | | | | 1 |
| BCLR | BCLR #xx:3,Rd | 1 | | | | | |
| | BCLR #xx:3,@ERd | 2 | | | 2 | | |
| | BCLR #xx:3,@aa:8 | 2 | | | 2 | | |
| | BCLR #xx:3,@aa:16 | 3 | | | 2 | | |
| | BCLR #xx:3,@aa:32 | 4 | | | 2 | | |
| | BCLR Rn,Rd | 1 | | | | | |
| | BCLR Rn,@ERd | 2 | | | 2 | | |
| | BCLR Rn,@aa:8 | 2 | | | 2 | | |
| | BCLR Rn,@aa:16 | 3 | | | 2 | | |
| | BCLR Rn,@aa:32 | 4 | | | 2 | | |
| BIAND | BIAND #xx:3,Rd | 1 | | | | | |
| | BIAND #xx:3,@ERd | 2 | | | 1 | | |
| | BIAND #xx:3,@aa:8 | 2 | | | 1 | | |
| | BIAND #xx:3,@aa:16 | 3 | | | 1 | | |
| | BIAND #xx:3,@aa:32 | 4 | | | 1 | | |
| BILD | BILD #xx:3,Rd | 1 | | | | | |
| | BILD #xx:3,@ERd | 2 | | | 1 | | |
| | BILD #xx:3,@aa:8 | 2 | | | 1 | | |
| | BILD #xx:3,@aa:16 | 3 | | | 1 | | |
| | BILD #xx:3,@aa:32 | 4 | | | 1 | | |
| BIOR | BIOR #xx:8,Rd | 1 | | | | | |
| | BIOR #xx:8,@ERd | 2 | | | 1 | | |
| | BIOR #xx:8,@aa:8 | 2 | | | 1 | | |
| | BIOR #xx:8,@aa:16 | 3 | | | 1 | | |
| | BIOR #xx:8,@aa:32 | 4 | | | 1 | | |
| BIST | BIST #xx:3,Rd | 1 | | | | | |
| | BIST #xx:3,@ERd | 2 | | | 2 | | |
| | BIST #xx:3,@aa:8 | 2 | | | 2 | | |
| | BIST #xx:3,@aa:16 | 3 | | | 2 | | |
| | BIST #xx:3,@aa:32 | 4 | | | 2 | | |
| BIXOR | BIXOR #xx:3,Rd | 1 | | | | | |
| | BIXOR #xx:3,@ERd | 2 | | | 1 | | |
| | BIXOR #xx:3,@aa:8 | 2 | | | 1 | | |
| | BIXOR #xx:3,@aa:16 | 3 | | | 1 | | |
| | BIXOR #xx:3,@aa:32 | 4 | | | 1 | | |
| BLD | BLD #xx:3,Rd | 1 | | | | | |
| | BLD #xx:3,@ERd | 2 | | | 1 | | |
| | BLD #xx:3,@aa:8 | 2 | | | 1 | | |
| | BLD #xx:3,@aa:16 | 3 | | | 1 | | |
| | BLD #xx:3,@aa:32 | 4 | | | 1 | | |
| BNOT | BNOT #xx:3,Rd | 1 | | | | | |
| | BNOT #xx:3,@ERd | 2 | | | 2 | | |
| | BNOT #xx:3,@aa:8 | 2 | | | 2 | | |
| | BNOT #xx:3,@aa:16 | 3 | | | 2 | | |
| | BNOT #xx:3,@aa:32 | 4 | | | 2 | | |
| | BNOT Rn,Rd | 1 | | | | | |
| | BNOT Rn,@ERd | 2 | | | 2 | | |
| | BNOT Rn,@aa:8 | 2 | | | 2 | | |
| | BNOT Rn,@aa:16 | 3 | | | 2 | | |

| Instruction | Mnemonic | Instruction | Branch | Stack | Byte Data | Word Data | Internal |
|-------------|-------------------|-----------------------------|---------|-------|-----------|-----------|----------|
| | | Fetch | Address | | | | |
| | | I | J | K | L | M | N |
| BNOT | BNOT Rn,@aa:32 | 4 | | | 2 | | |
| BOR | BOR #xx:3,Rd | 1 | | | | | |
| | BOR #xx:3,@ERd | 2 | | | 1 | | |
| | BOR #xx:3,@aa:8 | 2 | | | 1 | | |
| | BOR #xx:3,@aa:16 | 3 | | | 1 | | |
| | BOR #xx:3,@aa:32 | 4 | | | 1 | | |
| BSET | BSET #xx:3,Rd | 1 | | | | | |
| | BSET #xx:3,@ERd | 2 | | | 2 | | |
| | BSET #xx:3,@aa:8 | 2 | | | 2 | | |
| | BSET #xx:3,@aa:16 | 3 | | | 2 | | |
| | BSET #xx:3,@aa:32 | 4 | | | 2 | | |
| | BSET Rn,Rd | 1 | | | | | |
| | BSET Rn,@ERd | 2 | | | 2 | | |
| | BSET Rn,@aa:8 | 2 | | | 2 | | |
| | BSET Rn,@aa:16 | 3 | | | 2 | | |
| | BSET Rn,@aa:32 | 4 | | | 2 | | |
| BSR | BSR d:8 | 2 | | 2 | | | |
| | BSR d:16 | 2 | | 2 | | | 1 |
| BST | BST #xx:3,Rd | 1 | | | | | |
| | BST #xx:3,@ERd | 2 | | | 2 | | |
| | BST #xx:3,@aa:8 | 2 | | | 2 | | |
| | BST #xx:3,@aa:16 | 3 | | | 2 | | |
| | BST #xx:3,@aa:32 | 4 | | | 2 | | |
| BTST | BTST #xx:3,Rd | 1 | | | | | |
| | BTST #xx:3,@ERd | 2 | | | 1 | | |
| | BTST #xx:3,@aa:8 | 2 | | | 1 | | |
| | BTST #xx:3,@aa:16 | 3 | | | 1 | | |
| | BTST #xx:3,@aa:32 | 4 | | | 1 | | |
| | BTST Rn,Rd | 1 | | | | | |
| | BTST Rn,@ERd | 2 | | | 1 | | |
| | BTST Rn,@aa:8 | 2 | | | 1 | | |
| | BTST Rn,@aa:16 | 3 | | | 1 | | |
| | BTST Rn,@aa:32 | 4 | | | 1 | | |
| BXOR | BXOR #xx:3,Rd | 1 | | | | | |
| | BXOR #xx:3,@ERd | 2 | | | 1 | | |
| | BXOR #xx:3,@aa:8 | 2 | | | 1 | | |
| | BXOR #xx:3,@aa:16 | 3 | | | 1 | | |
| | BXOR #xx:3,@aa:32 | 4 | | | 1 | | |
| CLRMAC | CLRMAC | Cannot be used in this LSI. | | | | | |
| CMP | CMP.B #xx:8,Rd | 1 | | | | | |
| | CMP.B Rs,Rd | 1 | | | | | |
| | CMP.W #xx:16,Rd | 2 | | | | | |
| | CMP.W Rs,Rd | 1 | | | | | |
| | CMP.L #xx:32,ERd | 3 | | | | | |
| | CMP.L ERs,ERd | 1 | | | | | |
| DAA | DAA Rd | 1 | | | | | |
| DAS | DAS Rd | 1 | | | | | |

| Instruction | Mnemonic | Instruction | Branch | Stack | Byte Data | Word Data | Internal |
|-------------|---------------------|-----------------------------|---------|-------|-------------|-----------|----------|
| | | Fetch | Address | | | | |
| | | I | J | K | L | M | N |
| DEC | DEC.B Rd | 1 | | | | | |
| | DEC.W #1/2,Rd | 1 | | | | | |
| | DEC.L #1/2 ERd | 1 | | | | | |
| DIVXS | DIVXS.B Rs,Rd | 2 | | | | | 11 |
| | DIVXS.W Rs,ERd | 2 | | | | | 19 |
| DIVXU | DIVXU.B Rs,Rd | 1 | | | | | 11 |
| | DIVXU.W Rs,ERd | 1 | | | | | 19 |
| EEPMOV | EEPMOV.B | 2 | | | $2n+2^{+2}$ | | |
| | EEPMOV.W | 2 | | | $2n+2^{+2}$ | | |
| EXTS | EXTS.W Rd | 1 | | | | | |
| | EXTS.L ERd | 1 | | | | | |
| EXTU | EXTU.W Rd | 1 | | | | | |
| | EXTU.L ERd | 1 | | | | | |
| INC | INC.B Rd | 1 | | | | | |
| | INC.W #1/2,Rd | 1 | | | | | |
| | INC.L #1/2,ERd | 1 | | | | | |
| JMP | JMP @ERN | 2 | | | | | |
| | JMP @aa:24 | 2 | | | | | 1 |
| | JMP @@aa:8 | 2 | 2 | | | | 1 |
| JSR | JSR @ERn | 2 | | 2 | | | |
| | JSR @aa:24 | 2 | | 2 | | | 1 |
| | JSR @@aa:8 | 2 | 2 | 2 | | | |
| LCD | LDC #xx:8,CCR | 1 | | | | | |
| | LDC #xx:8,EXR | 2 | | | | | |
| | LDC Rs,CCR | 1 | | | | | |
| | LDC Rs,EXR | 1 | | | | | |
| | LDC @ERs,CCR | 2 | | | | 1 | |
| | LDC @ERs,EXR | 2 | | | | 1 | |
| | LDC @(d:16,ERs),CCR | 3 | | | | 1 | |
| | LDC @(d:16,ERs),EXR | 3 | | | | 1 | |
| | LDC @(d:32,ERs),CCR | 5 | | | | 1 | |
| | LDC @(d:32,ERs),EXR | 5 | | | | 1 | |
| | LDC @ERs+,CCR | 2 | | | | 1 | 1 |
| | LDC @ERs+,EXR | 2 | | | | 1 | 1 |
| | LDC @aa:16,CCR | 3 | | | | 1 | |
| | LDC @aa:16,EXR | 3 | | | | 1 | |
| | LDC @aa:32,CCR | 4 | | | | 1 | |
| | LDC @aa:32,EXR | 4 | | | | 1 | |
| LDM | LDM.L | 2 | | 4 | | | 1 |
| | @SP+,(ERn-ERn+1) | | | | | | |
| | LDM.L | 2 | | 6 | | | 1 |
| | @SP+,(ERn-ERn+2) | | | | | | |
| | LDM.L | 2 | | 8 | | | 1 |
| | @SP+,(ERn-ERn+3) | | | | | | |
| LDMAC | LDMAC ERs,MACH | Cannot be used in this LSI. | | | | | |
| | LDMAC ERs,MACL | | | | | | |
| MAC | MAC @ERn+,@ERm+ | | | | | | |

| Instruction | Mnemonic | Instruction | Branch | Stack | Byte Data | Word Data | Internal |
|-------------|-----------------------|-----------------------------|---------|-------|-----------|-----------|----------|
| | | Fetch | Address | | | | |
| | | I | J | K | L | M | N |
| MOV | MOV.B #xx:8,Rd | 1 | | | | | |
| | MOV.B Rs,Rd | 1 | | | | | |
| | MOV.B @ERs,Rd | 1 | | | 1 | | |
| | MOV.B @(d:16,ERs),Rd | 2 | | | 1 | | |
| | MOV.B @(d:32,ERs),Rd | 4 | | | 1 | | |
| | MOV.B @ERs+,Rd | 1 | | | 1 | | 1 |
| | MOV.B @aa:8,Rd | 1 | | | 1 | | |
| | MOV.B @aa:16,Rd | 2 | | | 1 | | |
| | MOV.B @aa:32,Rd | 3 | | | 1 | | |
| | MOV.B Rs,@ERd | 1 | | | 1 | | |
| | MOV.B Rs,@(d:16,ERd) | 2 | | | 1 | | |
| | MOV.B Rs,@(d:32,ERd) | 4 | | | 1 | | |
| | MOV.B Rs,@-ERd | 1 | | | 1 | | 1 |
| | MOV.B Rs,@aa:8 | 1 | | | 1 | | |
| | MOV.B Rs,@aa:16 | 2 | | | 1 | | |
| | MOV.B Rs,@aa:32 | 3 | | | 1 | | |
| | MOV.W #xx:16,Rd | 2 | | | | | |
| | MOV.W Rs,Rd | 1 | | | | | |
| | MOV.W @ERs,Rd | 1 | | | | 1 | |
| | MOV.W @(d:16,ERs),Rd | 2 | | | | 1 | |
| | MOV.W @(d:32,ERs),Rd | 4 | | | | 1 | |
| | MOV.W @ERs+,Rd | 1 | | | | 1 | 1 |
| | MOV.W @aa:16,Rd | 2 | | | | 1 | |
| | MOV.W @aa:32,Rd | 3 | | | | 1 | |
| | MOV.W Rs,@ERd | 1 | | | | 1 | |
| | MOV.W Rs,@(d:16,ERd) | 2 | | | | 1 | |
| | MOV.W Rs,@(d:32,ERd) | 4 | | | | 1 | |
| | MOV.W Rs,@-ERd | 1 | | | | 1 | 1 |
| | MOV.W Rs,@aa:16 | 2 | | | | 1 | |
| | MOV.W Rs,@aa:32 | 3 | | | | 1 | |
| | MOV.L #xx:32,ERd | 3 | | | | | |
| | MOV.L ERs,ERd | 1 | | | | | |
| | MOV.L @ERs,ERd | 2 | | | | 2 | |
| | MOV.L @(d:16,ERs),ERd | 3 | | | | 2 | |
| | MOV.L @(d:32,ERs),ERd | 5 | | | | 2 | |
| | MOV.L @ERs+,ERd | 2 | | | | 2 | 1 |
| | MOV.L @aa:16,ERd | 3 | | | | 2 | |
| | MOV.L @aa:32,ERd | 4 | | | | 2 | |
| | MOV.L ERs,@ERd | 2 | | | | 2 | |
| | MOV.L ERs,@(d:16,ERd) | 3 | | | | 2 | |
| | MOV.L ERs,@(d:32,ERd) | 5 | | | | 2 | |
| | MOV.L ERs,@-ERd | 2 | | | | 2 | 1 |
| | MOV.L ERs,@aa:16 | 3 | | | | 2 | |
| | MOV.L ERs,@aa:32 | 4 | | | | 2 | |
| MOVFP | MOVFP @:aa:16,Rd | Cannot be used in this LSI. | | | | | |
| MOVTPE | MOVTPE Rs,@:aa:16 | | | | | | |
| MULXS | MULXS.B Rs,Rd | 2 | | | | | 11 |
| | MULXS.W Rs,ERd | 2 | | | | | 19 |
| MULXU | MULXU.B Rs,Rd | 1 | | | | | 11 |
| | MULXU.W Rs,ERd | 1 | | | | | 19 |

| Instruction | Mnemonic | Instruction | Branch | Stack | Byte Data | Word Data | Internal |
|-------------|-----------------|-------------|---------|-------------------|-----------|-----------|----------|
| | | Fetch | Address | | | | |
| | | I | J | K | L | M | N |
| NEG | NEG.B Rd | 1 | | | | | |
| | NEG.W Rd | 1 | | | | | |
| | NEG.L ERd | 1 | | | | | |
| NOP | NOP | 1 | | | | | |
| NOT | NOT.B Rd | 1 | | | | | |
| | NOT.W Rd | 1 | | | | | |
| | NOT.L ERd | 1 | | | | | |
| OR | OR.B #xx:8,Rd | 1 | | | | | |
| | OR.B Rs,Rd | 1 | | | | | |
| | OR.W #xx:16,Rd | 2 | | | | | |
| | OR.W Rs,Rd | 1 | | | | | |
| | OR.L #xx:32,ERd | 3 | | | | | |
| | OR.L ERs,ERd | 2 | | | | | |
| ORC | ORC #xx:8,CCR | 1 | | | | | |
| | ORC #xx:8,EXR | 2 | | | | | |
| POP | POP.W Rn | 1 | | | | 1 | 1 |
| | POP.L ERn | 2 | | | | 2 | 1 |
| PUSH | PUSH.W Rn | 1 | | | | 1 | 1 |
| | PUSH.L ERn | 2 | | | | 2 | 1 |
| ROTL | ROTL.B Rd | 1 | | | | | |
| | ROTL.B #2,Rd | 1 | | | | | |
| | ROTL.W Rd | 1 | | | | | |
| | ROTL.W #2,Rd | 1 | | | | | |
| | ROTL.L ERd | 1 | | | | | |
| | ROTL.L #2,ERd | 1 | | | | | |
| ROTR | ROTR.B Rd | 1 | | | | | |
| | ROTR.B #2,Rd | 1 | | | | | |
| | ROTR.W Rd | 1 | | | | | |
| | ROTR.W #2,Rd | 1 | | | | | |
| | ROTR.L ERd | 1 | | | | | |
| | ROTR.L #2,ERd | 1 | | | | | |
| ROTXL | ROTXL.B Rd | 1 | | | | | |
| | ROTXL.B #2,Rd | 1 | | | | | |
| | ROTXL.W Rd | 1 | | | | | |
| | ROTXL.W #2,Rd | 1 | | | | | |
| | ROTXL.L ERd | 1 | | | | | |
| | ROTXL.L #2,ERd | 1 | | | | | |
| ROTXR | ROTXR.B Rd | 1 | | | | | |
| | RPTXR.B #2,Rd | 1 | | | | | |
| | ROTXR.W Rd | 1 | | | | | |
| | ROTXR.W #2,Rd | 1 | | | | | |
| | ROTXR.L ERd | 1 | | | | | |
| | ROTXR.L #2,ERd | 1 | | | | | |
| RTE | RTE | 2 | | 2/3 ^{*1} | | | 1 |
| RTS | RTS | 2 | | 2 | | | 1 |

| | | Instruction Fetch | Branch Address Read | Stack Operation | Byte Data Access | Word Data Access | Internal Operation |
|-------------|----------------------------------|-----------------------------|---------------------------|--------------------|---------------------|---------------------|-----------------------|
| Instruction | Mnemonic | I | J | K | L | M | N |
| SHAL | SHAL.B Rd | 1 | | | | | |
| | SHAL.B #2,Rd | 1 | | | | | |
| | SHAL.W Rd | 1 | | | | | |
| | SHAL.W #2,Rd | 1 | | | | | |
| | SHAL.L ERd | 1 | | | | | |
| | SHAL.L #2,ERd | 1 | | | | | |
| SHAR | SHAR.B Rd | 1 | | | | | |
| | SHAR.B #2,Rd | 1 | | | | | |
| | SHAR.W Rd | 1 | | | | | |
| | SHAR.W #2,Rd | 1 | | | | | |
| | SHAR.L ERd | 1 | | | | | |
| | SHAR.L #2,ERd | 1 | | | | | |
| SHLL | SHLL.B Rd | 1 | | | | | |
| | SHLL.B #2,Rd | 1 | | | | | |
| | SHLL.W Rd | 1 | | | | | |
| | SHLL.W #2,Rd | 1 | | | | | |
| | SHLL.L ERd | 1 | | | | | |
| | SHLL.L #2,ERd | 1 | | | | | |
| SHLR | SHLR.B Rd | 1 | | | | | |
| | SHLR.B #2,Rd | 1 | | | | | |
| | SHLR.W Rd | 1 | | | | | |
| | SHLR.W #2,Rd | 1 | | | | | |
| | SHLR.L ERd | 1 | | | | | |
| | SHLR.L #2,ERd | 1 | | | | | |
| SLEEP | SLEEP | 1 | | | | | 1 |
| STC | STC.B CCR,Rd | 1 | | | | | |
| | STC.B EXR,Rd | 1 | | | | | |
| | STC.W CCR,@ERd | 2 | | | | 1 | |
| | STC.W EXR,@ERd | 2 | | | | 1 | |
| | STC.W CCR,@(d:16,ERd) | 3 | | | | 1 | |
| | STC.W EXR,@(d:16,ERd) | 3 | | | | 1 | |
| | STC.W CCR,@(d:32,ERd) | 5 | | | | 1 | |
| | STC.W EXR,@(d:32,ERd) | 5 | | | | 1 | |
| | STC.W CCR,@-ERd | 2 | | | | 1 | 1 |
| | STC.W EXR,@-ERd | 2 | | | | 1 | 1 |
| | STC.W CCR,@aa:16 | 3 | | | | 1 | |
| | STC.W EXR,@aa:16 | 3 | | | | 1 | |
| STM | STM.L (ERn-ERn+1), @-Sp | 2 | | 4 | | | 1 |
| | STM.L (ERn-ERn+2), @-Sp | 2 | | 6 | | | 1 |
| | STM.L (ERn-ERn+3), @-Sp | 2 | | 8 | | | 1 |
| | | | | | | | |
| STMAC | STMAC MACH,ERd STMAC MACL,ERd | Cannot be used in this LSI. | | | | | |
| SUB | SUB.B Rs,Rd | 1 | | | | | |
| | SUB.W #xx:16,Rd | 2 | | | | | |
| | SUB.W Rs,Rd | 1 | | | | | |
| | SUB.L #xx:32,ERd | 3 | | | | | |
| | SUB.L ERs,ERd | 1 | | | | | |
| SUBS | SUBS #1/2/4,ERd | 1 | | | | | |

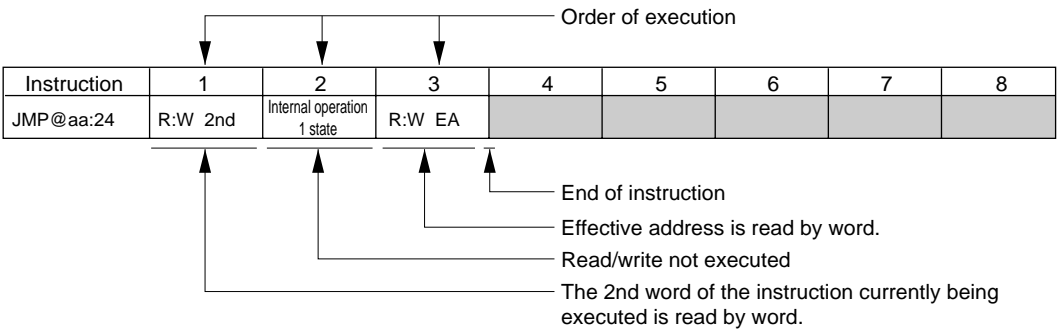
| Instruction | Mnemonic | Instruction | Branch | Stack | Byte Data | Word Data | Internal |
|-------------|------------------------|-------------|---------|-------------------|-----------|-----------|----------|
| | | Fetch | Address | | | | |
| | | I | J | K | L | M | N |
| SUBX | SUBX #xx:8,Rd | 1 | | | | | |
| | SUBX Rs,Rd | 1 | | | | | |
| TAS | TAS @ERd ^{*3} | 2 | | | 2 | | |
| TRAPA | TRAPA #x:2 | 2 | 2 | 2/3 ^{*1} | | | 2 |
| XOR | XOR.B #xx:8,Rd | 1 | | | | | |
| | XOR.B Rs,Rd | 1 | | | | | |
| | XOR.W #xx:16,Rd | 2 | | | | | |
| | XOR.W Rs,Rd | 1 | | | | | |
| | XOR.L #xx:32,ERd | 3 | | | | | |
| | XOR.L ERs,ERd | 2 | | | | | |
| XORC | XORC #xx:8,CCR | 1 | | | | | |
| | XORC #xx:8,EXR | 2 | | | | | |

- Notes:
1. 3 applies when EXR is valid, and 2 applies when invalid.
 2. Applies when the transfer data is n bytes.
 3. Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.

A.5 Bus Status During Instruction Execution

Table A.6 indicates execution status of each instruction available in this LSI. For the number of states required for each execution status, see table A.4, Number of States Required for Each Execution Status (Cycle).

[How to see the table]



[Legend]

| | |
|-------|--|
| R : B | Read by byte |
| R : W | Read by word |
| W : B | Write by byte |
| W : W | Write by word |
| : M | Bus not transferred immediately after this cycle |
| 2nd | Address of the 2nd word (3rd and 4th bytes) |
| 3rd | Address of the 3rd word (5th and 6th bytes) |
| 4th | Address of the 4th word (7th and 8th bytes) |
| 5th | Address of the 5th word (9th and 10th bytes) |
| NEXT | The head address of the instruction immediately after the instruction currently being executed |
| EA | Execution address |
| VEC | Vector address |

Table A.6 Instruction Execution Status

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---------------------|----------|----------------------------|------------|------------|---|---|---|---|---|
| ADD.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| ADD.B Rs,Rd | R:W NEXT | | | | | | | | |
| ADD.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| ADD.W Rs,Rd | R:W NEXT | | | | | | | | |
| ADD.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| ADD.L ERs,ERd | R:W NEXT | | | | | | | | |
| ADDS #1/2/4,ERd | R:W NEXT | | | | | | | | |
| ADDX #xx:8,Rd | R:W NEXT | | | | | | | | |
| ADDX Rs,Rd | R:W NEXT | | | | | | | | |
| AND.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| AND.B Rs,Rd | R:W NEXT | | | | | | | | |
| AND.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| AND.W Rs,Rd | R:W NEXT | | | | | | | | |
| AND.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| AND.L ERs,ERd | R:W 2nd | R:W NEXT | | | | | | | |
| ANDC #xx:8,CCR | R:W NEXT | | | | | | | | |
| ANDC #xx:8,EXR | R:W 2nd | R:W NEXT | | | | | | | |
| BAND #xx:3,Rd | R:W NEXT | | | | | | | | |
| BAND #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BAND #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BAND #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BAND #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | | | | | | |
| BRA d:8 (BT d:8) | R:W NEXT | R:W EA | | | | | | | |
| BRN d:8 (BT d:8) | R:W NEXT | R:W EA | | | | | | | |
| BHI d:8 | R:W NEXT | R:W EA | | | | | | | |
| BLS d:8 | R:W NEXT | R:W EA | | | | | | | |
| BCC d:8 (BHS d:8) | R:W NEXT | R:W EA | | | | | | | |
| BCS d:8 (BLO d:8) | R:W NEXT | R:W EA | | | | | | | |
| BNE d:8 | R:W NEXT | R:W EA | | | | | | | |
| BEQ d:8 | R:W NEXT | R:W EA | | | | | | | |
| BVC d:8 | R:W NEXT | R:W EA | | | | | | | |
| BVS d:8 | R:W NEXT | R:W EA | | | | | | | |
| BPL d:8 | R:W NEXT | R:W EA | | | | | | | |
| BMI d:8 | R:W NEXT | R:W EA | | | | | | | |
| BGE d:8 | R:W NEXT | R:W EA | | | | | | | |
| BLT d:8 | R:W NEXT | R:W EA | | | | | | | |
| BGT d:8 | R:W NEXT | R:W EA | | | | | | | |
| BLE d:8 | R:W NEXT | R:W EA | | | | | | | |
| BRA d:16 (BT d:16) | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BRN d:16 (BF d:16) | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BHI d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BLS d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BCC d:16 (BHS d:16) | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BCS d:16 (BLO d:16) | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BNE d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BEQ d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BVC d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BVS d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BPL d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BMI d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BGE d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|--------------------|----------|----------------------------|------------|------------|------------|--------|---|---|---|
| BLT d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BGT d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BLE d:16 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| BCLR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BCLR #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BCLR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BCLR Rn,Rd | R:W NEXT | | | | | | | | |
| BCLR Rn,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR Rn,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BCLR Rn,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BCLR Rn,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BIAND #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIAND #xx:3,ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIAND #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIAND #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BIAND #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BILD #xx:3,Rd | R:W NEXT | | | | | | | | |
| BILD #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BILD #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BILD #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BILD #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BIOR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIOR #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BOIR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BOIR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BOIR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BIST #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIST #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BIST #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BIST #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BIST #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BIXOR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BIXOR #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIXOR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BIXOR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BIXOR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BLD #xx:3,Rd | R:W NEXT | | | | | | | | |
| BLD #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BLD #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BLD #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BLD #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BNOT #xx:3,Rd | R:W NEXT | | | | | | | | |
| BNOT #xx:3,ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BNOT #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BNOT Rn,Rd | R:W NEXT | | | | | | | | |
| BNOT Rn,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BNOT Rn @aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|-------------------|-----------------------------|----------------------------|----------------|----------------|--------------|--------|---|---|---|
| BNOT Rn,@aa:16 | R:W 2nd | R:W 3rd | R:B:W EA | R:W:M NEXT | W:B EA | | | | |
| BNOT Rn,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BOR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BOR #xx:3,ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BOR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BOR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BOR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W NEXT | | | | |
| BSET #xx:3,Rd | R:W NEXT | | | | | | | | |
| BSET #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BSET #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BSET Rn,Rd | R:W NEXT | | | | | | | | |
| BSET Rn,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET Rn,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BSET Rn,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BSET Rn,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BSR d:8 | R:W NEXT | R:W EA | W:W:M stack(H) | W:W stack(L) | | | | | |
| BSR d:16 | R:W 2nd | Internal operation 1 state | R:W EA | W:W:M stack(H) | W:W stack(L) | | | | |
| BST #xx:3,Rd | R:W NEXT | | | | | | | | |
| BST #xx:3,@ERd | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BST #xx:3,@aa:8 | R:W 2nd | R:B:M EA | R:W:M NEXT | W:B EA | | | | | |
| BST #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B:M EA | R:W:M NEXT | W:B EA | | | | |
| BST #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B:M EA | R:W:M NEXT | W:B EA | | | |
| BTST #xx:3,Rd | R:W NEXT | | | | | | | | |
| BTST #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BTST #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BTST #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BTST #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BTST Rn,Rd | R:W NEXT | | | | | | | | |
| BTST Rn,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BTST Rn,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BTST Rn,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BTST Rn,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| BOXR #xx:3,Rd | R:W NEXT | | | | | | | | |
| BOXR #xx:3,@ERd | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BOXR #xx:3,@aa:8 | R:W 2nd | R:B EA | R:W:M NEXT | | | | | | |
| BOXR #xx:3,@aa:16 | R:W 2nd | R:W 3rd | R:B EA | R:W:M NEXT | | | | | |
| BOXR #xx:3,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:B EA | R:W:M NEXT | | | | |
| CLRMAC | Cannot be used in this LSI. | | | | | | | | |
| CMP.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| CMP.B Rs,Rd | R:W NEXT | | | | | | | | |
| CMP.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| CMP.W Rs,Rd | R:W NEXT | | | | | | | | |
| CMP.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| CMP.L ERs,ERd | R:W NEXT | | | | | | | | |
| DAA Rd | R:W NEXT | | | | | | | | |
| DAS Rd | R:W NEXT | | | | | | | | |
| DEC.B Rd | R:W NEXT | | | | | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|------------------------|-----------------------------|-----------------------------|-----------------------------|----------------------------|-----------------|----------|---|---|---|
| DEC.W #1/2,Rd | R:W NEXT | | | | | | | | |
| DEC.W #1/2,ERd | R:W NEXT | | | | | | | | |
| DIVXS.B Rs,Rd | R:W 2nd | R:W NEXT | Internal operation 11 state | | | | | | |
| DIVXS.W Rs,ERd | R:W 2nd | R:W NEXT | Internal operation 19 state | | | | | | |
| DIVXU.B Rs,Rd | R:W NEXT | Internal operation 11 state | | | | | | | |
| DIVXU.W Rs,ERd | R:W NEXT | Internal operation 19 state | | | | | | | |
| EEPMOV.B | R:W 2nd | R:B EAs *1 | R:B EAd *1 | R:B EAs *2 | W:B EAd *2 | R:W NEXT | | | |
| EEPMOV.W | R:W 2nd | R:B EAs *1 | R:B EAd *1 | R:B EAs *2 | W:B EAd *2 | R:W NEXT | | | |
| EXTS.W Rd | R:W NEXT | | | ← Repeat n times *2 → | | | | | |
| EXTS.L ERd | R:W NEXT | | | | | | | | |
| EXTU.W Rd | R:W NEXT | | | | | | | | |
| EXTU.L ERd | R:W NEXT | | | | | | | | |
| INC.B Rd | R:W NEXT | | | | | | | | |
| INC.W #1/2,Rd | R:W NEXT | | | | | | | | |
| INCL #1/2,ERd | R:W NEXT | | | | | | | | |
| JMP @ERn | R:W NEXT | R:W EA | | | | | | | |
| JMP @aa:24 | R:W 2nd | Internal operation 1 state | R:W EA | | | | | | |
| JMP @aa:8 | R:W NEXT | R:W:M aa:8 | R:W:M aa:8 | Internal operation 1 state | R:W EA | | | | |
| JSR @ERn | R:W NEXT | R:W EA | W:W:M stack(H) | W:W stack (L) | | | | | |
| JSR @aa:24 | R:W 2nd | Internal operation 1 state | R:W EA | W:W:M stack(H) | W:W stack (L) | | | | |
| JSR @aa:8 | R:W NEXT | R:W:M aa:8 | R:W aa:8 | W:W:M stack(H) | W:W stack (L) | R:W EA | | | |
| LCD #xx.8,CCR | R:W NEXT | | | | | | | | |
| LCD #xx.8,EXR | R:W 2nd | R:W NEXT | | | | | | | |
| LCD Rs,CCR | R:W NEXT | | | | | | | | |
| LCD Rs,EXR | R:W NEXT | | | | | | | | |
| LCD @ERs,CCR | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| LCD @ERs,EXR | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| LCD @(d:16,ERs),CCR | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LCD @(d:16,ERs),EXR | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LCD @(d:32,ERs),CCR | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | R:W EA | | | |
| LCD @(d:32,ERs),EXR | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | R:W EA | | | |
| LCD @ERs+,CCR | R:W 2nd | R:W NEXT | Internal operation 1 state | R:W EA | | | | | |
| LCD @ERs+,EXR | R:W 2nd | R:W NEXT | Internal operation 1 state | R:W EA | | | | | |
| LCD @aa:16,CCR | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LCD @aa:16,EXR | R:W 2nd | R:W 3rd | R:W NEXT | R:W EA | | | | | |
| LCD @aa:32,CCR | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:W EA | | | | |
| LCD @aa:32,EXR | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:W EA | | | | |
| LDML @SP+, (ERn-ERn+1) | R:W 2nd | R:W:M NEXT | Internal operation 1 state | R:W:M stack(H) *3 | R:W stack(L) *3 | | | | |
| LDML @SP+, (ERn-ERn+2) | R:W 2nd | R:W:M NEXT | Internal operation 1 state | R:W:M stack(H) *3 | R:W stack(L) *3 | | | | |
| LDML @SP+, (ERn-ERn+3) | R:W 2nd | R:W:M NEXT | Internal operation 1 state | R:W:M stack(H) *3 | R:W stack(L) *3 | | | | |
| LDMAC ERs,MACH | Cannot be used in this LSI. | | | | | | | | |
| LDMAC ERs,MACL | | | | | | | | | |
| MAC @ERn+,@ERm+ | | | | | | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|-----------------------|----------|----------------------------|----------------------------|----------|----------|----------|----------|---|---|
| MOV.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| MOV.B Rs,Rd | R:W NEXT | | | | | | | | |
| MOV.B @ERs,Rd | R:W NEXT | R:B EA | | | | | | | |
| MOV.B @(d:16,ERs),Rd | R:W 2nd | R:W NEXT | R:B EA | | | | | | |
| MOV.B @(d:32,ERs),Rd | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:B EA | | | | |
| MOV.B @ERs+,Rd | R:W NEXT | Internal operation 1 state | R:B EA | | | | | | |
| MOV.B @aa:8,Rd | R:W NEXT | R:B EA | | | | | | | |
| MOV.B @aa:16,Rd | R:W 2nd | R:W NEXT | R:B EA | | | | | | |
| MOV.B @aa:32,Rd | R:W 2nd | R:W 3rd | R:W NEXT | R:B EA | | | | | |
| MOV.B Rs,@ERd | R:W NEXT | W:B EA | | | | | | | |
| MOV.B Rs,@(d:16,ERd) | R:W 2nd | R:W NEXT | W:B EA | | | | | | |
| MOV.B Rs,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | W:B EA | | | | |
| MOV.B Rs,@-ERd | R:W NEXT | Internal operation 1 state | W:B EA | | | | | | |
| MOV.B Rs,@aa:8 | R:W NEXT | W:B EA | | | | | | | |
| MOV.B Rs,@aa:16 | R:W 2nd | R:W NEXT | W:B EA | | | | | | |
| MOV.B Rs,@aa:32 | R:W 2nd | R:W 3rd | R:W NEXT | W:B EA | | | | | |
| MOV.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| MOV.W Rs,Rd | R:W NEXT | | | | | | | | |
| MOV.W @ERs,Rd | R:W NEXT | R:W EA | | | | | | | |
| MOV.W @(d:16,ERs),Rd | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| MOV.W @(d:32,ERs),Rd | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | R:W EA | | | | |
| MOV.W @ERs+,Rd | R:W NEXT | Internal operation 1 state | R:W EA | | | | | | |
| MOV.W @aa:16,Rd | R:W 2nd | R:W NEXT | R:W EA | | | | | | |
| MOV.W @aa:32,Rd | R:W 2nd | R:W 3rd | R:W NEXT | R:B EA | | | | | |
| MOV.W Rs,@ERd | R:W NEXT | W:W EA | | | | | | | |
| MOV.W Rs,@(d:16,ERd) | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| MOV.W Rs,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | W:W EA | | | | |
| MOV.W Rs,@-ERd | R:W NEXT | Internal operation 1 state | W:W EA | | | | | | |
| MOV.W Rs,@aa:16 | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| MOV.W Rs,@aa:32 | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| MOV.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| MOV.L ERs,ERd | R:W NEXT | | | | | | | | |
| MOV.L @ERs,ERd | R:W 2nd | R:W:M NEXT | R:W:M EA | R:W EA+2 | | | | | |
| MOV.L @(d:16,ERs),ERd | R:W 2nd | R:W:M 3rd | R:W NEXT | R:W:M EA | R:W EA+2 | | | | |
| MOV.L @(d:32,ERs),ERd | R:W 2nd | R:W:M 3rd | R:W:M 4th | R:W 5th | R:W NEXT | R:W:M EA | R:W EA+2 | | |
| MOV.L @ERs+,ERd | R:W 2nd | R:W:M NEXT | Internal operation 1 state | R:W:M EA | R:W EA+2 | | | | |
| MOV.L @aa:16,ERd | R:W 2nd | R:W:M 3rd | R:W NEXT | R:W:M EA | R:W EA+2 | | | | |
| MOV.L @aa:32,ERd | R:W 2nd | R:W:M 3rd | R:W 4th | R:W NEXT | R:W:M EA | R:W EA+2 | | | |
| MOV.L ERs,@ERd | R:W 2nd | R:W:M NEXT | W:W:M EA | W:W EA+2 | | | | | |
| MOV.L ERs,@(d:16,ERd) | R:W 2nd | R:W:M 3rd | R:W NEXT | W:W:M EA | W:W EA+2 | | | | |
| MOV.L ERs,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | W:W:M EA | W:W EA+2 | | |
| MOV.L ERs,@-ERd | R:W 2nd | R:W:M NEXT | Internal operation 1 state | W:W:M EA | W:W EA+2 | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|-------------------|-----------------------------|-----------------------------|-----------------------------|----------|----------|----------|---|---|---|
| MOV.L ERs,@aa:16 | R:W 2nd | R:W:M 3rd | R:W NEXT | W:W:M EA | W:W EA+2 | | | | |
| MOV.L ERs,@aa:32 | R:W 2nd | R:W:M 3rd | R:W 4th | R:W NEXT | W:W:M EA | W:W EA+2 | | | |
| MOVFPPE @aa:16,Rd | Cannot be used in this LSI. | | | | | | | | |
| MOVTPE Rs,@aa:16 | | | | | | | | | |
| MULXS.B Rs,Rd | R:W 2nd | R:W NEXT | Internal operation 11 state | | | | | | |
| MULXS.W Rs,Rd | R:W 2nd | R:W NEXT | Internal operation 19 state | | | | | | |
| MULXU.B Rs,Rd | R:W NEXT | Internal operation 11 state | | | | | | | |
| MULXU.W Rs,Rd | R:W NEXT | Internal operation 19 state | | | | | | | |
| NEG.B Rd | R:W NEXT | | | | | | | | |
| NEG.W Rd | R:W NEXT | | | | | | | | |
| NEG.L ERd | R:W NEXT | | | | | | | | |
| NOP | R:W NEXT | | | | | | | | |
| NOT.B Rd | R:W NEXT | | | | | | | | |
| NOT.W Rd | R:W NEXT | | | | | | | | |
| NOT.L ERd | R:W NEXT | | | | | | | | |
| OR.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| OR.B Rs,Rd | R:W NEXT | | | | | | | | |
| OR.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| OR.W Rs,Rd | R:W NEXT | | | | | | | | |
| OR.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| OR.L ERs,ERd | R:W 2nd | R:W NEXT | | | | | | | |
| ORC #xx:8,CCR | R:W NEXT | | | | | | | | |
| ORC #xx:8,EXR | R:W 2nd | R:W NEXT | | | | | | | |
| POP.W Rn | R:W NEXT | Internal operation 1 state | R:W EA | | | | | | |
| POP.L ERn | R:W 2nd | R:W:M NEXT | Internal operation 1 state | R:W:M EA | R:W EA+2 | | | | |
| PUSH.W Rn | R:W NEXT | Internal operation 1 state | W:W EA | | | | | | |
| PUSH.L ERn | R:W 2nd | R:W:M NEXT | Internal operation 1 state | W:W:M EA | W:W EA+2 | | | | |
| ROTL.B Rd | R:W NEXT | | | | | | | | |
| ROTL.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTL.W Rd | R:W NEXT | | | | | | | | |
| ROTL.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTL.L ERd | R:W NEXT | | | | | | | | |
| ROTL.L #2, ERd | R:W NEXT | | | | | | | | |
| ROTR.B Rd | R:W NEXT | | | | | | | | |
| ROTR.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTR.W Rd | R:W NEXT | | | | | | | | |
| ROTR.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTR.L ERd | R:W NEXT | | | | | | | | |
| ROTR.L #2,ERd | R:W NEXT | | | | | | | | |
| ROTXL.B Rd | R:W NEXT | | | | | | | | |
| ROTXL.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTXL.W Rd | R:W NEXT | | | | | | | | |
| ROTXL.W #2,Rd | R:W NEXT | | | | | | | | |
| ROTXL.L ERd | R:W NEXT | | | | | | | | |
| ROTXL.L #2,ERd | R:W NEXT | | | | | | | | |
| ROTXR.B Rd | R:W NEXT | | | | | | | | |
| ROTXR.B #2,Rd | R:W NEXT | | | | | | | | |
| ROTXR.W Rd | R:W NEXT | | | | | | | | |
| ROTXR.W #2,Rd | R:W NEXT | | | | | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---------------------------|----------|--------------------------|----------------------------------|----------------------------------|----------------------------------|--------|---|---|---|
| ROTXR.L ERd | R:W NEXT | | | | | | | | |
| ROTXR.L #2.ERd | R:W NEXT | | | | | | | | |
| RTE | R:W NEXT | R:W stack(EXR) | R:W stack(H) | R:W stack(L) | Internal operation 1 state | R:W *4 | | | |
| RTS | R:W NEXT | R:W:M stack(H) | R:W stack(L) | Internal operation 1 state | R:W *4 | | | | |
| SHAL.B Rd | R:W NEXT | | | | | | | | |
| SHAL.B #2,Rd | R:W NEXT | | | | | | | | |
| SHAL.W Rd | R:W NEXT | | | | | | | | |
| SHAL.W #2,Rd | R:W NEXT | | | | | | | | |
| SHALL.ERd | R:W NEXT | | | | | | | | |
| SHALL.#2,ERd | R:W NEXT | | | | | | | | |
| SHAR.B Rd | R:W NEXT | | | | | | | | |
| SHAR.B #2,Rd | R:W NEXT | | | | | | | | |
| SHAR.W Rd | R:W NEXT | | | | | | | | |
| SHAR.W #2,Rd | R:W NEXT | | | | | | | | |
| SHAR.L ERd | R:W NEXT | | | | | | | | |
| SHAR.L #2,ERd | R:W NEXT | | | | | | | | |
| SHLL.B Rd | R:W NEXT | | | | | | | | |
| SHLL.B #2,Rd | R:W NEXT | | | | | | | | |
| SHLL.W Rd | R:W NEXT | | | | | | | | |
| SHLL.W #2,Rd | R:W NEXT | | | | | | | | |
| SHLL.L ERd | R:W NEXT | | | | | | | | |
| SHLL.L #2,ERd | R:W NEXT | | | | | | | | |
| SHLR.B Rd | R:W NEXT | | | | | | | | |
| SHLR.B #2,Rd | R:W NEXT | | | | | | | | |
| SHLR.W Rd | R:W NEXT | | | | | | | | |
| SHLR.W #2,Rd | R:W NEXT | | | | | | | | |
| SHLR.L ERd | R:W NEXT | | | | | | | | |
| SHLR.L #2,ERd | R:W NEXT | | | | | | | | |
| SLEEP | R:W NEXT | Internal operation: M | | | | | | | |
| STC CCR,Rd | R:W NEXT | | | | | | | | |
| STC EXR,Rd | R:W NEXT | | | | | | | | |
| STC CCR,@ERd | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| STC EXR,@ERd | R:W 2nd | R:W NEXT | W:W EA | | | | | | |
| STC CCR,@(d:16,ERd) | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| STC EXR,@(d:16,ERd) | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| STC CCR,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | W:W EA | | | |
| STC EXR,@(d:32,ERd) | R:W 2nd | R:W 3rd | R:W 4th | R:W 5th | R:W NEXT | W:W EA | | | |
| STC CCR,@-ERd | R:W 2nd | R:W NEXT | Internal operation 1 state | W:W EA | | | | | |
| STC EXR,@-ERd | R:W 2nd | R:W NEXT | Internal operation 1 state | W:W EA | | | | | |
| STC CCR,@aa:16 | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| STC EXR,@aa:16 | R:W 2nd | R:W 3rd | R:W NEXT | W:W EA | | | | | |
| STC CCR,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | W:W EA | | | | |
| STC EXR,@aa:32 | R:W 2nd | R:W 3rd | R:W 4th | R:W NEXT | W:W EA | | | | |
| STML (ERn- ERn+1),@-SP | R:W 2nd | R:W:M NEXT | Internal operation 1 state | W:W:M stack (H) *3 | W:W stack (L) *3 | | | | |
| STML (ERn- ERn+2),@-SP | R:W 2nd | R:W:M NEXT | Internal operation 1 state | W:W:M stack (H) *3 | W:W stack (L) *3 | | | | |
| STML (ERn- ERn+3),@-SP | R:W 2nd | R:W:M NEXT | Internal operation 1 state | W:W:M stack (H) *3 | W:W stack (L) *3 | | | | |

| Instruction | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|------------------------------|-----------------------------|----------------------------|----------------------------|--------------|----------------|-----------|-----------|----------------------------|--------|
| STMACH,ERd | Cannot be used in this LSI. | | | | | | | | |
| STMACH,ERd | | | | | | | | | |
| SUB.B Rs,Rd | R:W NEXT | | | | | | | | |
| SUB.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| SUB.W Rs,Rd | R:W NEXT | | | | | | | | |
| SUB.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| SUB.L ERs,ERd | R:W NEXT | | | | | | | | |
| SUB #1/2/4,ERd | R:W NEXT | | | | | | | | |
| SUBX #xx:8,Rd | R:W NEXT | | | | | | | | |
| SUBX Rs,Rd | R:W NEXT | | | | | | | | |
| TAS @ERd*8 | R:W 2nd | R:W NEXT | R:B:M EA | W:B EA | | | | | |
| TRAPA #x:2 | R:W NEXT | Internal operation 1 state | W:W stack(L) | W:W stack(H) | W:W stack(EXR) | R:W:M VEC | R:W VEC+2 | Internal operation 1 state | R:W *7 |
| XOR.B #xx:8,Rd | R:W NEXT | | | | | | | | |
| XOR.B Rs,Rd | R:W NEXT | | | | | | | | |
| XOR.W #xx:16,Rd | R:W 2nd | R:W NEXT | | | | | | | |
| XOR.W Rs,Rd | R:W NEXT | | | | | | | | |
| XOR.L #xx:32,ERd | R:W 2nd | R:W 3rd | R:W NEXT | | | | | | |
| XOR.L ERs,ERd | R:W 2nd | R:W NEXT | | | | | | | |
| XORC #xx:8,CCR | R:W NEXT | | | | | | | | |
| XORC #xx:8,EXR | R:W 2nd | R:W NEXT | | | | | | | |
| Reset exception handling | R:W:M VEC | R:W VEC+2 | Internal operation 1 state | R:W *5 | | | | | |
| Interrupt exception handling | R:W *6 | Internal operation 1 state | W:W stack(L) | W:W stack(H) | W:W stack(EXR) | R:W:M VEC | R:W VEC+2 | Internal operation 1 state | R:W *7 |

- Notes:
1. EAs is the contents of ER5, and EAd is the contents of ER6.
 2. 1 is added to EAs and EAd after execution. n is the initial value of R4L or R4. When 0 is set to n, R4L or R4 is not executed.
 3. Repeated twice for 2-unit retract/return, three times for 3-unit retract/return, and four times for 4-retract/return.
 4. Head address after return.
 5. Start address of the program.
 6. Pre-fetch address obtained by adding 2 to the PC to be retracted.
When returning from sleep mode, standby mode or watch mode, internal operation is executed instead of read operation.
 7. Head address of the interrupt process routine.
 8. Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.

A.6 Change of Condition Codes

This section explains change of condition codes after instruction execution of the CPU. Legend of the following tables is as follows.

| | |
|---------|---------------------------------------|
| m = 31: | Longword size |
| m = 15: | Word size |
| m = 7: | Byte size |
| Si: | Bit i of source operand |
| Di: | Bit i of destination operand |
| Ri: | Bit i of result |
| Dn: | Specified bit of destination operand |
| —: | No affection |
| ↑: | Changes depending on execution result |
| 0: | Always cleared to 0 |
| 1: | Always set to 1 |
| *: | Value undetermined |
| Z': | Z flag before execution |
| C': | C flag before execution |

Table A.7 Change of Condition Code

| Instruction | H | N | Z | V | C | Definition |
|-------------|-----------------------------|---|---|---|---|---|
| ADD | ↑ | ↑ | ↑ | ↑ | ↑ | $H = S_m \cdot 4 \cdot D_m \cdot 4 + D_m \cdot 4 \cdot \overline{R_m} \cdot 4 + S_m \cdot 4 \cdot \overline{R_m} \cdot 4$ $N = R_m$ $Z = \overline{R_m} \cdot R_m \cdot 1 \cdot \dots \dots \cdot \overline{R_0}$ $V = S_m \cdot D_m \cdot \overline{R_m} + S_m \cdot \overline{D_m} \cdot R_m$ $C = S_m \cdot D_m + D_m \cdot \overline{R_m} + S_m \cdot \overline{R_m}$ |
| ADDS | — | — | — | — | — | |
| ADDX | ↑ | ↑ | ↑ | ↑ | ↑ | $H = S_m \cdot 4 \cdot D_m \cdot 4 + D_m \cdot 4 \cdot \overline{R_m} \cdot 4 + S_m \cdot 4 \cdot \overline{R_m} \cdot 4$ $N = R_m$ $Z = Z' \cdot R_m \cdot \dots \dots \cdot \overline{R_0}$ $V = S_m \cdot D_m \cdot \overline{R_m} + S_m \cdot \overline{D_m} \cdot R_m$ $C = S_m \cdot D_m + D_m \cdot \overline{R_m} + S_m \cdot \overline{R_m}$ |
| AND | — | ↑ | ↑ | 0 | — | $N = R_m$ $Z = \overline{R_m} \cdot \overline{R_m} \cdot 1 \cdot \dots \dots \cdot \overline{R_0}$ |
| ANDC | ↑ | ↑ | ↑ | ↑ | ↑ | Value in the bit corresponding to execution result is stored. No flag change when EXR. |
| BAND | — | — | — | — | ↑ | $C = C' \cdot D_n$ |
| Bcc | — | — | — | — | — | |
| BCLR | — | — | — | — | — | |
| BIAND | — | — | — | — | ↑ | $C = C' \cdot \overline{D_n}$ |
| BILD | — | — | — | — | ↑ | $C = \overline{D_n}$ |
| BIOR | — | — | — | — | ↑ | $C = C' + \overline{D_n}$ |
| BIST | — | — | — | — | — | |
| BIXOR | — | — | — | — | ↑ | $C = C' \cdot D_n + \overline{C'} \cdot \overline{D_n}$ |
| BLD | — | — | — | — | ↑ | $C = D_n$ |
| BNOT | — | — | — | — | — | |
| BOR | — | — | — | — | ↑ | $C = C' + D_n$ |
| BSET | — | — | — | — | — | |
| BSR | — | — | — | — | — | |
| BST | — | — | — | — | — | |
| BTST | — | — | ↑ | — | — | $Z = \overline{D_n}$ |
| BXOR | — | — | — | — | ↑ | $C = C' \cdot \overline{D_n} + \overline{C'} \cdot D_n$ |
| CLRMAC | Cannot be used in this LSI. | | | | | |

| Instruction | H | N | Z | V | C | Definition |
|-------------|-----------------------------|---|---|---|---|---|
| CMP | ↑ | ↑ | ↑ | ↑ | ↑ | $H = \overline{Sm-4} \cdot \overline{Dm-4} + \overline{Dm-4} \cdot Rm-4 + Sm-4 \cdot Rm-4$ $N = Rm$ $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ $V = \overline{Sm} \cdot \overline{Dm} \cdot \overline{Rm} + Sm \cdot \overline{Dm} \cdot Rm$ $C = Sm \cdot \overline{Dm} + \overline{Dm} \cdot Rm + Sm \cdot Rm$ |
| DAA | * | ↑ | ↑ | * | ↑ | $N = Rm$ $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ C: Decimal addition carry |
| DAS | * | ↑ | ↑ | * | ↑ | $N = Rm$ $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ C: Decimal subtraction borrow |
| DEC | — | ↑ | ↑ | ↑ | — | $N = Rm$ $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ $V = Dm \cdot Rm$ |
| DIVXS | — | ↑ | ↑ | — | — | $N = Sm \cdot \overline{Dm} + \overline{Sm} \cdot Dm$ $Z = \overline{Sm} \cdot \overline{Sm-1} \cdot \dots \cdot \overline{S0}$ |
| DIVXU | — | ↑ | ↑ | — | — | $N = Sm$ $Z = \overline{Sm} \cdot \overline{Sm-1} \cdot \dots \cdot \overline{S0}$ |
| EEPMOV | — | — | — | — | — | |
| EXTS | — | ↑ | ↑ | 0 | — | $N = Rm$ $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ |
| EXTU | — | 0 | ↑ | 0 | — | $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ |
| INC | — | ↑ | ↑ | ↑ | — | $N = Rm$ $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ $V = \overline{Dm} \cdot Rm$ |
| JMP | — | — | — | — | — | |
| JSR | — | — | — | — | — | |
| LDC | ↑ | ↑ | ↑ | ↑ | ↑ | Value in the bit corresponding to execution result is stored. No flag change when EXR. |
| LDM | — | — | — | — | — | |
| LDMAC | Cannot be used in this LSI. | | | | | |
| MAC | | | | | | |
| MOV | — | ↑ | ↑ | 0 | — | $N = Rm$ $Z = \overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ |

| Instruc- tion | H | N | Z | V | C | Definition |
|------------------|-----------------------------|---|---|---|---|--|
| MOVFP | Cannot be used in this LSI. | | | | | |
| MOVTPE | | | | | | |
| MULXS | — | ↑ | ↑ | — | — | $N=R2m$ $Z=\overline{R2m} \cdot \overline{R2m-1} \cdot \dots \cdot \overline{R0}$ |
| MULXU | — | — | — | — | — | |
| NEG | ↑ | ↑ | ↑ | ↑ | ↑ | $H=Dm-4+Rm-4$ $N=Rm$ $Z=\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ $V=Dm \cdot Rm$ $C=Dm+Rm$ |
| NOP | — | — | — | — | — | |
| NOT | — | ↑ | ↑ | 0 | — | $N=Rm$ $Z=\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ |
| OR | — | ↑ | ↑ | 0 | — | $N=Rm$ $Z=\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ |
| ORC | ↑ | ↑ | ↑ | ↑ | ↑ | Value in the bit corresponding to execution result is stored. No flag change when EXR. |
| POP | — | ↑ | ↑ | 0 | — | $N=Rm$ $Z=\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ |
| PUSH | — | ↑ | ↑ | 0 | — | $N=Rm$ $Z=\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ |
| ROTL | — | ↑ | ↑ | 0 | ↑ | $N=Rm$ $Z=\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ $C=Dm$ (In case of 1 bit), $C=Dm-1$ (In case of 2 bits) |
| ROTR | — | ↑ | ↑ | 0 | ↑ | $N=Rm$ $Z=\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ $C=D0$ (In case of 1 bit), $C=D-1$ (In case of 2 bits) |
| ROTXL | — | ↑ | ↑ | 0 | ↑ | $N=Rm$ $Z=\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ $C=Dm$ (In case of 1 bit), $C=Dm-1$ (In case of 2 bits) |
| ROTXR | — | ↑ | ↑ | 0 | ↑ | $N=Rm$ $Z=\overline{Rm} \cdot \overline{Rm-1} \cdot \dots \cdot \overline{R0}$ $C=D0$ (In case of 1 bit), $C=D1$ (In case of 2 bits) |
| RTE | ↑ | ↑ | ↑ | ↑ | ↑ | Value in the bit corresponding to execution result is stored. |
| RTS | — | — | — | — | — | |

| Instruction | H | N | Z | V | C | Definition |
|-------------|-----------------------------|---|---|---|---|---|
| SHAL | — | ↑ | ↑ | ↑ | ↑ | $N=R_m$ $Z=\overline{R_m} \cdot \overline{R_m-1} \cdot \dots \cdot \overline{R_0}$ $V=\overline{D_m} \cdot \overline{D_m-1} + \overline{D_m} \cdot \overline{D_m-1}$ (In case of 1 bit) $V=\overline{D_m} \cdot \overline{D_m-1} \cdot \overline{D_m-2} \cdot \overline{D_m-1} \cdot \overline{D_m-2}$ (In case of 2 bits) $C=D_m$ (In case of 1 bit), $C=D_m-1$ (In case of 2 bits) |
| SHAR | — | ↑ | ↑ | 0 | ↑ | $N=R_m$ $Z=\overline{R_m} \cdot \overline{R_m-1} \cdot \dots \cdot \overline{R_0}$ $C=D_0$ (In case of 1 bit), $C=D_1$ (In case of 2 bits) |
| SHLL | — | ↑ | ↑ | 0 | ↑ | $N=R_m$ $Z=\overline{R_m} \cdot \overline{R_m-1} \cdot \dots \cdot \overline{R_0}$ $C=D_m$ (In case of 1 bit), $C=D_m-1$ (In case of 2 bits) |
| SHLR | — | 0 | ↑ | 0 | ↑ | $N=R_m$ $Z=\overline{R_m} \cdot \overline{R_m-1} \cdot \dots \cdot \overline{R_0}$ $C=D_0$ (In case of 1 bit), $C=D_1$ (In case of 2 bits) |
| SLEEP | — | — | — | — | — | |
| STC | — | — | — | — | — | |
| STM | — | — | — | — | — | |
| STMAC | Cannot be used in this LSI. | | | | | |
| SUB | ↑ | ↑ | ↑ | ↑ | ↑ | $H=S_m-4 \cdot \overline{D_m-4} + \overline{D_m-4} \cdot R_m-4 + S_m-4 \cdot R_m-4$ $N=R_m$ $Z=\overline{R_m} \cdot \overline{R_m-1} \cdot \dots \cdot \overline{R_0}$ $V=\overline{S_m} \cdot \overline{D_m} \cdot \overline{R_m} + S_m \cdot \overline{D_m} \cdot R_m$ $C=S_m \cdot \overline{D_m} + \overline{D_m} \cdot R_m + S_m \cdot R_m$ |
| SUBS | — | — | — | — | — | |
| SUBX | ↑ | ↑ | ↑ | ↑ | ↑ | $H=S_m-4 \cdot \overline{D_m-4} + \overline{D_m-4} \cdot R_m-4 + S_m-4 \cdot R_m-4$ $N=R_m$ $Z=\overline{Z'} \cdot \overline{R_m} \cdot \dots \cdot \overline{R_0}$ $V=\overline{S_m} \cdot \overline{D_m} \cdot \overline{R_m} + S_m \cdot \overline{D_m} \cdot R_m$ $C=S_m \cdot \overline{D_m} + \overline{D_m} \cdot R_m + S_m \cdot R_m$ |
| TAS | — | ↑ | ↑ | 0 | — | $N=D_m$ $Z=\overline{D_m} \cdot \overline{D_m-1} \cdot \dots \cdot \overline{D_0}$ |
| TRAPA | — | — | — | — | — | |
| XOR | — | ↑ | ↑ | 0 | — | $N=R_m$ $Z=\overline{R_m} \cdot \overline{R_m-1} \cdot \dots \cdot \overline{R_0}$ |
| XORC | ↑ | ↑ | ↑ | ↑ | ↑ | Value in the bit corresponding to execution result is stored. No flag change when EXR. |

Appendix B Internal I/O Registers

B.1 Addresses

| Address [®] | Register Name | R/W | Access | Bus Width | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Module Name |
|----------------------|---------------|-----|--------|-----------|--------|--------|--------|--------|--------|--------|-------|-------|------------------------|
| HD000 | DGKp | W | 16 | 16 | DGKp15 | DGKp14 | DGKp13 | DGKp12 | DGKp11 | DGKp10 | DGKp9 | DGKp8 | Drum digital filter |
| HD001 | | | | | DGKp7 | DGKp6 | DGKp5 | DGKp4 | DGKp3 | DGKp2 | DGKp1 | DGKp0 | |
| HD002 | DGKs | W | 16 | 16 | DGKs15 | DGKs14 | DGKs13 | DGKs12 | DGKs11 | DGKs10 | DGKs9 | DGKs8 | |
| HD003 | | | | | DGKs7 | DGKs6 | DGKs5 | DGKs4 | DGKs3 | DGKs2 | DGKs1 | DGKs0 | |
| HD004 | DAP | W | 16 | 16 | DAP15 | DAP14 | DAP13 | DAP12 | DAP11 | DAP10 | DAP9 | DAP8 | |
| HD005 | | | | | DAP7 | DAP6 | DAP5 | DAP4 | DAP3 | DAP2 | DAP1 | DAP0 | |
| HD006 | DBp | W | 16 | 16 | DBp15 | DBp14 | DBp13 | DBp12 | DBp11 | DBp10 | DBp9 | DBp8 | |
| HD007 | | | | | DBp7 | DBp6 | DBp5 | DBp4 | DBp3 | DBp2 | DBp1 | DBp0 | |
| HD008 | DAs | W | 16 | 16 | DAs15 | DAs14 | DAs13 | DAs12 | DAs11 | DAs10 | DAs9 | DAs8 | |
| HD009 | | | | | DAs7 | DAs6 | DAs5 | DAs4 | DAs3 | DAs2 | DAs1 | DAs0 | |
| HD00A | DBs | W | 16 | 16 | DBs15 | DBs14 | DBs13 | DBs12 | DBs11 | DBs10 | DBs9 | DBs8 | |
| HD00B | | | | | DBs7 | DBs6 | DBs5 | DBs4 | DBs3 | DBs2 | DBs1 | DBs0 | |
| HD00C | DOfp | W | 16 | 16 | DOfp15 | DOfp14 | DOfp13 | DOfp12 | DOfp11 | DOfp10 | DOfp9 | DOfp8 | |
| HD00D | | | | | DOfp7 | DOfp6 | DOfp5 | DOfp4 | DOfp3 | DOfp2 | DOfp1 | DOfp0 | |
| HD00E | DOfs | W | 16 | 16 | DOfs15 | DOfs14 | DOfs13 | DOfs12 | DOfs11 | DOfs10 | DOfs9 | DOfs8 | |
| HD00F | | | | | DOfs7 | DOfs6 | DOfs5 | DOfs4 | DOfs3 | DOfs2 | DOfs1 | DOfs0 | |
| HD010 | CGKp | W | 16 | 16 | CGKp15 | CGKp14 | CGKp13 | CGKp12 | CGKp11 | CGKp10 | CGKp9 | CGKp8 | Capstan digital filter |
| HD011 | | | | | CGKp7 | CGKp6 | CGKp5 | CGKp4 | CGKp3 | CGKp2 | CGKp1 | CGKp0 | |
| HD012 | CGKs | W | 16 | 16 | CGKs15 | CGKs14 | CGKs13 | CGKs12 | CGKs11 | CGKs10 | CGKs9 | CGKs8 | |
| HD013 | | | | | CGKs7 | CGKs6 | CGKs5 | CGKs4 | CGKs3 | CGKs2 | CGKs1 | CGKs0 | |
| HD014 | CAP | W | 16 | 16 | CAP15 | CAP14 | CAP13 | CAP12 | CAP11 | CAP10 | CAP9 | CAP8 | |
| HD015 | | | | | CAP7 | CAP6 | CAP5 | CAP4 | CAP3 | CAP2 | CAP1 | CAP0 | |
| HD016 | CBp | W | 16 | 16 | CBp15 | CBp14 | CBp13 | CBp12 | CBp11 | CBp10 | CBp9 | CBp8 | |
| HD017 | | | | | CBp7 | CBp6 | CBp5 | CBp4 | CBp3 | CBp2 | CBp1 | CBp0 | |
| HD018 | CAs | W | 16 | 16 | CAs15 | CAs14 | CAs13 | CAs12 | CAs11 | CAs10 | CAs9 | CAs8 | |
| HD019 | | | | | CAs7 | CAs6 | CAs5 | CAs4 | CAs3 | CAs2 | CAs1 | CAs0 | |
| HD01A | CBs | W | 16 | 16 | CBs15 | CBs14 | CBs13 | CBs12 | CBs11 | CBs10 | CBs9 | CBs8 | |
| HD01B | | | | | CBs7 | CBs6 | CBs5 | CBs4 | CBs3 | CBs2 | CBs1 | CBs0 | |
| HD01C | COfp | W | 16 | 16 | COfp15 | COfp14 | COfp13 | COfp12 | COfp11 | COfp10 | COfp9 | COfp8 | |
| HD01D | | | | | COfp7 | COfp6 | COfp5 | COfp4 | COfp3 | COfp2 | COfp1 | COfp0 | |
| HD01E | COfs | W | 16 | 16 | COfs15 | COfs14 | COfs13 | COfs12 | COfs11 | COfs10 | COfs9 | COfs8 | |
| HD01F | | | | | COfs7 | COfs6 | COfs5 | COfs4 | COfs3 | COfs2 | COfs1 | COfs0 | |
| HD020 | DZs | W | 16 | 16 | – | – | – | – | DZs11 | DZs10 | DZs9 | DZs8 | Digital filter |
| HD021 | | | | | DZs7 | DZs6 | DZs5 | DZs4 | DZs3 | DZs2 | DZs1 | DZs0 | |
| HD022 | DZp | W | 16 | 16 | – | – | – | – | DZp11 | DZp10 | DZp9 | DZp8 | |
| HD023 | | | | | DZp7 | DZp6 | DZp5 | DZp4 | DZp3 | DZp2 | DZp1 | DZp0 | |
| HD024 | CZs | W | 16 | 16 | – | – | – | – | CZs11 | CZs10 | CZs9 | CZs8 | |
| HD025 | | | | | CZs7 | CZs6 | CZs5 | CZs4 | CZs3 | CZs2 | CZs1 | CZs0 | |
| HD026 | CZp | W | 16 | 16 | – | – | – | – | CZp11 | CZp10 | CZp9 | CZp8 | |
| HD027 | | | | | CZp7 | CZp6 | CZp5 | CZp4 | CZp3 | CZp2 | CZp1 | CZp0 | |
| HD028 | DFIC | R/W | 8 | 16 | – | DROV | DPHA | DZPON | DZSON | DSG2 | DSG1 | DSG0 | |
| HD029 | CFIC | R/W | 8 | | – | CROV | CPHA | CZPON | CZSON | CSG2 | CSG1 | CSG0 | |
| HD02A | DFUCR | R/W | 8 | 16 | – | – | PTON | CP/DP | CFEPS | DFEPS | CFESS | DFESS | |

| Address* | Register Name | R/W | Access | Bus Width | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Module Name |
|----------|---------------|-----|--------|-----------|----------|----------|----------|-----------|----------|----------|---------|---------|---|
| HD030 | DFPR | W | 16 | 16 | DFPR15 | DFPR14 | DFPR13 | DFPR12 | DFPR11 | DFPR10 | DFPR9 | DFPR8 | Drum error detector |
| HD031 | | | | | DFPR7 | DFPR6 | DFPR5 | DFPR4 | DFPR3 | DFPR2 | DFPR1 | DFPR0 | |
| HD032 | DFER | R/W | 16 | 16 | DFER15 | DFER14 | DFER13 | DFER12 | DFER11 | DFER10 | DFER9 | DFER8 | |
| HD033 | | | | | DFER7 | DFER6 | DFER5 | DFER4 | DFER3 | DFER2 | DFER1 | DFER0 | |
| HD034 | DFRUDR | W | 16 | 16 | DFRUDR15 | DFRUDR14 | DFRUDR13 | DFRUDR12 | DFRUDR11 | DFRUDR10 | DFRUDR9 | DFRUDR8 | |
| HD035 | | | | | DFRUDR7 | DFRUDR6 | DFRUDR5 | DFRUDR4 | DFRUDR3 | DFRUDR2 | DFRUDR1 | DFRUDR0 | |
| HD036 | DFRLDR | W | 16 | 16 | DFRLDR15 | DFRLDR14 | DFRLDR13 | DFRLDR12 | DFRLDR11 | DFRLDR10 | DFRLDR9 | DFRLDR8 | |
| HD037 | | | | | DFRLDR7 | DFRLDR6 | DFRLDR5 | DFRLDR4 | DFRLDR3 | DFRLDR2 | DFRLDR1 | DFRLDR0 | |
| HD038 | DFVCR | R/W | 8 | 16 | DFCS1 | DFCS0 | DFOVF | DFRFON | DF-R/UNR | OPCNT | DFRCS1 | DFRCS0 | |
| HD039 | DPGCR | R/W | 8 | 16 | DPCS1 | DPCS0 | DPOVF | N/V | HSWES | – | – | – | |
| HD03A | DPPR2 | W | 16 | 16 | DPPR15 | DPPR14 | DPPR13 | DPPR12 | DPPR11 | DPPR10 | DPPR9 | DPPR8 | |
| HD03B | | | | | DPPR7 | DPPR6 | DPPR5 | DPPR4 | DPPR3 | DPPR2 | DPPR1 | DPPR0 | |
| HD03C | DPPR1 | W | 8 | 16 | – | – | – | – | DPPR19 | DPPR18 | DPPR17 | DPPR16 | |
| HD03D | DPER1 | W | 8 | 16 | – | – | – | – | DPER19 | DPER18 | DPER17 | DPER16 | |
| HD03E | DPER2 | W | 16 | 16 | DPER15 | DPER14 | DPER13 | DPER12 | DPER11 | DPER10 | DPER9 | DPER8 | |
| HD03F | | | | | DPER7 | DPER6 | DPER5 | DPER4 | DPER3 | DPER2 | DPER1 | DPER0 | |
| HD050 | CFPR | W | 16 | 16 | CFPR15 | CFPR14 | CFPR13 | CFPR12 | CFPR11 | CFPR10 | CFPR9 | CFPR8 | Capstan error detector |
| HD051 | | | | | CFPR7 | CFPR6 | CFPR5 | CFPR4 | CFPR3 | CFPR2 | CFPR1 | CFPR0 | |
| HD052 | CFER | R/W | 16 | 16 | CFER15 | CFER14 | CFER13 | CFER12 | CFER11 | CFER10 | CFER9 | CFER8 | |
| HD053 | | | | | CFER7 | CFER6 | CFER5 | CFER4 | CFER3 | CFER2 | CFER1 | CFER0 | |
| HD054 | CFRUDR | W | 16 | 16 | CFRUDR15 | CFRUDR14 | CFRUDR13 | CFRUDR12 | CFRUDR11 | CFRUDR10 | CFRUDR9 | CFRUDR8 | |
| HD055 | | | | | CFRUDR7 | CFRUDR6 | CFRUDR5 | CFRUDR4 | CFRUDR3 | CFRUDR2 | CFRUDR1 | CFRUDR0 | |
| HD056 | CFRLDR | W | 16 | 16 | CFRLDR15 | CFRLDR14 | CFRLDR13 | CFRLDR12 | CFRLDR11 | CFRLDR10 | CFRLDR9 | CFRLDR8 | |
| HD057 | | | | | CFRLDR7 | CFRLDR6 | CFRLDR5 | CFRLDR4 | CFRLDR3 | CFRLDR2 | CFRLDR1 | CFRLDR0 | |
| HD058 | CFVCR | R/W | 8 | 16 | CFCS1 | CFCS0 | CFOVF | CFRFON | CF-R/UNR | CPCNT | CFRCS1 | CFRCS0 | |
| HD059 | CPGCR | R/W | 8 | 16 | CPCS1 | CPCS0 | CPOVF | CR/RF | SELCFG2 | – | – | – | |
| HD05A | CPPR2 | W | 16 | 16 | CPH15 | CPH14 | CPH13 | CPH12 | CPH11 | CPH10 | CPH9 | CPH8 | |
| HD05B | | | | | CPH7 | CPH6 | CPH5 | CPH4 | CPH3 | CPH2 | CPH1 | CPH0 | |
| HD05C | CPPR1 | W | 8 | 16 | – | – | – | – | CPH19 | CPH18 | CPH17 | CPH16 | |
| HD05D | CPER1 | W | 8 | 16 | – | – | – | – | CPER19 | CPER18 | CPER17 | CPER16 | |
| HD05E | CPER2 | W | 16 | 16 | CPER15 | CPER14 | CPER13 | CPER12 | CPER11 | CPER10 | CPER9 | CPER8 | |
| HD05F | | | | | CPER7 | CPER6 | CPER5 | CPER4 | CPER3 | CPER2 | CPER1 | CPER0 | |
| HD060 | HSM1 | R/W | 8 | 16 | FLB | FLA | EMPB | EMPA | OVWB | OVWA | CLRB | CLRA | HSW timing generator * Assign to the same address. |
| HD061 | HSM2 | R/W | 8 | | FRT | FGR2OFF | LOP | EDG | ISEL | SOFG | OFG | VFF/NFF | |
| HD062 | HSLP | W | 8 | 16 | LOB3 | LOB2 | LOB1 | LOB0 | LOA3 | LOA2 | LOA1 | LOA0 | |
| HD064 | FPDRA | W | 16 | 16 | – | ADTRGA | STRIGA | NarrowFFA | VFFA | AFFA | VpulseA | MlevelA | |
| HD065 | | | | | PPGA7 | PPGA6 | PPGA5 | PPGA4 | PPGA3 | PPGA2 | PPGA1 | PPGA0 | |
| HD066 | FTPRA* | W | 16 | 16 | FTPRA15 | FTRPA14 | FTRPA13 | FTRPA12 | FTRPA11 | FTRPA10 | FTRPA9 | FTRPA8 | |
| HD066 | FTCTR* | R | 16 | | FTCTR15 | FTCTR14 | FTCTR13 | FTCTR12 | FTCTR11 | FTCTR10 | FTCTR9 | FTCTR8 | |
| HD067 | FTPRA* | W | 16 | 16 | FTPRA7 | FTPRA6 | FTPRA5 | FTPRA4 | FTPRA3 | FTPRA2 | FTPRA1 | FTPRA0 | |
| HD067 | FTCTR* | R | 16 | | FTCTR7 | FTCTR6 | FTCTR5 | FTCTR4 | FTCTR3 | FTCTR2 | FTCTR1 | FTCTR0 | |
| HD068 | FPDRB | W | 16 | 16 | – | ADTRGB | STRIGB | NarrowFFB | VFFB | AFFB | VpulseB | MlevelB | |
| HD069 | | | | | PPGB7 | PPGB6 | PPGB5 | PPGB4 | PPGB3 | PPGB2 | PPGB1 | PPGB0 | |
| HD06A | FTPRB | W | 16 | 16 | FTPRB15 | FTPRB14 | FTPRB13 | FTPRB12 | FTPRB11 | FTPRB10 | FTPRB9 | FTPRB8 | |
| HD06B | | | | | FTPRB7 | FTPRB6 | FTPRB5 | FTPRB4 | FTPRB3 | FTPRB2 | FTPRB1 | FTPRB0 | |
| HD06C | DFCTR* | W | 8 | 16 | ISEL2 | CCLR | CKSL | DFCRA4 | DFCRA3 | DFCRA2 | DFCRA1 | DFCRA0 | |
| HD06C | DFCRB | R | 8 | | – | – | – | DFCTR4 | DFCTR3 | DFCTR2 | DFCTR1 | DFCTR0 | |
| HD06D | DFCRB | W | 8 | 16 | – | – | – | DFCRB4 | DFCRB3 | DFCRB2 | DFCRB1 | DFCRB0 | |
| HD06E | CHCR | W | 8 | 16 | V/N | HSWPOL | CRH | HAH | SIG3 | SIG2 | SIG1 | SIG0 | 4-head special-effects playback |
| HD06F | ADDVR | R/W | 8 | | – | – | – | HMSK | HIZ | CUT | VPON | POL | Additional V |

| Address* | Register Name | R/W | Access | Bus Width | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Module Name |
|----------|---------------|-----|--------|-----------|---------|--------|---------|----------|---------|---------|----------|--------|--|
| HD070 | XDR | W | 16 | 16 | – | – | – | – | XR11 | XR10 | XR9 | XR8 | X-value, TRK-value |
| HD071 | | | | | XR7 | XR6 | XR5 | XR4 | XR3 | XR2 | XR1 | XR0 | |
| HD072 | TRDR | W | 16 | 16 | – | – | – | – | TRD11 | TRD10 | TRD9 | TRD8 | |
| HD073 | | | | | TRD7 | TRD6 | TRD5 | TRD4 | TRD3 | TRD2 | TRD1 | TRD0 | |
| HD074 | XTCR | R/W | 8 | 16 | – | CAPRF | AT/MÜ | TRK/X | EXC/REF | XCS | DVREF1 | DVREF0 | Drum 12-bit PWM |
| HD078 | DPWDR | R/W | 16 | 16 | – | – | – | – | DPWDR11 | DPWDR10 | DPWDR9 | DPWDR8 | |
| HD079 | | | | | DPWDR7 | DPWDR6 | DPWDR5 | DPWDR4 | DPWDR3 | DPWDR2 | DPWDR1 | DPWDR0 | |
| HD07A | DPWCR | W | 8 | 16 | DPOL | DDC | DHIZ | DH/L | DSFDF | DCCK2 | DCCK1 | DCCK0 | |
| HD07B | CPWCR | W | 8 | | CPOL | CDC | CHIZ | CH/L | CSF/DF | CCK2 | CCK1 | CCK0 | Capstan 12-bit PWM |
| HD07C | CPWDR | R/W | 16 | 16 | – | – | – | – | CPWDR11 | CPWDR10 | CPWDR9 | CPWDR8 | |
| HD07D | | | | | CPWDR7 | CPWDR6 | CPWDR5 | CPWDR4 | CPWDR3 | CPWDR2 | CPWDR1 | CPWDR0 | |
| HD080 | CTCR | W | 8 | 16 | NT/PAL | FLSC | FLSB | FSLA | CCS | LCTL | UNCTL | SLWM | |
| HD081 | CTLM | R/W | 8 | | ASM | REC/PB | FW/RV | MD4 | MD3 | MD3 | MD1 | MD0 | CTL circuit |
| HD082 | RCDR1 | W | 16 | 16 | – | – | – | – | CMT1B | CMT1A | CMT19 | CMT18 | |
| HD083 | | | | | CMT17 | CMT16 | CMT15 | CMT14 | CMT13 | CMT12 | CMT11 | CMT10 | |
| HD084 | RCDR2 | W | 16 | 16 | – | – | – | – | CMT2B | CMT2A | CMT29 | CMT28 | |
| HD085 | | | | | CMT27 | CMT26 | CMT25 | CMT24 | CMT23 | CMT22 | CMT21 | CMT20 | |
| HD086 | RCDR3 | W | 16 | 16 | – | – | – | – | CMT3B | CMT3A | CMT39 | CMT38 | |
| HD087 | | | | | CMT37 | CMT36 | CMT35 | CMT34 | CMT33 | CMT32 | CMT31 | CMT30 | |
| HD088 | RCDR4 | W | 16 | 16 | – | – | – | – | CMT4B | CMT4A | CMT49 | CMT48 | |
| HD089 | | | | | CMT47 | CMT46 | CMT45 | CMT44 | CMT43 | CMT42 | CMT41 | CMT40 | |
| HD08A | RCDR5 | W | 16 | 16 | – | – | – | – | CMT5B | CMT5A | CMT59 | CMT58 | |
| HD08B | | | | | CMT57 | CMT56 | CMT55 | CMT54 | CMT53 | CMT52 | CMT51 | CMT50 | |
| HD08C | DI/O | R/W | 8 | 16 | VCTR2 | VCTR1 | VCTR0 | – | BPON | BPS | BPF | DI/O | |
| HD08D | BTPR | R/W | 8 | | LSP7 | LSP6 | LSP5 | LSP4 | LSP3 | LSP2 | LSP1 | LSP0 | Reference signal generator |
| HD090 | RFD | W | 16 | 16 | REF15 | REF14 | REF13 | REF12 | REF11 | REF10 | REF9 | REF8 | |
| HD091 | | | | | REF7 | REF6 | REF5 | REF4 | REF3 | REF2 | REF1 | REF0 | |
| HD092 | CRF | W | 16 | 16 | CRF15 | CRF14 | CRF13 | CRF12 | CRF11 | CRF10 | CRF9 | CRF8 | |
| HD093 | | | | | CRF7 | CRF6 | CRF5 | CRF4 | CRF3 | CRF2 | CRF1 | CRF0 | * The TBC bit is available only in the H8S/2194C series. |
| HD094 | RFC | R/W | 16 | 16 | RFC15 | RFC14 | RFC13 | RFC12 | RFC11 | RFC10 | RFC9 | RFC8 | |
| HD095 | | | | | RFC7 | RFC6 | RFC5 | RFC4 | RFC3 | RFC2 | RFC1 | RFC0 | |
| HD096 | RFM | R/W | 8 | 16 | RCF | VNA | CVS | REX | CRD | OD/EV | VST | VEG | |
| HD097 | RFM2 | R/W | 8 | | (TBC)* | – | – | – | – | – | – | FDS | Frequency divider |
| HD098 | CTVC | R/W | 8 | 16 | CEX | CEG | – | – | – | CFG | HSW | CTL | |
| HD099 | CTLR | W | 8 | | CTL7 | CTL6 | CTL5 | CTL4 | CTL3 | CTL2 | CTL1 | CTL0 | |
| HD09A | CDVC | R/W | 8 | 16 | MCGain | | CMK | CMN | DVTRG | CRF | CPS1 | CPS0 | |
| HD09B | CDIVR1 | W | 8 | | – | CDV16 | CDV15 | CDV14 | CDV13 | CDV12 | CDV11 | CDV10 | |
| HD09C | CDIVR2 | W | 8 | 16 | – | CDV26 | CDV25 | CDV24 | CDV23 | CDV22 | CDV21 | CDV20 | |
| HD09D | CTMR | W | 8 | | – | – | CPM5 | CPM4 | CPM3 | CPM2 | CPM1 | CPM0 | |
| HD09E | FGCR | W | 8 | 16 | – | – | – | – | – | – | – | DRF | |
| HD0A0 | SPMR | R/W | 8 | 8 | CTLSTOP | – | CFGCOMP | EXCELRON | DPGSW | COMP | H.Amp.SW | C.Rot | Servo port control |
| HD0A1 | SPCR | R/W | 8 | 8 | – | – | – | – | SPCR4 | SPCR3 | SPCR2 | SPCR1 | |
| HD0A2 | SPDR | R/W | 8 | 8 | – | – | – | – | SPDR4 | SPDR3 | SPDR2 | SPDR1 | |
| HD0A3 | SVMCR | R/W | 8 | 8 | – | – | SVMCR5 | SVMCR4 | SVMCR3 | SVMCR2 | SVMCR1 | SVMCR0 | |
| HD0A4 | CTLGR | R/W | 8 | 8 | – | – | – | – | CTLFB | CTLGR3 | CTLGR2 | CTLGR1 | CTLGR0 |

| Address* | Register Name | R/W | Access | Bus Width | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Module Name |
|----------|---------------------|-----|--------|-----------|--------|--------|--------|---------|---------|---------|---------|---------|-------------------------|
| H'D0B0 | VTR | W | 8 | 16 | – | – | VTR5 | VTR4 | VTR3 | VTR2 | VTR1 | VTR0 | Sync detector |
| H'D0B1 | HTR | W | 8 | 16 | – | – | – | – | HTR3 | HTR2 | HTR1 | HTR0 | |
| H'D0B2 | HRTR | W | 8 | 16 | HRTR7 | HRTR6 | HRTR5 | HRTR4 | HRTR3 | HRTR2 | HRTR1 | HRTR0 | |
| H'D0B3 | HPWR | W | 8 | 16 | – | – | – | – | HPWR3 | HPWR2 | HPWR1 | HPWR0 | |
| H'D0B4 | NWR | W | 8 | 16 | – | – | NWR5 | NWR4 | NWR3 | NWR2 | NWR1 | NWR0 | |
| H'D0B5 | NDR | W | 8 | 16 | NDR7 | NDR6 | NDR5 | NDR4 | NDR3 | NDR2 | NDR1 | NDR0 | |
| H'D0B6 | SYNCR | R/W | 8 | 16 | – | – | – | – | NID/VD | NOIS | FLD | SYCT | |
| H'D0B8 | SIENR1 | R/W | 8 | 16 | IEDR3 | IEDR2 | IEDR1 | IECAP3 | IECAP2 | IECAP1 | IEHSW2 | IEHSW1 | |
| H'D0B9 | SIENR2 | R/W | 8 | 16 | – | – | – | – | – | – | IESNC | IESTL | Servo interrupt control |
| H'D0BA | SIRQR1 | R/W | 8 | 16 | IRDRM3 | IRDRM2 | IRDRM1 | IRRCAP3 | IRRCAP2 | IRRCAP1 | IRRHWS2 | IRRHWS1 | |
| H'D0BB | SIRQR2 | R/W | 8 | 16 | – | – | – | – | – | – | IRRSNC | IRRCTL | |
| H'D0C0 | 32 byte Data Buffer | R/W | 8 | 8 | | | | | | | | | 32-byte buffer SCI2 |
| H'D0C1 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0C2 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0C3 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0C4 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0C5 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0C6 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0C7 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0C8 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0C9 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0CA | | R/W | 8 | 8 | | | | | | | | | |
| H'D0CB | | R/W | 8 | 8 | | | | | | | | | |
| H'D0CC | | R/W | 8 | 8 | | | | | | | | | |
| H'D0CD | | R/W | 8 | 8 | | | | | | | | | |
| H'D0CE | | R/W | 8 | 8 | | | | | | | | | |
| H'D0CF | | R/W | 8 | 8 | | | | | | | | | |
| H'D0D0 | 32 byte Data Buffer | R/W | 8 | 8 | | | | | | | | | 32-byte buffer SCI2 |
| H'D0D1 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0D2 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0D3 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0D4 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0D5 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0D6 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0D7 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0D8 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0D9 | | R/W | 8 | 8 | | | | | | | | | |
| H'D0DA | | R/W | 8 | 8 | | | | | | | | | |
| H'D0DB | | R/W | 8 | 8 | | | | | | | | | |
| H'D0DC | | R/W | 8 | 8 | | | | | | | | | |
| H'D0DD | | R/W | 8 | 8 | | | | | | | | | |
| H'D0DE | | R/W | 8 | 8 | | | | | | | | | |
| H'D0DF | | R/W | 8 | 8 | | | | | | | | | |
| H'D0E0 | STAR | R/W | 8 | 8 | – | – | – | STA4 | STA3 | STA2 | STA1 | STA0 | 32-byte buffer SCI2 |
| H'D0E1 | EDAR | R/W | 8 | 8 | – | – | – | EDA4 | EDA3 | EDA2 | EDA1 | EDA0 | |
| H'D0E2 | SCR2 | R/W | 8 | 8 | TEIE | ABTIE | – | GAP1 | GAP0 | CKS2 | CKS1 | CKS0 | |
| H'D0E3 | SCSR2 | R/W | 8 | 8 | TEI | – | – | SOL | ORER | WT | ABT | STF | |

| Address* | Register Name | R/W | Access | Bus Width | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Module Name |
|----------|---------------|-----|--------|-----------|--------|--------|--------|----------|--------|--------|---------|--------|---|
| HD100 | TIER | R/W | 8 | 16 | ICIAE | ICIBE | ICICE | ICIDE | OCIAE | OCIBE | OVIE | ICSA | Timer X1 * OCRA and OCRB addresses are the same. Switched by OCSR bit in IOCR. |
| HD101 | TCSRX | R/W | 8 | 16 | ICFA | ICFB | ICFC | ICFD | OCFA | OCFB | OVF | CCLRA | |
| HD102 | FRCH | R/W | 8/16 | 16 | FRCH7 | FRCH6 | FRCH5 | FRCH4 | FRCH3 | FRCH2 | FRCH1 | FRCH0 | |
| HD103 | FRCL | | | | FRCL7 | FRCL6 | FRCL5 | FRCL4 | FRCL3 | FRCL2 | FRCL1 | FRCL0 | |
| HD104 | OXRAH* | R/W | 8/16 | 16 | OCRAH7 | OCRAH6 | OCRAH5 | OCRAH4 | OCRAH3 | OCRAH2 | OCRAH1 | OCRAH0 | |
| HD105 | OCRAL* | | | | OCRAL7 | OCRAL6 | OCRAL5 | OCRAL4 | OCRAL3 | OCRAL2 | OCRAL1 | OCRAL0 | |
| HD104 | OCRBH* | R/W | 8/16 | 16 | OCRBH7 | OCRBH6 | OCRBH5 | OCRBH4 | OCRBH3 | OCRBH2 | OCRBH1 | OCRBH0 | |
| HD105 | OCRBL* | | | | OCRBL7 | OCRBL6 | OCRBL5 | OCRBL4 | CORBL3 | CORBL2 | CORBL1 | CORBL0 | |
| HD106 | TCRX | R/W | 8 | 16 | IEDGA | IEDGB | IEDGC | IEDGD | BUFEA | FUFEB | CKS1 | CKS0 | |
| HD107 | TOCR | R/W | 8 | 16 | ICSB | ICSC | ICSD | OCRS | OEA | OEB | OLVLA | OLVLB | |
| HD108 | ICRAH | R | 8/16 | 16 | ICRAH7 | ICRAH6 | ICRAH5 | ICRAH4 | ICRAH3 | ICRAH2 | ICRAH1 | ICRAH0 | Timer B |
| HD109 | ICRAL | | | | ICRAL7 | ICRAL6 | ICRAL5 | ICRAL4 | ICRAL3 | ICRAL2 | ICRAL1 | ICRAL0 | |
| HD10A | ICRBH | R | 8/16 | 16 | ICRBH7 | ICRBH6 | ICRBH5 | ICRBH4 | ICRBH3 | ICRBH2 | ICRBH1 | ICRBH0 | |
| HD10B | ICRBL | | | | ICRBL7 | ICRBL6 | ICRBL5 | ICRBL4 | ICRBL3 | ICRBL2 | ICRBL1 | ICRBL0 | |
| HD10C | ICRCH | R | 8/16 | 16 | ICRCH7 | ICRCH6 | ICRCH5 | ICRCH4 | ICRCH3 | ICRCH2 | ICRCH1 | ICRCH0 | |
| HD10D | ICRCL | | | | ICRCL7 | ICRCL6 | ICRCL5 | ICRCL4 | ICRCL3 | ICRCL2 | ICRCL1 | ICRCL0 | |
| HD10E | ICRDH | R | 8/16 | 16 | ICRDH7 | ICRDH6 | ICRDH5 | ICRDH4 | ICRDH3 | ICRDH2 | ICRDH1 | ICRDH0 | |
| HD10F | ICRDL | | | | ICRDL7 | ICRDL6 | ICRDL5 | ICRDL4 | ICRDL3 | ICRDL2 | ICRDL1 | ICRDL0 | |
| HD110 | TMB | R/W | 8 | 8 | TMB17 | TMB1F | TMPIE | – | – | TMP12 | TMP11 | TCB10 | |
| HD111 | TCB | R | 8 | 8 | TCB17 | TCB16 | TCB15 | TCB14 | TCB13 | TCB12 | TCB11 | TCB10 | Timer L |
| HD111 | TLB | W | 8 | 8 | TLB17 | TLB16 | TLB15 | TLB14 | TLB13 | TLB12 | TLB11 | TLB10 | |
| HD112 | LMR | R/W | 8 | 8 | LMIF | LMIE | – | – | LMR3 | LMR2 | LMR1 | LMR0 | |
| HD113 | LTC | R | 8 | 8 | LTC7 | LTC6 | LTC5 | LTC4 | LTC3 | LTC2 | LTC1 | LTC0 | |
| HD113 | RCR | W | 8 | 8 | RCR7 | RCR6 | RCR5 | RCR4 | RCR3 | RCR2 | RCR1 | RCR0 | |
| HD118 | TMRM1 | R/W | 8 | 8 | CLR2 | AC/BR | RLD | RLCK | PS21 | PC20 | RLD/CAP | CPS | |
| HD119 | TMRM2 | R/W | 8 | 8 | LAT | RS11 | PS10 | PS31 | PS30 | CP/SLM | CAPF | SLW | |
| HD11A | TMRC1 | R | 8 | 8 | TMRC17 | TMRC16 | TMRC15 | TMRC14 | TMRC13 | TMRC12 | TMRC11 | TMRC10 | |
| HD11B | TMRC2 | R | 8 | 8 | TMRC27 | TMRC26 | TMRC25 | TMRC24 | TMRC23 | TMRC22 | TMRC21 | TMRC20 | |
| HD11C | TMRL1 | W | 8 | 8 | TMR17 | TMR16 | TMR15 | TMR14 | TMR13 | TMR12 | TMR11 | TMR10 | Timer R |
| HD11D | TMRL2 | W | 8 | 8 | TMR27 | TMR26 | TMR25 | TMR24 | TMR23 | TMR22 | TMR21 | TMR20 | |
| HD11E | TMRL3 | W | 8 | 8 | TMR37 | TMR36 | TMR35 | TMR34 | TMR33 | TMR32 | TMR31 | TMR30 | |
| HD11F | TMRC3 | R/W | 8 | 8 | TMRI3E | TMRI2E | TMRI1E | TMRI3 | TMRI2 | TMRI1 | – | – | |
| HD120 | PWDRL | W | 8 | 8 | PWDRL7 | PWDRL6 | PWDRL5 | PWDRL4 | PWDRL3 | PWDRL2 | PWDRL1 | PWDRL0 | |
| HD121 | PWDRU | W | 8 | 8 | – | – | PWDRU5 | PWDRU4 | PWDRU3 | PWDRU2 | PWDRU1 | PWDRU0 | |
| HD122 | PWCR | R/W | 8 | 8 | – | – | – | – | – | – | – | PWMCR0 | |
| HD126 | PWR0 | W | 8 | 8 | PW07 | PW06 | PW05 | PW04 | PW03 | PW02 | PW01 | PW00 | |
| HD127 | PWR1 | W | 8 | 8 | PW17 | PW16 | PW15 | PW14 | PW13 | PW12 | PW11 | PW10 | |
| HD128 | PWR2 | W | 8 | 8 | PW27 | PW26 | PW25 | PW24 | PW23 | PW22 | PW21 | PW20 | 8-bit PWM |
| HD129 | PWR3 | W | 8 | 8 | PW37 | PW36 | PW35 | PW34 | PW33 | PW32 | PW31 | PW30 | |
| HD12A | PW8CR | R/W | 8 | 8 | – | – | – | – | PWC3 | PWC2 | PWC1 | PWC0 | |
| HD12C | ICR1 | R | 8 | 8 | ICR17 | ICR16 | ICR15 | ICR14 | ICR13 | ICR12 | ICR11 | CIR10 | |
| HD12D | PCSR | R/W | 8 | 8 | ICIF | ICIE | ICEG | NCon/off | – | DCS2 | DCS1 | DCS0 | |

| Address* | Register Name | R/W | Access | Bus Width | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Module Name |
|----------|---------------|-----|--------|-----------|-------|-------|-------|-------|---------|--------|--------|---------|---|
| HD130 | ADRH | R | 16 | 8 | ADR9 | ADR8 | ADR7 | ADR6 | ADR5 | ADR4 | ADR3 | ADR2 | A/D |
| HD131 | ADRL | | | | ADR1 | ADR0 | — | — | — | — | — | — | |
| HD132 | AHRH | R | 16 | 8 | AHR9 | AHR8 | AHR7 | AHR6 | AHR5 | AHR4 | AHR3 | AHR2 | |
| HD133 | AHRL | | | | AHR1 | AHR0 | — | — | — | — | — | — | |
| HD134 | ADCR | R/W | 8 | 8 | CK | — | HCH1 | HCH0 | SCH3 | SCH2 | SCH1 | SCH0 | |
| HD135 | ADCSR | R/W | 8 | 8 | SEND | HEND | ADIE | SST | HST | BUSY | SCNL | — | |
| HD136 | ADTSR | R/W | 8 | 8 | — | — | — | — | — | — | TRGS1 | TRGS0 | |
| HD138 | TLK | W | 8/16 | 16 | TLR27 | TLR26 | TLR25 | TLR24 | TLR23 | TLR22 | TLR21 | TLR20 | Timer J |
| HD138 | TCK | R | 8/16 | 16 | TLR17 | TLR16 | TLR15 | TLR14 | TLR13 | TLR12 | TLR11 | TLR10 | |
| HD139 | TLJ | W | 8/16 | 16 | TDR27 | TDR26 | TDR25 | TDR24 | TDR23 | TDR22 | TDR21 | TDR20 | |
| HD139 | TCJ | R | 8/16 | 16 | TDR17 | TDR16 | TDR15 | TDR14 | TDR13 | TDR12 | TDR11 | TDR10 | * The PS22 bit is available only in the H8S/2194C series. |
| HD13A | TMJ | R/W | 8/16 | 16 | PS11 | PS10 | ST | 8/16 | PS21 | PS20 | TGL | T/R | |
| HD13B | TMJC | R/W | 8/16 | 16 | BUZZ1 | BUZZ0 | MON1 | MON0 | — | TMJ2IE | TMJ1IE | (PS22)* | |
| HD13C | TMJS | R/W | 8/16 | 16 | TMJ2I | TMJ1I | — | — | — | — | — | — | |
| HD148 | SMR1 | R/W | 8 | 8 | C/Ā | CHR | PE | O/Ē | STOP | MP | CKS1 | CKS0 | Clock synchronizati on/start-stop sync SCI |
| HD149 | BRR1 | R/W | 8 | 8 | — | — | — | — | — | — | — | — | |
| HD14A | SCR1 | R/W | 8 | 8 | TEI | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 | |
| HD14B | TDR1 | R/W | 8 | 8 | — | — | — | — | — | — | — | — | |
| HD14C | SSR1 | R/W | 8 | 8 | TDRE | RDRF | ORER | FER | PER | TEMD | MPB | MPBT | |
| HD14D | RDR1 | R | 8 | 8 | — | — | — | — | — | — | — | — | |
| HD14E | SCMR1 | R/W | 8 | 8 | — | — | — | — | SDIR | SINV | — | SMIF | |
| HD158 | ICCR | R/W | 8 | 8 | ICE | IEIC | MST | TRS | ACKÉ | BBSY | IRIC | SCP | IIC interface |
| HD159 | ICSR | R/W | 8 | 8 | ESTP | STOP | IRTR | AASX | AL | AAS | ADZ | ACKB | * Access varies depending on ICE bit. |
| HD15E | ICDR* | R/W | 8 | 8 | ICDR7 | ICDR6 | ICDR5 | ICDR4 | ICDR3 | ICDR2 | ICDR1 | ICDR0 | |
| HD15E | SARX* | R/W | 8 | 8 | SVAX6 | SVAX5 | SVAX4 | SVAX3 | SVAX2 | SVAX1 | SVAX0 | FSX | |
| HD15F | ICMR* | R/W | 8 | 8 | MLS | WAIT | CKS2 | CKS1 | CKS0 | BC2 | BC1 | BC0 | |
| HD15F | SAR* | R/W | 8 | 8 | SVA6 | SVA5 | SVA4 | SVA3 | SVA2 | SVA1 | SVA0 | FS | |
| HFFB0 | TAR0 | R/W | 8 | 8 | TA023 | TA022 | TA021 | TA020 | TA019 | TA018 | TA017 | TA016 | ATC |
| HFFB1 | | | | | TA015 | TA014 | TA013 | TA012 | TA011 | TA010 | TA009 | TA008 | |
| HFFB2 | | | | | TA007 | TA006 | TA005 | TA004 | TA003 | TA002 | TA001 | — | |
| HFFB3 | TAR1 | R/W | 8 | 8 | TA123 | TA122 | TA121 | TA120 | TA119 | TA118 | TA117 | TA116 | |
| HFFB4 | | | | | TA115 | TA114 | TA113 | TA112 | TA111 | TA110 | TA109 | TA108 | |
| HFFB5 | | | | | TA107 | TA106 | TA105 | TA104 | TA103 | TA102 | TA101 | — | |
| HFFB6 | TAR2 | R/W | 8 | 8 | TA223 | TA222 | TA221 | TA220 | TA219 | TA218 | TA217 | TA216 | |
| HFFB7 | | | | | TA215 | TA214 | TA213 | TA212 | TA211 | TA210 | TA209 | TA208 | |
| HFFB8 | | | | | TA207 | TA206 | TA205 | TA204 | TA203 | TA202 | TA201 | — | |
| HFFB9 | TRCR | R/W | 8 | 8 | — | — | — | — | — | TRC2 | TRC1 | TRC0 | |
| HFFBA | TMA | R/W | 8 | 8 | TMAOV | TMAIE | — | — | TMA3 | TMA2 | TMA1 | TMA0 | Timer A |
| HFFBB | TCA | R | 8 | 8 | TCA7 | TCA6 | TCA5 | TCA4 | TCA3 | TCA2 | TCA1 | TCA0 | |
| HFFBC | WTC SR | R/W | 8/16 | 16 | OVF | WT/IĒ | TME | RSTS | RST/NMI | CKS2 | CKS1 | CKS0 | WDT |
| HFFBD | WTCNT | R/W | 8/16 | 16 | — | — | — | — | — | — | — | — | |
| HFFC0 | PDR0 | R | 8 | 8 | PDR07 | PDR06 | PDR05 | PDR04 | PDR03 | PDR02 | PDR01 | PDR00 | Port data register |
| HFFC1 | PDR1 | R/W | 8 | 8 | PDR17 | PDR16 | PDR15 | PDR14 | PDR13 | PDR12 | PDR11 | PDR10 | |
| HFFC2 | PDR2 | R/W | 8 | 8 | PDR27 | PDR26 | PDR25 | PDR24 | PDR23 | PDR22 | PDR21 | PDR20 | |
| HFFC3 | PDR3 | R/W | 8 | 8 | PDR37 | PDR36 | PDR35 | PDR34 | PDR33 | PDR32 | PDR31 | PDR30 | |
| HFFC4 | PDR4 | R/W | 8 | 8 | PDR47 | PDR46 | PDR45 | PDR44 | PDR43 | PDR42 | PDR41 | PDR40 | |
| HFFC5 | PDR5 | R/W | 8 | 8 | — | — | — | — | PDR53 | PDR52 | PDR51 | PDR50 | |
| HFFC6 | PDR6 | R/W | 8 | 8 | PDR67 | PDR66 | PDR65 | PDR64 | PDR63 | PDR62 | PDR61 | PDR60 | |
| HFFC7 | PDR7 | R/W | 8 | 8 | PDR77 | PDR76 | PDR75 | PDR74 | PDR73 | PDR72 | PDR71 | PDR70 | |
| HFFC8 | PDR8 | R/W | 8 | 8 | PDR87 | PDR86 | PDR85 | PDR84 | PDR83 | PDR82 | PDR81 | PDR80 | |
| HFFCD | PMR0 | R/W | 8 | 8 | PMR07 | PMR06 | PMR05 | PMR04 | PMR03 | PMR02 | PMR01 | PMR00 | Port mode register |
| HFFCE | PMR1 | R/W | 8 | 8 | PMR17 | PMR16 | PMR15 | PMR14 | PMR13 | PMR12 | PMR11 | PMR10 | |
| HFFCF | PMR2 | R/W | 8 | 8 | PMR27 | PMR26 | PMR25 | — | — | — | — | PMR20 | |

| Address* | Register Name | R/W | Access | Bus Width | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Module Name |
|----------|---------------|-----|--------|-----------|--------|--------|--------|--------|--------|--------|---------|---------|---|
| HFFD0 | PMR3 | R/W | 8 | 8 | PMR37 | PMR36 | PMR35 | PMR34 | PMR33 | PMR32 | PMR31 | PMR30 | Port mode register |
| HFFD1 | PCR1 | W | 8 | 8 | PCR17 | PCR16 | PCR15 | PCR14 | PCR13 | PCR12 | PCR11 | PCR10 | Port control register |
| HFFD2 | PCR2 | W | 8 | 8 | PCR27 | PCR26 | PCR25 | PCR24 | PCR23 | PCR22 | PCR21 | PCR20 | |
| HFFD3 | PCR3 | W | 8 | 8 | PCR37 | PCR36 | PCR35 | PCR34 | PCR33 | PCR32 | PCR31 | PCR30 | |
| HFFD4 | PCR4 | W | 8 | 8 | PCR47 | PCR46 | PCR45 | PCR44 | PCR43 | PCR42 | PCR41 | PCR40 | |
| HFFD5 | PCR5 | W | 8 | 8 | — | — | — | — | PCR53 | PCR52 | PCR51 | PCR50 | |
| HFFD6 | PCR6 | W | 8 | 8 | PCR67 | PCR66 | PCR65 | PCR64 | PCR63 | PCR62 | PCR61 | PCR60 | |
| HFFD7 | PCR7 | W | 8 | 8 | PCR77 | PCR76 | PCR75 | PCR74 | PCR73 | PCR72 | PCR71 | PCR70 | |
| HFFD8 | PCR8 | W | 8 | 8 | PCR87 | PCR86 | PCR85 | PCR84 | PCR83 | PCR82 | PCR81 | PCR380 | |
| HFFDB | PMR4 | R/W | 8 | 8 | — | — | — | — | — | — | — | PMR40 | Port mode register |
| HFFDC | PMR5 | R/W | 8 | 8 | — | — | — | — | PMR53 | PMR52 | PMR51 | PMR50 | |
| HFFDD | PMR6 | R/W | 8 | 8 | PMR67 | PMR66 | PMR65 | PMR64 | PMR63 | PMR62 | PMR61 | PMR60 | |
| HFFDE | PMR7 | R/W | 8 | 8 | PMR77 | PMR76 | PMR75 | PMR74 | PMR73 | PMR72 | PMR71 | PMR70 | |
| HFFDF | PMR8 | R/W | 8 | 8 | — | — | — | — | PMR83 | PMR82 | PMR81 | PMR80 | |
| HFFE1 | PUR1 | R/W | 8 | 8 | PUR17 | PUR16 | PUR15 | PUR14 | PUR13 | PUR12 | PUR11 | PUR10 | Port pull-up select register |
| HFFE2 | PUR2 | R/W | 8 | 8 | PUR27 | PUR26 | PUR25 | PUR24 | PUR23 | PUR22 | PUR21 | PUR20 | |
| HFFE3 | PUR3 | R/W | 8 | 8 | PUR37 | PUR36 | PUR35 | PUR34 | PUR33 | PUR32 | PUR31 | PUR30 | |
| HFFE4 | RTPEGR | R/W | 8 | 8 | — | — | — | — | — | — | RTPEGR1 | RTPEGR0 | RTP TRG select |
| HFFE5 | RTPSR | R/W | 8 | 8 | RTPSR7 | RTPSR6 | RTPSR5 | RTPSR4 | RTPSR3 | RTPSR2 | RTPSR1 | RTPSR0 | |
| HFFE8 | SYSCR | R/W | 8 | 8 | — | — | INTM1 | INTM0 | XRST | NMIEG1 | NMIEG0 | — | System control register |
| HFFE9 | MDCR | R/W | 8 | 8 | — | — | — | — | — | — | — | MDS0 | |
| HFFEA | SBYCR | R/W | 8 | 8 | SSBY | STS2 | STS1 | STS0 | — | — | — | — | |
| HFFEB | LPWRCR | R/W | 8 | 8 | DTON | LSON | NESEL | — | — | — | SA1 | SA0 | |
| HFFEC | MSTPCRH | R/W | 8 | 8 | MSTP15 | MSTP14 | MSTP13 | MSTP12 | MSTP11 | MSTP10 | MSTP9 | MSTP8 | |
| HFFED | MSTPCRL | R/W | 8 | 8 | MSTP7 | MSTP6 | MSTP5 | MSTP4 | MSTP3 | MSTP2 | MSTP1 | MSTP0 | |
| HFFEE | STCR | R/W | 8 | 8 | — | IICX | IICRST | — | FLASHE | — | — | — | |
| HFFF0 | IEGR | R/W | 8 | 8 | — | IRQ5EG | IRQ4EG | IRQ3EG | IRQ2EG | IRQ1EG | IRQ0EG1 | IRQ0EG0 | IRQ edge |
| HFFF1 | IENR | R/W | 8 | 8 | — | — | IRQ5E | IRQ4E | IRQ3E | IRQ2E | IRQ1E | IRQ0E | IRQ enable |
| HFFF2 | IRQR | R/W | 8 | 8 | — | — | IRQ5F | IRQ4F | IRQ3F | IRQ2F | IRQ1F | IRQ0F | IRQ status |
| HFFF3 | ICRA | R/W | 8 | 8 | ICRA7 | ICRA6 | ICRA5 | ICRA4 | ICRA3 | ICRA2 | ICRA1 | ICRA0 | IRQ priority control |
| HFFF4 | ICRB | R/W | 8 | 8 | ICRB7 | ICRB6 | ICRB5 | ICRB4 | ICRB3 | ICRB2 | ICRB1 | ICRB0 | |
| HFFF5 | ICRC | R/W | 8 | 8 | ICRC7 | ICRC6 | ICRC5 | ICRC4 | ICRC3 | ICRC2 | ICRC1 | ICRC0 | |
| HFFF6 | ICRD | R/W | 8 | 8 | ICRD7 | ICRD6 | ICRD5 | ICRD4 | ICRD3 | ICRD2 | ICRD1 | ICRD0 | |
| HFFF8 | FLMCR1 | R/W | 8 | 8 | FWE | SWE | — | — | EV | PV | E | P | Only for FLASH version. |
| HFFF9 | FLMCR2 | R/W | 8 | 8 | FLER | — | — | — | — | — | ESU | PSU | |
| HFFFA | EBR1 | R/W | 8 | 8 | — | — | — | — | — | — | EB9 | EB8 | |
| HFFFB | EBR2 | R/W | 8 | 8 | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 | |
| HFFF8 | FLMCR1 | R/W | 8 | 8 | FWE | SWE | ESU1 | PSU1 | EV1 | PV1 | E1 | P1 | Only for FLASH version in the H8S/2194C |
| HFFF9 | FLMCR2 | R/W | 8 | 8 | FLER | — | ESU2 | PSU2 | EV2 | PV2 | E2 | P2 | |
| FFFFA | EBR1 | R/W | 8 | 8 | — | — | EB13 | EB12 | EB11 | EB10 | EB9 | EB8 | |
| FFFFB | EBR2 | R/W | 8 | 8 | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 | |

Note: * Lower 16 bits of the address.

B.2 Function List

H'D000: Gain Constant DGKp: Drum Digital Filter

H'D001: Gain Constant DGKp: Drum Digital Filter

H'D002: Gain Constant DGKs: Drum Digital Filter

H'D003: Gain Constant DGKs: Drum Digital Filter

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D004: Coefficient DAp: Drum Digital Filter

H'D005: Coefficient DAp: Drum Digital Filter

H'D006: Coefficient DBp: Drum Digital Filter

H'D007: Coefficient DBp: Drum Digital Filter

H'D008: Coefficient DAs: Drum Digital Filter

H'D009: Coefficient DAs: Drum Digital Filter

H'D00A: Coefficient DBs: Drum Digital Filter

H'D00B: Coefficient DBs: Drum Digital Filter

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D00C: Offset DOfp: Drum Digital Filter

H'D00D: Offset DOfp: Drum Digital Filter

H'D00E: Offset DOfs: Drum Digital Filter

H'D00F: Offset DOfs: Drum Digital Filter

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D010: Gain Constant CGKp: Capstan Digital Filter

H'D011: Gain Constant CGKp: Capstan Digital Filter

H'D012: Gain Constant CGKs: Capstan Digital Filter

H'D013: Gain Constant CGKs: Capstan Digital Filter

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D014: Coefficient C_{Ap}: Capstan Digital Filter

H'D015: Coefficient C_{Ap}: Capstan Digital Filter

H'D016: Coefficient C_{Bp}: Capstan Digital Filter

H'D017: Coefficient C_{Bp}: Capstan Digital Filter

H'D018: Coefficient C_{As}: Capstan Digital Filter

H'D019: Coefficient C_{As}: Capstan Digital Filter

H'D01A: Coefficient C_{Bs}: Capstan Digital Filter

H'D01B: Coefficient C_{Bs}: Capstan Digital Filter

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D01C: Offset C_{O_fp}: Capstan Digital Filter

H'D01D: Offset C_{O_fp}: Capstan Digital Filter

H'D01E: Offset C_{O_fs}: Capstan Digital Filter

H'D01F: Offset C_{O_fs}: Capstan Digital Filter

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D020: Delay Initialization Register DZs: Digital filter

H'D021: Delay Initialization Register DZs: Digital filter

H'D022: Delay Initialization Register DZp: Digital filter

H'D023: Delay Initialization Register DZp: Digital filter

H'D024: Delay Initialization Register CZs: Digital filter

H'D025: Delay Initialization Register CZs: Digital filter

H'D026: Delay Initialization Register CZp: Digital filter

H'D027: Delay Initialization Register CZp: Digital filter

| | | | | | | | | | | | | | | | | |
|-----------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

H'D028: Drum System Digital Filter Control Register DFIC: Digital Filter

| | | | | | | | | |
|-----------------|---|--------|-------|-------|-------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | DROV | DPHA | DZPON | DZSON | DSG2 | DSG1 | DSG0 |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | R/(W)* | R/(W) | R/W | R/W | R/W | R/W | R/W |

Drum system gain control bit

| DSG2 | DSG1 | DSG0 | Description |
|------|------|------|----------------------|
| 0 | 0 | 0 | x 1 |
| | | 1 | x 2 |
| | 1 | 0 | x 4 |
| | | 1 | x 8 |
| 1 | 0 | 0 | x 16 |
| | | 1 | (x 32)* |
| | 1 | 0 | (x 64)* |
| | | 1 | Invalid (do not set) |

Note: * Optional

Drum speed system Z⁻¹ initialization bit

| | |
|---|---|
| 0 | Speed system Z ⁻¹ reflects DZs value. |
| 1 | Speed system Z ⁻¹ does not select DZs value. |

Drum phase system Z⁻¹ initialization bit

| | |
|---|---|
| 0 | Phase system Z ⁻¹ reflects DZp value. |
| 1 | Phase system Z ⁻¹ does not select DZp value. |

Drum phase system filter computation start bit

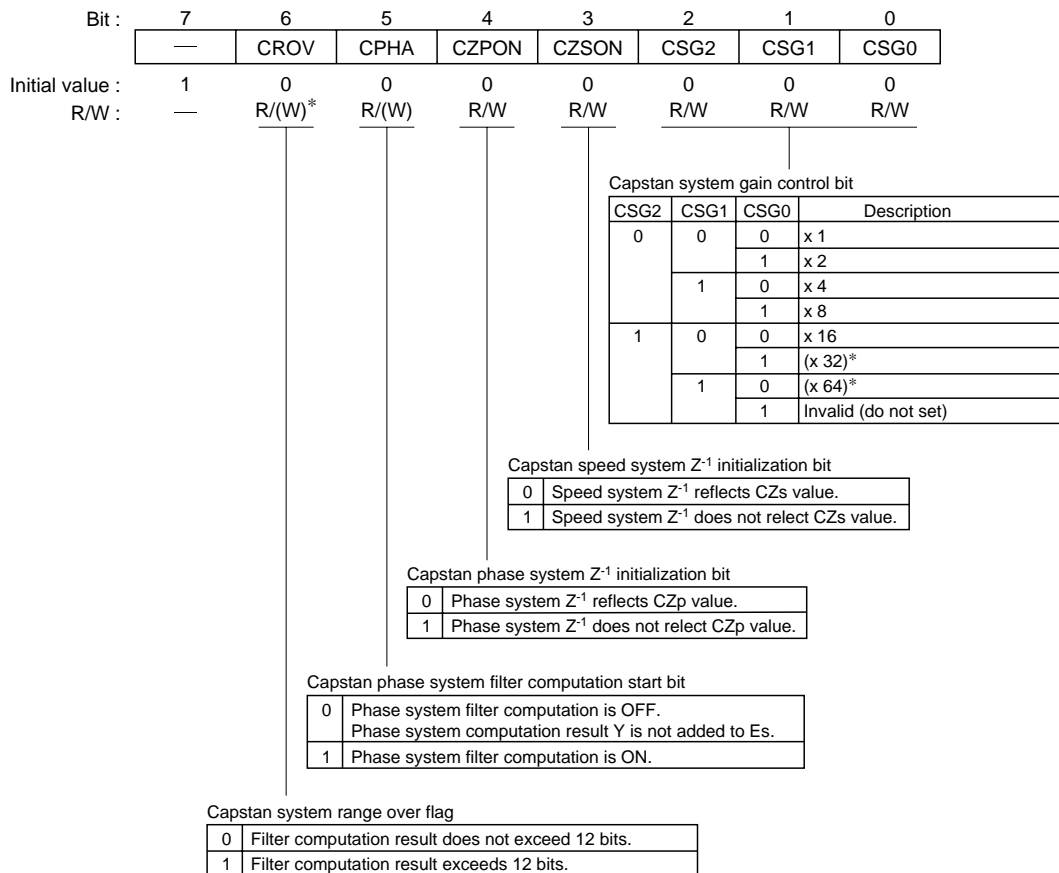
| | |
|---|--|
| 0 | Phase system filter computation is OFF. Phase system computation result Y is not added to Es. |
| 1 | Phase system filter computation is ON. |

Drum system range over flag

| | |
|---|--|
| 0 | Filter computation result does not exceed 12 bits. |
| 1 | Filter computation result exceeds 12 bits. |

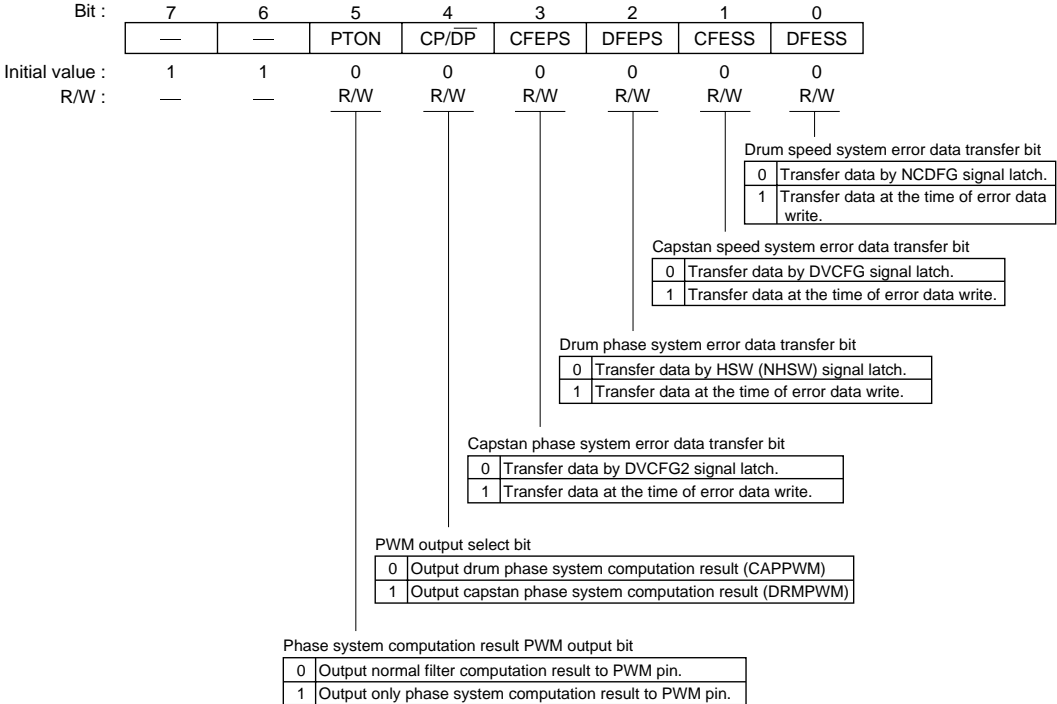
Note: * Only 0 can be written.

H'D029: Capstan System Digital Filter Control Register CFIC: Digital Filter



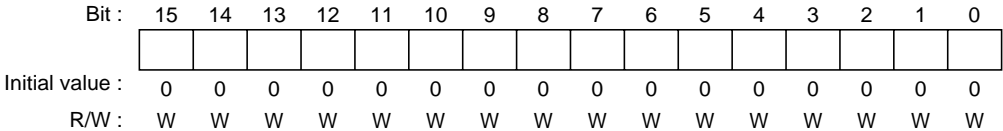
Note: * Only 0 can be written.

H'D02A: Digital Filter Control Register DFUCR: Digital Filter



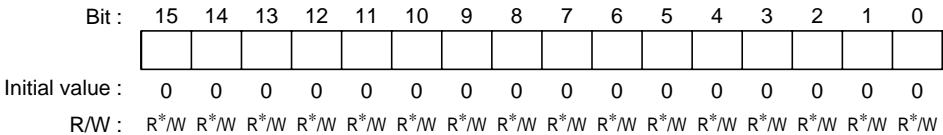
H'D030: Specified DFG Speed Preset Data Register DFPR: Drum Error Detector

H'D031: Specified DFG Speed Preset Data Register DFPR: Drum Error Detector



H'D032: DFG Speed Error Data Register DFER: Drum Error Detector

H'D033: DFG Speed Error Data Register DFER: Drum Error Detector



H'D034: DFG Lock Upper Data Register DFRUDR: Drum Error Detector

H'D035: DFG Lock Upper Data Register DFRUDR: Drum Error Detector

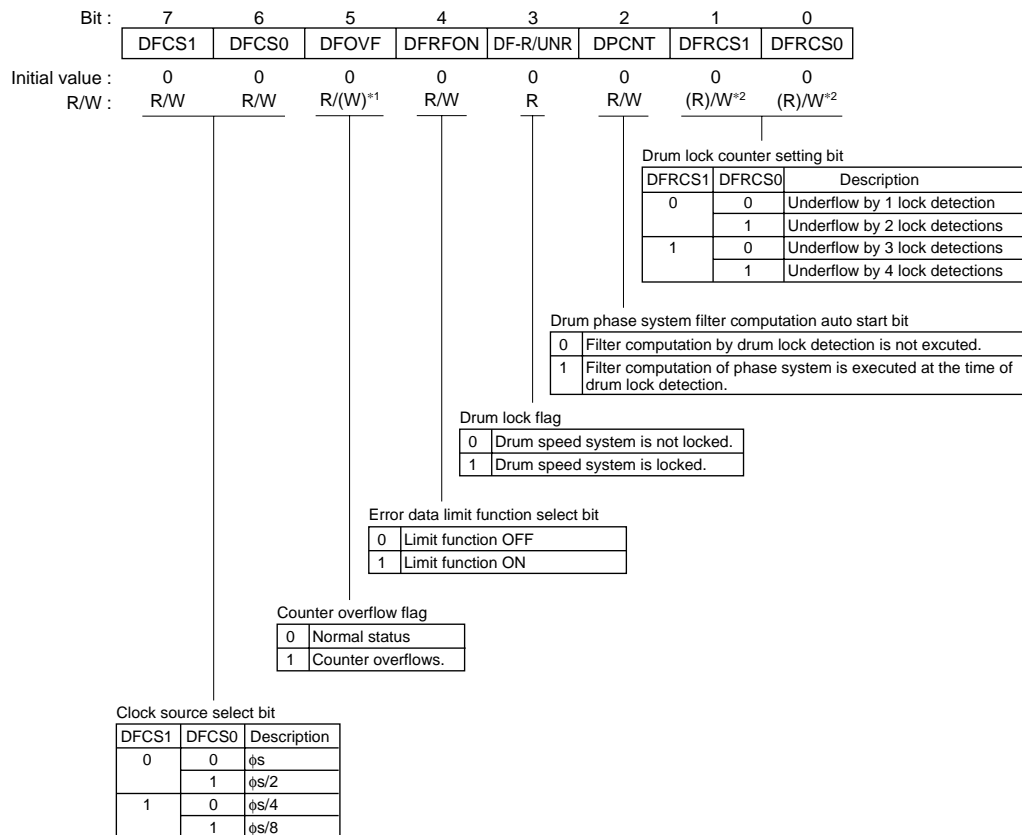
| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D036: DFG Lock Lower Data Register DFRLDR: Drum Error Detector

H'D037: DFG Lock Lower Data Register DFRLDR: Drum Error Detector

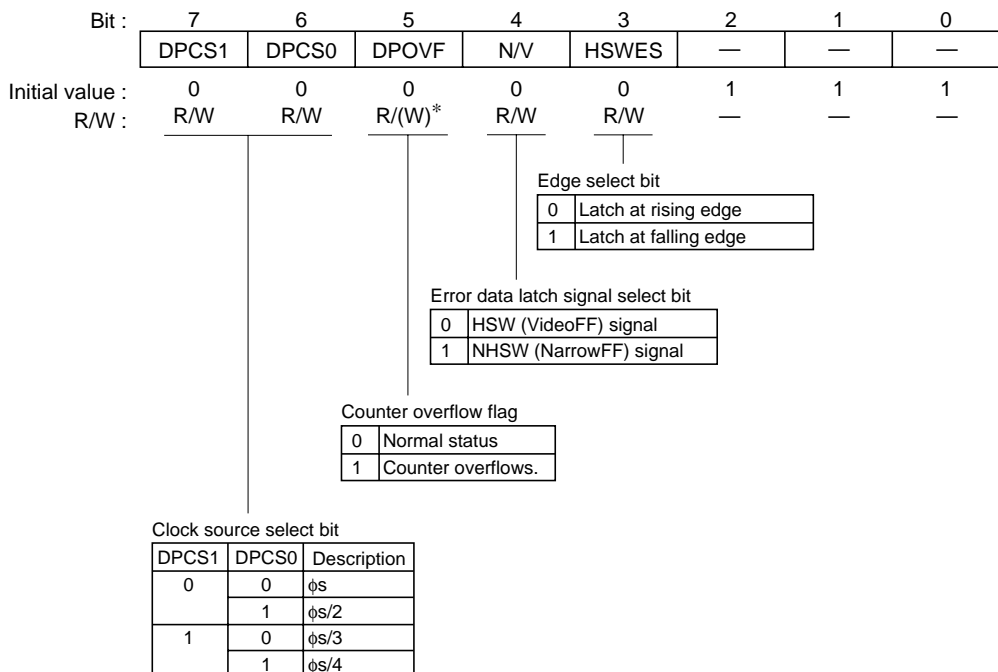
| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D038: Drum Speed Error Detection Control Register DFVCR: Drum Error Detector



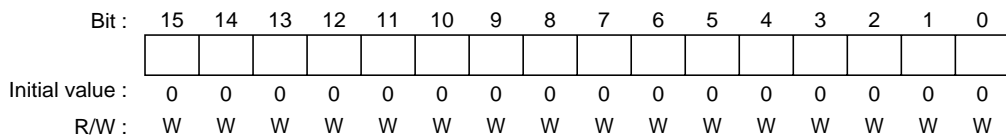
- Notes: 1. Only 0 can be written.
 2. When read, counter value is read.

H'D039: Drum Phase Error Detection Control Register DPGCR: Drum Error Detector

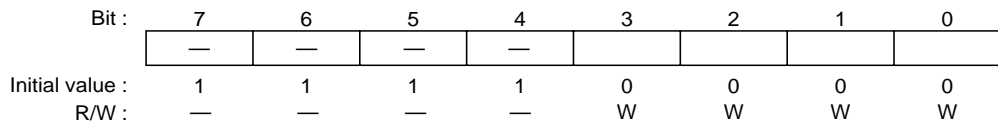


Note: * Only 0 can be written.

H'D03A: Specified Drum Phase Preset Data Register 2 DPPR2: Drum Error Detector



H'D03C: Specified Drum Phase Preset Data Register 1 DPPR1: Drum Error Detector



H'D03D: Drum Phase Error Data Register 1 DPER1: Drum Error Detector

| | | | | | | | | |
|-----------------|---|---|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | | | | |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | R*/W | R*/W | R*/W | R*/W |

H'D03E: Drum Phase Error Data Register 2 DPER2: Drum Error Detector

| | | | | | | | | | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

Note: * Note that only detected error data can be read.

H'D050: Specified CFG Speed Preset Data Register CFPR: Capstan Error Detector

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D052: CFG Speed Error Data Register CFER: Capstan Error Detector

| | | | | | | | | | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

Note: * Note that only detected error data can be read.

H'D054: CFG Lock Upper Data Register CFRUDR: Capstan Error Detector

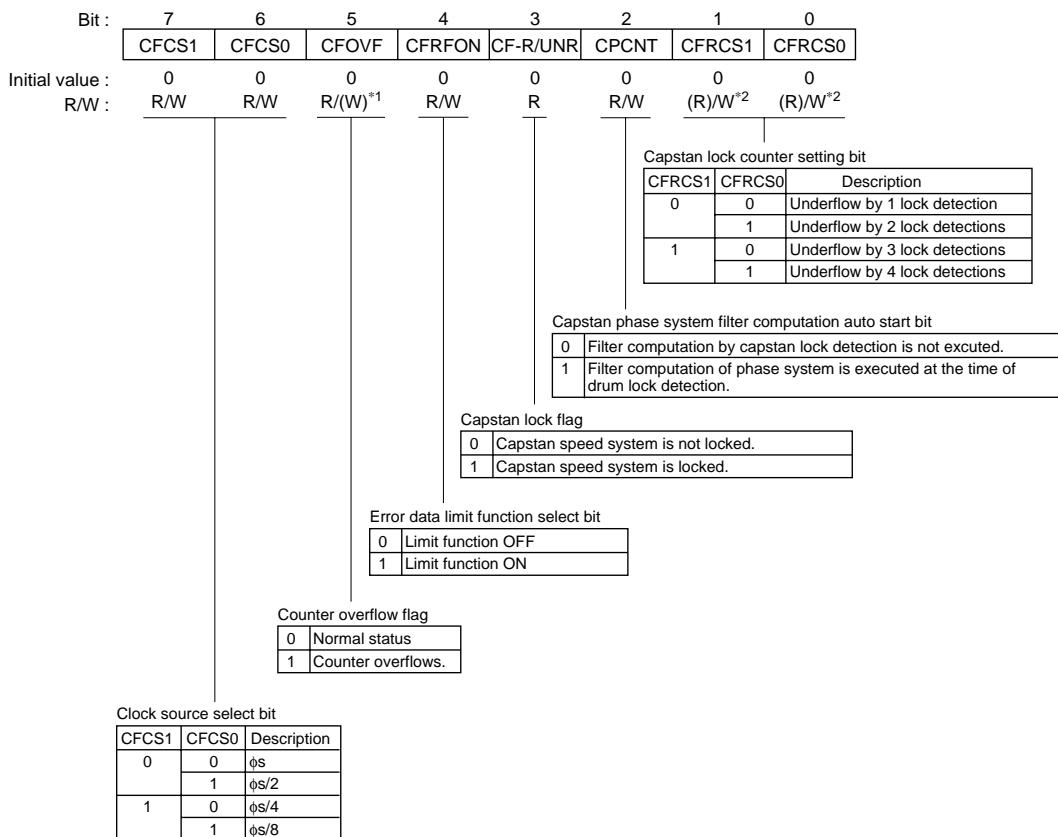
| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D056: CFG Lock Lower Data Register CFRLDR: Capstan Error Detector

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D058: Capstan Speed Error Detection Control Register

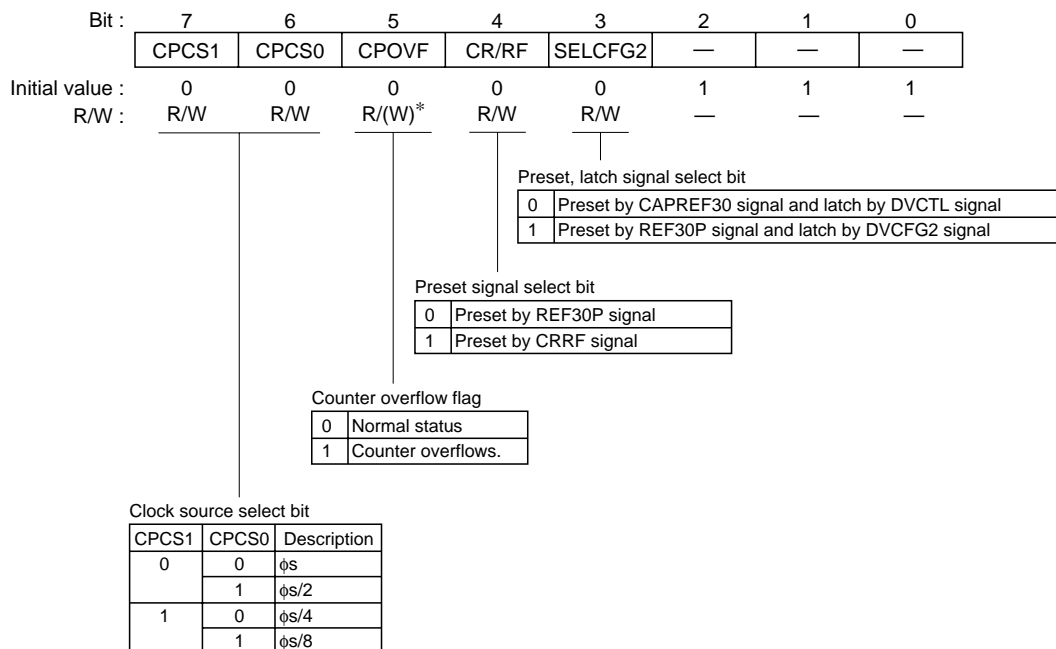
CFVCR: Capstan Error Detector



- Notes: 1. Only 0 can be written.
 2. When read, counter value is read.

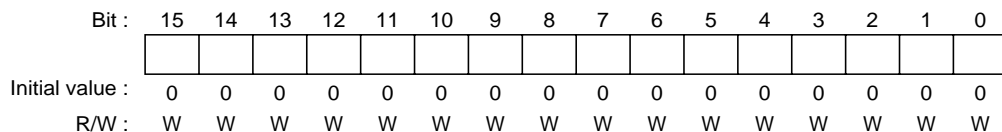
H'D059: Capstan Phase Error Detection Control Register

CPGCR: Capstan Error Detector

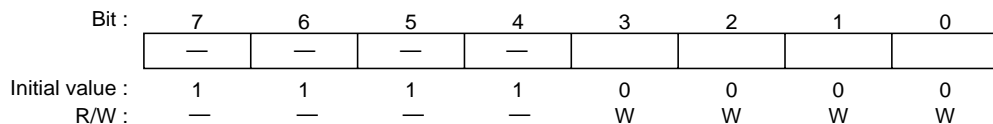


Note: * Only 0 can be written.

H'D05A: Specified Capstan Phase Preset Data Register 2 CPPR2: Capstan Error Detector



H'D05C: Specified Capstan Phase Preset Data Register 1 CPPR1: Capstan Error Detector



H'D05D: Capstan Phase Error Data Register 1 CPER1: Capstan Error Detector

| | | | | | | | | |
|-----------------|---|---|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | | | | |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | R*/W | R*/W | R*/W | R*/W |

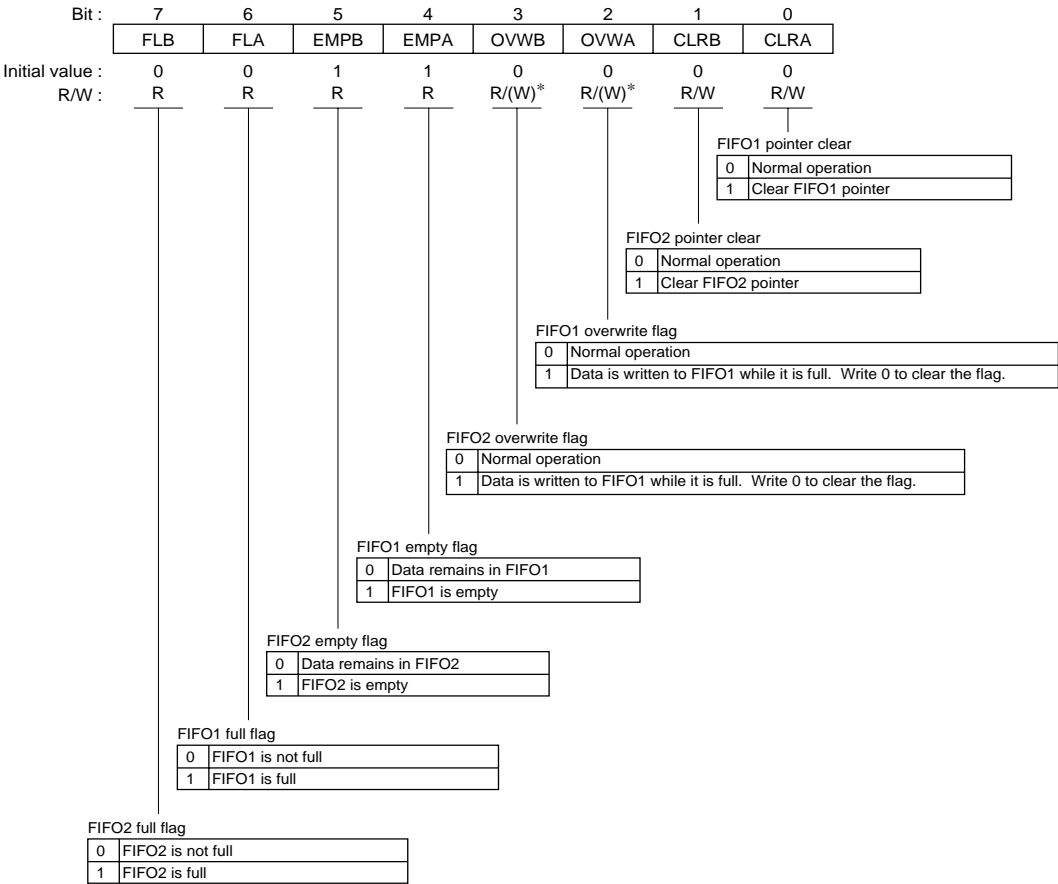
Note: * Note that only detected error data can be read.

H'D05E: Capstan Phase Error Data Register 2 CPER2: Capstan Error Detector

| | | | | | | | | | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

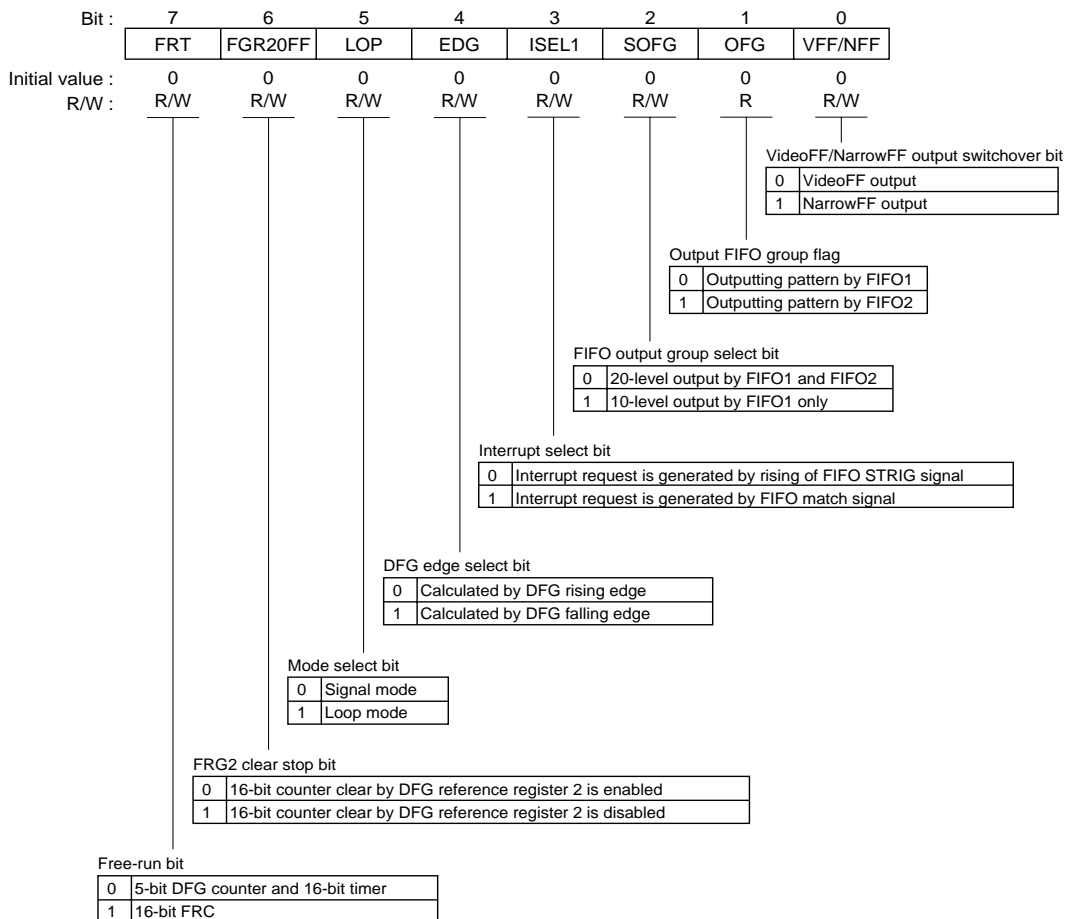
Note: * Note that only detected error data can be read.

H'D060: HSW Mode Register 1 HSM1: HSW Timing Generator



Note: * Only 0 can be written.

H'D061: HSW Mode Register 2 HSM2: HSW Timing Generator



H'D062: HSW Loop Stage Setting Register HSLP: HSW Timing Generator

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | LOB3 | LOB2 | LOB1 | LOB0 | LOA3 | LOA2 | LOA1 | LOA0 |
| Initial value : | * | * | * | * | * | * | * | * |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| FIFO1 stage setting bit _____ | | | | | |
|-------------------------------|-------|-------|-------|-------|-------------------------------|
| HSM2 | HSLP | | | | Description |
| Bit 5 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| LOP | LOA3 | LOA2 | LOA1 | LOA0 | |
| 0 | * | * | * | * | Single mode |
| 1 | 0 | 0 | 0 | 0 | Output stage 0 of FIFO1 |
| | | | | 1 | Output stage 0 and 1 of FIFO1 |
| | | | 1 | 0 | Output stage 0 to 2 of FIFO1 |
| | | | | 1 | Output stage 0 to 3 of FIFO1 |
| | | 1 | 0 | 0 | Output stage 0 to 4 of FIFO1 |
| | | | | 1 | Output stage 0 to 5 of FIFO1 |
| | 1 | 0 | 1 | 0 | Output stage 0 to 6 of FIFO1 |
| | | | | 1 | Output stage 0 to 7 of FIFO1 |
| | | | 0 | 0 | Output stage 0 to 8 of FIFO1 |
| | | | | 1 | Output stage 0 to 9 of FIFO1 |
| | | 1 | 1 | 0 | Setting disabled |
| | | | | 1 | |
| | | | | 0 | |
| | | | | 1 | |
| | | | | 0 | |

Note: * Don't care.

| FIFO2 stage setting bit _____ | | | | | |
|-------------------------------|-------|-------|-------|-------|-------------------------------|
| HSM2 | HSLP | | | | Description |
| Bit 5 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | |
| LOP | LOB3 | LOB2 | LOB1 | LOB0 | |
| 0 | * | * | * | * | Single mode |
| 1 | 0 | 0 | 0 | 0 | Output stage 0 of FIFO2 |
| | | | | 1 | Output stage 0 and 1 of FIFO2 |
| | | | 1 | 0 | Output stage 0 to 2 of FIFO2 |
| | | | | 1 | Output stage 0 to 3 of FIFO2 |
| | | 1 | 0 | 0 | Output stage 0 to 4 of FIFO2 |
| | | | | 1 | Output stage 0 to 5 of FIFO2 |
| | 1 | 0 | 1 | 0 | Output stage 0 to 6 of FIFO2 |
| | | | | 1 | Output stage 0 to 7 of FIFO2 |
| | | | 0 | 0 | Output stage 0 to 8 of FIFO2 |
| | | | | 1 | Output stage 0 to 9 of FIFO2 |
| | | 1 | 1 | 0 | Setting disabled |
| | | | | 1 | |
| | | | | 0 | |
| | | | | 1 | |
| | | | | 0 | |

Note: * Don't care.

H'D064: FIFO Output Pattern Register 1 FPDRA: HSW Timing Generator

| | | | | | | | | |
|-----------------|----|--------|--------|-----------|------|------|---------|---------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | ADTRGA | STRIGA | NarrowFFA | VFFA | AFFA | VpulseA | MlevelA |
| Initial value : | 1 | * | * | * | * | * | * | * |
| R/W : | — | W | W | W | W | W | W | W |

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PPGA7 | PPGA6 | PPGA5 | PPGA4 | PPGA3 | PPGA2 | PPGA1 | PPGA0 |
| Initial value : | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W |

H'D066: FIFO Timing Pattern Register 1 FTPRA: HSW Timing Generator

| | | | | | | | | | | | | | | | | |
|-----------------|---------|---------|---------|---------|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FTPRA15 | FTPRA14 | FTPRA13 | FTPRA12 | FTPRA11 | FTPRA10 | FTPRA9 | FTPRA8 | FTPRA7 | FTPRA6 | FTPRA5 | FTPRA4 | FTPRA3 | FTPRA2 | FTPRA1 | FTPRA0 |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

H'D066: FIFO Timer Capture Register 1 FTCTR: HSW Timing Generator

| | | | | | | | | | | | | | | | | |
|-----------------|---------|---------|---------|---------|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FTCTR15 | FTCTR14 | FTCTR13 | FTCTR12 | FTCTR11 | FTCTR10 | FTCTR9 | FTCTR8 | FTCTR7 | FTCTR6 | FTCTR5 | FTCTR4 | FTCTR3 | FTCTR2 | FTCTR1 | FTCTR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |

H'D068: FIFO Output Pattern Register 2 FPDRB: HSW Timing Generator

| | | | | | | | | |
|-----------------|----|--------|--------|-----------|------|------|---------|---------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | ADTRGB | STRIGB | NarrowFFB | VFFB | AFFB | VpulseB | MlevelB |
| Initial value : | 1 | * | * | * | * | * | * | * |
| R/W : | — | W | W | W | W | W | W | W |

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PPGB7 | PPGB6 | PPGB5 | PPGB4 | PPGB3 | PPGB2 | PPGB1 | PPGB0 |
| Initial value : | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W |

Note: * Undetermined

H'D06A: FIFO Timing Pattern Register 2 FTPRB: HSW Timing Generator

| | | | | | | | | | | | | | | | | |
|-----------------|---------|---------|---------|---------|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FTPRB15 | FTPRB14 | FTPRB13 | FTPRB12 | FTPRB11 | FTPRB10 | FTPRB9 | FTPRB8 | FTPRB7 | FTPRB6 | FTPRB5 | FTPRB4 | FTPRB3 | FTPRB2 | FTPRB1 | FTPRB0 |
| Initial value : | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

Note: * Undetermined

H'D06C: DFG Reference Register 1 DFCRA: HSW Timing Generator

| | | | | | | | | |
|-----------------|-------|------|------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ISEL2 | CCLR | CKSL | DFCRA4 | DFCRA3 | DFCRA2 | DFCRA1 | DFCRA0 |
| Initial value : | 0 | 0 | 0 | * | * | * | * | * |
| R/W : | W | W | W | W | W | W | W | W |

16-bit counter clock source select bit

| | |
|---|------------|
| 0 | ϕ S/4 |
| 1 | ϕ S/8 |

DFG counter clear bit

| | |
|---|-------------------------|
| 0 | Normal operation |
| 1 | Clear 5-bit DFG counter |

Interrupt select bit

| | |
|---|--|
| 0 | Interrupt request is generated by clear signal of 16-bit timer counter |
| 1 | Interrupt request is generated by VD signal in PB mode |

H'D06C: DFG Reference Count Register DFCTR: HSW Timing Generator

| | | | | | | | | |
|-----------------|---|---|---|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | DFCTR4 | DFCTR3 | DFCTR2 | DFCTR1 | DFCTR0 |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | R | R | R | R | R |

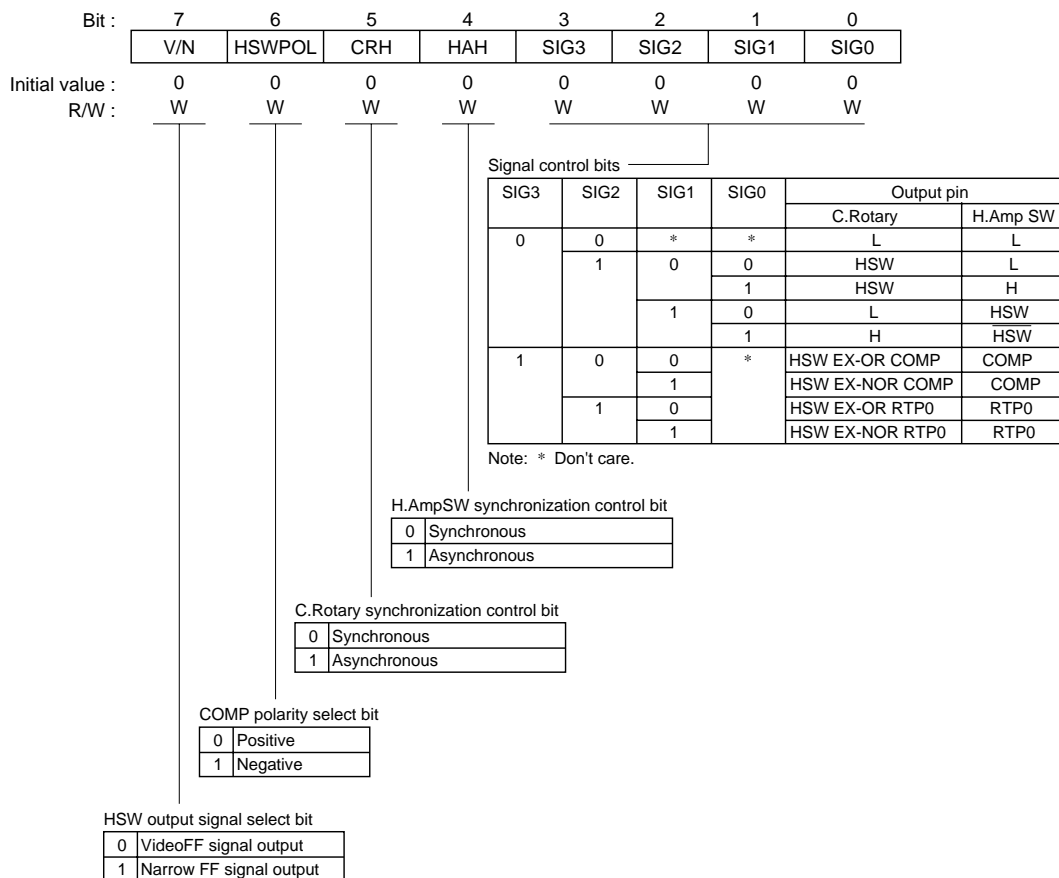
H'D06D: DFG Reference Register 2 DFCRB: HSW Timing Generator

| | | | | | | | | |
|-----------------|---|---|---|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | DFCRB4 | DFCRB3 | DFCRB2 | DFCRB1 | DFCRB0 |
| Initial value : | 1 | 1 | 1 | * | * | * | * | * |
| R/W : | — | — | — | W | W | W | W | W |

Note: * Undetermined

H'D06E: Special Effect Playback Control Register

CHCR: 4-head Special Effect Playback Circuit



H'D06F: Additional V Control Register ADDVR: Additional V Signal Generator

| | | | | | | | | |
|-----------------|---|---|---|------|-----|-----|------|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | HMSK | HiZ | CUT | VPON | POL |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | R/W | R/W | R/W | R/W | R/W |

| Additional V output control bits | | | |
|----------------------------------|------|-----|--|
| CUT | VPON | POL | Description |
| 0 | 0 | * | Low level |
| | 1 | 0 | Negative polarity (Figure 28.46) |
| | | 1 | Positive polarity (Figure 28.45) |
| 1 | * | 0 | Immediate level (high-impedance when HiZ bit = 1) |
| | | 1 | High level |

Note: * Don't care.

| High impedance bit | |
|--------------------|--|
| 0 | 3-level output from Vpulse pin |
| 1 | Vpulse pin is set as 3-state (H/L/HiZ) pin |

| OSCH mask bit | |
|---------------|----------------|
| 0 | OSCH added |
| 1 | OSCH not added |

H'D070: X-Value Data Register XDR: X-Value, TRK-Value

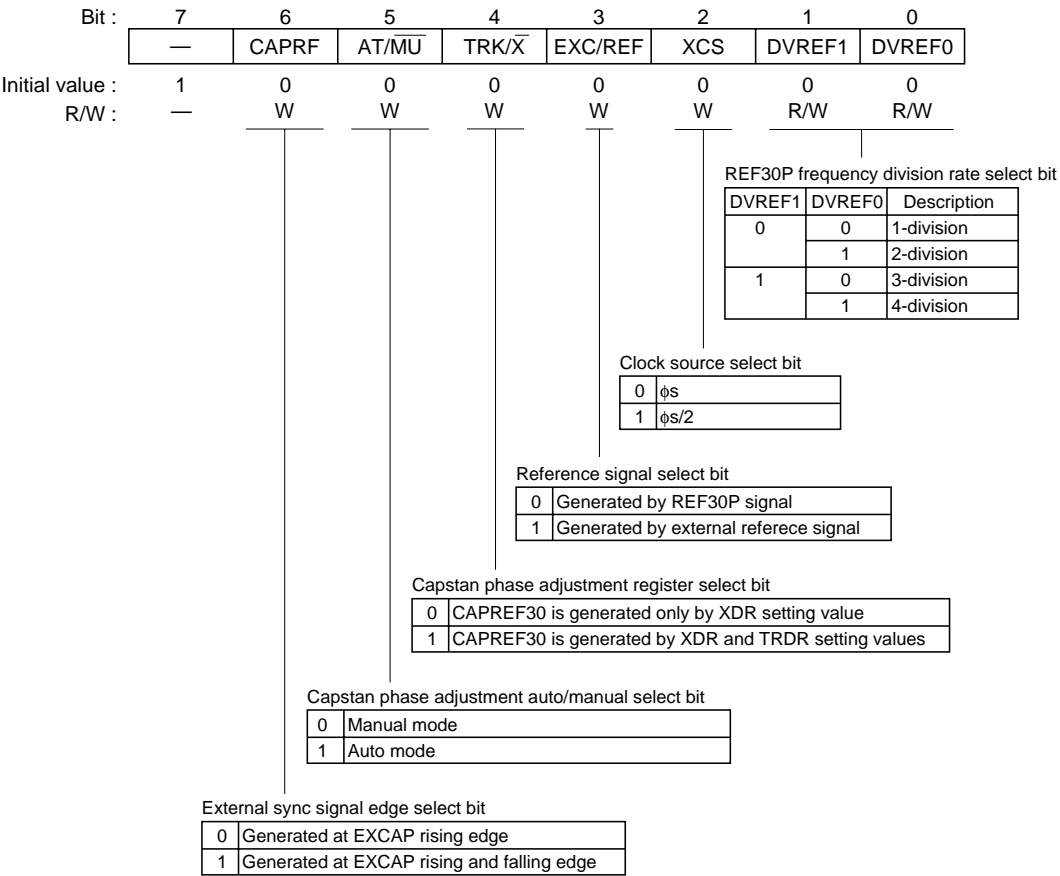
| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | XD11 | XD10 | XD9 | XD8 | XD7 | XD6 | XD5 | XD4 | XD3 | XD2 | XD1 | XD0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

H'D072: TRK-Value Data Register TRDR: X-Value, TRK-Value

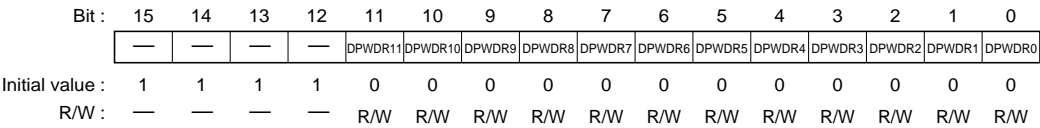
| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | TRD11 | TRD10 | TRD9 | TRD8 | TRD7 | TRD6 | TRD5 | TRD4 | TRD3 | TRD2 | TRD1 | TRD0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

H'D074: X-Value/TRK-Value Control Register

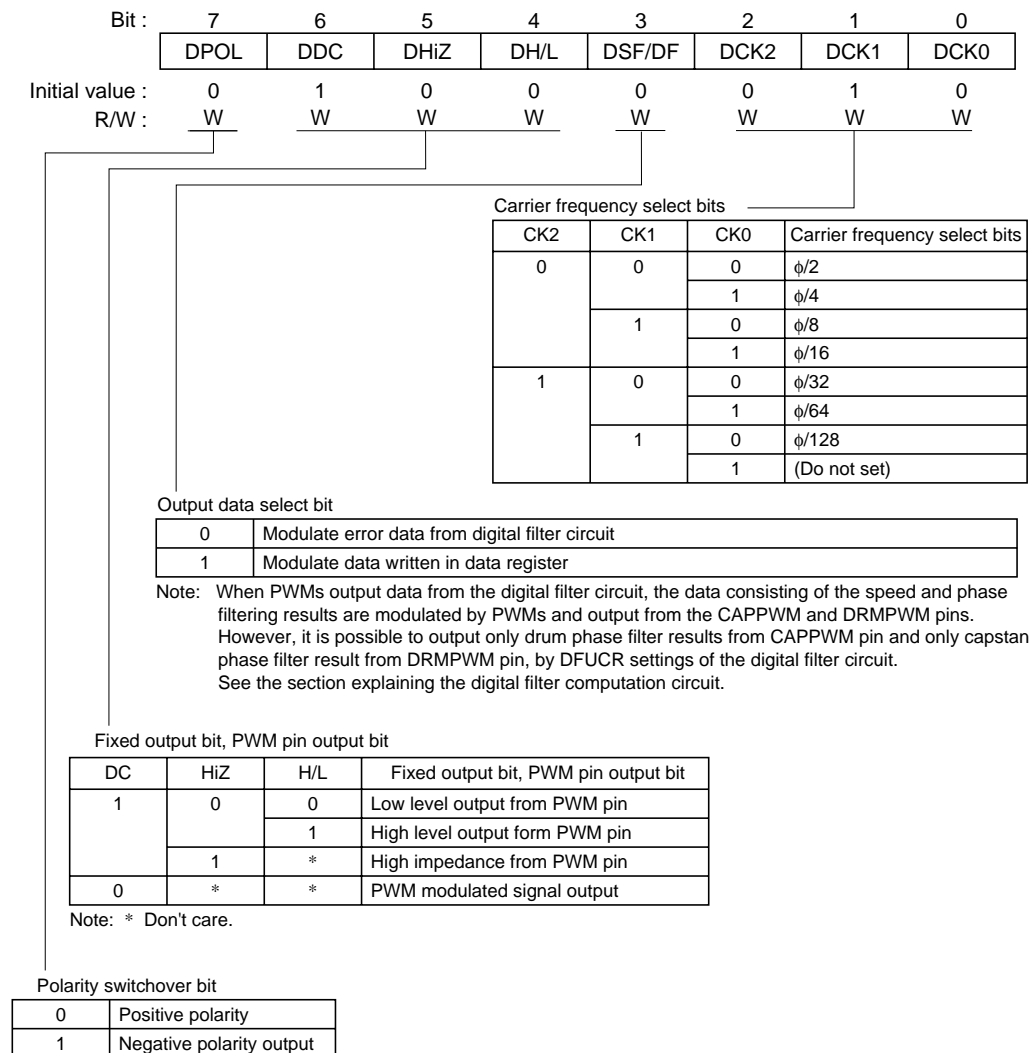
XTCR: X-Value, TRK-Value Adjustment Circuit



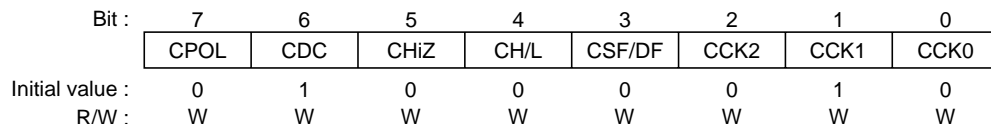
H'D078: Drum 12-bit PWM Data Register DPWDR: Drum 12-Bit PWM



H'D07A: Drum 12-Bit PWM Control Register DPWCR: Drum 12-Bit PWM



H'D07B: Capstan 12-Bit PWM Control Register CPWCR: Capstan 12-Bit PWM



H'D07C: Capstan 12-Bit PWM Data Register CPWDR: Capstan 12-Bit PWM

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|---------|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CPWDR11 | CPWDR10 | CPWDR9 | CPWDR8 | CPWDR7 | CPWDR6 | CPWDR5 | CPWDR4 | CPWDR3 | CPWDR2 | CPWDR1 | CPWDR0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

H'D080: CTL Control Register CTCR: CTL Circuit

| | | | | | | | | |
|-----------------|-------|------|------|------|-----|------|-------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | NT/PL | FSLC | FSLB | FSLA | CCS | LCTL | UNCTL | SLWM |
| Initial value : | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | R | W |

Mode select bit

| | |
|---|-------------|
| 0 | Normal mode |
| 1 | Slow mode |

CTL undetected bit

| | |
|---|------------|
| 0 | Detected |
| 1 | Undetected |

Long CTL bit

| | |
|---|---|
| 0 | Clock source (CCS) operates at the setting value |
| 1 | Clock source (CCS) operates for further 8-division after operating at the setting value |

Clock source select bit

| | |
|---|------------|
| 0 | ϕ_s |
| 1 | $\phi_s/2$ |

Operating frequency select bits

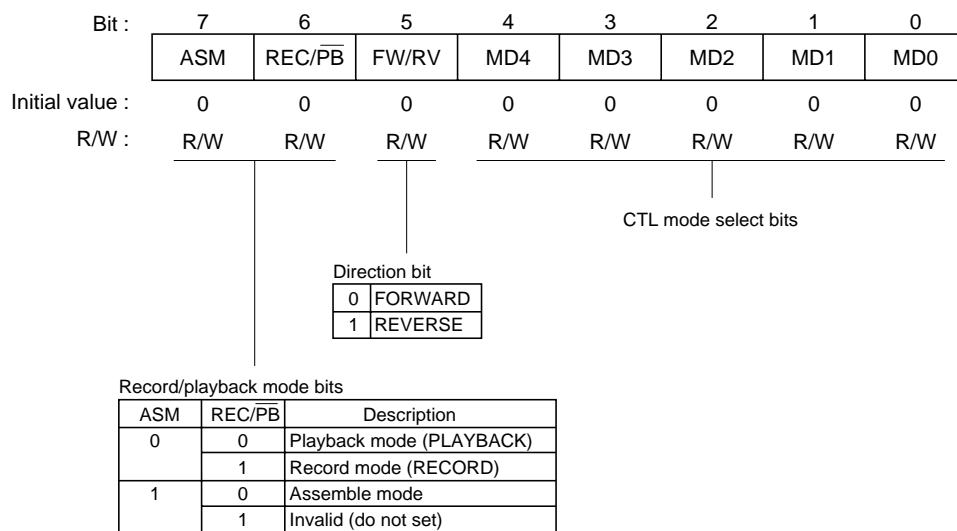
| FSLC | FSLB | FSLA | Description |
|------|------|------|-------------------------------|
| 0 | 0 | 0 | Reserved (do not set) |
| | | 1 | Reserved (do not set) |
| | 1 | 0 | fosc = 8 MHz |
| | | 1 | fosc = 10 MHz (Initial value) |
| 1 | * | * | Reserved (do not set) |

Note: * Don't care.

NTSC/PAL select bit

| | |
|---|-------------------------------|
| 0 | NTSC mode (frame rate: 30 Hz) |
| 1 | PAL mode (frame rate: 25 Hz) |

H'D081: CTL Mode Register CTLM: CTL Circuit



H'D082: REC-CTL Duty Data Register 1 RCDR1: CTL Circuit

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMT1B | CMT1A | CMT19 | CMT18 | CMT17 | CMT16 | CMT15 | CMT14 | CMT13 | CMT12 | CMT11 | CMT10 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

H'D084: REC-CTL Duty Data Register 2 RCDR2: CTL Circuit

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMT2B | CMT2A | CMT29 | CMT28 | CMT27 | CMT26 | CMT25 | CMT24 | CMT23 | CMT22 | CMT21 | CMT20 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

H'D086: REC-CTL Duty Data Register 3 RCDR3: CTL Circuit

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMT3B | CMT3A | CMT39 | CMT38 | CMT37 | CMT36 | CMT35 | CMT34 | CMT33 | CMT32 | CMT31 | CMT30 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

H'D088: REC-CTL Duty Data Register 4 RCDR4: CTL Circuit

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMT4B | CMT4A | CMT49 | CMT48 | CMT47 | CMT46 | CMT45 | CMT44 | CMT43 | CMT42 | CMT41 | CMT40 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

H'D08A: REC-CTL Duty Data Register 5 RCDR5: CTL Circuit

| | | | | | | | | | | | | | | | | |
|-----------------|----|----|----|----|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | CMT5B | CMT5A | CMT59 | CMT58 | CMT57 | CMT56 | CMT55 | CMT54 | CMT53 | CMT52 | CMT51 | CMT50 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R / W : | — | — | — | — | W | W | W | W | W | W | W | W | W | W | W | W |

H'D08C: Duty I/O Register DI/O: CTL Circuit

| | | | | | | | | |
|-----------------|-------|-------|-------|---|------|-----|--------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | VCTR2 | VCTR1 | VCTR0 | — | BPON | BPS | BPF | DI/O |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| R/W : | W | W | W | — | W | W | R/(W)* | R/W |

Duty I/O register

Bit pattern detection flag

| | |
|---|-------------------------------------|
| 0 | Bit pattern (8-bit) is not detected |
| 1 | Bit pattern (8-bit) is detected |

Bit pattern detection start bit

| | |
|---|------------------------------------|
| 0 | Normal status |
| 1 | Starts 8-bit bit pattern detection |

Bit pattern detection ON/OFF bit

| | |
|---|---------------------------|
| 0 | Bit pattern detection OFF |
| 1 | Bit pattern detection ON |

VISS interrupt setting bits

| VCTR2 | VCTR1 | VCTR0 | Number of 1-pulse for detection |
|-------|-------|-------|---------------------------------|
| 0 | 0 | 0 | 2 |
| | | 1 | 4 (SYNC mark) |
| | 1 | 0 | 6 |
| | | 1 | 8 (mark A, short) |
| 1 | 0 | 0 | 12 (mark A, long) |
| | | 1 | 16 |
| | 1 | 0 | 24 (mark B) |
| | | 1 | 32 |

Note: * Only 0 can be written.

H'D08D: Bit Pattern Register BTPR: CTL Circuit

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | LSP7 | LSP6 | LSP5 | LSP4 | LSP3 | LSP2 | LSP1 | LSP0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R*/W | R*/W | R*/W | R*/W | R*/W | R*/W | R*/W | R*/W |

Note: * Writes are disabled during bit pattern detection.

H'D090: Reference Frequency Register 1 RFD: Reference Signal Generator

| | | | | | | | | | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | REF15 | REF14 | REF13 | REF12 | REF11 | REF10 | REF9 | REF8 | REF7 | REF6 | REF5 | REF4 | REF3 | REF2 | REF1 | REF0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

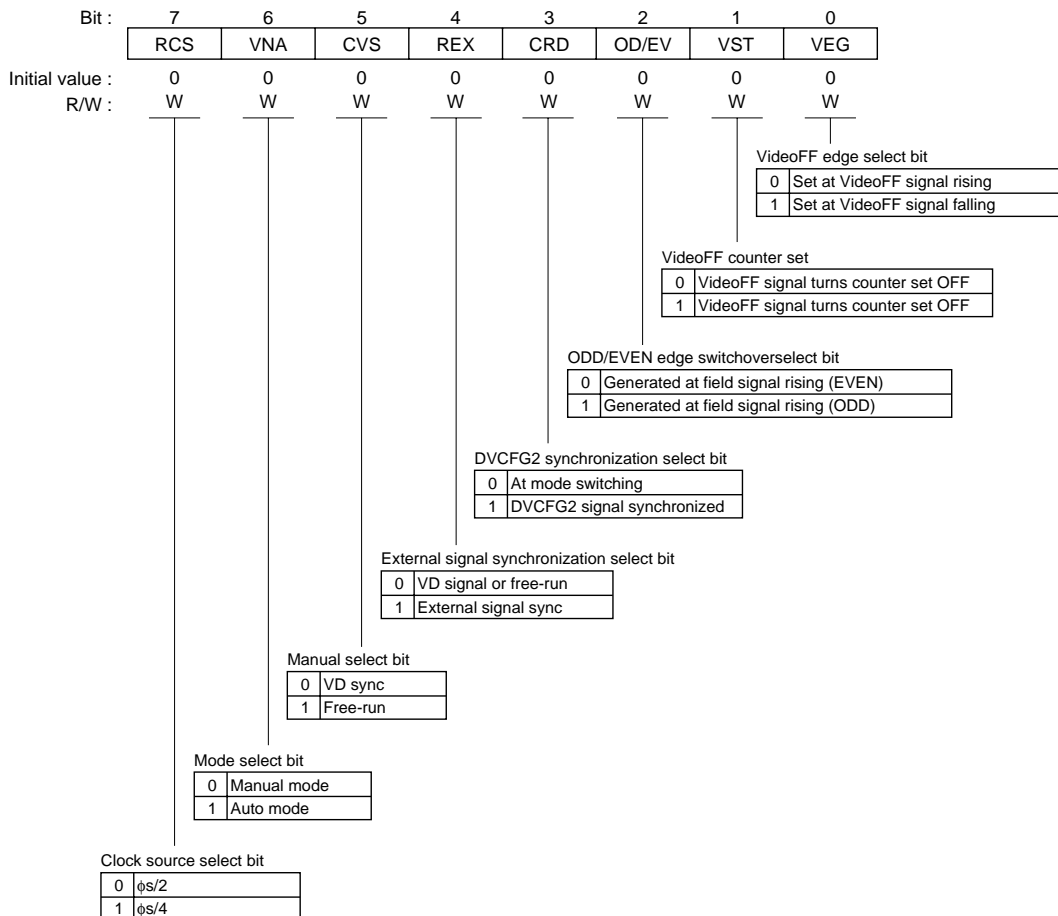
H'D092: Reference Frequency Register 2 CRF: Reference Signal Generator

| | | | | | | | | | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CRF15 | CRF14 | CRF13 | CRF12 | CRF11 | CRF10 | CRF9 | CRF8 | CRF7 | CRF6 | CRF5 | CRF4 | CRF3 | CRF2 | CRF1 | CRF0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

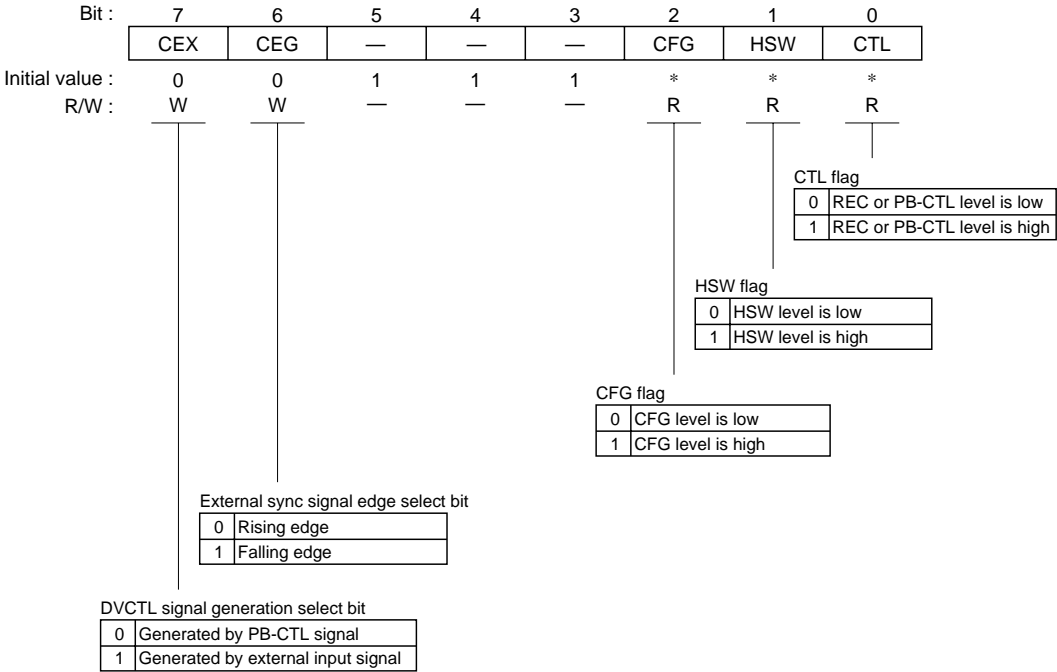
H'D094: REF30 Counter Register RFC: Reference Signal Generator

| | | | | | | | | | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|------|------|------|------|------|------|------|------|------|------|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | RFC15 | RFC14 | RFC13 | RFC12 | RFC11 | RFC10 | RFC9 | RFC8 | RFC7 | RFC6 | RFC5 | RFC4 | RFC3 | RFC2 | RFC1 | RFC0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 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 |

H'D096: Reference Frequency Mode Register RFM: Reference Signal Generator



H'D098: DVCTL Control Register CTVC: Frequency Divider

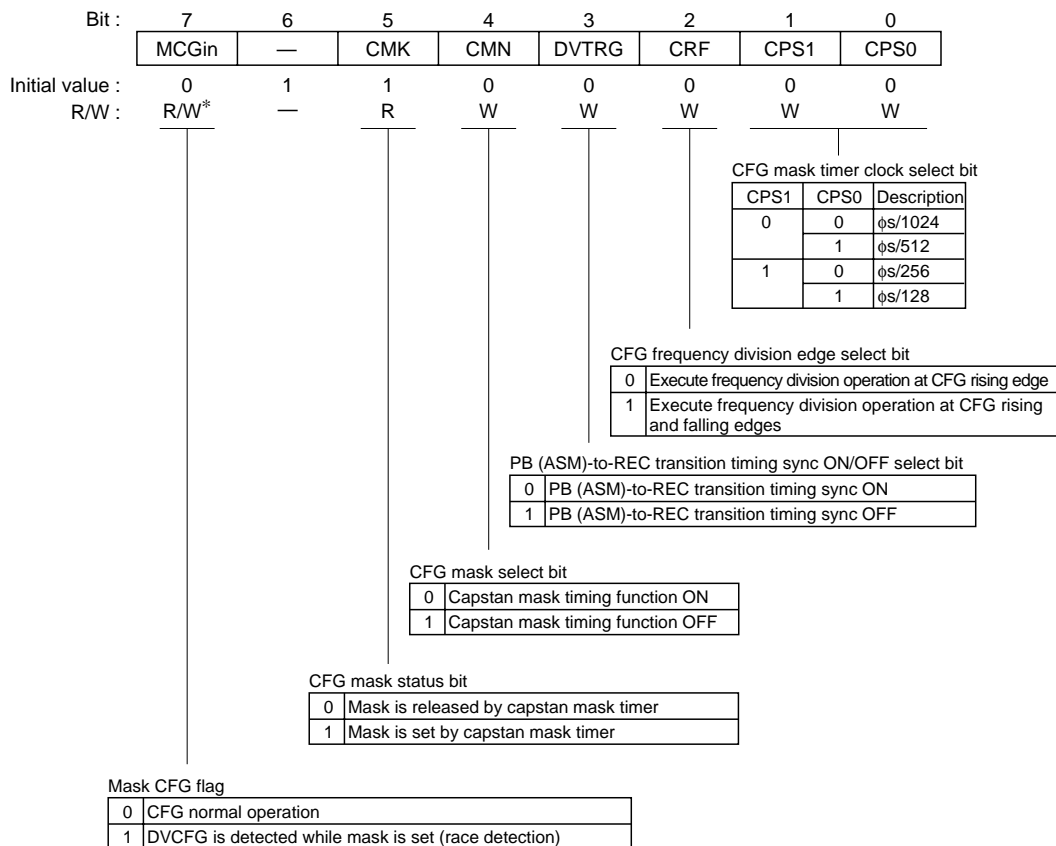


Note: * Undetermined

H'D099: CTL Frequency Division Register CTLR: Frequency Divider

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CTL7 | CTL6 | CTL5 | CTL4 | CTL3 | CTL2 | CTL1 | CTL0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

H'D09A: DVCFG Control Register CDVC: Frequency Divider



Note: * Only 0 can be written

H'D09B: CFG Frequency Division Register 1 CDIVR1: Frequency Divider

| | | | | | | | | |
|-----------------|---|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | CDV16 | CDV15 | CDV14 | CDV13 | CDV12 | CDV11 | CDV10 |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | W | W | W | W | W | W | W |

H'D09C: CFG Frequency Division Register 2 CDIVR2: Frequency Divider

| | | | | | | | | |
|-----------------|---|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | CDV26 | CDV25 | CDV24 | CDV23 | CDV22 | CDV21 | CDV20 |
| Initial value : | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | W | W | W | W | W | W | W |

H'D09D: DVCFG Mask Interval Register CTMR: Frequency Divider

| | | | | | | | | |
|-----------------|---|---|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | CPM5 | CPM4 | CPM3 | CPM2 | CPM1 | CPM0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | — | — | W | W | W | W | W | W |

H'D09E: FG Control Register FGCR: Frequency Divider

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | DRF |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| R/W : | — | — | — | — | — | — | — | W |

DFG edge select bit

| | |
|---|---------------------------------------|
| 0 | NCDFG signal rising edge is selected |
| 1 | NCDFG signal falling edge is selected |

H'D0A0: Servo Port Mode Register SPMR: Servo Port

| | | | | | | | | |
|-----------------|---------|---|---------|---------|-------|------|----------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CTLSTOP | — | CFGCOMP | EXCTLON | DPGSW | COMP | H.Amp.SW | C.Rot |
| Initial value : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | — | R/W | R/W | R/W | R/W | R/W | R/W |

C.Rotary pin function switch bit

| | |
|---|---|
| 0 | C.Rotary/PS0 pin functions as C.Rotary output pin |
| 1 | C.Rotary/PS0 pin functions as PS0 I/O pin |

H.AmpSW pin function switch bit

| | |
|---|---|
| 0 | H.AmpSW/PS1 pin functions as H.AmpSW output pin |
| 1 | H.AmpSW/PS1 pin functions as PS1 I/O pin |

COMP pin function switch bit

| | |
|---|--|
| 0 | COMP/PS2 pin functions as COMP input pin |
| 1 | COMP/PS2 pin functions as PS2 I/O pin |

DPG pin functions switch bit

| | |
|---|---|
| 0 | Separate input for drum control system input (DPG/PS3 pin functions as DPG input pin) |
| 1 | Weight input for drum control system input (DPG/PS3 pin functions as PS3 I/O pin) |

EXCTL pin function switch bit

| | |
|---|--|
| 0 | EXCTL/PS4 pin functions as EXCEL input pin |
| 1 | EXCTL/PS4 pin functions as PS4 I/O pin |

CFG input method switch bit

| | |
|---|--|
| 0 | Zero cross type comparator method for CFG signal input |
| 1 | Digital signal input method for CFG signal input |

CTLSTOP bit

| | |
|---|------------------------------|
| 0 | CTL circuit operates |
| 1 | CTL circuit does not operate |

H'D0A1: Servo Control Register SPCR: Servo Port

| | | | | | | | | |
|-----------------|---|---|---|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | SPCR4 | SPCR3 | SPCR2 | SPCR1 | SPCR0 |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | W | W | W | W | W |

| SPCRn | Description |
|-------|---------------------------------|
| 0 | PSn pin functions as input pin |
| 1 | PSn pin functions as output pin |

H'D0A2: Servo Data Register SPDR: Servo Port Controller

| | | | | | | | | |
|-----------------|---|---|---|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | SPDR4 | SPDR3 | SPDR2 | SPDR1 | SPDR0 |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | R/W | R/W | R/W | R/W | R/W |

H'D0A3: Servo Monitor Control Register SVMCR: Servo Port

| | | | | | | | | |
|-----------------|---|---|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | SVMCR5 | SVMCR4 | SVMCR3 | SVMCR2 | SVMCR1 | SVMCR0 |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

| SVMCR2 | SVMCR1 | SVMCR0 | Description |
|--------|--------|--------|---|
| 0 | 0 | 0 | REF30 signal is output from SV1 output pin |
| | | 1 | CAPREF30 signal is output from SV1 output pin |
| | 1 | 0 | CREF signal is output from SV1 output pin |
| | | 1 | CTLMONI signal is output from SV1 output pin |
| 1 | 0 | 0 | DVCFG signal is output from SV1 output pin |
| | | 1 | CFG signal is output from SV1 output pin |
| | 1 | 0 | DFG signal is output from SV1 output pin |
| | | 1 | DPG signal is output from SV1 output pin |

| SVMCR5 | SVMCR4 | SVMCR3 | Description |
|--------|--------|--------|---|
| 0 | 0 | 0 | REF30 signal is output from SV2 output pin |
| | | 1 | CAPREF30 signal is output from SV2 output pin |
| | 1 | 0 | CREF signal is output from SV2 output pin |
| | | 1 | CTLMONI signal is output from SV2 output pin |
| 1 | 0 | 0 | DVCFG signal is output from SV2 output pin |
| | | 1 | CFG signal is output from SV2 output pin |
| | 1 | 0 | DFG signal is output from SV2 output pin |
| | | 1 | DPG signal is output from SV2 output pin |

H'D0A4: CTL Gain Control Register CTLGR: Servo Port

Bit :
Initial value :
R/W :

7
—
1
—

6
—
1
—

5
CTLE/A
0
R/W

4
CTLFB
0
R/W

3
CTLGR3
0
R/W

2
CTLGR2
0
R/W

1
CTLGR1
0
R/W

0
CTLGR0
0
R/W

CTL amp gain setting bit

| CTLGR3 | CTLGR2 | CTLGR1 | CTLGR0 | CTL output gain |
|--------|--------|--------|--------|-----------------|
| 0 | 0 | 0 | 0 | 34.0 dB |
| | | | 1 | 36.5 dB |
| | | 1 | 0 | 39.0 dB |
| | | | 1 | 41.5 dB |
| | 1 | 0 | 0 | 44.0 dB |
| | | | 1 | 46.5 dB |
| | | 1 | 0 | 49.0 dB |
| | | | 1 | 51.5 dB |
| 1 | 0 | 0 | 0 | 54.0 dB |
| | | | 1 | 56.5 dB |
| | | 1 | 0 | 59.0 dB |
| | | | 1 | 61.5 dB |
| | 1 | 0 | 0 | 64.0 dB* |
| | | | 1 | 66.5 dB* |
| | | 1 | 0 | 69.0 dB* |
| | | | 1 | 71.5 dB* |

Note: * With a setting of 64.0dB or more, the CTLAMP is in a very sensitive status. When configuring the set board, be concerned about countermeasure against noise around the control head signal input port.
Also, thoroughly set the filter between the CTLAMP and CTLSMT.

CTL amp feedback SW bit

| | |
|---|-----------------|
| 0 | CTLFB SW is OFF |
| 1 | CTLFB SW is ON |

CTL select bit

| | |
|---|------------|
| 0 | AMP output |
| 1 | EXCTL |

H'D0B0: Vertical Sync Signal Threshold Value Register VTR: Sync Detector (Servo)

Bit :
Initial value :
R/W :

7
—
1
—

6
—
1
—

5
VTR5
0
W

4
VTR4
0
W

3
VTR3
0
W

2
VTR2
0
W

1
VTR1
0
W

0
VTR0
0
W

H'D0B1: Horizontal Sync Signal Threshold Value Register HTR: Sync Detector (Servo)

| | | | | | | | | |
|-----------------|---|---|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | HTR3 | HTR2 | HTR1 | HTR0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W |

H'D0B2: H Pulse Adjustment Start Time Setting Register HRTR: Sync Detector (Servo)

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | HRTR7 | HRTR6 | HRTR5 | HRTR4 | HRTR3 | HRTR2 | HRTR1 | HRTR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

H'D0B3: H Pulse Width Setting Register HPWR: Sync Detector (Servo)

| | | | | | | | | |
|-----------------|---|---|---|---|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | HPWR3 | HPWR2 | HPWR1 | HPWR0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W |

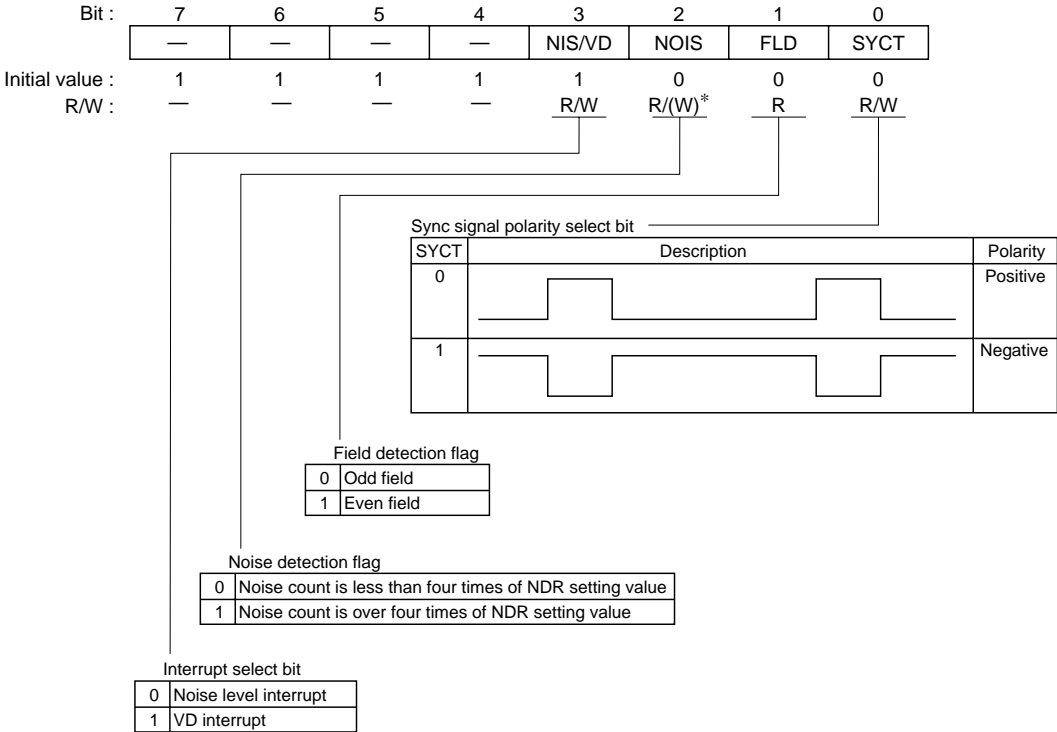
H'D0B4: Noise Detection Window Setting Register NWR: Sync Detector (Servo)

| | | | | | | | | |
|-----------------|---|---|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | NWR5 | NWR4 | NWR3 | NWR2 | NWR1 | NWR0 |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | W | W | W | W | W | W |

H'D0B5: Noise Detection Register NDR: Sync Detector (Servo)

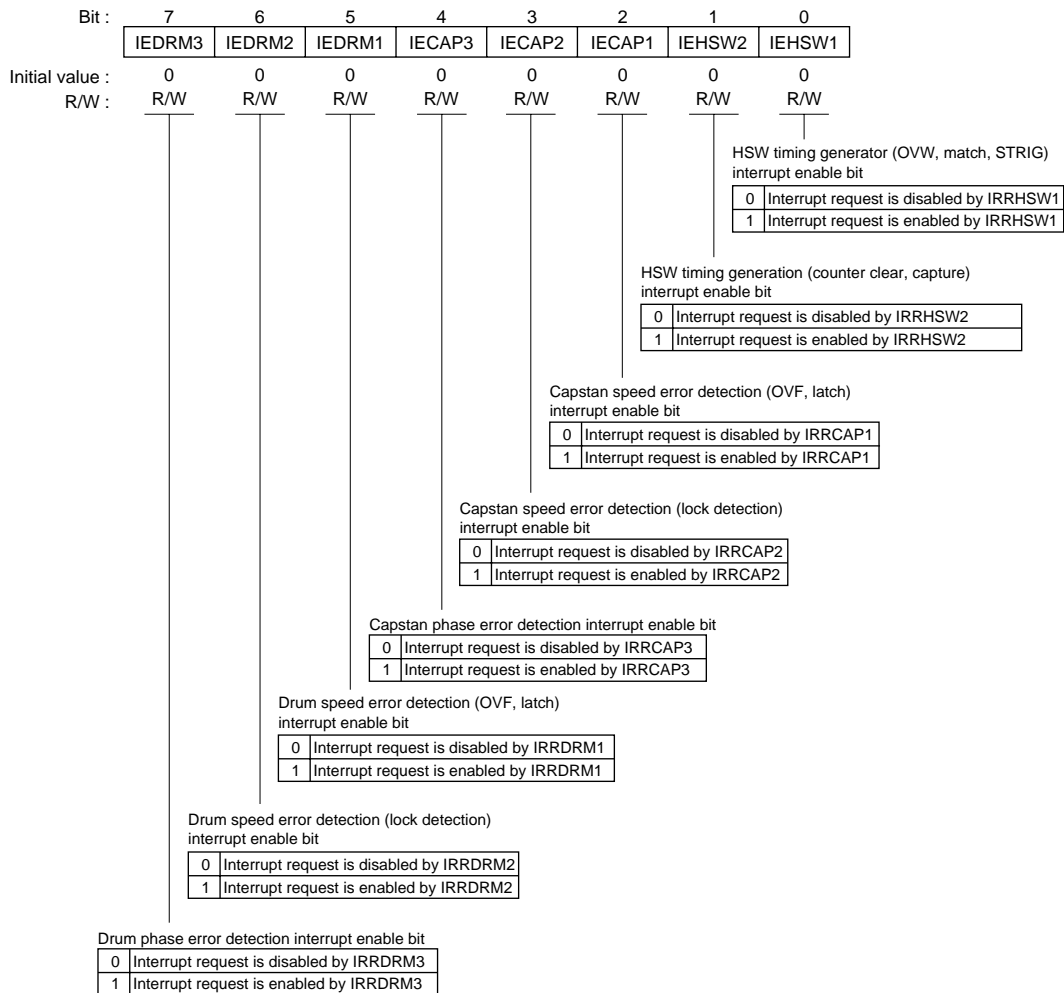
| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | NDR7 | NDR6 | NDR5 | NDR4 | NDR3 | NDR2 | NDR1 | NDR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

H'D0B6: Sync Signal Control Register SYNCR: Sync Detector (Servo)



Note: * Only 0 can be written.

H'D0B8: Servo Interrupt Enable Register 1 SIENR1: Servo Interrupt



H'D0B9: Servo Interrupt Enable Register 2 SIENR2: Servo Interrupt

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | IESNC | IECTL |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W : | — | — | — | — | — | — | R/W | R/W |

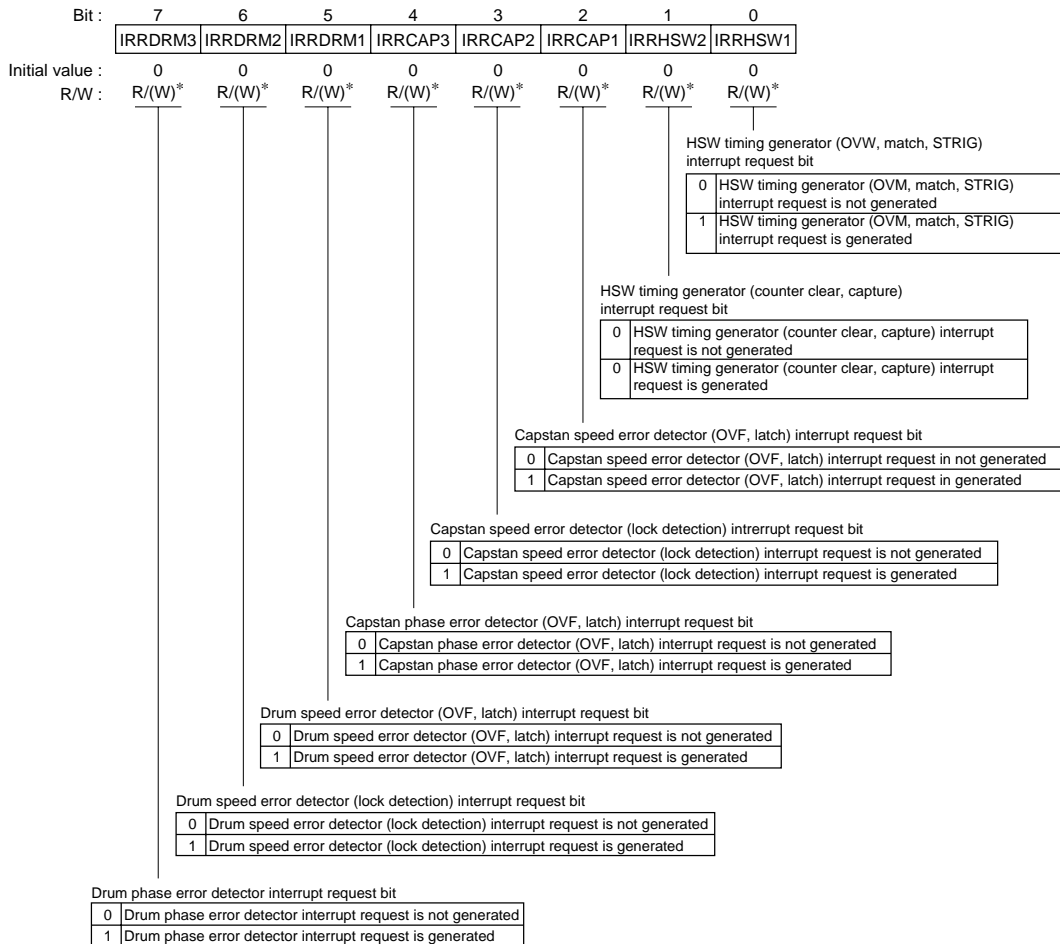
CTL interrupt enable bit

| | |
|---|---|
| 0 | Interrupt request is disabled by IRRCTL |
| 1 | Interrupt request is enabled by IRRCTL |

Vertical sync signal interrupt enable bit

| | |
|---|--|
| 0 | Interrupt (vertical sync signal interrupt) request is disabled by IRRSNC |
| 1 | Interrupt (vertical sync signal interrupt) request is enabled by IRRSNC |

H'D0BA: Servo Interrupt Request Register 1 SIRQR1: Servo Interrupt



Note: * Only 0 can be written to clear the flag.

H'D0BB: Servo Interrupt Request Register 2 SIRQR2: Servo Interrupt

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | IRRSNC | IRRCTL |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W : | — | — | — | — | — | — | R/(W)* | R/(W)* |

CTL interrupt request bit

| | |
|---|--|
| 0 | CTL interrupt request is not generated |
| 1 | CTL interrupt request is generated |

Vertical sync signal interrupt request bit

| | |
|---|---|
| 0 | Sync signal detector (VD, noise) interrupt request is not generated |
| 1 | Sync signal detector (VD, noise) interrupt request is generated |

Note: * Only 0 can be written to clear the flag.

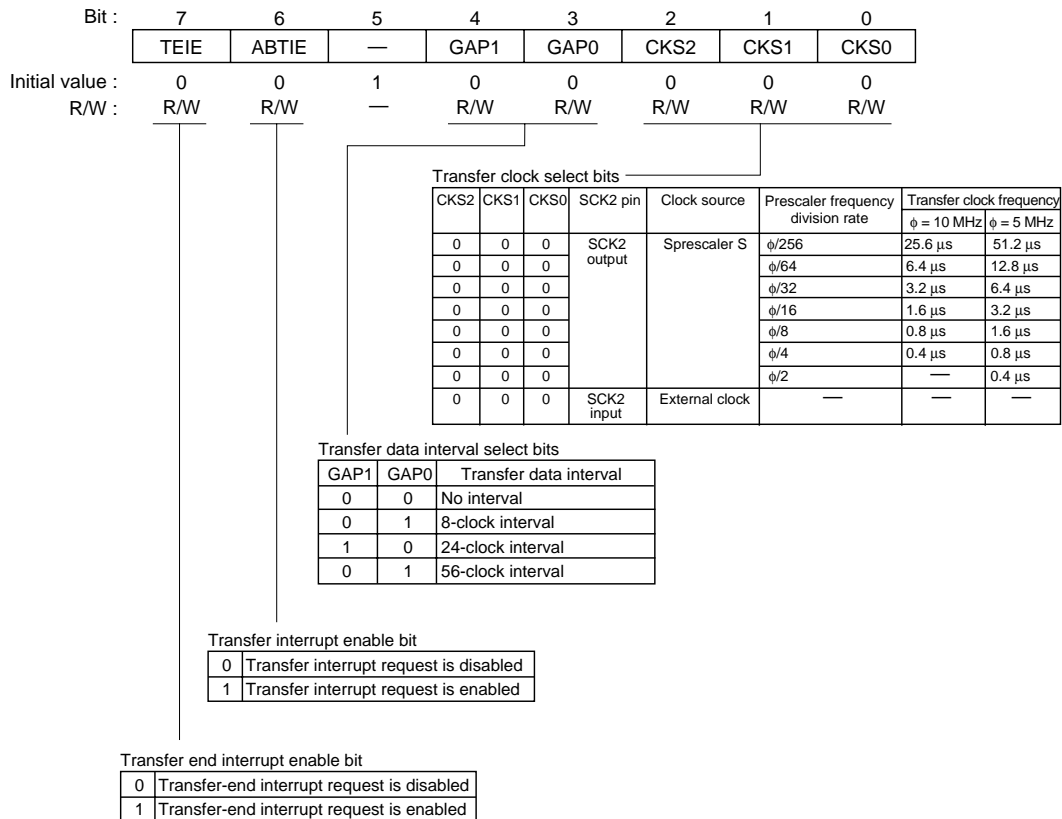
H'D0E0: Start Address Register STAR: 32-Byte Buffer SCI2

| | | | | | | | | |
|-----------------|---|---|---|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | STA4 | STA3 | STA2 | STA1 | STA0 |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | R/W | R/W | R/W | R/W | R/W |

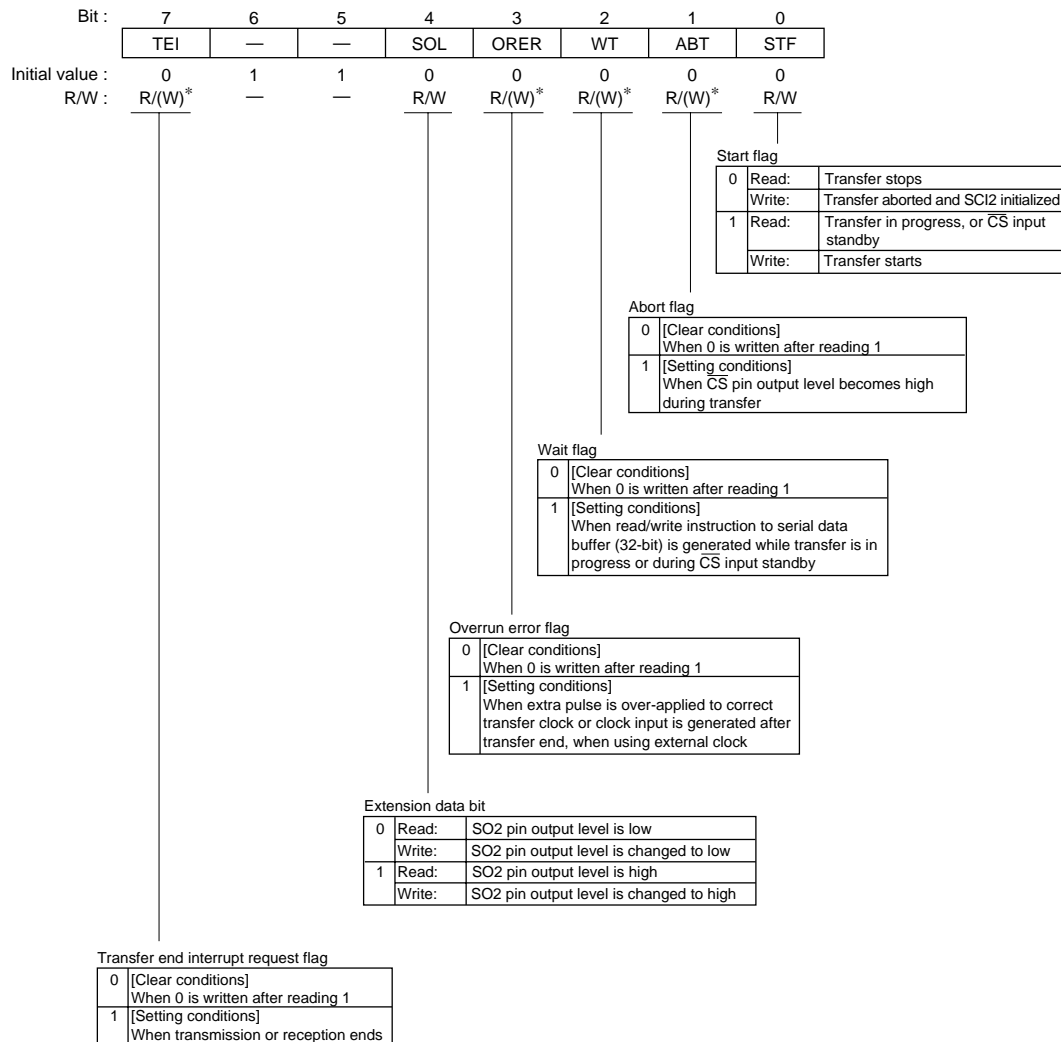
H'D0E1: End Address Register EDAR: 32-Byte Buffer SCI2

| | | | | | | | | |
|-----------------|---|---|---|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | EDA4 | EDA3 | EDA2 | EDA1 | EDA0 |
| Initial value : | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | R/W | R/W | R/W | R/W | R/W |

H'D0E2: Serial Control Register 2 SCR2: 32-Byte Buffer SCI2

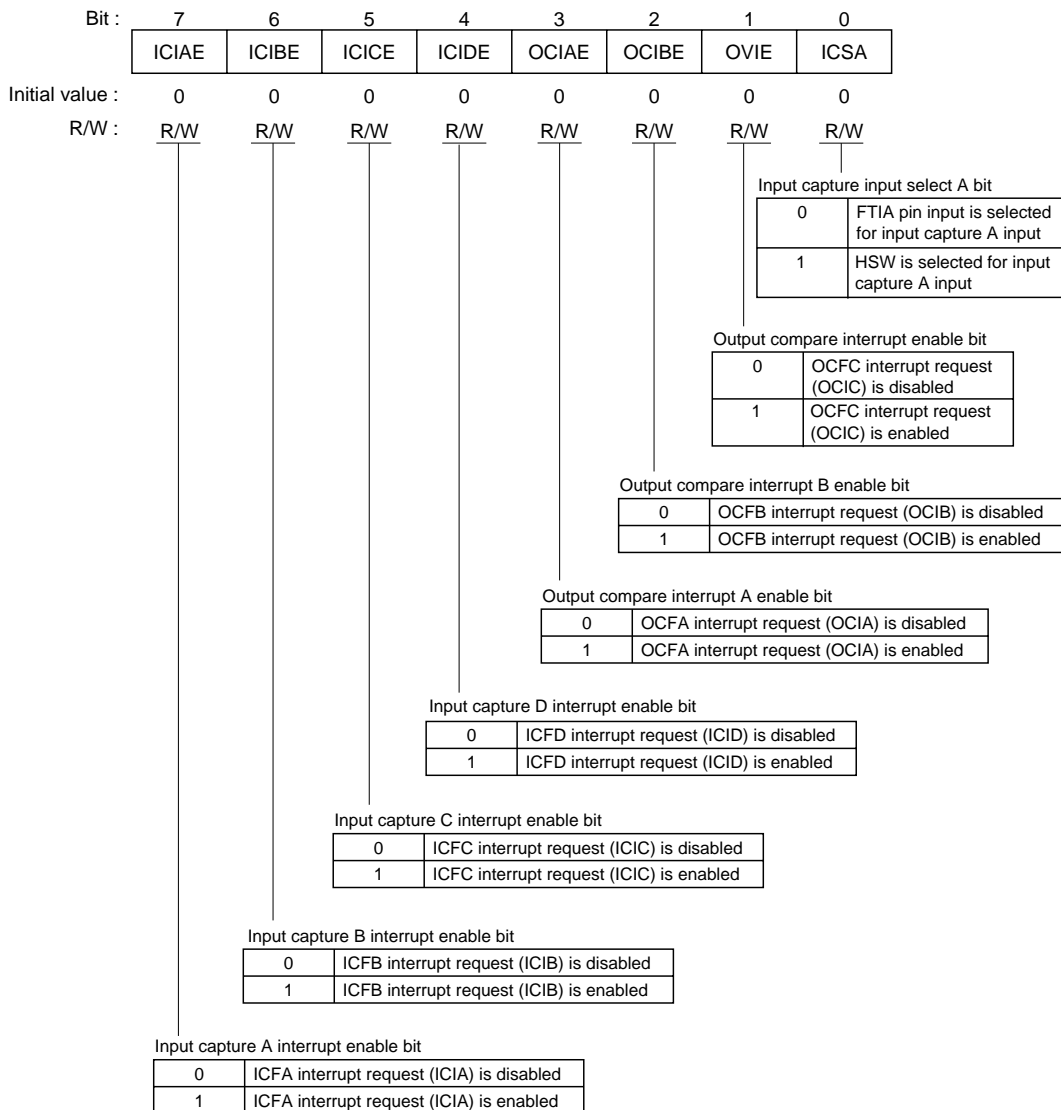


H'D0E3: Serial Control Status Register 2 SCSR2: 32-Byte Buffer SCI2

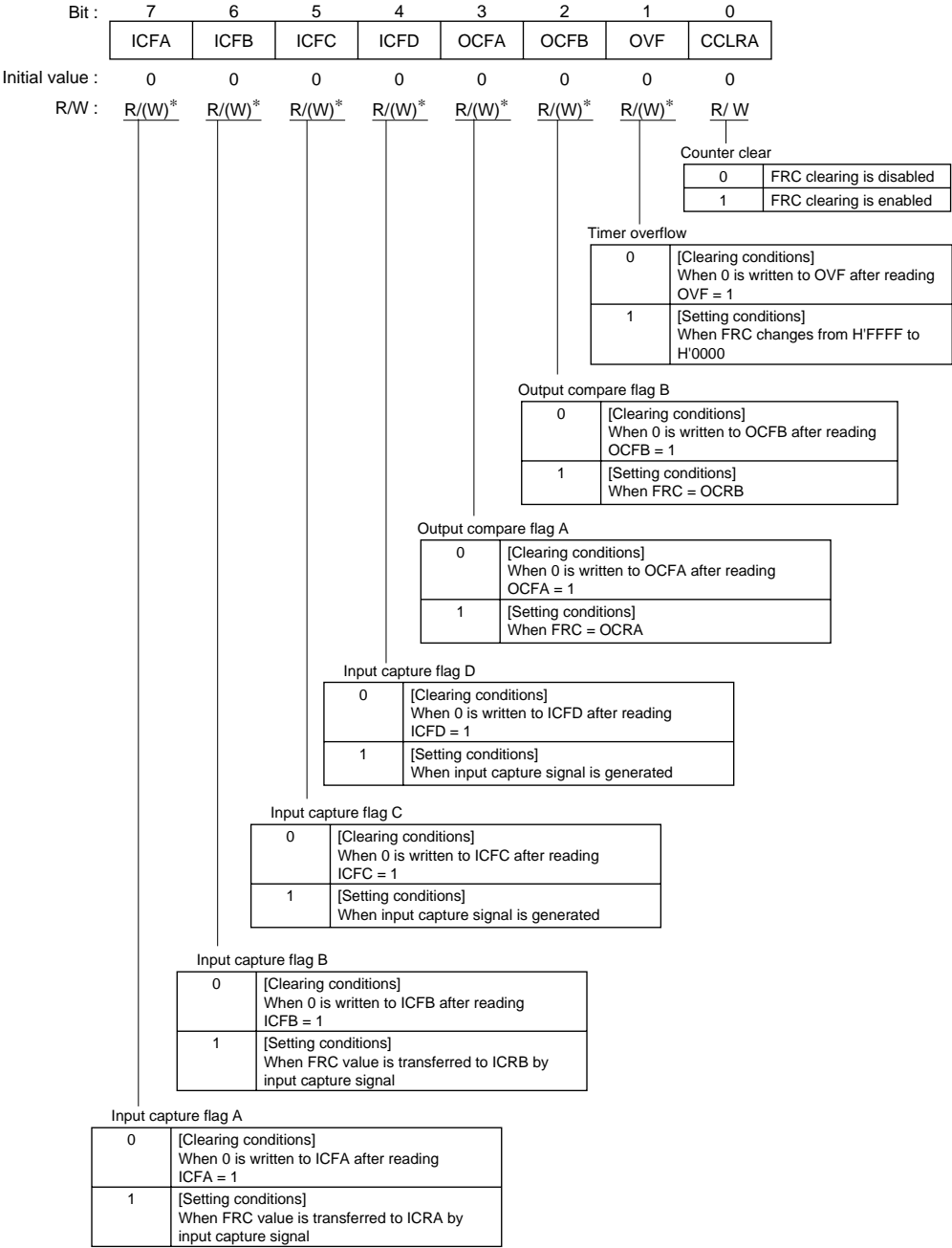


Note: * Only 0 can be written to clear the flag.

H'D100: Timer Interrupt Enable Register ITER: Timer X1



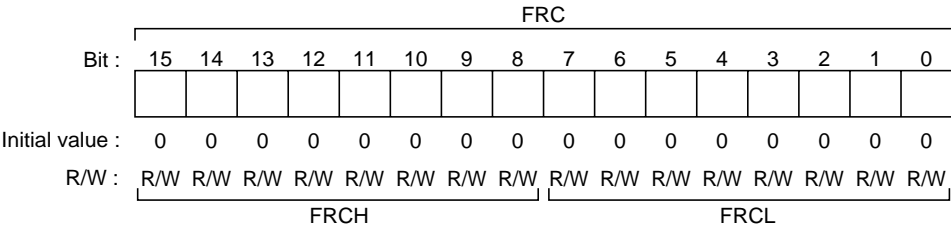
H'D101: Timer Control/Status Register X TCSR_X: Timer X1



Note: * Only 0 can be written to bits 7 to 1 to clear the flags.

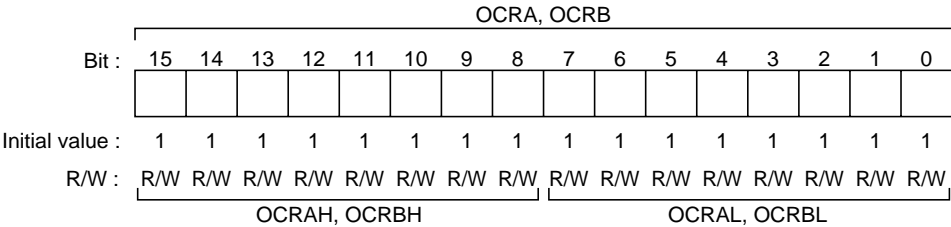
H'D102: Free Running Counter H FRCH: Timer X1

H'D103: Free Running Counter L FRCL: Timer X1

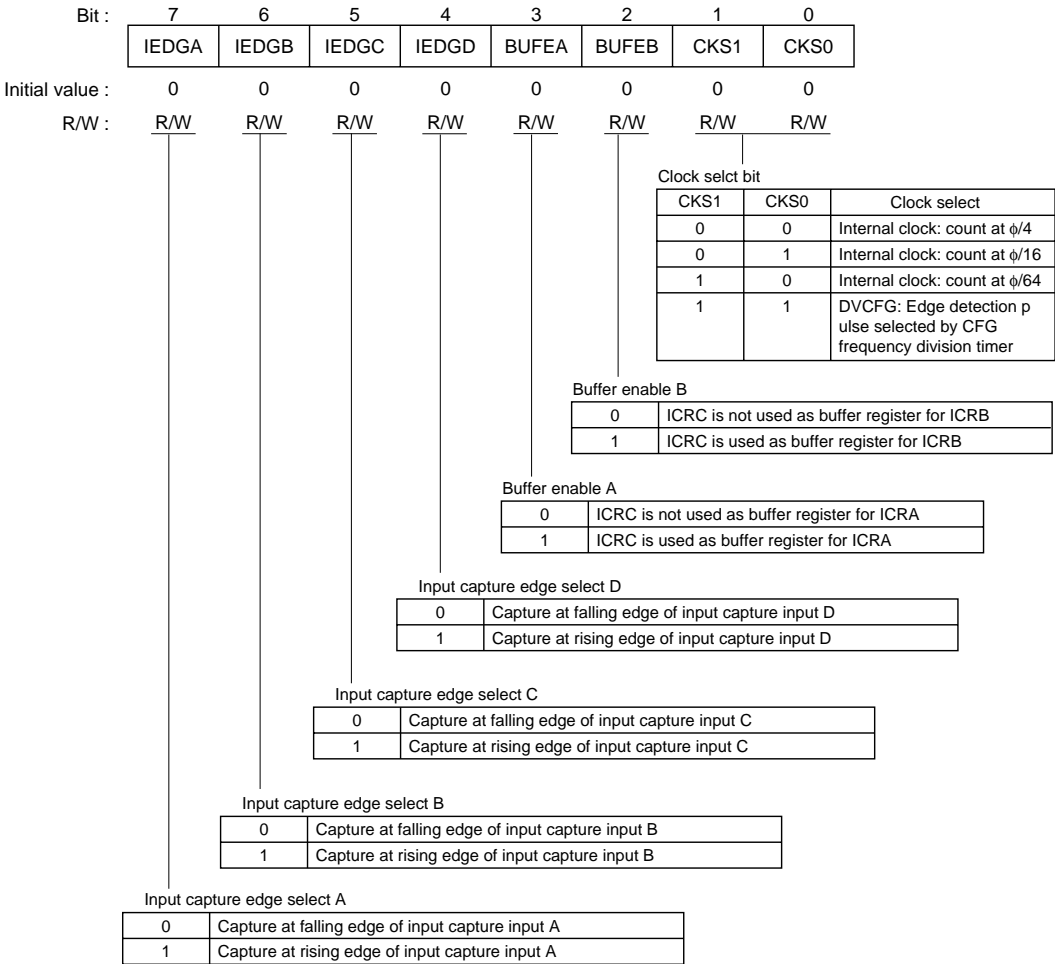


H'D104: Output Compare Register AH, BH OCRAH, OCRBH: Timer X1

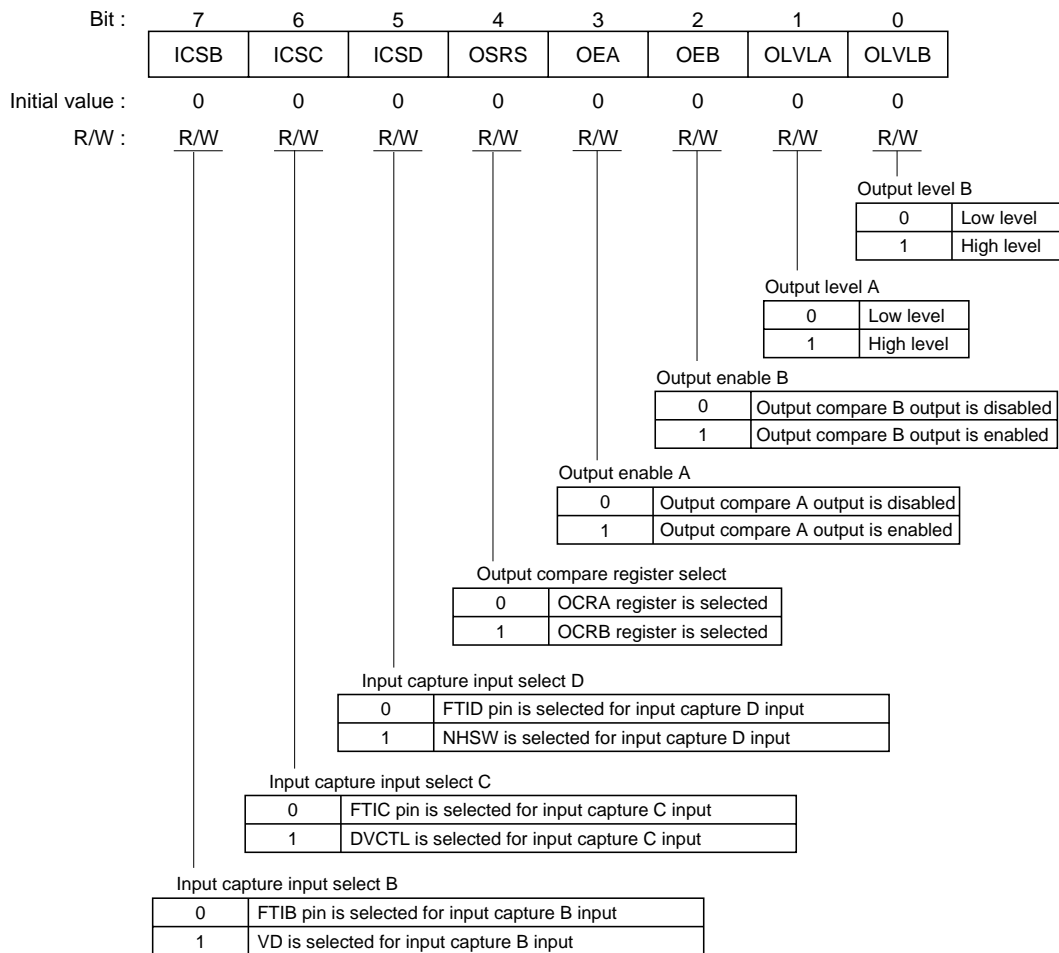
H'D105: Output Compare Register AL, BL OCRAL, OCRBL: Timer X1



H'D106: Timer Control Register X TCRX: Timer X1



H'D107: Timer Output Compare Control Register TOCR: Timer X1



H'D108: Input Capture Register AH ICRAH: Timer X1

H'D109: Input Capture Register AL ICRAL: Timer X1

H'D10A: Input Capture Register BH ICRBH: Timer X1

H'D10B: Input Capture Register BL ICRBL: Timer X1

H'D10C: Input Capture Register CH ICRCH: Timer X1

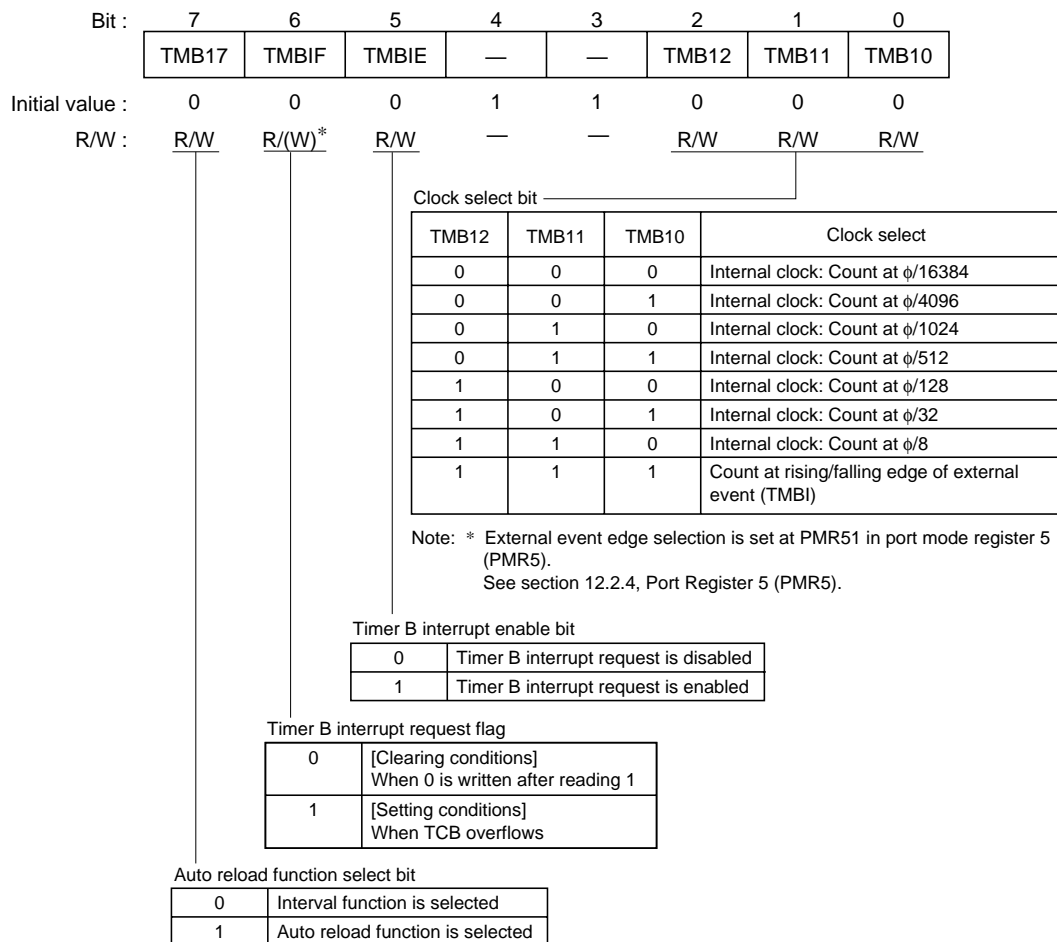
H'D10D: Input Capture Register CL ICRCL: Timer X1

H'D10E: Input Capture Register DH ICRDH: Timer X1

H'D10F: Input Capture Register DL ICRDL: Timer X1

| ICRA, ICRB, ICRCH, ICRD | | | | | | | | | | | | | | | | |
|----------------------------|----|----|----|----|----|----|---|----------------------------|---|---|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R |
| ICRAH, ICRBH, ICRCH, ICRDH | | | | | | | | ICRAL, ICRBL, ICRCL, ICRDL | | | | | | | | |

H'D110: Timer Mode Register B TMB: Timer B



Note: * Only 0 can be written to clear the flag.

H'D111: Timer Counter B TCB: Timer B

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TCB17 | TCB16 | TCB15 | TCB14 | TCB13 | TCB12 | TCB11 | TCB10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R |

H'D111: Timer Load RegisterB TLB: TimerB

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TLB17 | TLB16 | TLB15 | TLB14 | TLB13 | TLB12 | TLB11 | TLB10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

H'D112: Timer L Mode Register LMR: Timer L

| | | | | | | | | |
|-----------------|--------|------|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | LMIF | LMIE | — | — | IMR3 | IMR2 | IMR1 | IMR0 |
| Initial value : | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/W | — | — | R/W | R/W | R/W | R/W |

Clock select bit

| R2 | LMR1 | LMR0 | Clock select |
|----|------|------|---|
| 0 | 0 | 0 | Count at rising edge of PB and REC-CTL |
| | | 1 | Count at falling edge of PB and REC-CTL |
| | 1 | * | Count DVCFG2 |
| 1 | 0 | * | Internal clock: Count at $\phi/128$ |
| | 1 | * | Internal clock: Count at $\phi/64$ |

Note: * Don't care.

Up/down count control

| | |
|---|--------------------|
| 0 | Up count control |
| 1 | Down count control |

Timer L interrupt enable bit

| | |
|---|---------------------------------------|
| 0 | Timer L interrupt request is disabled |
| 1 | Timer L interrupt request is enabled |

Timer L interrupt request flag

| | |
|---|--|
| 0 | [Clearing conditions] When 0 is written after reading 1 |
| 1 | [Setting conditions] When LTC overflow, underflow or compare match clear occurs |

Note: * Only 0 can be written to clear the flag.

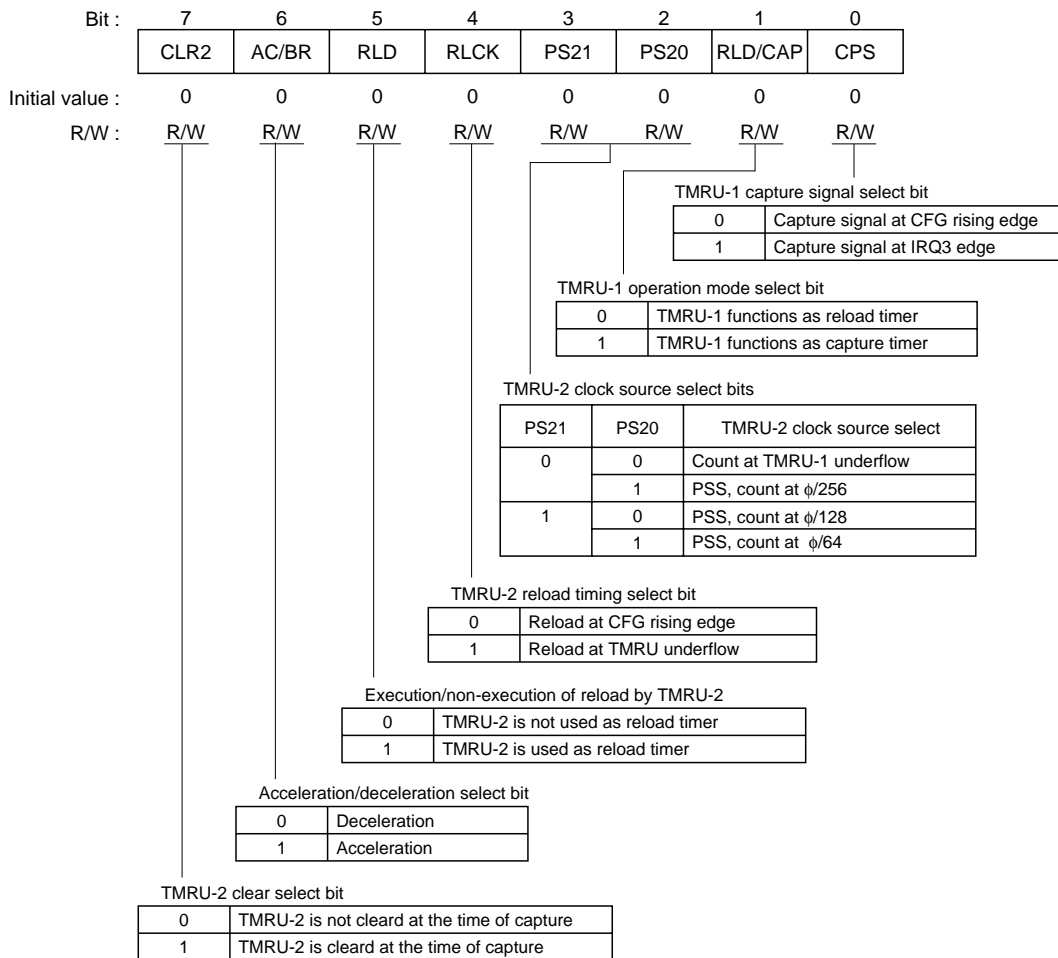
H'D113: Linear Time Counter LTC: Timer L

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | LTC7 | LTC6 | LTC5 | LTC4 | LTC3 | LTC2 | LTC1 | LTC0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R |

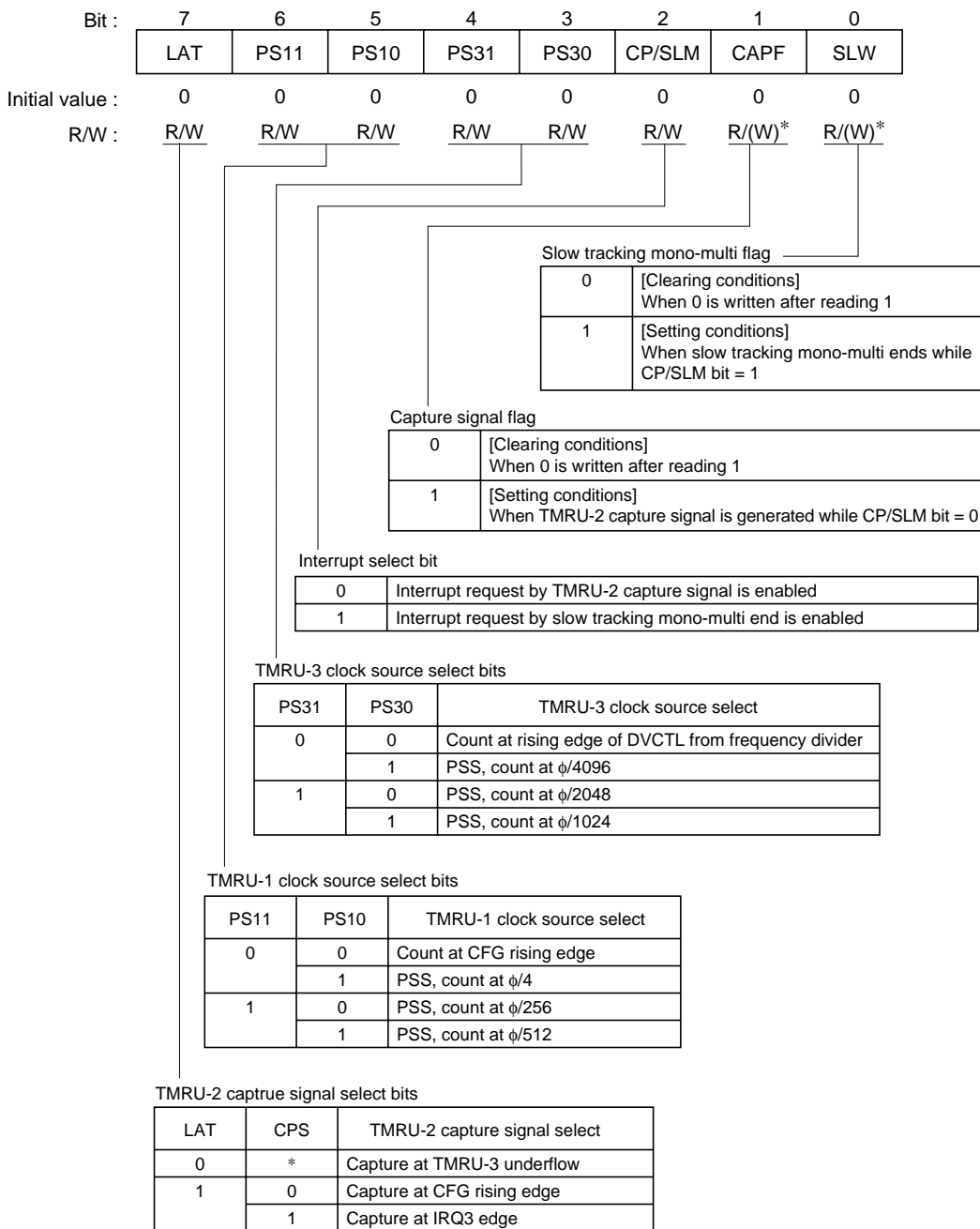
H'D113: Reload/Compare Match Register RCR: Timer L

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | RCR7 | RCR6 | RCR5 | RCR4 | RCR3 | RCR2 | RCR1 | RCR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

H'D118: Timer R Mode Register 1 TMRM1: Timer R



H'D119: Timer R Mode Register TMRM2: Timer R



Note: * Don't care.

Note: * Only 0 can be written to clear the flag.

H'D11A: Timer R Capture Register 1 TMRCP1: Time R

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMRC17 | TMRC16 | TMRC15 | TMRC14 | TMRC13 | TMRC12 | TMRC11 | TMRC10 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R | R | R | R | R | R | R | R |

H'D11B: Timer R Capture Register 2 TMRCP2: Time R

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMRC27 | TMRC26 | TMRC25 | TMRC24 | TMRC23 | TMRC22 | TMRC21 | TMRC20 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R | R | R | R | R | R | R | R |

H'D11C: Timer R Load Register 1 TMRL1: Timer R

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMR17 | TMR16 | TMR15 | TMR14 | TMR13 | TMR12 | TMR11 | TMR10 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W |

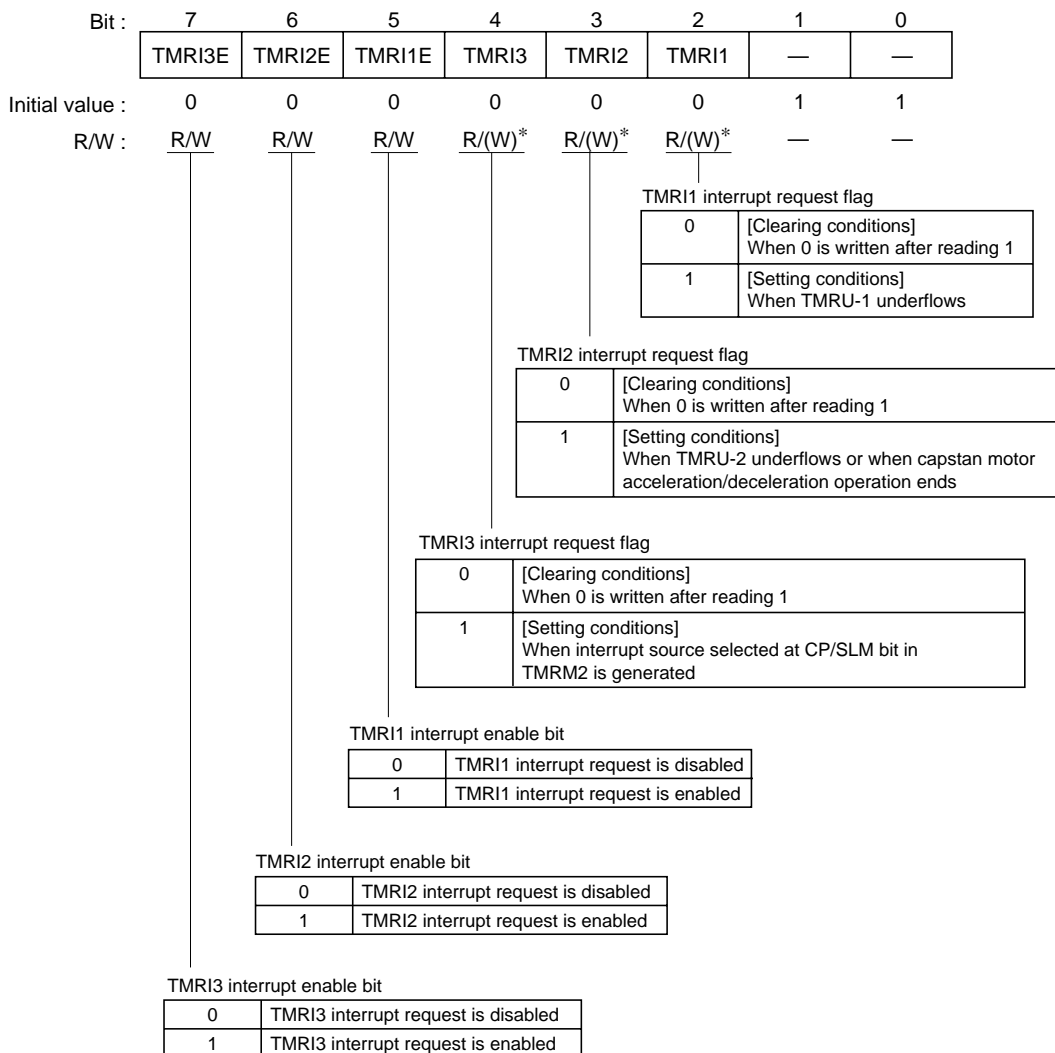
H'D11D: Timer R Load Register 2 TMRL2: Timer R

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMR27 | TMR26 | TMR25 | TMR24 | TMR23 | TMR22 | TMR21 | TMR20 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W |

H'D11E: Timer R Load Register 3 TMRL3: Timer R

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMR37 | TMR36 | TMR35 | TMR34 | TMR33 | TMR32 | TMR31 | TMR30 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W |

H'D11F: Timer R Control/Status Register TMRCs: Timer R



Note: * Only 0 can be written to clear the flag.

H'D120: PWM Data Register L PWDRL: 14-Bit PWM

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PWDRL7 | PWDRL6 | PWDRL5 | PWDRL4 | PWDRL3 | PWDRL2 | PWDRL1 | PWDRL0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

H'D121: PWM Data Register U PWDRU: 14-Bit PWM

| | | | | | | | | |
|-----------------|---|---|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | PWDRU5 | PWDRU4 | PWDRU3 | PWDRU2 | PWDRU1 | PWDRU0 |
| Initial value : | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | W | W | W | W | W | W |

H'D122: PWM Control Register PWCR: 14-Bit PWM

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | PWCR0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| R/W : | — | — | — | — | — | — | — | R/W |

Clock select bit

| | |
|---|---|
| 0 | Input clock is $\phi/2$ ($t\phi = 2/\phi$) Generate PWM waveform with conversion frequency of $16384/\phi$ and minimum pulse width of $1/\phi$ |
| 1 | Input clock is $\phi/4$ ($t\phi = 4/\phi$) Generate PWM waveform with conversion frequency of $32768/\phi$ and minimum pulse width of $2/\phi$ |

Note: * $t\phi$: PWM input clock frequency**H'D126: 8-Bit PWM Data Register 0 PWR0: 8-Bit PWM**

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PW07 | PW06 | PW05 | PW04 | PW03 | PW02 | PW01 | PW00 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

H'D127: 8-Bit PWM Data Register 1 PWR1: 8-Bit PWM

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PW17 | PW16 | PW15 | PW14 | PW13 | PW12 | PW11 | PW10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

H'D128: 8-Bit PWM Data Register 2 PWR2: 8-Bit PWM

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PW27 | PW26 | PW25 | PW24 | PW23 | PW22 | PW21 | PW20 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

H'D129: 8-Bit PWM Data Register 3 PWR3: 8-Bit PWM

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PW37 | PW36 | PW35 | PW34 | PW33 | PW32 | PW31 | PW30 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

H'D12A: 8-Bit PWM Control Register PW8CR: 8-Bit PWM

| | | | | | | | | |
|-----------------|---|---|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PWC3 | PWC2 | PWC1 | PWC0 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | R/W | R/W | R/W | R/W |

Output polarity select bits —

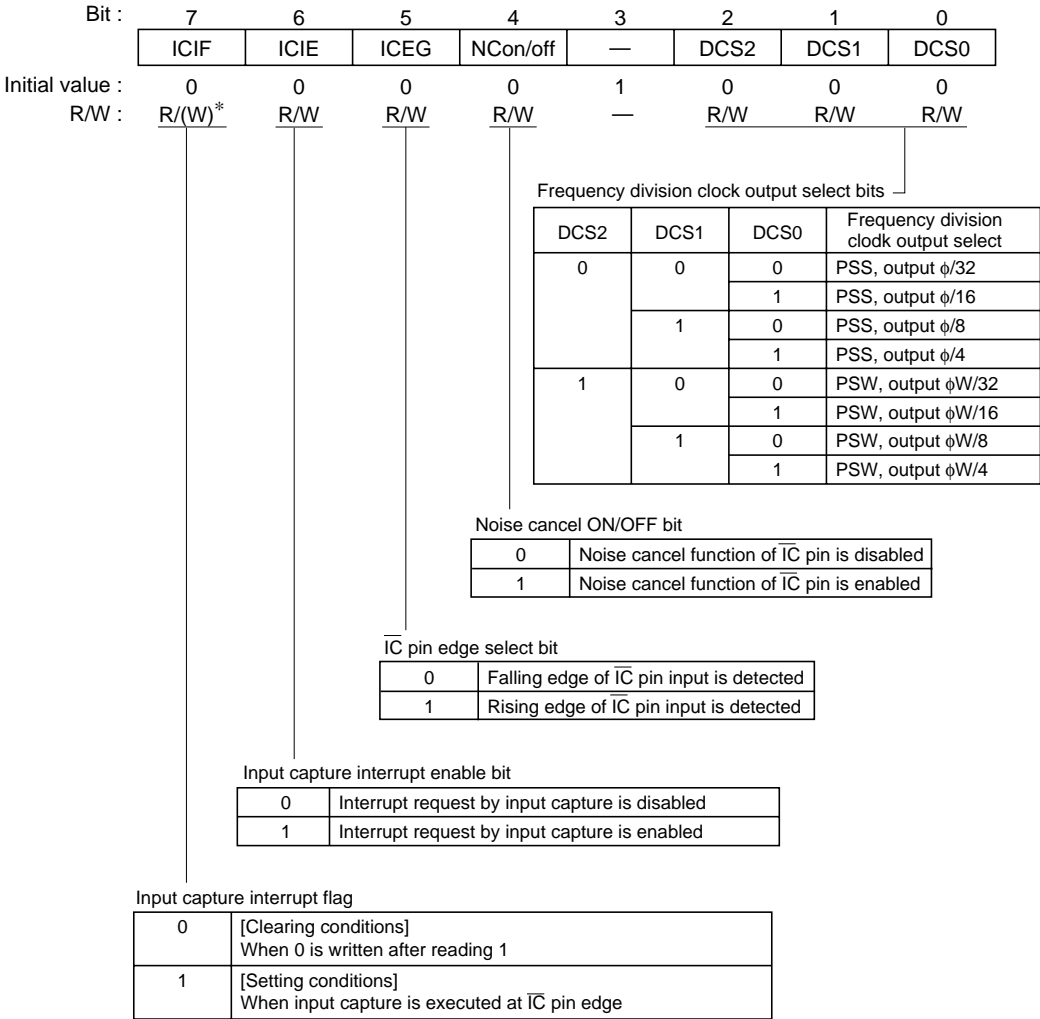
| | |
|---|-------------------|
| 0 | Positive polarity |
| 1 | Negative polarity |

(n = 3 to 0)

H'D12C: Input Capture Register 1 ICR1: PSU

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICR17 | ICR16 | ICR15 | ICR14 | ICR13 | ICR12 | ICR11 | ICR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R |

H'D12D: Prescaler Unit Control/Status Register PCSR: PSU



Note: * Only 0 can be written to clear the flag.

H'D130: Software Trigger A/D Result Register H ADRH: A/D Converter

H'D131: Software Trigger A/D Result Register L ADRL: A/D Converter

| ADRH | | | | | | | | | | ADRL | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ADR9 | ADR8 | ADR7 | ADR6 | ADR5 | ADR4 | ADR3 | ADR2 | ADR1 | ADR0 | — | — | — | — | — | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R | R | R | — | — | — | — | — | — |

H'D132: Hardware Trigger A/D Result Register H AHRH: A/D Converter

H'D133: Hardware Trigger A/D Result Register L AHRL: A/D Converter

| AHRH | | | | | | | | AHRL | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|------|------|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | AHR9 | AHR8 | AHR7 | AHR6 | AHR5 | AHR4 | AHR3 | AHR2 | AHR1 | AHR0 | — | — | — | — | — | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R | R | R | — | — | — | — | — | — |

H'D134: A/D Control Register ADCR: A/D Converter

| | | | | | | | | |
|-----------------|-----|---|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CK | — | HCH1 | HCH0 | SCH3 | SCH2 | SCH1 | SCH0 |
| Initial value : | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | — | R/W | R/W | R/W | R/W | R/W | R/W |

Software channel select bits

| SCH3 | SCH2 | SCH1 | SCH0 | Analog input channel |
|------|------|------|------|---|
| 0 | 0 | 0 | 0 | AN0 |
| | | | 1 | AN1 |
| | | 1 | 0 | AN2 |
| | | | 1 | AN3 |
| | 1 | 0 | 0 | AN4 |
| | | | 1 | AN5 |
| | | 1 | 0 | AN6 |
| | | | 1 | AN7 |
| 1 | 0 | 0 | 0 | AN8 |
| | | | 1 | AN9 |
| | | 1 | 0 | ANA |
| | | | 1 | ANB |
| | 1 | * | * | Software-triggered conversion channel is not selected |

- Notes: 1. If conversion is started by software when SCH3 to SCH0 are set to 11xx, the conversion result is undetermined. Hardware- or external-triggered conversion, however, will be performed on the channel selected by HCH1 and HCH0.
2. * Don't care.

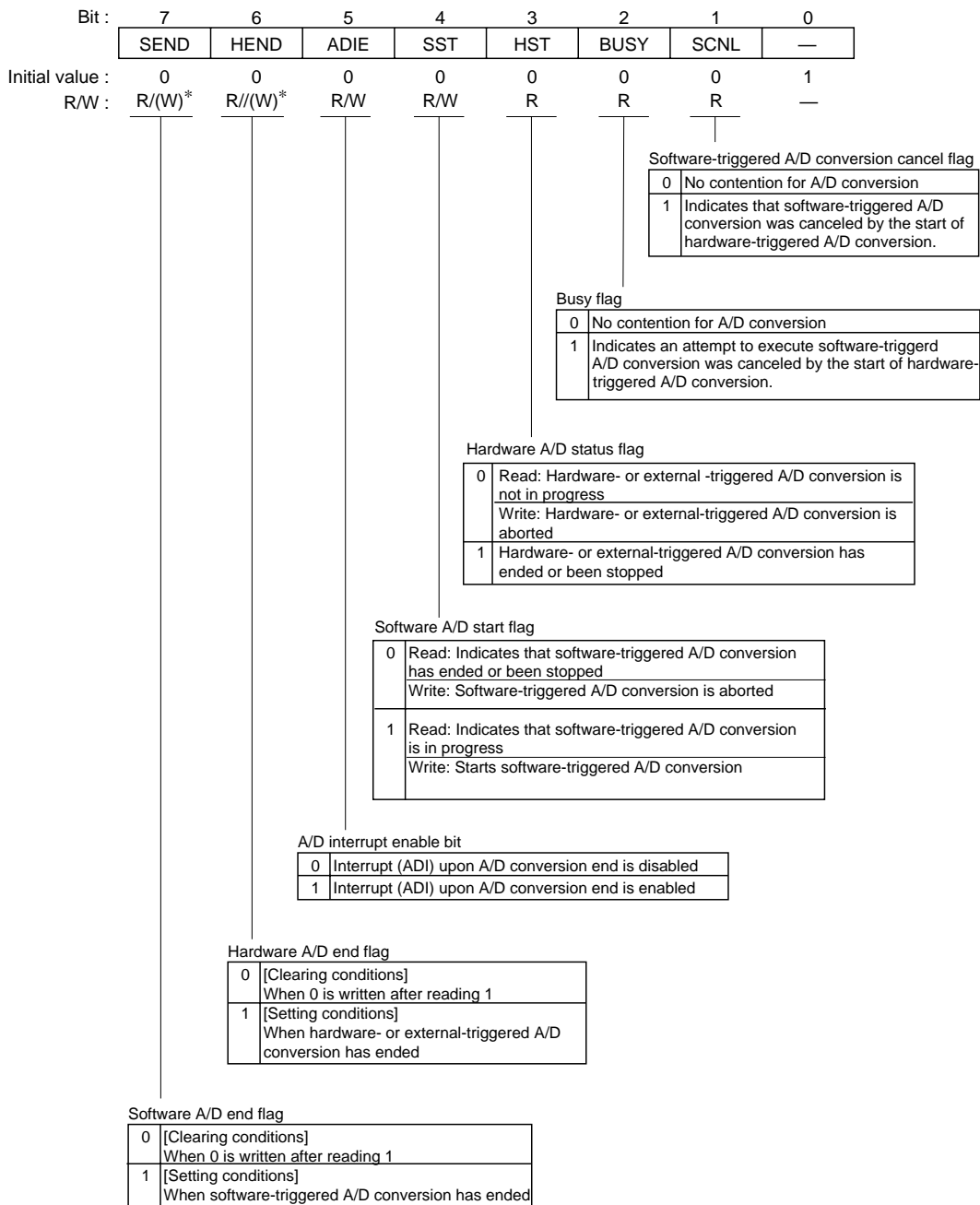
Hardware channel select bits

| HCH1 | HCH2 | Analog input channel |
|------|------|----------------------|
| 0 | 0 | AN8 |
| | 1 | AN9 |
| 1 | 0 | ANA |
| | 1 | ANB |

Clock select

| | |
|---|-----------------------------------|
| 0 | Conversion frequency = 266 states |
| 1 | Conversion frequency = 134 states |

H'D135: A/D Control/Status Register ADCSR: A/D Converter



Note: * Only 0 can be written to clear the flag.

H'D136: A/D Trigger Select Register ADTSR: A/D Converter

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | TRGS1 | TRGS0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W : | — | — | — | — | — | — | R/W | R/W |

Trigger select bits

| TRGS1 | TRGS0 | |
|-------|-------|--|
| 0 | 0 | Hardware- or external-triggered A/D conversion is disabled |
| | 1 | Hardware-triggered (ADTRG) A/D conversion is selected |
| 1 | 0 | Hardware-triggered (DFG) A/D conversion is selected |
| | 1 | External-triggered (ADTRG) A/D conversion is selected |

H'D138: Timer Load Register K TLK: Timer J

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TLR27 | TLR26 | TLR25 | TLR24 | TLR23 | TLR22 | TLR21 | TLR20 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W |

H'D138: Timer Counter K TCK: Timer J

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TDR27 | TDR26 | TDR25 | TDR24 | TDR23 | TDR22 | TDR21 | TDR20 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R | R | R | R | R | R | R | R |

H'D139: Timer Load Register J TLJ: Timer J

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TLR17 | TLR16 | TLR15 | TLR14 | TLR13 | TLR12 | TLR11 | TLR10 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | W | W | W | W | W | W | W | W |

H'D139: Timer Counter J TCJ: Timer J

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TDR17 | TDR16 | TDR15 | TDR14 | TDR13 | TDR12 | TDR11 | TDR10 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R | R | R | R | R | R | R | R |

H'D13A: Timer Mode Register J TMJ: Timer J

| | | | | | | | | |
|-----------------|------|------|-----|------|------|------|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PS11 | PS10 | ST | 8/16 | PS21 | PS20 | TGL | T/R |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R | R/W |

Timer output/remote-controller output select bit

| | |
|---|---|
| 0 | TMJ-1 timer output |
| 1 | TMJ-1 toggle output (data transmitted from remote controller) |

TMJ-2 toggle flag

| | |
|---|--------------------------|
| 0 | TMJ-2 toggle output is 0 |
| 1 | TMJ-2 toggle output is 1 |

TMJ-2 input clock select bits

| PS22 ^{*3} | PS21 | PS20 | TMJ-2 input clock select |
|--------------------|---------------|---------------|---|
| 1 | 0 | 0 | Counting by the PSS, $\phi/16384$ |
| | | 1 | Counting by the PSS, $\phi/2048$ |
| | 1 | 0 | Counting at underflowing of the TMJ-1 |
| | | 1 | Counting at the leading edge or the trailing edge of the external clock inputs (IRQ2) ^{*1} |
| 0 | ^{*2} | ^{*2} | Counting by the PSS, $\phi/1024$ (available only in the H8S/2194C series) |

Notes: 1. The edge selection for the external clock inputs is made by setting the register (IEGR). See section 6.2.4, Edge Select Register (IEGR) for more information.
2. Don't care.
3. Available only in the H8S/2194C series.

8-bit/16-bit operation select bit

| | |
|---|--|
| 0 | TMJ-1 and TMJ-2 operate separately |
| 1 | TMJ-1 and TMJ-2 operate together as 16-bit |

Remote-controlled operation start bit

| | |
|---|---|
| 0 | Stop TMJ-1 clock supply in remote control mode |
| 1 | Start TMJ-1 clock supply in remote control mode |

TMJ-1 input clock select bits

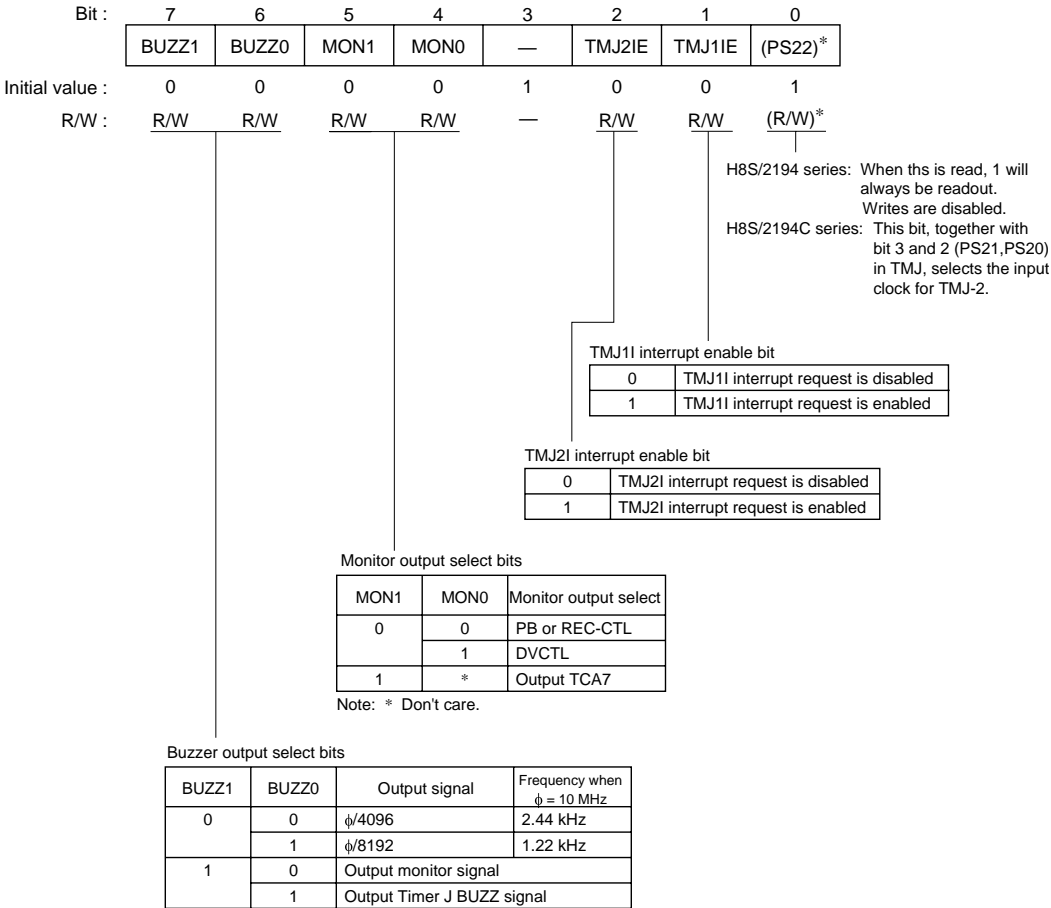
| | | |
|------|------|---|
| PS11 | PS10 | TMJ-1 input clock select |
| 0 | 0 | PSS, count at $\phi/512$ |
| | 1 | PSS, count at $\phi/256$ |
| 1 | 0 | PSS, count at $\phi/4$ |
| | 1 | Count at rising/falling edge of external clock (IRQ1) |

Note: * External clock edge selection is set in edge select register (IEGR).

See section explaining edge select register (IEGR).

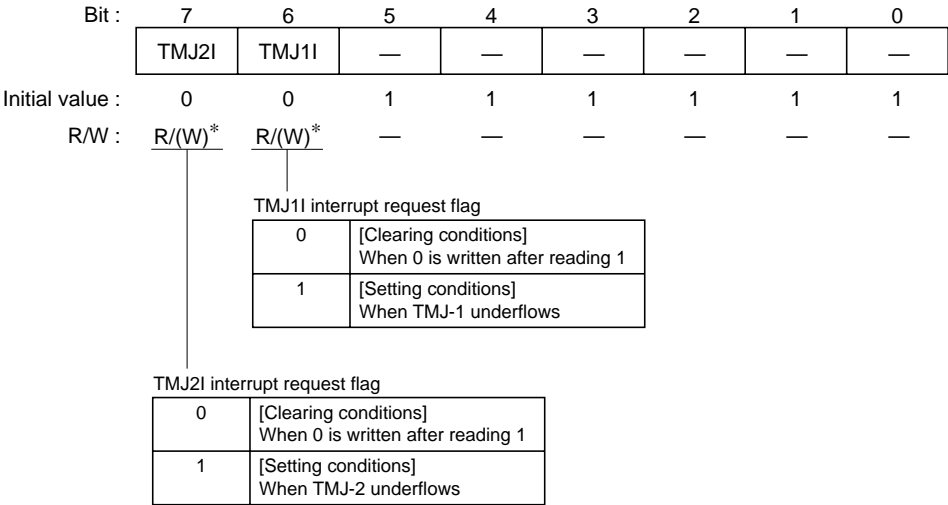
When using external clock in remote control mode, set opposite edges for IRQ1 and IRQ2 edges (eg. When falling edge is set for IRQ1, set rising edge for IRQ2).

H'D13B: Timer J Control Register TMJC: Timer J



Note: * Bit 0 is readable/writable only in the H8S/2194C series.

H'D13C: Timer J Status Register TMJS: Timer J



Note: * Only 0 can be written to clear the flag.

H'D148: Serial Mode Register SMR1: SCI1

Bit :
Initial value :
R/W :

7
0
R/W

6
0
R/W

5
0
R/W

4
0
R/W

3
0
R/W

2
0
R/W

1
0
R/W

0
0
R/W

C/A

CHR

PE

O/E

STOP

MP

CKS1

CKS0

Clock select

| CKS1 | CKS0 | Clock select |
|------|------|-----------------|
| 0 | 0 | ϕ clock |
| | 1 | $\phi/4$ clock |
| 1 | 0 | $\phi/16$ clock |
| | 1 | $\phi/64$ clock |

Multiprocessor mode

| | |
|---|-------------------------------------|
| 0 | Multiprocessor function is disabled |
| 1 | Multiprocessor format is selected |

Stop bit length

| | |
|---|--------------------------|
| 0 | 1 Stop bit* ¹ |
| 1 | 2 Stop bit* ² |

Notes: 1. In transmission, a single 1 bit (stop bit) is added to the end of a transmit character before it is sent.
2. In transmission, two 1 bits (stop bits) are added to the end of a transmit character before it is sent.

Parity mode

| | |
|---|---------------------------|
| 0 | Even parity* ¹ |
| 1 | Odd parity* ² |

Notes: 1. When even parity is set, parity bit addition is performed in transmission so that the total number of 1 bits in the transmit character plus the parity bit is even. In reception, a check is performed to see if the total number of 1 bits in the receive character plus the parity bit is even.
2. When odd parity is set, parity bit addition is performed in transmission so that the total number of 1 bits in the transmit character plus the parity bit is odd. In reception, a check is performed to see if the total number of 1 bits in the receive character plus the parity bit is odd.

Parity enable

| | |
|---|---|
| 0 | Parity bit addition and checking disabled |
| 1 | Parity bit addition and checking enabled* |

Note: * When the PE bit is set to 1, the parity (even or odd) specified by the O/E bit is added to transmit data before transmission. In reception, the parity bit is checked for the parity (even or odd) specified by the O/E bit.

Character length

| | |
|---|------------|
| 0 | 8-bit data |
| 1 | 7-bit data |

Note: * When 7-bit data is selected, the MSB (bit 7) of TDR is not transmitted, and LSB-first/MSB-first selection is not available.

Communication mode

| | |
|---|-----------------------------|
| 0 | Start-stop synchronous mode |
| 1 | Clock synchronous mode |

H'D149: Bit Rate Register BRR1: SCI1

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'D14A: Serial Control Register SCR1: SCI1

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Clock enable bits

| CKE1 | CKE0 | Clock select | |
|------|------|-----------------------------|---|
| 0 | 0 | Start-stop synchronous mode | Internal clock/SCK1 pin function as I/O port ^{*1} |
| | | Clock synchronous mode | Internal clock/SCK1 pin function as serial clock output ^{*1} |
| 1 | 0 | Start-stop synchronous mode | Internal clock/SCK1 pin function as clock output ^{*2} |
| | | Clock synchronous mode | Internal clock/SCK1 pin function as serial clock output |
| | 1 | Start-stop synchronous mode | External clock/SCK1 pin function as clock input ^{*3} |
| | | Clock synchronous mode | External clock/SCK1 pin function as clock input ^{*3} |

Notes: 1. Initial value
2. Outputs a clock of the same frequency as the bit rate.
3. Inputs a clock with a frequency 16 times the bit rate.

Transmit end interrupt enable bit

| | |
|---|---|
| 0 | Transmit-end interrupt (TEI) request is disabled* |
| 1 | Transmit-end interrupt (TEI) request is enabled* |

Note: * TEI cancellation can be performed by reading 1 from the TDRE flag in SSR, then clearing it to 0 and clearing the TEND flag to 0, or clearing the TEIE bit to 0.

Multiprocessor interrupt enable bit

| | |
|---|---|
| 0 | Multiprocessor interrupts are disabled (normal reception performed) [Clearing conditions] (1) When the MPIE bit is cleared to 0 (2) When data with MPB = 1 is received |
| 1 | Multiprocessor interrupt are enabled* Receive interrupt (RXI) requests, receive-error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SSR1 are disabled until data with the multiprocessor bit set to 1 is received. |

Note: * When receive data including MPB = 0 is received, receive data transfer from RSR to RDR1, receive error detection, and setting of the RDRF, FER, and ORER flags in SSR1, is not performed. When receive data with MPB = 1 is received, the MPB bit in SSR1 is set to 1, the MPIE bit is cleared to 0 automatically, and generation of RXI and ERI interrupts (when the TIE and RIE bits in SCR1 are set to 1) and FER and ORER flag setting is enabled.

Receive enable bit

| | |
|---|-------------------------------------|
| 0 | Reception is disabled ^{*1} |
| 1 | Reception is enabled ^{*2} |

Notes: 1. Clearing the RE bit to 0 does not affect the RDRF, FER, PER, and ORER flags, which retain their states.
2. Serial reception is started in this state when a start bit is detected in asynchronous mode or serial clock input is detected in synchronous mode.
SMR1 setting must be performed to decide the reception format before setting the RE bit to 1.

Transmit enable bit

| | |
|---|--|
| 0 | Transmission is disabled ^{*1} |
| 1 | Transmission is enabled ^{*2} |

Notes: 1. The TDRE flag in SSR1 is fixed at 1.
2. In this state, serial transmission is started when transmit data is written to TDR and TDRE flag in SSR1 is cleared to 0.
SMR1 setting must be performed to decide the transmission format before setting the TE bit to 1.

Receive interrupt enable bit

| | |
|---|--|
| 0 | Reveive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request is disabled* |
| 1 | Reveive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request is enabled |

Note: * RXI and ERI interrupt request cancellation can be performed by reading 1 from the RDRF, FER, PER, or ORER flag, then clearing the flag to 0, or clearing the RIE bit to 0.

Transmit interrupt enable bit

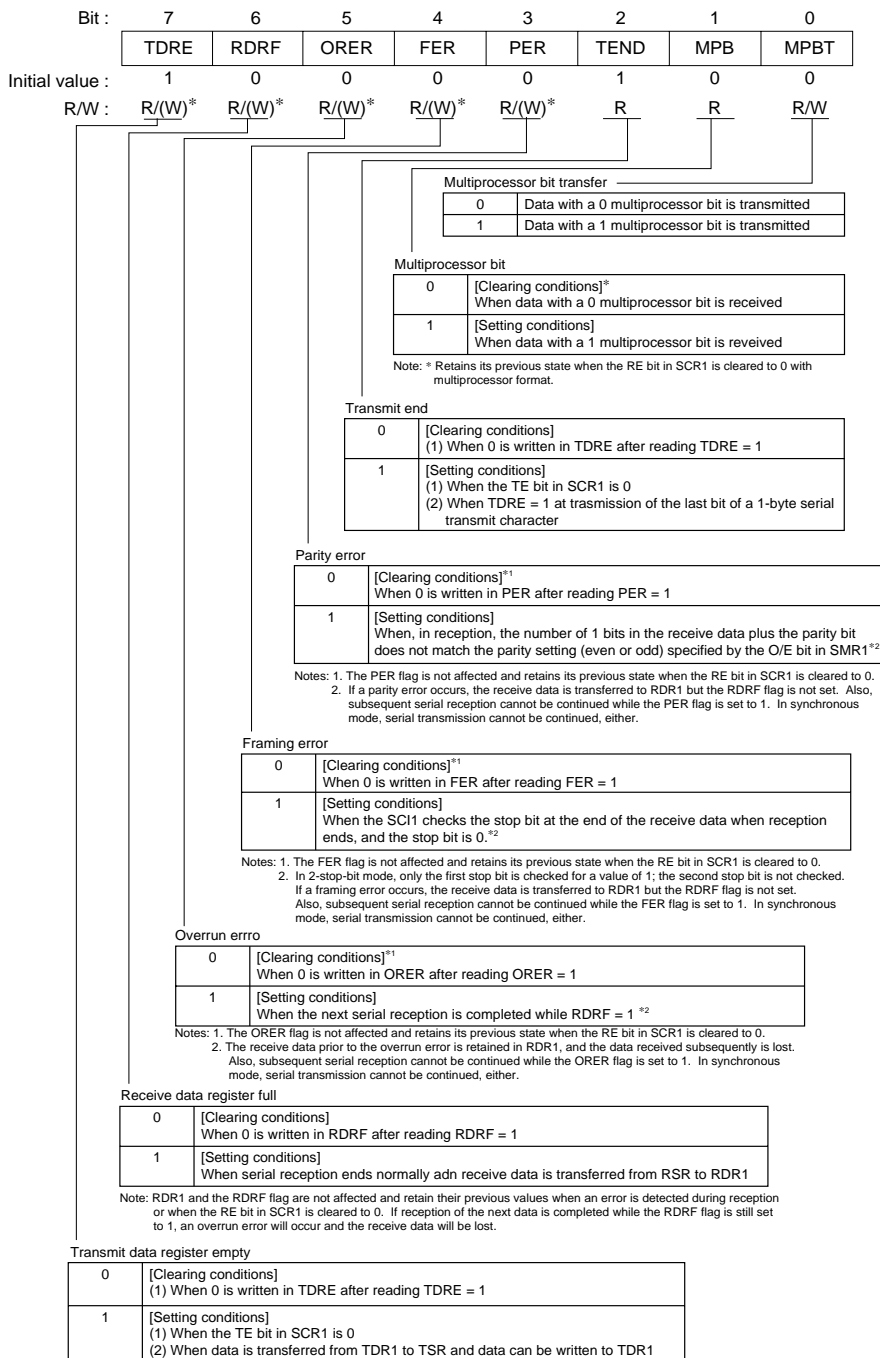
| | |
|---|--|
| 0 | Transmit-data-empty interrupt (TXI) request is disabled* |
| 1 | Transmit-data-empty interrupt (TXI) request is enabled |

Note: * TXI interrupt request cancellation can be performed by reading 1 from the TDRE flag, then clearing it to 0, or clearing the TIE bit to 0.

H'D14B: Transmit Data Register TDR1: SCI1

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'D14C: Serial Status Register SSR1: SCI1



Note: * Only 0 can be written to clear the flag.

H'D14D: Receive Data Register RDR1: SCI1

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R |

H'D14E: Serial Interface Mode Register SCMR1: SCI1

| | | | | | | | | |
|-----------------|---|---|---|---|------|------|---|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | SDIR | SINV | — | SMIF |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| R/W : | — | — | — | — | R/W | R/W | — | R/W |

Serial communication interface mode select

| | |
|---|-----------------|
| 0 | Normal SCI mode |
| 1 | Reserved mode |

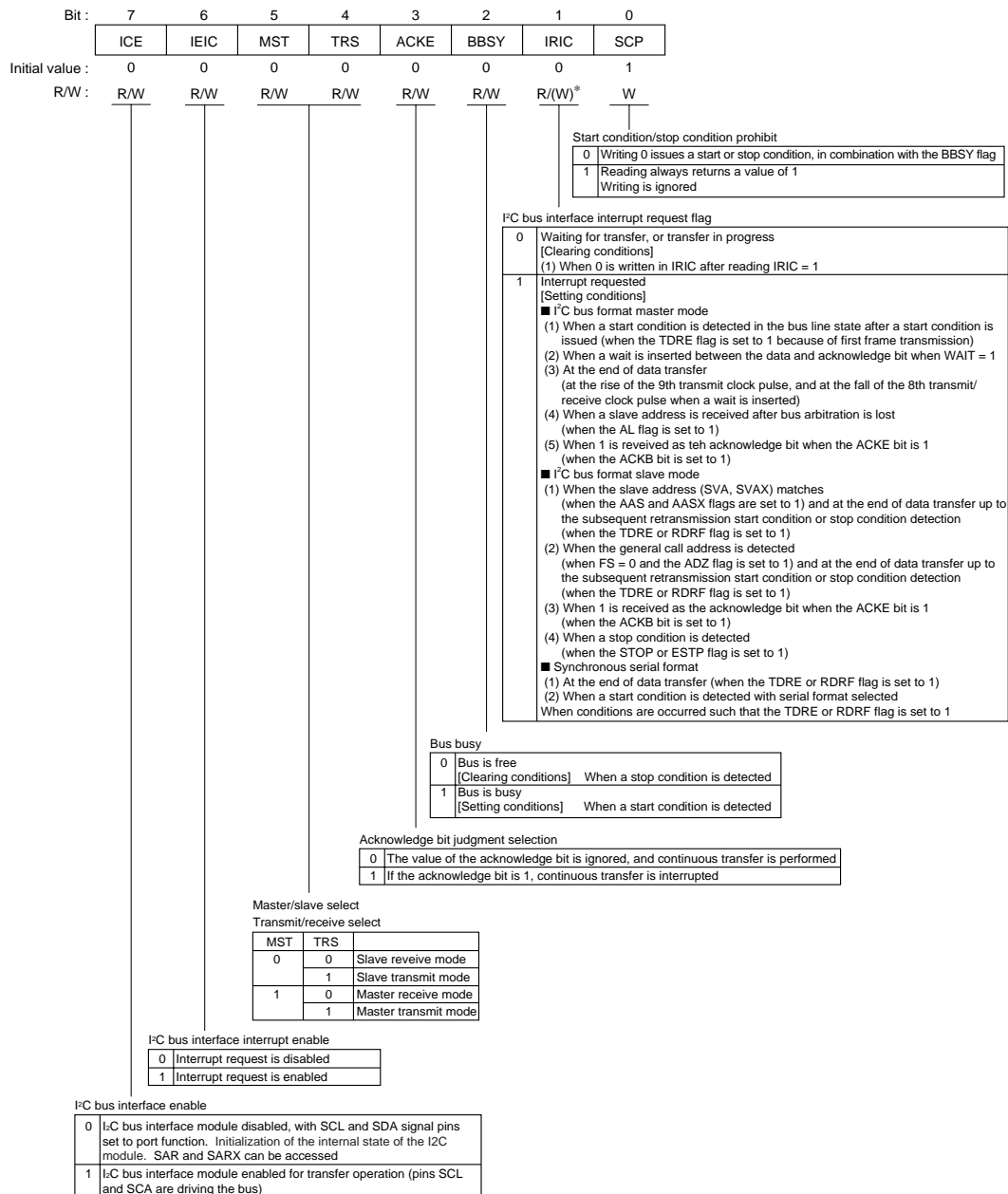
Data invert

| | |
|---|---|
| 0 | TDR1 contents are transmitted without modification Receive data is stored in RDR1 without modification |
| 1 | TDR1 contents are onverted before being transmitted Receive data is stored in RDR1 in inverted form |

Data transfer direction

| | |
|---|---|
| 0 | TDR1 contents are transmitted LSB-first Receive data is stored in RDR1 LSB-first |
| 1 | TDR1 contents are transmitted MSB-first Receive data is stored in RDR1 MSB-first |

H'D158: I²C Bus Control Register ICCR: IIC Bus Interface



Note: * Only 0 can be written to clear the flag.

H'D159: I²C Bus Status Register ICSR: IIC Bus Interface

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ESTP | STOP | IRTR | AASX | AL | AAS | ADZ | ACKB |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/W |

| |
|--|
| <div> Acknowledge bit 0 Receive mode: 0 is output at acknowledge output timing Transmit mode: Indicates that the receiving device has acknowledged the data (signal is 0) 1 Receive mode: 1 is output at acknowledge output timing Transmit mode: Indicates that the receiving device has not acknowledged the data (signal is 1) </div> |
|--|

| |
|--|
| <div> General call address recognition flag 0 General call address not recognized [Clearing conditions] (1) When ICDR data is written (transmit mode) or read (receive mode) (2) When 0 is written in ADZ after reading ADZ = 1 (3) In master mode 1 General call address recognized [Setting conditions] • When the general call address is detected in slave receive mode </div> |
|--|

| |
|--|
| <div> Slave address recognition flag 0 Slave address or general call address not recognized [Clearing conditions] (1) When ICDR data is written (transmit mode) or read (receive mode) (2) When 0 is written in AAS after reading AAS = 1 (3) In master mode 1 Slave address or general call address recognized [Setting conditions] • When the slave address or general call address is detected in slave receive mode </div> |
|--|

| |
|--|
| <div> Arbitration lost flag 0 Bus arbitration won [Clearing conditions] (1) When ICDR data is written (transmit mode) or read (receive mode) (2) When 0 is written in AL after reading AL = 1 1 Arbitration lost [Setting conditions] (1) If the internal SDA and SDA pin disagree at the rise of SCL in master transmit mode (2) If the internal SCL line is high at the fall of SCL in master transmit mode </div> |
|--|

| |
|--|
| <div> Second slave address recognition flag 0 Second slave address not recognized [Clearing conditions] (1) When 0 is written in AASX after reading AASX = 1 (2) When a start condition is detected (3) In master mode 1 Second slave address recognized [Setting conditions] • When the second slave address is detected in slave receive mode while FSX = 0 </div> |
|--|

| |
|--|
| <div> I²C bus interface continuous transmission/reception interrupt request flag 0 Waiting for transfer, or transfer in progress [Clearing conditions] (1) When 0 is written in IRTR after reading IRTR = 1 (2) When the IRIC flag is cleared to 0 1 Continuous transfer state [Setting conditions] ■ In I²C bus interface slave mode • When the TDRE or RDRF flag is set to 1 when AASX = 1 ■ In other mode • When the TDRE or RDRF flag is set to 1 </div> |
|--|

| |
|--|
| <div> Normal stop condition detection flag 0 No normal stop condition [Clearing conditions] (1) When 0 is written in STOP after reading STOP = 1 (2) When the IRIC flag is cleared to 0 1 ■ In I²C bus format slave mode Normal stop condition detected [Setting conditions] • When a stop condition is detected after completion of frame transfer ■ In other mode No meaning </div> |
|--|

| |
|--|
| <div> Error stop condition detection flag 0 No error stop condition [Clearing conditions] (1) When 0 is written in ESTP after reading ESTP = 1 (2) When the IRIC flag is cleared to 0 1 ■ In I²C bus format slave mode Error stop condition detected [Setting conditions] • When a stop condition is detected during frame transfer ■ In other mode No meaning </div> |
|--|

Note: * Only 0 can be written to clear the flag.

H'D15E: I²C Bus Data Register ICDR: IIC Bus Interface

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICDR7 | ICDR6 | ICDR5 | ICDR4 | ICDR3 | ICDR2 | ICDR1 | ICDR0 |
| Initial value : | — | — | — | — | — | — | — | — |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'D15E: Second Slave Address Register SARX: IIC Bus Interface

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | SVAX6 | SVAX5 | SVAX4 | SVAX3 | SVAX2 | SVAX1 | SVAX0 | FSX |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Format select
Used combined with SAR FS bit.

H'D15F: I²C Bus Mode Register ICMR: IIC Bus Interface

| | | | | | | | | |
|-----------------|-----|------|------|------|------|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | MLS | WAIT | CKS2 | CKS1 | CKS0 | BC2 | BC1 | BC0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Bit counter

| | | | Bit/frame | |
|-----|-----|-----|--------------------------|-----------------------------|
| BC2 | BC1 | BC0 | Clock sync serial format | I ² C bus format |
| 0 | 0 | 0 | 8 | 9 |
| | | 1 | 1 | 2 |
| | 1 | 0 | 2 | 3 |
| | | 1 | 3 | 4 |
| 0 | 0 | 0 | 4 | 5 |
| | | 1 | 5 | 6 |
| | 1 | 0 | 6 | 7 |
| | | 1 | 7 | 8 |

Transfer clock select bits

| IICX* | CKS2 | CKS1 | CKS0 | Clock | Transfer rate | | |
|-------|------|------|------|------------|---------------|--------------|---------------|
| | | | | | $\phi=5$ MHz | $\phi=8$ MHz | $\phi=10$ MHz |
| 0 | 0 | 0 | 0 | $\phi/28$ | 179 kHz | 286 kHz | 357 kHz |
| | | | 1 | $\phi/40$ | 125 kHz | 200 kHz | 250 kHz |
| | | 1 | 0 | $\phi/48$ | 104 kHz | 167 kHz | 208 kHz |
| | | | 1 | $\phi/64$ | 78.1 kHz | 125 kHz | 156 kHz |
| | 1 | 0 | 0 | $\phi/80$ | 62.5 kHz | 100 kHz | 125 kHz |
| | | | 1 | $\phi/100$ | 50.0 kHz | 80.0 kHz | 100 kHz |
| | | 1 | 0 | $\phi/112$ | 44.6 kHz | 71.4 kHz | 89.3 kHz |
| | | | 1 | $\phi/128$ | 39.1 kHz | 62.5 kHz | 78.1 kHz |
| | 1 | 0 | 0 | $\phi/56$ | 89.3 kHz | 143 kHz | 179 kHz |
| | | | 1 | $\phi/80$ | 62.5 kHz | 100 kHz | 125 kHz |
| | | 1 | 0 | $\phi/96$ | 52.1 kHz | 83.3 kHz | 104 kHz |
| | | | 1 | $\phi/128$ | 39.1 kHz | 62.5 kHz | 78.1 kHz |
| 1 | 0 | 0 | 0 | $\phi/160$ | 31.3 kHz | 50.0 kHz | 62.5 kHz |
| | | | 1 | $\phi/200$ | 25.0 kHz | 40.0 kHz | 50.0 kHz |
| | | 1 | 0 | $\phi/224$ | 22.3 kHz | 35.7 kHz | 44.6 kHz |
| | | | 1 | $\phi/256$ | 19.5 kHz | 31.3 kHz | 39.1 kHz |

Wait insertion bit

| | |
|---|---|
| 0 | Data and acknowledge bits transferred consecutively |
| 1 | Wait inserted between data and acknowledge bits |

MSB-first/LSB-first select

| | |
|---|-----------|
| 0 | MSB-first |
| 1 | LSB-first |

Note: * See STCR Bit 6.

H'D15F: Slave Address Register SAR: IIC Bus Interface

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | SVA6 | SVA5 | SVA4 | SVA3 | SVA2 | SVA1 | SVA0 | FS |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Format select bit

| SAR Bit 0 | SARX Bit 0 | Format select |
|--------------|---------------|---|
| FS | FX | |
| 0 | 0 | I ² C bus format • SAR and SARX slave addresses recognized |
| | 1 | I ² C bus format • SAR slave address recognized • SARX slave address ignored |
| 1 | 0 | I ² C bus format • SAR slave address ignored • SARX slave address recognized |
| | 1 | I ² C bus format • SAR and SARX slave addresses ignored |

H'FFB0: Trap Address Register 0 TAR0: ATC

H'FFB3: Trap Address Register 1 TAR1: ATC

H'FFB6: Trap Address Register 2 TAR2: ATC

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | A23 | A22 | A21 | A20 | A19 | A18 | A17 | A16 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | A7 | A6 | A5 | A4 | A3 | A2 | A1 | — |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | — |

H'FFB9: Address Trap Control Register ATCR: ATC

| | | | | | | | | |
|-----------------|---|---|---|---|---|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | TRC2 | TRC1 | TRC0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| R/W : | — | — | — | — | — | R/W | R/W | R/W |

| | |
|----------------|-------------------------------------|
| Trap control 0 | |
| 0 | Address trap function 0 is disabled |
| 1 | Address trap function 0 is enabled |

| | |
|----------------|-------------------------------------|
| Trap control 1 | |
| 0 | Address trap function 1 is disabled |
| 1 | Address trap function 1 is enabled |

| | |
|----------------|-------------------------------------|
| Trap control 2 | |
| 0 | Address trap function 2 is disabled |
| 1 | Address trap function 2 is enabled |

H'FFBA: Timer Mode Register A TMA: Timer A

| | | | | | | | | |
|-----------------|--------|-------|---|---|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TMAOV | TMAIE | — | — | TMA3 | TMA2 | TMA1 | TMA0 |
| Initial value : | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/W | — | — | R/W | R/W | R/W | R/W |

Clock select bits

| TMA3 | TMA2 | TMA1 | TMA0 | Prescaler frequency division rate (interval timer) or overflow frequency (time-base) | Operation mode |
|------|------|------|------|---|-------------------------|
| 0 | 0 | 0 | 0 | PSS, $\phi/16384$ | Interval timer mode |
| | | | 1 | PSS, $\phi/8192$ | |
| | | 1 | 0 | PSS, $\phi/4096$ | |
| | | | 1 | PSS, $\phi/1024$ | |
| | 1 | 0 | 0 | PSS, $\phi/512$ | |
| | | | 1 | PSS, $\phi/256$ | |
| | | 1 | 0 | PSS, $\phi/64$ | |
| | | | 1 | PSS, $\phi/16$ | |
| 1 | 0 | 0 | 0 | 1 s | Clock time base mode |
| | | | 1 | 0.5 s | |
| | | 1 | 0 | 0.25 s | |
| | | | 1 | 0.03125 s | |
| | 1 | 0 | 0 | Clear PSW and TCA to H'00 | |
| | | | 1 | | |
| | | 1 | 0 | | |
| | | | 1 | | |

Note: $\phi = f_{osc}$

Clock source, prescaler select bit

| | |
|---|-----------------------------|
| 0 | Timer A clock source is PSS |
| 1 | Timer A clock source is PSW |

Timer A interrupt enable bit

| | |
|---|---|
| 0 | Interrupt request by Timer A (TMAI) is disabled |
| 1 | Interrupt request by Timer A (TMAI) is enabled |

Timer A overflow flag

| | |
|---|--|
| 0 | [Clearing conditions] When 0 is written to TMAOV after reading TMAOV = 1 |
| 1 | [Setting conditions] When TCA overflows |

Note: * Only 0 can be written to clear the flag.

H'FFBB: Timer Counter A TCA: TimerA

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TCA7 | TCA6 | TCA5 | TCA4 | TCA3 | TCA2 | TCA1 | TCA0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R | R | R | R | R | R | R |

H'FFBC: Watchdog Timer Control/Status Register WTCNR: WDT

| | | | | | | | | |
|-----------------|--------|-------|-----|------|---------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | OVF | WT/IT | TME | RSTS | RST/NMI | CKS2 | CKS1 | CKS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/(W)* | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Reset or NMI

| | |
|---|-------------------------------------|
| 0 | NMI interrupt request is disabled |
| 1 | Internal reset request is generated |

Timer enable bit

| | |
|---|---|
| 0 | WTCNT is initialized to H'00 and halted |
| 1 | WTCNT counts |

Timer mode select bit

| | |
|---|--|
| 0 | Interval timer mode: Sends the CPU an interval timer interrupt request (WOVI) when WTCNT overflows |
| 1 | Watchdog timer mode: Sends the CPU a reset or NMI interrupt request when WTCNT overflows |

Overflow flag

| | |
|---|--|
| 0 | [Clearing conditions] (1) Write 0 in the TME bit (2) Read WTCNR when OVF = 1, then write 0 in OVF |
| 1 | [Setting conditions] When WTCNT overflows (changes from H'FF to H'00) (When internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset.) |

Note: * Only 0 can be written to clear the flag.

H'FFBD: Watchdog Timer Counter WTCNT: WDT

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'FFC0: Port Data Register 0 PDR0: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR07 | PDR06 | PDR05 | PDR04 | PDR03 | PDR02 | PDR01 | PDR00 |
| Initial value : | — | — | — | — | — | — | — | — |
| R/W : | R | R | R | R | R | R | R | R |

H'FFC1: Port Data Register 1 PDR1: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR17 | PDR16 | PDR15 | PDR14 | PDR13 | PDR12 | PDR11 | PDR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'FFC2: Port Data Register 2 PDR2: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR27 | PDR26 | PDR25 | PDR24 | PDR23 | PDR22 | PDR21 | PDR20 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'FFC3: Port Data Register 3 PDR3: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR37 | PDR36 | PDR35 | PDR34 | PDR33 | PDR32 | PDR31 | PDR30 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'FFC4: Port Data Register 4 PDR4: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR47 | PDR46 | PDR45 | PDR44 | PDR43 | PDR42 | PDR41 | PDR40 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'FFC5: Port Data Register 5 PDR5: I/O Port

| | | | | | | | | |
|-----------------|---|---|---|---|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PDR53 | PDR52 | PDR51 | PDR50 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | R/W | R/W | R/W | R/W |

H'FFC6: Port Data Register 6 PDR6: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR67 | PDR66 | PDR65 | PDR64 | PDR63 | PDR62 | PDR61 | PDR60 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'FFC7: Port Data Register 7 PDR7: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR77 | PDR76 | PDR75 | PDR74 | PDR73 | PDR72 | PDR71 | PDR70 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'FFC8: Port Data Register 8 PDR8: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PDR87 | PDR86 | PDR85 | PDR84 | PDR83 | PDR82 | PDR81 | PDR80 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

H'FFCD: Port Mode Register 0 PMR0: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR07 | PMR06 | PMR05 | PMR04 | PMR03 | PMR02 | PMR01 | PMR00 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | |
|--|---|
| P07/AN7 to P00/IRQ0 pin function select bits | |
| 0 | P0n/ANn pin functions as P0n input port |
| 1 | P0n/ANn pin functions as ANn input port |

(n = 7 to 0)

H'FFCE: Port Mode Register 1 PMR1: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR17 | PMR16 | PMR15 | PMR14 | PMR13 | PMR12 | PMR11 | PMR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

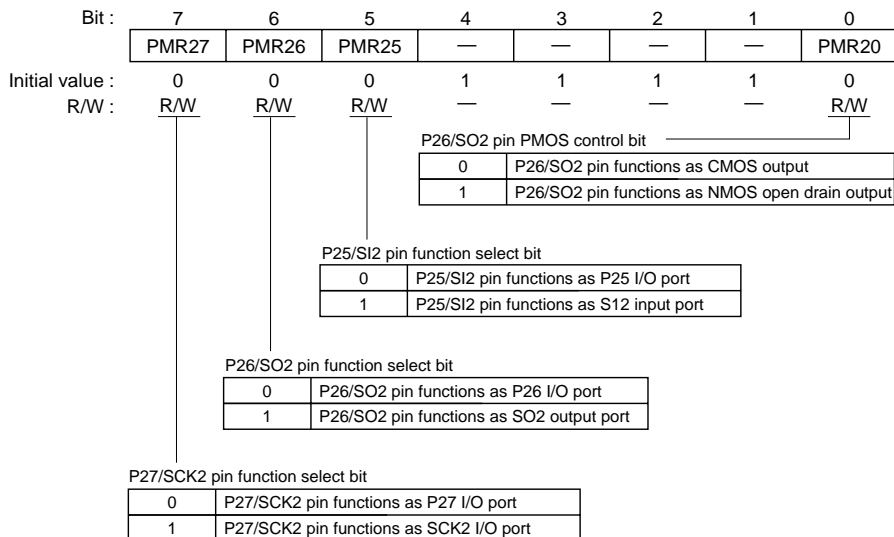
| | |
|---|---|
| P15/IRQ5 to P10/IRQ0 pin function select bits | |
| 0 | P1n/IRQn pin functions as P1n I/O port |
| 1 | P1n/IRQn pin functions as IRQn input port |

(n = 5 to 0)

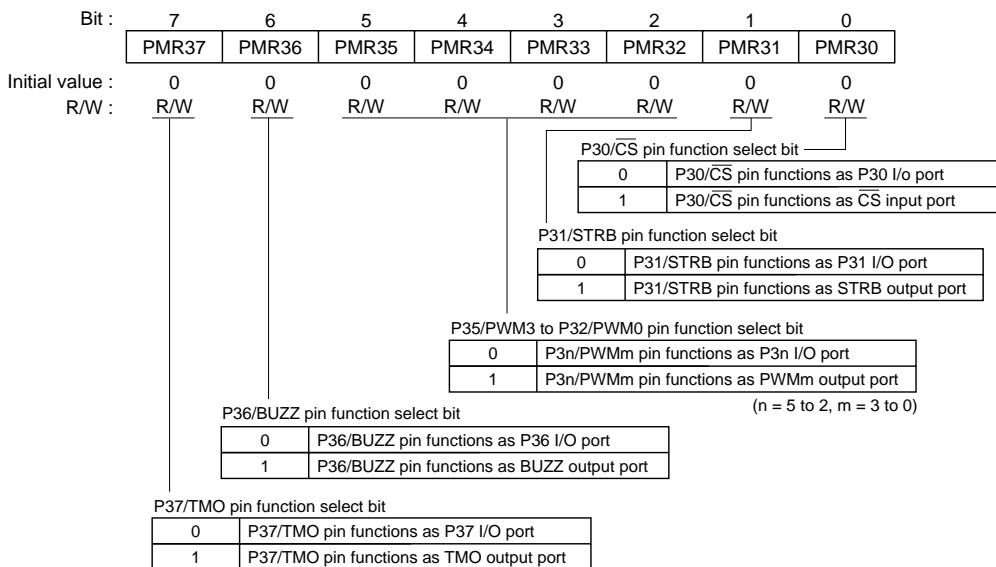
| | |
|--------------------------------|---------------------------------------|
| P16/IC pin function select bit | |
| 0 | P16/IC pin functions as P16 I/O port |
| 1 | P16/IC pin functions as IC input port |

| | |
|----------------------------------|--|
| P17/TMOW pin function select bit | |
| 0 | P17/TMOW pin functions as P17 I/O port |
| 1 | P17/TMOW pin functions as TMOW output port |

H'FFCF: Port Mode Register 2 PMR2: I/O Port



H'FFD0: Port Mode Register 3 PMR3: I/O Port



Notes: If the TMO pin is used for remote control sending, a careless timer output pulse may be output when the remote control mode is set after the output has been switched to the TMO output. Perform the switching and setting in the following order.

- [1] Set the remote control mode.
- [2] Set the TMJ-1 and 2 counter data of the timer J.
- [3] Switch the P37/TMO pin to the TMO output pin.
- [4] Set the ST bit to 1.

H'FFD1: Port Control Register 1 PCR1: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR17 | PCR16 | PCR15 | PCR14 | PCR13 | PCR12 | PCR11 | PCR10 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

| | |
|---|----------------------------------|
| 0 | P1n pin functions as input port |
| 1 | P1n pin functions as output port |

(n = 7 to 0)

H'FFD2: Port Control Register 2 PCR2: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR27 | PCR26 | PCR25 | PCR24 | PCR23 | PCR22 | PCR21 | PCR20 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

| | |
|---|----------------------------------|
| 0 | P2n pin functions as input port |
| 1 | P2n pin functions as output port |

(n = 7 to 0)

H'FFD3: Port Control Register 3 PCR3: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR37 | PCR36 | PCR35 | PCR34 | PCR33 | PCR32 | PCR31 | PCR30 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

| | |
|---|----------------------------------|
| 0 | P3n pin functions as input port |
| 1 | P3n pin functions as output port |

(n = 7 to 0)

H'FFD4: Port Control Register 4 PCR4: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR47 | PCR46 | PCR45 | PCR44 | PCR43 | PCR42 | PCR41 | PCR40 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

| | |
|---|----------------------------------|
| 0 | P4n pin functions as input port |
| 1 | P4n pin functions as output port |

(n = 7 to 0)

H'FFD5: Port Control Register 5 PCR5: I/O Port

| | | | | | | | | |
|-----------------|---|---|---|---|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PCR53 | PCR52 | PCR51 | PCR50 |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | W | W | W | W |

| | |
|---|----------------------------------|
| 0 | P5n pin functions as input port |
| 1 | P5n pin functions as output port |

(n = 3 to 0)

H'FFD6: Port Control Register 6 PCR6: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR67 | PCR66 | PCR65 | PCR64 | PCR63 | PCR62 | PCR61 | PCR60 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

| | | |
|-------|-------|--|
| PMR6n | PCR6n | Port Control Register 6 |
| 0 | 0 | P6n/RPn pin functions as P6n general purpose input port |
| | 1 | P6n/RPn pin functions as P6n general purpose output port |
| 1 | * | P6n/RPn pin functions as RPn realtime output port |

Note: * Don't care. (n = 7 to 0)

H'FFD7: Port Control Register 7 PCR7: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR77 | PCR76 | PCR75 | PCR74 | PCR73 | PCR72 | PCR71 | PCR70 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

| | |
|---|----------------------------------|
| 0 | P7n pin functions as input port |
| 1 | P7n pin functions as output port |

(n = 7 to 0)

H'FFD8: Port Control Register 8 PCR8: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PCR87 | PCR86 | PCR85 | PCR84 | PCR83 | PCR82 | PCR81 | PCR80 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | W | W | W | W | W | W | W | W |

| | |
|---|----------------------------------|
| 0 | P8n pin functions as input port |
| 1 | P8n pin functions as output port |

(n = 7 to 0)

H'FFDB: Port Mode Register 4 PMR4: I/O Port

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | PMR40 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| R/W : | — | — | — | — | — | — | — | R/W |

P40/PWM14 pin function select bit

| | |
|---|--|
| 0 | P40/PWM14 pin functions as P40 I/O port |
| 1 | P40/PWM14 pin functions as PWM14 output port |

H'FFDC: Port Mode Register 5: PMR5: I/O Port

| | | | | | | | | |
|-----------------|---|---|---|---|-------|-------|-------|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PMR53 | PMR52 | PMR51 | — |
| Initial value : | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |
| R/W : | — | — | — | — | R/W | R/W | R/W | — |

| | |
|---------------------------------|--|
| Timer B event input edge select | |
| 0 | The timer B event input detects the falling edge |
| 1 | The timer B event input detects the rising edge |

| | |
|----------------------------------|--|
| P52/TMBI pin function select bit | |
| 0 | P52/TMBI pin function as P52 I/O port |
| 1 | P52/TMBI pin functions as TMB input port |

| | |
|----------------------------------|--|
| P53/TRIG pin function select bit | |
| 0 | P53/TRIG pin function as P53 I/O port |
| 1 | P53/TRIG pin function as TRIG input port |

H'FFDD: Port Mode Register 6 PMR6: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR67 | PMR66 | PMR65 | PMR64 | PMR63 | PMR62 | PMR61 | PMR60 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | |
|--|--|
| P67/RP7 to P60/RP0 pin function select bit | |
| 0 | P6n/RPn pin functions as P6n I/O port |
| 1 | P6n/RPn pin functions as RPn output port |

(n = 7 to 0)

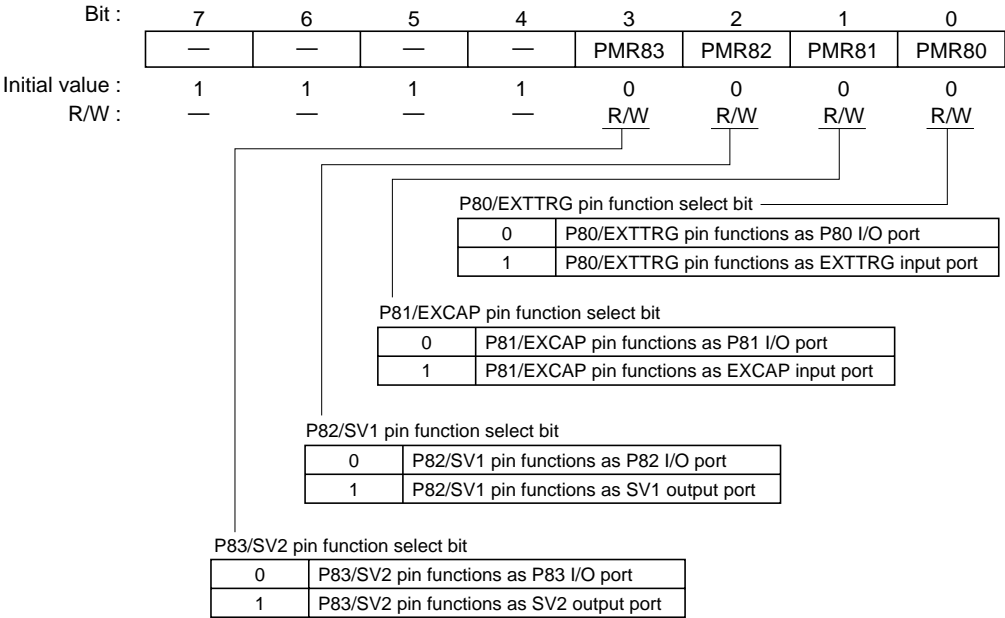
H'FFDE: Port Mode Register 7 PMR7: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PMR77 | PMR76 | PMR75 | PMR74 | PMR73 | PMR72 | PMR71 | PMR70 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

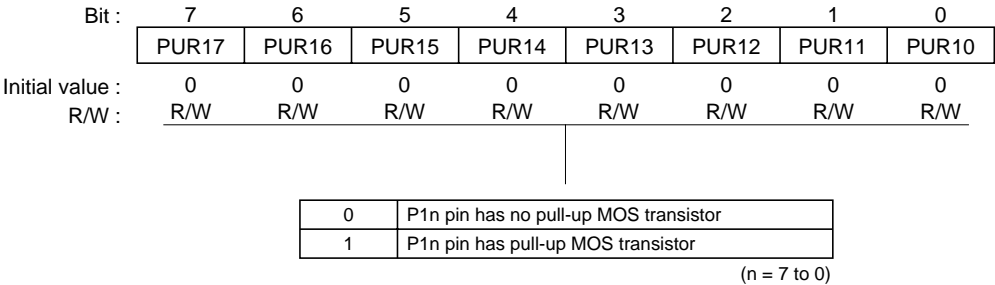
| | |
|--|--|
| P77/PPG7 to P70/PPG0 pin function select bit | |
| 0 | P7n/PPGn pin functions as P7n I/O port |
| 1 | P7n/PPGn pin functions as PPGn output port |

(n = 7 to 0)

H'FFDF: Port Mode Register 8 PMR8: I/O Port



H'FFE1: Pull-Up MOS Select Register 1 PUR1: I/O Port



H'FFE2: Pull-Up MOS Select Register 2 PUR2: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PUR27 | PUR26 | PUR25 | PUR24 | PUR23 | PUR22 | PUR21 | PUR20 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | |
|---|---------------------------------------|
| 0 | P2n pin has no pull-up MOS transistor |
| 1 | P2n pin has pull-up MOS transistor |

(n = 7 to 0)

H'FFE3: Pull-Up MOS Select Register 3 PUR3: I/O Port

| | | | | | | | | |
|-----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PUR37 | PUR36 | PUR35 | PUR34 | PUR33 | PUR32 | PUR31 | PUR30 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | |
|---|---------------------------------------|
| 0 | P3n pin has no pull-up MOS transistor |
| 1 | P3n pin has pull-up MOS transistor |

(n = 7 to 0)

H'FFE4: Realtime Output Trigger Edge Select Register RTPEGR: I/O Port

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---------|---------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | RTPEGR1 | RTPEGR0 |
| Initial value : | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| R/W : | — | — | — | — | — | — | R/W | R/W |

Realtime output trigger edge select bit

| RTPEGR1 | RTPEGR0 | Realtime output trigger edge select |
|---------|---------|---|
| 0 | 0 | Trigger input is disabled |
| | 1 | Rising edge of trigger input is selected |
| 1 | 0 | Falling edge of trigger input is selected |
| | 1 | Rising and falling edges of trigger input is selected |

H'FFE5: Realtime Output Trigger Select Register RTPSR: I/O Port

| | | | | | | | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | RTPSR7 | RTPSR6 | RTPSR5 | RTPSR4 | RTPSR3 | RTPSR2 | RTPSR1 | RTPSR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | |
|---|---|
| 0 | External trigger (TRIG pin) input is selected |
| 1 | Internal trigger (HSW) input is selected |

(n = 7 to 0)

H'FFE8: System Control Register SYSCR: System Control

| | | | | | | | | |
|-----------------|---|---|-------|-------|------|--------|--------|---|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | INTM1 | INTM0 | XRST | NMIEG1 | NMIEG0 | — |
| Initial value : | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| R/W : | — | — | R | R/W | R | R/W | R/W | — |

NMI edge select bits

| NMIEG1 | NMIEG0 | NMI edge select |
|--------|--------|---|
| 0 | 0 | Interrupt request is generated at falling edge of NMI input |
| 0 | 1 | Interrupt request is generated at rising edge of NMI input |
| 1 | * | Interrupt request is generated at rising or falling edge of NMI input |

Note: * Don't care.

External reset

| | |
|---|---|
| 0 | Reset is generated by watchdog timer overflow |
| 1 | Reset is generated by external reset input |

Interrupt control mode

| INTM1 | INTM0 | Interrupt control mode | Interrupt control |
|-------|-------|------------------------|--|
| 0 | 0 | 0 | Interrupt is controlled by I bit |
| 0 | 1 | 1 | Interrupt is controlled by I and UI bits and ICR |
| 1 | 0 | 2 | Cannot be used in the H8S/2194 Series |
| 1 | 1 | 3 | Cannot be used in the H8S/2194 Series |

H'FFE9: Mode Control Register MDCR: System Control

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|---|---------------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | — | MDS0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | —* |
| R / W : | — | — | — | — | — | — | — | R |
| | | | | | | | | Mode select 0 |

Note: * Determined by MD0 pin.

H'FFEA: Standby Control Register SBYCR: System Control

| | | | | | | | | |
|-----------------|------|------|------|------|---|---|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | SSBY | STS2 | STS1 | STS0 | — | — | SCK1 | SCK0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | — | — | R/W | R/W |

System clock select

| SCK1 | SCK0 | System clock select |
|------|------|----------------------------------|
| 0 | 0 | Bus master is in high-speed mode |
| 0 | 1 | Medium-speed clock is $\phi/16$ |
| 1 | 0 | Medium-speed clock is $\phi/32$ |
| 1 | 1 | Medium-speed clock is $\phi/64$ |

Standby timer select bits

| STS2 | STS1 | STS0 | Standby time |
|------|------|------|---------------|
| 0 | 0 | 0 | 8192 states |
| 0 | 0 | 1 | 16384 states |
| 0 | 1 | 0 | 32768 states |
| 0 | 1 | 1 | 65536 states |
| 1 | 0 | 0 | 131072 states |
| 1 | 0 | 1 | 262144 states |
| 1 | 1 | *1 | 16 states *2 |

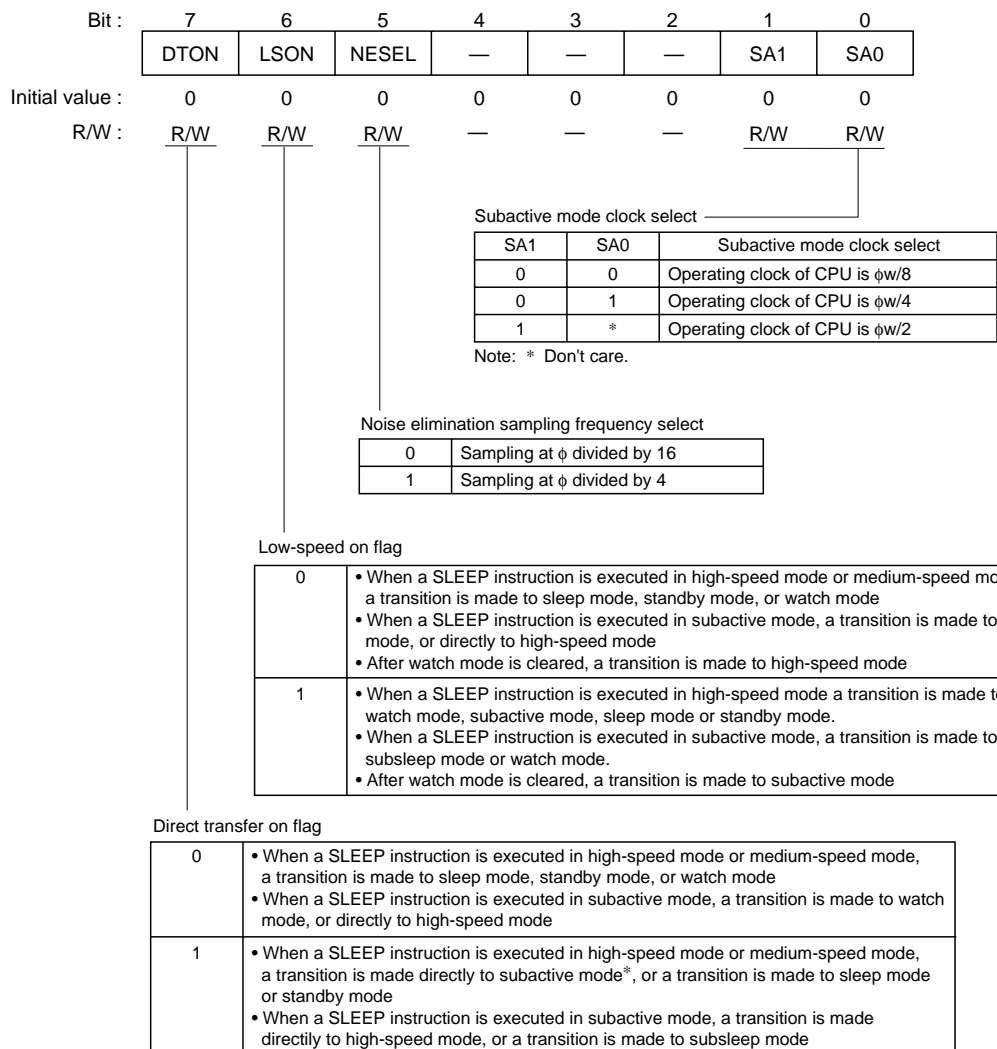
Notes: 1. Don't care.

2. The standby time is 32 states when transitioned to medium-speed mode $\phi/32$ (SCK = 1, SCK = 0).
Do not select 16 states for Flash ROM version.

Software standby

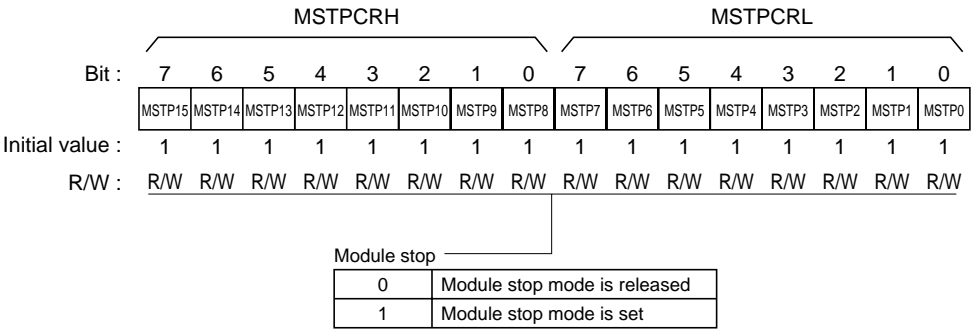
| | |
|---|--|
| 0 | Transition to sleep mode after execution of SLEEP instruction in high-speed mode or medium-speed mode Transition to subsleep mode after execution of SLEEP instruction in subactive mode |
| 1 | Transition to standby mode, subactive mode, or watch mode after execution of SLEEP instruction in high-speed mode or medium-speed mode Transition to watch mode or high-speed mode after execution of SLEEP instruction in subactive mode |

H'FFEB: Low-Power Control Register LPWRCR: System Control

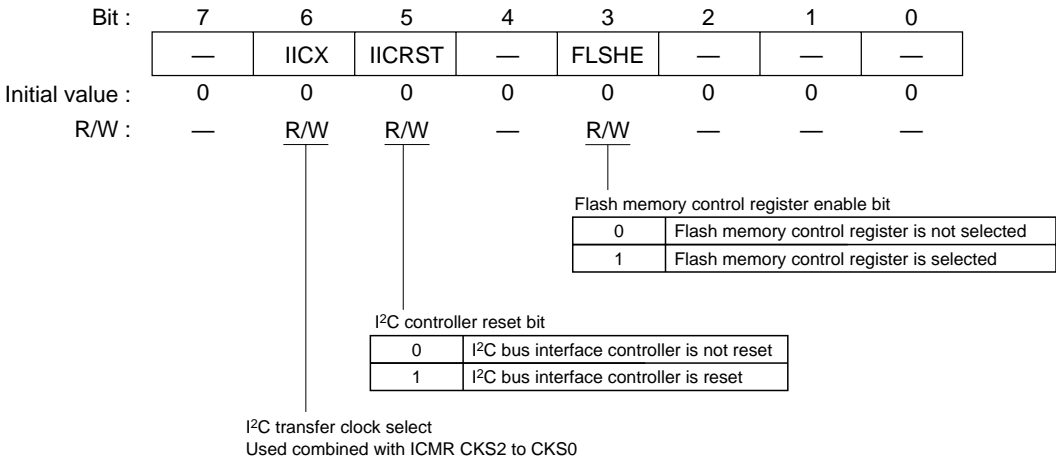


H'FFEC: Module Stop Control Register MSTPCRH: System Control

H'FFED: Module Stop Control Register MSTPCRL: System Control



H'FFEE: Serial Timer Control Register STCR: System Control



H'FFF0: IRQ Edge Select Register IEGR: Interrupt Controller

| | | | | | | | | |
|-----------------|---|--------|--------|--------|--------|--------|---------|---------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | IRQ5EG | IRQ4EG | IRQ3EG | IRQ2EG | IRQ1EG | IRQ0EG1 | IRQ0EG2 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

IRQ0 pin detected edge select bits

| IRQ0EG1 | IRQ0EG0 | IRQ0 pin detected edge select |
|---------|---------|---|
| 0 | 0 | Interrupt request generated at falling edge of IRQ0 pin input |
| 0 | 1 | Interrupt request generated at rising edge of IRQ0 pin input |
| 1 | * | Interrupt request generated at both falling and rising edge of IRQ0 pin input |

Note: * Don't care.

IRQ5 to IRQ1 pins detected edge select bits

| | |
|---|---|
| 0 | Interrupt request generated at falling edge of IRQn pin input |
| 1 | Interrupt request generated at rising edge of IRQn pin input |

(n = 5 to 1)

H'FFF1: IRQ Enable Register IENR: Interrupt Controller

| | | | | | | | | |
|-----------------|---|---|-------|-------|-------|-------|-------|-------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | IRQ5E | IRQ4E | IRQ3E | IRQ2E | IRQ1E | IRQ0E |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

IRQ5 to IRQ0 enable bits

| | |
|---|----------------------------|
| 0 | IRQn interrupt is disabled |
| 1 | IRQn interrupt is enabled |

(n = 5 to 0)

H'FFF2: IRQ Status Register IRQR: Interrupt Controller

| | | | | | | | | |
|-----------------|---|---|--------|--------|--------|--------|--------|--------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | IRQ5F | IRQ4F | IRQ3F | IRQ2F | IRQ1F | IRQ0F |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

IRQ5 to IRQ0 flag

| | |
|---|---|
| 0 | [Clearing conditions] Cleared by reading IRQnF set to 1, then writing 0 in IRQnF When IRQn interrupt exception handling is executed |
| 1 | [Setting conditions] (1) When a falling edge occurs in $\overline{\text{IRQn}}$ input while falling edge detection is set (IRQnEG = 0) (2) When a rising edge occurs in $\overline{\text{IRQn}}$ input while rising edge detection is set (IRQnEG = 0) (3) When a falling or rising edge occurs in IRQ0 input while both-edge detection is set (IRQ0EG1 = 1) |

(n = 5 to 0)

Note: * Only 0 can be written to clear the flag.

H'FFF3: Interrupt Control Register A ICRA: Interrupt Controller

H'FFF4: Interrupt Control Register B ICRB: Interrupt Controller

H'FFF5: Interrupt Control Register C ICRC: Interrupt Controller

H'FFF6: Interrupt Control Register D ICRD: Interrupt Controller

| | | | | | | | | |
|-----------------|------|------|------|------|------|------|------|------|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ICR7 | ICR6 | ICR5 | ICR4 | ICR3 | ICR2 | ICR1 | ICR0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

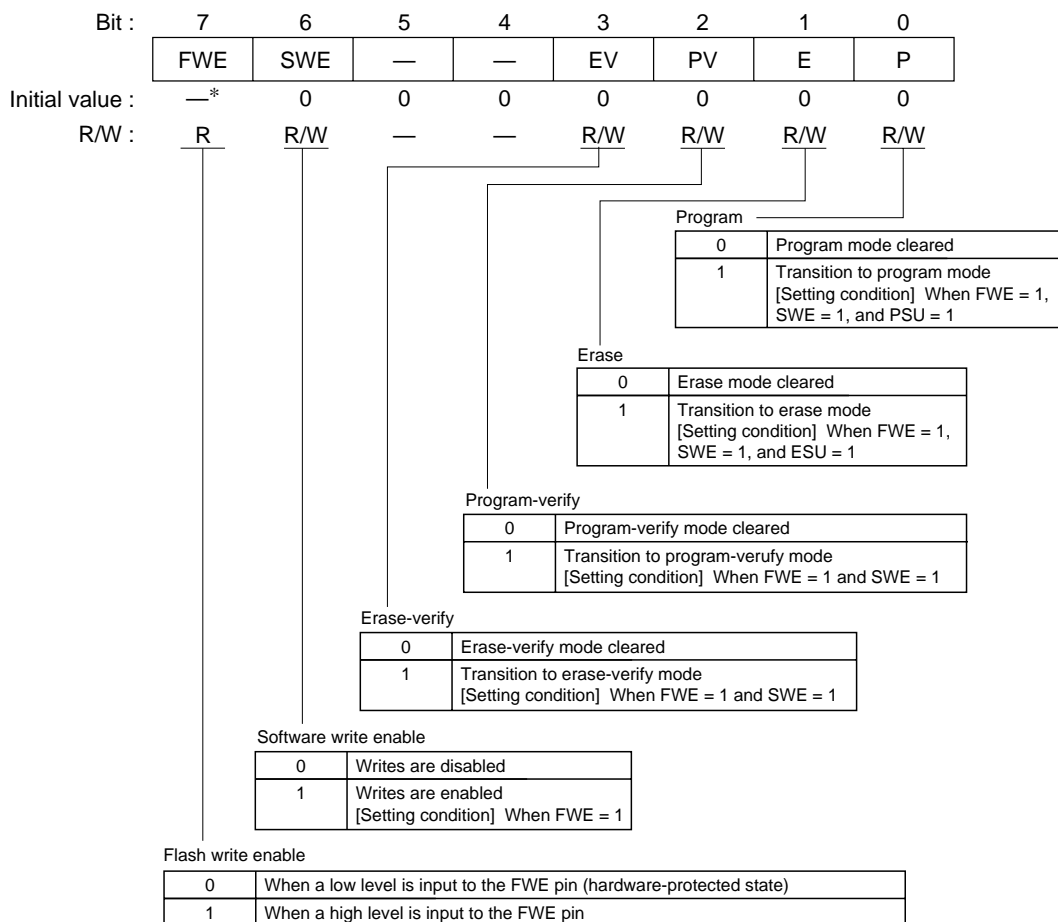
Interrupt control level

| | |
|---|--|
| 0 | Corresponding interrupt source is control level 0 (non-priority) |
| 1 | Corresponding interrupt source is control level 1 (priority) |

(n = 7 to 0)

H'FFF8: Flash Memory Control Register 1 FLMCR1:

FLASH ROM (H8S/2194 FLASH Version Only)



Note: * Determined by the state of the FWE pin.

H'FFF8: Flash Memory Control Register 1 FLMCR1:

FLASH ROM (H8S/2194C FLASH Version Only)

| | | | | | | | | |
|-----------------|-----|-----|------|------|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FWE | SWE | ESU1 | PSU1 | EV1 | PV1 | E1 | P1 |
| Initial value : | —* | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| | |
|---------|---|
| Program | |
| 0 | Program mode cleared |
| 1 | Transition to program mode [Setting condition] When FWE = 1, SWE = 1, and PSU = 1 |

| | |
|-------|---|
| Erase | |
| 0 | Erase mode cleared |
| 1 | Transition to erase mode [Setting condition] When FWE = 1, SWE = 1, and ESU = 1 |

| | |
|----------------|---|
| Program verify | |
| 0 | Program-verify mode cleared |
| 1 | Transition to program-verify mode [Setting condition] When FWE = 1 and SWE = 1 |

| | |
|--------------|---|
| Erase verify | |
| 0 | Erase-verify mode cleared |
| 1 | Transition to erase-verify mode [Setting condition] When FWE = 1 and SWE = 1 |

| | |
|---------------------|---|
| Program setup bit 1 | |
| 0 | Program-setup cleared |
| 1 | Program-setup [Setting condition] When FWE = 1 and SWE = 1 |

| | |
|-------------------|---|
| Erase setup bit 1 | |
| 0 | Erase-setup cleared |
| 1 | Erase-setup [Setting condition] When FWE = 1 and SWE = 1 |

| | |
|-----------------------|--|
| Software write enable | |
| 0 | Writes are disabled |
| 1 | Writes are enabled [Setting condition] When FWE = 1 |

| | |
|--------------------|---|
| Flash write enable | |
| 0 | When a low level is input to the FWE pin (hardware-protected state) |
| 1 | When a high level is input to the FWE pin |

Note: * Determined by the state of the FWE pin.

H'FFF9: Flash Memory Control Register 2 FLMCR2:**FLASH ROM (H8S/2194 FLASH Version Only)**

| | | | | | | | | |
|-----------------|------|---|---|---|---|---|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FLER | — | — | — | — | — | ESU | PSU |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R | — | — | — | — | — | R/W | R/W |

Program setup

| | |
|---|--|
| 0 | Program setup cleared |
| 1 | Program setup [Setting condition] When FWE = 1, and SWE = 1 |

Erase setup

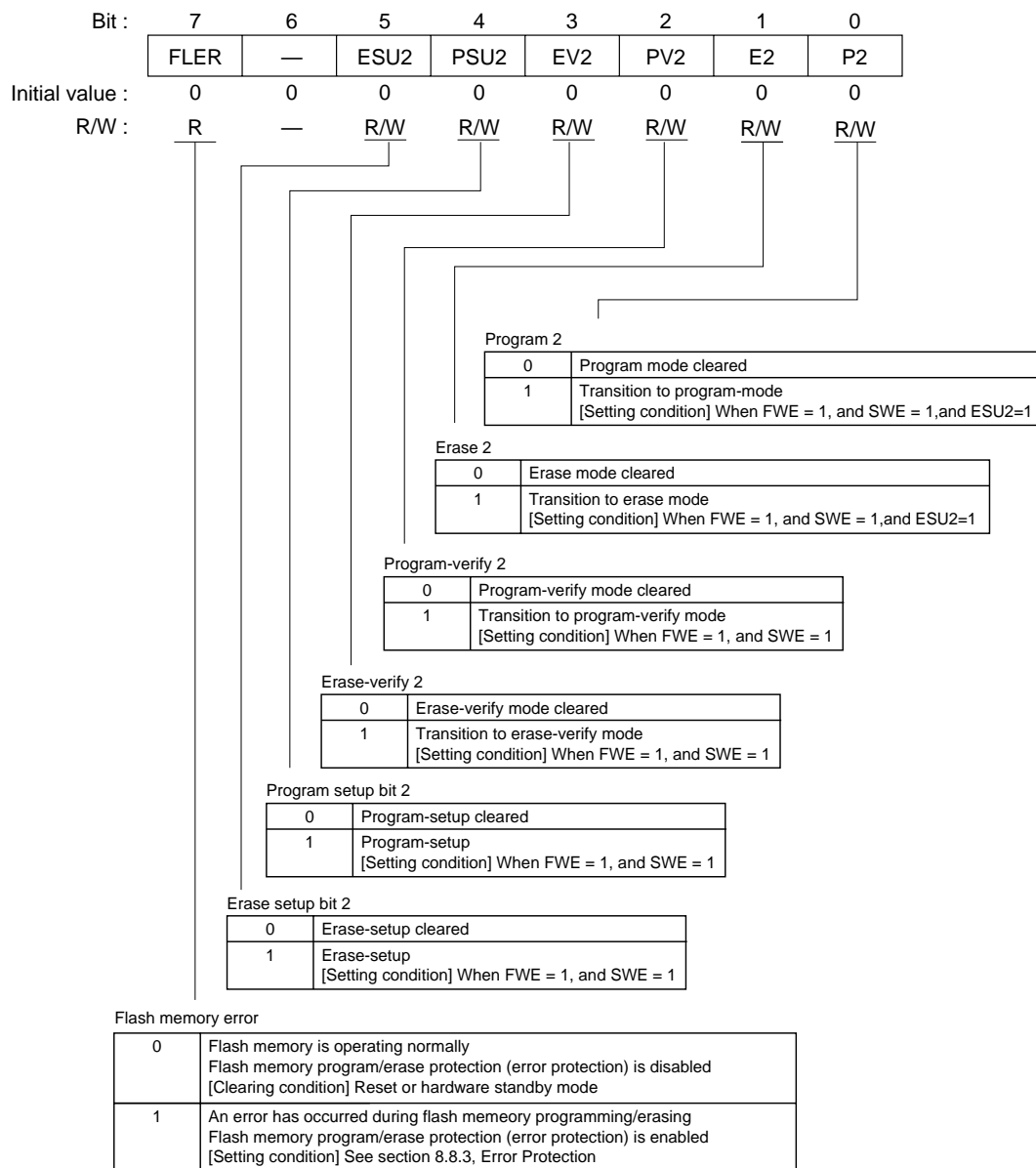
| | |
|---|--|
| 0 | Erase setup cleared |
| 1 | Erase setup [Setting condition] When FWE = 1, and SWE = 1 |

Flash memory error

| | |
|---|---|
| 0 | Flash memory is operating normally Flash memory program/erase protection (error protection) is disabled [Clearing condition] Reset or hardware standby mode |
| 1 | An error has occurred during flash memory programming/erasing Flash memory program/erase protection (error protection) is enabled [Setting condition] See section 7.8.3, Error Protection |

H'FFF9: Flash Memory Control Register 2 FLMCR2:

FLASH ROM (H8S/2194C FLASH Version Only)



H'FFFA: Erase Block Select Register 1 EBR1:**FLASH ROM (H8S/2194 FLASH Version Only)**

| | | | | | | | | |
|-----------------|---|---|---|---|---|---|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EBR1 | — | — | — | — | — | — | EB9 | EB8 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | — | — | — | — | R/W | R/W |

H'FFFA: Erase Block Select Register 1 EBR1:**FLASH ROM (H8S/2194C FLASH Version Only)**

| | | | | | | | | |
|-----------------|---|---|------|------|------|------|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EBR1 | — | — | EB13 | EB12 | EB11 | EB10 | EB9 | EB8 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | — | — | R/W | R/W | R/W | R/W | R/W | R/W |

H'FFFB: Erase Block Select Register 2 EBR2:**FLASH ROM (H8S/2194 FLASH Version Only)**

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EBR2 | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Division of Erase Block

| Block (size) | Address |
|-------------------|----------------------|
| 128-kbyte version | |
| EB0 (1k bytes) | H'000000 to H'0003FF |
| EB1 (1k bytes) | H'000400 to H'0007FF |
| EB2 (1k bytes) | H'000800 to H'000BFF |
| EB3 (1k bytes) | H'000C00 to H'000FFF |
| EB4 (28k bytes) | H'001000 to H'007FFF |
| EB5 (16k bytes) | H'008000 to H'00BFFF |
| EB6 (8k bytes) | H'00C000 to H'00DFFF |
| EB7 (8k bytes) | H'00E000 to H'00FFFF |
| EB8 (32k bytes) | H'010000 to H'017FFF |
| EB9 (32k bytes) | H'018000 to H'01FFFF |

H'FFFB: Erase Block Select Register 2 EBR2:**FLASH ROM (H8S/2194C FLASH Version Only)**

| | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EBR2 | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W : | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Division of Erase Block

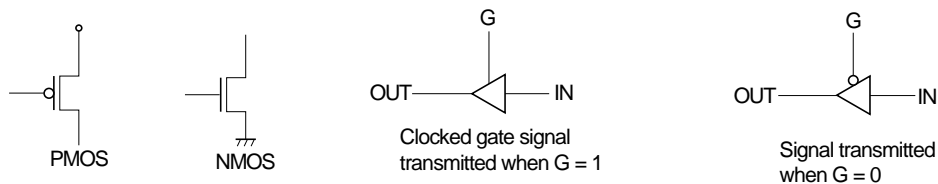
| Block (size) | Address |
|-------------------|----------------------|
| 256-kbyte version | |
| EB0 (1k bytes) | H'000000 to H'0003FF |
| EB1 (1k bytes) | H'000400 to H'0007FF |
| EB2 (1k bytes) | H'000800 to H'000BFF |
| EB3 (1k bytes) | H'000C00 to H'000FFF |
| EB4 (28k bytes) | H'001000 to H'007FFF |
| EB5 (16k bytes) | H'008000 to H'00BFFF |
| EB6 (8k bytes) | H'00C000 to H'00DFFF |
| EB7 (8k bytes) | H'00E000 to H'00FFFF |
| EB8 (32k bytes) | H'010000 to H'017FFF |
| EB9 (32k bytes) | H'018000 to H'01FFFF |
| EB10 (32k bytes) | H'020000 to H'027FFF |
| EB11 (32k bytes) | H'028000 to H'02FFFF |
| EB12 (32k bytes) | H'030000 to H'037FFF |
| EB13 (32k bytes) | H'038000 to H'03FFFF |

Appendix C Pin Circuit Diagrams

C.1 Pin Circuit Diagrams

Circuit diagrams for all pins except power supply pins are shown in table C.1.

Legend



[Symbols]

RD: Read signal

RST: Reset signal

LPM: Power-down mode signal (1 in standby, watch, and subactive modes)

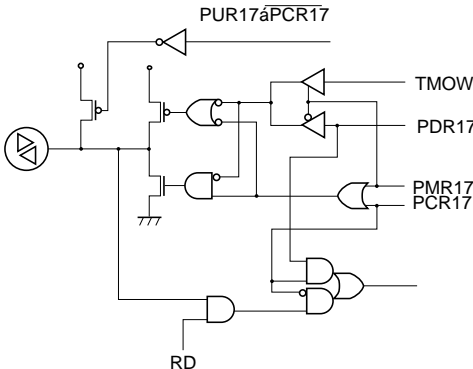
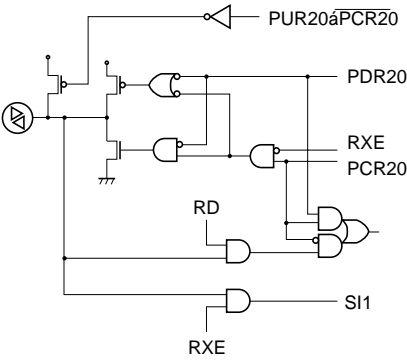
Hi-Z: High impedance

SLEEP: Sleep mode signal

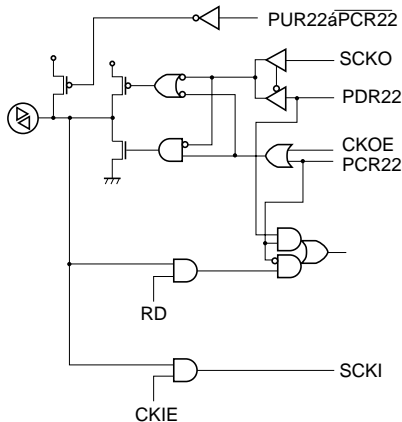
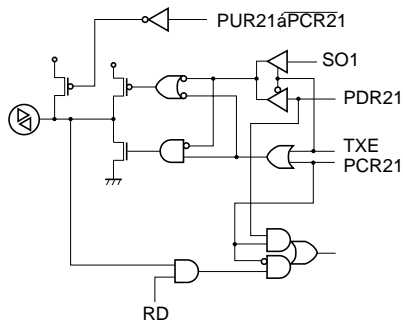
Note: Numbers given for resistance values, etc., are reference values.

Table C.1 Pin Circuit Diagrams

| | | Pin States | | |
|---|--|------------|------------|--|
| Pin Names | Circuit Diagram | At Reset | Sleep Mode | Power-Down Modes Other Than Sleep Mode |
| | | | | |
| P00/AN0 to P07/AN7 | <p>PMR0n-RD</p> <p>SCH3 to SCH0</p> | Hi-Z | Retained | Hi-Z |
| AN8 to ANB | <p>HCH1, HCH0</p> | Hi-Z | Retained | Hi-Z |
| P10/ $\overline{\text{IRQ0}}$ to P15/ $\overline{\text{IRQ5}}$ P16/ $\overline{\text{IC}}$ | <p>PUR1n$\overline{\text{PCR1n}}$</p> <p>PDR1n</p> <p>PMR1n</p> <p>PCR1n</p> <p>RD</p> <p>INT</p> <p>PMR1n</p> <p>INT = $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ5}}$, $\overline{\text{IC}}$</p> <p>n = 0 to 6</p> | Hi-Z | Retained | Pull-up MOS: OFF Subactive mode: Functions Other modes: Hi-Z |

| | | Pin States | | |
|-----------|---|------------|------------|--|
| Pin Names | Circuit Diagram | At Reset | Sleep Mode | Power-Down Modes Other Than Sleep Mode |
| P17/TMOW |  | Hi-Z | Retained | Pull-up MOS: OFF Subactive mode: Functions Other modes: Hi-Z |
| P20/SI1 |  <p>RXE: Input control signal determined by SCR and SMR.</p> | Hi-Z | Retained | Pull-up MOS: OFF Subactive mode: Functions Other modes: Hi-Z When SI1 input is selected, pin input should be fixed high or low. |

| Pin Names | Circuit Diagram | At Reset | Sleep Mode | Power-Down Modes Other Than Sleep Mode |
|-----------|--|----------|------------|--|
| P21/SO1 | <p>TXE: Output control signal determined by SCR and SMR.</p> | Hi-Z | Retained | Pull-up MOS: OFF Subactive mode: Functions Other modes: Hi-Z |
| P22/SCK1 | <p>SCKO: Transfer clock output SCKI: Transfer clock input CKOE: Transfer clock output control signal determined by SMR CKIE: Transfer clock input control signal determined by SMR and SCR</p> | Hi-Z | Retained | Pull-up MOS: OFF Subactive mode: Functions Other modes: Hi-Z |



Pin States

| Pin Names | Circuit Diagram | At Reset | Sleep Mode | Power-Down Modes Other Than Sleep Mode |
|--------------------|---|----------|------------|---|
| P23/SDA P24/SCL | <p>PUR2n$\overline{aPMCR2n}$</p> <p>PDR2n</p> <p>IICE PCR2n</p> <p>RD</p> <p>IICE SDA/SCL</p> <p>SDA/SCL</p> <p>IICE</p> <p>n = 3, 4</p> <p>IICE: I²C bus enable signal</p> | Hi-Z | Retained | Pull-up MOS: OFF Subactive mode: Functions Other modes: Hi-Z |
| P26/SO2 | <p>PUR26$\overline{aPMCR26}$</p> <p>SO2</p> <p>PDR26</p> <p>PDR26 PCR26</p> <p>RD</p> | Hi-Z | Retained | Pull-up MOS: OFF Subactive mode: Functions Other modes: Hi-Z |

HITACHI

Pin States

| Pin Names | Circuit Diagram | At Reset | Sleep Mode | Power-Down Modes Other Than Sleep Mode |
|---|---|----------|------------|--|
| P30/ \overline{CS} | | Hi-Z | Retained | Pull-up MOS: OFF Subactive mode: Functions Other modes: Hi-Z |
| P31/STRB P32/PWM0 P33/PWM1 P34/PWM2 P35/PWM3 P36/BUZZ P37/TMO | <p>OUR:</p> <p>P31/STRB: SC12 strobe output P32/PWM0: 8-bit PWM0 output P33/PWM1: 8-bit PWM1 output P34/PWM2: 8-bit PWM2 output P35/PWM3: 8-bit PWM3 output P36/BUZZ: Timer J buzzer output P37/TMO: Timer J timer output</p> | Hi-Z | Retained | Pull-up MOS: OFF Subactive mode: Functions Other modes: Hi-Z |

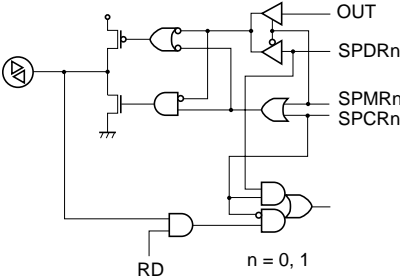
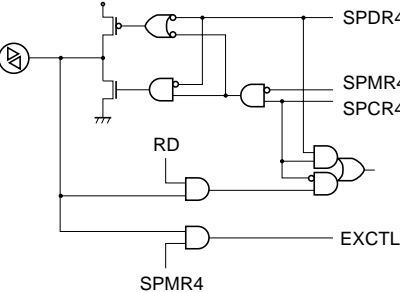
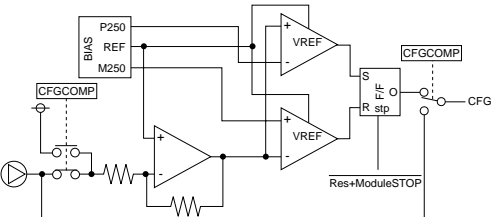
Pin States

| Pin Names | Circuit Diagram | At Reset | Sleep Mode | Power-Down Modes Other Than Sleep Mode |
|-----------|---|----------|------------|--|
| P47 | | Hi-Z | Retained | Subactive mode: Functions Other modes: Hi-Z |
| P50/ADTRG | <p>TRGE: A/D trigger input control signal</p> | Hi-Z | Retained | Subactive mode: Functions Other modes: Hi-Z When ADTRG input is selected, pin input should be fixed high or low. |
| P51 | | Hi-Z | Retained | Subactive mode: Functions Other modes: Hi-Z |

Pin States

| Pin Names | Circuit Diagram | At Reset | Sleep Mode | Power-Down Modes Other Than Sleep Mode |
|-------------------------|---|------------------|---|--|
| COMP/PS2 | | Hi-Z | Hi-Z | Hi-Z |
| | | | When COMP input is selected, pin input should be fixed high or low. | |
| AUDIOFF VIDEOFF | | Hi-Z | Hi-Z | Hi-Z |
| | LPM: power-down mode signal | | | |
| CAPPWM DRMPWM | | Low output | Low output | Low output |
| Vpulse | | Low output | Low output | Low output |
| | Note: Resistance values are reference value | | | |
| $\overline{\text{RES}}$ | | Low input (High) | | (High) |
| MD0 | | — | | |
| NMI | | | | When $\overline{\text{NMI}}$ is not used, pin input should be fixed high or low. |

Pin States

| Pin Names | Circuit Diagram | At Reset | Sleep Mode | Power-Down Modes Other Than Sleep Mode |
|---------------------------------|--|----------|------------|--|
| C.Rotary/PS0 H.Ampsw/ PS1 |  <p data-bbox="350 509 573 531">OUT = C.Rotary, H.Ampsw</p> | Hi-Z | Hi-Z | Hi-Z |
| EXCTL/PS4 |  | Hi-Z | Hi-Z | Hi-Z |
| CFG |  | — | — | — |

When EXCTL input is selected, pin input should be fixed high or low.

| Pin Names | Circuit Diagram | At Reset | Sleep Mode | Power-Down Modes Other Than Sleep Mode |
|--|--|---|-------------|--|
| DFG DPG/PS3 | | Hi-Z | — | Hi-Z |
| CTL (+) CTL (-) CTLREF CTLBias CTLFB CTLAmp (O) CTLSMT (i) | | — | — | — |
| X2 | | Oscillation | Oscillation | Oscillation |
| X1 | <p>Note: Resistance values are reference values.</p> | When the subclock is not used, set X1 = high and X2 = open. | | |
| OSC2 | | Oscillation | Oscillation | Low output |
| OSC1 | | | | — |

Appendix D Port States in the Difference Processing States

D.1 Pin Circuit Diagrams

Table D.1 Port States Overview

| Port | Reset | Active | Sleep | Standby | Watch | Subactive | Subsleep |
|-------------|---------------------|---------------------|---------------------|---------------------|---------------------|------------------|-----------------|
| P07 to P00 | High imped- ance | High imped- ance | High imped- ance | High imped- ance | High imped- ance | High impedance | High impedance |
| P17 to P10 | High imped- ance | Functions | Retained | High imped- ance | High imped- ance | Functions | Retained |
| P27 to P20 | High imped- ance | Functions | Retained | High imped- ance | High imped- ance | Functions | Retained |
| P37 to P30 | High imped- ance | Functions | Retained | High imped- ance | High imped- ance | Functions | Retained |
| P47 to P40 | High imped- ance | Functions | Retained | High imped- ance | High imped- ance | Functions | Retained |
| P53 to P50 | High imped- ance | Functions | Retained | High imped- ance | High imped- ance | Functions | Retained |
| P67 to P60 | High imped- ance | Functions | Retained | High imped- ance | High imped- ance | Functions | Retained |
| P77 to P70 | High imped- ance | Functions | Retained | High imped- ance | High imped- ance | Functions | Retained |
| P87 to P80 | High imped- ance | Functions | Retained | High imped- ance | High imped- ance | Functions | Retained |

Appendix E Usage Notes

E.1 Power Supply Rise and Fall Order

Figure E.1 shows the order in which the power supply pins rise when the chip is powered on, and the order in which they fall when the chip is powered down. If the power supply voltages cannot rise and fall simultaneously, power supply operations should be carried out in this order.

At power-on, wait until the microcomputer section power supply (V_{CC}) has risen to the prescribed voltage, then raise the other analog power supplies.

At power-down, drop the analog power supplies first, followed by the microcomputer section power supply (V_{CC}).

When powering up and down, ensure that the voltage applied to the pins does not exceed the respective power supply voltage.

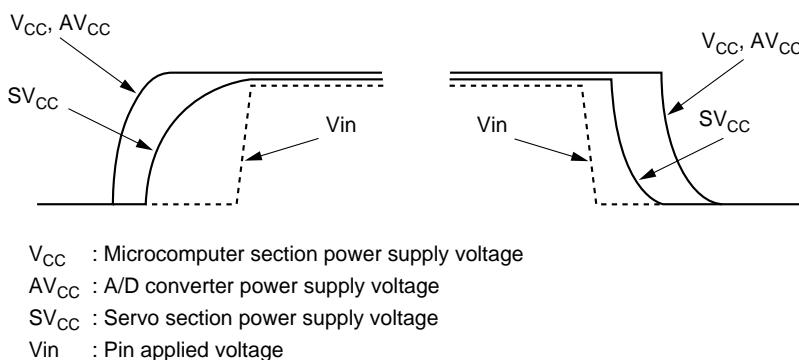


Figure E.1 Power Supply Rise and Fall Order

In power-down modes (except sleep mode), the analog power supplies can be turned off to reduce current dissipation. When the microcomputer section power supply (V_{CC}) is dropped to the backup voltage in a power-down mode, the order shown in figure E.2 should be followed. Make sure that the voltage applied to the pins does not exceed the respective power supply voltage.

The A/D converter power supply (AV_{CC}) should be set to the same potential as the microcomputer section power supply (V_{CC}). In all power-down modes except sleep mode, AV_{CC} is turned off inside the device. At this time, the AV_{CC} current dissipation is defined as AISTOP.

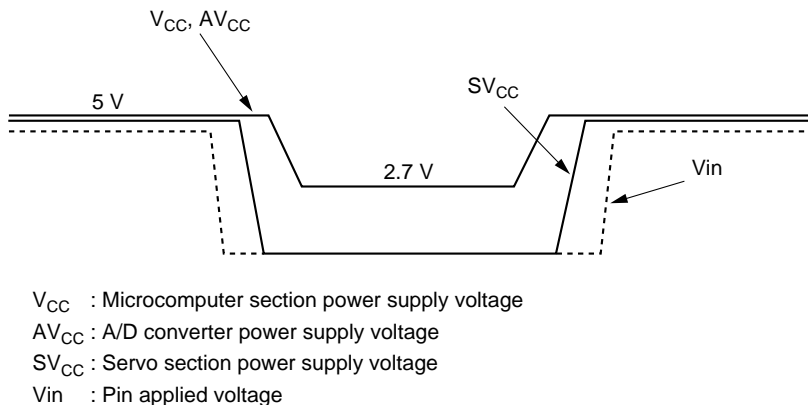


Figure E.2 Power Supply Control in Power-Down Modes

E.2 Pin Handling When the High-Speed Switching Circuit for Four-Head Special Playback Is Not Used

Table E.1 shows how the C.Rotary, H.AmpSW, and COMP pins should be handled when the switching circuit for four-head special-effects playback is not used. COMP is an input pin, and the other two are output pins.

When the switching circuit for four-head special-effects playback is not used, the related pins should be handled as shown below.

Table E.1 Pin Handling When the High-Speed Switching Circuit for Four-Head Special Playback Is Not Used

| Pin No. | Pin Name | Connection |
|---------|----------|--------------------|
| 103 | C.Rotary | OPEN (output pin)* |
| 104 | H.AMP SW | OPEN (output pin)* |
| 105 | COMP | V_{SS} |

Note: * Output depends on the special-effects control register (CHCR) value. If the initial value is used, the output is low-level.

E.3 Sample External Circuits

Examples of external circuits for the servo section, and sync signal detection circuit are shown in figures E.3, E.4.

(1) Servo Section

An example of the external circuit for the DRMPWM output and CAPPWM output pins is shown in figure E.3.

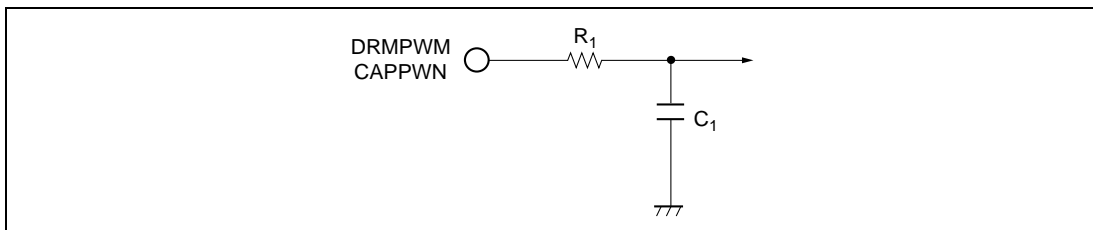


Figure E.3 Sample External Circuit for Servo Section

(2) Sync Signal Detection Circuit Section

Figure E.4 shows an example of the external circuit for the sync signal detection circuit section.

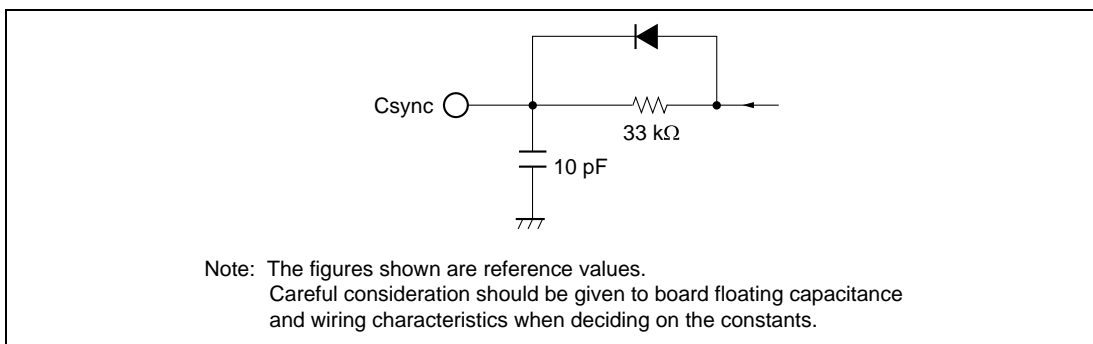


Figure E.4 Example of External Circuit for Sync Signal Detection Circuit Section

Appendix F List of Product Codes

Table F.1 Product Codes List of H8S/2194 Series and H8S/2194C Series

| Product Type | | | Product Code | Mark Code | Package (Hitachi Package Code) |
|------------------|-----------|------------------|--------------|-------------------|--------------------------------|
| H8S/2194 Series | H8S/2194 | Mask ROM version | HD6432194 | HD6432194 (***)F | 112-pin QFP (FP-112) |
| | | F-ZTAT version | HD64F2194 | HD64F2194F | 112-pin QFP (FP-112) |
| | H8S/2193 | Mask ROM version | HD6432193 | HD6432193 (***)F | 112-pin QFP (FP-112) |
| | H8S/2192 | Mask ROM version | HD6432192 | HD6432192 (***)F | 112-pin QFP (FP-112) |
| | H8S/2191 | Mask ROM version | HD6432191 | HD6432191 (***)F | 112-pin QFP (FP-112) |
| H8S/2194C Series | H8S/2194C | Mask ROM version | HD6432194C | HD6432194C (***)F | 112-pin QFP (FP-112) |
| | | F-ZTAT version | HD64F2194C | HD64F2194CF | 112-pin QFP (FP-112) |
| | H8S/2194B | Mask ROM version | HD6432194B | HD6432194B (***)F | 112-pin QFP (FP-112) |
| | H8S/2194A | Mask ROM version | HD6432194A | HD6432194A (***)F | 112-pin QFP (FP-112) |

Note: (**) is the ROM code.

Appendix G External Dimensions

Unit: mm

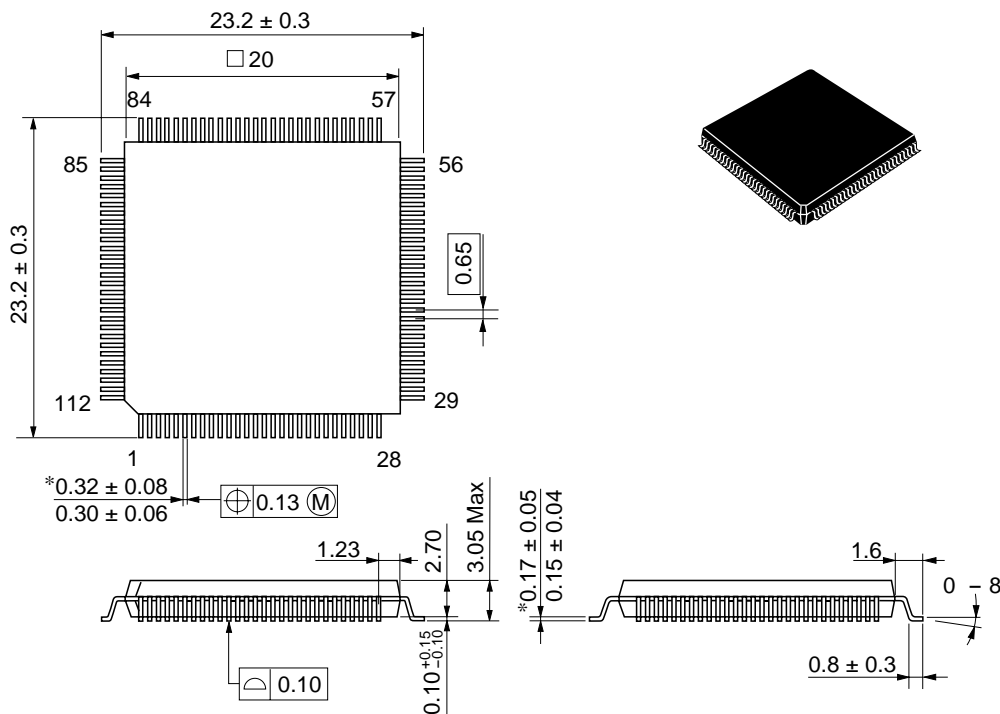

$$\frac{\text{*Dimension including the plating thickness}}{\text{Base material dimension}}$$

Figure G.1 External Dimensions (FP-112)

**H8S/2194 Series, H8S/2194C Series, H8S/2194 F-ZTATTM,
H8S/2194C F-ZTATTM Hardware Manual**

Publication Date: 1st Edition, November 1998

2nd Edition, November 2000

Published by: Electronic Devices Sales & Marketing Group
Semiconductor & Integrated Circuits
Hitachi, Ltd.

Edited by: Technical Documentation Group
Hitachi Kodaira Semiconductor Co., Ltd.

Copyright © Hitachi, Ltd., 1998. All rights reserved. Printed in Japan.