

To all our customers

Regarding the change of names mentioned in the document, such as Hitachi Electric and Hitachi XX, to Renesas Technology Corp.

The semiconductor operations of Mitsubishi Electric and Hitachi were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Hitachi, Hitachi, Ltd., Hitachi Semiconductors, and other Hitachi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Renesas Technology Home Page: <http://www.renesas.com>

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003

Cautions

Keep safety first in your circuit designs!

1. Renesas Technology Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corporation product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corporation or a third party.
2. Renesas Technology Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor for the latest product information before purchasing a product listed herein.
The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
Please also pay attention to information published by Renesas Technology Corporation by various means, including the Renesas Technology Corporation Semiconductor home page (<http://www.renesas.com>).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corporation or an authorized Renesas Technology Corporation product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corporation is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corporation for further details on these materials or the products contained therein.

Hitachi SuperH™ RISC engine

SH7014, SH7016,

SH7017F-ZTAT™

Hardware Manual

RENESAS

ADE-602-128C

Rev. 4.0

3/13/03

Hitachi, Ltd.

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.

Preface

The SH7014/16/17 CMOS single-chip microprocessors integrate a Hitachi-original architecture, high-speed CPU with peripheral functions required for system configuration.

The CPU has a RISC-type instruction set. Most instructions can be executed in one clock cycle, which greatly improves instruction execution speed. In addition, the 32-bit internal-bus architecture enhances data processing power. With this CPU, it has become possible to assemble low cost, high performance/high-functioning systems, even for applications that were previously impossible with microprocessors, such as real-time control, which demands high speeds. In particular, this LSI has a 1-kbyte on-chip cache, which allows an improvement in CPU performance during external memory access.

In addition, this LSI includes on-chip peripheral functions necessary for system configuration, such as large-capacity ROM (except the SH7014, which is ROMless) and RAM, timers, a serial communication interface (SCI), an A/D converter, an interrupt controller, and I/O ports. Memory or peripheral LSIs can be connected efficiently with an external memory access support function. This greatly reduces system cost.

This LSI has an F-ZTAT™ version with on-chip flash memory and a mask ROM version. These versions enable users to respond quickly and flexibly to changing application specifications, growing production volumes, and other conditions.

This hardware manual covers the SH7014/16/17/. For a detailed description of instructions, refer to the programming manual.

Related Manuals

SH7014/16/17 instruction execution: *SH-1/SH-2/SH-DSP Programming Manual*

For information on development systems, please contact a Hitachi sales representative.

Contents

| | | |
|-----------|--|----|
| Section 1 | SH7014/16/17 Overview..... | 1 |
| 1.1 | SH7014/16/17 Overview..... | 1 |
| 1.1.1 | SH7014/16/17 Series Features..... | 1 |
| 1.2 | Block Diagram..... | 6 |
| 1.3 | Pin Arrangement and Pin Functions..... | 8 |
| 1.3.1 | Pin Arrangement..... | 8 |
| 1.3.2 | Pin Arrangement by Mode..... | 10 |
| 1.3.3 | Pin Functions..... | 14 |
| Section 2 | CPU..... | 19 |
| 2.1 | Register Configuration..... | 19 |
| 2.1.1 | General Registers (Rn)..... | 19 |
| 2.1.2 | Control Registers..... | 20 |
| 2.1.3 | System Registers..... | 21 |
| 2.1.4 | Initial Values of Registers..... | 21 |
| 2.2 | Data Formats..... | 22 |
| 2.2.1 | Data Format in Registers..... | 22 |
| 2.2.2 | Data Format in Memory..... | 22 |
| 2.2.3 | Immediate Data Format..... | 22 |
| 2.3 | Instruction Features..... | 23 |
| 2.3.1 | RISC-Type Instruction Set..... | 23 |
| 2.3.2 | Addressing Modes..... | 26 |
| 2.3.3 | Instruction Format..... | 30 |
| 2.4 | Instruction Set by Classification..... | 33 |
| 2.5 | Processing States..... | 46 |
| 2.5.1 | State Transitions..... | 46 |
| 2.5.2 | Power-Down State..... | 48 |
| Section 3 | Operating Modes..... | 51 |
| 3.1 | Operating Modes, Types, and Selection..... | 51 |
| 3.2 | Explanation of Operating Modes..... | 52 |
| 3.3 | Pin Configuration..... | 52 |
| Section 4 | Clock Pulse Generator (CPG)..... | 53 |
| 4.1 | Overview..... | 53 |
| 4.1.1 | Block Diagram..... | 53 |
| 4.2 | Oscillator..... | 53 |
| 4.2.1 | Connecting a Crystal Oscillator..... | 53 |
| 4.2.2 | External Clock Input Method..... | 56 |

| | | |
|------------------|---|-----------|
| 4.2.3 | Prescaler | 56 |
| Section 5 | Exception Processing | 57 |
| 5.1 | Overview | 57 |
| 5.1.1 | Types of Exception Processing and Priority | 57 |
| 5.1.2 | Exception Processing Operations..... | 58 |
| 5.1.3 | Exception Processing Vector Table | 59 |
| 5.2 | Resets..... | 61 |
| 5.2.1 | Power-on Reset | 61 |
| 5.3 | Address Errors..... | 62 |
| 5.3.1 | Address Error Exception Processing..... | 63 |
| 5.4 | Interrupts | 63 |
| 5.4.1 | Interrupt Priority Level..... | 63 |
| 5.4.2 | Interrupt Exception Processing | 64 |
| 5.5 | Exceptions Triggered by Instructions | 64 |
| 5.5.1 | Trap Instructions | 65 |
| 5.5.2 | Illegal Slot Instructions | 65 |
| 5.5.3 | General Illegal Instructions | 66 |
| 5.6 | When Exception Sources Are Not Accepted..... | 66 |
| 5.6.1 | Immediately after a Delayed Branch Instruction | 66 |
| 5.6.2 | Immediately after an Interrupt-Disabled Instruction..... | 66 |
| 5.7 | Stack Status after Exception Processing Ends..... | 67 |
| 5.8 | Notes on Use | 68 |
| 5.8.1 | Value of Stack Pointer (SP) | 68 |
| 5.8.2 | Value of Vector Base Register (VBR)..... | 68 |
| 5.8.3 | Address Errors Caused by Stacking of Address Error Exception Processing | 68 |
| Section 6 | Interrupt Controller (INTC)..... | 69 |
| 6.1 | Overview | 69 |
| 6.1.1 | Features | 69 |
| 6.1.2 | Block Diagram | 69 |
| 6.1.3 | Pin Configuration..... | 71 |
| 6.1.4 | Register Configuration..... | 71 |
| 6.2 | Interrupt Sources | 72 |
| 6.2.1 | NMI Interrupts | 72 |
| 6.2.2 | IRQ Interrupts | 72 |
| 6.2.3 | On-Chip Peripheral Module Interrupts | 73 |
| 6.2.4 | Interrupt Exception Vectors and Priority Rankings..... | 73 |
| 6.3 | Description of Registers | 76 |
| 6.3.1 | Interrupt Priority Registers A–H (IPRA–IPRH) | 76 |
| 6.3.2 | Interrupt Control Register (ICR) | 78 |
| 6.3.3 | IRQ Status Register (ISR) | 79 |
| 6.4 | Interrupt Operation | 80 |

| | | |
|---|--|-----------|
| 6.4.1 | Interrupt Sequence | 80 |
| 6.4.2 | Stack after Interrupt Exception Processing | 82 |
| 6.5 | Interrupt Response Time | 82 |
| 6.6 | Data Transfer with Interrupt Request Signals | 84 |
| 6.6.1 | Handling DMAC Activating Sources but Not CPU Interrupt Sources | 85 |
| 6.6.2 | Treating CPU Interrupt Sources but Not DMAC Activating Sources | 85 |
| Section 7 Cache Memory (CAC) | | 87 |
| 7.1 | Overview | 87 |
| 7.1.1 | Features | 87 |
| 7.1.2 | Block Diagram | 88 |
| 7.1.3 | Register Configuration | 88 |
| 7.2 | Register Explanation | 89 |
| 7.2.1 | Cache Control Register (CCR) | 89 |
| 7.3 | Address Array and Data Array | 90 |
| 7.3.1 | Cache Address Array Read/Write Space | 91 |
| 7.3.2 | Cache Data Array Read/Write Space | 91 |
| 7.4 | Cautions on Use | 92 |
| 7.4.1 | Cache Initialization | 92 |
| 7.4.2 | Forced Access to Address Array and Data Array | 92 |
| 7.4.3 | Cache Miss Penalty and Cache Fill Timing | 92 |
| 7.4.4 | Cache Hit after Cache Miss | 94 |
| Section 8 Bus State Controller (BSC) | | 95 |
| 8.1 | Overview | 95 |
| 8.1.1 | Features | 95 |
| 8.1.2 | Block Diagram | 96 |
| 8.1.3 | Pin Configuration | 97 |
| 8.1.4 | Register Configuration | 97 |
| 8.1.5 | Address Map | 98 |
| 8.2 | Description of Registers | 101 |
| 8.2.1 | Bus Control Register 1 (BCR1) | 101 |
| 8.2.2 | Bus Control Register 2 (BCR2) | 103 |
| 8.2.3 | Wait Control Register 1 (WCR1) | 106 |
| 8.2.4 | Wait Control Register 2 (WCR2) | 107 |
| 8.2.5 | DRAM Area Control Register (DCR) | 108 |
| 8.2.6 | Refresh Timer Control/Status Register (RTC SR) | 111 |
| 8.2.7 | Refresh Timer Counter (RTCNT) | 113 |
| 8.2.8 | Refresh Time Constant Register (RTCOR) | 114 |
| 8.3 | Accessing Ordinary Space | 115 |
| 8.3.1 | Basic Timing | 115 |
| 8.3.2 | Wait State Control | 116 |
| 8.3.3 | $\overline{\text{CS}}$ Assert Period Extension | 118 |

| | | |
|--|---|-----|
| 8.4 | DRAM Access..... | 119 |
| 8.4.1 | DRAM Direct Connection | 119 |
| 8.4.2 | Basic Timing | 120 |
| 8.4.3 | Wait State Control..... | 121 |
| 8.4.4 | Burst Operation | 125 |
| 8.4.5 | Refresh Timing..... | 127 |
| 8.5 | Address/Data Multiplex I/O Space Access | 128 |
| 8.5.1 | Basic Timing | 128 |
| 8.5.2 | Wait State Control..... | 130 |
| 8.5.3 | CS Assertion Extension | 131 |
| 8.6 | Waits between Access Cycles | 132 |
| 8.6.1 | Prevention of Data Bus Conflicts..... | 132 |
| 8.6.2 | Simplification of Bus Cycle Start Detection | 133 |
| 8.7 | Bus Arbitration | 134 |
| 8.8 | Memory Connection Examples | 134 |
| 8.9 | On-chip Peripheral I/O Register Access | 136 |
| 8.10 | CPU Operation when Program Is in External Memory..... | 137 |
| Section 9 Direct Memory Access Controller (DMAC) | | 139 |
| 9.1 | Overview | 139 |
| 9.1.1 | Features | 139 |
| 9.1.2 | Block Diagram | 140 |
| 9.1.3 | Pin Configuration..... | 141 |
| 9.1.4 | Register Configuration..... | 142 |
| 9.2 | Register Descriptions..... | 143 |
| 9.2.1 | DMA Source Address Registers 0, 1 (SAR0, SAR1) | 143 |
| 9.2.2 | DMA Destination Address Registers 0, 1 (DAR0, DAR1) | 143 |
| 9.2.3 | DMA Transfer Count Registers 0, 1 (DMATCR0, DMATCR1) | 144 |
| 9.2.4 | DMA Channel Control Registers 0, 1 (CHCR0, CHCR1)..... | 145 |
| 9.2.5 | DMAC Operation Register (DMAOR) | 149 |
| 9.3 | Operation..... | 151 |
| 9.3.1 | DMA Transfer Flow..... | 151 |
| 9.3.2 | DMA Transfer Requests..... | 153 |
| 9.3.3 | Channel Priority | 155 |
| 9.3.4 | DMA Transfer Types | 155 |
| 9.3.5 | Address Modes..... | 156 |
| 9.3.6 | Dual Address Mode | 157 |
| 9.3.7 | Bus Modes..... | 160 |
| 9.3.8 | Relationship between Request Modes and Bus Modes by DMA Transfer Category..... | 161 |
| 9.3.9 | Bus Mode and Channel Priority Order..... | 162 |
| 9.3.10 | Number of Bus Cycle States and $\overline{\text{DREQ}}$ Pin Sample Timing..... | 162 |
| 9.3.11 | DMA Transfer Ending Conditions..... | 177 |

| | | |
|--|---|------------|
| 9.3.12 | DMAC Access from CPU | 178 |
| 9.4 | Examples of Use | 178 |
| 9.4.1 | Example of DMA Transfer between On-Chip SCI and External Memory | 178 |
| 9.4.2 | Example of DMA Transfer between External RAM and External Device with DACK | 179 |
| 9.5 | Cautions on Use | 180 |
| Section 10 Multifunction Timer Pulse Unit (MTU) | | 181 |
| 10.1 | Overview | 181 |
| 10.1.1 | Features | 181 |
| 10.1.2 | Block Diagram | 185 |
| 10.1.3 | Pin Configuration | 186 |
| 10.1.4 | Register Configuration | 187 |
| 10.2 | MTU Register Descriptions | 188 |
| 10.2.1 | Timer Control Register (TCR) | 188 |
| 10.2.2 | Timer Mode Register (TMDR) | 192 |
| 10.2.3 | Timer I/O Control Register (TIOR) | 194 |
| 10.2.4 | Timer Interrupt Enable Register (TIER) | 202 |
| 10.2.5 | Timer Status Register (TSR) | 204 |
| 10.2.6 | Timer Counters (TCNT) | 207 |
| 10.2.7 | Timer General Register (TGR) | 208 |
| 10.2.8 | Timer Start Register (TSTR) | 208 |
| 10.2.9 | Timer Synchro Register (TSYR) | 209 |
| 10.3 | Bus Master Interface | 210 |
| 10.3.1 | 16-Bit Registers | 210 |
| 10.3.2 | 8-Bit Registers | 211 |
| 10.4 | Operation | 212 |
| 10.4.1 | Overview | 212 |
| 10.4.2 | Basic Functions | 212 |
| 10.4.3 | Synchronous Operation | 218 |
| 10.4.4 | Buffer Operation | 221 |
| 10.4.5 | Cascade Connection Mode | 224 |
| 10.4.6 | PWM Mode | 225 |
| 10.4.7 | Phase Counting Mode | 229 |
| 10.5 | Interrupts | 236 |
| 10.5.1 | Interrupt Sources and Priority Ranking | 236 |
| 10.5.2 | DMAC Activation | 237 |
| 10.5.3 | A/D Converter Activation | 237 |
| 10.6 | Operation Timing | 238 |
| 10.6.1 | Input/Output Timing | 238 |
| 10.6.2 | Interrupt Signal Timing | 242 |
| 10.7 | Notes and Precautions | 246 |
| 10.7.1 | Input Clock Limitations | 246 |

| | | |
|--|---|------------|
| 10.7.2 | Note on Cycle Setting | 246 |
| 10.7.3 | Contention between TCNT Write and Clear | 247 |
| 10.7.4 | Contention between TCNT Write and Increment | 248 |
| 10.7.5 | Contention between Buffer Register Write and Compare Match | 249 |
| 10.7.6 | Contention between TGR Read and Input Capture | 250 |
| 10.7.7 | Contention between TGR Write and Input Capture | 251 |
| 10.7.8 | Contention between Buffer Register Write and Input Capture | 252 |
| 10.7.9 | Contention between TGR Write and Compare Match | 253 |
| 10.7.10 | TCNT2 Write and Overflow/Underflow Contention in Cascade Connection | 253 |
| 10.7.11 | Contention between Overflow/Underflow and Counter Clearing | 255 |
| 10.7.12 | Contention between TCNT Write and Overflow/Underflow | 256 |
| 10.7.13 | Cautions on Carrying Out Buffer Operation of Channel 0 in PWM Mode | 256 |
| 10.8 | MTU Output Pin Initialization | 257 |
| 10.8.1 | Operating Modes | 257 |
| 10.8.2 | Reset Start Operation | 257 |
| 10.8.3 | Operation in Case of Re-Setting Due to Error during Operation, Etc | 257 |
| 10.8.4 | Overview of Initialization Procedures and Mode Transitions in Case of Error during Operation, etc | 258 |
| Section 11 Watchdog Timer (WDT) | | 275 |
| 11.1 | Overview | 275 |
| 11.1.1 | Features | 275 |
| 11.1.2 | Block Diagram | 276 |
| 11.1.3 | Pin Configuration | 276 |
| 11.1.4 | Register Configuration | 277 |
| 11.2 | Register Descriptions | 277 |
| 11.2.1 | Timer Counter (TCNT) | 277 |
| 11.2.2 | Timer Control/Status Register (TCSR) | 278 |
| 11.2.3 | Reset Control/Status Register (RSTCSR) | 280 |
| 11.2.4 | Register Access | 281 |
| 11.3 | Operation | 282 |
| 11.3.1 | Watchdog Timer Mode | 282 |
| 11.3.2 | Interval Timer Mode | 284 |
| 11.3.3 | Clearing the Standby Mode | 284 |
| 11.3.4 | Timing of Setting the Overflow Flag (OVF) | 285 |
| 11.3.5 | Timing of Setting the Watchdog Timer Overflow Flag (WOVF) | 285 |
| 11.4 | Notes on Use | 286 |
| 11.4.1 | TCNT Write and Increment Contention | 286 |
| 11.4.2 | Changing CKS2 to CKS0 Bit Values | 286 |
| 11.4.3 | Changing between Watchdog Timer/Interval Timer Modes | 286 |
| 11.4.4 | System Reset with $\overline{\text{WDTOVF}}$ | 287 |
| 11.4.5 | Internal Reset with the Watchdog Timer | 287 |

| | | |
|------------|--|-----|
| Section 12 | Serial Communication Interface (SCI) | 289 |
| 12.1 | Overview | 289 |
| 12.1.1 | Features | 289 |
| 12.1.2 | Block Diagram | 290 |
| 12.1.3 | Pin Configuration | 291 |
| 12.1.4 | Register Configuration | 291 |
| 12.2 | Register Descriptions | 292 |
| 12.2.1 | Receive Shift Register (RSR) | 292 |
| 12.2.2 | Receive Data Register (RDR) | 292 |
| 12.2.3 | Transmit Shift Register (TSR) | 292 |
| 12.2.4 | Transmit Data Register (TDR) | 293 |
| 12.2.5 | Serial Mode Register (SMR) | 293 |
| 12.2.6 | Serial Control Register (SCR) | 296 |
| 12.2.7 | Serial Status Register (SSR) | 299 |
| 12.2.8 | Bit Rate Register (BRR) | 303 |
| 12.3 | Operation | 318 |
| 12.3.1 | Overview | 318 |
| 12.3.2 | Operation in Asynchronous Mode | 320 |
| 12.3.3 | Multiprocessor Communication | 330 |
| 12.3.4 | Clock Synchronous Operation | 338 |
| 12.4 | SCI Interrupt Sources and the DMAC | 349 |
| 12.5 | Notes on Use | 350 |
| 12.5.1 | TDR Write and TDRE Flags | 350 |
| 12.5.2 | Simultaneous Multiple Receive Errors | 350 |
| 12.5.3 | Break Detection and Processing | 351 |
| 12.5.4 | Sending a Break Signal | 351 |
| 12.5.5 | Receive Error Flags and Transmitter Operation (Clock Synchronous Mode Only) | 351 |
| 12.5.6 | Receive Data Sampling Timing and Receive Margin in the Asynchronous Mode | 351 |
| 12.5.7 | Constraints on DMAC Use | 353 |
| 12.5.8 | Cautions for Clock Synchronous External Clock Mode | 353 |
| 12.5.9 | Caution for Clock Synchronous Internal Clock Mode | 353 |
| Section 13 | High Speed A/D Converter –SH7014– | 355 |
| 13.1 | Overview | 355 |
| 13.1.1 | Features | 355 |
| 13.1.2 | Block Diagram | 356 |
| 13.1.3 | Pin Configuration | 356 |
| 13.1.4 | Register Configuration | 357 |
| 13.2 | Register Descriptions | 358 |
| 13.2.1 | A/D Data Registers A–H (ADDRA–ADDRH) | 358 |
| 13.2.2 | A/D Control/Status Register (ADCSR) | 359 |

| | | |
|--|---|-----|
| 13.2.3 | A/D Control Register (ADCR)..... | 362 |
| 13.3 | Bus Master Interface | 363 |
| 13.4 | Operation..... | 366 |
| 13.4.1 | Select-Single Mode | 366 |
| 13.4.2 | Select-Scan Mode..... | 367 |
| 13.4.3 | Group-Single Mode..... | 368 |
| 13.4.4 | Group-Scan Mode | 369 |
| 13.4.5 | Buffer Operation | 370 |
| 13.4.6 | Simultaneous Sampling Operation..... | 373 |
| 13.4.7 | Conversion Start Modes..... | 375 |
| 13.4.8 | A/D Conversion Time | 378 |
| 13.5 | Interrupts | 379 |
| 13.6 | Notes on Use | 380 |
| 13.6.1 | Analog Input Voltage Range..... | 380 |
| 13.6.2 | AV_{CC} , AV_{SS} Input Voltages | 380 |
| 13.6.3 | Input Ports | 380 |
| 13.6.4 | Conversion Start Modes..... | 380 |
| 13.6.5 | A/D Conversion Termination..... | 380 |
| 13.6.6 | Handling of Analog Input Pins | 381 |
| Section 14 Mid-Speed A/D Converter –SH7016, SH7017–..... | | 383 |
| 14.1 | Overview | 383 |
| 14.1.1 | Features | 383 |
| 14.1.2 | Block Diagram | 384 |
| 14.1.3 | Pin Configuration..... | 385 |
| 14.1.4 | Register Configuration..... | 386 |
| 14.2 | Register Descriptions..... | 387 |
| 14.2.1 | A/D Data Register A to D (ADDRA to ADDR D)..... | 387 |
| 14.2.2 | A/D Control/Status Register (ADCSR) | 388 |
| 14.2.3 | A/D Control Register (ADCR)..... | 391 |
| 14.3 | Interface with CPU | 391 |
| 14.4 | Operation..... | 393 |
| 14.4.1 | Single Mode (SCAN=0)..... | 393 |
| 14.4.2 | Scan Mode (SCAN=1)..... | 395 |
| 14.4.3 | Input Sampling and A/D Conversion Time | 397 |
| 14.4.4 | MTU Trigger Input Timing..... | 399 |
| 14.5 | Interrupt..... | 400 |
| 14.6 | A/D Conversion Precision Definitions..... | 401 |
| 14.7 | Usage Notes..... | 402 |
| 14.7.1 | Analog Voltage Settings..... | 402 |
| 14.7.2 | Handling of Analog Input Pins..... | 402 |

| | | |
|------------|--|-----|
| Section 15 | Compare Match Timer (CMT) | 405 |
| 15.1 | Overview | 405 |
| 15.1.1 | Features | 405 |
| 15.1.2 | Block Diagram | 405 |
| 15.1.3 | Register Configuration | 407 |
| 15.2 | Register Descriptions | 408 |
| 15.2.1 | Compare Match Timer Start Register (CMSTR) | 408 |
| 15.2.2 | Compare Match Timer Control/Status Register (CMCSR) | 409 |
| 15.2.3 | Compare Match Timer Counter (CMCNT) | 410 |
| 15.2.4 | Compare Match Timer Constant Register (CMCOR) | 411 |
| 15.3 | Operation | 411 |
| 15.3.1 | Period Count Operation | 411 |
| 15.3.2 | CMCNT Count Timing | 412 |
| 15.4 | Interrupts | 412 |
| 15.4.1 | Interrupt Sources and DTC Activation | 412 |
| 15.4.2 | Compare Match Flag Set Timing | 412 |
| 15.4.3 | Compare Match Flag Clear Timing | 413 |
| 15.5 | Notes on Use | 414 |
| 15.5.1 | Contention between CMCNT Write and Compare Match | 414 |
| 15.5.2 | Contention between CMCNT Word Write and Incrementation | 415 |
| 15.5.3 | Contention between CMCNT Byte Write and Incrementation | 416 |
| Section 16 | Pin Function Controller | 417 |
| 16.1 | Overview | 417 |
| 16.2 | Register Configuration | 428 |
| 16.3 | Register Descriptions | 429 |
| 16.3.1 | Port A I/O Register L (PAIORL) | 429 |
| 16.3.2 | Port A Control Registers L1, L2 (PACRL1 and PACRL2) | 430 |
| 16.3.3 | Port B I/O Register (PBIOR) | 435 |
| 16.3.4 | Port B Control Registers (PBCR1 and PBCR2) | 436 |
| 16.3.5 | Port C I/O Register (PCIOR) –SH7016, SH7017– | 441 |
| 16.3.6 | Port C Control Register (PCCR) –SH7016, SH7017– | 442 |
| 16.3.7 | Port D I/O Register L (PDIORL) –SH7016, SH7017– | 445 |
| 16.3.8 | Port D Control Register L (PDCRL) –SH7016, SH7017– | 446 |
| 16.3.9 | Port E I/O Register (PEIOR) | 450 |
| 16.3.10 | Port E Control Registers 1, 2 (PECR1 and PECR2) | 451 |
| Section 17 | I/O Ports (I/O) | 455 |
| 17.1 | Overview | 455 |
| 17.2 | Port A | 455 |
| 17.2.1 | Register Configuration | 456 |
| 17.2.2 | Port A Data Register L (PADRL) | 457 |
| 17.3 | Port B | 458 |

| | | |
|-------------------|---|------------|
| 17.3.1 | Register Configuration..... | 458 |
| 17.3.2 | Port B Data Register (PBDR)..... | 459 |
| 17.4 | Port C –SH7016, SH7017– | 460 |
| 17.4.1 | Register Configuration..... | 460 |
| 17.4.2 | Port C Data Register (PCDR)..... | 461 |
| 17.5 | Port D –SH7016, SH7017– | 462 |
| 17.5.1 | Register Configuration..... | 462 |
| 17.5.2 | Port D Data Register L (PDDRL) | 463 |
| 17.6 | Port E..... | 464 |
| 17.6.1 | Register Configuration..... | 464 |
| 17.6.2 | Port E Data Register (PEDR)..... | 465 |
| 17.7 | Port F..... | 466 |
| 17.7.1 | Register Configuration..... | 466 |
| 17.7.2 | Port F Data Register (PFDR) | 466 |
| Section 18 | 128 kB Flash Memory (F-ZTAT) | 469 |
| 18.1 | Features | 469 |
| 18.2 | Overview | 470 |
| 18.2.1 | Block Diagram | 470 |
| 18.2.2 | Mode Transitions | 471 |
| 18.2.3 | On-Board Programming Modes..... | 472 |
| 18.2.4 | Flash Memory Emulation in RAM..... | 474 |
| 18.2.5 | Differences between Boot Mode and User Program Mode | 475 |
| 18.2.6 | Block Configuration..... | 476 |
| 18.3 | Pin Configuration | 476 |
| 18.4 | Register Configuration | 477 |
| 18.5 | Register Descriptions..... | 478 |
| 18.5.1 | Flash Memory Control Register 1 (FLMCR1)..... | 478 |
| 18.5.2 | Flash Memory Control Register 2 (FLMCR2)..... | 481 |
| 18.5.3 | Erase Block Register 1 (EBR1) | 482 |
| 18.5.4 | RAM Emulation Register (RAMER)..... | 483 |
| 18.6 | On-Board Programming Modes | 484 |
| 18.6.1 | Boot Mode..... | 485 |
| 18.6.2 | User Program Mode | 489 |
| 18.7 | Programming/Erasing Flash Memory | 490 |
| 18.7.1 | Program Mode..... | 490 |
| 18.7.2 | Program-Verify Mode | 491 |
| 18.7.3 | Erase Mode..... | 497 |
| 18.7.4 | Erase-Verify Mode..... | 498 |
| 18.8 | Protection..... | 504 |
| 18.8.1 | Hardware Protection..... | 504 |
| 18.8.2 | Software Protection | 505 |
| 18.8.3 | Error Protection | 506 |

| | | |
|---|--|---------|
| 18.9 | Flash Memory Emulation in RAM..... | 508 |
| 18.10 | Note on Flash Memory Programming/Erasing | 510 |
| 18.11 | Flash Memory Programmer Mode | 510 |
| 18.11.1 | Socket Adapter Pin Correspondence Diagram..... | 511 |
| 18.11.2 | Programmer Mode Operation..... | 513 |
| 18.11.3 | Memory Read Mode..... | 514 |
| 18.11.4 | Auto-Program Mode | 517 |
| 18.11.5 | Auto-Erase Mode | 519 |
| 18.11.6 | Status Read Mode..... | 521 |
| 18.11.7 | Status Polling | 522 |
| 18.11.8 | Programmer Mode Transition Time..... | 522 |
| 18.11.9 | Notes on Memory Programming..... | 523 |
| Section 19 Mask ROM..... | | 525 |
| 19.1 | Overview | 525 |
| Section 20 RAM..... | | 527 |
| 20.1 | Overview | 527 |
| Section 21 Power-Down State | | 529 |
| 21.1 | Overview | 529 |
| 21.1.1 | Power-Down States | 529 |
| 21.1.2 | Related Register | 530 |
| 21.2 | Standby Control Register (SBYCR) | 530 |
| 21.3 | Sleep Mode..... | 531 |
| 21.3.1 | Transition to Sleep Mode..... | 531 |
| 21.3.2 | Canceling Sleep Mode | 531 |
| 21.4 | Standby Mode | 531 |
| 21.4.1 | Transition to Standby Mode..... | 531 |
| 21.4.2 | Canceling the Standby Mode | 533 |
| 21.4.3 | Standby Mode Application Example | 534 |
| Section 22 Electrical Characteristics (5 V 28.7 MHz)..... | | 535 |
| 22.1 | Absolute Maximum Ratings..... | 535 |
| 22.2 | DC Characteristics..... | 536 |
| 22.3 | AC Characteristics..... | 538 |
| 22.3.1 | Clock Timing | 538 |
| 22.3.2 | Control Signal Timing | 540 |
| 22.3.3 | Bus Timing..... | 542 |
| 22.3.4 | Direct Memory Access Controller Timing..... | 555 |
| 22.3.5 | Multifunction Timer Pulse Unit Timing | 557 |
| 22.3.6 | I/O Port Timing | 558 |
| 22.3.7 | Watchdog Timer Timing | 558 |

| | | |
|--|---|-----|
| 22.3.8 | Serial Communication Interface Timing..... | 559 |
| 22.3.9 | High Speed A/D Converter Timing – SH7014– | 560 |
| 22.3.10 | Mid-speed Converter Timing –SH7016, SH7017–..... | 561 |
| 22.3.11 | Measuring Conditions for AC Characteristics | 562 |
| 22.4 | A/D Converter Characteristics | 563 |
| Appendix A On-Chip Supporting Module Registers | | 565 |
| A.1 | Addresses..... | 565 |
| Appendix B I/O Port Block Diagrams..... | | 574 |
| Appendix C Pin States..... | | 599 |
| Appendix D Notes when Converting the F-ZTAT Application Software to the Mask-ROM Versions | | 605 |
| Appendix E Product Code Lineup..... | | 606 |
| Appendix F Package Dimensions..... | | 607 |

Section 1 SH7014/16/17 Overview

1.1 SH7014/16/17 Overview

The SH7014/16/17 CMOS single-chip microprocessors integrate a Hitachi-original architecture, high-speed CPU with peripheral functions required for system configuration.

The CPU has a RISC-type instruction set. Most instructions can be executed in one clock cycle, which greatly improves instruction execution speed. In addition, the 32-bit internal-bus architecture enhances data processing power. With this CPU, it has become possible to assemble low cost, high performance/high-functioning systems, even for applications that were previously impossible with microprocessors, such as real-time control, which demands high speeds. In particular, the SH7040 series has a 1-kbyte on-chip cache, which allows an improvement in CPU performance during external memory access.

In addition, this LSI includes on-chip peripheral functions necessary for system configuration, such as large-capacity ROM (except the SH7014, which is ROMless) and RAM, timers, a serial communication interface (SCI), an A/D converter, an interrupt controller, and I/O ports. Memory or peripheral LSIs can be connected efficiently with an external memory access support function. This greatly reduces system cost.

This LSI has an F-ZTAT™* version with on-chip flash memory and a mask ROM version. These versions enable users to respond quickly and flexibly to changing application specifications, growing production volumes, and other conditions.

Notes: F-ZTAT (Flexible ZTAT) is a trademark of Hitachi, Ltd.

1.1.1 SH7014/16/17 Series Features

CPU:

- Original Hitachi architecture
- 32-bit internal data bus
- General-register machine
 - Sixteen 32-bit general registers
 - Three 32-bit control registers
 - Four 32-bit system registers
- RISC-type instruction set
 - Instruction length: 16-bit fixed length for improved code efficiency
 - Load-store architecture (basic operations are executed between registers)
 - Delayed branch instructions reduce pipeline disruption during branch
 - Instruction set based on C language

- Instruction execution time: one instruction/cycle (35 ns/instruction at 28.7-MHz operation)
- Address space: Architecture supports 4 Gbytes
- On-chip multiplier: multiplication operations (32 bits × 32 bits → 64 bits) and multiplication/accumulation operations (32 bits × 32 bits + 64 bits → 64 bits) executed in two to four cycles
- Five-stage pipeline

Cache Memory:

- 1-kbyte instruction cache
- Caching of instruction codes and PC relative read data
- 4-byte line length (1 longword: 2 instruction lengths)
- 256 entry cache tags
- Direct map method
- On-chip ROM/RAM, and on-chip I/O areas not objects of cache
- Used in common with on-chip RAM; 2 kbytes of on-chip RAM used as address array/data array when cache is enabled

Interrupt Controller (INTC):

- Seven external interrupt pins (NMI, $\overline{\text{IRQ6}}$)
- Twenty-eight internal interrupt sources
- Sixteen programmable priority levels

Bus State Controller (BSC):

- Supports external extended memory access
 - 8-bit, or 16-bit external data bus
- Memory address space divided into five areas (four areas of SRAM space, one area of DRAM space) with the following settable features:
 - Number of wait cycles
 - Outputs chip-select signals for each area
 - During DRAM space access:
 - Outputs $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ signals for DRAM
 - Can generate a RAS precharge time assurance T_p cycle
- DRAM burst access function
 - Supports high-speed access mode for DRAM
- DRAM refresh function
 - Programmable refresh interval
 - Supports CAS-before-RAS refresh and self-refresh modes
- Wait cycles can be inserted using an external WAIT signal

- Address data multiplex I/O devices can be accessed

Note: No bus release

Direct Memory Access Controller (DMAC) (2 Channels):

- Supports cycle-steal and burst transfers
- Supports single address mode and dual address mode transfers
- Priority order: fixed at channel 0 > channel 1
- Transfer counter: 16 bits
- Transfer request sources: external DREQ input, auto-request, and on-chip supporting modules
- Address space: 4 Gbytes
- Choice of 8-, 16-, or 32-bit transfer data size

Multifunction Timer/Pulse Unit (MTU) (3 Channels):

- Maximum 8 types of waveform output or maximum 16 types of pulse I/O processing possible based on 16-bit timer, 3 channels
- 8 dual-use output compare/input capture registers
- 8 independent comparators
- 8 types of counter input clock
- Input capture function
- Pulse output mode
 - One shot, toggle, PWM
- Phase calculation mode
 - 2-phase encoder calculation processing

Compare Match Timer (CMT) (Two Channels):

- 16-bit free-running counter
- One compare register
- Generates an interrupt request upon compare match

Watchdog Timer (WDT) (One Channel):

- Watchdog timer or interval timer
- Count overflow can generate an internal reset, external signal, or interrupt

Serial Communication Interface (SCI) (Two Channels):

(Per Channel):

- Asynchronous or clock-synchronous mode is selectable
- Can transmit and receive simultaneously (full duplex)
- On-chip dedicated baud rate generator
- Multiprocessor communication function

I/O Ports:

- SH7014
 - Input/output: 35
 - Input: 8
 - Total: 43
- SH7016/17
 - Input/output: 74
 - Input: 8
 - Total: 82

A/D Converter:

- 10 bits \times 8 channels
- The SH7014 has a high-speed A/D converter, while the SH7016 and SH7017 have a mid-speed A/D converter.

On-Chip Memory:

- ROM
 - SH7014: ROMless
 - SH7016: 64 kbytes (mask ROM)
 - SH7017: 128 kbytes (flash ROM)
- RAM: SH7014/16: 3 kbytes (1 kbyte when cache is used)
SH7017: 4 kbytes (2 kbytes when cache is used)

Operating Modes:

- Operating modes
 - Non-extended ROM mode (SH7014/16/17)
 - Extended ROM mode (SH7016/17)
 - Single-chip mode (SH7016/17)
- Processing states

- Program execution state
- Exception processing state
- Power-down modes
 - Sleep mode
 - Software standby mode

Clock Pulse Generator (CPG):

- On-chip clock pulse generator
 - On-chip clock-doubling PLL circuit

Product Lineup:

| Product Code | On-Chip ROM | On-Chip RAM | A/D Precision | Frequency/ Voltage | Temperature | Package |
|---------------|---------------------------|-------------|------------------------------|-----------------------|--------------|---------|
| HD6417014F28 | ROMless | 3 kB | ± 15 LSB (high-speed A/D) | 28.7 MHz/5 V | -20 to +75°C | FP-112 |
| HD6417014RF28 | ROMless | 3 kB | ± 8 LSB (high-speed A/D) | 28.7 MHz/5 V | -20 to +75°C | FP-112 |
| HD6437016F28 | 64 kB mask ROM | 3 kB | ± 4 LSB (mid-speed A/D) | 28.7 MHz/5 V | -20 to +75°C | FP-112 |
| HD64F7017F28 | 128 kB flash memory | 4 kB | ± 4LSB (mid-speed A/D) | 28.7 MHz/5 V | -20 to +75°C | FP-112 |

1.2 Block Diagram

Figure 1.1 is a block diagram of the SH7014. Figure 1.2 is a block diagram of the SH7016/17.

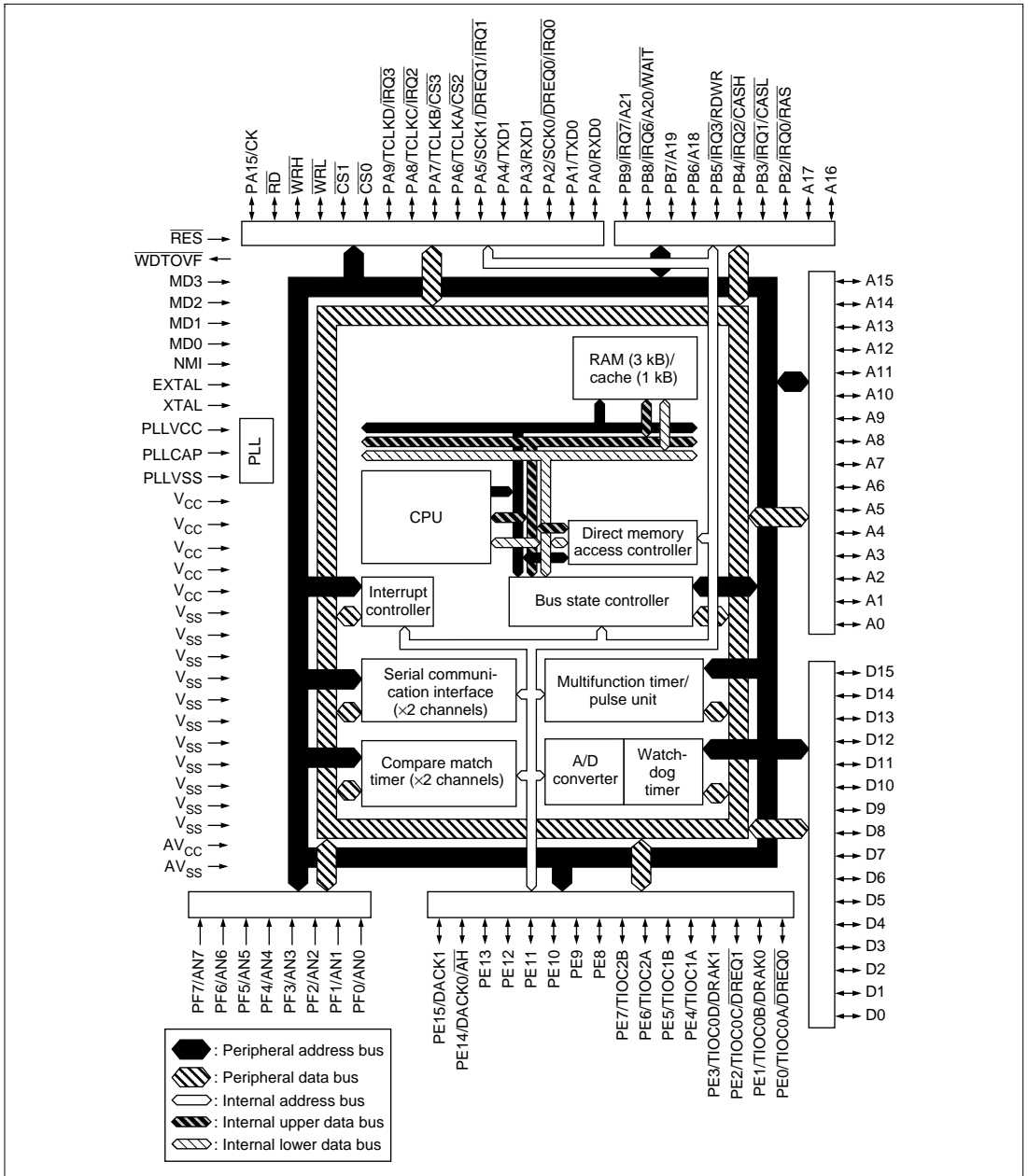
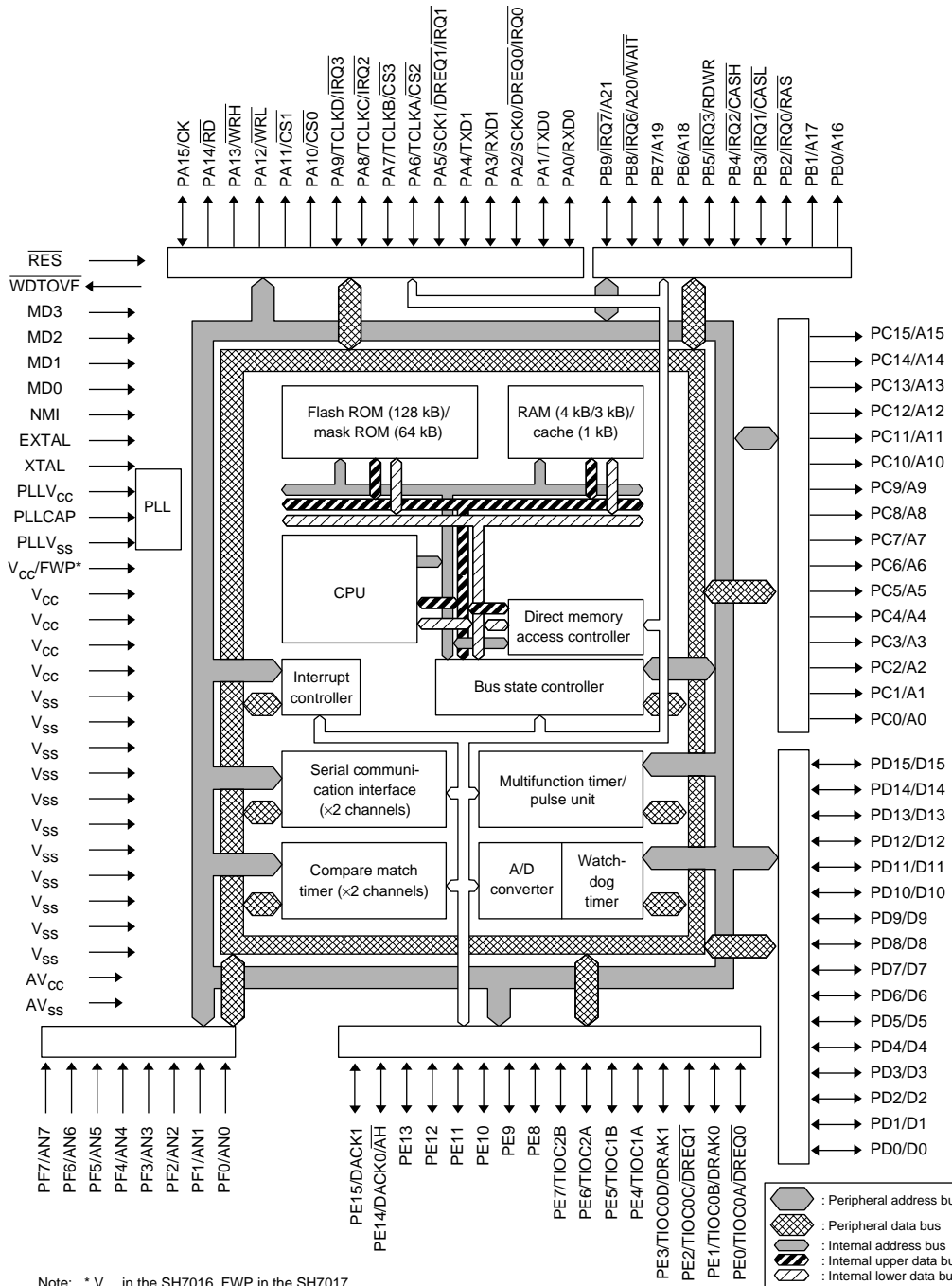


Figure 1.1 Block Diagram of the SH7014



Note: * V_{CC} in the SH7016, FWP in the SH7017.

Figure 1.2 Block Diagram of the SH7016, SH7017

1.3 Pin Arrangement and Pin Functions

1.3.1 Pin Arrangement

Figure 1.3 shows the pin arrangement for the SH7014 (top view).

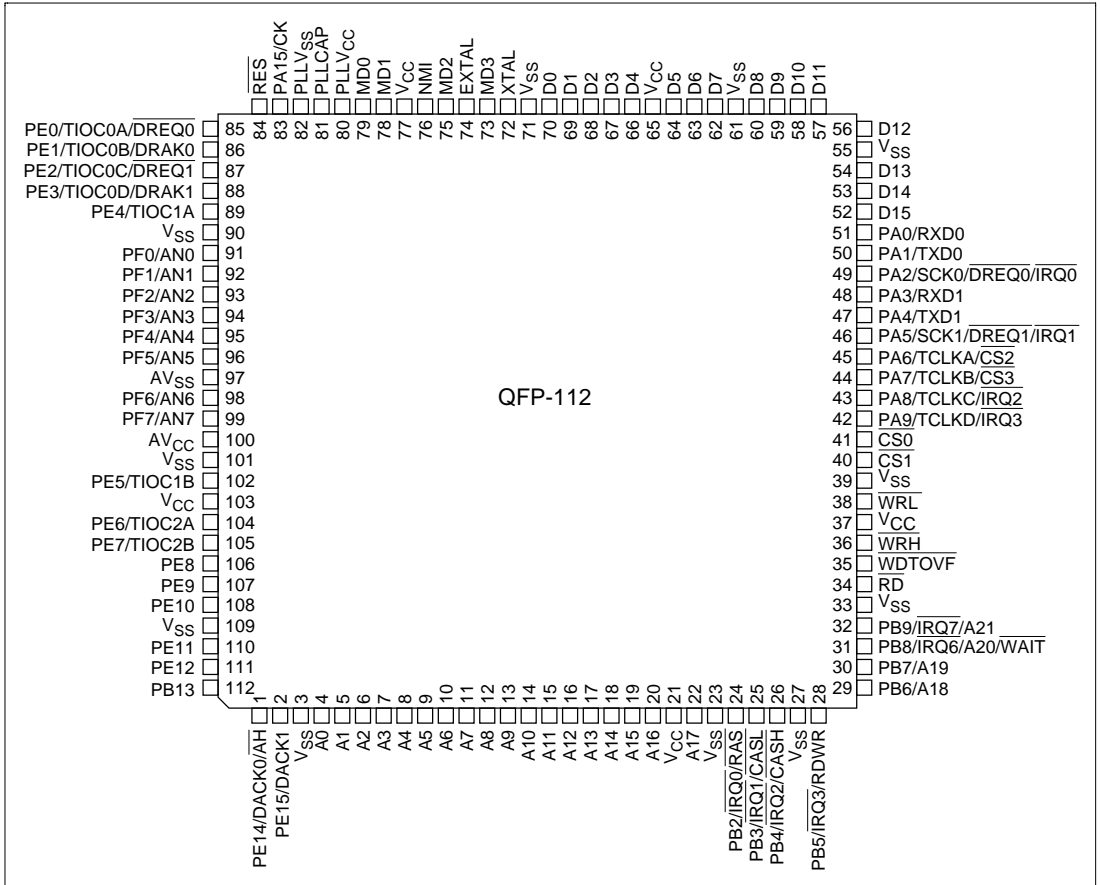


Figure 1.3 SH7014 Pin Arrangement (QFP-112 Top View)

Figure 1.4 shows the pin arrangement for the 144-pin QFP pin arrangement.

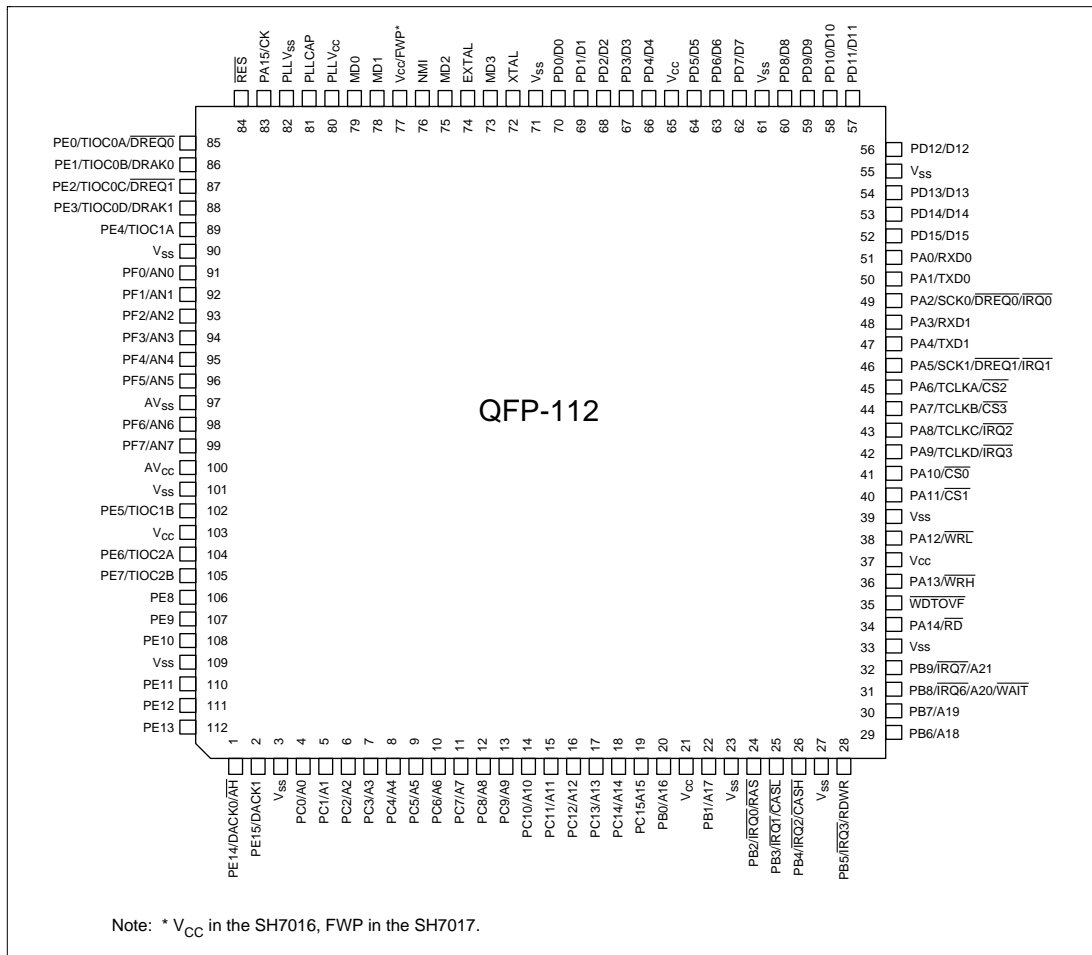


Figure 1.4 SH7016, SH7017 Pin Arrangement (QFP-112 Top View)

1.3.2 Pin Arrangement by Mode

Table 1.1 Pin Arrangement by Mode for SH7017F (QFP-112 Pin)

| Pin No. | MCU Mode | Programmer Mode |
|---------|-----------------------------|-----------------|
| 1 | PE14/DACK0/ \overline{AH} | NC |
| 2 | PE15/DACK1 | NC |
| 3 | V_{SS} | V_{SS} |
| 4 | PC0/A0 | A0 |
| 5 | PC1/A1 | A1 |
| 6 | PC2/A2 | A2 |
| 7 | PC3/A3 | A3 |
| 8 | PC4/A4 | A4 |
| 9 | PC5/A5 | A5 |
| 10 | PC6/A6 | A6 |
| 11 | PC7/A7 | A7 |
| 12 | PC8/A8 | A8 |
| 13 | PC9/A9 | A9 |
| 14 | PC10/A10 | A10 |
| 15 | PC11/A11 | A11 |
| 16 | PC12/A12 | A12 |
| 17 | PC13/A13 | A13 |
| 18 | PC14/A14 | A14 |
| 19 | PC15/A15 | A15 |
| 20 | PB0/A16 | A16 |
| 21 | V_{CC} | V_{CC} |
| 22 | PB1/A17 | NC |
| 23 | V_{SS} | V_{SS} |
| 24 | PB2/ $\overline{IRQ0/RAS}$ | NC |
| 25 | PB3/ $\overline{IRQ1/CASL}$ | NC |
| 26 | PB4/ $\overline{IRQ2/CASH}$ | A17 |
| 27 | V_{SS} | V_{SS} |
| 28 | PB5/ $\overline{IRQ3/RDWR}$ | NC |

Table 1.1 Pin Arrangement by Mode for SH7017F (QFP-112 Pin) (cont)

| Pin No. | MCU Mode | Programmer Mode |
|----------------|--|------------------------|
| 29 | PB6/A18 | NC |
| 30 | PB7/A19 | NC |
| 31 | PB8/ $\overline{\text{IRQ6}}$ /A20/ $\overline{\text{WAIT}}$ | NC |
| 32 | PB9/ $\overline{\text{IRQ7}}$ /A21 | NC |
| 33 | V_{SS} | V_{SS} |
| 34 | PA14/ $\overline{\text{RD}}$ | NC |
| 35 | $\overline{\text{WDTOVF}}$ | NC |
| 36 | PA13/ $\overline{\text{WRH}}$ | NC |
| 37 | V_{CC} | V_{CC} |
| 38 | PA12/ $\overline{\text{WRL}}$ | NC |
| 39 | V_{SS} | V_{SS} |
| 40 | PA11/ $\overline{\text{CS1}}$ | NC |
| 41 | PA10/ $\overline{\text{CS0}}$ | NC |
| 42 | PA9/ $\overline{\text{TCLKD}}$ / $\overline{\text{IRQ3}}$ | $\overline{\text{CE}}$ |
| 43 | PA8/ $\overline{\text{TCLKC}}$ / $\overline{\text{IRQ2}}$ | $\overline{\text{OE}}$ |
| 44 | PA7/ $\overline{\text{TCLKB}}$ / $\overline{\text{CS3}}$ | $\overline{\text{WE}}$ |
| 45 | PA6/ $\overline{\text{TCLKA}}$ / $\overline{\text{CS2}}$ | NC |
| 46 | PA5/ $\overline{\text{SCK1}}$ / $\overline{\text{DREQ1}}$ / $\overline{\text{IRQ1}}$ | V_{CC} |
| 47 | PA4/ $\overline{\text{TXD1}}$ | NC |
| 48 | PA3/ $\overline{\text{RXD1}}$ | NC |
| 49 | PA2/ $\overline{\text{SCK0}}$ / $\overline{\text{DREQ0}}$ / $\overline{\text{IRQ0}}$ | V_{CC} |
| 50 | PA1/ $\overline{\text{TXD0}}$ | V_{CC} |
| 51 | PA0/ $\overline{\text{RXD0}}$ | NC |
| 52 | PD15/D15 | NC |
| 53 | PD14/D14 | NC |
| 54 | PD13/D13 | NC |
| 55 | V_{SS} | V_{SS} |
| 56 | PD12/D12 | NC |
| 57 | PD11/D11 | NC |
| 58 | PD10/D10 | NC |

Table 1.1 Pin Arrangement by Mode for SH7017F (QFP-112 Pin) (cont)

| Pin No. | MCU Mode | Programmer Mode |
|----------------|---------------------------------------|-------------------------------|
| 59 | PD9/D9 | NC |
| 60 | PD8/D8 | NC |
| 61 | V _{SS} | V _{SS} |
| 62 | PD7/D7 | D7 |
| 63 | PD6/D6 | D6 |
| 64 | PD5/D5 | D5 |
| 65 | V _{CC} | V _{CC} |
| 66 | PD4/D4 | D4 |
| 67 | PD3/D3 | D3 |
| 68 | PD2/D2 | D2 |
| 69 | PD1/D1 | D1 |
| 70 | PD0/D0 | D0 |
| 71 | V _{SS} | V _{SS} |
| 72 | XTAL | XTAL |
| 73 | MD3 | MD3 |
| 74 | EXTAL | EXTAL |
| 75 | MD2 | MD2 |
| 76 | NMI | V _{CC} |
| 77 | V _{CC} (FWP) | FWE |
| 78 | MD1 | MD1 |
| 79 | MD0 | MD0 |
| 80 | PLL _{V_{CC}} | PLL _{V_{CC}} |
| 81 | PLLCAP | PLLCAP |
| 82 | PLL _{V_{SS}} | PLL _{V_{SS}} |
| 83 | PA15/CK | NC |
| 84 | $\overline{\text{RES}}$ | RES |
| 85 | PE0/TIOC0A/ $\overline{\text{DREQ0}}$ | NC |
| 86 | PE1/TIOC0B/DRAK0 | NC |
| 87 | PE2/TIOC0C/ $\overline{\text{DREQ1}}$ | NC |
| 88 | PE3/TIOC0D/DRAK1 | NC |

Table 1.1 Pin Arrangement by Mode for SH7017F (QFP-112 Pin) (cont)

| Pin No. | MCU Mode | Programmer Mode |
|----------------|------------------|------------------------|
| 89 | PE4/TIOC1A | NC |
| 90 | V _{SS} | V _{SS} |
| 91 | PF0/AN0 | V _{SS} |
| 92 | PF1/AN1 | V _{SS} |
| 93 | PF2/AN2 | V _{SS} |
| 94 | PF3/AN3 | V _{SS} |
| 95 | PF4/AN4 | V _{SS} |
| 96 | PF5/AN5 | V _{SS} |
| 97 | AV _{SS} | V _{SS} |
| 98 | PF6/AN6 | V _{SS} |
| 99 | PF7/AN7 | V _{SS} |
| 100 | AV _{CC} | V _{CC} |
| 101 | V _{SS} | V _{SS} |
| 102 | PE5/TIOC1B | NC |
| 103 | V _{CC} | V _{CC} |
| 104 | PE6/TIOC2A | NC |
| 105 | PE7/TIOC2B | NC |
| 106 | PE8 | NC |
| 107 | PE9 | NC |
| 108 | PE10 | NC |
| 109 | V _{SS} | V _{SS} |
| 110 | PE11 | NC |
| 111 | PE12 | NC |
| 112 | PE13 | NC |

1.3.3 Pin Functions

Table 1.2 lists the pin functions.

Table 1.2 Pin Functions

| Classification | Symbol | I/O | Name | Function |
|------------------------|--|-----|-----------------------------|--|
| Power supply | V_{CC} | I | Supply | Connects to power supply. Connect all V_{CC} pins to the system supply. No operation will occur if there are any open pins. |
| | V_{SS} | I | Ground | Connects to ground. Connect all V_{SS} pins to the system ground. No operation will occur if there are any open pins. |
| Clock | $PLL_{V_{CC}}$ | I | PLL supply | On-chip PLL oscillator supply. |
| | $PLL_{V_{SS}}$ | I | PLL ground | On-chip PLL oscillator ground. |
| | PLLCAP | I | PLL capacitance | On-chip PLL oscillator external capacitance connection pin. |
| | EXTAL | I | External clock | Connect a crystal oscillator. Also, an external clock can be input to the EXTAL pin. |
| | XTAL | I | Crystal | Connect a crystal oscillator. |
| | CK | O | System clock | Supplies the system clock to peripheral devices. |
| System control | \overline{RES} | I | Power-on reset | Power-on reset when low |
| | \overline{WDTOVF} | O | Watchdog timer overflow | Overflow output signal from WDT |
| Operating mode control | MD0–MD3 | I | Mode set | Determines the operating mode. Do not change input value during operation. |
| | FWP | I | Flash memory write protect | Protects flash memory from being written or deleted. |
| Interrupts | NMI | I | Non-maskable interrupt | Non-maskable interrupt request pin. Enables selection of whether to accept on the rising or falling edge. |
| | $\overline{IRQ0}$ – $\overline{IRQ3}$, $\overline{IRQ6}$, $\overline{IRQ7}$ | I | Interrupt requests 0–3, 6,7 | Maskable interrupt request pins. Allows selection of level input and edge input. |
| Address bus | A0–A21 | O | Address bus | Outputs addresses. |

Table 1.2 Pin Functions (cont)

| Classification | Symbol | I/O | Name | Function |
|--|--------------------------------------|------------|--|---|
| Data bus | D0–D15 | I/O | Data bus | 16-bit (QFP-112 pin version) or 32-bit (QFP-144 pin version) bidirectional data bus. |
| Bus control | $\overline{CS0}$ to $\overline{CS3}$ | O | Chip selects 0–3 | Chip select signals for external memory or devices. |
| | \overline{RD} | O | Read | Indicates reading from an external device. |
| | \overline{WRH} | O | Upper write | Indicates writing the upper 8 bits (15–8) of external data. |
| | \overline{WRL} | O | Lower write | Indicates writing the lower 8 bits (7–0) of external data. |
| | \overline{WAIT} | I | Wait | Input causes insertion of wait cycles into the bus cycle during external space access. |
| | \overline{RAS} | O | Row address strobe | Timing signal for DRAM row address strobe. |
| | \overline{CASH} | O | Upper column address strobe | Timing signal for DRAM column address strobe. Output when the upper 8 bits of data are accessed. |
| | \overline{CASL} | O | Lower column address strobe | Timing signal for DRAM column address strobe. Output when the lower 8 bits of data are accessed. |
| | RDWR | O | DRAM read/write | DRAM write strobe signal. |
| | \overline{AH} | O | Address hold | Address hold timing signal for devices using an address/data multiplex bus. |
| Bus control multifunction timer/pulse unit | TCLKA | I | MTU timer clock input | Input pins for external clocks to the MTU counter. |
| | TCLKB | | | |
| | TCLKC | | | |
| | TCLKD | | | |
| | TIOC0A | I/O | MTU input capture/output compare (channel 0) | Channel 0 input capture input/output compare output/PWM output pins. |
| | TIOC0B | | | |
| | TIOC0C | | | |
| TIOC0D | | | | |

Table 1.2 Pin Functions (cont)

| Classification | Symbol | I/O | Name | Function |
|---|--|--|--|---|
| Bus control multifunction timer/pulse unit (cont) | TIOC1A | I/O | MTU input capture/output compare (channel 1) | Channel 1 input capture input/output compare output/PWM output pins. |
| | TIOC1B | | | |
| | TIOC2A | I/O | MTU input capture/output compare (channel 2) | Channel 2 input capture input/output compare output/PWM output pins. |
| | TIOC2B | | | |
| Direct memory access controller (DMAC) | $\overline{\text{DREQ0}}$ to $\overline{\text{DREQ1}}$ | I | DMA transfer request (channels 0, 1) | Input pin for external requests for DMA transfer. |
| | DRAK0 to DRAK1 | O | DREQ request acknowledgment (channels 0, 1) | Output the input sampling acknowledgment of external DMA transfer requests. |
| | DACK0 to DACK1 | O | DMA transfer strobe (channels 0, 1) | Output a strobe to the external I/O of external DMA transfer requests. |
| Serial communication interface (SCI) | TxD0 to TxD1 | O | Transmit data (channels 0, 1) | SCI0, SCI1 transmit data output pins. |
| | RxD0 to RxD1 | I | Receive data (channels 0, 1) | SCI0, SCI1 receive data input pins. |
| | SCK0 to SCK1 | I/O | Serial clock (channels 0, 1) | SCI0, SCI1 clock input/output pins. |
| A/D Converter | AV_{CC} | I | Analog supply | Analog supply; connected to V_{CC} . |
| | AV_{SS} | I | Analog ground | Analog supply; connected to V_{SS} . |
| | AN0 to AN7 | I | Analog input | Analog signal input pins. |
| I/O ports | PA0 to PA9, PA15 (SH7014) | I/O | General purpose port | General purpose input/output port pins. |
| | PA0 to PA15 (SH7016/17) | | | Each bit can be designated for input/output. |
| | PB2 to PB9 (SH7014) | I/O | General purpose port | General purpose input/output port pins. |
| PB0 to PB9 (SH7016/17) | | Each bit can be designated for input/output. | | |

Table 1.2 Pin Functions (cont)

| Classification | Symbol | I/O | Name | Function |
|-----------------------|----------------------------|------------|----------------------|---|
| I/O ports | PC0 to PC15 (SH7016/17) | I/O | General purpose port | General purpose input/output port pins. Each bit can be designated for input/output. |
| | PD0 to PD15 (SH7016/17) | I/O | General purpose port | General purpose input/output port pins. Each bit can be designated for input/output. |
| | PE0 to PE15 | I/O | General purpose port | General purpose input/output port pins. Each bit can be designated for input/output. |
| | PF0 to PF7 | I | General purpose port | General purpose input port pins. |

Usage Notes

1. Unused input pins should be pulled up or pulled down.
2. The $\overline{\text{WDT0VF}}$ pin should not be pulled down in the SH7017 F-ZTAT version. However, if it is necessary to pull this pin down, a resistance of 100 k Ω or higher should be used.

Section 2 CPU

2.1 Register Configuration

The register set consists of sixteen 32-bit general registers, three 32-bit control registers and four 32-bit system registers.

2.1.1 General Registers (Rn)

The sixteen 32-bit general registers (Rn) are numbered R0–R15. General registers are used for data processing and address calculation. R0 is also used as an index register. Several instructions have R0 fixed as their only usable register. R15 is used as the hardware stack pointer (SP). Saving and recovering the status register (SR) and program counter (PC) in exception processing is accomplished by referencing the stack using R15. Figure 2.1 shows the general registers.

| | |
|------------------------------------|---|
| 31 | 0 |
| R0*1 | |
| R1 | |
| R2 | |
| R3 | |
| R4 | |
| R5 | |
| R6 | |
| R7 | |
| R8 | |
| R9 | |
| R10 | |
| R11 | |
| R12 | |
| R13 | |
| R14 | |
| R15, SP (hardware stack pointer)*2 | |

- Notes:
1. R0 functions as an index register in the indirect indexed register addressing mode and indirect indexed GBR addressing mode. In some instructions, R0 functions as a fixed source register or destination register.
 2. R15 functions as a hardware stack pointer (SP) during exception processing.

Figure 2.1 General Registers

2.1.2 Control Registers

The 32-bit control registers consist of the 32-bit status register (SR), global base register (GBR), and vector base register (VBR). The status register indicates processing states. The global base register functions as a base address for the indirect GBR addressing mode to transfer data to the registers of on-chip peripheral modules. The vector base register functions as the base address of the exception processing vector area (including interrupts). Figure 2.2 shows a control register.

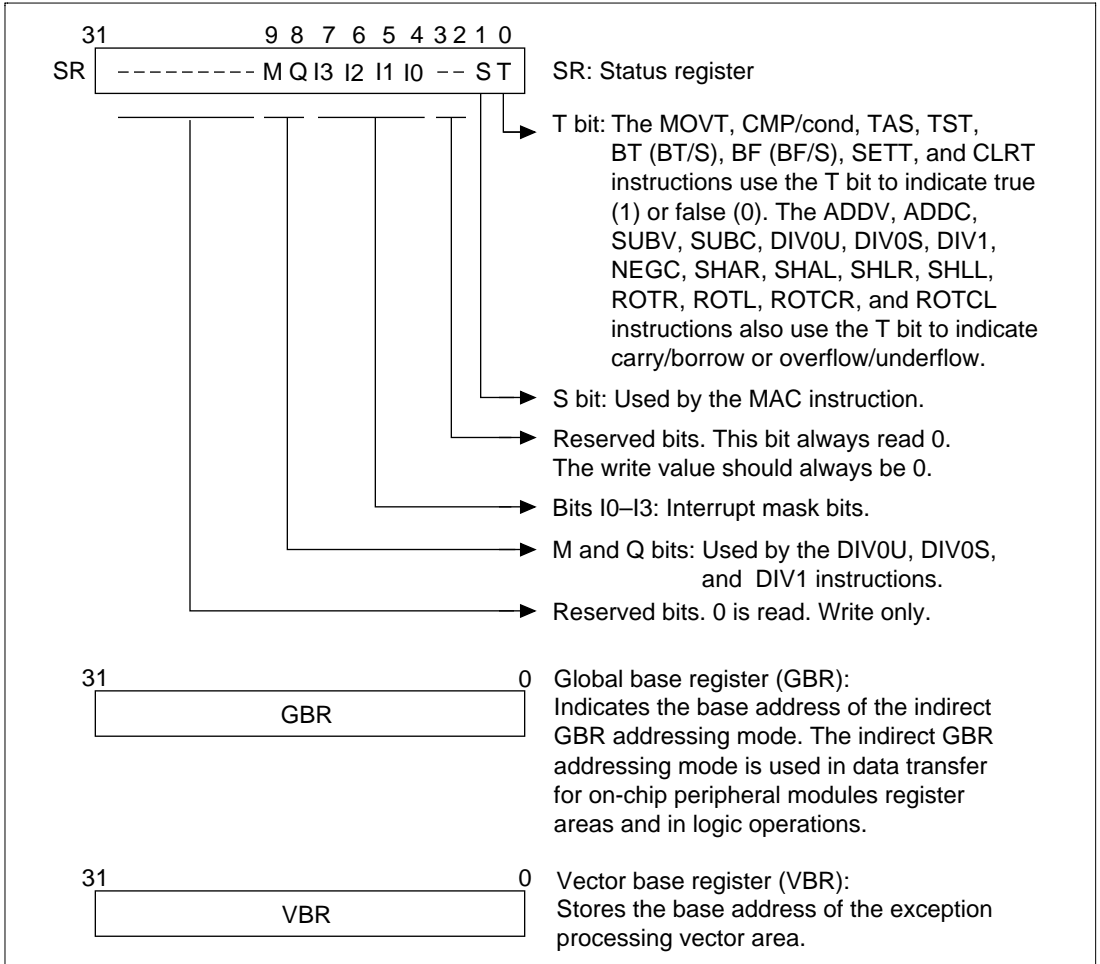


Figure 2.2 Control Registers

2.1.3 System Registers

System registers consist of four 32-bit registers: high and low multiply and accumulate registers (MACH and MACL), the procedure register (PR), and the program counter (PC). The multiply and accumulate registers store the results of multiply and accumulate operations. The procedure register stores the return address from the subroutine procedure. The program counter stores program addresses to control the flow of the processing. Figure 2.3 shows a system register.

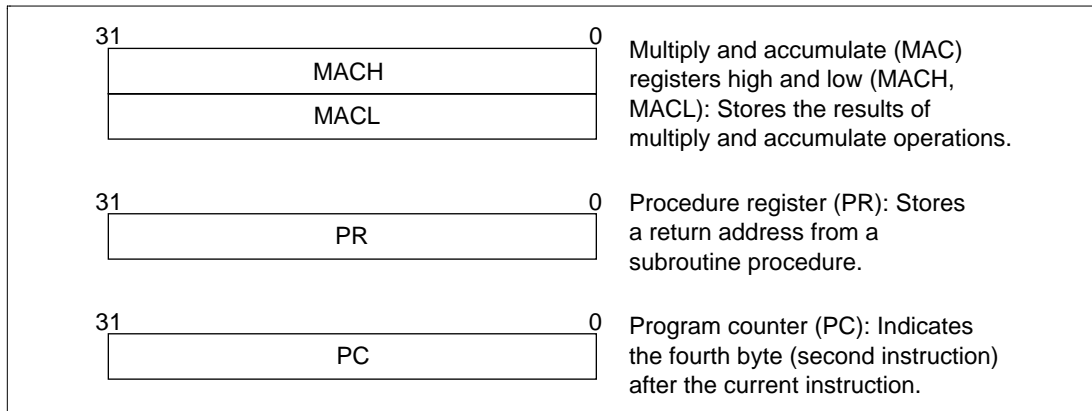


Figure 2.3 System Registers

2.1.4 Initial Values of Registers

Table 2.1 lists the values of the registers after reset.

Table 2.1 Initial Values of Registers

| Classification | Register | Initial Value |
|-------------------|----------------|--|
| General registers | R0–R14 | Undefined |
| | R15 (SP) | Value of the stack pointer in the vector address table |
| Control registers | SR | Bits I3–I0 are 1111 (H'F), reserved bits are 0, and other bits are undefined |
| | GBR | Undefined |
| | VBR | H'00000000 |
| System registers | MACH, MACL, PR | Undefined |
| | PC | Value of the program counter in the vector address table |

2.2 Data Formats

2.2.1 Data Format in Registers

Register operands are always longwords (32 bits). When the memory operand is only a byte (8 bits) or a word (16 bits), it is sign-extended into a longword when loaded into a register (figure 2.4).

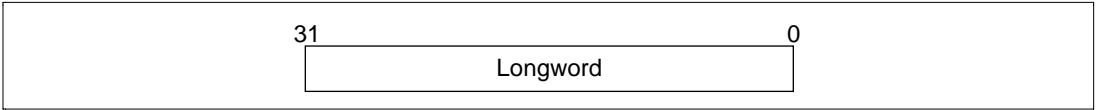


Figure 2.4 Longword Operand

2.2.2 Data Format in Memory

Memory data formats are classified into bytes, words, and longwords. Byte data can be accessed from any address, but an address error will occur if you try to access word data starting from an address other than $2n$ or longword data starting from an address other than $4n$. In such cases, the data accessed cannot be guaranteed. The hardware stack area, referred to by the hardware stack pointer (SP, R15), uses only longword data starting from address $4n$ because this area holds the program counter and status register (figure 2.5).

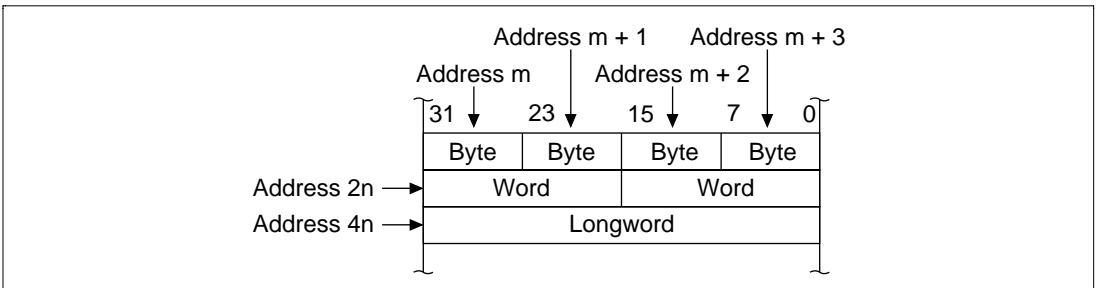


Figure 2.5 Byte, Word, and Longword Alignment

2.2.3 Immediate Data Format

Byte (8-bit) immediate data resides in an instruction code. Immediate data accessed by the MOV, ADD, and CMP/EQ instructions is sign-extended and handled in registers as longword data. Immediate data accessed by the TST, AND, OR, and XOR instructions is zero-extended and handled as longword data. Consequently, AND instructions with immediate data always clear the upper 24 bits of the destination register.

Word or longword immediate data is not located in the instruction code, but instead is stored in a memory table. An immediate data transfer instruction (MOV) accesses the memory table using the PC relative addressing mode with displacement.

2.3 Instruction Features

2.3.1 RISC-Type Instruction Set

All instructions are RISC type. This section details their functions.

16-Bit Fixed Length: All instructions are 16 bits long, increasing program code efficiency.

One Instruction per Cycle: The microprocessor can execute basic instructions in one cycle using the pipeline system. Instructions are executed in 35 ns at 28.7 MHz.

Data Length: Longword is the standard data length for all operations. Memory can be accessed in bytes, words, or longwords. Byte or word data accessed from memory is sign-extended and handled as longword data. Immediate data is sign-extended for arithmetic operations or zero-extended for logic operations. It also is handled as longword data (table 2.2).

Table 2.2 Sign Extension of Word Data

| SH7014/16/17 CPU | Description | Example of Conventional CPU |
|------------------------|--|-----------------------------|
| MOV.W @ (disp, PC), R1 | Data is sign-extended to 32 bits, and R1 becomes H'00001234. It is next operated upon by an ADD instruction. | ADD.W #H'1234, R0 |
| ADD R1, R0 | | |
| | | |
| .DATA.W H'1234 | | |

Note: @ (disp, PC) accesses the immediate data.

Load-Store Architecture: Basic operations are executed between registers. For operations that involve memory access, data is loaded to the registers and executed (load-store architecture). Instructions such as AND that manipulate bits, however, are executed directly in memory.

Delayed Branch Instructions: Unconditional branch instructions are delayed. Executing the instruction that follows the branch instruction and then branching reduces pipeline disruption during branching (table 2.3). There are two types of conditional branch instructions: delayed branch instructions and ordinary branch instructions.

Table 2.3 Delayed Branch Instructions

| SH7014/16/17 CPU | | Description | Example of Conventional CPU | |
|------------------|-------|---|-----------------------------|-------|
| BRA | TRGET | Executes an ADD before branching to TRGET | ADD.W | R1,R0 |
| ADD | R1,R0 | | BRA | TRGET |

Multiplication/Accumulation Operation: 16-bit \times 16-bit \rightarrow 32-bit multiplication operations are executed in one to two cycles. 16-bit \times 16-bit + 64-bit \rightarrow 64-bit multiplication/accumulation operations are executed in two to three cycles. 32-bit \times 32-bit \rightarrow 64-bit and 32-bit \times 32-bit + 64bit \rightarrow 64-bit multiplication/accumulation operations are executed in two to four cycles.

T Bit: The T bit in the status register changes according to the result of the comparison, and in turn is the condition (true/false) that determines if the program will branch. The number of instructions that change the T bit is kept to a minimum to improve the processing speed (table 2.4).

Table 2.4 T Bit

| SH7014/16/17 CPU | | Description | Example of Conventional CPU | |
|------------------|--------|--|-----------------------------|--------|
| CMP/GE | R1,R0 | T bit is set when $R0 \geq R1$. The program branches to TRGET0 when $R0 \geq R1$ and to TRGET1 when $R0 < R1$. | CMP.W | R1,R0 |
| BT | TRGET0 | | BGE | TRGET0 |
| BF | TRGET1 | | BLT | TRGET1 |
| ADD | #1,R0 | T bit is not changed by ADD. T bit is set when $R0 = 0$. The program branches if $R0 = 0$. | SUB.W | #1,R0 |
| CMP/EQ | #0,R0 | | BEQ | TRGET |
| BT | TRGET | | | |

Immediate Data: Byte (8-bit) immediate data resides in instruction code. Word or longword immediate data is not input via instruction codes but is stored in a memory table. An immediate data transfer instruction (MOV) accesses the memory table using the PC relative addressing mode with displacement (table 2.5).

Table 2.5 Immediate Data Accessing

| Classification | SH7014/16/17 CPU | | Example of Conventional CPU |
|------------------|------------------|---------------|-----------------------------|
| 8-bit immediate | MOV | #H' 12,R0 | MOV.B #H' 12,R0 |
| 16-bit immediate | MOV.W | @(disp,PC),R0 | MOV.W #H' 1234,R0 |
| | | | |
| | .DATA.W | H' 1234 | |
| 32-bit immediate | MOV.L | @(disp,PC),R0 | MOV.L #H' 12345678,R0 |
| | | | |
| | .DATA.L | H' 12345678 | |

Note: @(disp, PC) accesses the immediate data.

Absolute Address: When data is accessed by absolute address, the value already in the absolute address is placed in the memory table. Loading the immediate data when the instruction is executed transfers that value to the register and the data is accessed in the indirect register addressing mode (table 2.6).

Table 2.6 Absolute Address Accessing

| Classification | SH7014/16/17 CPU | | Example of Conventional CPU |
|------------------|------------------|---------------|-----------------------------|
| Absolute address | MOV.L | @(disp,PC),R1 | MOV.B @H' 12345678,R0 |
| | MOV.B | @R1,R0 | |
| | | | |
| | .DATA.L | H' 12345678 | |

Note: @(disp,PC) accesses the immediate data.

16-Bit/32-Bit Displacement: When data is accessed by 16-bit or 32-bit displacement, the pre-existing displacement value is placed in the memory table. Loading the immediate data when the instruction is executed transfers that value to the register and the data is accessed in the indirect indexed register addressing mode (table 2.7).

Table 2.7 Displacement Accessing

| Classification | SH7014/16/17 CPU | | Example of Conventional CPU |
|---------------------|------------------|---------------|-----------------------------|
| 16-bit displacement | MOV.W | @(disp,PC),R0 | MOV.W @(H' 1234,R1),R2 |
| | MOV.W | @(R0,R1),R2 | |
| | | | |
| | .DATA.W | H' 1234 | |

Note: @(disp,PC) accesses the immediate data.

2.3.2 Addressing Modes

Table 2.8 describes addressing modes and effective address calculation.

Table 2.8 Addressing Modes and Effective Addresses

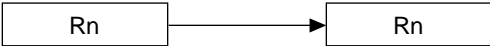
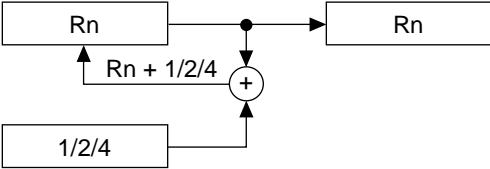
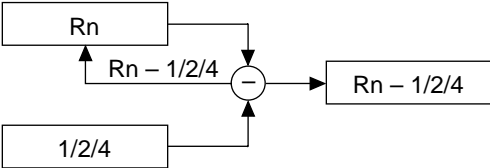
| Addressing Mode | Instruction Format | Effective Addresses Calculation | Equation |
|---|--------------------|--|---|
| Direct register addressing | Rn | The effective address is register Rn. (The operand is the contents of register Rn.) | — |
| Indirect register addressing | @Rn | The effective address is the content of register Rn.  | Rn |
| Post-increment indirect register addressing | @Rn+ | The effective address is the content of register Rn. A constant is added to the content of Rn after the instruction is executed. 1 is added for a byte operation, 2 for a word operation, and 4 for a longword operation.  | Rn (After the instruction executes) Byte: Rn + 1 → Rn Word: Rn + 2 → Rn Longword: Rn + 4 → Rn |
| Pre-decrement indirect register addressing | @-Rn | The effective address is the value obtained by subtracting a constant from Rn. 1 is subtracted for a byte operation, 2 for a word operation, and 4 for a longword operation.  | Byte: Rn - 1 → Rn Word: Rn - 2 → Rn Longword: Rn - 4 → Rn (Instruction executed with Rn after calculation) |

Table 2.8 Addressing Modes and Effective Addresses (cont)

| Addressing Mode | Instruction Format | Effective Addresses Calculation | Equation |
|--|--------------------|--|--|
| Indirect register addressing with displacement | @(disp:4, Rn) | <p>The effective address is Rn plus a 4-bit displacement (disp). The value of disp is zero-extended, and remains the same for a byte operation, is doubled for a word operation, and is quadrupled for a longword operation.</p> | <p>Byte: $Rn + disp$</p> <p>Word: $Rn + disp \times 2$</p> <p>Longword: $Rn + disp \times 4$</p> |
| Indirect indexed register addressing | @(R0, Rn) | <p>The effective address is the Rn value plus R0.</p> | $Rn + R0$ |
| Indirect GBR addressing with displacement | @(disp:8, GBR) | <p>The effective address is the GBR value plus an 8-bit displacement (disp). The value of disp is zero-extended, and remains the same for a byte operation, is doubled for a word operation, and is quadrupled for a longword operation.</p> | <p>Byte: $GBR + disp$</p> <p>Word: $GBR + disp \times 2$</p> <p>Longword: $GBR + disp \times 4$</p> |

Table 2.8 Addressing Modes and Effective Addresses (cont)

| Addressing Mode | Instruction Format | Effective Addresses Calculation | Equation |
|--|--------------------|--|---|
| Indirect indexed GBR addressing | @(R0, GBR) | The effective address is the GBR value plus the R0. | $GBR + R0$ |
| | | | |
| PC relative addressing with displacement | @(disp:8, PC) | The effective address is the PC value plus an 8-bit displacement (disp). The value of disp is zero-extended, and is doubled for a word operation, and quadrupled for a longword operation. For a longword operation, the lowest two bits of the PC value are masked. | Word: $PC + disp \times 2$ Longword: $PC \& H'FFFFFFC + disp \times 4$ |
| | | | |

Table 2.8 Addressing Modes and Effective Addresses (cont)

| Addressing Mode | Instruction Format | Effective Addresses Calculation | Equation |
|------------------------|--------------------|--|----------------------|
| PC relative addressing | disp:8 | The effective address is the PC value sign-extended with an 8-bit displacement (disp), doubled, and added to the PC value. | $PC + disp \times 2$ |
| | | | |
| | disp:12 | The effective address is the PC value sign-extended with a 12-bit displacement (disp), doubled, and added to the PC value. | $PC + disp \times 2$ |
| | | | |
| | Rn | The effective address is the register PC value plus Rn. | $PC + Rn$ |
| | | | |
| Immediate addressing | #imm:8 | The 8-bit immediate data (imm) for the TST, AND, OR, and XOR instructions are zero-extended. | — |
| | #imm:8 | The 8-bit immediate data (imm) for the MOV, ADD, and CMP/EQ instructions are sign-extended. | — |
| | #imm:8 | The 8-bit immediate data (imm) for the TRAPA instruction is zero-extended and is quadrupled. | — |

2.3.3 Instruction Format

Table 2.9 lists the instruction formats for the source operand and the destination operand. The meaning of the operand depends on the instruction code. The symbols are used as follows:

- xxxx: Instruction code
- mmmm: Source register
- nnnn: Destination register
- iiiii: Immediate data
- dddd: Displacement

Table 2.9 Instruction Formats

| Instruction Formats | Source Operand | Destination Operand | Example |
|---|--|---------------------------------------|----------------|
| 0 format <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 5px auto;"> 15 0 <div style="display: flex; justify-content: space-around; width: 100%;"> xxxx xxxx xxxx xxxx </div> </div> | — | — | NOP |
| n format <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 5px auto;"> 15 0 <div style="display: flex; justify-content: space-around; width: 100%;"> xxxx nnnn xxxx xxxx </div> </div> | — | nnnn: Direct register | MOVT Rn |
| | Control register or system register | nnnn: Direct register | STS MACH, Rn |
| | Control register or system register | nnnn: Indirect pre-decrement register | STC.L SR, @-Rn |
| m format <div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 5px auto;"> 15 0 <div style="display: flex; justify-content: space-around; width: 100%;"> xxxx mmmm xxxx xxxx </div> </div> | mmmm: Direct register | Control register or system register | LDC Rm, SR |
| | mmmm: Indirect post-increment register | Control register or system register | LDC.L @Rm+, SR |
| | mmmm: Direct register | — | JMP @Rm |
| | mmmm: PC relative using Rm | — | BRAF Rm |

Table 2.9 Instruction Formats (cont)

| Instruction Formats | Source Operand | Destination Operand | Example | | | | |
|--|---|---|-----------------------|------|---|---|-----------------------|
| nm format | mmmm: Direct register | nnnn: Direct register | ADD Rm, Rn | | | | |
| 15 <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td style="width: 25%;">xxxx</td> <td style="width: 25%;">nnnn</td> <td style="width: 25%;">mmmm</td> <td style="width: 25%;">xxxx</td> </tr> </table> 0 | xxxx | nnnn | mmmm | xxxx | mmmm: Direct register | nnnn: Indirect register | MOV.L Rm, @Rn |
| xxxx | nnnn | mmmm | xxxx | | | | |
| | mmmm: Indirect post-increment register (multiply/accumulate) nnnn*: Indirect post-increment register (multiply/accumulate) | MACH, MACL | MAC.W @Rm+, @Rn+ | | | | |
| | mmmm: Indirect post-increment register | nnnn: Direct register | MOV.L @Rm+, Rn | | | | |
| | mmmm: Direct register | nnnn: Indirect pre-decrement register | MOV.L Rm, @-Rn | | | | |
| | mmmm: Direct register | nnnn: Indirect indexed register | MOV.L Rm, @(R0, Rn) | | | | |
| md format | mmmmdddd: indirect register with displacement | R0 (Direct register) | MOV.B @(disp, Rm), R0 | | | | |
| 15 <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td style="width: 25%;">xxxx</td> <td style="width: 25%;">xxxx</td> <td style="width: 25%;">mmmm</td> <td style="width: 25%;">dddd</td> </tr> </table> 0 | xxxx | xxxx | mmmm | dddd | R0 (Direct register) | nnnndddd: Indirect register with displacement | MOV.B R0, @(disp, Rn) |
| xxxx | xxxx | mmmm | dddd | | | | |
| nd4 format | | | | | | | |
| 15 <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td style="width: 25%;">xxxx</td> <td style="width: 25%;">xxxx</td> <td style="width: 25%;">nnnn</td> <td style="width: 25%;">dddd</td> </tr> </table> 0 | xxxx | xxxx | nnnn | dddd | | | |
| xxxx | xxxx | nnnn | dddd | | | | |
| nmd format | mmmm: Direct register | nnnndddd: Indirect register with displacement | MOV.L Rm, @(disp, Rn) | | | | |
| 15 <table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td style="width: 25%;">xxxx</td> <td style="width: 25%;">nnnn</td> <td style="width: 25%;">mmmm</td> <td style="width: 25%;">dddd</td> </tr> </table> 0 | xxxx | nnnn | mmmm | dddd | mmmmdddd: Indirect register with displacement | nnnn: Direct register | MOV.L @(disp, Rm), Rn |
| xxxx | nnnn | mmmm | dddd | | | | |

Note: In multiply/accumulate instructions, nnnn is the source register.

Table 2.9 Instruction Formats (cont)

| Instruction Formats | Source Operand | Destination Operand | Example |
|--|---|---|---------------------------------------|
| d format 15 0 <div style="border: 1px solid black; padding: 2px; display: inline-block;"> xxxx xxxx dddd dddd </div> | ddddddd: Indirect GBR with displacement | R0 (Direct register) | MOV.L @(disp,GBR),R0 |
| | R0(Direct register) | ddddddd: Indirect GBR with displacement | MOV.L R0,@(disp,GBR) |
| | ddddddd: PC relative with displacement | R0 (Direct register) | MOVA @(disp,PC),R0 |
| | ddddddd: PC relative | — | BF label |
| d12 format 15 0 <div style="border: 1px solid black; padding: 2px; display: inline-block;"> xxxx dddd dddd dddd </div> | dddddddddd: PC relative | — | BRA label (label = disp + PC) |
| nd8 format 15 0 <div style="border: 1px solid black; padding: 2px; display: inline-block;"> xxxx nnnn dddd dddd </div> | ddddddd: PC relative with displacement | nnnn: Direct register | MOV.L @(disp,PC),Rn |
| i format 15 0 <div style="border: 1px solid black; padding: 2px; display: inline-block;"> xxxx xxxx iiii iiii </div> | iiiiiii: Immediate | Indirect indexed GBR | AND.B #imm,@(R0,GBR) |
| | iiiiiii: Immediate | R0 (Direct register) | AND #imm,R0 |
| | iiiiiii: Immediate | — | TRAPA #imm |
| ni format 15 0 <div style="border: 1px solid black; padding: 2px; display: inline-block;"> xxxx nnnn iiii iiii </div> | iiiiiii: Immediate | nnnn: Direct register | ADD #imm,Rn |

2.4 Instruction Set by Classification

Table 2.10 Classification of Instructions

| Classification | Types | Operation | | No. of Instructions |
|-----------------------|-----------------------------------|-----------|--|---------------------|
| | | Code | Function | |
| Data transfer | 5 | MOV | Data transfer, immediate data transfer, peripheral module data transfer, structure data transfer | 39 |
| | | MOVA | Effective address transfer | |
| | | MOVT | T bit transfer | |
| | | SWAP | Swap of upper and lower bytes | |
| | | XTRCT | Extraction of the middle of registers connected | |
| Arithmetic operations | 21 | ADD | Binary addition | 33 |
| | | ADDC | Binary addition with carry | |
| | | ADDV | Binary addition with overflow check | |
| | | CMP/cond | Comparison | |
| | | DIV1 | Division | |
| | | DIV0S | Initialization of signed division | |
| | | DIV0U | Initialization of unsigned division | |
| | | DMULS | Signed double-length multiplication | |
| | | DMULU | Unsigned double-length multiplication | |
| | | DT | Decrement and test | |
| | | EXTS | Sign extension | |
| | | EXTU | Zero extension | |
| | | MAC | Multiply/accumulate, double-length multiply/accumulate operation | |
| | | MUL | Double-length multiply operation | |
| | | MULS | Signed multiplication | |
| | | MULU | Unsigned multiplication | |
| | | NEG | Negation | |
| | | NEGC | Negation with borrow | |
| | | SUB | Binary subtraction | |
| | | SUBC | Binary subtraction with borrow | |
| SUBV | Binary subtraction with underflow | | | |

Table 2.10 Classification of Instructions (cont)

| Classification | Types | Operation | | No. of Instructions |
|------------------|-------|-----------|---|---------------------|
| | | Code | Function | |
| Logic operations | 6 | AND | Logical AND | 14 |
| | | NOT | Bit inversion | |
| | | OR | Logical OR | |
| | | TAS | Memory test and bit set | |
| | | TST | Logical AND and T bit set | |
| | | XOR | Exclusive OR | |
| Shift | 10 | ROTL | One-bit left rotation | 14 |
| | | ROTR | One-bit right rotation | |
| | | ROTCL | One-bit left rotation with T bit | |
| | | ROTCR | One-bit right rotation with T bit | |
| | | SHAL | One-bit arithmetic left shift | |
| | | SHAR | One-bit arithmetic right shift | |
| | | SHLL | One-bit logical left shift | |
| | | SHLLn | n-bit logical left shift | |
| | | SHLR | One-bit logical right shift | |
| | | SHLRn | n-bit logical right shift | |
| Branch | 9 | BF | Conditional branch, conditional branch with delay (Branch when T = 0) | 11 |
| | | BT | Conditional branch, conditional branch with delay (Branch when T = 1) | |
| | | BRA | Unconditional branch | |
| | | BRAF | Unconditional branch | |
| | | BSR | Branch to subroutine procedure | |
| | | BSRF | Branch to subroutine procedure | |
| | | JMP | Unconditional branch | |
| | | JSR | Branch to subroutine procedure | |
| | | RTS | Return from subroutine procedure | |

Table 2.10 Classification of Instructions (cont)

| Classification | Types | Operation | | No. of Instructions |
|----------------|-------|-----------|----------------------------------|---------------------|
| | | Code | Function | |
| System control | 11 | CLRT | T bit clear | 31 |
| | | CLRMAC | MAC register clear | |
| | | LDC | Load to control register | |
| | | LDS | Load to system register | |
| | | NOP | No operation | |
| | | RTE | Return from exception processing | |
| | | SETT | T bit set | |
| | | SLEEP | Shift into power-down mode | |
| | | STC | Storing control register data | |
| | | STS | Storing system register data | |
| | | TRAPA | Trap exception handling | |
| Total: | | 62 | | 142 |

Table 2.11 shows the format used in tables 2.12 to 2.17, which list instruction codes, operation, and execution states in order by classification.

Table 2.11 Instruction Code Format

| Item | Format | Explanation |
|------------------|--------------------|---|
| Instruction | OP . Sz SRC , DEST | OP: Operation code Sz: Size (B: byte, W: word, or L: longword) SRC: Source DEST: Destination Rm: Source register Rn: Destination register imm: Immediate data disp: Displacement* ¹ |
| Instruction code | MSB ↔ LSB | m m m m: Source register n n n n: Destination register 0000: R0 0001: R1 . . . 1111: R15 iiii: Immediate data dddd: Displacement |
| Operation | →, ← | Direction of transfer |
| | (xx) | Memory operand |
| | M/Q/T | Flag bits in the SR |
| | & | Logical AND of each bit |
| | | Logical OR of each bit |
| | ^ | Exclusive OR of each bit |
| | ~ | Logical NOT of each bit |
| | <<n | n-bit left shift |
| | >>n | n-bit right shift |
| Execution cycles | — | Value when no wait states are inserted* ² |
| T bit | — | Value of T bit after instruction is executed. An em-dash (—) in the column means no change. |

Notes: 1. Depending on the operand size, displacement is scaled $\times 1$, $\times 2$, or $\times 4$. For details, see the *SH-1/SH-2/SH-DSP Programming Manual*.

2. Instruction execution cycles: The execution cycles shown in the table are minimums. The actual number of cycles may be increased when (1) contention occurs between instruction fetches and data access, or (2) when the destination register of the load instruction (memory → register) and the register used by the next instruction are the same.

Table 2.12 Data Transfer Instructions

| Instruction | Instruction Code | Operation | Execution Cycles | T Bit |
|---------------------|------------------|---|------------------|-------|
| MOV #imm,Rn | 1110nnnniiiiiii | #imm → Sign extension → Rn | 1 | — |
| MOV.W @(disp,PC),Rn | 1001nnnnddddddd | (disp × 2 + PC) → Sign extension → Rn | 1 | — |
| MOV.L @(disp,PC),Rn | 1101nnnnddddddd | (disp × 4 + PC) → Rn | 1 | — |
| MOV Rm,Rn | 0110nnnnnnmm0011 | Rm → Rn | 1 | — |
| MOV.B Rm,@Rn | 0010nnnnnnmm0000 | Rm → (Rn) | 1 | — |
| MOV.W Rm,@Rn | 0010nnnnnnmm0001 | Rm → (Rn) | 1 | — |
| MOV.L Rm,@Rn | 0010nnnnnnmm0010 | Rm → (Rn) | 1 | — |
| MOV.B @Rm,Rn | 0110nnnnnnmm0000 | (Rm) → Sign extension → Rn | 1 | — |
| MOV.W @Rm,Rn | 0110nnnnnnmm0001 | (Rm) → Sign extension → Rn | 1 | — |
| MOV.L @Rm,Rn | 0110nnnnnnmm0010 | (Rm) → Rn | 1 | — |
| MOV.B Rm,@-Rn | 0010nnnnnnmm0100 | Rn-1 → Rn, Rm → (Rn) | 1 | — |
| MOV.W Rm,@-Rn | 0010nnnnnnmm0101 | Rn-2 → Rn, Rm → (Rn) | 1 | — |
| MOV.L Rm,@-Rn | 0010nnnnnnmm0110 | Rn-4 → Rn, Rm → (Rn) | 1 | — |
| MOV.B @Rm+,Rn | 0110nnnnnnmm0100 | (Rm) → Sign extension → Rn, Rm + 1 → Rm | 1 | — |
| MOV.W @Rm+,Rn | 0110nnnnnnmm0101 | (Rm) → Sign extension → Rn, Rm + 2 → Rm | 1 | — |
| MOV.L @Rm+,Rn | 0110nnnnnnmm0110 | (Rm) → Rn, Rm + 4 → Rm | 1 | — |
| MOV.B R0,@(disp,Rn) | 10000000nnnndddd | R0 → (disp + Rn) | 1 | — |
| MOV.W R0,@(disp,Rn) | 10000001nnnndddd | R0 → (disp × 2 + Rn) | 1 | — |
| MOV.L Rm,@(disp,Rn) | 0001nnnnnnmmdddd | Rm → (disp × 4 + Rn) | 1 | — |
| MOV.B @(disp,Rm),R0 | 10000100nnnndddd | (disp + Rm) → Sign extension → R0 | 1 | — |
| MOV.W @(disp,Rm),R0 | 10000101nnnndddd | (disp × 2 + Rm) → Sign extension → R0 | 1 | — |
| MOV.L @(disp,Rm),Rn | 0101nnnnnnmmdddd | (disp × 4 + Rm) → Rn | 1 | — |
| MOV.B Rm,@(R0,Rn) | 0000nnnnnnmm0100 | Rm → (R0 + Rn) | 1 | — |

Table 2.12 Data Transfer Instructions (cont)

| Instruction | Instruction Code | Operation | Execution Cycles | T Bit |
|----------------------|-------------------------|--|-------------------------|--------------|
| MOV.W Rm,@(R0,Rn) | 0000nnnnmmmm0101 | Rm → (R0 + Rn) | 1 | — |
| MOV.L Rm,@(R0,Rn) | 0000nnnnmmmm0110 | Rm → (R0 + Rn) | 1 | — |
| MOV.B @(R0,Rm),Rn | 0000nnnnmmmm1100 | (R0 + Rm) → Sign extension → Rn | 1 | — |
| MOV.W @(R0,Rm),Rn | 0000nnnnmmmm1101 | (R0 + Rm) → Sign extension → Rn | 1 | — |
| MOV.L @(R0,Rm),Rn | 0000nnnnmmmm1110 | (R0 + Rm) → Rn | 1 | — |
| MOV.B R0,@(disp,GBR) | 11000000dddddddd | R0 → (disp + GBR) | 1 | — |
| MOV.W R0,@(disp,GBR) | 11000001dddddddd | R0 → (disp × 2 + GBR) | 1 | — |
| MOV.L R0,@(disp,GBR) | 11000010dddddddd | R0 → (disp × 4 + GBR) | 1 | — |
| MOV.B @(disp,GBR),R0 | 11000100dddddddd | (disp + GBR) → Sign extension → R0 | 1 | — |
| MOV.W @(disp,GBR),R0 | 11000101dddddddd | (disp × 2 + GBR) → Sign extension → R0 | 1 | — |
| MOV.L @(disp,GBR),R0 | 11000110dddddddd | (disp × 4 + GBR) → R0 | 1 | — |
| MOVA @(disp,PC),R0 | 11000111dddddddd | disp × 4 + PC → R0 | 1 | — |
| MOVT Rn | 0000nnnn00101001 | T → Rn | 1 | — |
| SWAP.B Rm,Rn | 0110nnnnmmmm1000 | Rm → Swap the bottom two bytes → Rn | 1 | — |
| SWAP.W Rm,Rn | 0110nnnnmmmm1001 | Rm → Swap two consecutive words → Rn | 1 | — |
| XTRCT Rm,Rn | 0010nnnnmmmm1101 | Rm: Middle 32 bits of Rn → Rn | 1 | — |

Table 2.13 Arithmetic Operation Instructions

| Instruction | | Instruction Code | Operation | Execution Cycles | T Bit |
|-------------|----------|------------------|--|------------------|--------------------|
| ADD | Rm, Rn | 0011nnnnnnmm1100 | $Rn + Rm \rightarrow Rn$ | 1 | — |
| ADD | #imm, Rn | 0111nnnniiiiiiii | $Rn + imm \rightarrow Rn$ | 1 | — |
| ADDC | Rm, Rn | 0011nnnnnnmm1110 | $Rn + Rm + T \rightarrow Rn$, Carry $\rightarrow T$ | 1 | Carry |
| ADDV | Rm, Rn | 0011nnnnnnmm1111 | $Rn + Rm \rightarrow Rn$, Overflow $\rightarrow T$ | 1 | Overflow |
| CMP/EQ | #imm, R0 | 10001000iiiiiiii | If $R0 = imm$, $1 \rightarrow T$ | 1 | Comparison result |
| CMP/EQ | Rm, Rn | 0011nnnnnnmm0000 | If $Rn = Rm$, $1 \rightarrow T$ | 1 | Comparison result |
| CMP/HS | Rm, Rn | 0011nnnnnnmm0010 | If $Rn \geq Rm$ with unsigned data, $1 \rightarrow T$ | 1 | Comparison result |
| CMP/GE | Rm, Rn | 0011nnnnnnmm0011 | If $Rn \geq Rm$ with signed data, $1 \rightarrow T$ | 1 | Comparison result |
| CMP/HI | Rm, Rn | 0011nnnnnnmm0110 | If $Rn > Rm$ with unsigned data, $1 \rightarrow T$ | 1 | Comparison result |
| CMP/GT | Rm, Rn | 0011nnnnnnmm0111 | If $Rn > Rm$ with signed data, $1 \rightarrow T$ | 1 | Comparison result |
| CMP/PL | Rn | 0100nnnn00010101 | If $Rn > 0$, $1 \rightarrow T$ | 1 | Comparison result |
| CMP/PZ | Rn | 0100nnnn00010001 | If $Rn \geq 0$, $1 \rightarrow T$ | 1 | Comparison result |
| CMP/STR | Rm, Rn | 0010nnnnnnmm1100 | If Rn and Rm have an equivalent byte, $1 \rightarrow T$ | 1 | Comparison result |
| DIV1 | Rm, Rn | 0011nnnnnnmm0100 | Single-step division (Rn/Rm) | 1 | Calculation result |
| DIV0S | Rm, Rn | 0010nnnnnnmm0111 | MSB of Rn $\rightarrow Q$, MSB of Rm $\rightarrow M$, $M \wedge Q \rightarrow T$ | 1 | Calculation result |
| DIV0U | | 000000000011001 | $0 \rightarrow M/Q/T$ | 1 | 0 |

Table 2.13 Arithmetic Operation Instructions (cont)

| Instruction | Instruction Code | Operation | Execution Cycles | T Bit |
|-----------------|--------------------|--|------------------|-------------------|
| DMULS.L Rm,Rn | 0011nnnnnnmmmm1101 | Signed operation of $Rn \times Rm \rightarrow MACH, MACL$ $32 \times 32 \rightarrow 64$ bit | 2 to 4* | — |
| DMULU.L Rm,Rn | 0011nnnnnnmmmm0101 | Unsigned operation of $Rn \times Rm \rightarrow MACH, MACL$ $32 \times 32 \rightarrow 64$ bit | 2 to 4* | — |
| DT Rn | 0100nnnn00010000 | $Rn - 1 \rightarrow Rn$, when Rn is 0, $1 \rightarrow T$. When Rn is nonzero, $0 \rightarrow T$ | 1 | Comparison result |
| EXTS.B Rm,Rn | 0110nnnnnnmmmm1110 | A byte in Rm is sign-extended $\rightarrow Rn$ | 1 | — |
| EXTS.W Rm,Rn | 0110nnnnnnmmmm1111 | A word in Rm is sign-extended $\rightarrow Rn$ | 1 | — |
| EXTU.B Rm,Rn | 0110nnnnnnmmmm1100 | A byte in Rm is zero-extended $\rightarrow Rn$ | 1 | — |
| EXTU.W Rm,Rn | 0110nnnnnnmmmm1101 | A word in Rm is zero-extended $\rightarrow Rn$ | 1 | — |
| MAC.L @Rm+,@Rn+ | 0000nnnnnnmmmm1111 | Signed operation of $(Rn) \times (Rm) + MAC \rightarrow MAC$ $32 \times 32 \rightarrow 64$ bit | 3/(2 to 4)* | — |
| MAC.W @Rm+,@Rn+ | 0100nnnnnnmmmm1111 | Signed operation of $(Rn) \times (Rm) + MAC \rightarrow MAC$ $16 \times 16 + 64 \rightarrow 64$ bit | 3/(2)* | — |
| MUL.L Rm,Rn | 0000nnnnnnmmmm0111 | $Rn \times Rm \rightarrow MACL, 32 \times 32 \rightarrow 32$ bit | 2 to 4* | — |
| MULS.W Rm,Rn | 0010nnnnnnmmmm1111 | Signed operation of $Rn \times Rm \rightarrow MAC$ $16 \times 16 \rightarrow 32$ bit | 1 to 3* | — |
| MULU.W Rm,Rn | 0010nnnnnnmmmm1110 | Unsigned operation of $Rn \times Rm \rightarrow MAC$ $16 \times 16 \rightarrow 32$ bit | 1 to 3* | — |
| NEG Rm,Rn | 0110nnnnnnmmmm1011 | $0 - Rm \rightarrow Rn$ | 1 | — |
| NEGC Rm,Rn | 0110nnnnnnmmmm1010 | $0 - Rm - T \rightarrow Rn$, Borrow $\rightarrow T$ | 1 | Borrow |

Table 2.13 Arithmetic Operation Instructions (cont)

| Instruction | | Instruction Code | Operation | Execution Cycles | T Bit |
|--------------------|--------|-------------------------|------------------------------|-------------------------|--------------|
| SUB | Rm, Rn | 0011nnnnnnnnmm1000 | Rn-Rm → Rn | 1 | — |
| SUBC | Rm, Rn | 0011nnnnnnnnmm1010 | Rn-Rm-T → Rn, Borrow → T | 1 | Borrow |
| SUBV | Rm, Rn | 0011nnnnnnnnmm1011 | Rn-Rm → Rn, Underflow → T | 1 | Overflow |

Note: The normal minimum number of execution cycles. (The number in parentheses is the number of cycles when there is contention with following instructions.)

Table 2.14 Logic Operation Instructions

| Instruction | | Instruction Code | Operation | Execution Cycles | T Bit |
|-------------|------------------|------------------|--|------------------|-------------|
| AND | Rm, Rn | 0010nnnnmmmm1001 | $Rn \& Rm \rightarrow Rn$ | 1 | — |
| AND | #imm, R0 | 11001001iiiiiii | $R0 \& imm \rightarrow R0$ | 1 | — |
| AND.B | #imm, @(R0, GBR) | 11001101iiiiiii | $(R0 + GBR) \& imm \rightarrow (R0 + GBR)$ | 3 | — |
| NOT | Rm, Rn | 0110nnnnmmmm0111 | $\sim Rm \rightarrow Rn$ | 1 | — |
| OR | Rm, Rn | 0010nnnnmmmm1011 | $Rn Rm \rightarrow Rn$ | 1 | — |
| OR | #imm, R0 | 11001011iiiiiii | $R0 imm \rightarrow R0$ | 1 | — |
| OR.B | #imm, @(R0, GBR) | 11001111iiiiiii | $(R0 + GBR) imm \rightarrow (R0 + GBR)$ | 3 | — |
| TAS.B | @Rn | 0100nnnn00011011 | If (Rn) is 0, $1 \rightarrow T$; $1 \rightarrow$ MSB of (Rn)* | 4 | Test result |
| TST | Rm, Rn | 0010nnnnmmmm1000 | $Rn \& Rm$; if the result is 0, $1 \rightarrow T$ | 1 | Test result |
| TST | #imm, R0 | 11001000iiiiiii | $R0 \& imm$; if the result is 0, $1 \rightarrow T$ | 1 | Test result |
| TST.B | #imm, @(R0, GBR) | 11001100iiiiiii | $(R0 + GBR) \& imm$; if the result is 0, $1 \rightarrow T$ | 3 | Test result |
| XOR | Rm, Rn | 0010nnnnmmmm1010 | $Rn \wedge Rm \rightarrow Rn$ | 1 | — |
| XOR | #imm, R0 | 11001010iiiiiii | $R0 \wedge imm \rightarrow R0$ | 1 | — |
| XOR.B | #imm, @(R0, GBR) | 11001110iiiiiii | $(R0 + GBR) \wedge imm \rightarrow (R0 + GBR)$ | 3 | — |

Note: The on-chip DMAC bus cycles are not inserted between the read and write cycles of TAS instruction execution.

Table 2.15 Shift Instructions

| Instruction | Instruction Code | Operation | Execution Cycles | T Bit |
|--------------------|-------------------------|---|-------------------------|--------------|
| ROTL | Rn 0100nnnn00000100 | $T \leftarrow Rn \leftarrow \text{MSB}$ | 1 | MSB |
| ROTR | Rn 0100nnnn00000101 | $\text{LSB} \rightarrow Rn \rightarrow T$ | 1 | LSB |
| ROTCL | Rn 0100nnnn00100100 | $T \leftarrow Rn \leftarrow T$ | 1 | MSB |
| ROTCR | Rn 0100nnnn00100101 | $T \rightarrow Rn \rightarrow T$ | 1 | LSB |
| SHAL | Rn 0100nnnn00100000 | $T \leftarrow Rn \leftarrow 0$ | 1 | MSB |
| SHAR | Rn 0100nnnn00100001 | $\text{MSB} \rightarrow Rn \rightarrow T$ | 1 | LSB |
| SHLL | Rn 0100nnnn00000000 | $T \leftarrow Rn \leftarrow 0$ | 1 | MSB |
| SHLR | Rn 0100nnnn00000001 | $0 \rightarrow Rn \rightarrow T$ | 1 | LSB |
| SHLL2 | Rn 0100nnnn00001000 | $Rn \ll 2 \rightarrow Rn$ | 1 | — |
| SHLR2 | Rn 0100nnnn00001001 | $Rn \gg 2 \rightarrow Rn$ | 1 | — |
| SHLL8 | Rn 0100nnnn00011000 | $Rn \ll 8 \rightarrow Rn$ | 1 | — |
| SHLR8 | Rn 0100nnnn00011001 | $Rn \gg 8 \rightarrow Rn$ | 1 | — |
| SHLL16 | Rn 0100nnnn00101000 | $Rn \ll 16 \rightarrow Rn$ | 1 | — |
| SHLR16 | Rn 0100nnnn00101001 | $Rn \gg 16 \rightarrow Rn$ | 1 | — |

Table 2.16 Branch Instructions

| Instruction | | Instruction Code | Operation | Exec. Cycles | T Bit |
|-------------|-------|------------------|---|--------------|-------|
| BF | label | 10001011dddddddd | If T = 0, disp × 2 + PC → PC; if T = 1, nop | 3/1* | — |
| BF/S | label | 10001111dddddddd | Delayed branch, if T = 0, disp × 2 + PC → PC; if T = 1, nop | 3/1* | — |
| BT | label | 10001001dddddddd | If T = 1, disp × 2 + PC → PC; if T = 0, nop | 3/1* | — |
| BT/S | label | 10001101dddddddd | Delayed branch, if T = 1, disp × 2 + PC → PC; if T = 0, nop | 2/1* | — |
| BRA | label | 1010dddddddddddd | Delayed branch, disp × 2 + PC → PC | 2 | — |
| BRAF | Rm | 0000mmmm00100011 | Delayed branch, Rm + PC → PC | 2 | — |
| BSR | label | 1011dddddddddddd | Delayed branch, PC → PR, disp × 2 + PC → PC | 2 | — |
| BSRF | Rm | 0000mmmm00000011 | Delayed branch, PC → PR, Rm + PC → PC | 2 | — |
| JMP | @Rm | 0100mmmm00101011 | Delayed branch, Rm → PC | 2 | — |
| JSR | @Rm | 0100mmmm00001011 | Delayed branch, PC → PR, Rm → PC | 2 | — |
| RTS | | 000000000001011 | Delayed branch, PR → PC | 2 | — |

Note: One state when it does not branch.

Table 2.17 System Control Instructions

| Instruction | Instruction Code | Operation | Exec. Cycles | T Bit |
|------------------|------------------|---------------------------------------|--------------|-------|
| CLRT | 000000000001000 | 0 → T | 1 | 0 |
| CLRMACH | 000000000101000 | 0 → MACH, MACL | 1 | — |
| LDC Rm, SR | 0100mmmm00001110 | Rm → SR | 1 | LSB |
| LDC Rm, GBR | 0100mmmm00011110 | Rm → GBR | 1 | — |
| LDC Rm, VBR | 0100mmmm00101110 | Rm → VBR | 1 | — |
| LDC.L @Rm+, SR | 0100mmmm00000111 | (Rm) → SR, Rm + 4 → Rm | 3 | LSB |
| LDC.L @Rm+, GBR | 0100mmmm00010111 | (Rm) → GBR, Rm + 4 → Rm | 3 | — |
| LDC.L @Rm+, VBR | 0100mmmm00100111 | (Rm) → VBR, Rm + 4 → Rm | 3 | — |
| LDS Rm, MACH | 0100mmmm00001010 | Rm → MACH | 1 | — |
| LDS Rm, MACL | 0100mmmm00011010 | Rm → MACL | 1 | — |
| LDS Rm, PR | 0100mmmm00101010 | Rm → PR | 1 | — |
| LDS.L @Rm+, MACH | 0100mmmm00000110 | (Rm) → MACH, Rm + 4 → Rm | 1 | — |
| LDS.L @Rm+, MACL | 0100mmmm00010110 | (Rm) → MACL, Rm + 4 → Rm | 1 | — |
| LDS.L @Rm+, PR | 0100mmmm00100110 | (Rm) → PR, Rm + 4 → Rm | 1 | — |
| NOP | 000000000001001 | No operation | 1 | — |
| RTE | 000000000101011 | Delayed branch, stack area → PC/SR | 4 | — |
| SETT | 000000000011000 | 1 → T | 1 | 1 |
| SLEEP | 000000000011011 | Sleep | 3* | — |
| STC SR, Rn | 0000nnnn00000010 | SR → Rn | 1 | — |
| STC GBR, Rn | 0000nnnn00010010 | GBR → Rn | 1 | — |
| STC VBR, Rn | 0000nnnn00100010 | VBR → Rn | 1 | — |
| STC.L SR, @-Rn | 0100nnnn00000011 | Rn-4 → Rn, SR → (Rn) | 2 | — |
| STC.L GBR, @-Rn | 0100nnnn00010011 | Rn-4 → Rn, GBR → (Rn) | 2 | — |
| STC.L VBR, @-Rn | 0100nnnn00100011 | Rn-4 → Rn, BR → (Rn) | 2 | — |
| STS MACH, Rn | 0000nnnn00001010 | MACH → Rn | 1 | — |
| STS MACL, Rn | 0000nnnn00011010 | MACL → Rn | 1 | — |
| STS PR, Rn | 0000nnnn00101010 | PR → Rn | 1 | — |

Table 2.17 System Control Instructions (cont)

| Instruction | Instruction Code | Operation | Exec. Cycles | T Bit |
|--------------------|-------------------------|--------------------------------|---------------------|--------------|
| STS.L MACH,@-Rn | 0100nnnn00000010 | Rn-4 → Rn, MACH → (Rn) | 1 | — |
| STS.L MACL,@-Rn | 0100nnnn00010010 | Rn-4 → Rn, MACL → (Rn) | 1 | — |
| STS.L PR,@-Rn | 0100nnnn00100010 | Rn-4 → Rn, PR → (Rn) | 1 | — |
| TRAPA #imm | 11000011iiiiiiii | PC/SR → stack area, (imm) → PC | 8 | — |

Note: The number of execution cycles before the chip enters sleep mode: The execution cycles shown in the table are minimums. The actual number of cycles may be increased when (1) contention occurs between instruction fetches and data access, or (2) when the destination register of the load instruction (memory → register) and the register used by the next instruction are the same.

2.5 Processing States

2.5.1 State Transitions

The CPU has four processing states: reset, exception processing, program execution and power-down. Figure 2.6 shows the transitions between the states.

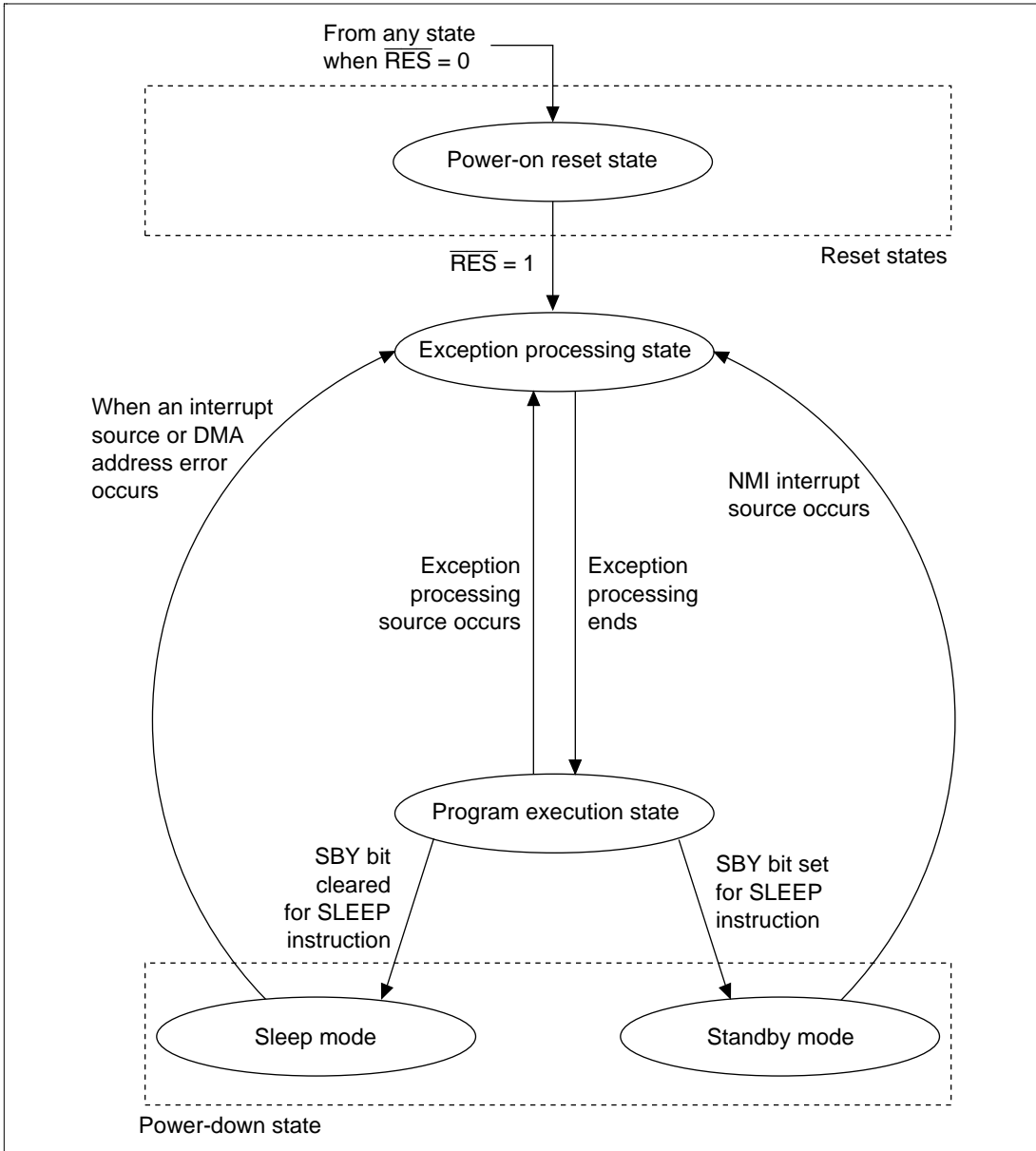


Figure 2.6 Transitions between Processing States

Reset State: The CPU resets in the reset state. When the \overline{RES} pin level goes low, a power-on reset results.

Exception Processing State: The exception processing state is a transient state that occurs when exception processing sources such as resets or interrupts alter the CPU's processing state flow.

For a reset, the initial values of the program counter (PC) (execution start address) and stack pointer (SP) are fetched from the exception processing vector table and stored; the CPU then branches to the execution start address and execution of the program begins.

For an interrupt, the stack pointer (SP) is accessed and the program counter (PC) and status register (SR) are saved to the stack area. The exception service routine start address is fetched from the exception processing vector table; the CPU then branches to that address and the program starts executing, thereby entering the program execution state.

Program Execution State: In the program execution state, the CPU sequentially executes the program.

Power-Down State: In the power-down state, the CPU operation halts and power consumption declines. The SLEEP instruction places the CPU in the power-down state. This state has two modes: sleep mode and standby mode.

2.5.2 Power-Down State

Besides the ordinary program execution states, the CPU also has a power-down state in which CPU operation halts, lowering power consumption. There are two power-down state modes: sleep mode and standby mode.

Sleep Mode: When standby bit SBY (in the standby control register SBYCR) is cleared to 0 and a SLEEP instruction executed, the CPU moves from program execution state to sleep mode. In the sleep mode, the CPU halts and the contents of its internal registers and the data in on-chip cache (or on-chip RAM) is maintained. The on-chip peripheral modules other than the CPU do not halt in the sleep mode.

To return from sleep mode, use a power-on reset, any interrupt, or a DMA address error; the CPU returns to the ordinary program execution state through the exception processing state.

Standby Mode: To enter the standby mode, set the standby bit SBY (in the standby control register SBYCR) to 1 and execute a SLEEP instruction. In standby mode, all CPU, on-chip peripheral module, and oscillator functions are halted. However, when entering standby mode, the DMA master enable bit of the DMAC should be set to 0. If multiplication-related instructions are being executed at the time of entry into standby mode, the values of MACH and MACL will become undefined.

To return from standby mode, use a power-on reset or an NMI interrupt. For resets, the CPU returns to ordinary program execution state through the exception processing state when placed in a reset state for the duration of the oscillator stabilization time. For NMI interrupts, the CPU returns to ordinary program execution state through the exception processing state after the oscillator stabilization time has elapsed. In this mode, power consumption drops markedly, since the oscillator stops (table 2.18).

Table 2.18 Power-Down State

| | | State | | | | | | | |
|----------|--|-------|------|----------------------------|---------------|------------------------------|---------------------------|--|--|
| Mode | Transition Conditions | Clock | CPU | On-Chip Peripheral Modules | CPU Registers | On-Chip Cache or On-Chip RAM | I/O Port Pins | Canceling | |
| Sleep | Execute SLEEP instruction with SBY bit cleared to 0 in SBYCR | Run | Halt | Run | Held | Held | Held | <ul style="list-style-type: none"> • Interrupt • DMA address error • Power-on reset | |
| Stand-by | Execute SLEEP instruction with SBY bit set to 1 in SBYCR | Halt | Halt | Halt and initialize* | Held | Held | Held or Hi-Z (selectable) | <ul style="list-style-type: none"> • NMI interrupt • Power-on reset | |

Note: Differs depending on the peripheral module and pin.

Section 3 Operating Modes

3.1 Operating Modes, Types, and Selection

This LSI has five operating modes and three clock modes, determined by the setting of the mode pins (MD3–MD0). Do not change the mode pin settings during LSI operation (while power is on).

Table 3.1 indicates the setting method for the operating mode.

Table 3.1 Operating Mode Setting

| Mode No. | Pin Setting | | | | Mode Name | On-Chip ROM | CS0 Area |
|-----------------|-------------|-------------------|-------------------|---------|-------------------------------------|-------------|------------------------------|
| | FWP | MD3* ¹ | MD2* ¹ | MD1 MD0 | | | |
| 0 | 1 | x | x | 0 0 | MCU mode 0 | Not Active | 8-bit space |
| 1 | 1 | x | x | 0 1 | MCU mode 1 | Not Active | 16-bit space |
| 2* ⁴ | 1 | x | x | 1 0 | MCU mode 2 | Active | 8/16-bit space* ² |
| 3* ⁴ | 1 | x | x | 1 1 | Single chip mode | Active | — |
| | 0 | x | x | 0 0 | Boot mode* ³ | Active | 8/16-bit space* ² |
| | 0 | x | x | 0 1 | | | — |
| | 0 | x | x | 1 0 | User programming mode* ³ | Active | 8/16-bit space* ² |
| | 0 | x | x | 1 1 | | | — |
| | 1 | 1 | 1 | 0 1 | Flash programmer mode* ³ | Active | — |

Notes: 1. MD2 and MD3 pins select the clock mode in modes 0–3 (table 3.2).

2. Set by BCR2 of BSC.

3. Only F-ZTAT.

4. Only SH7016, SH7017.

Table 3.2 indicates the setting method for the clock mode.

Table 3.2 Clock Mode Setting

| MD3 | MD2 | Clock Mode |
|-----|-----|------------|
| 0 | 0 | PLL ON × 1 |
| 0 | 1 | PLL ON × 2 |
| 1 | 0 | PLL ON × 4 |
| 1 | 1 | Reserved |

3.2 Explanation of Operating Modes

Table 3.3 describes the operating modes.

Table 3.3 Operating Modes

| Mode | Description |
|---------------------------|---|
| (MCU) Mode 0 | CS0 area becomes an external memory space with 8-bit bus width. |
| (MCU) Mode 1 | CS0 area becomes an external memory space with 16-bit bus width. |
| (MCU) Mode 2 | The on-chip ROM becomes effective. The bus width for the on-chip ROM space is 32 bit. |
| Mode 3 (single chip mode) | Any port can be used, but external addresses can not be employed. |
| Clock mode | The input waveform frequency can be used as is, doubled or quadrupled as an internal clock in modes 0 to 3. |

3.3 Pin Configuration

Table 3.4 describes the function of each operating mode related pin.

Table 3.4 Operating Mode Pin Function

| Pin Name | Input/Output | Function |
|----------|--------------|--|
| XTAL | Input | Connects to a crystal oscillator |
| EXTAL | Input | Connects to a crystal oscillator, or used for external clock input pin |
| PLLCAP | Input | Connects to a capacitor for PLL circuit operation |
| MD0 | Input | Designates operating mode through the level applied to this pin |
| MD1 | Input | Designates operating mode through the level applied to this pin |
| MD2 | Input | Designates clock mode through the level applied to this pin |
| MD3 | Input | Designates clock mode through the level applied to this pin |

Section 4 Clock Pulse Generator (CPG)

4.1 Overview

This LSI has an on-chip clock pulse generator (CPG) that generates the system clock (ϕ), as well as the internal clock ($\phi/2$ to $\phi/8192$). The CPG consists of an oscillator, a PLL, and a prescaler.

4.1.1 Block Diagram

A block diagram of the clock pulse generator is shown in figure 4.1.

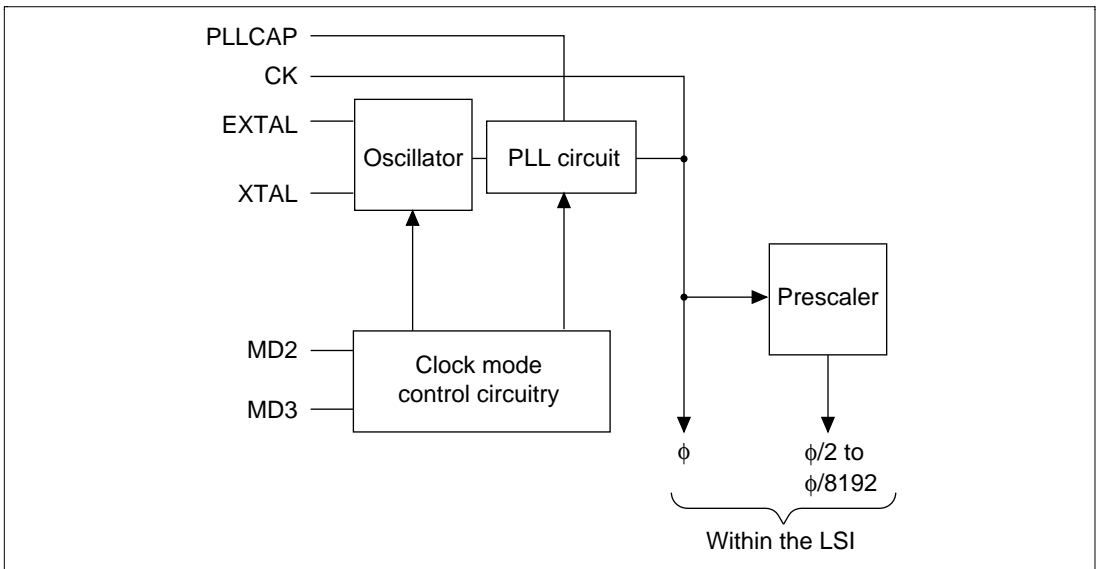


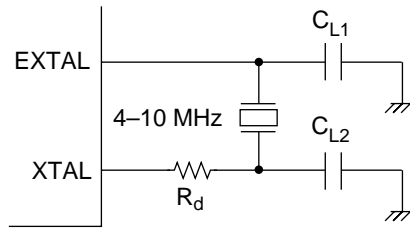
Figure 4.1 Block Diagram of the Clock Pulse Generator

4.2 Oscillator

Clock pulses can be supplied from a connected crystal resonator or an external clock.

4.2.1 Connecting a Crystal Oscillator

Circuit Configuration: A crystal oscillator can be connected as shown in figure 4.2. Use the damping resistance (R_d) listed in table 4.1. Use a 4–10 MHz crystal oscillator (consult your dealer concerning the compatibility of the crystal oscillator and the LSI).



$$C_{L1} = C_{L2} = 18\text{--}22 \text{ pF (Recommended value)}$$

Figure 4.2 Connection of the Crystal Oscillator (Example)

Table 4.1 Damping Resistance Values (Recommended Values)

| Parameter | Frequency (MHz) | | |
|--------------------|-----------------|-----|----|
| | 4 | 8 | 10 |
| R _d (Ω) | 500 | 200 | 0 |

Crystal Oscillator: Figure 4.3 shows an equivalent circuit of the crystal oscillator. Use a crystal oscillator with the characteristics listed in table 4.2.

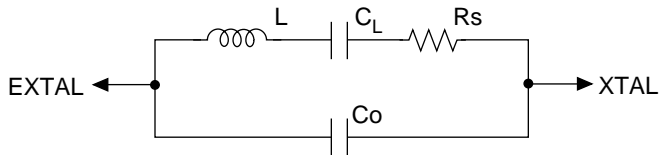


Figure 4.3 Crystal Oscillator Equivalent Circuit

Table 4.2 Crystal Oscillator Parameters

| Parameter | Frequency (MHz) | | |
|-------------------------|-----------------|----|----|
| | 4 | 8 | 10 |
| R _s max (Ω) | 120 | 80 | 60 |
| C _o max (pF) | 7 | 7 | 7 |

Notes on Board Design: When connecting a crystal oscillator, observe the following precautions:

- To prevent induction from interfering with correct oscillation, do not route any signal lines near the oscillator circuitry.
- When designing the board, place the crystal oscillator and its load capacitors as close as possible to the XTAL and EXTAL pins.

Figures 4.4 show the precautions regarding oscillator block board settings.

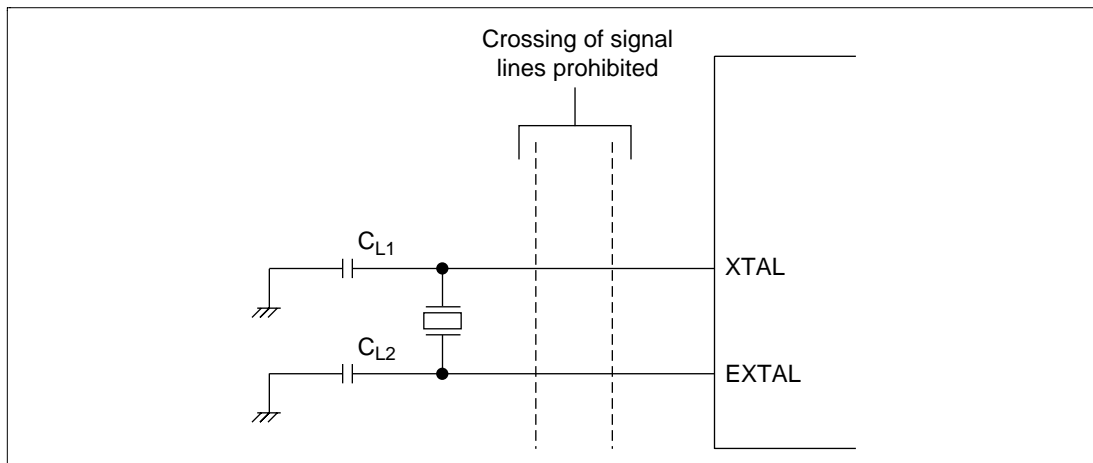


Figure 4.4 Cautions for Oscillator Circuit System Board Design

External circuitry such as that shown in figure 4.5 is recommended around the PLL.

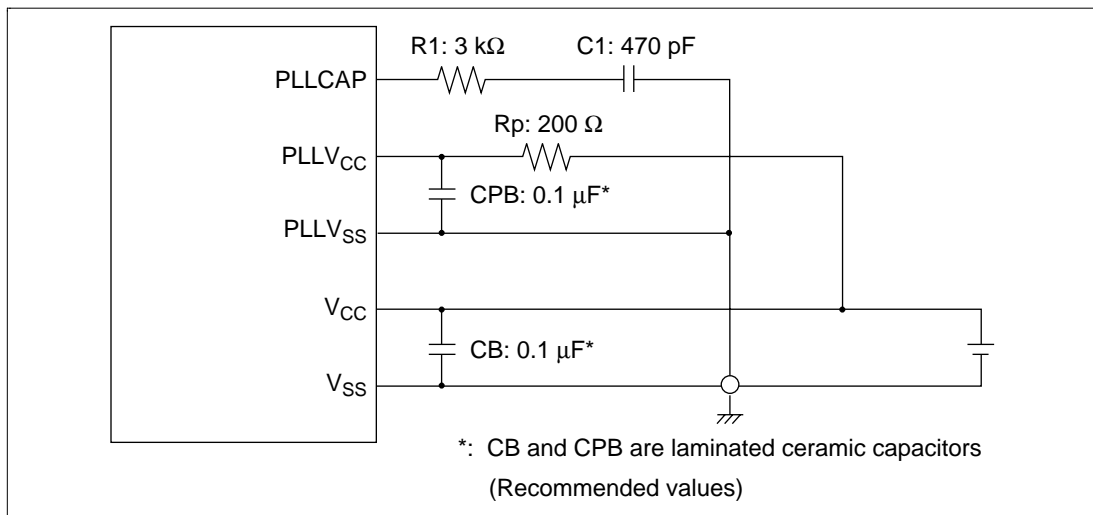


Figure 4.5 Cautions for Use of PLL Oscillator Circuit

Place oscillation stabilization capacitor C1 and resistor R1 near the PLL CAP pin, and ensure that these lines do not cross any other signal lines. Supply the C1 ground from PLL V_{SS}.

Also, separate PLL V_{CC} and PLL V_{SS}, and the other V_{CC} and V_{SS} pins, from the board power supply source, and be sure to insert bypass capacitors CPB and CB close to the pins.

4.2.2 External Clock Input Method

Figure 4.6 shows an example of an external clock input connection. In this case, make the external clock high level to stop it when in standby mode. During operation, make the external input clock frequency 4–10 MHz.

When leaving the XTAL pin open, make sure the parasitic capacitance is less than 10 pF.

Even when inputting an external clock, be sure to delay until after the oscillation stabilization time (upon power-on) or after release from standby, in order to ensure the PLL stabilization time.

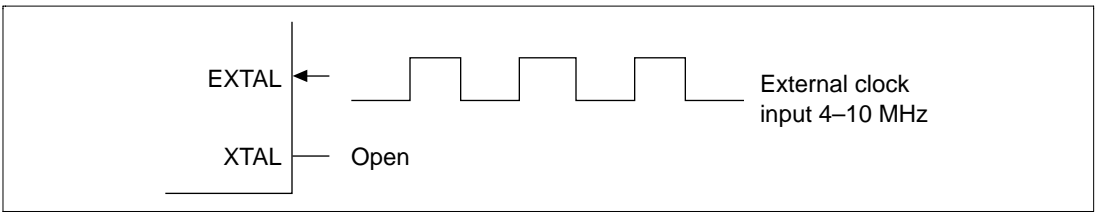


Figure 4.6 Example of External Clock Connection

4.2.3 Prescaler

The prescaler divides the system clock (ϕ) to generate an internal clock ($\phi/2$ to $\phi/8192$) for supply to peripheral modules.

Section 5 Exception Processing

5.1 Overview

5.1.1 Types of Exception Processing and Priority

Exception processing is started by four sources: resets, address errors, interrupts and instructions and have the priority shown in table 5.1. When several exception processing sources occur at once, they are processed according to the priority shown.

Table 5.1 Types of Exception Processing and Priority Order

| Exception | Source | Priority | |
|---------------|--|--|--|
| Reset | Power-on reset | High | |
| Address error | CPU address error | | |
| | DMAC address error | | |
| Interrupt | NMI | | |
| | User break | | |
| | IRQ | | |
| | On-chip peripheral modules: | <ul style="list-style-type: none">• Direct memory access controller (DMAC)• Multifunction timer/pulse unit (MTU)• Serial communication interface (SCI)• A/D converter (A/D)• Compare match timer (CMT)• Watchdog timer (WDT)• Bus state controller (BSC) | |
| | | | |
| Instructions | Trap instruction (TRAPA instruction) | | |
| | General illegal instructions (undefined code) | | |
| | Illegal slot instructions (undefined code placed directly after a delay branch instruction* ¹ or instructions that rewrite the PC* ²) | Low | |

- Notes:
1. Delayed branch instructions: JMP, JSR, BRA, BSR, RTS, RTE, BF/S, BT/S, BSRF, BRAF.
 2. Instructions that rewrite the PC: JMP, JSR, BRA, BSR, RTS, RTE, BT, BF, TRAPA, BF/S, BT/S, BSRF, BRAF.

5.1.2 Exception Processing Operations

The exception processing sources are detected and begin processing according to the timing shown in table 5.2.

Table 5.2 Timing of Exception Source Detection and the Start of Exception Processing

| Exception | Source | Timing of Source Detection and Start of Processing |
|----------------|------------------------------|--|
| Power-on reset | | Starts when the $\overline{\text{RES}}$ pin changes from low to high. |
| Address error | | Detected when instruction is decoded and starts when the previous executing instruction finishes executing. |
| Interrupts | | Detected when instruction is decoded and starts when the previous executing instruction finishes executing. |
| Instructions | Trap instruction | Starts from the execution of a TRAPA instruction. |
| | General illegal instructions | Starts from the decoding of undefined code anytime except after a delayed branch instruction (delay slot). |
| | Illegal slot instructions | Starts from the decoding of undefined code placed in a delayed branch instruction (delay slot) or of instructions that rewrite the PC. |

When exception processing starts, the CPU operates as follows:

1. Exception processing triggered by reset:

The initial values of the program counter (PC) and stack pointer (SP) are fetched from the exception processing vector table (PC and SP are respectively the H'00000000 and H'00000004 addresses). See section 5.1.3, Exception Processing Vector Table, for more information. 0 is then written to the vector base register (VBR) and 1111 is written to the interrupt mask bits (I3–I0) of the status register (SR). The program begins running from the PC address fetched from the exception processing vector table.

2. Exception processing triggered by address errors, interrupts and instructions:

SR and PC are saved to the stack indicated by R15. For interrupt exception processing, the interrupt priority level is written to the SR's interrupt mask bits (I3–I0). For address error and instruction exception processing, the I3–I0 bits are not affected. The start address is then fetched from the exception processing vector table and the program begins running from that address.

5.1.3 Exception Processing Vector Table

Before exception processing begins running, the exception processing vector table must be set in memory. The exception processing vector table stores the start addresses of exception service routines. (The reset exception processing table holds the initial values of PC and SP.)

All exception sources are given different vector numbers and vector table address offsets, from which the vector table addresses are calculated. During exception processing, the start addresses of the exception service routines are fetched from the exception processing vector table, which indicated by this vector table address.

Table 5.3 shows the vector numbers and vector table address offsets. Table 5.4 shows how vector table addresses are calculated.

Table 5.3 Exception Processing Vector Table

| Exception Sources | | Vector Numbers | Vector Table Address Offset |
|--------------------------------|------------|----------------|-----------------------------|
| Power-on reset | PC | 0 | H'00000000–H'00000003 |
| | SP | 1 | H'00000004–H'00000007 |
| (Reserved by system) | | 2 | H'00000008–H'0000000B |
| (Reserved by system) | | 3 | H'0000000C–H'0000000F |
| General illegal instruction | | 4 | H'00000010–H'00000013 |
| (Reserved by system) | | 5 | H'00000014–H'00000017 |
| Slot illegal instruction | | 6 | H'00000018–H'0000001B |
| (Reserved by system) | | 7 | H'0000001C–H'0000001F |
| (Reserved by system) | | 8 | H'00000020–H'00000023 |
| CPU address error | | 9 | H'00000024–H'00000027 |
| DMAC address error | | 10 | H'00000028–H'0000002B |
| Interrupts | NMI | 11 | H'0000002C–H'0000002F |
| | User break | 12 | H'00000030–H'00000033 |
| (Reserved by system) | | 13 | H'00000034–H'00000037 |
| | | : | : |
| | | 31 | H'0000007C–H'0000007F |
| Trap instruction (user vector) | | 32 | H'00000080–H'00000083 |
| | | : | : |
| | | 63 | H'000000FC–H'000000FF |

Table 5.3 Exception Processing Vector Table (cont)

| Exception Sources | Vector Numbers | Vector Table Address Offset | |
|----------------------------|-----------------------|------------------------------------|-----------------------|
| Interrupts | IRQ0 | 64 | H'00000100–H'00000103 |
| | IRQ1 | 65 | H'00000104–H'00000107 |
| | IRQ2 | 66 | H'00000108–H'0000010B |
| | IRQ3 | 67 | H'0000010C–H'0000010F |
| (Reserved by system) | 68 | H'00000110–H'00000113 | |
| (Reserved by system) | 69 | H'00000114–H'00000117 | |
| Interrupt | IRQ6 | 70 | H'00000118–H'0000011B |
| | IRQ7 | 71 | H'0000011C–H'0000011F |
| On-chip peripheral module* | 72 | | H'00000120–H'00000124 |
| | : | | : |
| | 255 | | H'000003FC–H'000003FF |

Note: The vector numbers and vector table address offsets for each on-chip peripheral module interrupt are given in section 6, Interrupt Controller, and table 6.3, Interrupt Exception Processing Vectors and Priorities.

Table 5.4 Calculating Exception Processing Vector Table Addresses

| Exception Source | Vector Table Address Calculation |
|--|---|
| Resets | Vector table address = (vector table address offset) = (vector number) × 4 |
| Address errors, interrupts, instructions | Vector table address = VBR + (vector table address offset) = VBR + (vector number) × 4 |

- Notes: 1. VBR: Vector base register
2. Vector table address offset: See table 5.3.
3. Vector number: See table 5.3.

5.2 Resets

5.2.1 Power-on Reset

When the $\overline{\text{RES}}$ pin is driven low, the LSI does a power-on reset. To reliably reset the LSI, the $\overline{\text{RES}}$ pin should be kept at low for at least the duration of the oscillation settling time when applying power or when in standby mode (when the clock circuit is halted) or at least $20 t_{\text{cyc}}$ (when the clock circuit is running). During power-on reset, CPU internal status and all registers of on-chip peripheral modules are initialized. See Appendix B, Pin Status, for the status of individual pins during the power-on reset status.

In the power-on reset status, power-on reset exception processing starts when the $\overline{\text{RES}}$ pin is first driven low for a set period of time and then returned to high. The CPU will then operate as follows:

1. The initial value (execution start address) of the program counter (PC) is fetched from the exception processing vector table.
2. The initial value of the stack pointer (SP) is fetched from the exception processing vector table.
3. The vector base register (VBR) is cleared to H'00000000 and the interrupt mask bits (I3–I0) of the status register (SR) are set to H'F (1111).
4. The values fetched from the exception processing vector table are set in the program counter (PC) and SP and the program begins executing.

Be certain to always perform power-on reset processing when turning the system power on.

5.3 Address Errors

Address errors occur when instructions are fetched or data read or written, as shown in table 5.5.

Table 5.5 Bus Cycles and Address Errors

| Bus Cycle | | | |
|--|----------------------|--|-----------------------|
| Type | Bus Master | Bus Cycle Description | Address Errors |
| Instruction fetch | CPU | Instruction fetched from even address | None (normal) |
| | | Instruction fetched from odd address | Address error occurs |
| | | Instruction fetched from other than on-chip peripheral module space* | None (normal) |
| | | Instruction fetched from on-chip peripheral module space* | Address error occurs |
| Data read/write | CPU or DMAC | Word data accessed from even address | None (normal) |
| | | Word data accessed from odd address | Address error occurs |
| | | Longword data accessed from a longword boundary | None (normal) |
| | | Longword data accessed from other than a long-word boundary | Address error occurs |
| | | Byte or word data accessed in on-chip peripheral module space* | None (normal) |
| | | Longword data accessed in 16-bit on-chip peripheral module space* | None (normal) |
| | | Longword data accessed in 8-bit on-chip peripheral module space* | Address error occurs |
| External memory space accessed in single-chip mode | Address error occurs | | |

Note: See section 8, Bus State Controller.

5.3.1 Address Error Exception Processing

When an address error occurs, the bus cycle in which the address error occurred ends. When the executing instruction then finishes, address error exception processing starts up. The CPU operates as follows:

1. The status register (SR) is saved to the stack.
2. The program counter (PC) is saved to the stack. The PC value saved is the start address of the instruction to be executed after the last executed instruction.
3. The exception service routine start address is fetched from the exception processing vector table that corresponds to the address error that occurred and the program starts executing from that address. The jump that occurs is not a delayed branch.

5.4 Interrupts

Table 5.6 shows the sources that start up interrupt exception processing. These are divided into NMI, user breaks, IRQ and on-chip peripheral modules.

Table 5.6 Interrupt Sources

| Type | Request Source | Number of Sources |
|---------------------------|--|-------------------|
| NMI | NMI pin (external input) | 1 |
| IRQ | $\overline{\text{IRQ0}}$ – $\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$ (external input) | 6 |
| On-chip peripheral module | Direct memory access controller (DMAC) | 2 |
| | Multifunction timer pulse unit (MTU) | 13 |
| | Serial communication interface (SCI) | 8 |
| | A/D converter | 1 |
| | Compare match timer (CMT) | 2 |
| | Watchdog timer (WDT) | 1 |
| | Bus state controller (BSC) | 1 |

Each interrupt source is allocated a different vector number and vector table offset. See section 6, Interrupt Controller, and table 6.3, Interrupt Exception Processing Vectors and Priorities, for more information on vector numbers and vector table address offsets.

5.4.1 Interrupt Priority Level

The interrupt priority order is predetermined. When multiple interrupts occur simultaneously (overlap), the interrupt controller (INTC) determines their relative priorities and starts up processing according to the results.

The priority order of interrupts is expressed as priority levels 0–16, with priority 0 the lowest and priority 16 the highest. The NMI interrupt has priority 16 and cannot be masked, so it is always accepted. IRQ interrupts and on-chip peripheral module interrupt priority levels can be set freely using the INTC's interrupt priority level setting registers A through H (IPRA to IPRH) as shown in table 5.7. The priority levels that can be set are 0–15. Level 16 cannot be set. See section 6.3.1, Interrupt Priority Registers A-H (IPRA-IPRH), for more information on IPRA to IPRH.

Table 5.7 Interrupt Priority Order

| Type | Priority Level | Comment |
|---------------------------|-----------------------|---|
| NMI | 16 | Fixed priority level. Cannot be masked. |
| IRQ | 0–15 | Set with interrupt priority level setting registers A through H (IPRA to IPRH). |
| On-chip peripheral module | 0–15 | Set with interrupt priority level setting registers A through H (IPRA to IPRH). |

5.4.2 Interrupt Exception Processing

When an interrupt occurs, its priority level is ascertained by the interrupt controller (INTC). NMI is always accepted, but other interrupts are only accepted if they have a priority level higher than the priority level set in the interrupt mask bits (I3–I0) of the status register (SR).

When an interrupt is accepted, exception processing begins. In interrupt exception processing, the CPU saves SR and the program counter (PC) to the stack. The priority level value of the accepted interrupt is written to SR bits I3–I0. For NMI, however, the priority level is 16, but the value set in I3–I0 is H'F (level 15). Next, the start address of the exception service routine is fetched from the exception processing vector table for the accepted interrupt, that address is jumped to and execution begins. See section 6.4, Interrupt Operation, for more information on the interrupt exception processing.

5.5 Exceptions Triggered by Instructions

Exception processing can be triggered by trap instructions, general illegal instructions, and illegal slot instructions, as shown in table 5.8.

Table 5.8 Types of Exceptions Triggered by Instructions

| Type | Source Instruction | Comment |
|------------------------------|--|--|
| Trap instructions | TRAPA | — |
| Illegal slot instructions | Undefined code placed immediately after a delayed branch instruction (delay slot) and instructions that rewrite the PC | Delayed branch instructions: JMP, JSR, BRA, BSR, RTS, RTE, BF/S, BT/S, BSRF, BRAF Instructions that rewrite the PC: JMP, JSR, BRA, BSR, RTS, RTE, BT, BF, TRAPA, BF/S, BT/S, BSRF, BRAF |
| General illegal instructions | Undefined code anywhere besides in a delay slot | — |

5.5.1 Trap Instructions

When a TRAPA instruction is executed, trap instruction exception processing starts up. The CPU operates as follows:

1. The status register (SR) is saved to the stack.
2. The program counter (PC) is saved to the stack. The PC value saved is the start address of the instruction to be executed after the TRAPA instruction.
3. The exception service routine start address is fetched from the exception processing vector table that corresponds to the vector number specified in the TRAPA instruction. That address is jumped to and the program starts executing. The jump that occurs is not a delayed branch.

5.5.2 Illegal Slot Instructions

An instruction placed immediately after a delayed branch instruction is said to be placed in a delay slot. When the instruction placed in the delay slot is undefined code, illegal slot exception processing starts up when that undefined code is decoded. Illegal slot exception processing also starts up when an instruction that rewrites the program counter (PC) is placed in a delay slot. The processing starts when the instruction is decoded. The CPU handles an illegal slot instruction as follows:

1. The status register (SR) is saved to the stack.
2. The program counter (PC) is saved to the stack. The PC value saved is the jump address of the delayed branch instruction immediately before the undefined code or the instruction that rewrites the PC.
3. The exception service routine start address is fetched from the exception processing vector table that corresponds to the exception that occurred. That address is jumped to and the program starts executing. The jump that occurs is not a delayed branch.

5.5.3 General Illegal Instructions

When undefined code placed anywhere other than immediately after a delayed branch instruction (i.e., in a delay slot) is decoded, general illegal instruction exception processing starts up. The CPU handles general illegal instructions the same as illegal slot instructions. Unlike processing of illegal slot instructions, however, the program counter value stored is the start address of the undefined code.

5.6 When Exception Sources Are Not Accepted

When an address error or interrupt is generated after a delayed branch instruction or interrupt-disabled instruction, it is sometimes not accepted immediately but stored instead, as shown in table 5.9. When this happens, it will be accepted when an instruction that can accept the exception is decoded.

Table 5.9 Generation of Exception Sources Immediately after a Delayed Branch Instruction or Interrupt-Disabled Instruction

| Point of Occurrence | Exception Source | |
|---|------------------|--------------|
| | Address Error | Interrupt |
| Immediately after a delayed branch instruction* ¹ | Not accepted | Not accepted |
| Immediately after an interrupt-disabled instruction* ² | Accepted | Not accepted |

Notes: 1. Delayed branch instructions: JMP, JSR, BRA, BSR, RTS, RTE, BF/S, BT/S, BSRF, BRAF

2. Interrupt-disabled instructions: LDC, LDC.L, STC, STC.L, LDS, LDS.L, STS, STS.L

5.6.1 Immediately after a Delayed Branch Instruction

When an instruction placed immediately after a delayed branch instruction (delay slot) is decoded, neither address errors nor interrupts are accepted. The delayed branch instruction and the instruction located immediately after it (delay slot) are always executed consecutively, so no exception processing occurs during this period.

5.6.2 Immediately after an Interrupt-Disabled Instruction

When an instruction immediately following an interrupt-disabled instruction is decoded, interrupts are not accepted. Address errors are accepted.

5.7 Stack Status after Exception Processing Ends

The status of the stack after exception processing ends is as shown in table 5.10.

Table 5.10 Types of Stack Status After Exception Processing Ends

| Types | Stack Status |
|-----------------------------|---|
| Address error | <p>SP → Address of instruction after executed instruction 32 bits SR 32 bits</p> |
| Trap instruction | <p>SP → Address of instruction after TRAPA instruction 32 bits SR 32 bits</p> |
| General illegal instruction | <p>SP → Start address of illegal instruction 32 bits SR 32 bits</p> |
| Interrupt | <p>SP → Address of instruction after executed instruction 32 bits SR 32 bits</p> |
| Illegal slot instruction | <p>SP → Jump destination address of delay branch instruction 32 bits SR 32 bits</p> |

5.8 Notes on Use

5.8.1 Value of Stack Pointer (SP)

The value of the stack pointer must always be a multiple of four. If it is not, an address error will occur when the stack is accessed during exception processing.

5.8.2 Value of Vector Base Register (VBR)

The value of the vector base register must always be a multiple of four. If it is not, an address error will occur when the stack is accessed during exception processing.

5.8.3 Address Errors Caused by Stacking of Address Error Exception Processing

When the stack pointer is not a multiple of four, an address error will occur during stacking of the exception processing (interrupts, etc.) and address error exception processing will start up as soon as the first exception processing is ended. Address errors will then also occur in the stacking for this address error exception processing. To ensure that address error exception processing does not go into an endless loop, no address errors are accepted at that point. This allows program control to be shifted to the address error exception service routine and enables error processing.

When an address error occurs during exception processing stacking, the stacking bus cycle (write) is executed. During stacking of the status register (SR) and program counter (PC), the SP is -4 for both, so the value of SP will not be a multiple of four after the stacking either. The address value output during stacking is the SP value, so the address where the error occurred is itself output. This means the write data stacked will be undefined.

Section 6 Interrupt Controller (INTC)

6.1 Overview

The interrupt controller (INTC) ascertains the priority of interrupt sources and controls interrupt requests to the CPU. The INTC has registers for setting the priority of each interrupt which can be used by the user to order the priorities in which the interrupt requests are processed.

6.1.1 Features

The INTC has the following features:

- 16 levels of interrupt priority: By setting the eight interrupt-priority level registers, the priorities of IRQ interrupts and on-chip peripheral module interrupts can be set in 16 levels for different request sources.
- NMI noise canceler function: NMI input level bits indicate the NMI pin status. By reading these bits with the interrupt exception service routine, the pin status can be confirmed, enabling it to be used as a noise canceler.

6.1.2 Block Diagram

Figure 6.1 is a block diagram of the INTC.

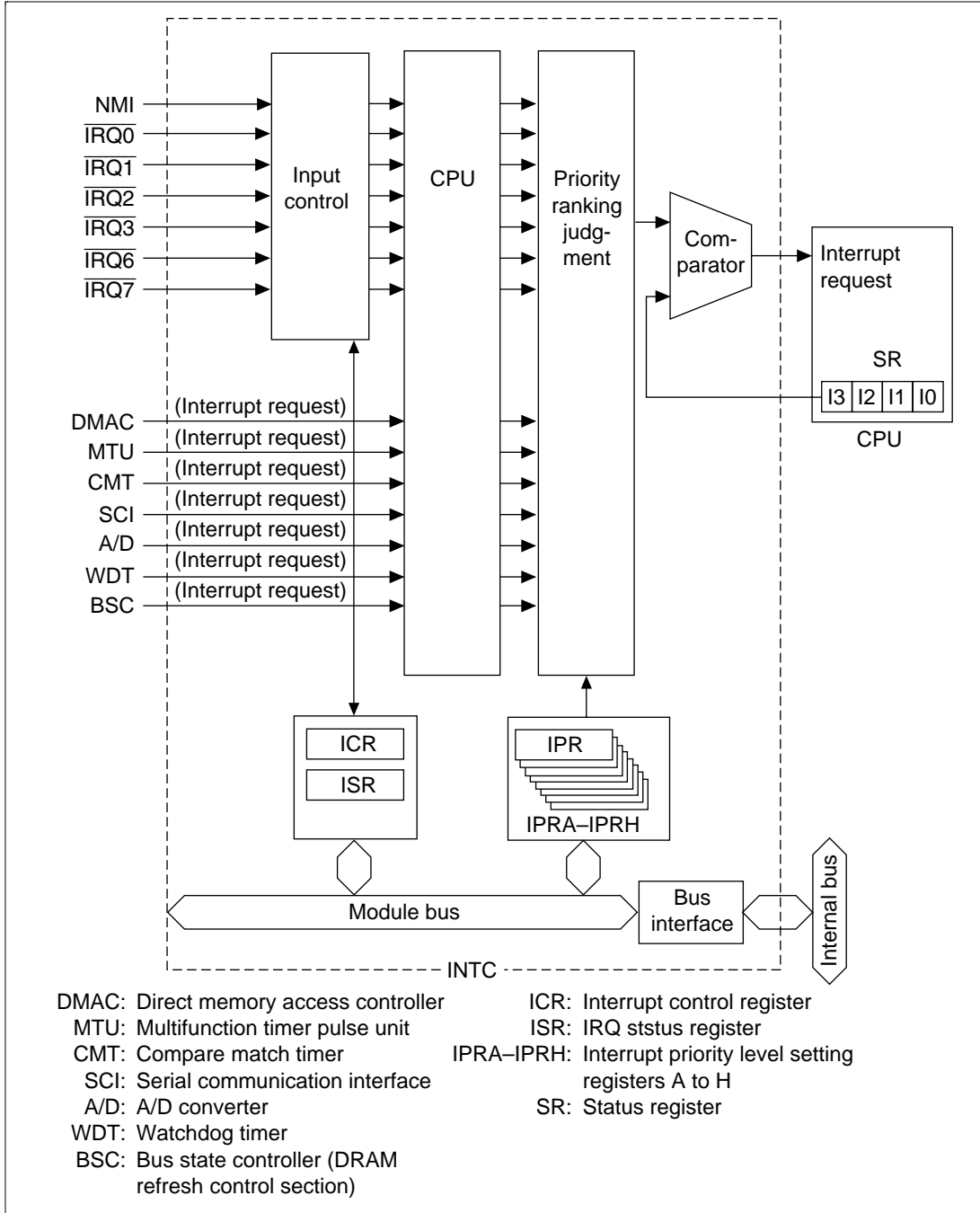


Figure 6.1 INTC Block Diagram

6.1.3 Pin Configuration

Table 6.1 shows the INTC pin configuration.

Table 6.1 Pin Configuration

| Name | Abbreviation | I/O | Function |
|----------------------------------|--|-----|--|
| Non-maskable interrupt input pin | NMI | I | Input of non-maskable interrupt request signal |
| Interrupt request input pins | $\overline{\text{IRQ0}}\text{--}\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$ | I | Input of maskable interrupt request signals |

6.1.4 Register Configuration

The INTC has the 10 registers shown in table 6.2. These registers set the priority of the interrupts and control external interrupt input signal detection.

Table 6.2 Register Configuration

| Name | Abbr. | R/W | Initial Value | Address | Access Sizes |
|-------------------------------|-------|--------|---------------|------------|--------------|
| Interrupt priority register A | IPRA | R/W | H'0000 | H'FFFF8348 | 8, 16, 32 |
| Interrupt priority register B | IPRB | R/W | H'0000 | H'FFFF834A | 8, 16, 32 |
| Interrupt priority register C | IPRC | R/W | H'0000 | H'FFFF834C | 8, 16, 32 |
| Interrupt priority register D | IPRD | R/W | H'0000 | H'FFFF834E | 8, 16, 32 |
| Interrupt priority register E | IPRE | R/W | H'0000 | H'FFFF8350 | 8, 16, 32 |
| Interrupt priority register F | IPRF | R/W | H'0000 | H'FFFF8352 | 8, 16, 32 |
| Interrupt priority register G | IPRG | R/W | H'0000 | H'FFFF8354 | 8, 16, 32 |
| Interrupt priority register H | IPRH | R/W | H'0000 | H'FFFF8356 | 8, 16, 32 |
| Interrupt control register | ICR | R/W | *1 | H'FFFF8358 | 8, 16, 32 |
| IRQ status register | ISR | R(W)*2 | H'0000 | H'FFFF835A | 8, 16, 32 |

Notes: 1. The value when the NMI pin is high is H'8000; when the NMI pin is low, it is H'0000.
2. Only 0 can be written, in order to clear flags.

6.2 Interrupt Sources

There are three types of interrupt sources: NMI, IRQ, and on-chip peripheral modules. Each interrupt has a priority expressed as a priority level (0 to 16, with 0 the lowest and 16 the highest). Giving an interrupt a priority level of 0 masks it.

6.2.1 NMI Interrupts

The NMI interrupt has priority 16 and is always accepted. Input at the NMI pin is detected by edge. Use the NMI edge select bit (NMIE) in the interrupt control register (ICR) to select either the rising or falling edge. NMI interrupt exception processing sets the interrupt mask level bits (I3–I0) in the status register (SR) to level 15.

6.2.2 IRQ Interrupts

IRQ interrupts are requested by input from pins $\overline{\text{IRQ0}}$ – $\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$. Set the IRQ sense select bits (IRQ0S–IRQ3S, IRQ6S, IRQ7S) of the interrupt control register (ICR) to select low level detection or falling edge detection for each pin. The priority level can be set from 0 to 15 for each pin using the interrupt priority registers A and B (IPRA–IPRB).

When IRQ interrupts are set to low level detection, an interrupt request signal is sent to the INTC during the period the IRQ pin is low level. Interrupt request signals are not sent to the INTC when the IRQ pin becomes high level. Interrupt request levels can be confirmed by reading the IRQ flags (IRQ0F–IRQ3F, IRQ6F, IRQ7F) of the IRQ status register (ISR).

When IRQ interrupts are set to falling edge detection, interrupt request signals are sent to the INTC upon detecting a change on the IRQ pin from high to low level. IRQ interrupt request detection results are maintained until the interrupt request is accepted. Confirmation that IRQ interrupt requests have been detected is possible by reading the IRQ flags (IRQ0F–IRQ3F, IRQ6F, IRQ7F) of the IRQ status register (ISR), and by writing a 0 after reading a 1, IRQ interrupt request detection results can be withdrawn.

In IRQ interrupt exception processing, the interrupt mask bits (I3–I0) of the status register (SR) are set to the priority level value of the accepted IRQ interrupt.

6.2.3 On-Chip Peripheral Module Interrupts

On-chip peripheral module interrupts are interrupts generated by the following on-chip peripheral modules:

- Direct memory access controller (DMAC)
- Multifunction timer pulse unit (MTU)
- Compare match timer (CMT)
- Serial communication interface (SCI)
- A/D converter (A/D)
- Watchdog timer (WDT)
- Bus state controller (BSC)

A different interrupt vector is assigned to each interrupt source, so the exception service routine does not have to decide which interrupt has occurred. Priority levels between 0 and 15 can be assigned to individual on-chip peripheral modules in interrupt priority registers C–H (IPRC–IPRH).

On-chip peripheral module interrupt exception processing sets the interrupt mask level bits (I3–I0) in the status register (SR) to the priority level value of the on-chip peripheral module interrupt that was accepted.

6.2.4 Interrupt Exception Vectors and Priority Rankings

Table 6.3 lists interrupt sources and their vector numbers, vector table address offsets and interrupt priorities.

Each interrupt source is allocated a different vector number and vector table address offset. Vector table addresses are calculated from vector numbers and address offsets. In interrupt exception processing, the exception service routine start address is fetched from the vector table indicated by the vector table address. See table 5.4, Calculating Exception Processing Vector Table Addresses.

IRQ interrupts and on-chip peripheral module interrupt priorities can be set freely between 0 and 15 for each pin or module by setting interrupt priority registers A–H (IPRA–IPRH). The ranking of interrupt sources for IPRC–IPRH, however, must be the order listed under Priority Order Within IPR Setting Range in table 6.3 and cannot be changed. A power-on reset assigns priority level 0 to IRQ interrupts and on-chip peripheral module interrupts. If the same priority level is assigned to two or more interrupt sources and interrupts from those sources occur simultaneously, their priority order is the default priority order indicated at the right in table 6.3.

Table 6.3 Interrupt Exception Processing Vectors and Priorities

| Interrupt Source | Interrupt Vector | | Interrupt Priority (Initial Value) | Corresponding IPR (Bits) | Priority within IPR Setting Range | Default Priority |
|------------------|------------------|-----------------------------|------------------------------------|--------------------------|-----------------------------------|------------------|
| | Vector No. | Vector Table Address Offset | | | | |
| NMI | 11 | H'0000002C to H'0000002F | 16 | — | — | High |
| IRQ0 | 64 | H'00000100 to H'00000103 | 0 to 15 (0) | IPRA (15–12) | — | |
| IRQ1 | 65 | H'00000104 to H'00000107 | 0 to 15 (0) | IPRA (11–8) | — | |
| IRQ2 | 66 | H'00000108 to H'0000010B | 0 to 15 (0) | IPRA (7–4) | — | |
| IRQ3 | 67 | H'0000010C to H'0000010F | 0 to 15 (0) | IPRA (3–0) | — | |
| IRQ6 | 70 | H'00000118 to H'0000011B | 0 to 15 (0) | IPRB (7–4) | — | |
| IRQ7 | 71 | H'0000011C to H'0000011F | 0 to 15 (0) | IPRB (3–0) | — | |
| DMAC0 | DEI0 | 72 | H'00000120 to H'00000123 | 0 to 15 (0) | IPRC (15–12) | — |
| DMAC1 | DEI1 | 76 | H'00000130 to H'00000133 | 0 to 15 (0) | IPRC (11–8) | — |
| MTU0 | TGI0A | 88 | H'00000160 to H'00000163 | 0 to 15 (0) | IPRD (15–12) | High |
| | TGI0B | 89 | H'00000164 to H'00000167 | 0 to 15 (0) | | |
| | TGI0C | 90 | H'00000168 to H'0000016B | 0 to 15 (0) | | |
| | TGI0D | 91 | H'0000016C to H'0000016F | 0 to 15 (0) | Low | |
| | TCI0V | 92 | H'00000170 to H'00000173 | 0 to 15 (0) | IPRD (11–8) | — |

Table 6.3 Interrupt Exception Processing Vectors and Priorities (cont)

| Interrupt Source | | Interrupt Vector | | Interrupt Priority (Initial Value) | Corresponding IPR (Bits) | Priority within IPR Setting Range | Default Priority |
|------------------|-------|------------------|-----------------------------|------------------------------------|--------------------------|-----------------------------------|------------------|
| | | Vector No. | Vector Table Address Offset | | | | |
| MTU1 | TGI1A | 96 | H'00000180 to H'00000183 | 0 to 15 (0) | IPRD (7–4) | High | High |
| | TGI1B | 97 | H'00000184 to H'00000187 | | | 0 to 15 (0) | Low |
| | TCI1V | 100 | H'00000190 to H'00000193 | 0 to 15 (0) | IPRD (3–0) | High | |
| | TCI1U | 101 | H'00000194 to H'00000197 | 0 to 15 (0) | | Low | |
| MTU2 | TGI2A | 104 | H'000001A0 to H'000001A3 | 0 to 15 (0) | IPRE (15–12) | High | |
| | TGI2B | 105 | H'000001A4 to H'000001A7 | | | 0 to 15 (0) | Low |
| | TCI2V | 108 | H'000001B0 to H'000001B3 | 0 to 15 (0) | IPRE (11–8) | High | |
| | TCI2U | 109 | H'000001B4 to H'000001B7 | 0 to 15 (0) | | Low | |
| SCI0 | ERI0 | 128 | H'00000200 to H'00000203 | 0 to 15 (0) | IPRF (7–4) | High | |
| | RXI0 | 129 | H'00000204 to H'00000207 | | | 0 to 15 (0) | |
| | TXI0 | 130 | H'00000208 to H'0000020B | 0 to 15 (0) | | | |
| | TEI0 | 131 | H'0000020C to H'0000020F | 0 to 15 (0) | | Low | |
| SCI1 | ERI1 | 132 | H'00000210 to H'00000213 | 0 to 15 (0) | IPRF (3–0) | High | |
| | RXI1 | 133 | H'00000214 to H'00000217 | | | 0 to 15 (0) | |
| | TXI1 | 134 | H'00000218 to H'0000021B | 0 to 15 (0) | | | |
| | TEI1 | 135 | H'0000021C to H'0000021F | 0 to 15 (0) | | Low | Low |

Table 6.3 Interrupt Exception Processing Vectors and Priorities (cont)

| Interrupt Source | | Interrupt Vector | | Interrupt Priority (Initial Value) | Corresponding IPR (Bits) | Priority within IPR Setting Range | Default Priority |
|------------------|------|------------------|-----------------------------|------------------------------------|--------------------------|-----------------------------------|------------------|
| | | Vector No. | Vector Table Address Offset | | | | |
| A/D* | ADI | 136 | H'00000220 to H'00000223 | 0 to 15 (0) | IPRG (15–12) | — | High |
| | | 138 | H'00000228 to H'0000022B | | | | |
| CMT0 | CMI0 | 144 | H'00000240 to H'00000243 | 0 to 15 (0) | IPRG (7–4) | — | |
| CMT1 | CMI1 | 148 | H'00000250 to H'00000253 | 0 to 15 (0) | IPRG (3–0) | — | |
| WDT | ITI | 152 | H'00000260 to H'00000263 | 0 to 15 (0) | IPRH (15–12) | High | |
| BSC | CMI | 153 | H'00000264 to H'00000267 | 0 to 15 (0) | | Low | Low |

Note: Vector No. 136 = SH7014 only
138 = SH7016, SH7017 only

6.3 Description of Registers

6.3.1 Interrupt Priority Registers A–H (IPRA–IPRH)

Interrupt priority registers A–H (IPRA–IPRH) are 16-bit readable/writable registers that set priority levels from 0 to 15 for IRQ interrupts and on-chip peripheral module interrupts. Correspondence between interrupt request sources and each of the IPRA–IPRH bits is shown in table 6.4.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | | | | | | | | |
| 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 |
| | | | | | | | | |
| 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 |

Table 6.4 Interrupt Request Sources and IPRA–IPRH

| Register | Bits | | | |
|-------------------------------|----------|----------|----------|----------|
| | 15–12 | 11–8 | 7–4 | 3–0 |
| Interrupt priority register A | IRQ0 | IRQ1 | IRQ2 | IRQ3 |
| Interrupt priority register B | Reserved | Reserved | IRQ6 | IRQ7 |
| Interrupt priority register C | DMAC0 | DMAC1 | Reserved | Reserved |
| Interrupt priority register D | MTU0 | MTU0 | MTU1 | MTU1 |
| Interrupt priority register E | MTU2 | MTU2 | Reserved | Reserved |
| Interrupt priority register F | Reserved | Reserved | SCI0 | SCI1 |
| Interrupt priority register G | A/D | Reserved | CMT0 | CMT1 |
| Interrupt priority register H | WDT, BSC | Reserved | Reserved | Reserved |

As indicated in table 6.4, four $\overline{\text{IRQ}}$ pins or groups of 4 on-chip peripheral modules are allocated to each register. Each of the corresponding interrupt priority ranks are established by setting a value from H'0 (0000) to H'F (1111) in each of the four-bit groups 15–12, 11–8, 7–4 and 3–0. Interrupt priority rank becomes level 0 (lowest) by setting H'0, and level 15 (highest) by setting H'F. If multiple on-chip peripheral modules are assigned to WDT and BSC, those multiple modules are set to the same priority rank.

IPRA–IPRH are initialized to H'0000 by a power-on reset or a manual reset. They are not initialized in standby mode.

6.3.2 Interrupt Control Register (ICR)

The ICR is a 16-bit register that sets the input signal detection mode of the external interrupt input pin NMI and $\overline{\text{IRQ0}}\text{--}\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$ and indicates the input signal level to the NMI pin. It is initialized by power-on reset, but is not initialized by standby mode.

| | | | | | | | | |
|----------------|-------|-------|-------|-------|----|----|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | NMIL | — | — | — | — | — | — | NMIE |
| Initial value: | * | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | IRQ0S | IRQ1S | IRQ2S | IRQ3S | — | — | IRQ6S | IRQ7S |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R | R | R/W | R/W |

Note: When NMI input is high: 1; when NMI input is low: 0

- Bit 15—NMI Input Level (NMIL): Sets the level of the signal input at the NMI pin. This bit can be read to determine the NMI pin level. This bit cannot be modified.

| Bit 15: NMIL | Description |
|--------------|-------------------------|
| 0 | NMI input level is low |
| 1 | NMI input level is high |

- Bits 14 to 9, 3, 2—Reserved: These bits always read as 0. The write value should always be 0.
- Bit 8—NMI Edge Select (NMIE)

| Bit 8: NMIE | Description |
|-------------|--|
| 0 | Interrupt request is detected on falling edge of NMI input (initial value) |
| 1 | Interrupt request is detected on rising edge of NMI input |

- Bits 7 to 4, 1, 0—IRQ0–IRQ3, IRQ6, IRQ7 Sense Select (IRQ0S–IRQ3S, IRQ6S, IRQ7S): These bits set the IRQ0–IRQ3, IRQ6, IRQ7 interrupt request detection mode.

| Bits 7–4, 1, 0: IRQ0S–IRQ3S, IRQ6S, IRQ7S | Description |
|---|---|
| 0 | Interrupt request is detected on low level of IRQ input (initial value) |
| 1 | Interrupt request is detected on falling edge of IRQ input |

6.3.3 IRQ Status Register (ISR)

The ISR is a 16-bit register that indicates the interrupt request status of the external interrupt input pins $\overline{\text{IRQ0}}\text{--}\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$. When IRQ interrupts are set to edge detection, held interrupt requests can be withdrawn by writing a 0 to IRQnF after reading an $\text{IRQnF} = 1$.

A power-on reset initializes ISR but the standby mode does not.

| | | | | | | | | |
|----------------|-------|-------|-------|-------|----|----|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| | | | | | | | | |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | IRQ0F | IRQ1F | IRQ2F | IRQ3F | — | — | IRQ6F | IRQ7F |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R | R | R/W | R/W |

- Bits 15 to 8, 3, 2—Reserved: These bits always read as 0. The write value should always be 0.
- Bits 7 to 4, 1, 0—IRQ0–IRQ3, IRQ6, IRQ7 Flags (IRQ0F–IRQ3F, IRQ6F, IRQ7F): These bits display the IRQ0–IRQ3, IRQ6, IRQ7 interrupt request status.

Bits 7–4, 1, 0: IRQ0F–IRQ3F, IRQ6F, IRQ7F

| | Detection Setting | Description |
|---|-------------------|---|
| 0 | Level detection | No IRQn interrupt request exists. Clear conditions: When $\overline{\text{IRQn}}$ input is high level |
| | Edge detection | No IRQn interrupt request was detected. (initial value) Clear conditions: 1. When a 0 is written after reading $\text{IRQnF} = 1$ status 2. When IRQn interrupt exception processing has been executed |
| 1 | Level detection | An IRQn interrupt request exists. Set conditions: When $\overline{\text{IRQn}}$ input is low level |
| | Edge detection | An IRQn interrupt request was detected. Set conditions: When a falling edge occurs at an $\overline{\text{IRQn}}$ input |

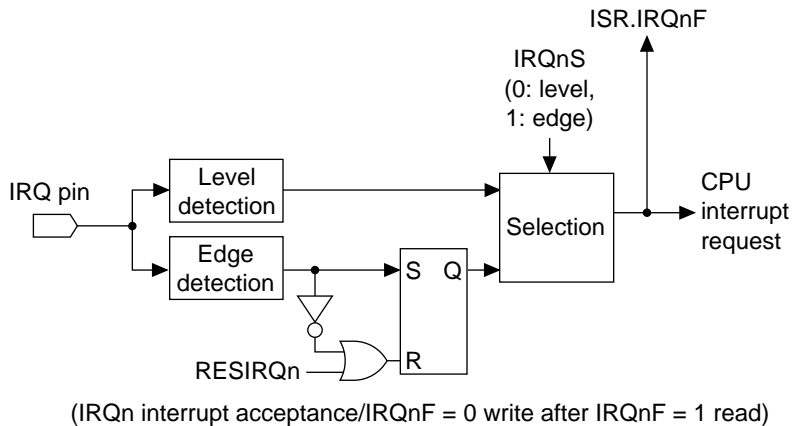


Figure 6.2 External Interrupt Process

6.4 Interrupt Operation

6.4.1 Interrupt Sequence

The sequence of interrupt operations is explained below. Figure 6.3 is a flowchart of the operations.

1. The interrupt request sources send interrupt request signals to the interrupt controller.
2. The interrupt controller selects the highest priority interrupt in the interrupt requests sent, following the priority levels set in interrupt priority level setting registers A–H (IPRA–IPRH). Lower-priority interrupts are ignored. They are held pending until interrupt requests designated as edge-detect type are accepted. For IRQ interrupts, however, withdrawal is possible by accessing the IRQ status register (ISR). See section 6.2.2, IRQ Interrupts, for details. Interrupts held pending due to edge detection are cleared by a power-on reset. If two of these interrupts have the same priority level or if multiple interrupts occur within a single module, the interrupt with the highest default priority or the highest priority within its IPR setting range (as indicated in table 6.3) is selected.
3. The interrupt controller compares the priority level of the selected interrupt request with the interrupt mask bits (I3–I0) in the CPU’s status register (SR). If the request priority level is equal to or less than the level set in I3–I0, the request is ignored. If the request priority level is higher than the level in bits I3–I0, the interrupt controller accepts the interrupt and sends an interrupt request signal to the CPU.
4. The CPU detects the interrupt request sent from the interrupt controller when it decodes the next instruction to be executed. Instead of executing the decoded instruction, the CPU starts interrupt exception processing (figure 6.4).
5. SR and PC are saved onto the stack.

6. The priority level of the accepted interrupt is copied to the interrupt mask level bits (I3 to I0) in the status register (SR).
7. The CPU reads the start address of the exception service routine from the exception vector table for the accepted interrupt, jumps to that address, and starts executing the program there. This jump is not a delay branch.

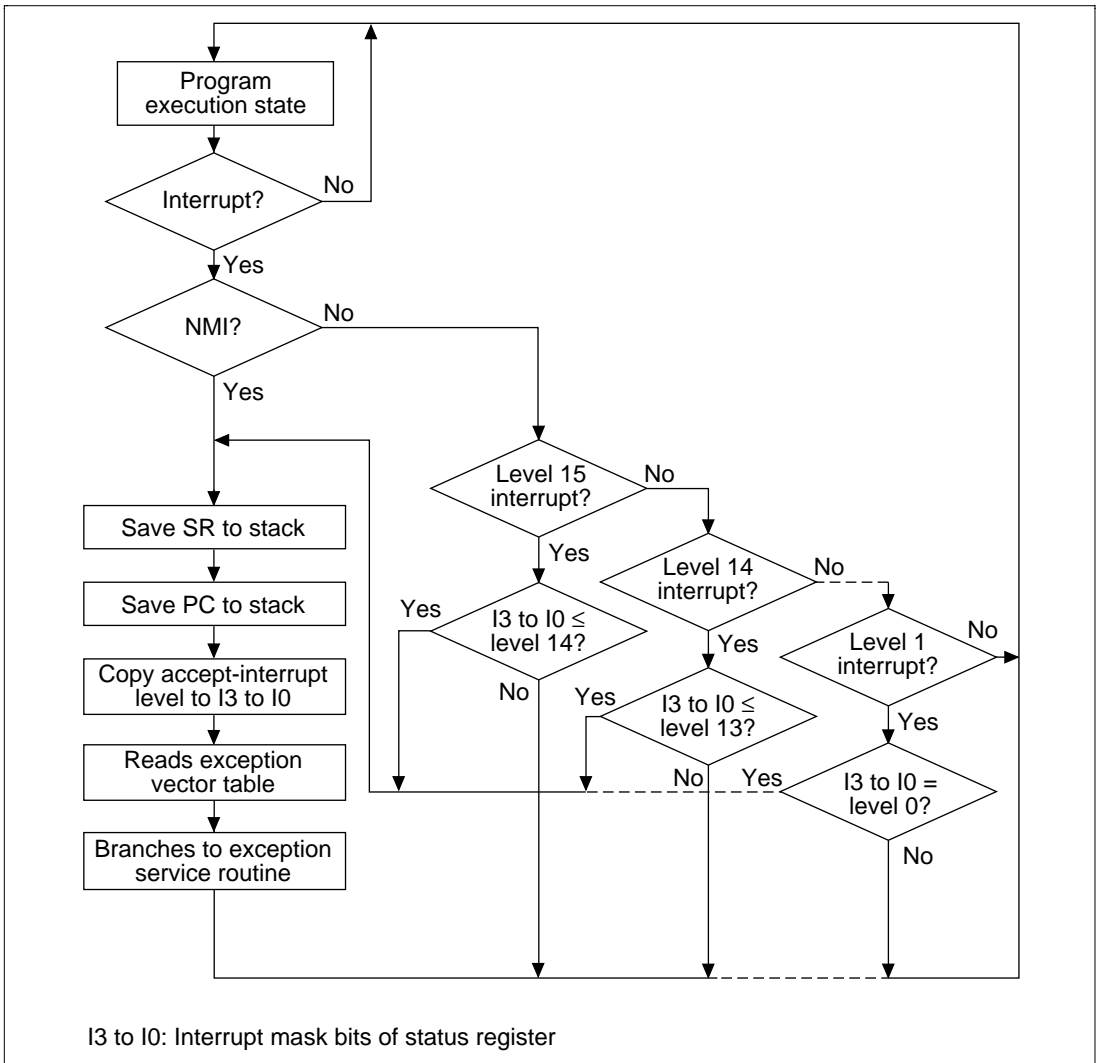


Figure 6.3 Interrupt Sequence Flowchart

6.4.2 Stack after Interrupt Exception Processing

Figure 6.4 shows the stack after interrupt exception processing.

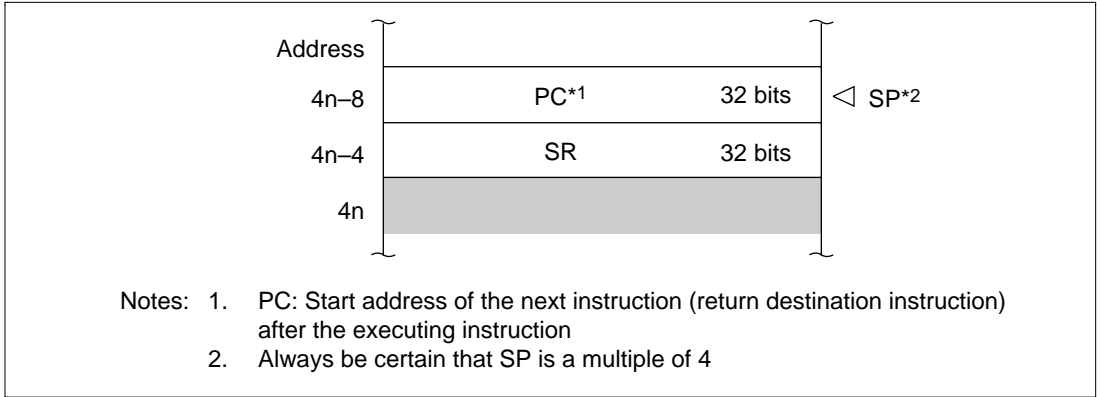


Figure 6.4 Stack after Interrupt Exception Processing

6.5 Interrupt Response Time

Table 6.5 indicates the interrupt response time, which is the time from the occurrence of an interrupt request until the interrupt exception processing starts and fetching of the first instruction of the interrupt service routine begins. Figure 6.5 shows the pipeline when an IRQ interrupt is accepted.

Table 6.5 Interrupt Response Time

| Item | Number of States | | Notes | |
|--|------------------------|-----------------------------|---|-----------------------------------|
| | NMI, Peripheral Module | IRQ | | |
| DMAC active judgment | 0 or 1 | 1 | 1 state required for interrupt signals for which DMAC activation is possible | |
| Compare identified interrupt priority with SR mask level | 2 | 3 | | |
| Wait for completion of sequence currently being executed by CPU | $X (\geq 0)$ | | The longest sequence is for interrupt or address-error exception processing ($X = 4 + m1 + m2 + m3 + m4$). If an interrupt-masking instruction follows, however, the time may be even longer. | |
| Time from start of interrupt exception processing until fetch of first instruction of exception service routine starts | $5 + m1 + m2 + m3$ | | Performs the PC and SR saves and vector address fetch. | |
| Interrupt response time | Total: | $7 + m1 + m2 + m3$ | $9 + m1 + m2 + m3$ | |
| | Minimum: | 10 | 12 | 0.35 to 0.42 μ s at 28.7 MHz |
| | Maximum: | $12 + 2(m1 + m2 + m3) + m4$ | $13 + 2(m1 + m2 + m3) + m4$ | 0.67 to 0.70 μ s at 28.7 MHz* |

Note: When $m1 = m2 = m3 = m4 = 1$

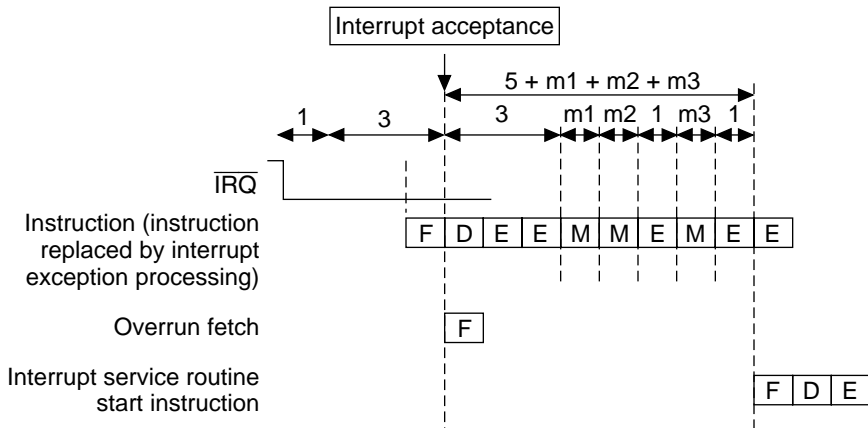
$m1$ – $m4$ are the number of states needed for the following memory accesses.

$m1$: SR save (longword write)

$m2$: PC save (longword write)

$m3$: Vector address read (longword read)

$m4$: Fetch first instruction of interrupt service routine



F: Instruction fetch (instruction fetched from memory where program is stored).
 D: Instruction decoding (fetched instruction is decoded).
 E: Instruction execution (data operation and address calculation is performed according to the results of decoding).
 M: Memory access (data in memory is accessed).

Figure 6.5 Pipeline when an IRQ Interrupt is Accepted

6.6 Data Transfer with Interrupt Request Signals

The following data transfers can be done using interrupt request signals:

- Activate DMAC only, without generating CPU interrupt

Among interrupt sources, those designated as DMAC activating sources are masked and not input to the INTC. The masking condition is listed below:

$$\text{Mask condition} = \text{DME} \bullet (\text{DE0} \bullet \text{source selection 0} + \text{DE1})$$

Figure 6.6 shows a control block diagram.

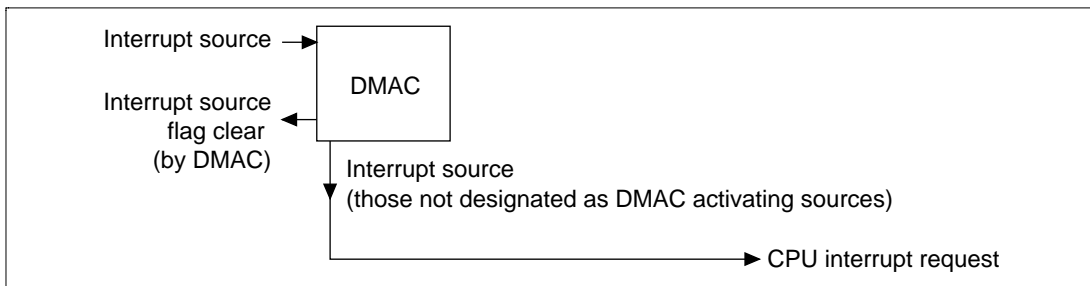


Figure 6.6 Interrupt Control Block Diagram

6.6.1 Handling DMAC Activating Sources but Not CPU Interrupt Sources

1. Select the DMAC as a source and set the DME bit to 1. CPU interrupt sources and DTC activating sources are masked regardless of the interrupt priority level register settings or DTC register settings.
2. Activating sources are applied to the DMAC when interrupts occur.
3. The DMAC clears activating sources at the time of data transfer.

6.6.2 Treating CPU Interrupt Sources but Not DMAC Activating Sources

1. Either do not select the DMAC as a source, or clear the DME bit to 0.
2. When interrupts occur, interrupt requests are sent to the CPU.
3. The CPU clears the interrupt source and performs the necessary processing in the interrupt processing routine.

Section 7 Cache Memory (CAC)

7.1 Overview

The LSI has an on-chip cache memory (CAC) with 1 kbyte of cache data and a 256-entry cache tag. The cache data and cache tag space can be used as on-chip RAM space when the cache is not being used.

7.1.1 Features

The CAC has the following features. The cache tag and cache data configuration is shown in figure 7.1.

- 1-kbyte capacity
- External memory (CS space and DRAM space) instruction code and PC relative data caching
- 256 entry cache tag (tag address 15 bits)
- 4-byte line length
- Direct map replacement algorithm
- Valid flag (1 bit) included for purges

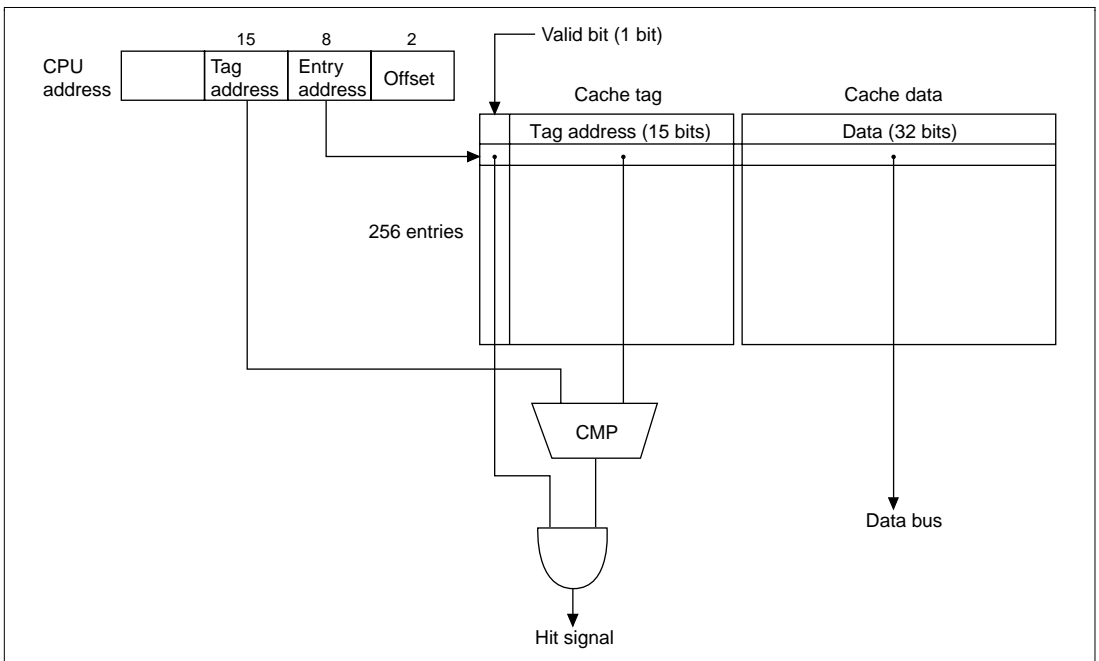


Figure 7.1 Cache Tag and Cache Data Configuration

7.2 Register Explanation

7.2.1 Cache Control Register (CCR)

The cache control register (CCR) selects the cache enable/disable of each space.

The CCR is a 16-bit readable/writable register. It is initialized to H'0000 by power on resets, but is not initialized by standby mode.

| | | | | | | | | |
|----------------|----|----|----|------------|-----------|-----------|-----------|-----------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | CE DRAM | CE CS3 | CE CS2 | CE CS1 | CE CS0 |
| Initial value: | * | * | * | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R/W | R/W | R/W | R/W | R/W |

Note: Bits 15–5 are undefined.

- Bits 15–5—Reserved: Reading these bits gives undefined values. The write value should always be 0.
- Bit 4—DRAM Space Cache Enable (CEDRAM): Selects whether to use DRAM space as a cache object (enable) or to exclude it (disable). A 0 disables, and a 1 enables such use.

Bit 4: CEDRAM

Description

| | |
|---|---|
| 0 | DRAM space cache disabled (initial value) |
|---|---|

| | |
|---|--------------------------|
| 1 | DRAM space cache enabled |
|---|--------------------------|

- Bit 3—CS3 Space Cache Enable (CECS3): Selects whether to use CS3 space as a cache object (enable) or to exclude it (disable). A 0 disables, and a 1 enables such use.

Bit 3: CECS3

Description

| | |
|---|--|
| 0 | CS3 space cache disabled (initial value) |
|---|--|

| | |
|---|-------------------------|
| 1 | CS3 space cache enabled |
|---|-------------------------|

- Bit 2—CS2 Space Cache Enable (CECS2): Selects whether to use CS2 space as a cache object (enable) or to exclude it (disable). A 0 disables, and a 1 enables such use.

| Bit 2: CECS2 | Description |
|--------------|--|
| 0 | CS2 space cache disabled (initial value) |
| 1 | CS2 space cache enabled |

- Bit 1—CS1 Space Cache Enable (CECS1): Selects whether to use CS1 space as a cache object (enable) or to exclude it (disable). A 0 disables, and a 1 enables such use.

| Bit 1: CECS1 | Description |
|--------------|--|
| 0 | CS1 space cache disabled (initial value) |
| 1 | CS1 space cache enabled |

- Bit 0—CS0 Space Cache Enable (CECS0): Selects whether to use CS0 space as a cache object (enable) or to exclude it (disable). A 0 disables, and a 1 enables such use.

| Bit 0: CECS0 | Description |
|--------------|--|
| 0 | CS0 space cache disabled (initial value) |
| 1 | CS0 space cache enabled |

7.3 Address Array and Data Array

There is a special cache space for controlling the cache. This space is divided into an address array and a data array, where addresses (tag address, including valid bit) and data (4-byte line length) for cache control are recorded. The special cache space is shown in table 7.2. It can be used as on-chip RAM space when the cache is not being used.

Table 7.2 Special Cache Space

| Space Classification | Address | Size | Bus Width |
|----------------------|----------------------------|---------|-----------|
| Address array | H'FFFFFF000 to H'FFFFFF3FF | 1 kbyte | 32 bits |
| Data array | H'FFFFFF400 to H'FFFFFF7FF | 1 kbyte | 32 bits |

7.3.1 Cache Address Array Read/Write Space

The cache address array has a compulsory read/write (figure 7.3).

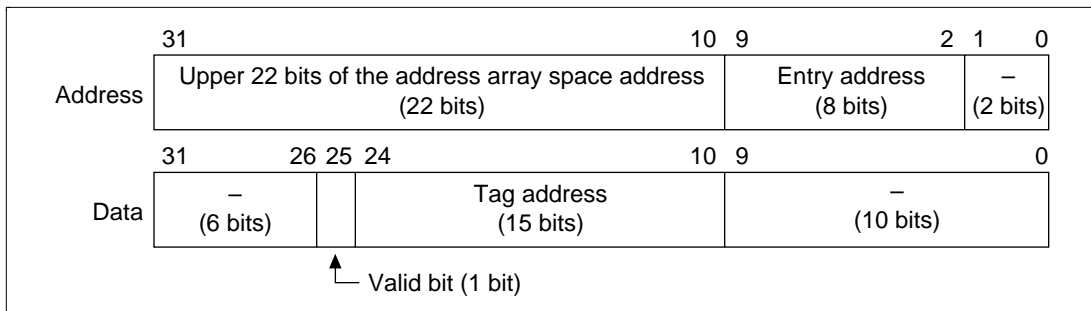


Figure 7.3 Cache Address Array

Address Array Read: Designates entry address and reads out the corresponding tag address value/valid bit value.

Address Array Write: Designates entry address and writes the designated tag address value/valid bit value.

7.3.2 Cache Data Array Read/Write Space

The cache data array has a compulsory read/write (figure 7.4).

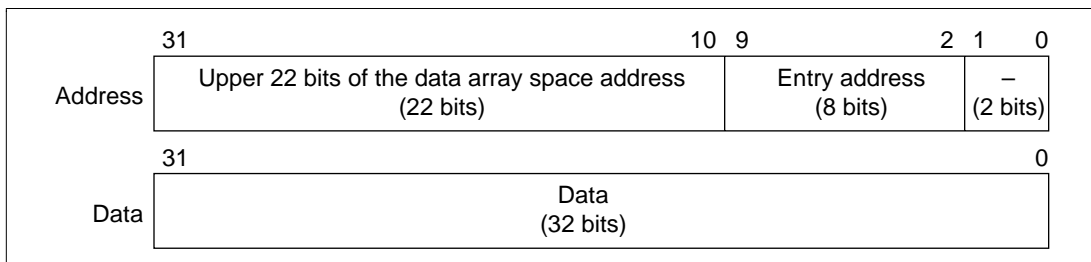


Figure 7.4 Cache Data Array

Data Array Read: Designates entry address and reads out the corresponding line of data.

Data Array Write: Designates entry address and writes designated data to the corresponding line.

7.4 Cautions on Use

7.4.1 Cache Initialization

Always initialize the cache before enabling it. Specifically, use an address array write to write 0 to all valid bits for all entries (256 times), that is, those in the address range H'FFFFFF000–H'FFFFFF3FFF.

Writes to the address array or data array by the CPU, or DMAC are not possible while the cache is enabled. For reads, undefined values will be read out.

7.4.2 Forced Access to Address Array and Data Array

While the cache is enabled, it is not possible to write to the address array or data array via the CPU, or DMAC, and a read will return an undefined value. The cache must be disabled before making a forced access to the address array or data array.

7.4.3 Cache Miss Penalty and Cache Fill Timing

When a cache miss occurs, a single idle cycle is generated as a penalty immediately before the cache fill (access from external memory in the event of a cache miss), as shown in figure 7.5. However, in the case of consecutive cache misses, idle cycles are not generated for the second and subsequent cache misses, as shown in figure 7.6.

As the timing for a cache fill from normal space, the CS assert period immediately before the end of the bus cycle (or the last bus cycle when two or four bus cycles are generated, such as in a word access to 8-bit space) is extended by an additional cycle, as shown in figures 7.5 and 7.6.

Similarly, as the timing for a cache fill from DRAM space, the RAS assert period immediately before the end of the bus cycle is extended by an additional cycle as shown in figure 7.7. In RAS down mode, the next bus cycle is delayed by one cycle as shown in figure 7.8.

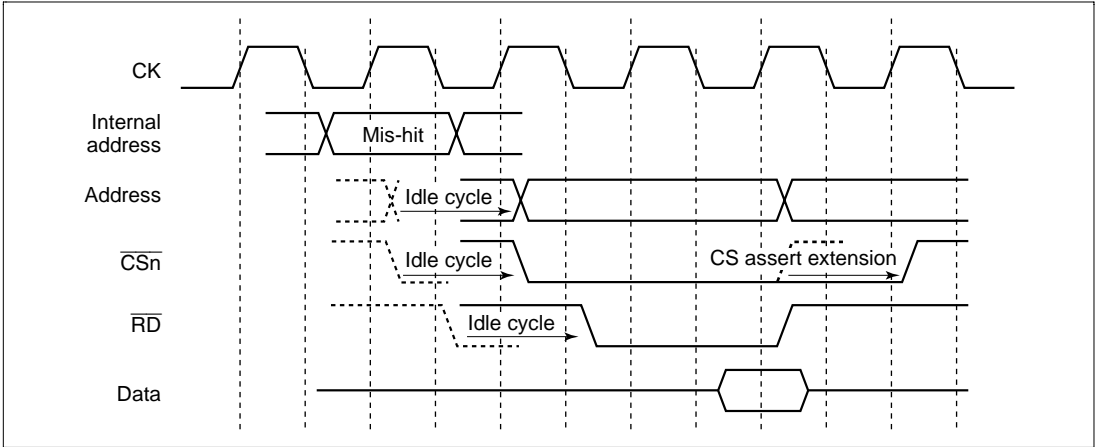


Figure 7.5 Cache Fill Timing in Case of Non-Consecutive Cache Miss from Normal Space (No Wait, No CS Assert Extension)

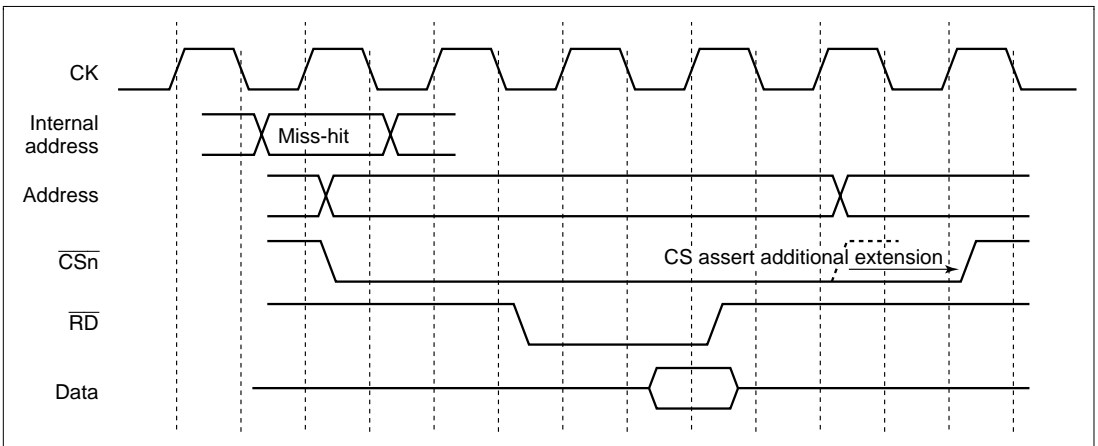


Figure 7.6 Cache Fill Timing in Case of Consecutive Cache Misses from Normal Space (No Wait, CS Assert Extension)

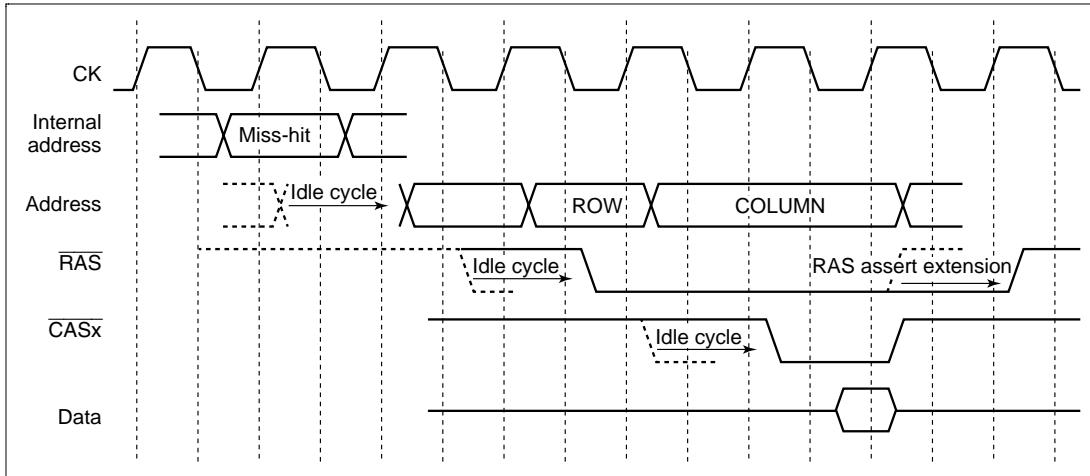


Figure 7.7 Cache Fill Timing in Case of Non-Consecutive Cache Miss from DRAM Space (Normal Mode, TPC = 0, RCD = 0, No Wait)

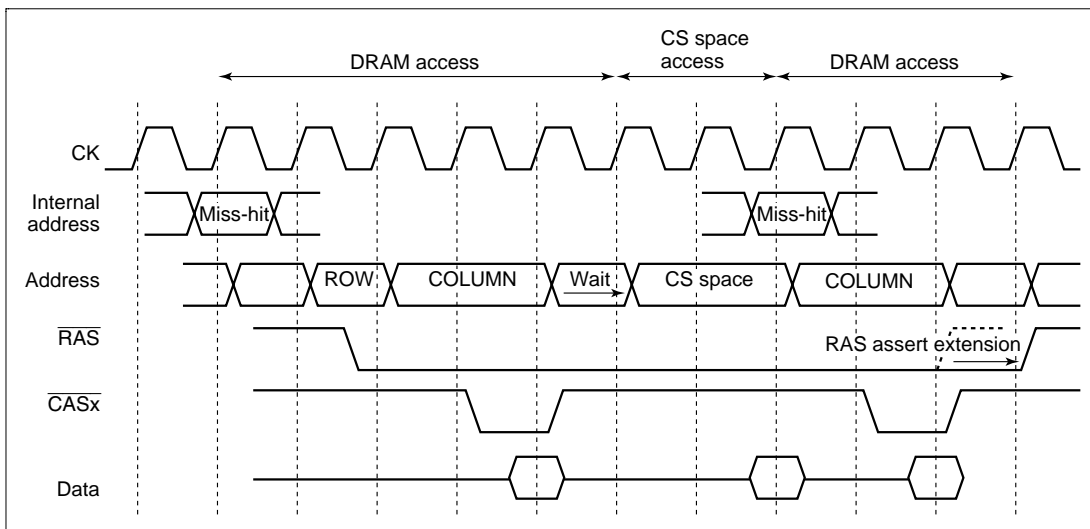


Figure 7.8 Cache Fill Timing in Case of Consecutive Cache Misses from DRAM Space (RAS Down Mode, TPC = 0, RCD = 0, No Wait)

7.4.4 Cache Hit after Cache Miss

The first cache hit after a cache miss is regarded as a cache miss, and a cache fill without idle cycle generation is performed. The next hit operates as a cache hit.

Section 8 Bus State Controller (BSC)

8.1 Overview

The bus state controller (BSC) divides up the address spaces and outputs control for various types of memory. This enables memories like DRAM, SRAM, and ROM to be linked directly to the LSI without external circuitry.

8.1.1 Features

The BSC has the following features:

- Address space is divided into five spaces
 - A maximum linear 2 Mbytes for CS0 address space in modes with on-chip ROM enabled, and a maximum linear 4 Mbytes in modes with on-chip ROM disabled
 - A maximum linear 4 Mbytes for each of address spaces CS1, CS2, and CS3
 - A maximum linear 16 Mbytes for DRAM dedicated space
 - Bus width can be selected for each space (8 or 16 bits)
 - Wait states can be inserted by software for each space
 - Wait states can be inserted via the $\overline{\text{WAIT}}$ pin in external memory space accesses
 - Outputs control signals for each space according to the type of memory connected
- On-chip RAM interfaces
 - On-chip RAM access of 32 bits in 1 state
 - On-chip ROM access of 32 bits in 1 state
- Direct interface to DRAM
 - Multiplexes row/column addresses according to DRAM capacity
 - Supports high-speed page mode and RAS down mode
- Access control for each type of memory, peripheral LSI
 - Address/data multiplex function
- Refresh
 - Supports CAS-before-RAS refresh (auto-refresh) and self-refresh
- Refresh counter can be used as an interval timer
 - Interrupt request generated upon compare match (CMI interrupt request signal)

8.1.2 Block Diagram

Figure 8.1 shows the BSC block diagram.

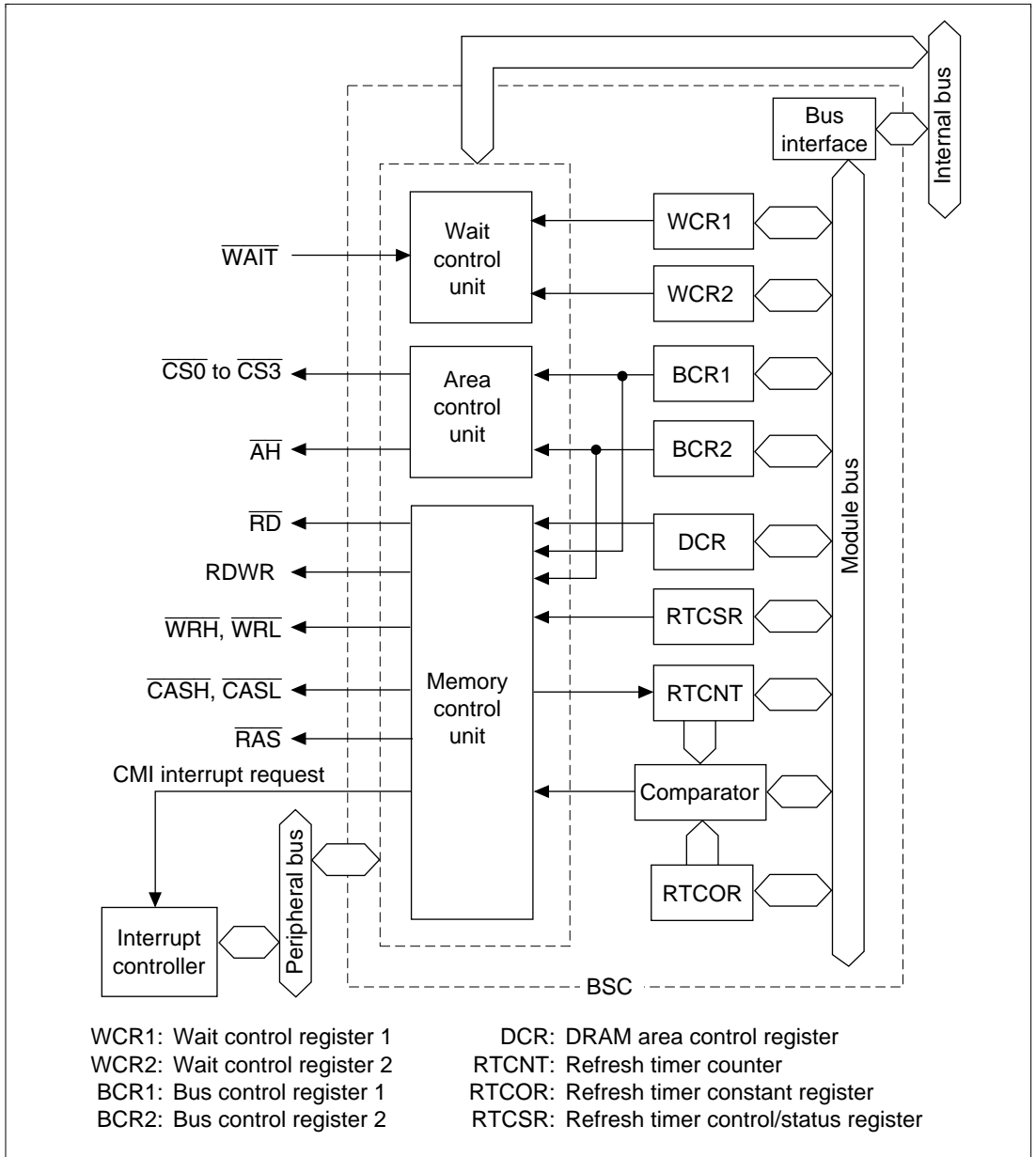


Figure 8.1 BSC Block Diagram

8.1.3 Pin Configuration

Table 8.1 shows the bus state controller pin configuration.

Table 8.1 Pin Configuration

| Signal | I/O | Description |
|--|-----|---|
| A21–A0 | O | Address output (A21–A18 become input ports in power-on reset) |
| D15–D0 | I/O | 16-bit data bus. D15–D0 are address output and data I/O during address/data multiplex I/O. |
| $\overline{\text{CS0}}$ – $\overline{\text{CS3}}$ | O | Chip select |
| $\overline{\text{RD}}$ | O | Strobe that indicates the read cycle for ordinary space/multiplex I/O. Also output during DRAM access. |
| $\overline{\text{WRH}}$ | O | Strobe that indicates a write cycle to the 3rd byte (D15–D8) for ordinary space/multiplex I/O. Also output during DRAM access. |
| $\overline{\text{WRL}}$ | O | Strobe that indicates a write cycle to the least significant byte (D7–D0) for ordinary space/multiplex I/O. Also output during DRAM access. |
| $\overline{\text{RDWR}}$ | O | Strobe indicating a write cycle to DRAM (used for DRAM space) |
| $\overline{\text{RAS}}$ | O | RAS signal for DRAM (used for DRAM space) |
| $\overline{\text{CASH}}$ | O | CAS signal when accessing the higher byte (D15–D8) of DRAM (used for DRAM space) |
| $\overline{\text{CASL}}$ | O | CAS signal when accessing the lower byte (D7–D0) of DRAM (used for DRAM space) |
| $\overline{\text{AH}}$ | O | Signal to hold the address during address/data multiplex |
| $\overline{\text{WAIT}}$ | I | Wait state request signal |

8.1.4 Register Configuration

The BSC has eight registers. These registers are used to control wait states, bus width, and interfaces with memories like DRAM, SRAM, and ROM, as well as refresh control. The register configurations are listed in table 8.2.

All registers are 16 bits. Do not access DRAM space before completing the memory interface settings. All BSC registers are all initialized by a power-on reset. Values are maintained in standby mode.

Table 8.2 Register Configuration

| Name | Abbr. | R/W | Initial Value | Address | Access Size |
|---------------------------------------|-------|-----|---------------|------------|-------------|
| Bus control register 1 | BCR1 | R/W | H'200F | H'FFFF8620 | 8, 16, 32 |
| Bus control register 2 | BCR2 | R/W | H'FFFF | H'FFFF8622 | 8, 16, 32 |
| Wait state control register 1 | WCR1 | R/W | H'FFFF | H'FFFF8624 | 8, 16, 32 |
| Wait state control register 2 | WCR2 | R/W | H'000F | H'FFFF8626 | 8, 16, 32 |
| DRAM area control register | DCR | R/W | H'0000 | H'FFFF862A | 8, 16, 32 |
| Refresh timer control/status register | RTCSR | R/W | H'0000 | H'FFFF862C | 8, 16, 32 |
| Refresh timer counter | RTCNT | R/W | H'0000 | H'FFFF862E | 8, 16, 32 |
| Refresh time constant register | RTCOR | R/W | H'0000 | H'FFFF8630 | 8, 16, 32 |

8.1.5 Address Map

Figure 8.2 shows the address format used by the SH7014.

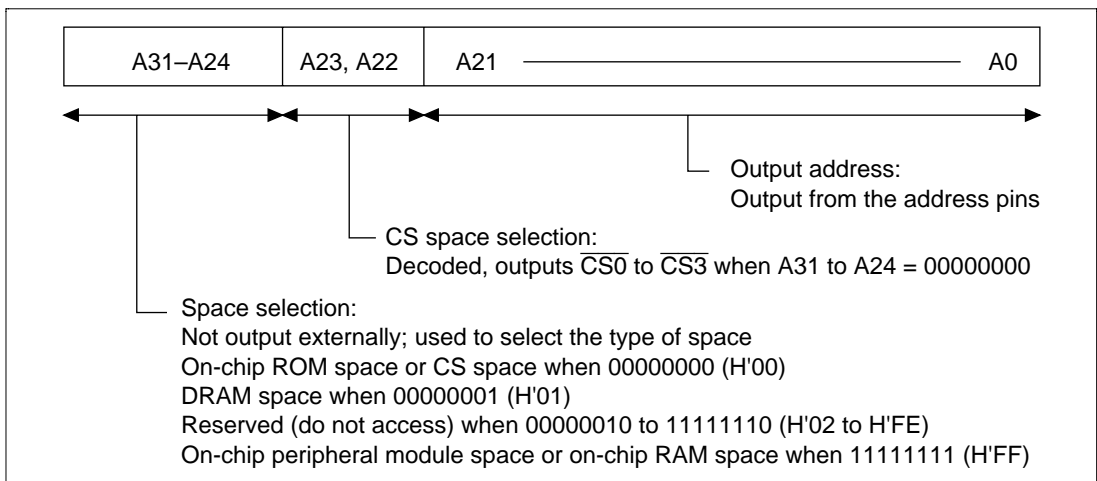


Figure 8.2 Address Format

This LSI uses 32-bit addresses:

- A31 to A24 are used to select the type of space and are not output externally.
- Bits A23 and A22 are decoded and output as chip select signals ($\overline{CS0}$ to $\overline{CS3}$) for the corresponding areas when bits A31 to A24 are 00000000.
- A21 to A0 are output externally.

Table 8.3 shows an address map.

Table 8.3 Address Map

- In on-chip ROM enabled mode*⁶

| Address | Space | Memory | Size | Bus Width |
|--|---------------------------|---------------------------------------|------------|-------------------------|
| H'00000000 to H'0001FFFF* ⁷ | On-chip ROM | On-chip ROM | 128 kbytes | 32 bits |
| H'00020000 to H'001FFFFFF | Reserved | | | |
| H'00200000 to H'003FFFFFF | CS0 space | Ordinary space | 2 Mbytes | 8/16 bits* ¹ |
| H'00400000 to H'007FFFFFF | CS1 space | Ordinary space | 4 Mbytes | 8/16 bits* ¹ |
| H'00800000 to H'00BFFFFFF | CS2 space | Ordinary space | 4 Mbytes | 8/16 bits* ¹ |
| H'00C00000 to H'00FFFFFF | CS3 space | Ordinary space or multiplex I/O space | 4 Mbytes | 8/16 bits* ² |
| H'01000000 to H'01FFFFFF | DRAM space | DRAM | 16 Mbytes | 8/16 bits* ¹ |
| H'02000000 to H'FFFF7FFF | Reserved | | | |
| H'FFFF8000 to H'FFFF87FF | On-chip peripheral module | On-chip peripheral module | 2 kbytes | 8/16 bits |
| H'FFFF8800 to H'FFFFEFFF | Reserved | | | |
| H'FFFFF000 to H'FFFFFFF* ⁷ | On-chip RAM | On-chip RAM | 4 kbytes | 32 bits |

- In on-chip ROM disabled mode

| Address | Space | Memory | Size | Bus Width |
|---------------------------------------|---------------------------|---------------------------------------|-----------|-------------------------|
| H'00000000 to H'003FFFFFFF | CS0 space | Ordinary space | 4 Mbytes | 8/16 bits* ³ |
| H'00400000 to H'007FFFFFFF | CS1 space | Ordinary space | 4 Mbytes | 8/16 bits* ¹ |
| H'00800000 to H'00BFFFFFFF | CS2 space | Ordinary space | 4 Mbytes | 8/16 bits* ¹ |
| H'00C00000 to H'00FFFFFFF | CS3 space | Ordinary space or multiplex I/O space | 4 Mbytes | 8/16 bits* ² |
| H'01000000 to H'01FFFFFFF | DRAM space | DRAM | 16 Mbytes | 8/16 bits* ¹ |
| H'02000000 to H'FFFFFF7FFF | Reserved | | | |
| H'FFFF8000 to H'FFFF87FF | On-chip peripheral module | On-chip peripheral module | 2 kbytes | 8/16 bits |
| H'FFFF8800 to H'FFFFEFFF | Reserved | | | |
| H'FFFFF000 to H'FFFFFFF* ⁷ | On-chip RAM | On-chip RAM | 4 kbytes | 32 bits |

- Notes:
1. Selected by on-chip register settings.
 2. Ordinary space: selected by on-chip register settings.
Multiplex I/O space: 8/16 bits selected by the A14 bit.
 3. 8/16 bits selected by the mode pin.
 4. In the 64-kbyte on-chip ROM version (SH7016), on-chip ROM addresses are H'00000000 to H'0000FFFF, and addresses H'00010000 to H'0001FFFF are reserved space.
 5. In single-chip mode, space other than on-chip ROM, on-chip RAM, and on-chip peripheral module space cannot be used.
 6. SH7016/17 only.
 7. In the 3-kbyte on-chip RAM versions (SH7014/16), on-chip RAM addresses are H'FFFFF000 to H'FFFFFBFF, and addresses H'FFFFFC00 to H'FFFFFFF are reserved space.
 8. Do not access reserved space, as operation cannot be guaranteed in this case.

8.2 Description of Registers

8.2.1 Bus Control Register 1 (BCR1)

BCR1 is a 16-bit read/write register that selects multiplex I/O, and specifies the bus size of the CS spaces.

Write bits 8–0 of BCR1 during the initialization stage after a power-on reset, and do not change the values thereafter. In on-chip ROM ineffective mode, do not access any CS space other than CS0 until after completion of register initialization.

BCR1 is initialized by power-on resets to H'200F, but is not initialized by software standbys.

| | | | | | | | | |
|----------------|----|----|----|----|------|------|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | IOE |
| Initial value: | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | A3SZ | A2SZ | A1SZ | A0SZ |
| Initial value: | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| R/W: | R | R | R | R | R/W | R/W | R/W | R/W |

Note: Do not write 1 to bits 4 to 7, as operation is not guaranteed in this case.

- Bits 15, 14, 12–9—Reserved: These bits always read as 0. The write value should always be 0.
- Bit 13—Reserved: This bit always read as 1. The write value should always be 1.
- Bit 8—Multiplex I/O Enable (IOE): Selects the use of CS3 space as ordinary space or address/data multiplex I/O space. A 0 selects ordinary space and a 1 selects address/data multiplex I/O space. When address/data multiplex I/O space is selected, the address and data are multiplexed and output from the data bus. When CS3 space is an address/data multiplex I/O space, bus size is decided by the A14 bit (A14 = 0: 8 bit, A14 = 1: 16 bit).

| Bit 8: IOE | Description |
|------------|---|
| 0 | CS3 space is ordinary space (initial value) |
| 1 | CS3 space is address/data multiplex I/O space |

- Bits 7 to 4—Reserved: These bits read 0 after a reset. The write value should always be 0. Operation is not guaranteed if the write value is 1.

- Bit 3—CS3 Space Size Specification (A3SZ): Specifies the CS3 space bus size. This is effective only when CS3 space is ordinary space. When CS3 space is an address/data multiplex I/O space, bus size is decided by the A14 bit.

| Bit 3: A3SZ | Description |
|-------------|-----------------------------------|
| 0 | Byte (8-bit) size (initial value) |
| 1 | Word (16-bit) size |

- Bit 2—CS2 Space Size Specification (A2SZ): Specifies the CS2 space bus size.

| Bit 2: A2SZ | Description |
|-------------|------------------------------------|
| 0 | Byte (8-bit) size |
| 1 | Word (16-bit) size (initial value) |

- Bit 1—CS1 Space Size Specification (A1SZ): Specifies the CS1 space bus size.

| Bit 1: A1SZ | Description |
|-------------|------------------------------------|
| 0 | Byte (8-bit) size |
| 1 | Word (16-bit) size (initial value) |

- Bit 0—CS0 Space Size Specification (A0SZ): Specifies the CS0 space bus size.

| Bit 0: A0SZ | Description |
|-------------|------------------------------------|
| 0 | Byte (8-bit) size |
| 1 | Word (16-bit) size (initial value) |

8.2.2 Bus Control Register 2 (BCR2)

BCR2 is a 16-bit read/write register that specifies the number of idle cycles and \overline{CS} signal assert extension of each CS space.

BCR2 is initialized by power-on resets to H'FFFF, but is not initialized by software standbys.

| | | | | | | | | |
|----------------|------|------|------|------|------|------|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | IW31 | IW30 | IW21 | IW20 | IW11 | IW10 | IW01 | IW00 |
| 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 |
| | | | | | | | | |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CW3 | CW2 | CW1 | CW0 | SW3 | SW2 | SW1 | SW0 |
| 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 |

- Bits 15–8—Idles between Cycles (IW31, IW30, IW21, IW20, IW11, IW10, IW01, IW00): These bits specify idle cycles inserted between consecutive accesses when the second one is to a different CS area after a read. Idles are used to prevent data conflict between ROM (and other memories, which are slow to turn the read data buffer off), fast memories, and I/O interfaces. Even when access is to the same area, idle cycles must be inserted when a read access is followed immediately by a write access. The idle cycles to be inserted comply with the area specification of the previous access. Refer to section 8.6, Waits between Access Cycles, for details.

IW31, IW30 specify the idle between cycles for CS3 space; IW21, IW20 specify the idle between cycles for CS2 space; IW11, IW10 specify the idle between cycles for CS1 space and IW01, IW00 specify the idle between cycles for CS0 space.

| Bit 15: IW31 | Bit 14: IW30 | Description |
|--------------|--------------|---|
| 0 | 0 | No idle cycle after accessing CS3 space |
| 0 | 1 | Inserts one idle cycle |
| 1 | 0 | Inserts two idle cycles |
| 1 | 1 | Inserts three idle cycles (initial value) |

| Bit 13: IW21 | Bit 12: IW20 | Description |
|--------------|--------------|---|
| 0 | 0 | No idle cycle after accessing CS2 space |
| | 1 | Inserts one idle cycle |
| 1 | 0 | Inserts two idle cycles |
| | 1 | Inserts three idle cycles (initial value) |

| Bit 11: IW11 | Bit 10: IW10 | Description |
|--------------|--------------|---|
| 0 | 0 | No idle cycle after accessing CS1 space |
| | 1 | Inserts one idle cycle |
| 1 | 0 | Inserts two idle cycles |
| | 1 | Inserts three idle cycles (initial value) |

| Bit 9: IW01 | Bit 8: IW00 | Description |
|-------------|-------------|---|
| 0 | 0 | No idle cycle after accessing CS0 space |
| | 1 | Inserts one idle cycle |
| 1 | 0 | Inserts two idle cycles |
| | 1 | Inserts three idle cycles (initial value) |

- Bits 7–4—Idle Specification for Continuous Access (CW3, CW2, CW1, CW0): The continuous access idle specification makes insertions to clearly delineate the bus intervals by once negating the \overline{CSn} signal when doing consecutive accesses of the same CS space. When a write immediately follows a read, the number of idle cycles inserted is the larger of the two values specified by IW and CW. Refer to section 8.6, Waits between Access Cycles, for details.

CW3 specifies the continuous access idles for CS3 space; CW2 specifies the continuous access idles for CS2 space; CW1 specifies the continuous access idles for CS1 space and CW0 specifies the continuous access idles for CS0 space.

| Bit 7: CW3 | Description |
|------------|--|
| 0 | No CS3 space continuous access idle cycles |
| 1 | One CS3 space continuous access idle cycle (initial value) |

| Bit 6: CW2 | Description |
|------------|--|
| 0 | No CS2 space continuous access idle cycles |
| 1 | One CS2 space continuous access idle cycle (initial value) |

| Bit 5: CW1 | Description |
|------------|--|
| 0 | No CS1 space continuous access idle cycles |
| 1 | One CS1 space continuous access idle cycle (initial value) |

| Bit 4: CW0 | Description |
|------------|--|
| 0 | No CS0 space continuous access idle cycles |
| 1 | One CS0 space continuous access idle cycle (initial value) |

- Bits 3–0— $\overline{\text{CS}}$ Assert Extension Specification (SW3, SW2, SW1, SW0): The $\overline{\text{CS}}$ assert cycle extension specification is for making insertions to prevent extension of the $\overline{\text{RD}}$ signal or $\overline{\text{WRx}}$ signal assert period beyond the length of the $\overline{\text{CSn}}$ signal assert period. Extended cycles insert one cycle before and after each bus cycle, which simplifies interfaces with external devices and also has the effect of extending write data hold time. Refer to section 8.3.3, $\overline{\text{CS}}$ Assert Extension Function, for details.

SW3 specifies the $\overline{\text{CS}}$ assert extension for CS3 space access; SW2 specifies the $\overline{\text{CS}}$ assert extension for CS2 space access; SW1 specifies the $\overline{\text{CS}}$ assert extension for CS1 space access and SW0 specifies the $\overline{\text{CS}}$ assert extension for CS0 space access.

| Bit 3: SW3 | Description |
|------------|---|
| 0 | No CS3 space $\overline{\text{CS}}$ assert extension |
| 1 | CS3 space $\overline{\text{CS}}$ assert extension (initial value) |

| Bit 2: SW2 | Description |
|------------|---|
| 0 | No CS2 space $\overline{\text{CS}}$ assert extension |
| 1 | CS2 space $\overline{\text{CS}}$ assert extension (initial value) |

| Bit 1: SW1 | Description |
|------------|---|
| 0 | No CS1 space $\overline{\text{CS}}$ assert extension |
| 1 | CS1 space $\overline{\text{CS}}$ assert extension (initial value) |

| Bit 0: SW0 | Description |
|------------|---|
| 0 | No CS0 space $\overline{\text{CS}}$ assert extension |
| 1 | CS0 space $\overline{\text{CS}}$ assert extension (initial value) |

8.2.3 Wait Control Register 1 (WCR1)

WCR1 is a 16-bit read/write register that specifies the number of wait cycles (0–15) for each CS space.

WCR1 is initialized by power-on resets to H'FFFF, but is not initialized by software standbys.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | W33 | W32 | W31 | W30 | W23 | W22 | W21 | W20 |
| 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 |
| | | | | | | | | |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | W13 | W12 | W11 | W10 | W03 | W02 | W01 | W00 |
| 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 |

- Bits 15–12—CS3 Space Wait Specification (W33, W32, W31, W30): Specifies the number of waits for CS3 space access.

| Bit 15: W33 | Bit 14: W32 | Bit 13: W31 | Bit 12: W30 | Description |
|----------------|----------------|----------------|----------------|---|
| 0 | 0 | 0 | 0 | No wait (external wait input disabled) |
| 0 | 0 | 0 | 1 | 1 wait external wait input enabled |
| ... | | | | |
| 1 | 1 | 1 | 1 | 15 wait external wait input enabled (initial value) |

- Bits 11–8—CS2 Space Wait Specification (W23, W22, W21, W20): Specifies the number of waits for CS2 space access.

| Bit 11: W23 | Bit 10: W22 | Bit 9: W21 | Bit 8: W20 | Description |
|----------------|----------------|---------------|---------------|---|
| 0 | 0 | 0 | 0 | No wait (external wait input disabled) |
| 0 | 0 | 0 | 1 | 1 wait external wait input enabled |
| ... | | | | |
| 1 | 1 | 1 | 1 | 15 wait external wait input enabled (initial value) |

- Bits 7–4—CS1 Space Wait Specification (W13, W12, W11, W10): Specifies the number of waits for CS1 space access.

| Bit 7: W13 | Bit 6: W12 | Bit 5: W11 | Bit 4: W10 | Description |
|---------------|---------------|---------------|---------------|---|
| 0 | 0 | 0 | 0 | No wait (external wait input disabled) |
| 0 | 0 | 0 | 1 | 1 wait external wait input enabled |
| ... | | | | |
| 1 | 1 | 1 | 1 | 15 wait external wait input enabled (initial value) |

- Bits 3–0—CS0 Space Wait Specification (W03, W02, W01, W00): Specifies the number of waits for CS0 space access.

| Bit 3: W03 | Bit 2: W02 | Bit 1: W01 | Bit 0: W00 | Description |
|---------------|---------------|---------------|---------------|---|
| 0 | 0 | 0 | 0 | No wait (external wait input disabled) |
| 0 | 0 | 0 | 1 | 1 wait external wait input enabled |
| ... | | | | |
| 1 | 1 | 1 | 1 | 15 wait external wait input enabled (initial value) |

8.2.4 Wait Control Register 2 (WCR2)

WCR2 is a 16-bit read/write register that specifies the number of access cycles for DRAM space and CS space for DMA single address mode transfers.

Do not perform any DMA single address transfers before WCR2 is set.

WCR2 is initialized by power-on resets to H'000F, but is not initialized by software standbys.

| | | | | | | | | |
|----------------|----|----|------|------|------|------|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | DDW1 | DDW0 | DSW3 | DSW2 | DSW1 | DSW0 |
| Initial value: | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| R/W: | R | R | R/W | R/W | R/W | R/W | R/W | R/W |

- Bits 15–6—Reserved: These bits always read as 0. The write value should always be 0.

- Bits 5–4—DRAM Space DMA Single Address Mode Access Wait Specification (DDW1, DDW0): Specifies the number of waits for DRAM space access during DMA single address mode accesses. These bits are independent of the DWW and DWR bits of the DCR.

| Bit 5: DDW1 | Bit 4: DDW0 | Description |
|-------------|-------------|--|
| 0 | 0 | 2-cycle (no wait) external wait disabled (initial value) |
| | 1 | 3-cycle (1 wait) external wait disabled |
| 1 | 0 | 4-cycle (2 wait) external wait enabled |
| | 1 | 5-cycle (3 wait) external wait enabled |

- Bits 3–0—CS Space DMA Single Address Mode Access Wait Specification (DSW3, DSW2, DSW1, DSW0): Specifies the number of waits for CS space access (0–15) during DMA single address mode accesses. These bits are independent of the W bits of the WCR1.

| Bit 3: DSW3 | Bit 2: DSW2 | Bit 1: DSW1 | Bit 0: DSW0 | Description |
|-------------|-------------|-------------|-------------|---|
| 0 | 0 | 0 | 0 | No wait (external wait input disabled) |
| 0 | 0 | 0 | 1 | 1 wait (external wait input enabled) |
| ... | | | | |
| 1 | 1 | 1 | 1 | 15 wait (external wait input enabled) (initial value) |

8.2.5 DRAM Area Control Register (DCR)

DCR is a 16-bit read/write register that selects the number of waits, operation mode, number of address multiplex shifts and the like for DRAM control.

Do not perform any DRAM space accesses before DCR initial settings are completed.

DCR is initialized by power-on resets to H'0000, but is not initialized by software standbys.

| | | | | | | | | |
|----------------|-----|-----|-------|-------|------|------|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | TPC | RCD | TRAS1 | TRAS0 | DWW1 | DWW0 | DWR1 | DWR0 |
| 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 |
| | DIW | — | BE | RASD | — | SZ0 | AMX1 | AMX0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R | R/W | R/W | R | R/W | R/W | R/W |

- **Bit 15—RAS Precharge Cycle Count (TPC):** Specifies the minimum number of cycles after $\overline{\text{RAS}}$ is negated before next assert.

| Bit 15: TPC | Description |
|--------------------|----------------------------|
| 0 | 1.5 cycles (initial value) |
| 1 | 2.5 cycles |

- **Bit 14—RAS-CAS Delay Cycle Count (RCD):** Specifies the number of row address output cycles.

| Bit 14: RCD | Description |
|--------------------|-------------------------|
| 0 | 1 cycle (initial value) |
| 1 | 2 cycles |

- **Bits 13–12—CAS-Before-RAS Refresh RAS Assert Cycle Count (TRAS1–TRAS0):** Specify the number of RAS assert cycles for CAS before RAS refreshes.

| Bit 13: TRAS1 | Bit 12: TRAS0 | Description |
|----------------------|----------------------|----------------------------|
| 0 | 0 | 2.5 cycles (initial value) |
| | 1 | 3.5 cycles |
| 1 | 0 | 4.5 cycles |
| | 1 | 5.5 cycles |

- **Bits 11–10—DRAM Write Cycle Wait Count (DWW1–DWW0):** Specifies the number of DRAM write cycle column address output cycles.

| Bit 11: DWW1 | Bit 10: DWW0 | Description |
|---------------------|---------------------|--|
| 0 | 0 | 2-cycle (no wait) external wait disabled (initial value) |
| | 1 | 3-cycle (1 wait) external wait disabled |
| 1 | 0 | 4-cycle (2 wait) external wait enabled |
| | 1 | 5-cycle (3 wait) external wait enabled |

- Bits 9–8—DRAM Read Cycle Wait Count (DWR1–DWR0): Specifies the number of DRAM read cycle column address output cycles.

| Bit 9: DWR1 | Bit 8: DWR0 | Description |
|-------------|-------------|--|
| 0 | 0 | 2-cycle (no wait) external wait disabled (initial value) |
| | 1 | 3-cycle (1 wait) external wait disabled |
| 1 | 0 | 4-cycle (2 wait) external wait enabled |
| | 1 | 5-cycle (3 wait) external wait enabled |

- Bit 7—DRAM Idle Cycle Count (DIW): Specifies whether to insert idle cycles, either when accessing a different external space (CS space) or when doing a DRAM write, after DRAM reads.

| Bit 7: DIW | Description |
|------------|--------------------------------|
| 0 | No idle cycles (initial value) |
| 1 | 1 idle cycle |

- Bit 6—Reserved: This bit always read as 0. The write value should always be 0.
- Bit 5—Burst Enable (BE): Specifies the DRAM operation mode.

| Bit 5: BE | Description |
|-----------|------------------------------------|
| 0 | Burst disabled (initial value) |
| 1 | DRAM high-speed page mode enabled. |

- Bit 4—RAS Down Mode (RASD): Specifies the DRAM operation mode.

| Bit 4: RASD | Description |
|-------------|--|
| 0 | Access DRAM by RAS up mode (initial value) |
| 1 | Access DRAM by RAS down mode |

- Bit 3—Reserved: This bit read 0 after a reset. The write value should always be 0. Operation is not guaranteed if the write value is 1.
- Bit 2—DRAM Bus Width Specification (SZ0): Specifies the DRAM space bus width.

| Bit 2: SZ0 | Description |
|------------|------------------------------|
| 0 | Byte (8-bit) (initial value) |
| 1 | Word (16-bit) |

- Bits 1–0—DRAM Address Multiplex (AMX1–AMX0): Specifies the DRAM address multiplex count.

| Bit 1: AMX1 | Bit 0: AMX0 | Description |
|-------------|-------------|-----------------------|
| 0 | 0 | 9 bit (initial value) |
| | 1 | 10 bit |
| 1 | 0 | 11 bit |
| | 1 | 12 bit |

8.2.6 Refresh Timer Control/Status Register (RTCSR)

RTCSR is a 16-bit read/write register that selects the refresh mode and the clock input to the refresh timer counter (RTCNT), and controls compare match interrupts (CMI).

RTCSR is initialized by power-on resets to H'0000, but is not initialized by software standbys.

| | | | | | | | | |
|----------------|----|-----|------|------|------|------|------|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | CMF | CMIE | CKS2 | CKS1 | CKS0 | RFSH | RMD |
| 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 | R/W |

- Bits 15–7—Reserved: These bits always read as 0. The write value should always be 0
- Bit 6—Compare Match Flag (CMF): This status flag, which indicates that the values of RTCNT and RTCOR match, is set/cleared under the following conditions:

| Bit 6: CMF | Description |
|------------|--|
| 0 | Clear condition: After RTCSR is read when CMF is 1, 0 is written in CMF. (initial value) |
| 1 | Set condition: RTCNT = RTCOR. When both RTCNT and RTCOR are in an initialized state (when values have not been rewritten since initialization, and RTCNT has not had its value changed due to a count-up), RTCNT and RTCOR match, as both are H'0000, but in this case CMF is not set. |

- Bit 5—Compare Match Interrupt Enable (CMIE): Enables or disables an interrupt request caused by the CMF bit of the RTCSR when CMF is set to 1.

| Bit 5: CMIE | Description |
|-------------|---|
| 0 | Disables an interrupt request caused by CMF (initial value) |
| 1 | Enables an interrupt request caused by CMF |

- Bits 4–2—Clock Select (CKS2–CKS0): Select the clock to input to RTCNT from among the seven types of internal clock obtained from dividing the system clock (ϕ).

| Bit 4: CKS2 | Bit 3: CKS1 | Bit 2: CKS0 | Description |
|-------------|-------------|-------------|--------------------------------|
| 0 | 0 | 0 | Stops count-up (initial value) |
| | | 1 | $\phi/2$ |
| | 1 | 0 | $\phi/8$ |
| | | 1 | $\phi/32$ |
| 1 | 0 | 0 | $\phi/128$ |
| | | 1 | $\phi/512$ |
| | 1 | 0 | $\phi/2048$ |
| | | 1 | $\phi/4096$ |

- Bit 1—Refresh Control (RFSH): Selects whether to use refresh control for DRAM.

| Bit 1: RFSH | Description |
|-------------|-------------------------------------|
| 0 | Do not refresh DRAM (initial value) |
| 1 | Refresh DRAM |

- Bit 0—Refresh Mode (RMD): When the RFSH bit is 1, this bit selects normal refresh or self-refresh. When the RFSH bit is 1, self-refresh mode is entered immediately after the RMD bit is set to 1. When RMD is cleared to 0, a CAS-before-RAS refresh is performed at the interval set in the refresh time constant register (RTCNT).

When set for self-refresh, the this LSI enters self-refresh mode immediately unless it is in the middle of a DRAM access. If it is, it enters self-refresh mode when the access ends. Refresh requests from the interval timer are ignored in self-refresh mode.

| Bit 0: RMD | Description |
|------------|--|
| 0 | CAS-before-RAS refresh (initial value) |
| 1 | Self-refresh |

8.2.7 Refresh Timer Counter (RTCNT)

RTCNT is a 16-bit read/write register that is used as an 8-bit up counter for refreshes or generating interrupt requests.

RTCNT counts up with the clock selected by the CKS2–CKS0 bits of the RTCSR. RTCNT values can always be read/written by the CPU. When RTCNT matches RTCOR, RTCNT is cleared to H'0000 and the CMF flag of the RTCSR is set to 1. If the RFSH bit of RTCSR is 1 and the RMD bit is 0 at this time, a CAS-before-RAS refresh is performed. Additionally, if the CMIE bit of RTCSR is a 1, a compare match interrupt (CMI) is generated.

Bits 15–8 are reserved and play no part in counter operation. They are always read as 0.

RTCNT is initialized by power-on resets to H'0000, but is not initialized by software standbys.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|----|----|----|----|----|----|---|---|
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |

| 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 |

8.2.8 Refresh Time Constant Register (RTCOR)

RTCOR is a 16-bit read/write register that establishes the compare match period with RTCNT. The values of RTCOR and RTCNT are constantly compared. When the values correspond, the compare match flag of RTCSR is set and RTCNT is cleared to 0.

When the refresh bit (RFSH) of the RTCSR is set to 1 and the RMD bit is 0, a refresh request signal is produced by this match. The refresh request signal is held until a refresh operation is performed. If the refresh request is not processed before the next match, the previous request becomes ineffective.

When the CMIE bit of the RTSCR is set to 1, an interrupt request is sent to the interrupt controller by this match signal. The interrupt request is output continuously until the CMF bit of the RTSCR is cleared.

Bits 15–8 are reserved and cannot be used in setting the period. They always read 0.

RTCOR is initialized by power-on resets to H'0000, but is not initialized by software standbys.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|----|----|----|----|----|----|---|---|
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |

| 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 |

8.3 Accessing Ordinary Space

A strobe signal is output by ordinary space accesses to provide primarily for SRAM or ROM direct connections.

8.3.1 Basic Timing

Figure 8.3 shows the basic timing of ordinary space accesses. Ordinary access bus cycles are performed in 2 states.

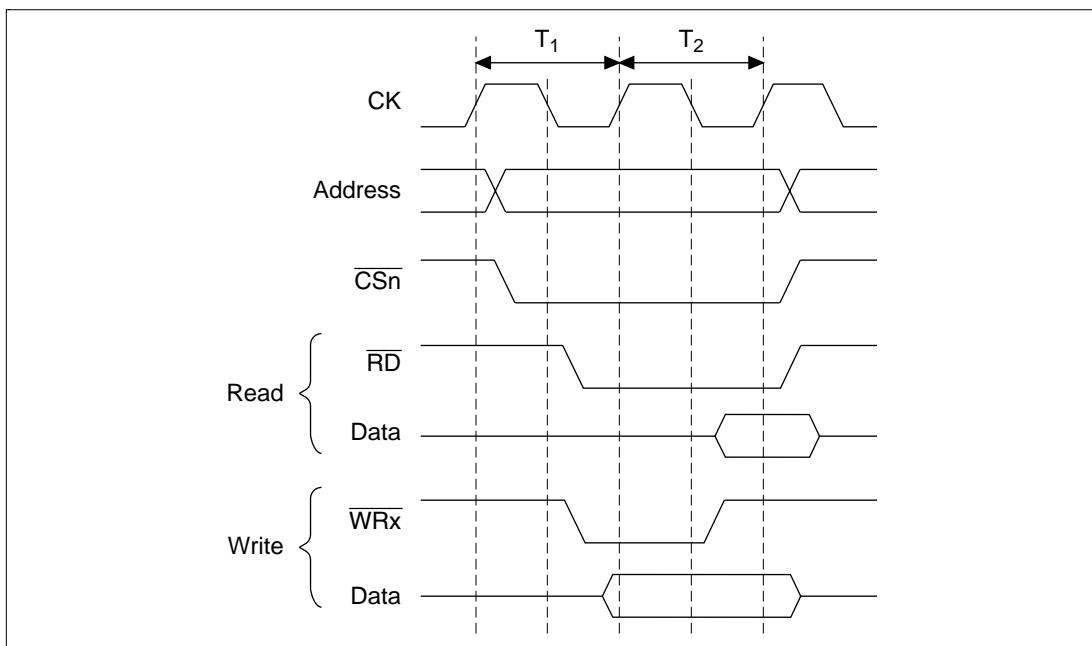


Figure 8.3 Basic Timing of Ordinary Space Access

During a read, irrespective of operand size, all bits in the data bus width for the access space (address) are fetched by the LSI on \overline{RD} , using the required byte locations.

During a write, the following signals are associated with transfer of these actual byte locations: \overline{WRH} (bits 15–8) and \overline{WRL} (bits 7–0).

8.3.2 Wait State Control

The number of wait states inserted into ordinary space access states can be controlled using the WCR settings (figure 8.4). The specified number of T_w cycles are inserted as software wait cycles at the timing shown in figure 8.4.

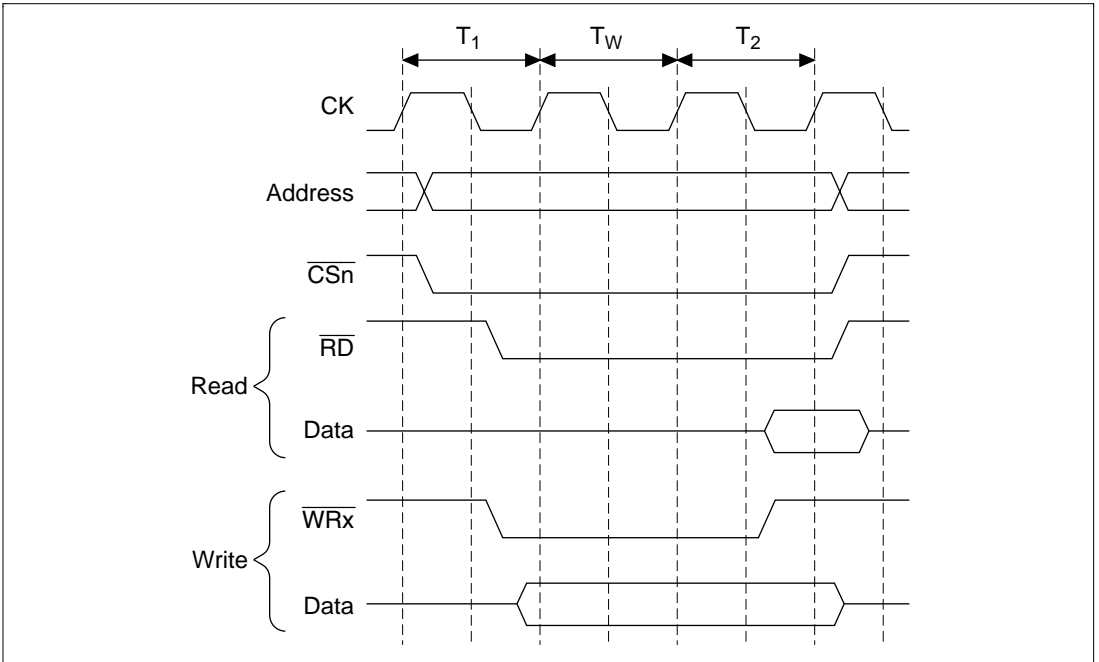


Figure 8.4 Wait Timing of Ordinary Space Access (Software Wait Only)

When the wait is specified by software using WCR, the wait input \overline{WAIT} signal from outside is sampled. Figure 8.5 shows the \overline{WAIT} signal sampling. The \overline{WAIT} signal is sampled at the clock rise one cycle before the clock rise when T_w state shifts to T_2 state.

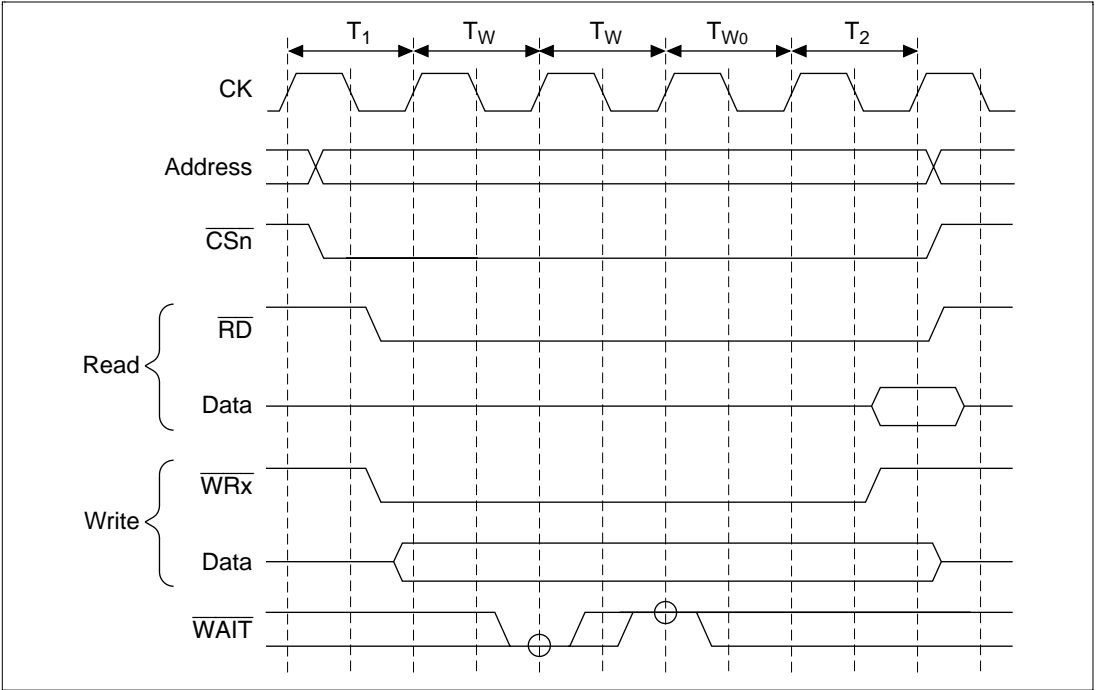


Figure 8.5 Wait State Timing of Ordinary Space Access (Wait States from Software Wait 2 State + $\overline{\text{WAIT}}$ Signal)

8.3.3 $\overline{\text{CS}}$ Assert Period Extension

Idle cycles can be inserted to prevent extension of the $\overline{\text{RD}}$ signal or $\overline{\text{WRx}}$ signal assert period beyond the length of the $\overline{\text{CSn}}$ signal assert period by setting the SW3–SW0 bits of BCR2. This allows for flexible interfaces with external circuitry. The timing is shown in figure 8.6. T_h and T_f cycles are added respectively before and after the ordinary cycle. Only $\overline{\text{CSn}}$ is asserted in these cycles; $\overline{\text{RD}}$ and $\overline{\text{WRx}}$ signals are not. Further, data is extended up to the T_f cycle, which is effective for gate arrays and the like, which have slower write operations.

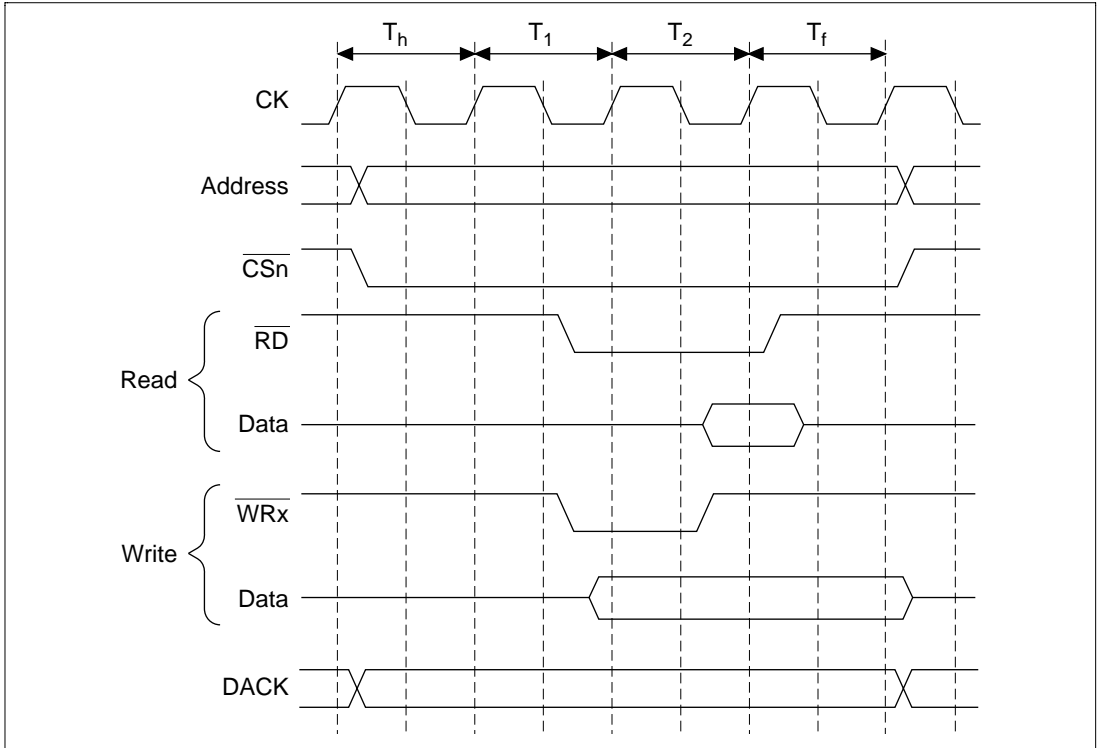


Figure 8.6 $\overline{\text{CS}}$ Assert Period Extension Function

8.4 DRAM Access

8.4.1 DRAM Direct Connection

When address space A31–A24 = H'01 has been accessed, the corresponding space becomes a 16-Mbyte DRAM space, and the DRAM interface function can be used to directly connect this LSI to DRAM.

Row address and column address are always multiplexed for DRAM space. The amount of row address multiplexing can be selected as from 9 to 12 bits by setting the AMX1 and AMX0 bits of the DCR.

Table 8.4 AMX Bits and Address Multiplex Output

| AMX1 | AMX0 | Shift Amount | Row Address | | Column Address | |
|------|------|--------------|----------------|-------------|----------------|-------------|
| | | | Output Address | Output Pins | Output Address | Output Pins |
| 0 | 0 | 9 bits | A21–A15 | A21–A15 | A21–A0 | A21–A0 |
| | | | A14–A0 | A23–A9 | | |
| 0 | 1 | 10 bits | A21–A14 | A21–A14 | A21–A0 | A21–A0 |
| | | | A13–A0 | A23–A10 | | |
| 1 | 0 | 11 bits | A21–A13 | A21–A13 | A21–A0 | A21–A0 |
| | | | A12–A0 | A23–A11 | | |
| 1 | 1 | 12 bits | A21–A12 | A21–A12 | A21–A0 | A21–A0 |
| | | | A11–A0 | A23–A12 | | |

In addition to ordinary read and write accesses, burst mode access using high speed page mode is supported.

8.4.2 Basic Timing

This LSI supports 2 CAS format DRAM access. The DRAM access basic timing is a minimum of 3 cycles for normal mode. Figure 8.7 shows the basic DRAM access timing. DRAM space access is controlled by $\overline{\text{RAS}}$, $\overline{\text{CASx}}$ and RDWR signals. The following signals are associated with transfer of these actual byte locations: $\overline{\text{CASH}}$ (bits 15–8) and $\overline{\text{CASL}}$ (bits 7–0). However, the signals for ordinary space, $\overline{\text{WRx}}$ and $\overline{\text{RD}}$, are also output during the DMAC single transfer column address cycle period. T_p is the precharge cycle, T_r is the RAS assert cycle, T_{c1} is the CAS assert cycle and T_{c2} is the read data fetch cycle.

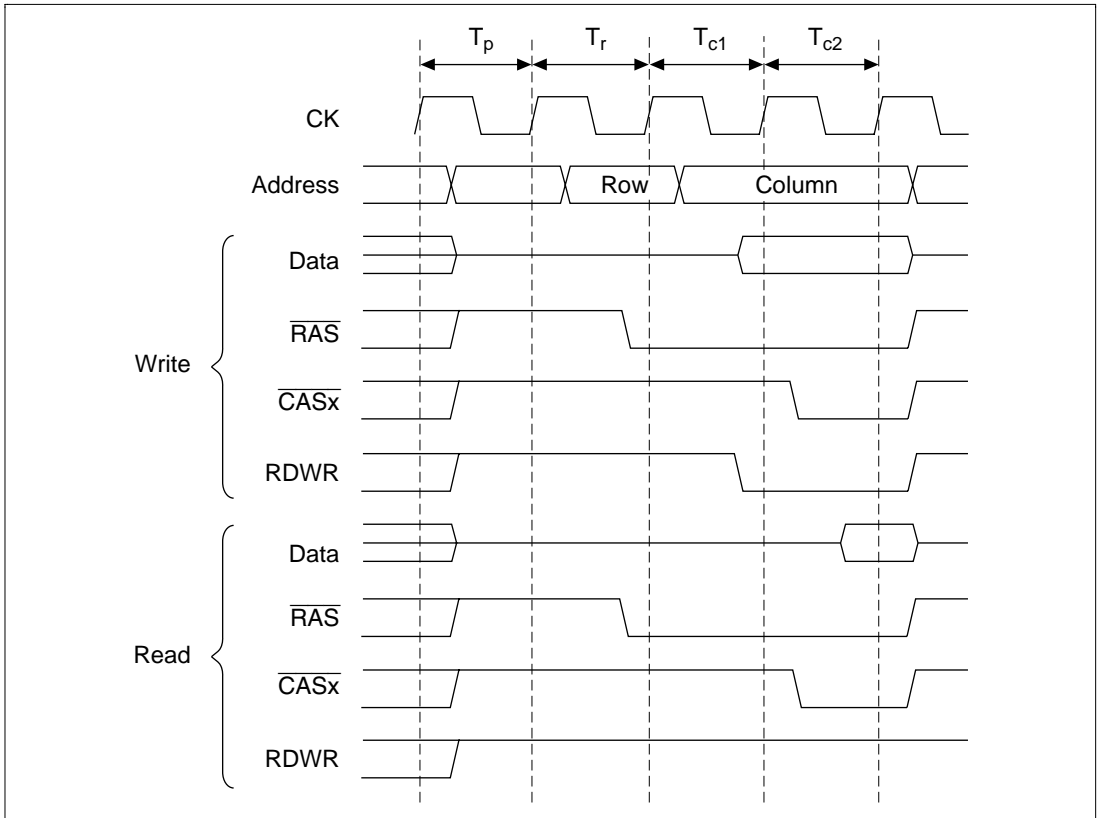


Figure 8.7 DRAM Bus Cycle (Normal Mode, TPC = 0, RCD = 0, No Waits)

8.4.3 Wait State Control

Wait state insertion during DRAM space access is controlled by setting the TPC, RCD, DWW1, DWW0, DWR1, and DWR0 bits of the DCR. TPC and RCD are common to both reads and writes. The timing with waits inserted is shown in figures 8.8 through 8.11. External waits can be inserted at the time of software waits 2, 3. The sampling location is the same as that of ordinary space: at one cycle before the T_{c2} cycle clock rise. Wait cycles are extended by external waits.

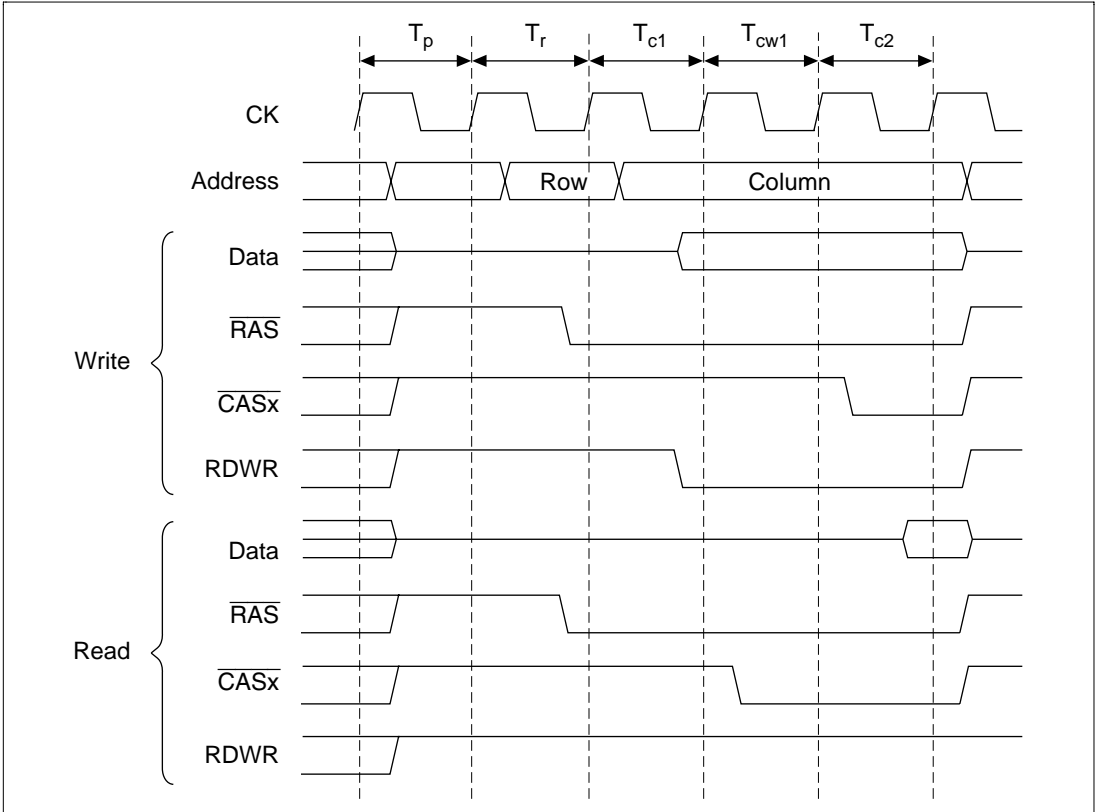


Figure 8.8 DRAM Bus Cycle (Normal Mode, TPC = 0, RCD = 0, One Wait)

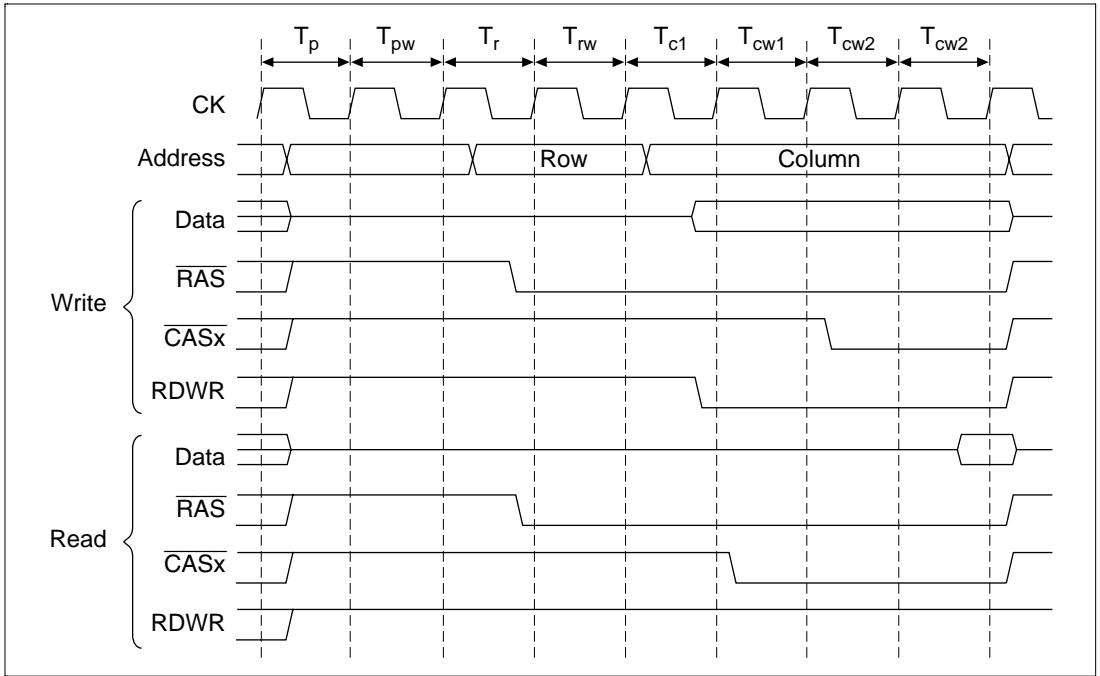


Figure 8.9 DRAM Bus Cycle (Normal Mode, TPC = 1, RCD = 1, Two Waits)

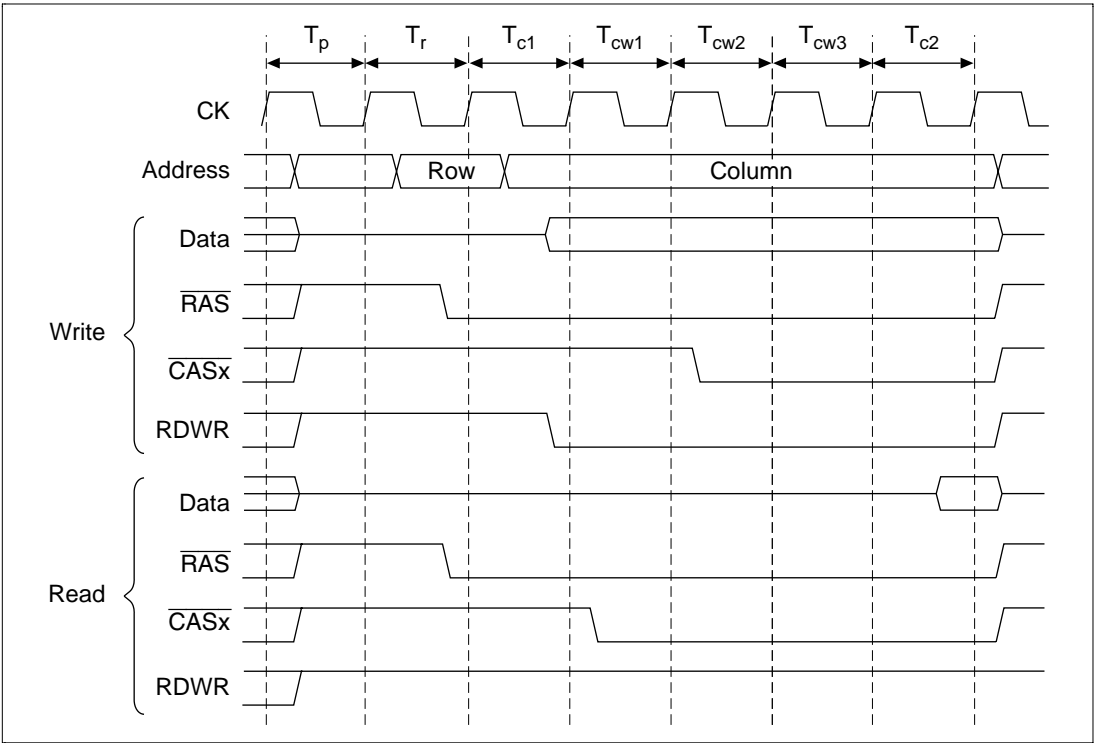


Figure 8.10 DRAM Bus Cycle (Normal Mode, TPC = 0, RCD = 0, Three Waits)

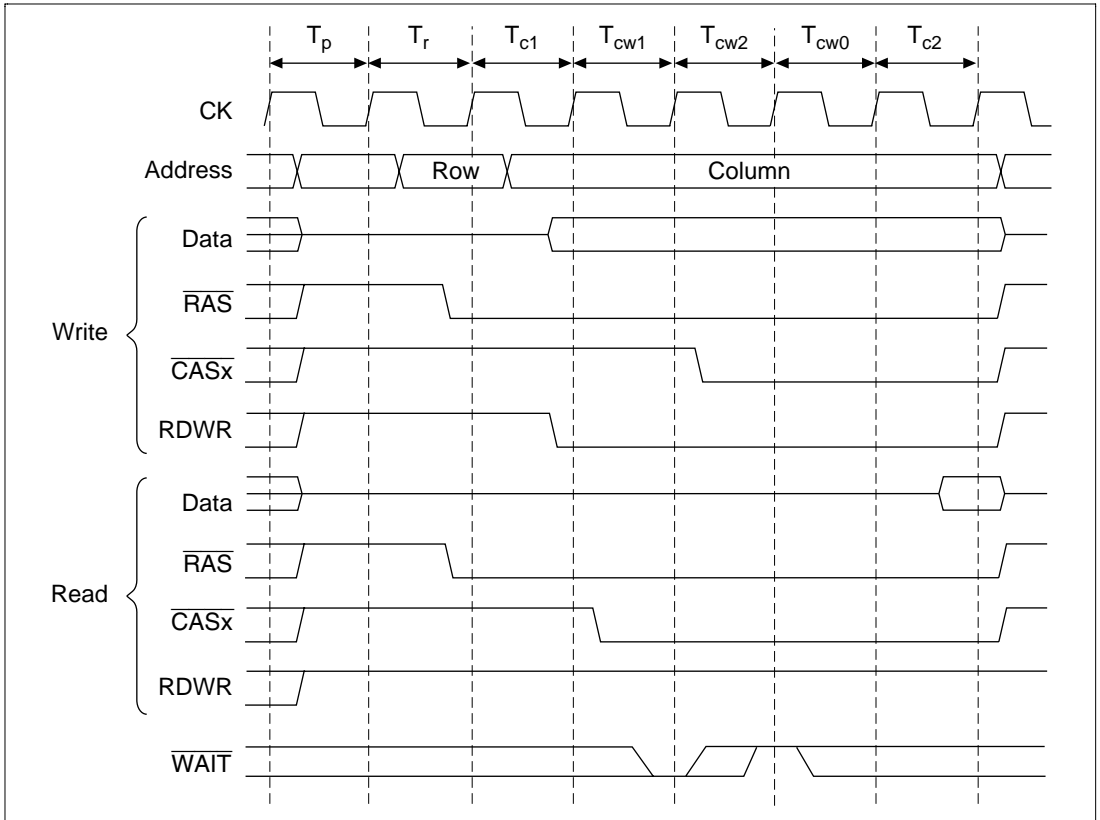


Figure 8.11 DRAM Bus Cycle (Normal Mode, TPC = 0, RCD= 0, Two Waits + Wait Due to WAIT Signal)

8.4.4 Burst Operation

High-Speed Page Mode: When the burst enable bit (BE) of the DCR is set, burst accesses can be performed using high speed page mode. The timing is shown in figure 8.12. Wait cycles can be inserted during burst accesses by using the DCR.

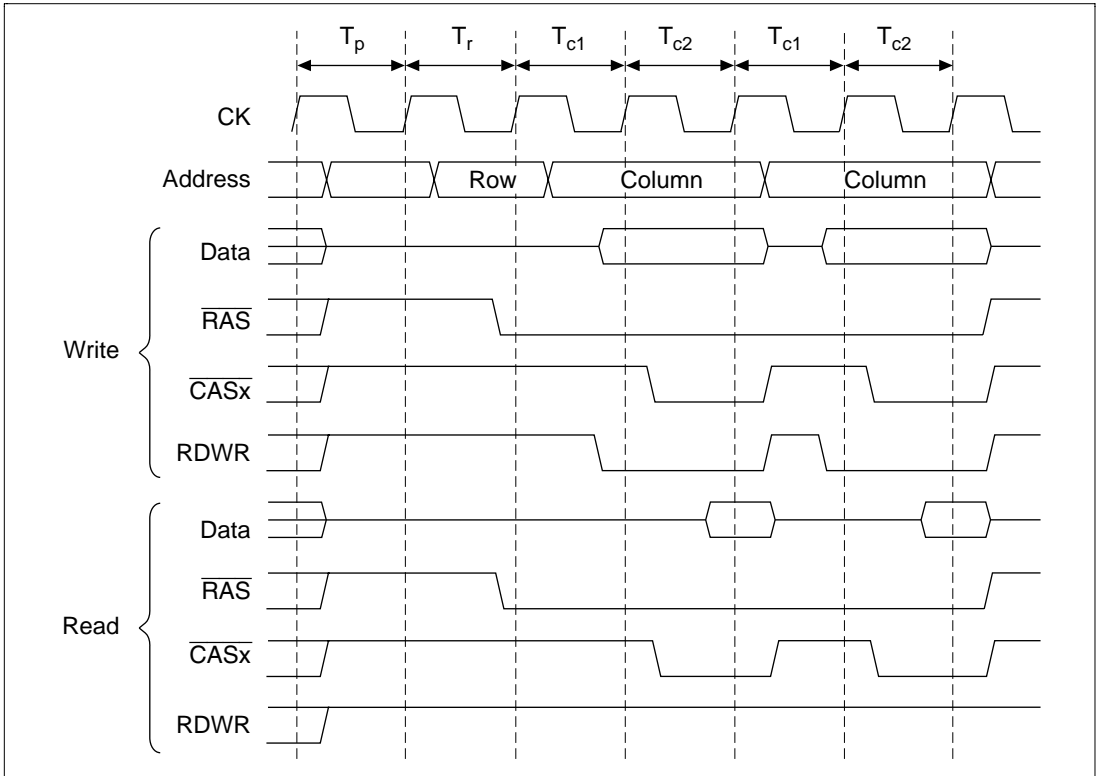


Figure 8.12 DRAM Bus Cycle (High-Speed Page Mode)

RAS Down Mode: There are some instances where even if burst operation is selected, continuous accesses to DRAM will not occur, but another space will be accessed instead part way through the access. In such cases, if the $\overline{\text{RAS}}$ signal is maintained at low level during the time the other space is accessed, it is possible to continue burst operation at the time the next DRAM same row address is accessed. This is called RAS down mode.

To use RAS down mode, set both the BE and RASD bits of the DCR to 1.

Figures 8.13 and 8.14 show operation in RAS up and down modes.

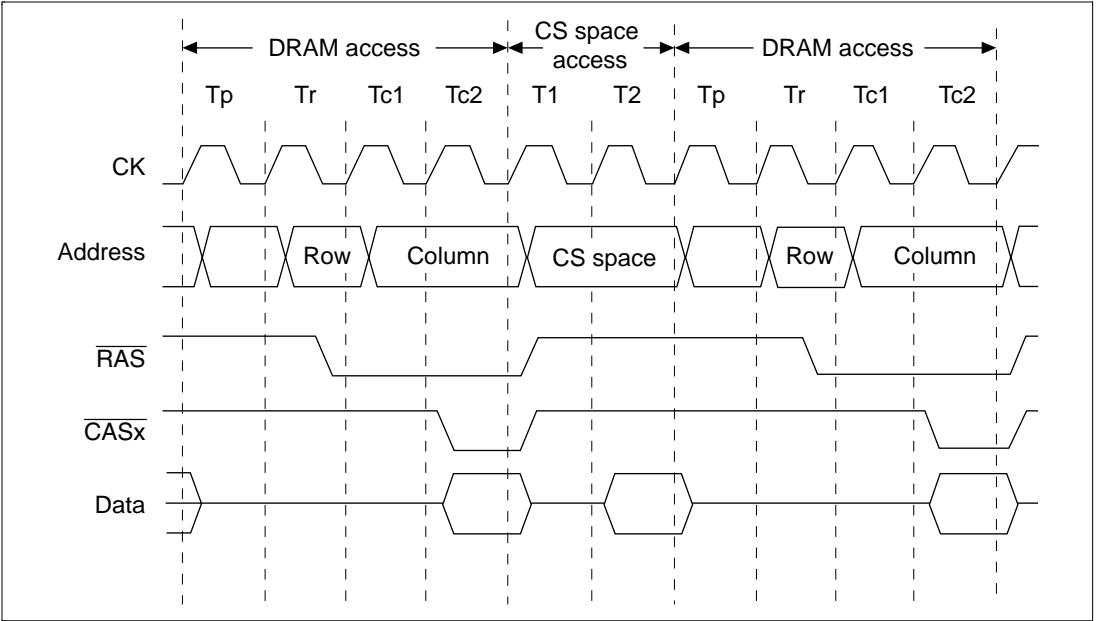


Figure 8.13 DRAM Access Normal Operation (RAS Up Mode)

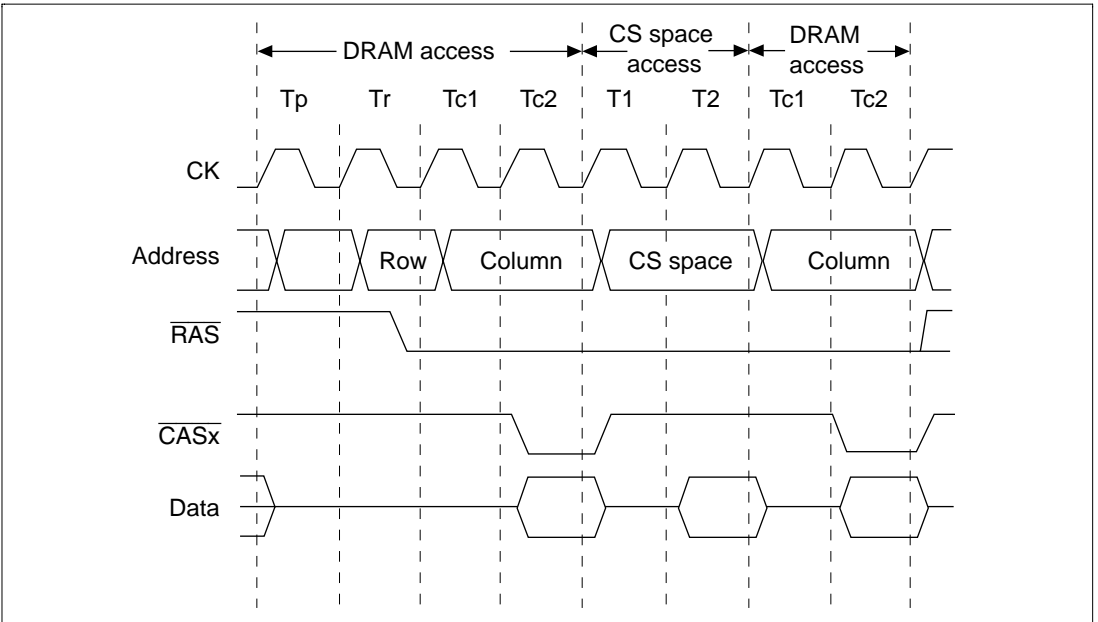


Figure 8.14 RAS Down Mode

8.4.5 Refresh Timing

The bus state controller is equipped with a function to control refreshes of DRAM. CAS-before-RAS (CBR) refresh or self-refresh can be selected by setting the RTCSR's RMD bit.

CAS-before-RAS Refresh: For CBR refreshes, set the RCR's RMD bit to 0 and the RFSH bit to 1. Also write the values in RTCNT and RTCOR necessary to fulfill the refresh interval prescribed for the DRAM being used. When a clock is selected with the CKS2–CKS0 bits of the RSTCR, RTCNT starts counting up from the value at that time. The RTCNT value is constantly compared to the RTCOR value and a CBR refresh is performed when the two match. RTCNT is cleared at that time and the count starts again. Figure 8.15 shows the timing for the CBR refresh operation.

The number of RAS assert cycles in the refresh cycle is set by the TRAS1, TRAS0 bits of the DCR.

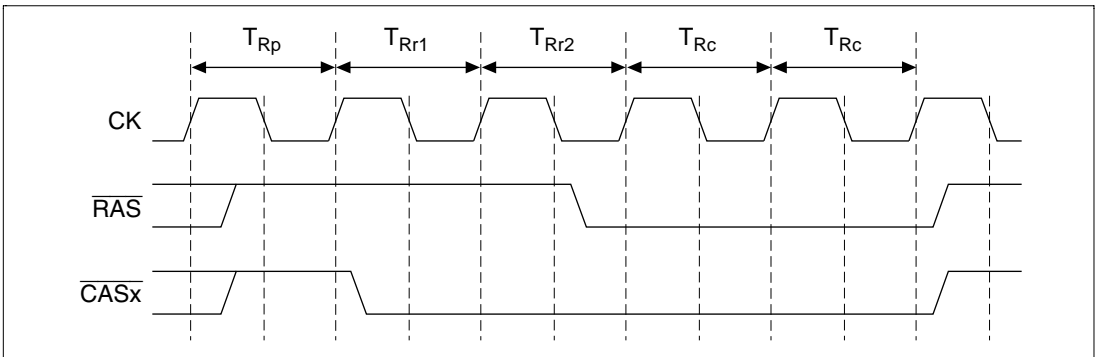


Figure 8.15 CAS-Before-RAS Refresh Timing (TRAS1, TRAS0 = 0, 0)

Self-Refresh: When both the RMD and RFSH bits of the RTCSR are set to 1, the $\overline{\text{CAS}}$ signal and $\overline{\text{RAS}}$ signal are output and the DRAM enters self-refresh mode, as shown in figure 8.16. Do not access DRAM during self-refreshes, in order to preserve DRAM data. When performing DRAM accesses, first cancel the self-refresh, then access only after doing individual refreshes for all row addresses within the time prescribed for the particular DRAM.

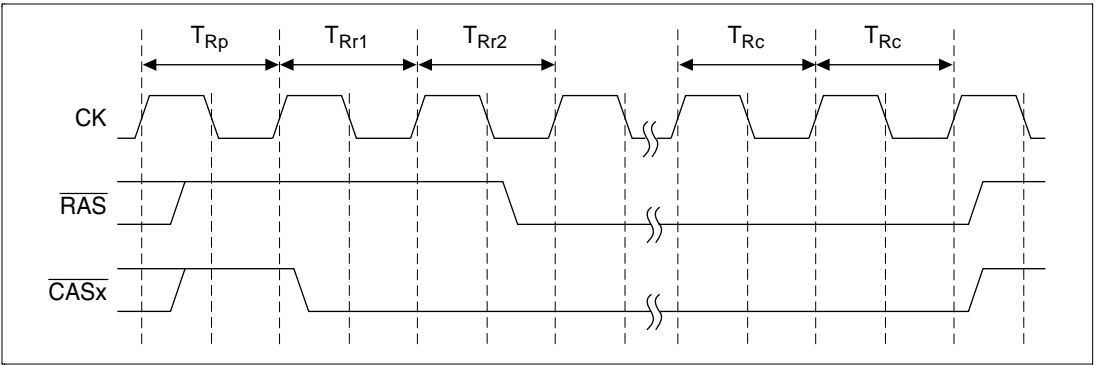


Figure 8.16 Self-Refresh Timing

8.5 Address/Data Multiplex I/O Space Access

When the BCR1 register IOE bit is set to 1, the D15 to D0 pins can be used for multiplexed address/data I/O for the CS3 space. Consequently, peripheral LSIs requiring address/data multiplexing can be directly connected to this LSI.

Address/data multiplex I/O space bus width is selected by the A14 bit, and is 8 bit when A14 = 0 and 16 bit when A14 = 1.

8.5.1 Basic Timing

When the IOE bit of the BCR1 is set to 1, CS3 space becomes address/data multiplex I/O space. When this space is accessed, addresses and data are multiplexed. When the A14 address bit is 0, the bus size becomes 8 bit and addresses and data are input and output through the D7 to D0 pins. When the A14 address bit is 1, the bus size becomes 16 bit and address output and data I/O occur through the D15 to D0 pins. Access for the address/data multiplex I/O space is controlled by the $\overline{\text{AH}}$, $\overline{\text{RD}}$ and $\overline{\text{WRx}}$ signals.

Address/data multiplex I/O space accesses are done after a 3-cycle (fixed) address output, as an ordinary space type access (figure 8.17).

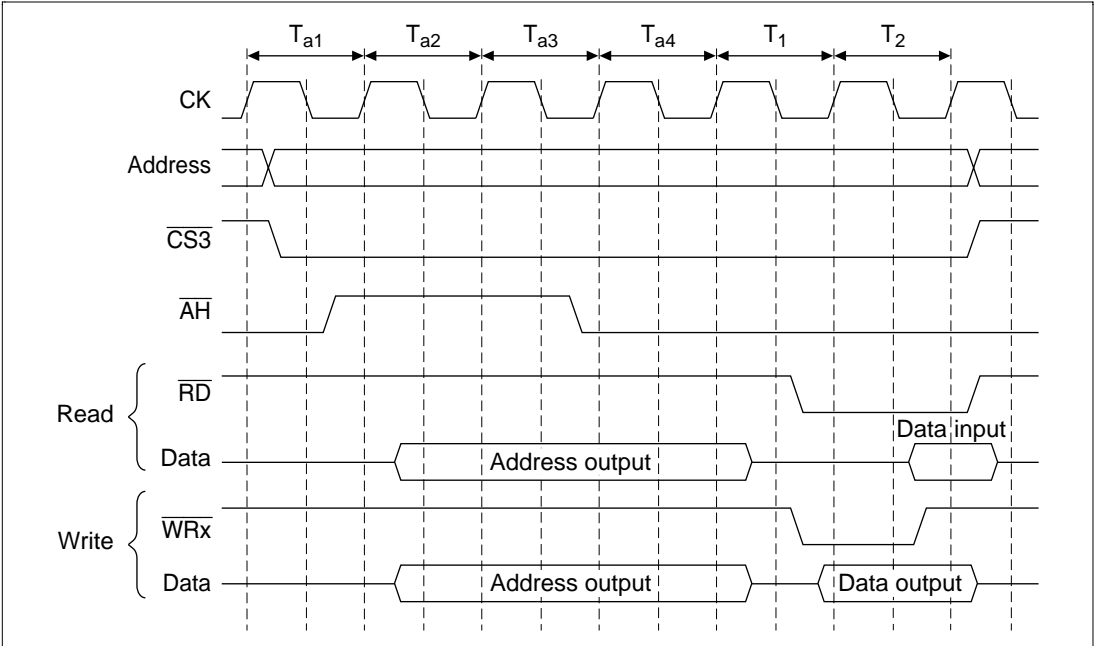


Figure 8.17 Address/Data Multiplex I/O Space Access Timing (No Waits)

8.5.2 Wait State Control

Setting the WCR controls waits during address/data multiplex I/O space accesses. Software wait and external wait insertion timing is the same as during ordinary space accesses. The timing for one software wait + one external wait inserted is shown in figure 8.18.

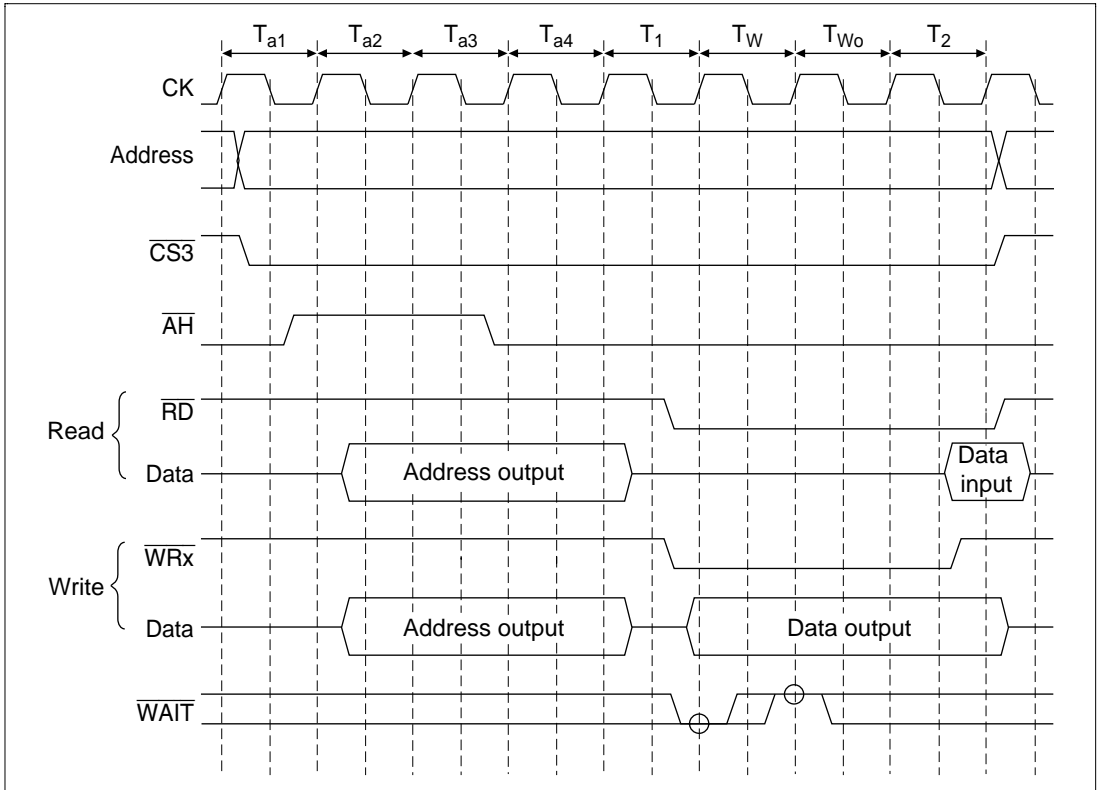


Figure 8.18 Address/Data Multiplex I/O Space Access Wait State Timing (One Software Wait + One External Wait)

8.5.3 CS Assertion Extension

The timing diagram when CS assertion extension is set for address/data multiplexed I/O space access is shown in figure 8.19.

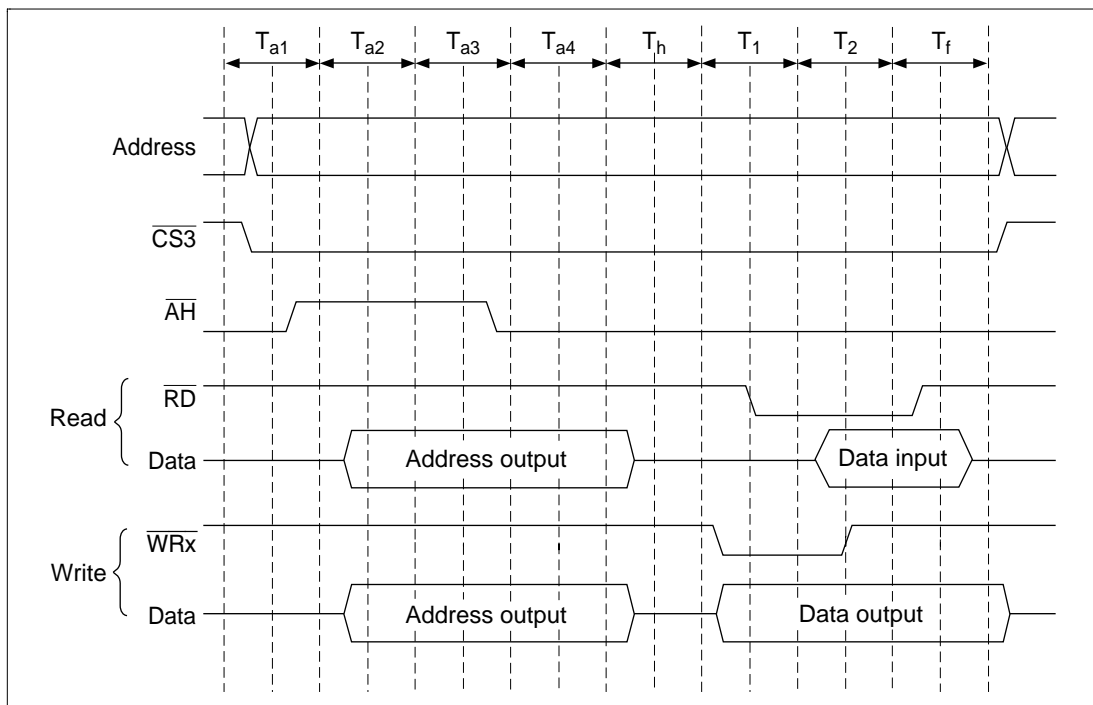


Figure 8.19 Wait Timing for Address/Data Multiplexed I/O Space When CS Assertion Extension is Set

8.6 Waits between Access Cycles

When a read from a slow device is completed, data buffers may not go off in time to prevent data conflicts with the next access. If there is a data conflict during memory access, the problem can be solved by inserting a wait in the access cycle.

To enable detection of bus cycle starts, waits can be inserted between access cycles during continuous accesses of the same CS space by negating the CSn signal once.

8.6.1 Prevention of Data Bus Conflicts

For the two cases of write cycles after read cycles, and read cycles for a different area after read cycles, waits are inserted so that the number of idle cycles specified by the IW31 to IW00 bits of the BCR2 and the DIW of the DCR occur. When idle cycles already exist between access cycles, only the number of empty cycles remaining beyond the specified number of idle cycles are inserted.

Figure 8.20 shows an example of idles between cycles. In this example, 1 idle between CSn space cycles has been specified, so when a CSm space write immediately follows a CSn space read cycle, 1 idle cycle is inserted.

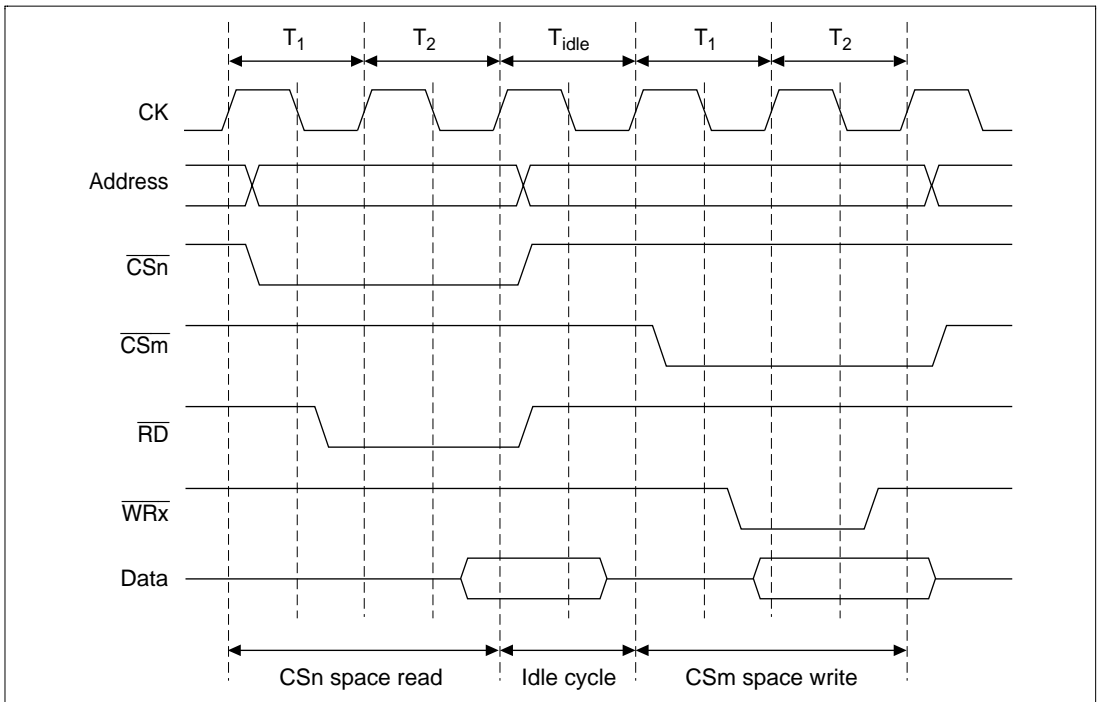


Figure 8.20 Idle Cycle Insertion Example

IW31 and IW30 specify the number of idle cycles required after a CS3 space read either to read other external spaces, or for this LSI, to do write accesses. In the same manner, IW21 and IW20 specify the number of idle cycles after a CS2 space read, IW11 and IW10, the number after a CS1 space read, and IW01 and IW00, the number after a CS0 space read.

DIW specifies the number of idle cycles required, after a DRAM space read either to read other external spaces (CS space), or for this LSI, to do write accesses.

0 to 3 cycles can be specified for CS space, and 0 to 1 cycle for DRAM space.

8.6.2 Simplification of Bus Cycle Start Detection

For consecutive accesses of the same CS space, waits are inserted so that the number of idle cycles designated by the CW3 to CW0 bits of the BCR2 occur. However, for write cycles after reads, the number of idle cycles inserted will be the larger of the two values defined by the IW and CW bits. When idle cycles already exist between access cycles, waits are not inserted. Figure 8.21 shows an example. A continuous access idle is specified for CSn space, and CSn space is consecutively write accessed.

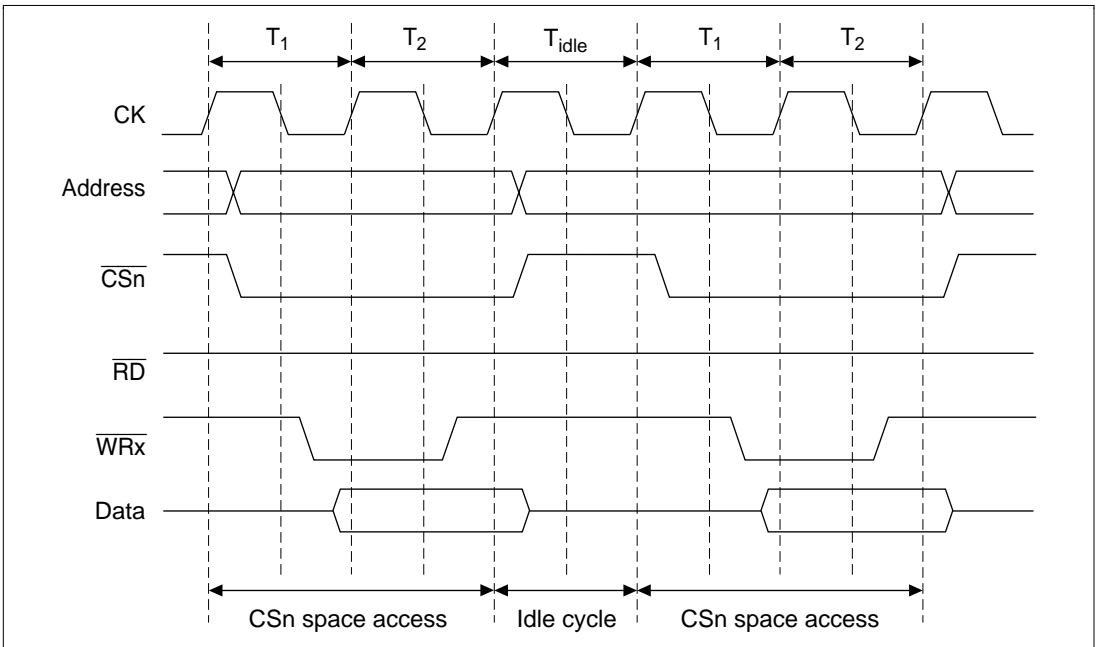


Figure 8.21 Same Space Consecutive Access Idle Cycle Insertion Example

8.7 Bus Arbitration

It has two internal bus masters, the CPU and the DMAC. The priority ranking for determining bus right transfer between these bus masters is:

Refresh > DMAC > CPU

8.8 Memory Connection Examples

Figures 8.22–8.27 show examples of the memory connections.

As A21–A18 become input ports in power-on reset, they should be handled (e.g. pulled down) as necessary.

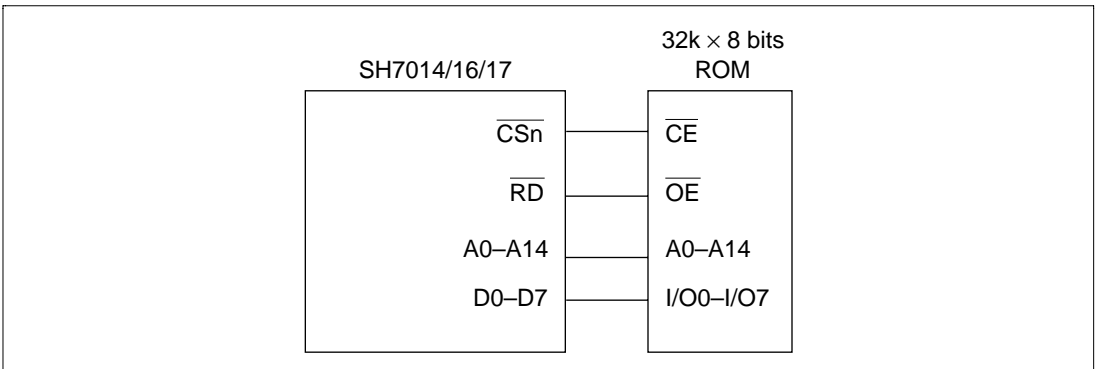


Figure 8.22 8-Bit Data Bus Width ROM Connection

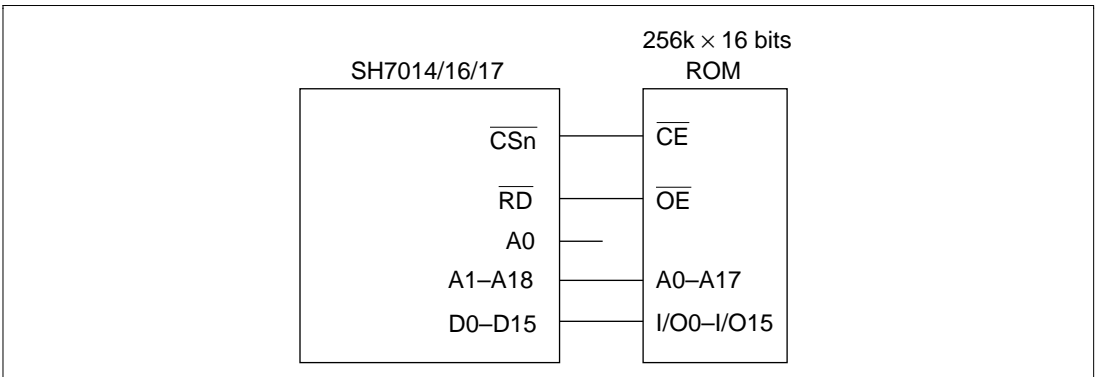


Figure 8.23 16-Bit Data Bus Width ROM Connection

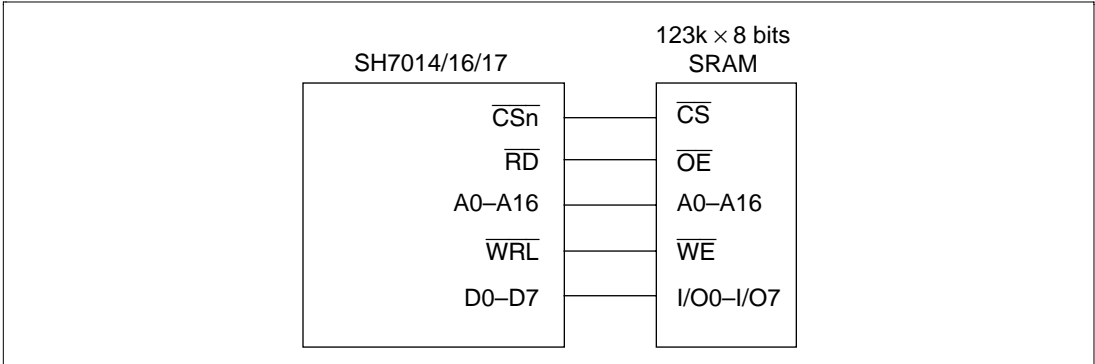


Figure 8.24 8-Bit Data Bus Width SRAM Connection

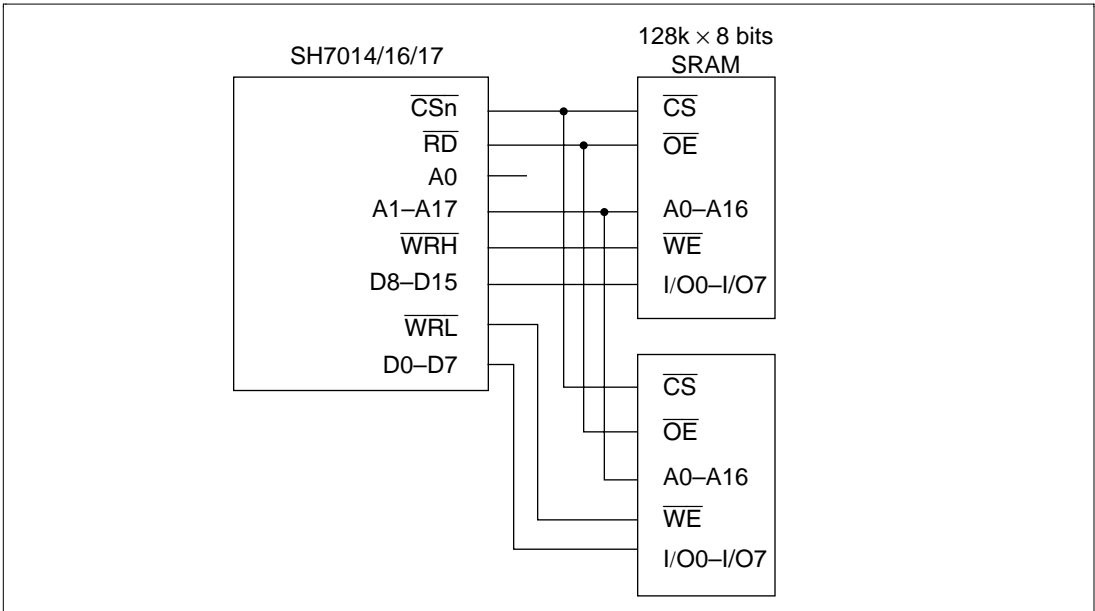


Figure 8.25 16-Bit Data Bus Width SRAM Connection

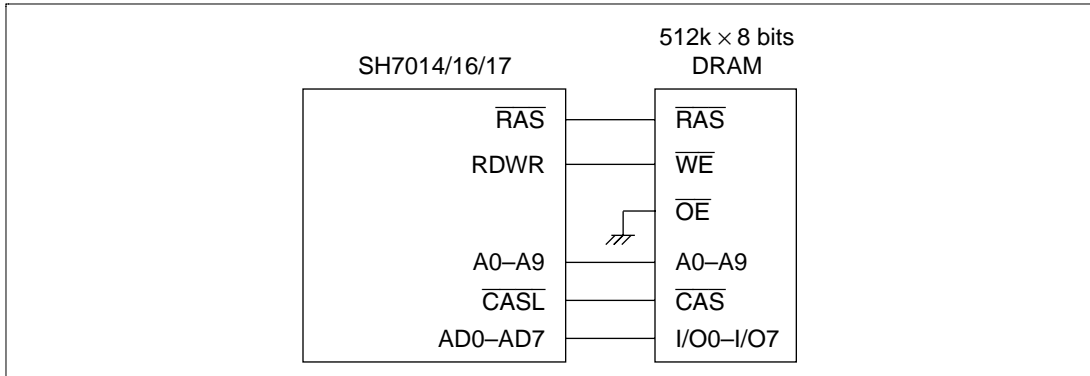


Figure 8.26 8-Bit Data Bus Width DRAM Connection

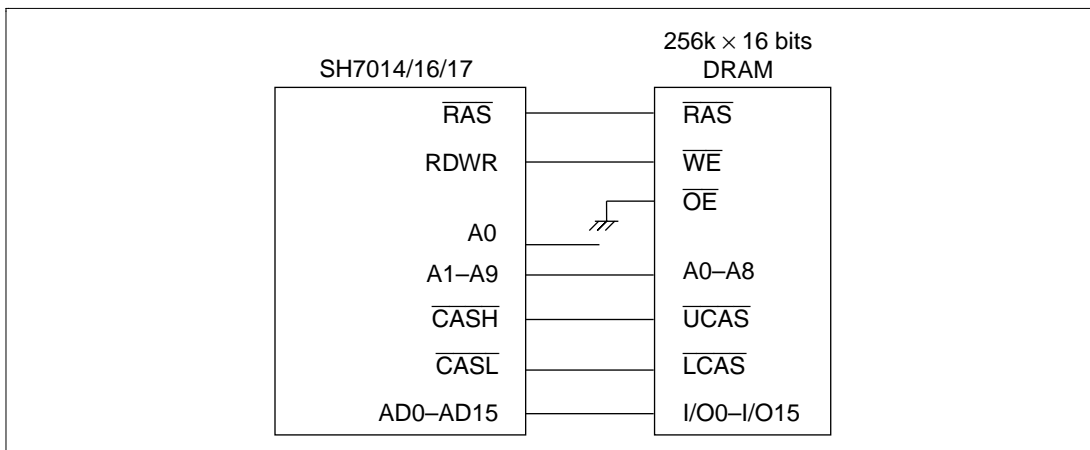


Figure 8.27 16-Bit Data Bus Width DRAM Connection

8.9 On-chip Peripheral I/O Register Access

On-chip peripheral I/O registers are accessed from the bus state controller, as shown in Table 8.5.

Table 8.5 On-chip Peripheral I/O Register Access

| On-chip Peripheral Module | SCI | MTU | INTC | PFC, PORT | CMT | A/D* | WDT | DMAC | CACHE |
|---------------------------|------|-------|-------|-----------|-------|-------|-------|-------|-------|
| Connected bus width | 8bit | 16bit | 16bit | 16bit | 16bit | 16bit | 16bit | 16bit | 16bit |
| Access cycle | 2cyc | 2cyc | 2cyc | 2cyc | 2cyc | 2cyc | 3cyc | 3cyc | 3cyc |

Note: The A/D in the SH7016 and SH7017 has an 8-bit width and is accessed in 3 cyc.

8.10 CPU Operation when Program Is in External Memory

In the SH7014, SH7016, and SH7017 F-ZTAT™, two words (equivalent to two instructions) are normally fetched in a single instruction fetch. This is also true when the program is located in external memory, irrespective of whether the external memory bus width is 8 or 16 bits.

If the program counter value immediately after the program branches is an odd-word ($2n + 1$) address, or if the program counter value immediately before the program branches is an even-word ($2n$) address, the CPU will always fetch 32 bits (equivalent to two instructions) that include the respective word instruction.

Section 9 Direct Memory Access Controller (DMAC)

9.1 Overview

The SH7014 includes an on-chip two-channel direct memory access controller (DMAC). The DMAC can be used in place of the CPU to perform high-speed data transfers among external devices equipped with DACK (transfer request acknowledge signal), external memories, memory-mapped external devices, and on-chip peripheral modules (except for the DMAC and BSC). Using the DMAC reduces the burden on the CPU and increases operating efficiency of the LSI as a whole.

9.1.1 Features

The DMAC has the following features:

- Two channels
- Four Gbytes of address space in the architecture
- Byte, word, or longword selectable data transfer unit
- 64 k (65,536) transfers
- Single or dual address mode.
 - Single address mode: Either the transfer source or transfer destination (peripheral device) is accessed by a DACK signal while the other is accessed by address. One transfer unit of data is transferred in each bus cycle.
 - Dual address mode: Both the transfer source and transfer destination are accessed by address. Values set in a DMAC internal register indicate the accessed address for both the transfer source and transfer destination. Two bus cycles are required for one data transfer.
- Channel function: Dual address mode and single address mode external requests can be accepted.
- Transfer requests: There are three DMAC transfer activation requests, as indicated below.
 - External request: From two DREQ pins. DREQ can be detected either by falling edge or by low level.
 - Requests from on-chip peripheral modules: Transfer requests from on-chip modules such as SCI or A/D. These can be received by all channels.
 - Auto-request: The transfer request is generated automatically within the DMAC.
- Selectable bus modes: Cycle-steal mode or burst mode
- The DMAC priority order is fixed at $0 > 1$.

9.1.2 Block Diagram

Figure 9.1 is a block diagram of the DMAC.

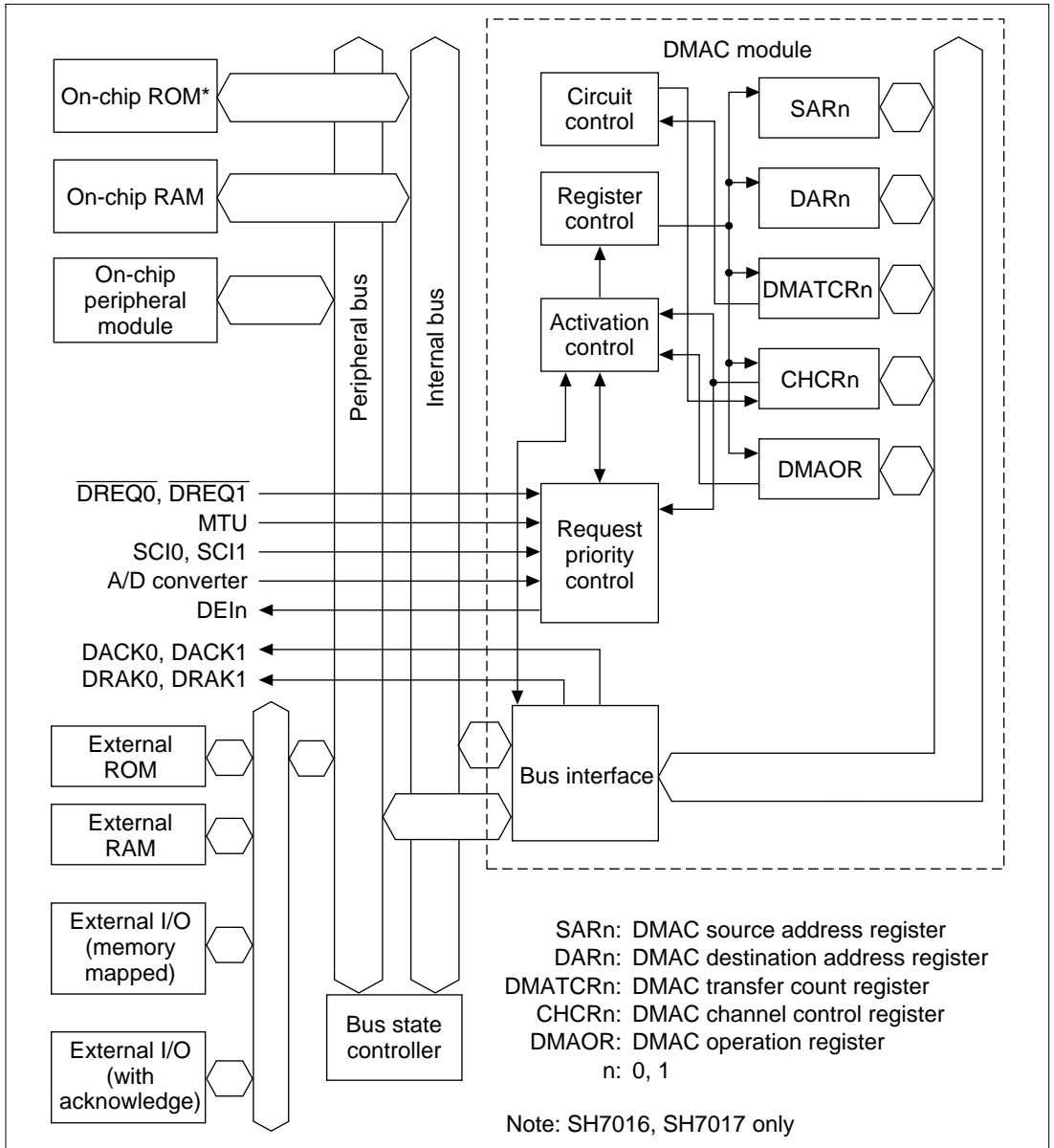


Figure 9.1 DMAC Block Diagram

9.1.3 Pin Configuration

Table 9.1 shows the DMAC pins.

Table 9.1 DMAC Pin Configuration

| Channel | Name | Symbol | I/O | Function |
|---------|---|---------------------------|-----|---|
| 0 | DMA transfer request | $\overline{\text{DREQ0}}$ | I | DMA transfer request input from external device to channel 0 |
| | DMA transfer request acknowledge | DACK0 | O | DMA transfer strobe output from channel 0 to external device |
| | $\overline{\text{DREQ0}}$ acceptance confirmation | DRAK0 | O | Sampling receive acknowledge output for DMA transfer request input from external source |
| 1 | DMA transfer request | $\overline{\text{DREQ1}}$ | I | DMA transfer request input from external device to channel 1 |
| | DMA transfer request acknowledge | DACK1 | O | DMA transfer strobe output from channel 1 to external device |
| | $\overline{\text{DREQ1}}$ acceptance confirmation | DRAK1 | O | Sampling receive acknowledge output for DMA transfer request input from external source |

9.1.4 Register Configuration

Table 9.2 summarizes the DMAC registers. DMAC has a total of 9 registers. Each channel has four control registers. One other control register is shared by all channels

Table 9.2 DMAC Registers

| Channel | Name | Abbreviation | R/W | Initial Value | Address | Register Size | Access Size |
|---------|------------------------------------|--------------|-------------------|---------------|------------|---------------|----------------------|
| 0 | DMA source address register 0 | SAR0 | R/W | Undefined | H'FFFF86C0 | 32 bit | 16, 32* ² |
| | DMA destination address register 0 | DAR0 | R/W | Undefined | H'FFFF86C4 | 32 bit | 16, 32* ² |
| | DMA transfer count register 0 | DMATCR0 | R/W | Undefined | H'FFFF86C8 | 32 bit | 16, 32* ³ |
| | DMA channel control register 0 | CHCR0 | R/W* ¹ | H'00000000 | H'FFFF86CC | 32 bit | 16, 32* ² |
| 1 | DMA source address register 1 | SAR1 | R/W | Undefined | H'FFFF86D0 | 32 bit | 16, 32* ² |
| | DMA destination address register 1 | DAR1 | R/W | Undefined | H'FFFF86D4 | 32 bit | 16, 32* ² |
| | DMA transfer count register 1 | DMATCR1 | R/W | Undefined | H'FFFF86D8 | 32 bit | 16, 32* ³ |
| | DMA channel control register 1 | CHCR1 | R/W* ¹ | H'00000000 | H'FFFF86DC | 32 bit | 16, 32* ² |
| Shared | DMA operation register | DMAOR | R/W* ¹ | H'0000 | H'FFFF86B0 | 16 bit | 16* ⁴ |

- Notes:
1. Write 0 after reading 1 in bit 1 of CHCR0, CHCR1 and in bits 1 and 2 of the DMAOR to clear flags. No other writes are allowed.
 2. For 16-bit access of SAR0, SAR1, DAR0, DAR1, and CHCR0, CHCR1, the 16-bit value on the side not accessed is held.
 3. DMATCR has a 16-bit configuration: bits 0–15. Writing to the upper 16 bits (bits 16–31) is invalid, and these bits always read 0.
 4. Only 16-bit access for DMAOR.
 5. Do not attempt to access an empty address. If an access is attempted, the system operation is not guaranteed.

9.2 Register Descriptions

9.2.1 DMA Source Address Registers 0, 1 (SAR0, SAR1)

DMA source address registers 0, 1 (SAR0, SAR1) are 32-bit read/write registers that specify the source address of a DMA transfer. These registers have a count function, and during a DMA transfer, they indicate the next source address. In single-address mode, SAR values are ignored when a device with DACK has been specified as the transfer source.

Specify a 16-bit or 32-bit boundary address when doing 16-bit or 32-bit data transfers. Operation cannot be guaranteed on any other addresses.

The initial value after power-on resets or in software standby mode is undefined.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | ... | ... | 2 | 1 | 0 |
| | | | | ... | ... | | | |
| Initial value: | — | — | — | ... | ... | — | — | — |
| R/W: | R/W | R/W | R/W | ... | ... | R/W | R/W | R/W |

9.2.2 DMA Destination Address Registers 0, 1 (DAR0, DAR1)

DMA destination address registers 0, 1 (DAR0, DAR1) are 32-bit read/write registers that specify the destination address of a DMA transfer. These registers have a count function, and during a DMA transfer, they indicate the next destination address. In single-address mode, DAR values are ignored when a device with DACK has been specified as the transfer destination.

Specify a 16-bit or 32-bit boundary address when doing 16-bit or 32-bit data transfers. Operation cannot be guaranteed on any other address. The initial value after power-on resets or in software standby mode, is undefined.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit: | 23 | 22 | 21 | ... | ... | 2 | 1 | 0 |
| | | | | ... | ... | | | |
| Initial value: | — | — | — | ... | ... | — | — | — |
| R/W: | R/W | R/W | R/W | ... | ... | R/W | R/W | R/W |

9.2.3 DMA Transfer Count Registers 0, 1 (DMATCR0, DMATCR1)

DMA transfer count registers 0, 1 (DMATCR0, DMATCR1) are 16-bit read/write registers that specify the transfer count for the channel (byte count, word count, or longword count). Specifying a H'000001 gives a transfer count of 1, while H'000000 gives the maximum setting, 65,536 transfers. The initial value after power-on resets or in software standby mode is undefined.

Always write 0 to the upper 16 bits of a DMATCR.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | — | — | — |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — |
| 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 |
| | | | | | | | | |
| Initial value: | — | — | — | — | — | — | — | — |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

9.2.4 DMA Channel Control Registers 0, 1 (CHCR0, CHCR1)

DMA channel control registers 0, 1 (CHCR0, CHCR1) is a 32-bit read/write register where the operation and transmission of each channel is designated. Bits 31–19 and bit 7 should always read 0. The written value should also be 0. They are initialized to 0 by a power-on reset and in software standby mode.

| | | | | | | | | |
|----------------|----|----|----|----|----|----|----|----|
| Bit: | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |

| | | | | | | | | |
|----------------|----|----|----|----|----|-----|-----|-----|
| Bit: | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| | — | — | — | — | — | RL | AM | AL |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | R | R | R | R | R | R/W | R/W | R/W |

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | DM1 | DM0 | SM1 | SM0 | RS3 | RS2 | RS1 | RS0 |
| 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 |
| | — | DS | TM | TS1 | TS0 | IE | TE | DE |
| 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)* | R/W |

Note: TE bit: Allows only 0 write after reading 1.

- Bit 18—Request Check Level (RL): Selects whether to output DRACK notifying external device of $\overline{\text{DREQ}}$ received, with active high or active low.

| Bit 18: RL | Description |
|------------|---|
| 0 | Output DRACK with active high (initial value) |
| 1 | Output DRACK with active low |

- Bit 17—Acknowledge Mode (AM): In dual address mode, selects whether to output DACK in the data write cycle or data read cycle. In single address mode, DACK is always output irrespective of the setting of this bit.

| Bit 17: AM | Description |
|------------|--|
| 0 | Outputs DACK during read cycle (initial value) |
| 1 | Outputs DACK during write cycle |

- Bit 16—Acknowledge Level (AL): Specifies whether to set DACK (acknowledge) signal output to active high or active low.

| Bit 16: AL | Description |
|------------|------------------------------------|
| 0 | Active high output (initial value) |
| 1 | Active low output |

- Bits 15 and 14—Destination Address Mode 1, 0 (DM1 and DM0): These bits specify increment/decrement of the DMA transfer destination address. These bit specifications are ignored when transferring data from an external device to address space in single address mode.

| Bit 15: DM1 | Bit 14: DM0 | Description |
|-------------|-------------|--|
| 0 | 0 | Destination address fixed (initial value) |
| 0 | 1 | Destination address incremented (+1 during 8-bit transfer, +2 during 16-bit transfer, +4 during 32-bit transfer) |
| 1 | 0 | Destination address decremented (−1 during 8-bit transfer, −2 during 16-bit transfer, −4 during 32-bit transfer) |
| 1 | 1 | Setting prohibited |

- Bits 13 and 12—Source Address Mode 1, 0 (SM1 and SM0): These bits specify increment/decrement of the DMA transfer source address. These bit specifications are ignored when transferring data from an external device to address space in single address mode.

| Bit 13: SM1 | Bit 12: SM0 | Description |
|-------------|-------------|---|
| 0 | 0 | Source address fixed (initial value) |
| 0 | 1 | Source address incremented (+1 during 8-bit transfer, +2 during 16-bit transfer, +4 during 32-bit transfer) |
| 1 | 0 | Source address decremented (−1 during 8-bit transfer, −2 during 16-bit transfer, −4 during 32-bit transfer) |
| 1 | 1 | Setting prohibited |

- Bits 11–8—Resource Select 3–0 (RS3–RS0): These bits specify the transfer request source.

| Bit 11: RS3 | Bit 10: RS2 | Bit 9: RS1 | Bit 8: RS0 | Description |
|----------------|----------------|---------------|---------------|--|
| 0 | 0 | 0 | 0 | External request, dual address mode (initial value) |
| 0 | 0 | 0 | 1 | Prohibited |
| 0 | 0 | 1 | 0 | External request, single address mode. External address space → external device. |
| 0 | 0 | 1 | 1 | External request, single address mode. External device → external address space. |
| 0 | 1 | 0 | 0 | Auto-request |
| 0 | 1 | 0 | 1 | Prohibited |
| 0 | 1 | 1 | 0 | MTU TGI0A |
| 0 | 1 | 1 | 1 | MTU TGI1A |
| 1 | 0 | 0 | 0 | MTU TGI2A |
| 1 | 0 | 0 | 1 | Prohibited |
| 1 | 0 | 1 | 0 | Prohibited |
| 1 | 0 | 1 | 1 | A/D ADI |
| 1 | 1 | 0 | 0 | SCI0 TXI0 |
| 1 | 1 | 0 | 1 | SCI0 RXI0 |
| 1 | 1 | 1 | 0 | SCI1 TXI1 |
| 1 | 1 | 1 | 1 | SCI1 RXI1 |

- Bit 6— $\overline{\text{DREQ}}$ Select (DS): Sets the sampling method for the $\overline{\text{DREQ}}$ pin in external request mode to either low-level detection or falling-edge detection. When specifying an on-chip peripheral module or auto-request as the transfer request source, this bit setting is ignored. The sampling method is fixed at falling-edge detection in cases other than auto-request.

| Bit 6: DS | Description |
|-----------|-------------------------------------|
| 0 | Low-level detection (initial value) |
| 1 | Falling-edge detection |

- Bit 5—Transfer Mode (TM): Specifies the bus mode for data transfer.

| Bit 5: TM | Description |
|-----------|----------------------------------|
| 0 | Cycle steal mode (initial value) |
| 1 | Burst mode |

- Bits 4 and 3—Transfer Size 1, 0 (TS1, TS0): Specifies size of data for transfer.

| Bit 4: TS1 | Bit 3: TS0 | Description |
|------------|------------|--|
| 0 | 0 | Specifies byte size (8 bits) (initial value) |
| 0 | 1 | Specifies word size (16 bits) |
| 1 | 0 | Specifies longword size (32 bits) |
| 1 | 1 | Prohibited |

- Bit 2—Interrupt Enable (IE): When this bit is set to 1, interrupt requests are generated after the number of data transfers specified in the DMATCR (when TE = 1).

| Bit 2: IE | Description |
|-----------|---|
| 0 | Interrupt request not generated after DMATCR-specified transfer count (initial value) |
| 1 | Interrupt request enabled on completion of DMATCR specified number of transfers |

- Bit 1—Transfer End Flag (TE): This bit is set to 1 after the number of data transfers specified by the DMATCR. At this time, if the IE bit is set to 1, an interrupt request is generated. If data transfer ends before TE is set to 1 (for example, due to an NMI or address error, or clearing of the DE bit or DME bit of the DMAOR) the TE is not set to 1. With this bit set to 1, data transfer is disabled even if the DE bit is set to 1.

| Bit 1: TE | Description |
|-----------|---|
| 0 | DMATCR-specified transfer count not ended (initial value) Clear condition: 0 write after TE = 1 read, Power-on reset, standby mode |
| 1 | DMATCR specified number of transfers completed |

- Bit 0—DMAC Enable (DE): DE enables operation in the corresponding channel.

| Bit 0: DE | Description |
|-----------|---|
| 0 | Operation of the corresponding channel disabled (initial value) |
| 1 | Operation of the corresponding channel enabled |

Transfer mode is entered if this bit is set to 1 when auto-request is specified (RS3–RS0 settings). With an external request or on-chip module request, when a transfer request occurs after this bit is set to 1, transfer is enabled. If this bit is cleared during a data transfer, transfer is suspended.

If the DE bit has been set, but TE = 1, then if the DME bit of the DMAOR is 0, and the NMI or AE bit of the DMAOR is 1, transfer enable mode is not entered.

9.2.5 DMAC Operation Register (DMAOR)

The DMAOR is a 16-bit read/write register that specifies the transfer mode of the DMAC. Bits 15–3 of this register always read as 0. The write value should always be 0.

Register values are initialized to 0 during power-on reset or in software standby mode.

| | | | | | | | | |
|----------------|----|----|----|----|----|--------|--------|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | AE | NMIF | DME |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R/(W)* | R/(W)* | R |

Note: 0 write only is valid after 1 is read.

- **Bit 2—Address Error Flag (AE):** Indicates that an address error has occurred during DMA transfer. If this bit is set during a data transfer, transfers on all channels are suspended. The CPU cannot write a 1 to the AE bit. Clearing is effected by 0 write after 1 read.

| Bit 2: AE | Description |
|------------------|---|
| 0 | No address error, DMA transfer enabled (initial value) Clearing condition: Write AE = 0 after reading AE = 1 |
| 1 | Address error, DMA transfer disabled Setting condition: Address error due to DMAC |

- **Bit 1—NMI Flag (NMIF):** Indicates input of an NMI. This bit is set irrespective of whether the DMAC is operating or suspended. If this bit is set during a data transfer, transfers on all channels are suspended. The CPU is unable to write a 1 to the NMIF. Clearing is effected by a 0 write after 1 read.

| Bit 1: NMIF | Description |
|--------------------|---|
| 0 | No NMI interrupt, DMA transfer enabled (initial value) Clearing condition: Write NMIF = 0 after reading NMIF = 1 |
| 1 | NMI has occurred, DMC transfer prohibited Set condition: NMI interrupt occurrence |

- **Bit 0—DMAC Master Enable (DME):** This bit enables activation of the entire DMAC. When the DME bit and DE bit of the CHCR for the corresponding channel are set to 1, that channel is transfer-enabled. If this bit is cleared during a data transfer, transfers on all channels are suspended.

Even when the DME bit is set, when the TE bit of the CHCR is 1, or its DE bit is 0, transfer is disabled in the case of an NMI of the DMAOR or when AE = 1.

| Bit 0: DME | Description |
|-------------------|---|
| 0 | Disable operation on all channels (initial value) |
| 1 | Enable operation on all channels |

9.3 Operation

When there is a DMA transfer request, the DMAC starts the transfer according to the predetermined channel priority order; when the transfer end conditions are satisfied, it ends the transfer. Transfers can be requested in three modes: auto-request, external request, and on-chip peripheral module request. Transfer can be in either the single address mode or the dual address mode. The bus mode can be either burst or cycle steal.

9.3.1 DMA Transfer Flow

After the DMA source address registers (SAR), DMA destination address registers (DAR), DMA transfer count register (DMATCR), DMA channel control registers (CHCR), and DMA operation register (DMAOR) are set to the desired transfer conditions, the DMAC transfers data according to the following procedure:

1. The DMAC checks to see if transfer is enabled ($DE = 1$, $DME = 1$, $TE = 0$, $NMIF = 0$, $AE = 0$).
2. When a transfer request comes and transfer has been enabled, the DMAC transfers 1 transfer unit of data (determined by TS0 and TS1 setting). For an auto-request, the transfer begins automatically when the DE bit and DME bit are set to 1. The DMATCR value will be decremented by 1 upon each transfer. The actual transfer flows vary by address mode and bus mode.
3. When the specified number of transfers have been completed (when DMATCR reaches 0), the transfer ends normally. If the IE bit of the CHCR is set to 1 at this time, a DEI interrupt is sent to the CPU.
4. When an address error occurs in the DMAC or an NMI interrupt is generated, the transfer is aborted. Transfers are also aborted when the DE bit of the CHCR or the DME bit of the DMAOR are changed to 0.

Figure 9.2 is a flowchart of this procedure.

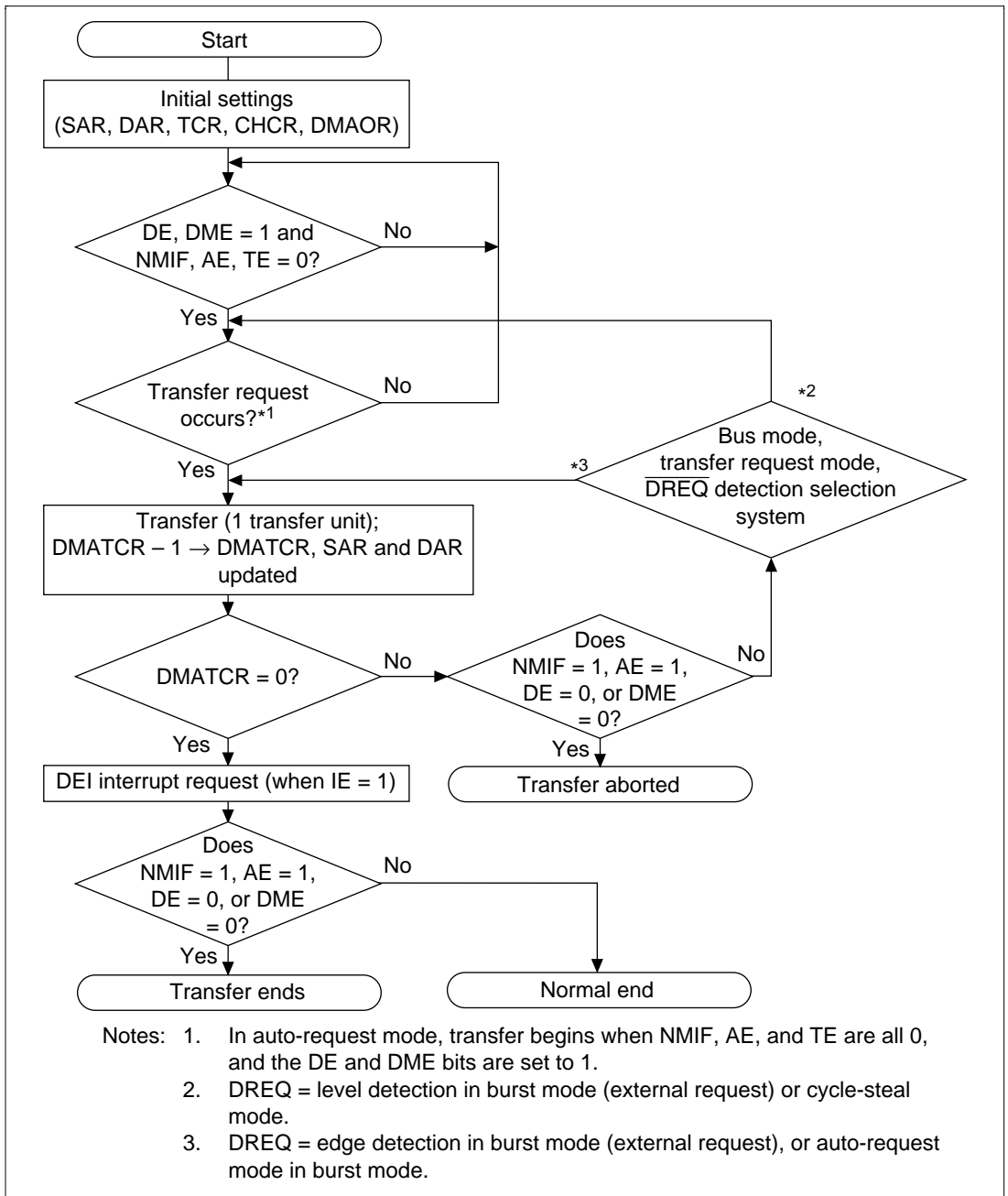


Figure 9.2 DMAC Transfer Flowchart

9.3.2 DMA Transfer Requests

DMA transfer requests are usually generated in either the data transfer source or destination, but they can also be generated by devices and on-chip peripheral modules that are neither the source nor the destination. Transfers can be requested in three modes: auto-request, external request, and on-chip peripheral module request. The request mode is selected in the RS3–RS0 bits of the DMA channel control registers 0, 1 (CHCR0, CHCR1).

Auto-Request Mode: When there is no transfer request signal from an external source, as in a memory-to-memory transfer or a transfer between memory and an on-chip peripheral module unable to request a transfer, the auto-request mode allows the DMAC to automatically generate a transfer request signal internally. When the DE bits of CHCR0, CHCR1 and the DME bit of the DMAOR are set to 1, the transfer begins (so long as the TE bits of CHCR0, CHCR1 and the NMIF and AE bits of DMAOR are all 0).

External Request Mode: In this mode a transfer is performed at the request signal ($\overline{\text{DREQ}}$) of an external device. Choose one of the modes shown in table 9.3 according to the application system. When this mode is selected, if the DMA transfer is enabled ($\text{DE} = 1$, $\text{DME} = 1$, $\text{TE} = 0$, $\text{NMIF} = 0$, $\text{AE} = 0$), a transfer is performed upon a request at the $\overline{\text{DREQ}}$ input. Choose to detect $\overline{\text{DREQ}}$ by either the falling edge or low level of the signal input with the DS bit of CHCR0, CHCR1 ($\text{DS} = 0$ is level detection, $\text{DS} = 1$ is edge detection). The source of the transfer request does not have to be the data transfer source or destination.

Table 9.3 Selecting External Request Modes with the RS Bits

| RS3 | RS2 | RS1 | RS0 | Address Mode | Source | Destination |
|-----|-----|-----|-----|---------------------|--|--|
| 0 | 0 | 0 | 0 | Dual address mode | Any* | Any* |
| 0 | 0 | 1 | 0 | Single address mode | External memory or memory-mapped external device | External device with DACK |
| 0 | 0 | 1 | 1 | Single address mode | External device with DACK | External memory or memory-mapped external device |

Note: External memory, memory-mapped external device, on-chip memory, on-chip peripheral module (excluding DMAC, BSC).

On-Chip Peripheral Module Request Mode: In this mode a transfer is performed at the transfer request signal (interrupt request signal) of an on-chip peripheral module. As indicated in table 9.4, there are eight transfer request signals: three from the multifunction timer pulse unit (MTU), which are compare match or input capture interrupts; the receive data full interrupts (RxI) and transmit data empty interrupts (TxI) of the two serial communication interfaces (SCI); and the A/D conversion end interrupt (ADI) of the A/D converter. When DMA transfers are enabled ($\text{DE} = 1$,

DME = 1, TE = 0, NMIF = 0, AE = 0), a transfer is performed upon the input of a transfer request signal.

The transfer request source need not be the data transfer source or transfer destination. However, when the transfer request is set by RxI (transfer request because SCI's receive data is full), the transfer source must be the SCI's receive data register (RDR). When the transfer request is set by TxI (transfer request because SCI's transmit data is empty), the transfer destination must be the SCI's transmit data register (TDR). Also, if the transfer request is set to the A/D converter, the data transfer destination must be the A/D converter register.

Table 9.4 Selecting On-Chip Peripheral Module Request Modes with the RS Bits

| RS3 | RS2 | RS1 | RS0 | DMAC Transfer Request Source | DMA Transfer Request Signal Source | Destination | Bus Mode | |
|-----|-----|-----|-----|-----------------------------------|------------------------------------|--------------------|-------------------|-------------------|
| 0 | 1 | 1 | 0 | MTU* ² | TGI0A | Any* ¹ | Burst/cycle steal | |
| 0 | 1 | 1 | 1 | MTU* ² | TGI1A | Any* ¹ | Burst/cycle steal | |
| 1 | 0 | 0 | 0 | MTU* ² | TGI2A | Any* ¹ | Burst/cycle steal | |
| 1 | 0 | 0 | 1 | Prohibited | | | | |
| 1 | 0 | 1 | 0 | Prohibited | | | | |
| 1 | 0 | 1 | 1 | A/D | ADI | ADDR* ⁴ | Burst/cycle steal | |
| 1 | 1 | 0 | 0 | SCI0* ³ transmit block | Txl0 | Any* ¹ | TDR0 | Burst/cycle steal |
| 1 | 1 | 0 | 1 | SCI0* ³ receive block | Rxl0 | RDR0 | Any* ¹ | Burst/cycle steal |
| 1 | 1 | 1 | 0 | SCI1* ³ transmit block | Txl1 | Any* ¹ | TDR1 | Burst/cycle steal |
| 1 | 1 | 1 | 1 | SCI1* ³ receive block | Rxl1 | RDR1 | Any* ¹ | Burst/cycle steal |

Notes: 1. External memory, memory-mapped external device, on-chip memory, on-chip peripheral module (excluding DMAC, BSC).

2. MTU: Multifunction timer pulse unit.

3. SCI0, SCI1: Serial communications interface.

4. ADDR: A/D converter's A/D register.

In order to output a transfer request from an on-chip peripheral module, set the relevant interrupt enable bit for each module, and output an interrupt signal.

When an on-chip peripheral module's interrupt request signal is used as a DMA transfer request signal, interrupts for the CPU are not generated.

When a DMA transfer is conducted corresponding with one of the transfer request signals in table 9.4, it is automatically discontinued. In cycle steal mode this occurs in the first transfer, and in burst mode with the last transfer.

9.3.3 Channel Priority

When the DMAC receives simultaneous transfer requests on two or more channels, it selects a channel according to a predetermined priority order. The channel priority order is fixed at channel 0 > channel 1.

9.3.4 DMA Transfer Types

The DMAC supports the transfers shown in table 9.5. It can operate in the single address mode, in which either the transfer source or destination is accessed using an acknowledge signal, or dual address mode, in which both the transfer source and destination addresses are output. The DMAC has two bus modes: cycle-steal mode and burst mode.

Table 9.5 Supported DMA Transfers

| Source | Destination | | | | |
|-------------------------------|---------------------------|-----------------|-------------------------------|----------------|---------------------------|
| | External Device with DACK | External Memory | Memory-Mapped External Device | On-Chip Memory | On-Chip Peripheral Module |
| External device with DACK | Not available | Single | Single | Not available | Not available |
| External memory | Single | Dual | Dual | Dual | Dual |
| Memory-mapped external device | Single | Dual | Dual | Dual | Dual |
| On-chip memory | Not available | Dual | Dual | Dual | Dual |
| On-chip peripheral module | Not available | Dual | Dual | Dual | Dual |

Notes: 1. Single: Single address mode

2. Dual: Dual address mode

9.3.5 Address Modes

Single Address Mode: In the single address mode, both the transfer source and destination are external; one (selectable) is accessed by a DACK signal while the other is accessed by an address. In this mode, the DMAC performs the DMA transfer in 1 bus cycle by simultaneously outputting a transfer request acknowledge DACK signal to one external device to access it while outputting an address to the other end of the transfer. Figure 9.3 shows an example of a transfer between an external memory and an external device with DACK in which the external device outputs data to the data bus while that data is written in external memory in the same bus cycle.

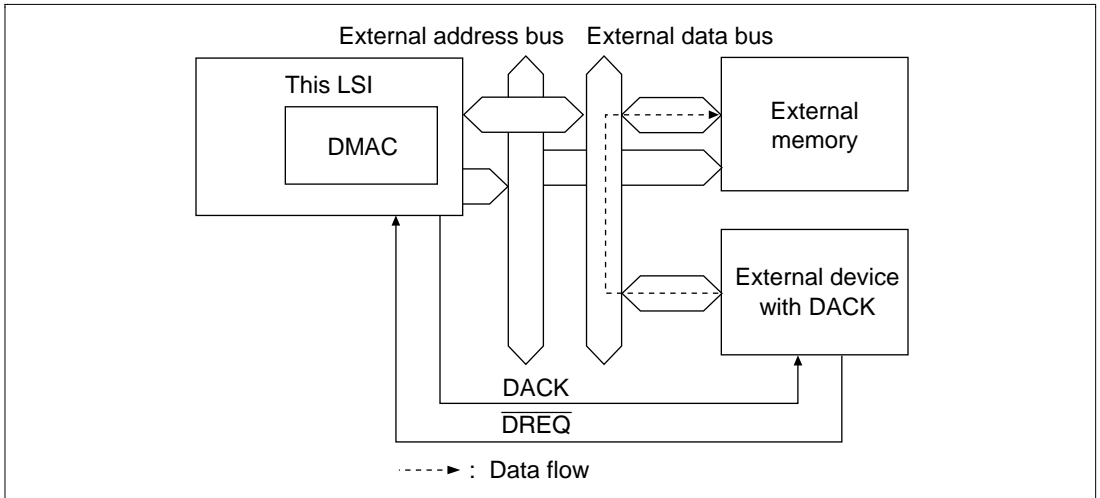
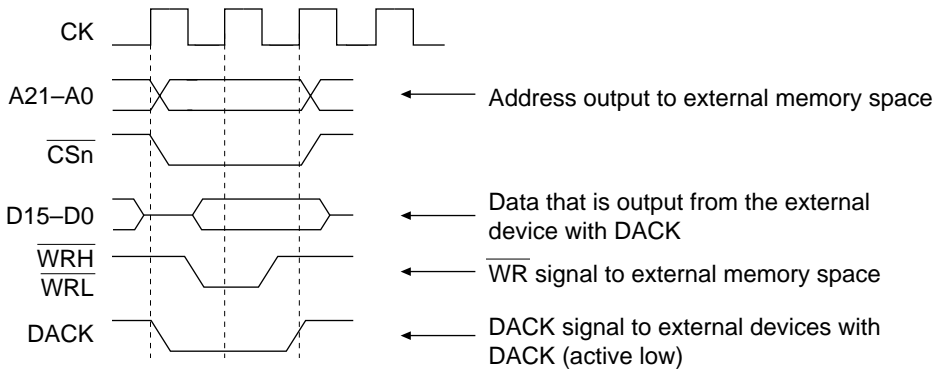
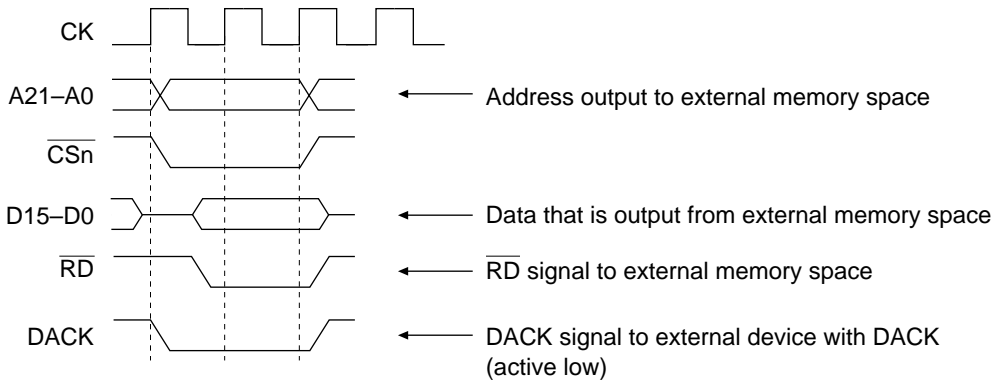


Figure 9.3 Data Flow in Single Address Mode

Two types of transfers are possible in the single address mode: (a) transfers between external devices with DACK and memory-mapped external devices, and (b) transfers between external devices with DACK and external memory. The only transfer requests for either of these is the external request (\overline{DREQ}). Figure 9.4 shows the DMA transfer timing for the single address mode.



a. External device with DACK to external memory space



b. External memory space to external device with DACK

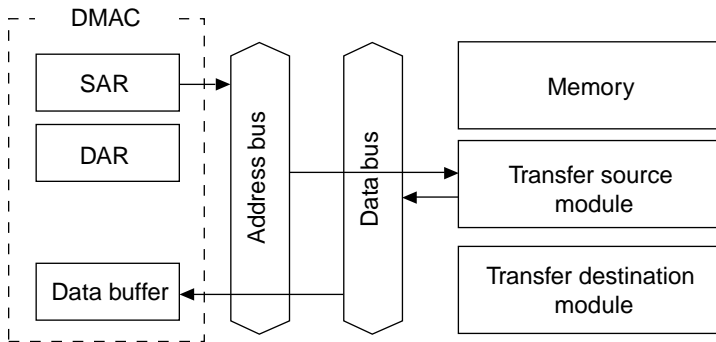
Figure 9.4 Example of DMA Transfer Timing in the Single Address Mode

9.3.6 Dual Address Mode

Dual address mode is used for access of both the transfer source and destination by address. Transfer source and destination can be accessed either internally or externally.

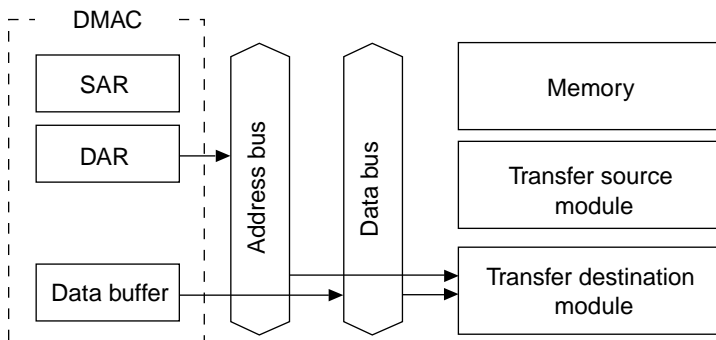
Dual Address Mode: Data is read from the transfer source during the data read cycle, and written to the transfer destination during the write cycle, so transfer is conducted in two bus cycles. At this time, the transfer data is temporarily stored in the DMAC. With the kind of external memory transfer shown in figure 9.5, data is read from one of the memories by the DMAC during a read cycle, then written to the other external memory during the subsequent write cycle. Figure 9.6 shows the timing for this operation.

1st bus cycle



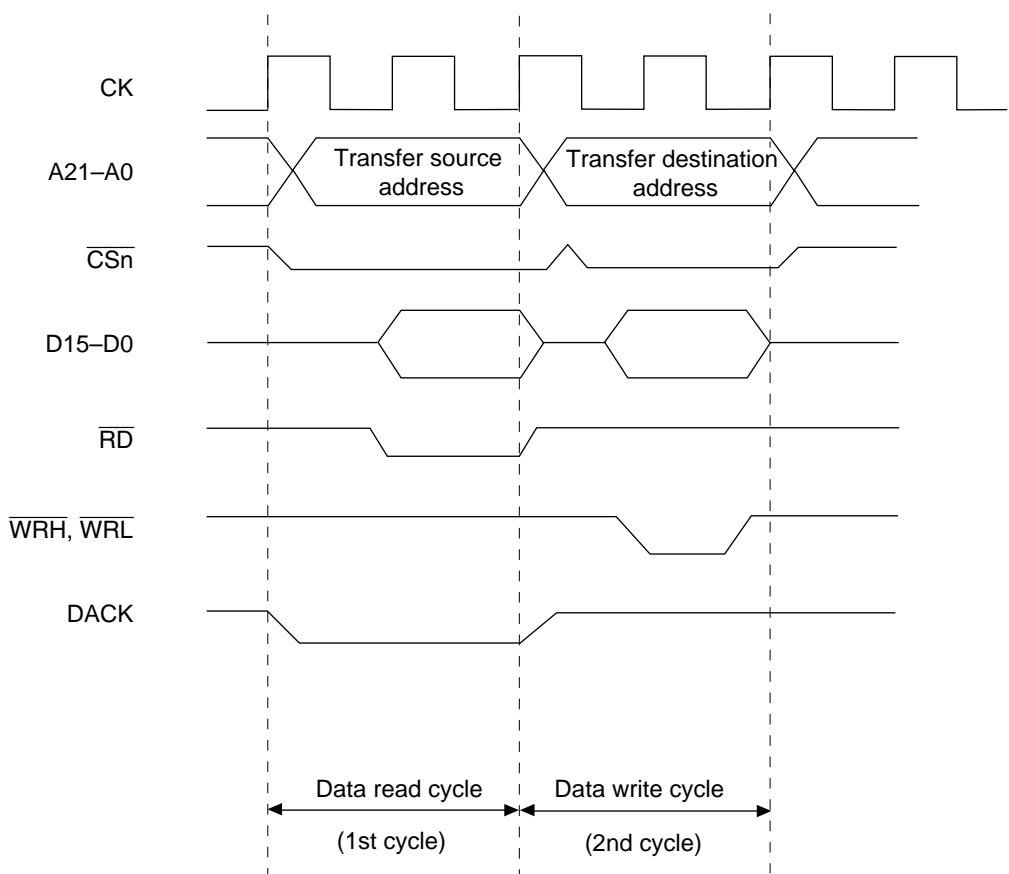
The SAR value is taken as the address, and data is read from the transfer source module and stored temporarily in the DMAC.

2nd bus cycle



The DAR value is taken as the address, and data stored in the DMAC's data buffer is written to the transfer destination module.

Figure 9.5 Operation during Dual Address Mode



Note: Transfer between external memories with DACK are output during read cycle.

Figure 9.6 Example of Transfer Timing in Dual Address Mode

9.3.7 Bus Modes

Select the appropriate bus mode in the TM bits of CHCR0, CHCR1. There are two bus modes: cycle steal and burst.

Cycle-Steal Mode: In the cycle steal mode, the bus right is given to another bus master after each one-transfer-unit (byte, word, or longword) DMAC transfer. When the next transfer request occurs, the bus rights are obtained from the other bus master and a transfer is performed for one transfer unit. When that transfer ends, the bus right is passed to the other bus master. This is repeated until the transfer end conditions are satisfied.

The cycle steal mode can be used with all categories of transfer destination, transfer source and transfer request. Figure 9.7 shows an example of DMA transfer timing in the cycle steal mode. Transfer conditions are dual address mode and $\overline{\text{DREQ}}$ level detection.

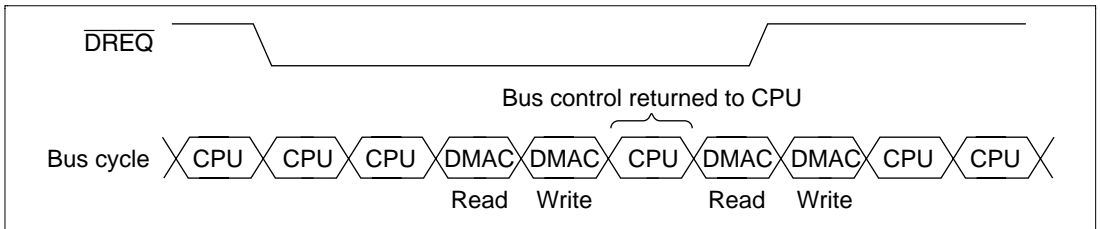


Figure 9.7 DMA Transfer Example in the Cycle-Steal Mode

Burst Mode: Once the bus right is obtained, the transfer is performed continuously until the transfer end condition is satisfied. In the external request mode with low level detection of the $\overline{\text{DREQ}}$ pin, however, when the $\overline{\text{DREQ}}$ pin is driven high, the bus passes to the other bus master after the bus cycle of the DMAC that currently has an acknowledged request ends, even if the transfer end conditions have not been satisfied.

Figure 9.8 shows an example of DMA transfer timing in the burst mode. Transfer conditions are single address mode and $\overline{\text{DREQ}}$ level detection.

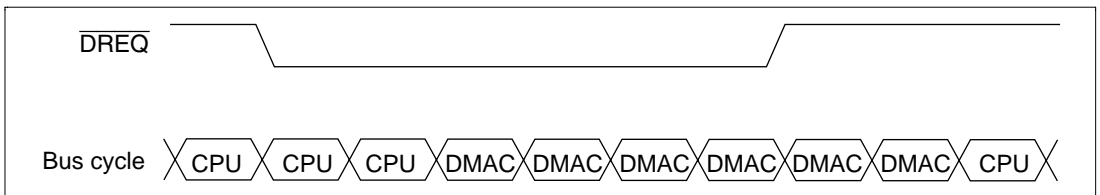


Figure 9.8 DMA Transfer Example in the Burst Mode

9.3.8 Relationship between Request Modes and Bus Modes by DMA Transfer Category

Table 9.6 shows the relationship between request modes and bus modes by DMA transfer category.

Table 9.6 Relationship of Request Modes and Bus Modes by DMA Transfer Category

| Address Mode | Transfer Category | Request Mode | Bus ^{*5} Mode | Transfer Size (Bits) | Usable Channels |
|---|---|-------------------|------------------------|-----------------------|--------------------|
| Single | External device with DACK and external memory | External | B/C | 8/16/32 | 0,1 |
| | External device with DACK and memory-mapped external device | External | B/C | 8/16/32 | 0, 1 |
| Dual | External memory and external memory | Any ^{*1} | B/C | 8/16/32 | 0, 1 |
| | External memory and memory-mapped external device | Any ^{*1} | B/C | 8/16/32 | 0, 1 |
| | Memory-mapped external device and memory-mapped external device | Any ^{*1} | B/C | 8/16/32 | 0, 1 |
| | External memory and on-chip memory | Any ^{*1} | B/C | 8/16/32 | 0, 1 |
| | External memory and on-chip peripheral module | Any ^{*2} | B/C ^{*3} | 8/16/32 ^{*4} | 0, 1 |
| | Memory-mapped external device and on-chip memory | Any ^{*1} | B/C | 8/16/32 | 0, 1 |
| | Memory-mapped external device and on-chip peripheral module | Any ^{*2} | B/C ^{*3} | 8/16/32 ^{*4} | 0, 1 |
| | On-chip memory and on-chip memory | Any ^{*1} | B/C | 8/16/32 | 0, 1 |
| | On-chip memory and on-chip peripheral module | Any ^{*2} | B/C ^{*3} | 8/16/32 ^{*4} | 0, 1 ^{*1} |
| On-chip peripheral module and on-chip peripheral module | Any ^{*2} | B/C ^{*3} | 8/16/32 ^{*4} | 0, 1 | |

- Notes:
1. External request, auto-request or on-chip peripheral module request enabled. However, in the case of on-chip peripheral module request, it is not possible to specify the SCI or A/D converter for the transfer request source.
 2. External request, auto-request or on-chip peripheral module request possible. However, if transfer request source is also the SCI or A/D converter, the transfer source or transfer destination must be the SCI or A/D converter.
 3. When the transfer request source is the SCI, only cycle steal mode is possible.
 4. Access size permitted by register of on-chip peripheral module that is the transfer source or transfer destination.
 5. B: Burst, C: Cycle steal

9.3.9 Bus Mode and Channel Priority Order

When a given channel is transferring in burst mode, and a transfer request is issued to channel 0, which has a higher priority ranking, transfer on channel 0 begins immediately. Channel 1 transfer is continued after transfer on channel 0 are completely ended, whether the channel 0 setting is cycle steal mode or burst mode.

9.3.10 Number of Bus Cycle States and $\overline{\text{DREQ}}$ Pin Sample Timing

Number of States in Bus Cycle: The number of states in the bus cycle when the DMAC is the bus master is controlled by the bus state controller (BSC) just as it is when the CPU is the bus master. For details, see section 8, Bus State Controller.

$\overline{\text{DREQ}}$ Pin Sampling Timing and DRAK Signal: In external request mode, the $\overline{\text{DREQ}}$ pin is sampled by either falling edge or low-level detection. When a $\overline{\text{DREQ}}$ input is detected, a DMAC bus cycle is issued and DMA transfer effected, at the earliest, after three states. However, in burst mode when single address operation is specified, a dummy cycle is inserted for the first bus cycle. In this case, the actual data transfer starts from the second bus cycle. Data is transferred continuously from the second bus cycle. The dummy cycle is not counted in the number of transfer cycles, so there is no need to recognize the dummy cycle when setting the DMATCR.

$\overline{\text{DREQ}}$ sampling from the second time begins from the start of the transfer one bus cycle prior to the DMAC transfer generated by the previous sampling.

DRAK is output once for the first $\overline{\text{DREQ}}$ sampling, irrespective of transfer mode or $\overline{\text{DREQ}}$ detection method. In burst mode, using edge detection, $\overline{\text{DREQ}}$ is sampled for the first cycle only, so DRAK is also output for the first cycle only. Therefore, the $\overline{\text{DREQ}}$ signal negate timing can be ascertained, and this facilitates handshake operations of transfer requests with the DMAC.

Cycle Steal Mode Operations: In cycle steal mode, $\overline{\text{DREQ}}$ sampling timing is the same irrespective of dual or single address mode, or whether edge or low-level $\overline{\text{DREQ}}$ detection is used.

For example, DMAC transfer begins (figure 9.9), at the earliest, three cycles from the first sampling timing. The second sampling begins at the start of the transfer one bus cycle prior to the start of the DMAC transfer initiated by the first sampling (i.e., from the start of the CPU(3) transfer). At this point, if DREQ detection has not occurred, sampling is executed every cycle thereafter.

As in figure 9.10, whatever cycle the CPU transfer cycle is, the next sampling begins from the start of the transfer one bus cycle before the DMAC transfer begins.

Figure 9.9 shows an example of output during DACK read and figure 9.10 an example of output during DACK write.

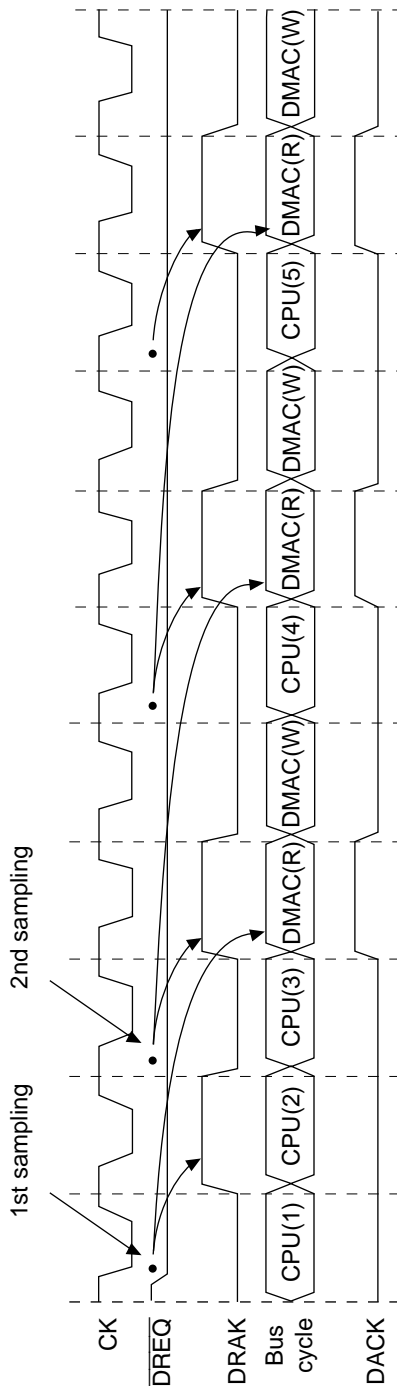
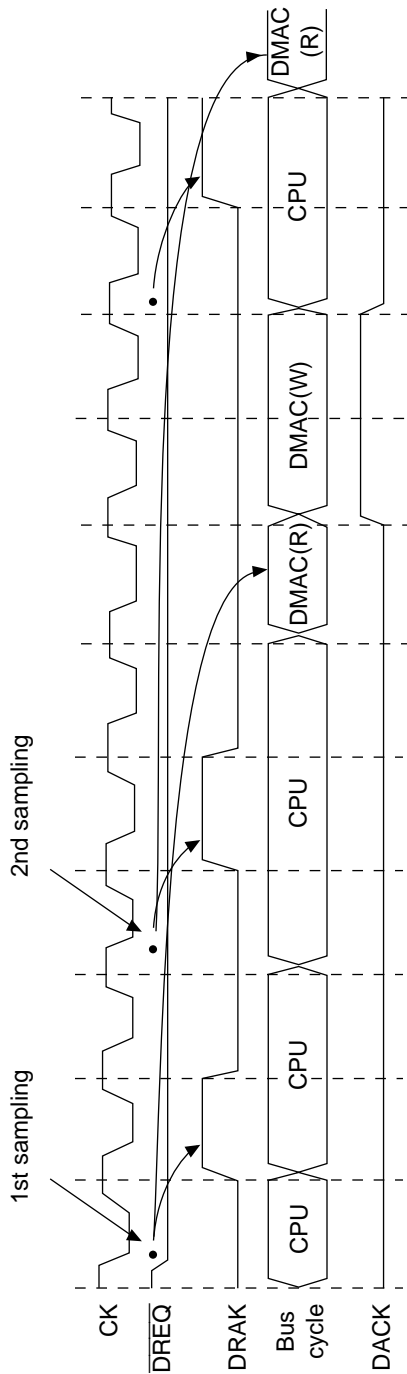


Figure 9.9 Cycle Steal, Dual Address and Level Detection (Fastest Operation)



Note: With cycle-steal and dual address operation, sampling timing is the same whether DREQ detection is by level or by edge.

Figure 9.10 Cycle Steal, Dual Address and Level Detection (Normal Operation)

Figures 9.11 and 9.12 show cycle steal mode and single address mode. In this case, transfer begins at earliest three cycles after the first $\overline{\text{DREQ}}$ sampling. The second sampling begins from the start of the transfer one bus cycle before the start of the first DMAC transfer. In single address mode, the $\overline{\text{DACK}}$ signal is output during the DMAC transfer period.

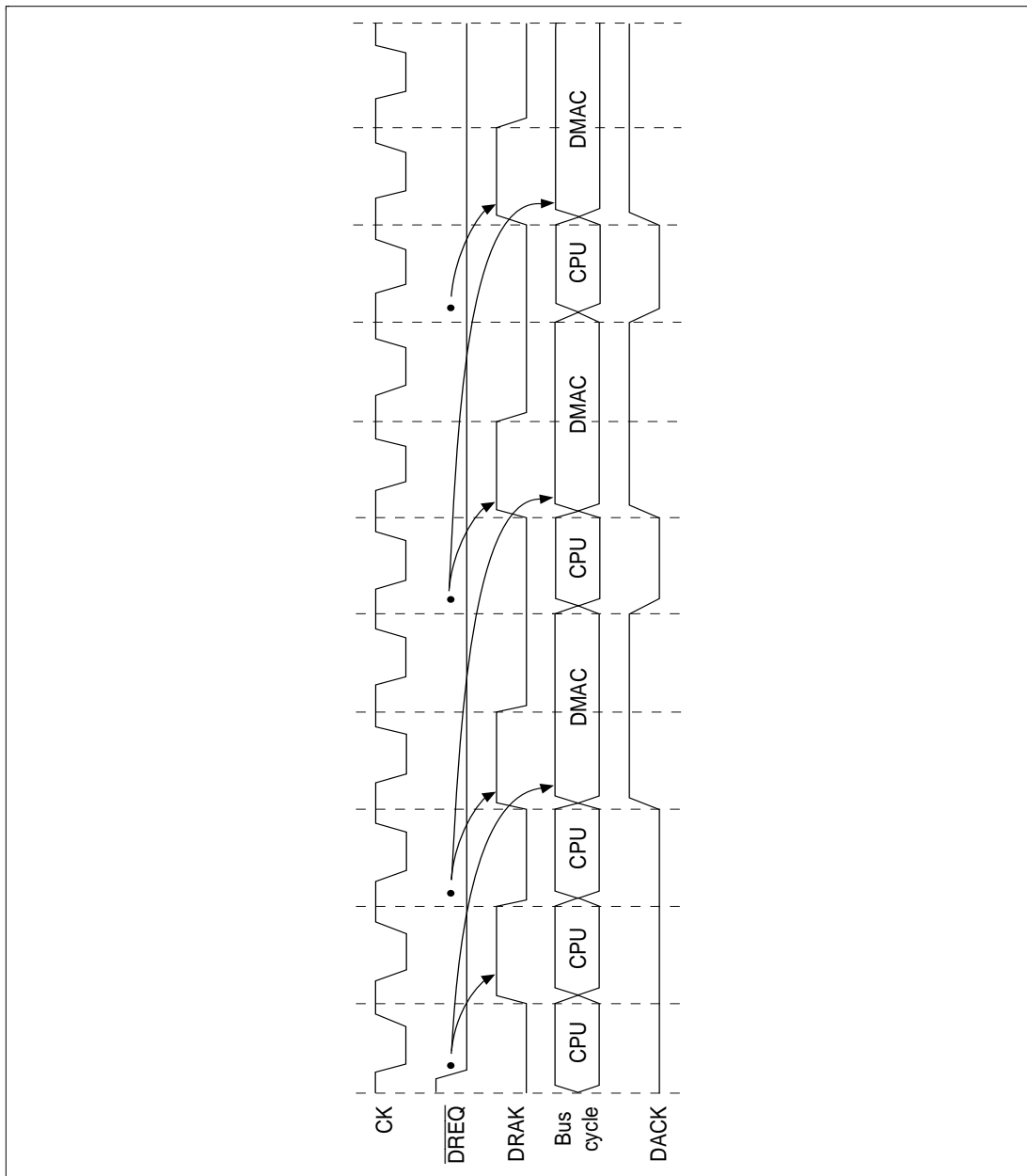


Figure 9.11 Cycle Steal, Single Address and Level Detection (Fastest Operation)

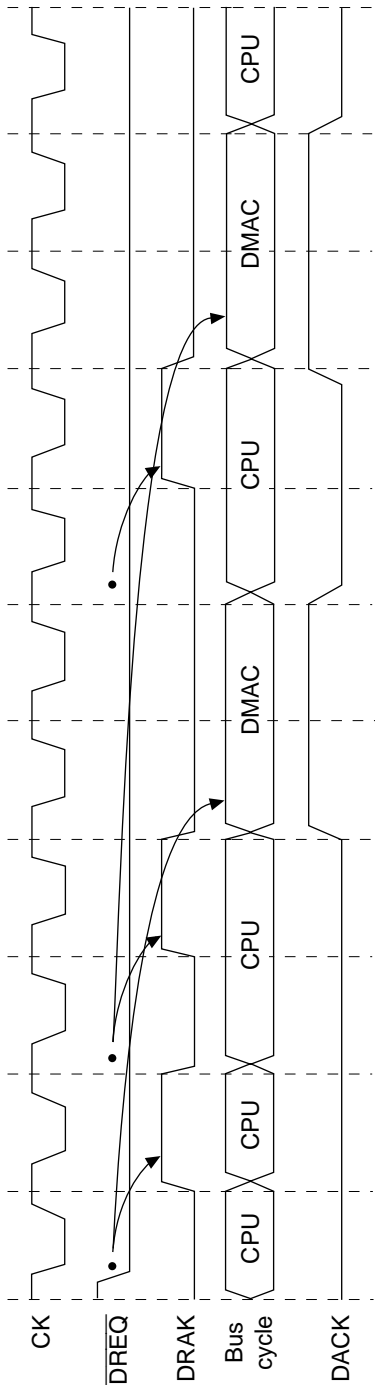


Figure 9.12 Cycle Steal, Single Address and Level Detection (Normal Operation)

Burst Mode, Dual Address, and Level Detection: $\overline{\text{DREQ}}$ sampling timing in burst mode with dual address and level detection is shown in figures 9.13 and 9.14. $\overline{\text{DREQ}}$ sampling timing in burst mode with dual address and level detection is virtually the same as that of cycle steal mode.

For example, DMAC transfer begins (figure 9.13), at the earliest, three cycles after the timing of the first sampling. The second sampling also begins from the start of the transfer one bus cycle before the start of the first DMAC transfer. In burst mode, as long as transfer requests are issued, DMAC transfer continues. Therefore, the “transfer one bus cycle before the start of the DMAC transfer” may be a DMAC transfer.

In burst mode, the DACK output period is the same as that of cycle steal mode. Figure 9.14 shows the normal operation of this burst mode.

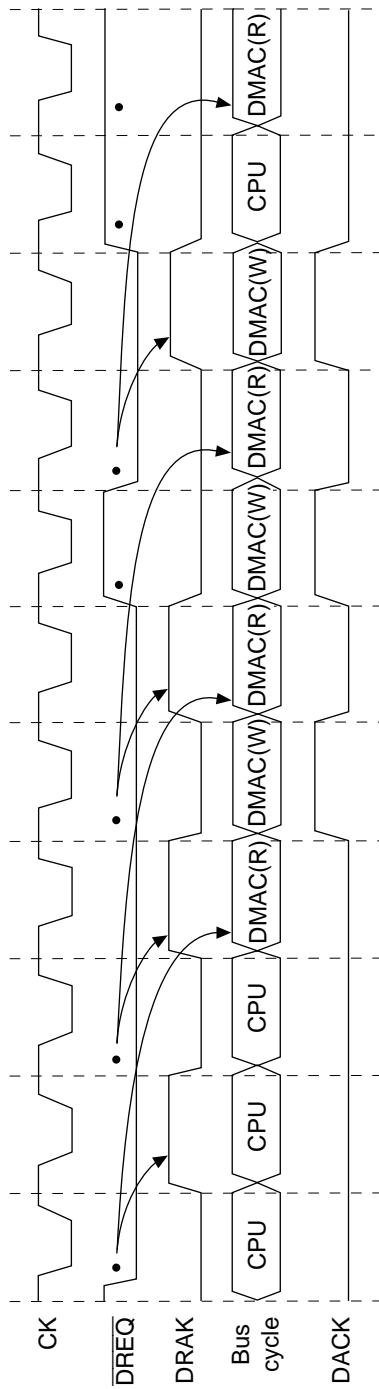


Figure 9.13 Burst Mode, Dual Address and Level Detection (Fastest Operation)

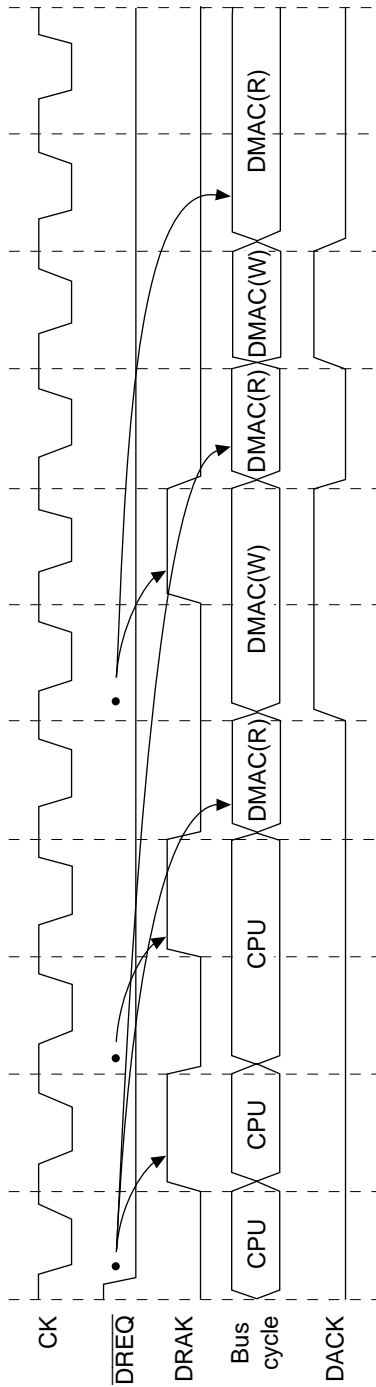


Figure 9.14 Burst Mode, Dual Address and Level Detection (Normal Operation)

Burst Mode, Single Address, and Level Detection: $\overline{\text{DREQ}}$ sampling timing in burst mode with single address and level detection is shown in figures 9.15 and 9.16.

In burst mode with single address and level detection, a dummy cycle is inserted as one bus cycle, at the earliest, three cycles after timing of the first sampling. Data during this period is undefined, and the DACK signal is not output. Nor is the number of DMAC transfers counted. The actual DMAC transfer begins after one dummy bus cycle output.

The dummy cycle is not counted either at the start of the second sampling (transfer one bus cycle before the start of the first DMAC transfer). Therefore, the second sampling is not conducted from the bus cycle starting the dummy cycle, but from the start of the CPU(3) bus cycle.

Thereafter, as long as $\overline{\text{DREQ}}$ is continuously sampled, no dummy cycle is inserted. $\overline{\text{DREQ}}$ sampling timing during this period begins from the start of the transfer one bus cycle before the start of DMAC transfer, in the same way as with cycle steal mode.

As with the fourth sampling in figure 9.15, once DMAC transfer is interrupted, a dummy cycle is again inserted at the start as soon as DMAC transfer is resumed.

The DACK output period in burst mode is the same as in cycle steal mode.

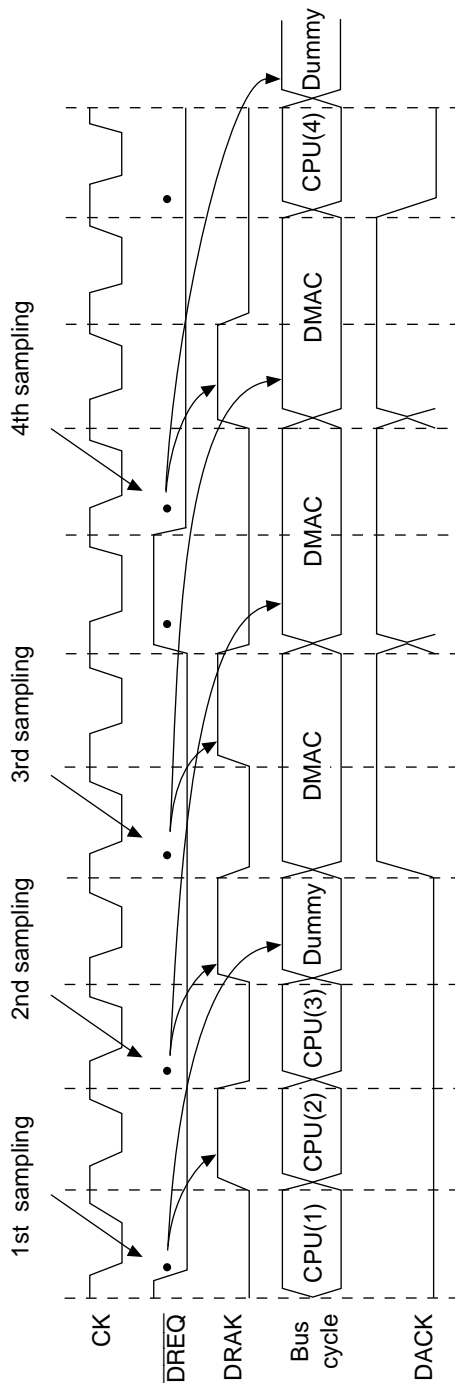


Figure 9.15 Burst Mode, Single Address and Level Detection (Fastest Operation)

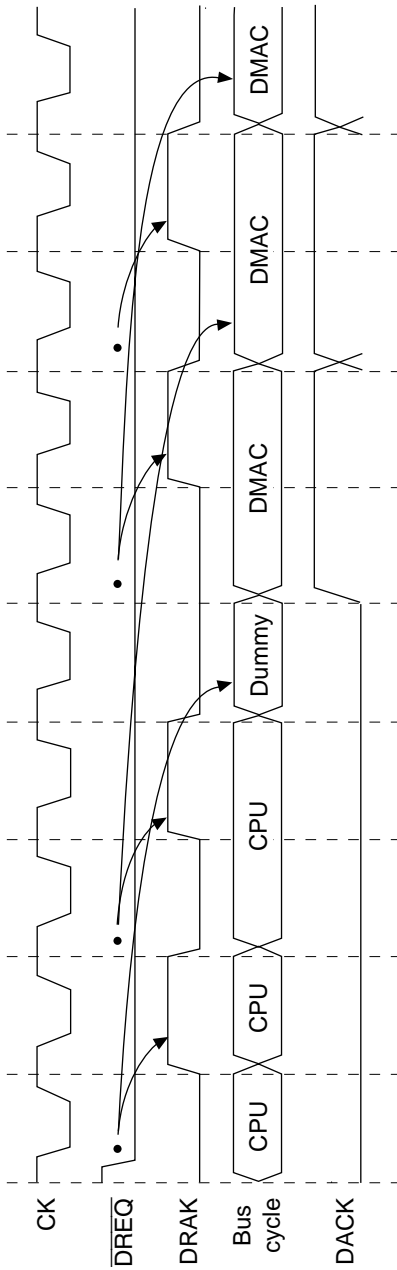


Figure 9.16 Burst Mode, Single Address and Level Detection (Normal Operation)

Burst Mode, Dual Address, and Edge Detection: In burst mode with dual address and edge detection, $\overline{\text{DREQ}}$ sampling is conducted only on the first cycle.

In figure 9.17, DMAC transfer begins, at the earliest, three cycles after the timing of the first sampling. Thereafter, DMAC transfer continues until the end of the data transfer count set in the TCR. $\overline{\text{DREQ}}$ sampling is not conducted during this period. Therefore, DRAK is output on the first cycle only.

When DMAC transfer is resumed after being halted by a NMI or address error, be sure to reinput an edge request. The remaining transfer restarts after the first DRAK output.

The DACK output period in burst mode is the same as in cycle steal mode.

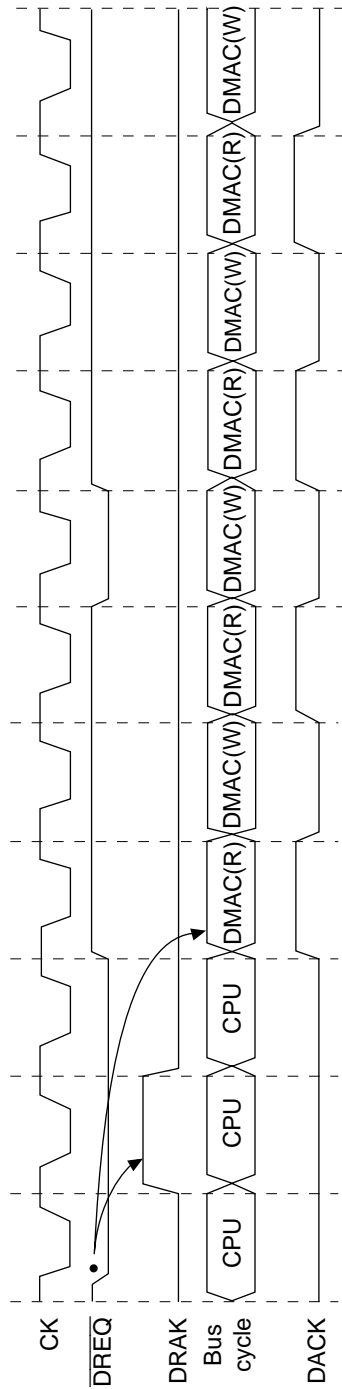


Figure 9.17 Burst Mode, Dual Address and Edge Detection

Burst Mode, Single Address, and Edge Detection: In burst mode with single address and edge detection, $\overline{\text{DREQ}}$ sampling is conducted only on the first cycle. In figure 9.18, a dummy cycle is inserted, at the earliest, three cycles after the timing for the first sampling. During this period, data is undefined, and DACK is not output. Nor is the number of DMAC transfers counted. Thereafter, DMAC transfer continues until the data transfer count set in the DMATCR has ended. $\overline{\text{DREQ}}$ sampling is not conducted during this period. Therefore, DRAK is output on the first cycle only.

When DMAC transfer is resumed after being halted by a NMI or address error, be sure to reinput an edge request. DRAK is output once, and the remaining transfer restarts after output of one dummy cycle.

The DACK output period in burst mode is the same as in cycle steal mode.

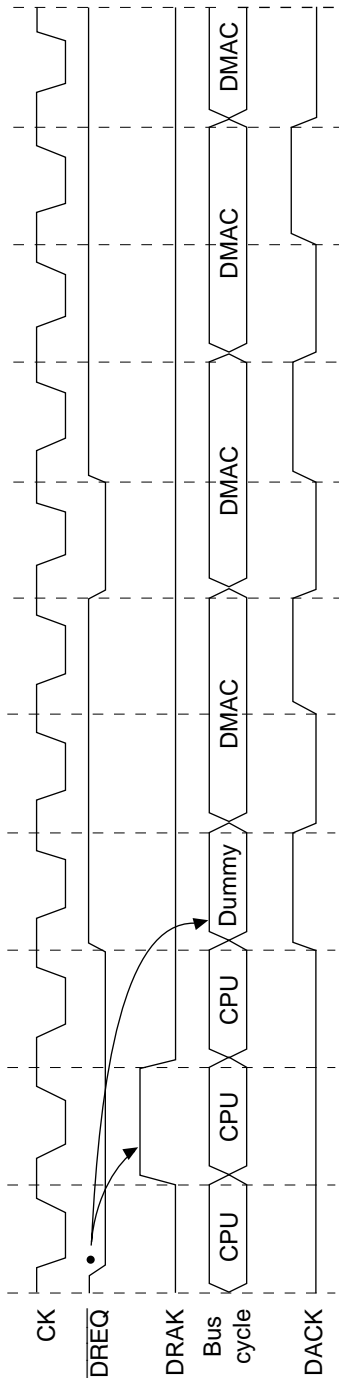


Figure 9.18 Burst Mode, Single Address and Edge Detection

9.3.11 DMA Transfer Ending Conditions

The DMA transfer ending conditions vary for individual channels ending and for all channels ending together.

Individual Channel Ending Conditions: There are two ending conditions. A transfer ends when the value of the channel's DMA transfer count register (DMATCR) is 0, or when the DE bit of the channel's CHCR is cleared to 0.

- When DMATCR is 0: When the DMATCR value becomes 0 and the corresponding channel's DMA transfer ends, the transfer end flag bit (TE) is set in the CHCR. If the IE (interrupt enable) bit has been set, a DMAC interrupt (DEI) is requested of the CPU.
- When DE of CHCR is 0: Software can halt a DMA transfer by clearing the DE bit in the channel's CHCR. The TE bit is not set when this happens.

Conditions for Ending All Channels Simultaneously: Transfers on all channels end when the NMIF (NMI flag) bit or AE (address error flag) bit is set to 1 in the DMAOR, or when the DME bit in the DMAOR is cleared to 0.

- When the NMIF or AE bit is set to 1 in DMAOR: When an NMI interrupt or DMAC address error occurs, the NMIF or AE bit is set to 1 in the DMAOR and all channels stop their transfers. The DMAC obtains the bus rights, and if these flags are set to 1 during execution of a transfer, DMAC halts operation when the transfer processing currently being executed ends, and transfers the bus right to the other bus master. Consequently, even if the NMIF or AE bits are set to 1 during a transfer, the DMA source address register (SAR), designation address register (DAR), and DMA transfer count register (DMATCR) are all updated. The TE bit is not set. To resume the transfers after NMI interrupt or address error processing, clear the appropriate flag bit to 0. To avoid restarting a transfer on a particular channel, clear its DE bit to 0.

When the processing of a one unit transfer is complete. In a dual address mode direct address transfer, even if an address error occurs or the NMI flag is set during read processing, the transfer will not be halted until after completion of the following write processing. In such a case, SAR, DAR, and DMATCR values are updated. In the same manner, the transfer is not halted in dual address mode indirect address transfers until after the final write processing has ended.

- When DME is cleared to 0 in DMAOR: Clearing the DME bit to 0 in the DMAOR aborts the transfers on all channels. The TE bit is not set.

9.3.12 DMAC Access from CPU

The space addressed by the DMAC is 3-cycle space. Therefore, when the CPU becomes the bus master and accesses the DMAC, a minimum of three basic clock (CLK) cycles are required for one bus cycle. Also, since the DMAC is located in word space, while a word-size access to the DMAC is completed in one bus cycle, a longword-size access is automatically divided into two word accesses, requiring two bus cycles (six basic clock cycles). These two bus cycles are executed consecutively; a different bus cycle is never inserted between the two word accesses. This applies to both write accesses and read accesses.

9.4 Examples of Use

9.4.1 Example of DMA Transfer between On-Chip SCI and External Memory

In this example, on-chip serial communication interface channel 0 (SCI0) received data is transferred to external memory using the DMAC channel 1.

Table 9.7 indicates the transfer conditions and the setting values of each of the registers.

Table 9.7 Transfer Conditions and Register Set Values for Transfer between On-chip SCI and External Memory

| Transfer Conditions | Register | Value |
|---|-----------------|--------------|
| Transfer source: RDR0 of on-chip SCI0 | SAR1 | H'FFFF81A5 |
| Transfer destination: external memory | DAR1 | H'00400000 |
| Transfer count: 64 times | DMATCR1 | H'00000040 |
| Transfer source address: fixed | CHCR1 | H'00004D05 |
| Transfer destination address: incremented | | |
| Transfer request source: SCI0 (RDR0) | | |
| Bus mode: cycle steal | | |
| Transfer unit: byte | | |
| Interrupt request generation at end of transfer | | |
| Channel priority ranking: 0 > 1 | DMAOR | H'0001 |

9.4.2 Example of DMA Transfer between External RAM and External Device with DACK

In this example, an external request, serial address mode transfer with external memory as the transfer source and an external device with DACK as the transfer destination is executed using DMAC channel 1.

Table 9.8 indicates the transfer conditions and the setting values of each of the registers.

Table 9.8 Transfer Conditions and Register Set Values for Transfer between External RAM and External Device with DACK

| Transfer Conditions | Register | Value |
|--|-----------------|------------------|
| Transfer source: external RAM | SAR1 | H'00400000 |
| Transfer destination: external device with DACK | DAR1 | (access by DACK) |
| Transfer count: 32 times | DMATCR1 | H'00000020 |
| Transfer source address: decremented | CHCR1 | H'00002269 |
| Transfer destination address: (setting ineffective) | | |
| Transfer request source: external pin ($\overline{\text{DREQ1}}$) edge detection | | |
| Bus mode: burst | | |
| Transfer unit: word | | |
| No interrupt request generation at end of transfer | | |
| Channel priority ranking: $0 > 1$ | DMAOR | H'0001 |

9.5 Cautions on Use

1. Other than the DMA operation register (DMAOR) accessing in word (16 bit) units, access all registers in word (16 bit) or longword (32 bit) units.
2. When rewriting the RS0–RS3 bits of CHCR0, CHCR1, first clear the DE bit to 0 (set the DE bit to 0 before doing rewrites with a CHCR byte address).
3. When an NMI interrupt is input, the NMIF bit of the DMAOR is set even when the DMAC is not operating.
4. Set the DME bit of the DMAOR to 0 and make certain that any DMAC received transfer request processing has been completed before entering standby mode.
5. Do not access the DMAC or BSC on-chip peripheral modules from the DMAC.
6. When activating the DMAC, do the CHCR or DMAOR setting as the final step. There are instances where abnormal operation will result if any other registers are established last.
7. After the DMATCR count becomes 0 and the DMA transfer ends normally, always write a 0 to the DMATCR, even when executing the maximum number of transfers on the same channel. There are instances where abnormal operation will result if this is not done.
8. When detecting external requests by falling edge, maintain the external request pin at high level when performing the DMAC establishment.
9. When operating in single address mode, establish an external address as the address. There are instances where abnormal operation will result if an internal address is established.
10. Do not access DMAC register empty addresses (H'FFFF86B2 to H'FFFF86BF, H'FFFF86E4 to H'FFFF86FF). Operation cannot be guaranteed when empty addresses are accessed.

Section 10 Multifunction Timer Pulse Unit (MTU)

10.1 Overview

The SuperH microcomputer has an on-chip 16-bit multifunction timer pulse unit (MTU) with three channels of 16-bit timers.

10.1.1 Features

- Can process a maximum of eight different pulse outputs and inputs.
- Has eight timer general registers (TGR): four for channel 0, and two each for channels 1 and 2 that can be set to function independently as output compare or input capture. The channel 0 TGRC and TGRD registers can be used as buffer registers.
- Can select eight counter input clock sources for all channels
- All channels can be set for the following operating modes:
 - Compare match waveform output: 0 output/1 output/toggle output selectable.
 - Input capture function: Selectable rising edge, falling edge, or both rising and falling edge detection.
 - Counter clearing function: Counters can be cleared by a compare-match or input capture.
 - Synchronizing mode: Two or more timer counters (TCNT) can be written to simultaneously. Two or more timer counters can be simultaneously cleared by a compare-match or input capture. Counter synchronization functions enable synchronized register input/output.
 - PWM mode: PWM output can be provided with any duty cycle. When combined with the counter synchronizing function, enables up to eight-phase PWM output. (With channels 0 to 2 set to PWM mode and synchronized (channel 0 to 2 phase output: 3, 2, 2).)
- Channels 0 can be set for buffer operation
 - Input capture register double buffer configuration possible
 - Output compare register automatic re-write possible
- Channels 1, 2 can be independently set to the phase counting mode
 - Two-phase encoder pulse up/down count possible
- Cascade connection operation
 - Can be operated as a 32-bit counter by using the channel 2 input clock for channel 1 overflow/underflow
- High speed access via internal 16-bit bus

- Thirteen interrupt sources
 - Channel 0 has four compare-match/input capture interrupts and one overflow interrupt which can be requested independently.
 - Channels 1 and 2 have two compare-match/input capture interrupts, one overflow interrupt, and one underflow interrupt which can be requested independently.
- Automatic transfer of register data

Block transfer, 1-word data transfers and 1-byte data transfers are possible through DMAC activation.
- A/D converter conversion start trigger can be generated
 - Channels 0 to 2 compare-match/input capture signals can be used as A/D converter conversion start triggers.

Table 10.1 summarizes the MTU functions.

Table 10.1 MTU Functions

| Item | | Channel 0 | Channel 1 | Channel 2 |
|------------------------------------|--------|---|--------------------------------------|--------------------------------------|
| Counter clocks | | Internal: $\phi/1$, $\phi/4$, $\phi/16$, $\phi/64$, $\phi/256$, $\phi/1024$ External: Eight to each channel from TCLKA, TCLKB, TCLKC, and TCLKD | | |
| General registers | | TGR0A | TGR1A | TGR2A |
| | | TGR0B | TGR1B | TGR2B |
| General registers/buffer registers | | TGR0C | No | No |
| | | TGR0D | | |
| Input/output pins | | TIOC0A | TIOC1A | TIOC2A |
| | | TIOC0B | TIOC1B | TIOC2B |
| | | TIOC0C | | |
| | | TIOC0D | | |
| Counter clear function | | TGR compare-match or input capture | TGR compare-match or input capture | TGR compare-match or input capture |
| Compare match output | 0 | Yes | Yes | Yes |
| | 1 | Yes | Yes | Yes |
| | Toggle | Yes | Yes | Yes |
| Input capture function | | Yes | Yes | Yes |
| Synchronization | | Yes | Yes | Yes |
| Buffer operation | | Yes | No | No |
| PWM mode 1 | | Yes | Yes | Yes |
| PWM mode 2 | | Yes | Yes | Yes |
| Phase counting mode | | No | Yes | Yes |
| DMAC activation | | TGR0A compare match or input capture | TGR1A compare match or input capture | TGR2A compare match or input capture |

Table 10.1 MTU Functions (cont)

| Item | Channel 0 | Channel 1 | Channel 2 |
|------------------------------|---------------------------------------|---------------------------------------|---------------------------------------|
| A/D conversion start trigger | TGR0A com-pare match or input capture | TGR1A com-pare match or input capture | TGR2A com-pare match or input capture |
| Interrupt sources | Compare match/input capture 0A | Compare match/input capture 1A | Compare match/input capture 2A |
| | Compare match/input capture 0B | Compare match/input capture 1B | Compare match/input capture 2B |
| | Compare match/input capture 0C | Overflow | Overflow |
| | Compare match/input capture 0D | Underflow | Underflow |
| | Overflow | — | — |

10.1.2 Block Diagram

Figure 10.1 is the block diagram of the MTU.

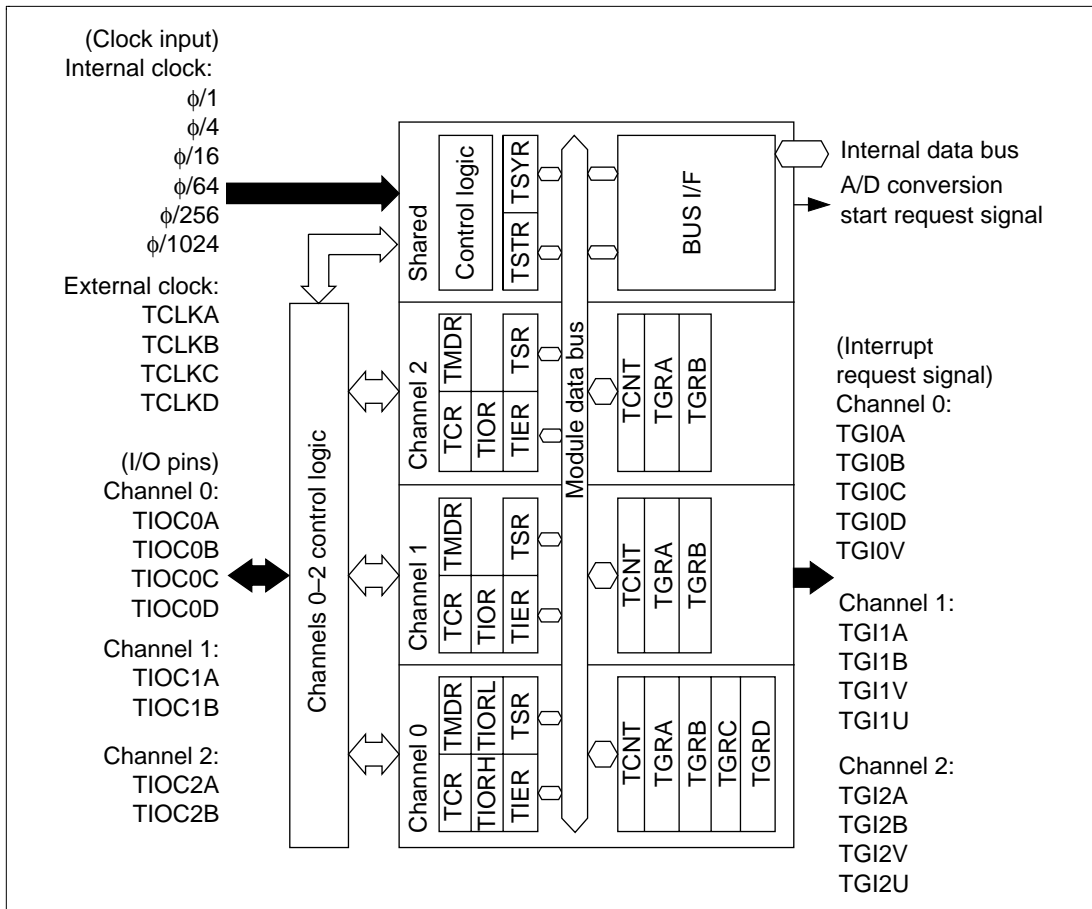


Figure 10.1 MTU Block Diagram

10.1.3 Pin Configuration

Table 10.2 summarizes the MTU pins.

Table 10.2 Pin Configuration

| Channel | Name | Pin Name | I/O | Function |
|---------|---------------------------------------|----------|-----|--|
| Shared | Clock input A | TCLKA | I | Clock A input pin (A-phase input pin in channel 1 phase counting mode) |
| | Clock input B | TCLKB | I | Clock B input pin (B-phase input pin in channel 1 phase counting mode) |
| | Clock input C | TCLKC | I | Clock C input pin (A-phase input pin in channel 2 phase counting mode) |
| | Clock input D | TCLKD | I | Clock D input pin (B-phase input pin in channel 2 phase counting mode) |
| 0 | Input capture/output compare-match 0A | TIOC0A | I/O | TGR0A input capture input/output compare output/PWM output pin |
| | Input capture/output compare-match 0B | TIOC0B | I/O | TGR0B input capture input/output compare output/PWM output pin |
| | Input capture/output compare-match 0C | TIOC0C | I/O | TGR0C input capture input/output compare output/PWM output pin |
| | Input capture/output compare-match 0D | TIOC0D | I/O | TGR0D input capture input/output compare output/PWM output pin |
| 1 | Input capture/output compare-match 1A | TIOC1A | I/O | TGR1A input capture input/output compare output/PWM output pin |
| | Input capture/output compare-match 1B | TIOC1B | I/O | TGR1B input capture input/output compare output/PWM output pin |
| 2 | Input capture/output compare-match 2A | TIOC2A | I/O | TGR2A input capture input/output compare output/PWM output pin |
| | Input capture/output compare-match 2B | TIOC2B | I/O | TGR2B input capture input/output compare output/PWM output pin |

Note: The TIOC pins output undefined values when they are set to input capture and timer output by the pin function controller (PFC).

10.1.4 Register Configuration

Table 10.3 summarizes the MTU register configuration.

Table 10.3 Register Configuration

| Channel | Name | Abbreviation | R/W | Initial Value | Address | Access Size (Bits) *1 |
|---------|-----------------------------------|--------------|---------|---------------|------------|-----------------------|
| Shared | Timer start register | TSTR | R/W | H'00 | H'FFFF8240 | 8, 16, 32 |
| | Timer synchro register | TSYR | R/W | H'00 | H'FFFF8241 | |
| 0 | Timer control register 0 | TCR0 | R/W | H'00 | H'FFFF8260 | 16, 32 |
| | Timer mode register 0 | TMDR0 | R/W | H'C0 | H'FFFF8261 | |
| | Timer I/O control register 0H | TIOR0H | R/W | H'00 | H'FFFF8262 | |
| | Timer I/O control register 0L | TIOR0L | R/W | H'00 | H'FFFF8263 | |
| | Timer interrupt enable register 0 | TIER0 | R/W | H'40 | H'FFFF8264 | |
| | Timer status register 0 | TSR0 | R/(W)*2 | H'C0 | H'FFFF8265 | |
| | Timer counter 0 | TCNT0 | R/W | H'0000 | H'FFFF8266 | |
| | General register 0A | TGR0A | R/W | H'FFFF | H'FFFF8268 | |
| | General register 0B | TGR0B | R/W | H'FFFF | H'FFFF826A | |
| | General register 0C | TGR0C | R/W | H'FFFF | H'FFFF826C | |
| | General register 0D | TGR0D | R/W | H'FFFF | H'FFFF826E | |
| 1 | Timer control register 1 | TCR1 | R/W | H'00 | H'FFFF8280 | 8, 16, 32 |
| | Timer mode register 1 | TMDR1 | R/W | H'C0 | H'FFFF8281 | |
| | Timer I/O control register 1 | TIOR1 | R/W | H'00 | H'FFFF8282 | |
| | Timer interrupt enable register 1 | TIER1 | R/W | H'40 | H'FFFF8284 | |
| | Timer status register 1 | TSR1 | R/(W)*2 | H'C0 | H'FFFF8285 | |
| | Timer counter 1 | TCNT1 | R/W | H'0000 | H'FFFF8286 | |
| | General register 1A | TGR1A | R/W | H'FFFF | H'FFFF8288 | |
| | General register 1B | TGR1B | R/W | H'FFFF | H'FFFF828A | |

Table 10.3 Register Configuration (cont)

| Channel | Name | Abbreviation | R/W | Initial Value | Address | Access Size (Bits) *1 |
|---------|-----------------------------------|--------------|---------|---------------|------------|-----------------------|
| 2 | Timer control register 2 | TCR2 | R/W | H'00 | H'FFFF82A0 | 8, 16, 32 |
| | Timer mode register 2 | TMDR2 | R/W | H'C0 | H'FFFF82A1 | |
| | Timer I/O control register 2 | TIOR2 | R/W | H'00 | H'FFFF82A2 | |
| | Timer interrupt enable register 2 | TIER2 | R/W | H'40 | H'FFFF82A4 | |
| | Timer status register 2 | TSR2 | R/(W)*2 | H'C0 | H'FFFF82A5 | |
| | Timer counter 2 | TCNT2 | R/W | H'0000 | H'FFFF82A6 | 16, 32 |
| | General register 2A | TGR2A | R/W | H'FFFF | H'FFFF82A8 | |
| | General register 2B | TGR2B | R/W | H'FFFF | H'FFFF82AA | |

Notes: 1. 16-bit registers (TCNT, TGR) cannot be read or written in 8-bit units.
 2. Write 0 to clear flags.

10.2 MTU Register Descriptions

10.2.1 Timer Control Register (TCR)

The TCR is an 8-bit read/write register for controlling the TCNT counter for each channel. The MTU has three TCR registers, one for each of the channels 0 to 2. TCR is initialized to H'00 by a power-on reset or the standby mode.

Channel 0: TCR0

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| 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 |

Channels 1, 2: TCR1, TCR2

| | | | | | | | | |
|----------------|---|-------|-------|-------|-------|-------|-------|-------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 |
| 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 | R/W |

- Bits 7–5—Counter Clear 2, 1, 0 (CCLR2, CCLR1, CCLR0): Select the counter clear source for the TCNT counter.

— Channels 0

| Bit 7: CCLR2 | Bit 6: CCLR1 | Bit 5: CCLR0 | Description |
|-----------------|-----------------|-----------------|---|
| 0 | 0 | 0 | TCNT clear disabled (initial value) |
| | | 1 | TCNT is cleared by TGRA compare-match or input capture |
| | 1 | 0 | TCNT is cleared by TGRB compare-match or input capture |
| | | 1 | Synchronizing clear: TCNT is cleared in synchronization with clear of other channel counters operating in sync.* ¹ |
| 1 | 0 | 0 | TCNT clear disabled |
| | | 1 | TCNT is cleared by TGRC compare-match or input capture* ² |
| | 1 | 0 | TCNT is cleared by TGRD compare-match or input capture* ² |
| | | 1 | Synchronizing clear: TCNT is cleared in synchronization with clear of other channel counters operating in sync* ¹ |

- Notes:
1. Setting the SYNC bit of the TSYR to 1 sets the synchronization.
 2. When TGRC or TGRD are functioning as buffer registers, TCNT is not cleared because the buffer registers have priority and compare-match/input captures do not occur.

— Channels 1, 2

| Bit 7: Reserved* ¹ | Bit 6: CCLR1 | Bit 5: CCLR0 | Description |
|----------------------------------|-----------------|-----------------|--|
| 0 | 0 | 0 | TCNT clear disabled (initial value) |
| | | 1 | TCNT is cleared by TGRA compare-match or input capture |
| | 1 | 0 | TCNT is cleared by TGRB compare-match or input capture |
| | | 1 | Synchronizing clear: TCNT is cleared in synchronization with clear of other channel counters operating in sync* ² |

- Notes:
1. The bit 7 of channels 1 and 2 is reserved. It always reads 0, and cannot be modified.
 2. Setting the SYNC bit of the TSYR to 1 sets the synchronization.

- Bits 4–3—Clock Edge 1, 0 (CKEG1 and CKEG0): CKEG1 and CKEG0 select the input clock edges. When counting is done on both edges of the internal clock the input clock frequency becomes 1/2 (Example: both edges of $\phi/4$ = rising edge of $\phi/2$). When phase count mode is used with channels 1, 2, these settings are ignored, as the phase count mode settings have priority.

Bit 4: Bit 3:
CKEG1 CKEG0 Description

| | | |
|---|---|--|
| 0 | 0 | Count on rising edges (initial value) |
| | 1 | Count on falling edges |
| 1 | X | Count on both rising and falling edges |

Notes: 1. X: 0 or 1, don't care.
 2. Internal clock edge selection is effective when the input clock is $\phi/4$ or slower. These settings are ignored when $\phi/1$, or the overflow/underflow of another channel is selected for the input clock.

- Bits 2–0—Timer Prescaler 2–0 (TPSC2–TPSC0): TPSC2–TPSC0 select the counter clock source for the TCNT. An independent clock source can be selected for each channel. Table 10.4 shows the possible settings for each channel.

Table 10.4 MTU Clock Sources

| Chan- nel | Internal Clock | | | | | | Other Channel Overflow/ Underflow | External Clock | | | |
|--------------|----------------|----------|-----------|-----------|------------|-------------|---|----------------|-----------|-----------|-----------|
| | $\phi/1$ | $\phi/4$ | $\phi/16$ | $\phi/64$ | $\phi/256$ | $\phi/1024$ | | TCL KA | TCL KB | TCL KC | TCL KD |
| 0 | O | O | O | O | X | X | X | O | O | O | O |
| 1 | O | O | O | O | O | X | O | O | O | X | X |
| 2 | O | O | O | O | X | O | X | O | O | O | X |

Note: Symbols: O: Setting possible X: Setting impossible

— Channel 0

Bit 2: Bit 1: Bit 0:
TPSC2 TPSC1 TPSC0 Description

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | Internal clock: count with $\phi/1$ (initial value) |
| | | 1 | Internal clock: count with $\phi/4$ |
| | 1 | 0 | Internal clock: count with $\phi/16$ |
| | | 1 | Internal clock: count with $\phi/64$ |
| 1 | 0 | 0 | External clock: count with the TCLKA pin input |
| | | 1 | External clock: count with the TCLKB pin input |
| | 1 | 0 | External clock: count with the TCLKC pin input |
| | | 1 | External clock: count with the TCLKD pin input |

— Channel 1

| Bit 2: TPSC2 | Bit 1: TPSC1 | Bit 0: TPSC0 | Description |
|-----------------|-----------------|-----------------|---|
| 0 | 0 | 0 | Internal clock: count with $\phi/1$ (initial value) |
| | | 1 | Internal clock: count with $\phi/4$ |
| | 1 | 0 | Internal clock: count with $\phi/16$ |
| | | 1 | Internal clock: count with $\phi/64$ |
| 1 | 0 | 0 | External clock: count with the TCLKA pin input |
| | | 1 | External clock: count with the TCLKB pin input |
| | 1 | 0 | Internal clock: count with $\phi/256$ |
| | | 1 | Count with the TCNT2 overflow/underflow |

Note: These settings are ineffective when channel 1 is in phase counting mode.

— Channel 2

| Bit 2: TPSC2 | Bit 1: TPSC1 | Bit 0: TPSC0 | Description |
|-----------------|-----------------|-----------------|---|
| 0 | 0 | 0 | Internal clock: count with $\phi/1$ (initial value) |
| | | 1 | Internal clock: count with $\phi/4$ |
| | 1 | 0 | Internal clock: count with $\phi/16$ |
| | | 1 | Internal clock: count with $\phi/64$ |
| 1 | 0 | 0 | External clock: count with the TCLKA pin input |
| | | 1 | External clock: count with the TCLKB pin input |
| | 1 | 0 | External clock: count with the TCLKC pin input |
| | | 1 | Internal clock: count with $\phi/1024$ |

Note: These settings are ineffective when channel 2 is in phase counting mode.

10.2.2 Timer Mode Register (TMDR)

The TMDR is an 8-bit read/write register that sets the operating mode for each channel. The MTU has three TMDR registers, one for each channel. TMDR is initialized to H'C0 by a power-on reset or the standby mode.

Channel 0: TMDR0

| | | | | | | | | |
|----------------|---|---|-----|-----|-----|-----|-----|-----|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 |
| Initial value: | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R/W | R/W | R/W | R/W | R/W | R/W |

Channels 1, 2: TMDR1, TMDR2

| | | | | | | | | |
|----------------|---|---|---|---|-----|-----|-----|-----|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | MD3 | MD2 | MD1 | MD0 |
| Initial value: | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R/W | R/W | R/W | R/W |

- Bits 7, 6—Reserved: These bits always read as 1. The write value should always be 1.
- Bit 5—Buffer Operation B (BFB): Designates whether to use the TGRB register for normal operation, or buffer operation in combination with the TGRD register. When using TGRD as a buffer register, no TGRD register input capture/output compares are generated.

This bit is reserved in channels 1 and 2, which have no TGRD registers. This bit always read as 0. The write value should always be 0.

| Bit 5: BFB | Description |
|------------|--|
| 0 | TGRB operates normally (initial value) |
| 1 | TGRB and TGRD buffer operation |

- Bit 4—Buffer Operation A (BFA): Designates whether to use the TGRA register for normal operation, or buffer operation in combination with the TGRC register. When using TGRC as a buffer register, no TGRC register input capture/output compares are generated.
- This bit is reserved in channels 1 and 2, which have no TGRC registers. This bit always read as 0. The write value should always be 0.

| Bit 4: BFA | Description |
|------------|--|
| 0 | TGRA operates normally (initial value) |
| 1 | TGRA and TGRC buffer operation |

- Bits 3–0—Modes 3–0 (MD3–MD0): These bits set the timer operation mode.

| Bit 3: MD3 | Bit 2: MD2 | Bit 1: MD1 | Bit 0: MD0 | Description |
|---------------|---------------|---------------|-----------------------|----------------------------------|
| 0 | 0 | 0 | 0 | Normal operation (initial value) |
| | | | 1 | Reserved (do not set) |
| | 1 | 0 | 0 | PWM mode 1 |
| | | | 1 | PWM mode 2 |
| | | 1 | 0 | Phase counting mode 1* |
| | | | 1 | Phase counting mode 2* |
| | 1 | 0 | 0 | Reserved (do not set) |
| | | | 1 | Reserved (do not set) |
| 1 | | 0 | Reserved (do not set) | |
| | | 1 | Reserved (do not set) | |
| 1 | | 0 | 0 | Reserved (do not set) |
| | | | 1 | Reserved (do not set) |
| | 1 | 0 | Reserved (do not set) | |
| | | 1 | Reserved (do not set) | |

Note: Phase measurement mode can not be set for channel 0.

10.2.3 Timer I/O Control Register (TIOR)

The TIOR is a register that controls the TGR. The MTU has four TIOR registers, two for channels 0 and one each for channels 1 and 2. TIOR is initialized to H'00 by a power-on reset or the standby mode.

Channel 0: TIOR0H

Channels 1, 2: TIOR1, TIOR2

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| | I OB3 | I OB2 | I OB1 | I OB0 | I OA3 | I OA2 | I OA1 | I OA0 |
| 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 |

Bits 7–4—I/O Control B3–B0 (IOB3–IOB0): These bits set the TGRB register function.

Bits 3–0—I/O Control A3–B0 (IOA3–IOA0): These bits set the TGRA register function.

Channel 0: TIOR0L

| | | | | | | | | |
|----------------|------|------|------|------|------|------|------|------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 |
| 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: When the TGRC or TGRD registers are set for buffer operation, these settings become ineffective and the operation is as a buffer register.

- Bits 7–4—I/O Control D3–D0 (IOD3–IOD0): These bits set the TGRD register function.
- Bits 3–0—I/O Control C3–C0 (IOC3–IOC0): These bits set the TGRC register function.

Channel 0 (TIOR0H Register)

- Bits 7–4—I/O Control B3–B0 (IOB3–IOB0): These bits set the TGR0B register function.

Bit 7: IOB3 **Bit 6:** IOB2 **Bit 5:** IOB1 **Bit 4:** IOB0 **Description**

| Bit 7: IOB3 | Bit 6: IOB2 | Bit 5: IOB1 | Bit 4: IOB0 | Description | | |
|----------------|----------------|----------------|----------------|-------------------------------------|--|--|
| 0 | 0 | 0 | 0 | TGR0B is an output compare register | Output disabled (initial value) | |
| | | | 1 | | Initial output is 0 | Output 0 on compare-match |
| | | | 1 | 0 | | Output 1 on compare-match |
| | 1 | 0 | 0 | 1 | | Toggle output on compare-match |
| | | | | 1 | | Output disabled |
| | | | | 1 | | Initial output is 1 |
| 1 | 0 | 0 | 1 | | Output 1 on compare-match | |
| | | | 1 | | Toggle output on compare-match | |
| | | | 1 | | Output disabled | |
| | 1 | 0 | 0 | 0 | TGR0B is an input capture register | Capture input source is the TIOC0B pin |
| | | | | 1 | | Input capture on rising edge |
| | | | | 1 | 0 | |
| 1 | 0 | 0 | 1 | | Input capture on both edges | |
| | | | 1 | | Capture input source is channel 1/ count clock | |
| | | | 1 | 0 | | Input capture on TCNT1 count up/count down |
| 1 | 0 | 0 | 1 | | | |
| | | | 1 | | | |

- Bits 3–0—I/O Control A3–A0 (IOA3–IOA0): These bits set the TGR0A register function.

| Bit 3: IOA3 | Bit 2: IOA2 | Bit 1: IOA1 | Bit 0: IOA0 | Description | |
|----------------|----------------|----------------|----------------|------------------------------|---------------------------------|
| 0 | 0 | 0 | 0 | TGR0A | Output disabled (initial value) |
| | | | 1 | is an output | Initial output is 0 |
| | | 1 | 0 | compare | Output 0 on compare-match |
| | | | 1 | register | Output 1 on compare-match |
| | 1 | 0 | 0 | | Toggle output on compare-match |
| | | | 1 | | Output disabled |
| | | 1 | 0 | | Initial output is 1 |
| | | | 1 | | Output 0 on compare-match |
| 1 | 0 | 0 | 0 | TGR0A | Capture input |
| | | | 1 | is an input | source is the |
| | | 1 | 0 | capture | TIOC0A pin |
| | | | 1 | register | Input capture on rising edge |
| | 1 | 0 | 0 | | Input capture on falling edge |
| | | | 1 | | Input capture on both edges |
| | | 1 | 0 | | Capture input |
| | | | 1 | | source is |
| 1 | 0 | | channel 1/ | Input capture on TCNT1 count | |
| | 1 | | count clock | up/count down | |

Channel 0 (TIOR0L Register)

- Bits 7–4—I/O Control D3–D0 (IOD3–IOD0): These bits set the TGR0D register function.

| Bit 7: IOD3 | Bit 6: IOD2 | Bit 5: IOD1 | Bit 4: IOD0 | Description | | |
|----------------|----------------|----------------|----------------|--|---------------------------------|-------------------------------|
| 0 | 0 | 0 | 0 | TGR0D is an output compare register | Output disabled (initial value) | |
| | | | 1 | | Initial output is 0 | Output 0 on compare-match |
| | | 1 | 0 | | Output 1 on compare-match | |
| | | | 1 | | Toggle output on compare-match | |
| | 1 | 0 | 0 | Output disabled | | |
| | | | 1 | Initial output is 1 | Output 0 on compare-match | |
| | | 1 | 0 | Output 1 on compare-match | | |
| | | | 1 | Toggle output on compare-match | | |
| 1 | 0 | 0 | 0 | TGR0D is an input capture register | Capture input | Input capture on rising edge |
| | | | 1 | | source is the TIOC0D pin | Input capture on falling edge |
| | | 1 | 0 | | Input capture on both edges | |
| | | | 1 | | | |
| | 1 | 0 | 0 | Capture input | Input capture on TCNT1 count | |
| | | | 1 | source is channel 1/ count clock | up/count down | |
| | | 1 | 0 | | | |
| | | | 1 | | | |

Note: When the BFB bit of TMDR0 is set to 1 and TGR0D is being used as a buffer register, these settings become ineffective and input capture/output compares do not occur.

- Bits 3–0—I/O Control C3–C0 (IOC3–IOC0): These bits set the TGR0C register function.

| Bit 3: IOC3 | Bit 2: IOC2 | Bit 1: IOC1 | Bit 0: IOC0 | Description | | |
|----------------|----------------|----------------|----------------|---|--|------------------------------|
| 0 | 0 | 0 | 0 | TGR0C is an output compare register | Output disabled (initial value) | |
| | | | 1 | | Initial output is 0 | Output 0 on compare-match |
| | | 1 | 0 | | Output 1 on compare-match | |
| | | | 1 | | Toggle output on compare-match | |
| | 1 | 0 | 0 | Output disabled | Initial output is 1 | Output 0 on compare-match |
| | | | 1 | | Output 1 on compare-match | |
| | | 1 | 0 | | Output 1 on compare-match | |
| | | | 1 | | Toggle output on compare-match | |
| 1 | 0 | 0 | 0 | TGR0C is an input capture register | Capture input source is the TIOC0C pin | Input capture on rising edge |
| | | | 1 | | Input capture on falling edge | |
| | | 1 | 0 | | Input capture on both edges | |
| | | | 1 | | | |
| | 1 | 0 | 0 | Capture input source is channel 1/ count clock | Input capture on TCNT1 count up/count down | |
| | | | 1 | | | |
| | | 1 | 0 | | | |
| | | | 1 | | | |

Note: When the BFA bit of TMDR0 is set to 1 and TGR0C is being used as a buffer register, these settings become ineffective and input capture/output compares do not occur.

Channel 1 (TIOR1 Register)

- Bits 7–4—I/O Control B3–B0 (IOB3–IOB0): These bits set the TGR1B register function.

| Bit 7: IOB3 | Bit 6: IOB2 | Bit 5: IOB1 | Bit 4: IOB0 | Description | |
|----------------|----------------|----------------|----------------|--|---------------------------------|
| 0 | 0 | 0 | 0 | TGR1B | Output disabled (initial value) |
| | | | 1 | is an output | Initial output is 0 |
| | | 1 | 0 | compare | Output 0 on compare-match |
| | | | 1 | register | Output 1 on compare-match |
| | 1 | 0 | 0 | | Toggle output on compare-match |
| | | | 1 | | Output disabled |
| | | 1 | 0 | | Initial output is 1 |
| | | | 1 | | Output 0 on compare-match |
| 1 | 0 | 0 | 0 | TGR1B | Capture input |
| | | | 1 | is an input | source is the |
| | | 1 | 0 | capture | TIOC1B pin |
| | | | 1 | register | Input capture on rising edge |
| | 1 | 0 | 0 | | Input capture on falling edge |
| | | | 1 | | Input capture on both edges |
| | | 1 | 0 | | Capture input |
| | | | 1 | | source TGR0C |
| 1 | 0 | | compare/match | Input capture on channel TGR0C | |
| | 1 | | input capture | compare-match/input capture generation | |

- Bits 3–0—I/O Control A3–A0 (IOA3–IOA0): These bits set the TGR1A register function.

| Bit 3: IOA3 | Bit 2: IOA2 | Bit 1: IOA1 | Bit 0: IOA0 | Description | | |
|----------------|----------------|----------------|----------------|--|--|---|
| 0 | 0 | 0 | 0 | TGR1A is an output compare register | Output disabled (initial value) | |
| | | | 1 | | Initial output is 0 | Output 0 on compare-match |
| | | 1 | 0 | Output 1 on compare-match | | |
| | | | 1 | Toggle output on compare-match | | |
| | 1 | 0 | 0 | TGR1A is an input capture register | Output disabled | |
| | | | 1 | | Initial output is 1 | Output 0 on compare-match |
| | | 1 | 0 | Output 1 on compare-match | | |
| | | | 1 | Toggle output on compare-match | | |
| 1 | 0 | 0 | 0 | TGR1A is an input capture register | Capture input source is the TIOC1A pin | Input capture on rising edge |
| | | | 1 | | Input capture on falling edge | |
| | | 1 | 0 | Input capture on both edges | | |
| | | | 1 | | | |
| | 1 | 0 | 0 | TGR0A is an input capture register | Capture input source is TGR0A compare- match/input capture | Input capture on channel 0/TGR0A compare-match/input capture generation |
| | | | 1 | | | |
| | | 1 | 0 | | | |
| | | | 1 | | | |

Channel 2 (TIOR2 Register)

- Bits 7–4—I/O Control B3–B0 (IOB3–IOB0): These bits set the TGR2B register function.

| Bit 7: IOB3 | Bit 6: IOB2 | Bit 5: IOB1 | Bit 4: IOB0 | Description | | | |
|----------------|----------------|----------------|--------------------------------|--|---|--|-------------------------------|
| 0 | 0 | 0 | 0 | TGR2B is an output compare register | Output disabled (initial value) | | |
| | | | 1 | | Initial output is 0 | Output 0 on compare-match | |
| | | | 1 | | Output 1 on compare-match | | |
| | | 1 | Toggle output on compare-match | | | | |
| | | 1 | 0 | | 0 | Output disabled | |
| | | | 1 | | Initial output is 1 | Output 0 on compare-match | |
| | 1 | | Output 1 on compare-match | | | | |
| | 1 | 0 | 0 | 0 | TGR2B is an input capture register | Capture input source is the TIOC2B pin | |
| | | | | 1 | | Input capture on rising edge | Input capture on falling edge |
| | | | | 1 | | Input capture on both edges | |
| | | 1 | 0 | 0 | | Input capture on rising edge | |
| | | | 1 | Input capture on falling edge | | | |
| 1 | | | Input capture on both edges | | | | |

- Bits 3–0—I/O Control A3–A0 (IOA3–IOA0): These bits set the TGR2A register function.

| Bit 3: IOA3 | Bit 2: IOA2 | Bit 1: IOA1 | Bit 0: IOA0 | Description | |
|----------------|----------------|----------------|-----------------------------|--------------|---------------------------------|
| 0 | 0 | 0 | 0 | TGR2A | Output disabled (initial value) |
| | | | 1 | is an output | Initial output is 0 |
| | | 1 | 0 | compare | Output 0 on compare-match |
| | | | 1 | register | Output 1 on compare-match |
| | 1 | 0 | 0 | | Toggle output on compare-match |
| | | | 1 | | Output disabled |
| | | 1 | 0 | | Initial output is 1 |
| | | | 1 | | Output 0 on compare-match |
| 1 | 0 | 0 | 0 | TGR2A | Capture input |
| | | | 1 | is an input | source is the |
| | | 1 | 0 | capture | TIOC2A pin |
| | | | 1 | register | Input capture on rising edge |
| | 1 | 0 | 0 | | Input capture on falling edge |
| | | | 1 | | Input capture on both edges |
| | | 1 | 0 | | Input capture on rising edge |
| | | | 1 | | Input capture on falling edge |
| 1 | 0 | | Input capture on both edges | | |
| | 1 | | | | |

10.2.4 Timer Interrupt Enable Register (TIER)

The TIER is an 8-bit register that controls the enable/disable of interrupt requests for each channel. The MTU has three TIER registers, one each for channel. TIER is initialized to H'40 by a reset or by standby mode.

Channel 0: TIER0

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|------|---|---|-------|-------|-------|-------|-------|
| | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA |
| Initial value: | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R | R | R/W | R/W | R/W | R/W | R/W |

Channels 1, 2: TIER1, TIER2

| | | | | | | | | |
|----------------|------|---|-------|-------|---|---|-------|-------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA |
| Initial value: | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R | R/W | R/W | R | R | R/W | R/W |

- Bit 7—A/D Conversion Start Request Enable (TTGE): Enables or disables generation of an A/D conversion start request by a TGRA register input capture/compare-match.

Bit 7: TTGE

| | Description |
|---|---|
| 0 | Disable A/D conversion start requests (initial value) |
| 1 | Enable A/D conversion start request generation |

- Bit 6—Reserved: This bit always read as 1. The write value should always be 1.
- Bit 5—Underflow Interrupt Enable (TCIEU): Enables or disables interrupt requests when the underflow flag (TCFU) of the channel 1, 2 timer status register (TSR) is set to 1.
This bit is reserved for channel 0. It always reads as 0. The write value should always be 1.

Bit 5: TCIEU

| | Description |
|---|---|
| 0 | Disable UDF interrupt requests (TCIU) (initial value) |
| 1 | Enable UDF interrupt requests (TCIU) |

- Bit 4—Overflow Interrupt Enable (TCIEV): Enables or disables interrupt requests when the overflow flag TCFV of the timer status register (TSR) is set to 1.

Bit 4: TCIEV

| | Description |
|---|--|
| 0 | Disable TCFV interrupt requests (TCIV) (initial value) |
| 1 | Enable TCFV interrupt requests (TCIV) |

- Bit 3—TGR Interrupt Enable D (TGIED): Enables or disables interrupt TGFD requests when the TGFD bit of the channel 0 TSR register is set to 1.

This bit is reserved for channels 1 and 2. It always reads as 1. The write value should always be 1.

Bit 3: TGIED

| | Description |
|---|---|
| 0 | Disable interrupt requests (TGID) due to the TGFD bit (initial value) |
| 1 | Enable interrupt requests (TGID) due to the TGFD bit |

- Bit 2—TGR Interrupt Enable C (TGIEC): Enables or disables TGFC interrupt requests when the TGFC bit of the Channel 0 TSR register is set to 1.

This bit is reserved for channels 1 and 2. It always reads as 1. The write value should always be 1.

| Bit 2: TGIEC | Description |
|--------------|---|
| 0 | Disable interrupt requests (TGIC) due to the TGFC bit (initial value) |
| 1 | Enable interrupt requests (TGIC) due to the TGFC bit |

- Bit 1—TGR Interrupt Enable B (TGIEB): Enables or disables TGFB interrupt requests when the TGFB bit of the TSR register is set to 1.

| Bit 1: TGIEB | Description |
|--------------|---|
| 0 | Disable interrupt requests (TGIB) due to the TGFB bit (initial value) |
| 1 | Enable interrupt requests (TGIB) due to the TGFB bit |

- Bit 0—TGR Interrupt Enable A (TGIEA): Enables or disables TGFA interrupt requests when the TGFA bit of the TSR register is set to 1.

| Bit 0: TGIEA | Description |
|--------------|---|
| 0 | Disable interrupt requests (TGIA) due to the TGFA bit (initial value) |
| 1 | Enable interrupt requests (TGIA) due to the TGFA bit |

10.2.5 Timer Status Register (TSR)

The timer status register (TSR) is an 8-bit register that indicates the status of each channel. The MTU has five TSR registers, one each for channel. TSR is initialized to H'C0 by a power-on reset or by standby mode.

Channel 0: TSR0

| | | | | | | | | |
|----------------|---|---|---|--------|--------|--------|--------|--------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA |
| Initial value: | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R/(W)* | R/(W)* | R/(W)* | R/(W)* | R/(W)* |

Note: Only 0 writes to clear the flags are possible.

Channels 1, 2: TSR1, TSR2

| | | | | | | | | |
|----------------|------|---|--------|--------|---|---|--------|--------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA |
| Initial value: | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R/(W)* | R/(W)* | R | R | R/(W)* | R/(W)* |

Note: Only 0 writes to clear the flags are possible.

- Bit 7—Count Direction Flag (TCFD): This status flag indicates the count direction of the channel 1, 2 TCNT counters.

This bit is reserved in channel 0. This bit always reads as 1. The write value should always be 1.

| Bit 7: TCFD | Description |
|-------------|--------------------------------|
| 0 | TCNT counts down |
| 1 | TCNT counts up (initial value) |

- Bit 6—Reserved: This bit always reads as 1. The write value should always be 1.
- Bit 5—Underflow Flag (TCFU): This status flag indicates the occurrence of a channel 1, 2 TCNT counter underflow.

This bit is reserved in channel 0. This bit always reads as 0. The write value should always be 0.

| Bit 5: TCFU | Description |
|-------------|--|
| 0 | Clear condition: With TCFU=1, a 0 write to TCFU after reading it (initial value) |
| 1 | Set condition: When the TCNT value underflows (H'0000 → H'FFFF) |

- Bit 4—Overflow Flag (TCFV): This status flag indicates the occurrence of a TCNT counter overflow.

| Bit 4: TCFV | Description |
|-------------|--|
| 0 | Clear condition: With TCFV =1, a 0 write to TCFV after reading it**1 (initial value) |
| 1 | Set condition: When the TCNT value overflows (H'FFFF → H'0000)**2 |

- Bit 3—Input Capture/Output Compare Flag D (TGFD): This status flag indicates the occurrence of a channel 0 TGRD register input capture or compare-match.

This bit is reserved in channels 1 and 2. It always reads as 0. The write value should always be 0.

| Bit 3: TGFD | Description |
|--------------------|---|
| 0 | Clear condition: With TGFD = 1, a 0 write to TGFD following a read (initial value) |
| 1 | Set conditions: <ul style="list-style-type: none">• When TGRD is functioning as an output compare register (TCNT = TGRD)• When TGRD is functioning as input capture (the TCNT value is sent to TGRD by the input capture signal) |

- Bit 2—Input Capture/Output Compare Flag C (TGFC): This status flag indicates the occurrence of a Channel 0 TGRC register input capture or compare-match.
This bit is reserved for channels 1 and 2. It always reads as 0. The write value should always be 0.

| Bit 2: TGFC | Description |
|--------------------|---|
| 0 | Clear condition: With TGFC = 1, a 0 write to TGFC following a read (initial value) |
| 1 | Set conditions: <ul style="list-style-type: none">• When TGRC is functioning as an output compare register (TCNT = TGRC)• When TGRC is functioning as input capture (the TCNT value is sent to TGRC by the input capture signal) |

- Bit 1—Input Capture/Output Compare Flag B (TGFB): This status flag indicates the occurrence of a TGRB register input capture or compare-match.

| Bit 1: TGFB | Description |
|--------------------|---|
| 0 | Clear condition: With TGFB = 1, a 0 write to TGFB following a read (initial value) |
| 1 | Set conditions: <ul style="list-style-type: none">• When TGRB is functioning as an output compare register (TCNT = TGRB)• When TGRB is functioning as input capture (the TCNT value is sent to TGRB by the input capture signal) |

- **Bit 0—Input Capture/Output Compare Flag A (TGFA):** This status flag indicates the occurrence of a TGRA register input capture or compare-match.

| Bit 0: TGFA | Description |
|-------------|--|
| 0 | Clear condition: With TGFA = 1, a 0 write to TGFA following a read (Cleared by DMAC transfer due to TGFA) (initial value) |
| 1 | Set conditions: <ul style="list-style-type: none"> • When TGRA is functioning as an output compare register (TCNT = TGRA) • When TGRA is functioning as input capture (the TCNT value is sent to TGRA by the input capture signal) |

10.2.6 Timer Counters (TCNT)

The timer counters (TCNT) are 16-bit counters, with one for each channel, for a total of three.

The TCNT are initialized to H'0000 by a power-on reset and when in standby mode. Accessing the TCNT counters in 8-bit units is prohibited. Always access in 16-bit units.

| Channel | Abbreviation | Function |
|---------|--------------|------------------------------|
| 0 | TCNT0 | Increment counter |
| 1 | TCNT1 | Increment/decrement counter* |
| 2 | TCNT2 | Increment/decrement counter* |

Note: Can only be used as an increment/decrement counter in phase counting mode, with other channel overflow/underflow counting. It becomes an increment counter in all other cases.

| | | | | | | | | |
|----------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> |
| 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 |
| | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> |
| 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 |

10.2.7 Timer General Register (TGR)

Each timer general register (TGR) is a 16-bit register that can function as either an output compare register or an input capture register. There are a total of eight TGR, four for channel 0, and two each for channels 1 and 2. The TGRC and TGRD of channel 0 can be set to operate as buffer registers. The TGR register and buffer register combinations are TGRA with TGRC, and TGRB with TGRD.

The TGRs are initialized to H'FFFF by a power-on reset or in standby mode. Accessing of the TGRs in 8-bit units is disabled; they may only be accessed in 16-bit units.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | | | | | | | | |
| 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 |
| 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 |

10.2.8 Timer Start Register (TSTR)

The timer start register (TSTR) is an 8-bit read/write register that starts and stops the timer counters (TCNT) of channels 0–2. TSTR is initialized to H'00 upon power-on reset or standby mode.

| | | | | | | | | |
|----------------|---|---|---|---|---|------|------|------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | CST2 | CST1 | CST0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R/W | R/W | R/W |

- Bits 2–0—Counter Start 4–0 (CST4–CST0): Select the start and stop of the timer counters (TCNT). The counter start to channel and bit to channel correspondence are indicated in the tables below.

| Counter Start | Channel |
|---------------|-------------------|
| CST2 | Channel 2 (TCNT2) |
| CST1 | Channel 1 (TCNT1) |
| CST0 | Channel 0 (TCNT0) |

| Bit n: CSTn | Description |
|-------------|---------------------------------------|
| 0 | TCNTn count is halted (initial value) |
| 1 | TCNTn counts |

Note: n = 2 to 0.

If 0 is written to the CST bit during operation with the TIOC pin in output status, the counter stops, but the TIOC pin output compare output level is maintained. If a write is done to the TIOR register while the CST bit is a 0, the pin output level is updated to the established initial output value.

- Bits 7–3—Reserved: These bits always read as 0. The write value should always be 0.

10.2.9 Timer Synchro Register (TSYR)

The timer synchro register (TSYR) is an 8-bit read/write register that selects independent or synchronous TCNT counter operation for channels 0–2. Channels for which 1 is set in the corresponding bit will be synchronized. TSYR is initialized to H'00 upon power-on reset or standby mode.

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|---|---|---|---|-------|-------|-------|
| | — | — | — | — | — | SYNC2 | SYNC1 | SYNC0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R/W | R/W | R/W |

- Bits 2–0—Timer Synchronization 2–0 (SYNC2–SYNC0): Selects operation independent of, or synchronized to, other channels. Synchronous operation allows synchronous clears due to multiple TCNT synchronous presets and other channel counter clears. A minimum of two channels must have SYNC bits set to 1 for synchronous operation. For synchronization clearing, it is necessary to set the TCNT counter clear sources (the CCLR2–CCLR0 bits of the TCR register), in addition to the SYNC bit. The counter start to channel and bit-to-channel correspondence are indicated in the tables below.

| Counter Start | Channel |
|---------------|-------------------|
| SYNC2 | Channel 2 (TCNT2) |
| SYNC1 | Channel 1 (TCNT1) |
| SYNC0 | Channel 0 (TCNT0) |

| Bit n: SYNCn | Description |
|--------------|---|
| 0 | Timer counter (TCNTn) independent operation (initial value) (TCNTn preset/clear unrelated to other channels) |
| 1 | Timer counter synchronous operation* ¹ TCNTn synchronous preset/ synchronous clear* ² possible |

Note: 1. Minimum of two channel SYNC bits must be set to 1 for synchronous operation.
 2. TCNT counter clear sources (CCLR2–CCLR0 bits of the TCR register) must be set in addition to the SYNC bit in order to have clear synchronization.
 3. n = 2 to 0.

- Bits 7–3—Reserved: These bits always read as 0. The write value should always be 0.

10.3 Bus Master Interface

10.3.1 16-Bit Registers

The timer counters (TCNT) and general registers (TGR) are 16-bit registers. A 16-bit data bus to the bus master enables 16-bit read/writes. 8-bit read/write is not possible. Always access in 16-bit units. Figure 10.2 shows an example of 16-bit register access operation.

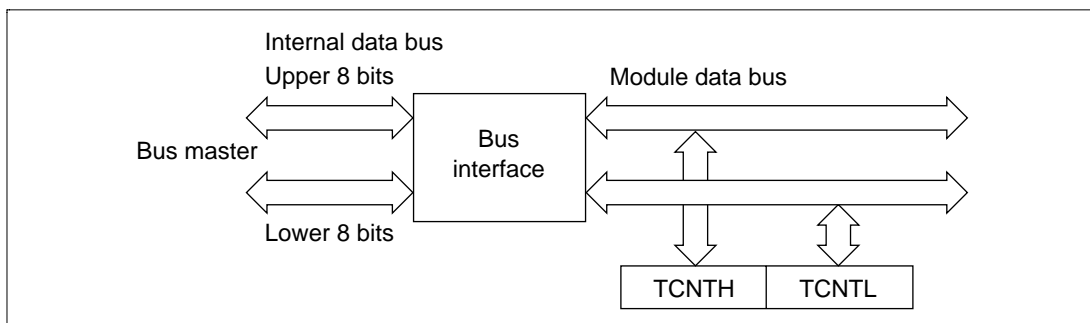


Figure 10.2 16-Bit Register Access Operation (Bus Master ↔ TCNT (16 Bit))

10.3.2 8-Bit Registers

All registers other than the TCNT and general registers (TGR) are 8-bit registers. These are connected to the CPU by a 16-bit data bus, so 16-bit read/writes and as 8-bit read/writes are both possible (figures 10.3 to 10.5).

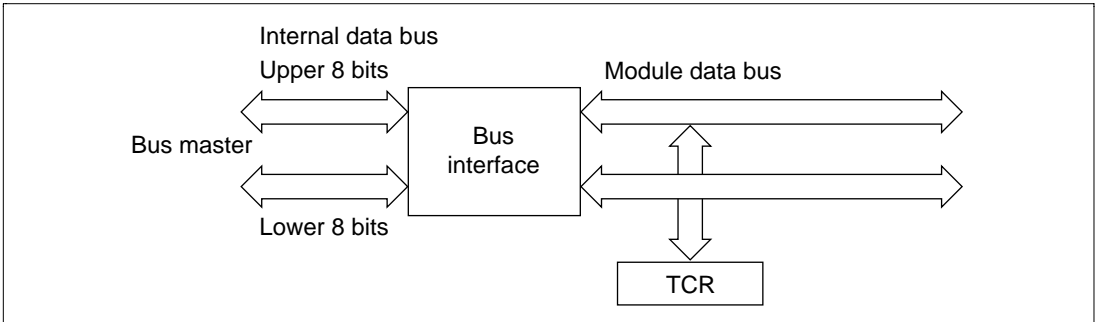


Figure 10.3 8-Bit Register Access Operation (Bus Master ↔ TCR (Upper 8 Bits))

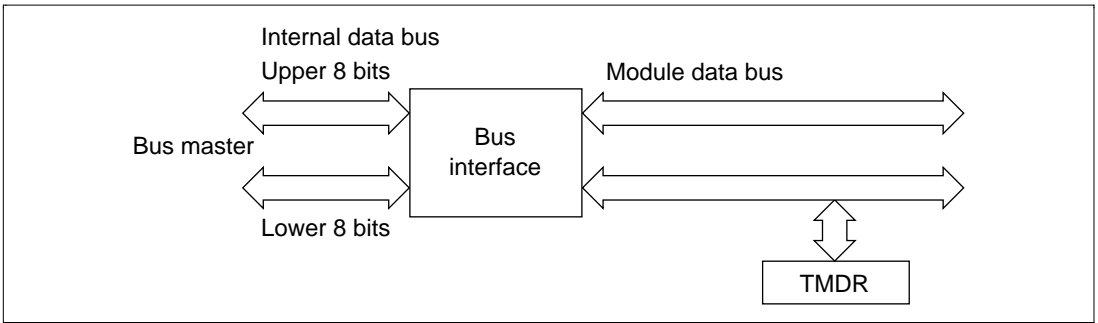


Figure 10.4 8-Bit Register Access Operation (Bus Master ↔ TMDR (Lower 8 Bits))

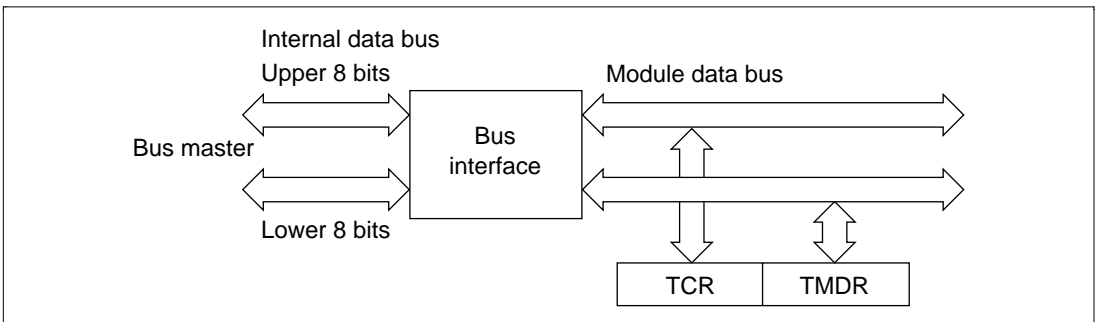


Figure 10.5 8-Bit Register Access Operation (Bus Master ↔ TCR, TMDR (16 Bit))

10.4 Operation

10.4.1 Overview

The operation modes are described below.

Ordinary Operation: Each channel has a timer counter (TCNT) and general register (TGR). The TCNT is an upcounter and can also operate as a free-running counter, periodic counter or external event counter. General registers (TGR) can be used as output compare registers or input capture registers.

Synchronized Operation: The TCNT of a channel set for synchronized operation does a synchronized preset. When any TCNT of a channel operating in the synchronized mode is rewritten, the TCNTs of other channels are simultaneously rewritten as well. The timer synchronization bits of the TSYR registers of multiple channels set for synchronous operation can be set to clear the TCNTs simultaneously.

Buffer Operation: When TGR is an output compare register, the buffer register value of the corresponding channel is transferred to the TGR when a compare-match occurs. When TGR is an input capture register, the TCNT counter value is transferred to the TGR when an input capture occur simultaneously the value previously stored in the TGR is transferred to the buffer register.

Cascade Connection Operation: The channel 1 and channel 2 counters (TCNT1 and TCNT2) can be connected together to operate as a 32-bit counter.

PWM Mode: In PWM mode, a PWM waveform is output. The output level can be set by the TIOR register. Each TGR can be set for PWM waveform output with a duty cycle between 0% and 100%.

Phase Counting Mode: In phase counting mode, the phase differential between two clocks input from the channel 1 and channel 2 external clock input pins is detected and the TCNT counter operates as an up/down counter. In phase counting mode, the corresponding TCLK pins become clock inputs and TCNT functions as an up/down counter. It can be used as a two-phase encoder pulse input.

10.4.2 Basic Functions

Always select MTU external pin set function using the pin function controller (PFC).

Counter Operation: When a start bit (CST0–CST2) in the timer start register (TSTR) is set to 1, the corresponding timer counter (TCNT) starts counting. There are two counting modes: a free-running mode and a periodic mode.

To select the counting operation (figure 10.6):

1. Set bits TPSC2–TPSC0 in the TCR to select the counter clock. At the same time, set bits CKEG1 and CKEG0 in the TCR to select the desired edge of the input clock.
2. To operate as a periodic counter, set the CCLR2–CCLR0 bits in the TCR to select TGR as a clearing source for the TCNT.
3. Set the TGR selected in step 2 as an output compare register using the timer I/O control register (TIOR).
4. Write the desired cycle value in the TGR selected in step 2.
5. Set the CST bit in the TSTR to 1 to start counting.

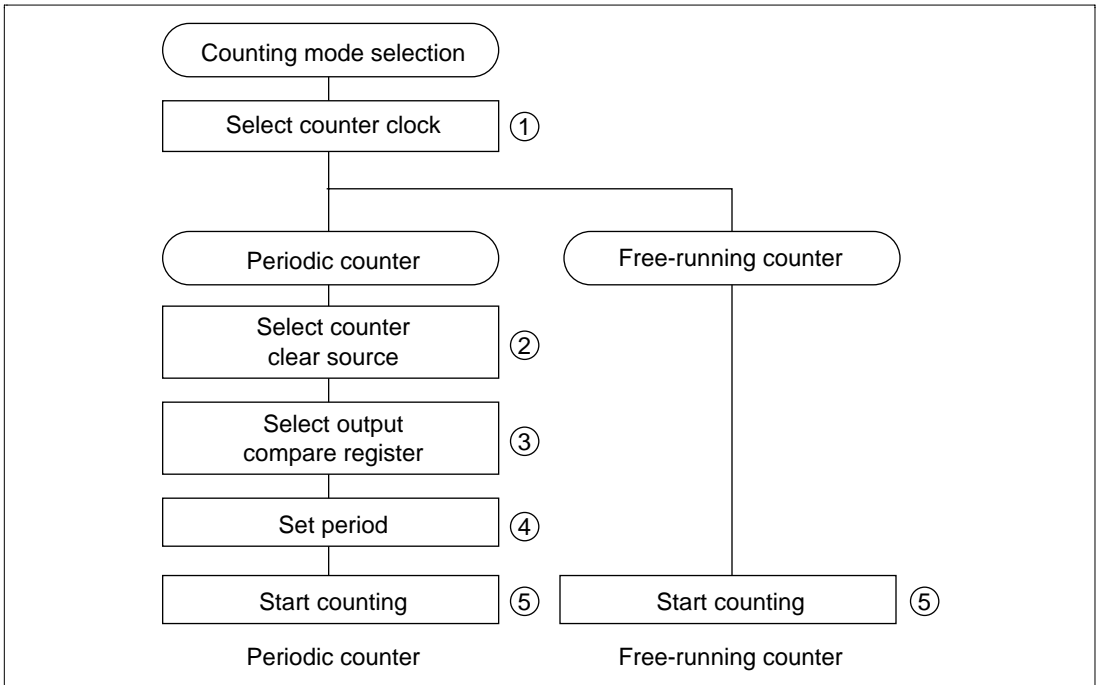


Figure 10.6 Procedure for Selecting the Counting Operation

Free-Running Counter Operation Example: A reset of the MTU timer counters (TCNT) leaves them all in the free-running mode. When a bit in the TSTR is set to 1, the corresponding timer counter operates as a free-running counter and begins to increment. When the count overflows from H'FFFF–H'0000, the TCFV bit in the timer status register (TSR) is set to 1. If the TCIEV bit in the timer's corresponding timer interrupt enable register (TIER) is set to 1, the MTU will make an interrupt request to the interrupt controller. After the TCNT overflows, counting continues from H'0000. Figure 10.7 shows an example of free-running counter operation.

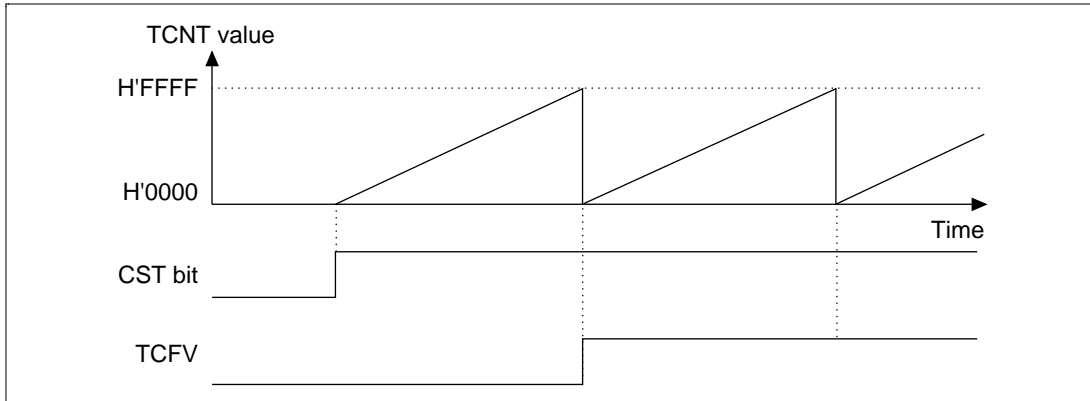


Figure 10.7 Free-Running Counter Operation

Periodic Counter Operation Example: Periodic counter operation is obtained for a given channel's TCNT by selecting compare-match as a TCNT clear source. Set the TGR register for period setting to output compare register and select counter clear upon compare-match using the CCLR2–CCLR0 bits of the timer control register (TCR). After these settings, the TCNT begins incrementing as a periodic counter when the corresponding bit of TSTR is set to 1. When the count matches the TGR register value, the TGF bit in the TSR is set to 1 and the counter is cleared to H'0000. If the TGIE bit of the corresponding TIER is set to 1 at this point, the MTU will make an interrupt request to the interrupt controller. After the compare-match, TCNT continues counting from H'0000. Figure 10.8 shows an example of periodic counting.

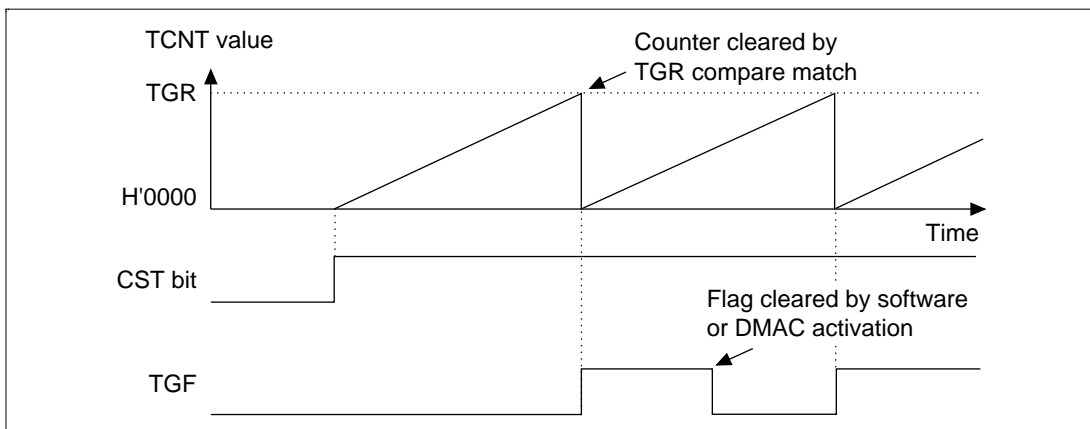


Figure 10.8 Periodic Counter Operation

Compare-Match Waveform Output Function: The MTU can output 0 level, 1 level, or toggle output from the corresponding output pins upon compare-matches.

Procedure for selecting the compare-match waveform output operation (figure 10.9):

1. Set the TIOR to select 0 output or 1 output for the initial value, and 0 output, 1 output, or toggle output for compare-match output. The TIOC pin will output the set initial value until the first compare-match occurs.
2. Set a value in the TGR to select the compare-match timing.
3. Set the CST bit in the TSTR to 1 to start counting.

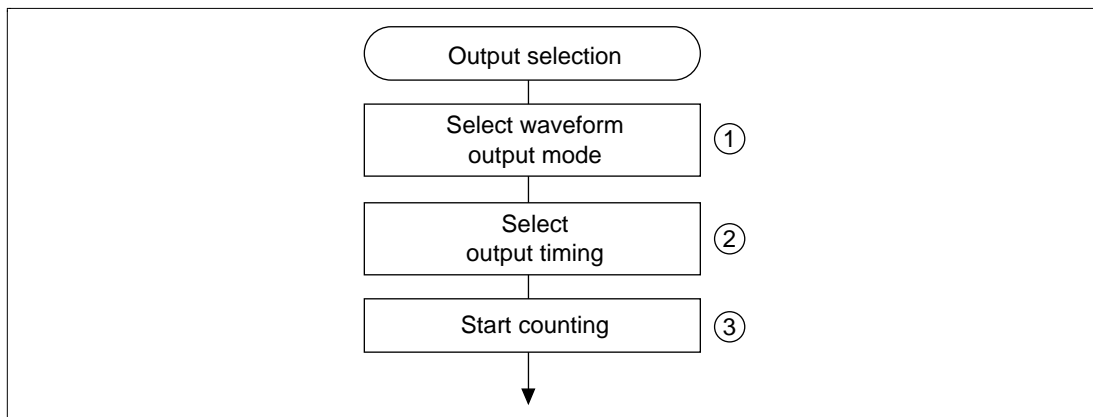


Figure 10.9 Procedure for Selecting Compare Match Waveform Output Operation

Waveform Output Operation (0 Output/1 Output): Figure 10.10 shows 0 output/1 output. In the example, TCNT is a free-running counter, 1 is output upon compare-match A and 0 is output upon compare-match B. When the pin level matches the set level, the pin level does not change.

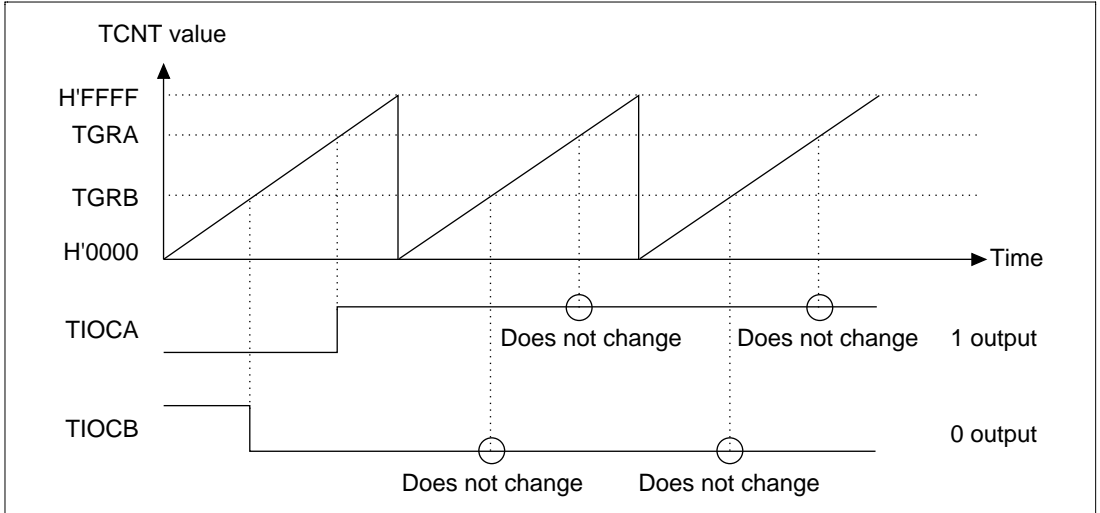


Figure 10.10 Example of 0 Output/1 Output

Waveform Output Operation (Toggle Output): Figure 10.11 shows the toggle output. In the example, the TCNT operates as a periodic counter cleared by compare-match B, with toggle output at both compare-match A and compare-match B.

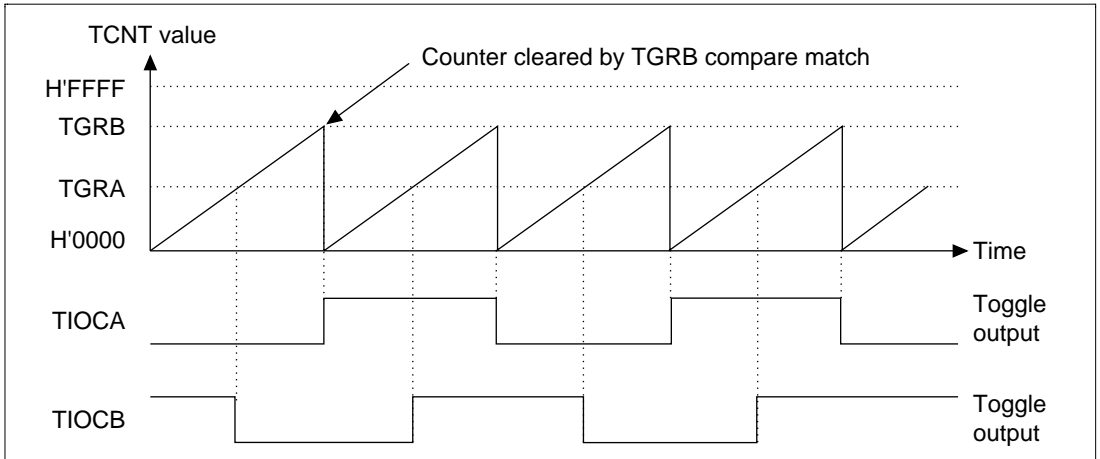


Figure 10.11 Example of Toggle Output

Input Capture Function: In the input capture mode, the TCNT value is transferred into the TGR register when the input edge is detected at the input capture/output compare pin (TIOC).

Detection can take place on the rising edge, falling edge, or both edges. Channels 0 and 1 can use other channel counter input clocks or compare-match signals as input capture sources.

The procedure for selecting the input capture operation (figure 10.12) is:

1. Set the TIOR to select the input capture function of the TGR, then select the input capture source, and rising edge, falling edge, or both edges as the input edge.
2. Set the CST bit in the TSTR to 1 to start the TCNT counting.

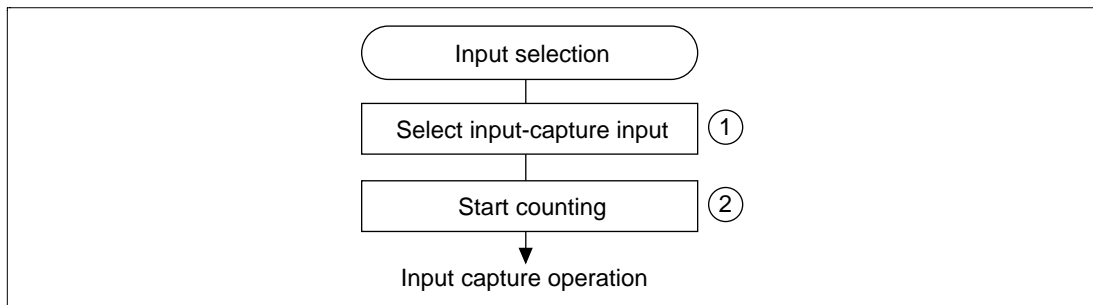


Figure 10.12 Procedure for Selecting Input Capture Operation

Input Capture Operation: Figure 10.13 shows input capture. The falling edge of TIOCB and both edges of TIOCA are selected as input capture input edges. In the example, TCNT is set to clear at the input capture of the TGRB register.

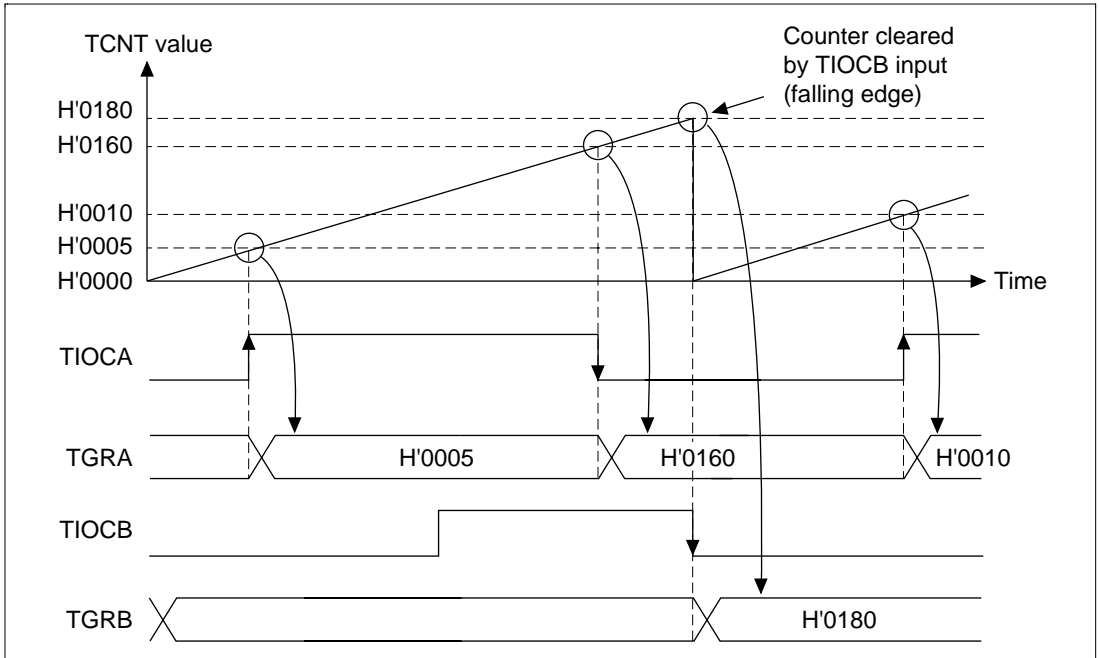


Figure 10.13 Input Capture Operation

10.4.3 Synchronous Operation

In the synchronizing mode, two or more timer counters can be rewritten simultaneously (synchronized preset). Multiple timer counters can also be cleared simultaneously using TCR settings (synchronized clear).

The synchronizing mode can increase the number of TGR registers for a single time base. All three channels can be set for synchronous operation.

Procedure for Selecting the Synchronizing Mode (Figure 10.14):

1. Set 1 in the SYNC bit of the timer synchro register (TSYR) to use the corresponding channel in the synchronizing mode.
2. When a value is written in the TCNT in any of the synchronized channels, the same value is simultaneously written in the TCNT in the other channels.
3. Set the counter to clear with output compare/input capture using bits CCLR2–CCLR0 in the TCR.
4. Set the counter clear source to synchronized clear using the CCLR2–CCLR0 bits of the TCR.
5. Set the CST bits for the corresponding channels in the TSTR to 1 to start counting in the TCNT.

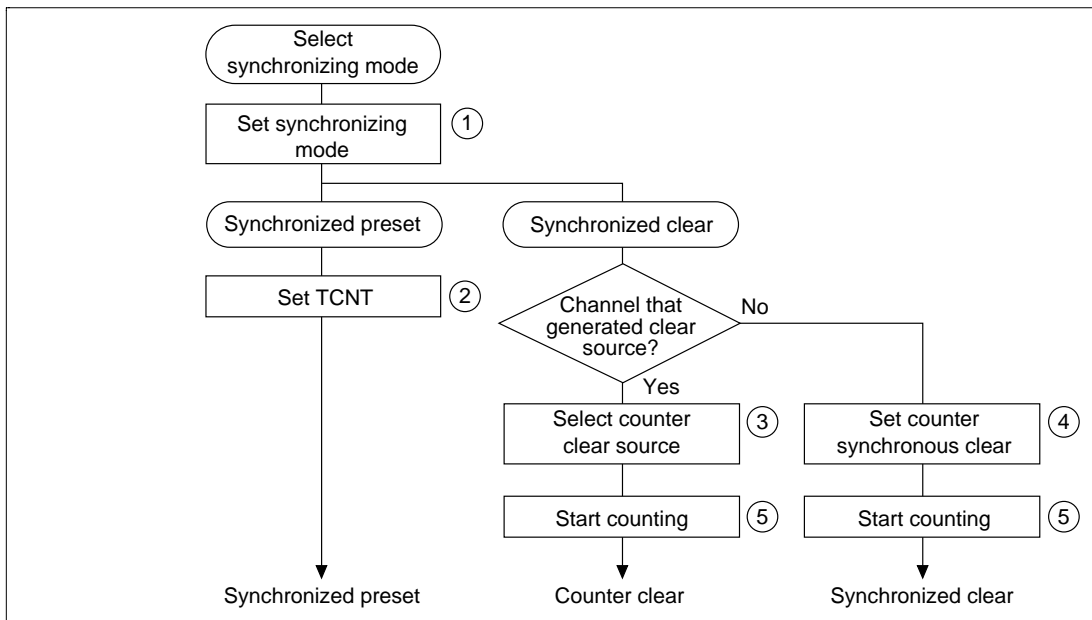


Figure 10.14 Procedure for Selecting Synchronizing Operation

Synchronized Operation: Figure 10.15 shows an example of synchronized operation. Channels 0, 1, and 2 are set to synchronized operation and PWM mode 1. Channel 0 is set for a counter clear upon compare-match with TGR0B. Channels 1 and 2 are set for synchronous counter clears by synchronous presets and TGR0B register compare-matches. Accordingly, a three-phase PWM waveform with the data set in the TGR0B register as its PWM period is output from the TIOC0A, TIOC1A, and TIOC2A pins.

See section 10.4.6, PWM Mode, for details on the PWM mode.

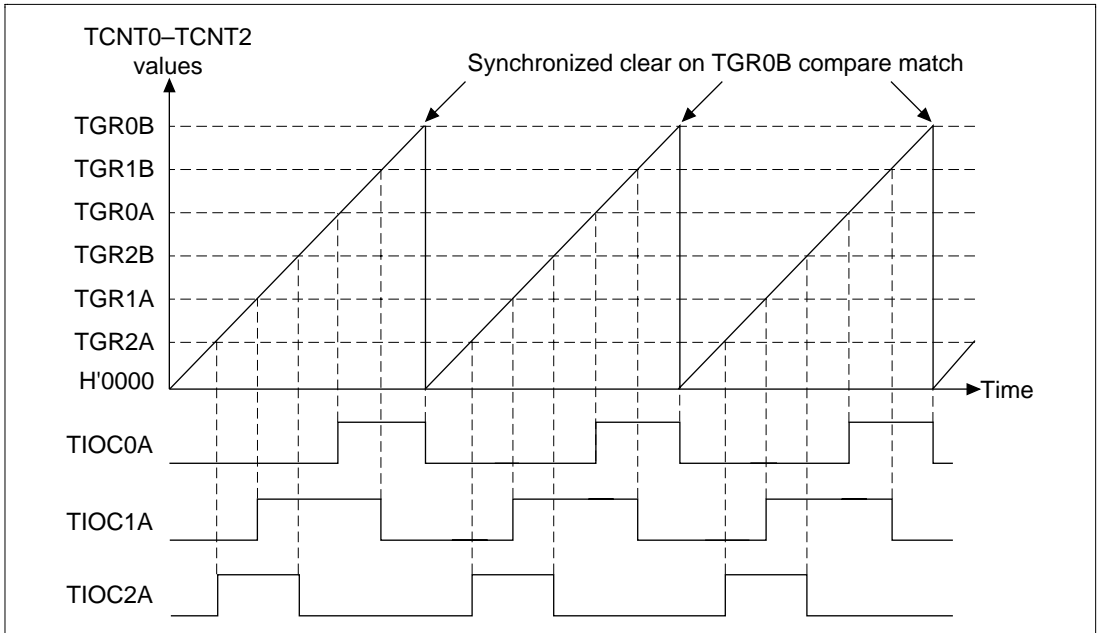


Figure 10.15 Synchronized Operation Example

10.4.4 Buffer Operation

Buffer operation is a function of channels 0, 3, and 4. TGRC and TGRD can be used as buffer registers. Table 10.5 shows the register combinations for buffer operation.

Table 10.5 Register Combinations

| Channel | General Register | Buffer Register |
|---------|------------------|-----------------|
| 0 | TGR0A | TGR0C |
| | TGR0B | TGR0D |

The buffer operation differs, depending on whether the TGR has been set as an input capture register or an output compare register.

When TGR Is an Output Compare Register: When a compare-match occurs, the corresponding channel buffer register value is transferred to the general register. Figure 10.16 shows an example.

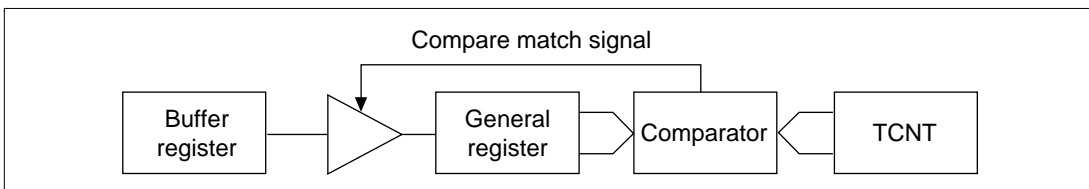


Figure 10.16 Compare Match Buffer Operation

When TGR Is an Input Capture Register: When an input capture occurs, the timer counter (TCNT) value is transferred to the general register (TGR), and the value that had been held up to that time in the TGR is transferred to the buffer register (figure 10.17).

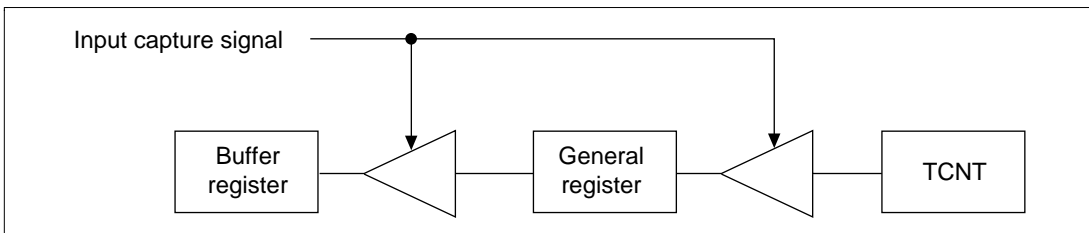


Figure 10.17 Input Capture Buffer Operation

Procedure for Setting Buffer Mode (Figure 10.18):

1. Use the timer I/O control register (TIOR) to set the TGR as either an input capture or output compare register.
2. Use the timer mode register (TMDR) BFA, and BFB bits to set the TGR for buffer mode.
3. Set the CST bit in the TSTR to 1 to start the count operation.

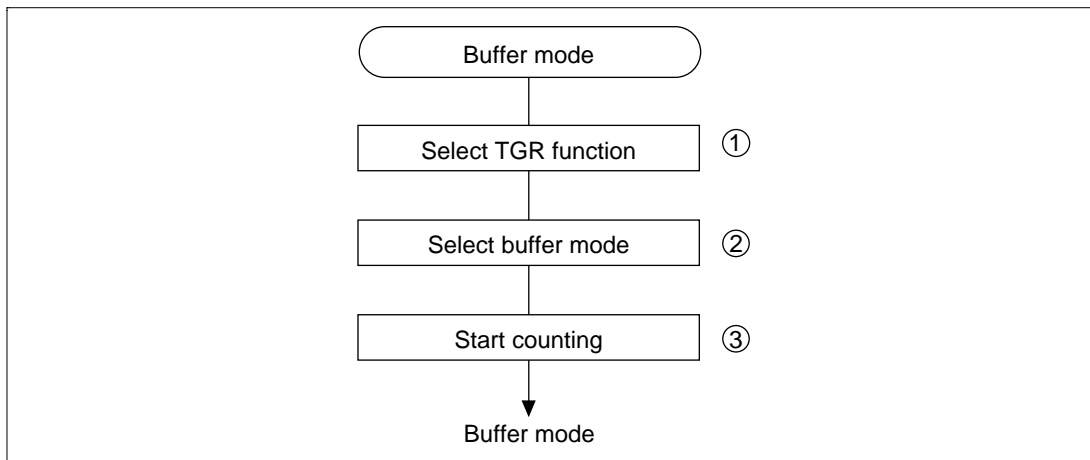


Figure 10.18 Buffer Operation Setting Procedure

Buffer Operation Examples—when TGR Is an Output Compare Register: Figure 10.19 shows an example of channel 0 set to PWM mode 1, and the TGRA and TGRC registers set for buffer operation.

The TCNT counter is cleared by a compare-match B, and the output is a 1 upon compare-match A and 0 output upon compare-match B. Because buffer mode is selected, a compare-match A changes the output, and the buffer register TGRC value is simultaneously transferred to the general register TGRA. This operation is repeated with each occurrence of a compare-match A.

See section 10.4.6, PWM Mode, for details on the PWM mode.

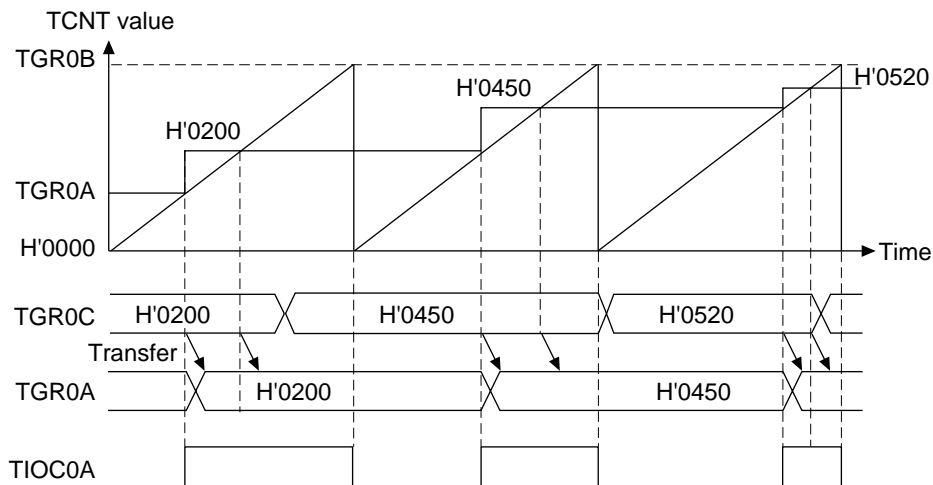


Figure 10.19 Buffer Operation Example (Output Compare Register)

Buffer Operation Examples—when TGR Is an Input Capture Register: Figure 10.20 shows an example of TGRA set as an input capture register with the TGRA and TGRB registers set for buffer operation.

The TCNT counter is cleared by a TGRA register input capture, and the TIOCA pin input capture input edge is selected as both rising and falling edge. Because buffer mode is selected, an input capture A causes the TCNT counter value to be stored in the TGRA register, and the value that was stored in the TGRA up until that time is simultaneously transferred to the TGRC register.

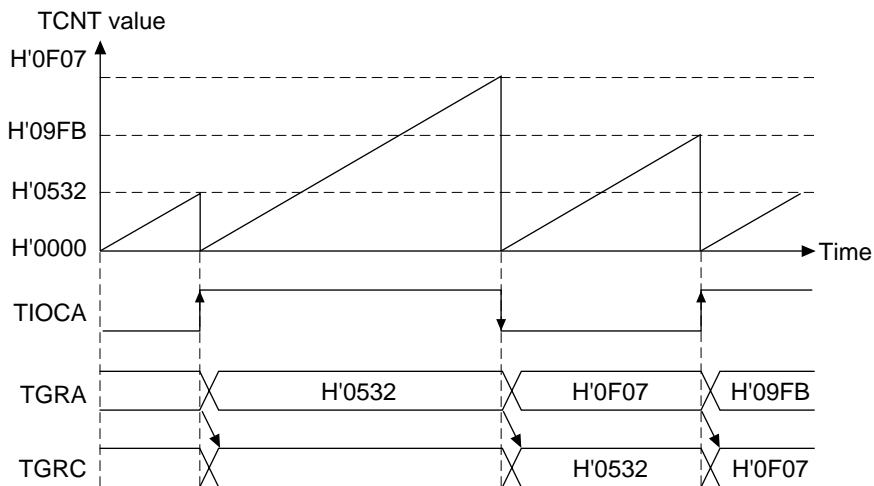


Figure 10.20 Buffer Operation Example (Input Capture Register)

10.4.5 Cascade Connection Mode

Cascade connection mode is a function that connects the 16-bit counters of two channels together to act as a 32-bit counter.

This function operates by using the TPSC2–TPSC0 bits of the TCR register to set the channel 1 counter clock to count by TCNT2 counter overflow/underflow.

Note: When channel 1 is set to phase counting mode, the counter clock settings become ineffective.

Table 10.6 shows the cascade connection combinations.

Table 10.6 Cascade Connection Combinations

| Combination | Upper 16 Bits | Lower 16 Bits |
|----------------------|---------------|---------------|
| Channel 1, channel 2 | TCNT1 | TCNT2 |

Procedure for Setting Cascade Connection Mode (Figure 10.21):

1. Set the TPSC2–TPSC 0 bits of the channel 1 timer control register (TCR) to B'111 to select “count by TCNT2 overflow/underflow.”
2. Set the CST bits corresponding to the upper and lower 16 bits in the TSTR to 1 to start the count operation.

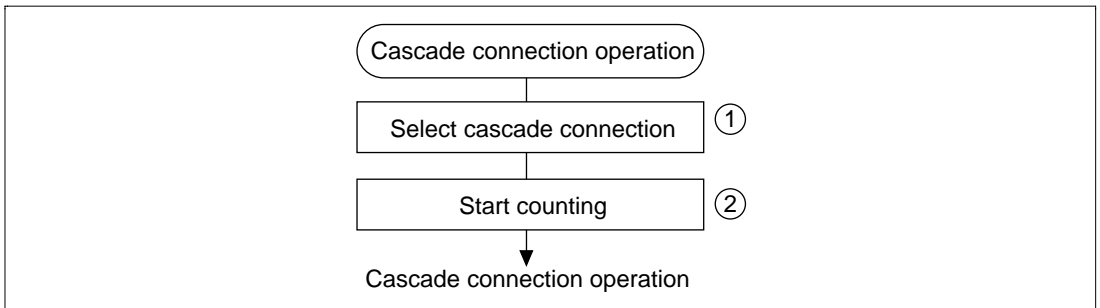


Figure 10.21 Procedure for Selecting Cascade Connection Mode

Cascade Connection Operation Examples—Phase Counting Mode: Figure 10.22 shows an example of operation when the TCNT1 counter is set to count on TCNT2 overflow/underflow and channel 2 is set to phase counting mode.

The TCNT1 counter increments with a TCNT2 counter overflow and decrements with a TCNT2 underflow.

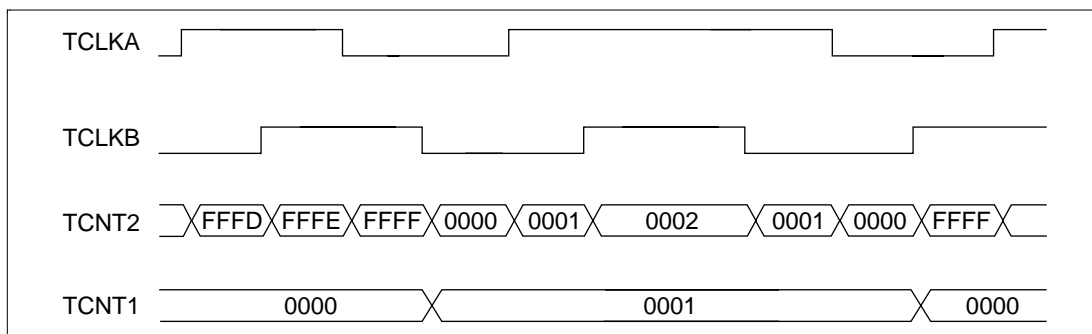


Figure 10.22 Cascade Connection Operation Example (Phase Counting Mode)

10.4.6 PWM Mode

PWM mode outputs the various PWM waveforms from output pins. Output levels of 0 output, 1 output, or toggle output can be selected as the output level for the compare-match of each TGR.

A period can be set for a register by using the TGR compare-match as a counter clear source. All five channels can be independently set to PWM mode. Synchronous operation is also possible.

There are two PWM modes:

- **PWM mode 1**
Generates PWM output using the TGRA and TGRB registers, and TGRC and TGRD registers as pairs. The initial output values are those established in the TGRA and TGRC registers. When the values set in TGR registers being used as a pair are equal, output values will not change even if a compare-match occurs.
A maximum of four-phase PWM output is possible for PWM mode 1.
- **PWM mode 2**
Generates PWM output using one TGR register as a period register and another as a duty cycle register. The output value of each pin upon a counter clear is the initial value established by the TIOR register. When the values set in the period register and duty register are equal, output values will not change even if a compare-match occurs.

Table 10.7 lists the combinations of PWM output pins and registers.

Table 10.7 Combinations of PWM Output Pins and Registers

| Channel | Register | Output Pin | |
|-------------|----------|------------|------------|
| | | PWM Mode 1 | PWM Mode 2 |
| 0 (AB pair) | TGR0A | TIOC0A | TIOC 0A |
| | TGR0B | | TIOC 0B |
| 0 (CD pair) | TGR0C | TIOC0C | TIOC 0C |
| | TGR0D | | TIOC 0D |
| 1 | TGR1A | TIOC1A | TIOC 1A |
| | TGR1B | | TIOC 1B |
| 2 | TGR2A | TIOC2A | TIOC 2A |
| | TGR2B | | TIOC 2B |

Note: PWM output of the period setting TGR is not possible in PWM mode 2.

Procedure for Selecting the PWM Mode (Figure 10.23):

1. Set bits TPSC2–TPSC0 in the TCR to select the counter clock source. At the same time, set bits CKEG1 and CKEG0 in the TCR to select the desired edge of the input clock.
2. Set bits CCLR2 to CCLR0 in the TCR to select the TGR to be used as a counter clear source.
3. Set the period in the TGR selected in step 2, and the duty cycle in another TGR.
4. Using the timer I/O control register (TIOR), set the TGR selected in step 3 to act as an output compare register, and select the initial value and output value.
5. Set the MD3–MD 0 bits in TMDR to select the PWM mode.
6. Set the CST bit in the TSTR to 1 to let the TCNT start counting.

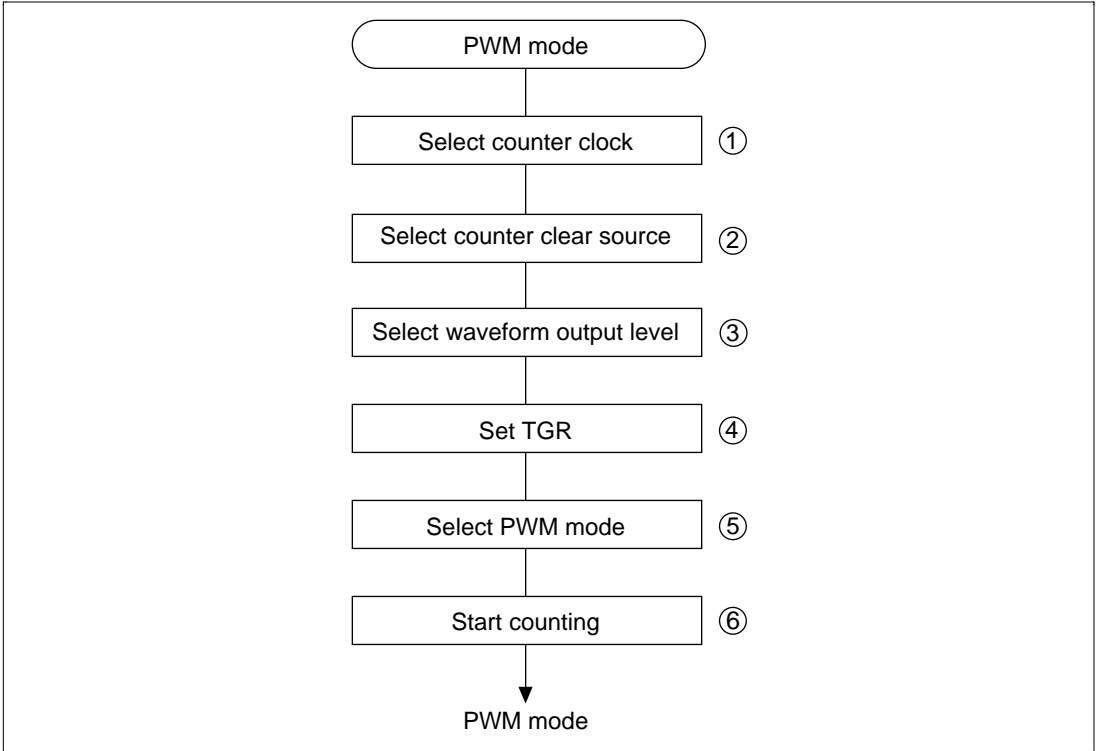


Figure 10.23 Procedure for Selecting the PWM Mode

PWM Mode Operation Examples—PWM Mode 1 (Figure 10.24): A TGRA register compare-match is used as a TCNT counter clear source, the TGRA register initial output value and output compare output value are both 0, and the TGRB register output compare output value is a 1. In this example, the value established in the TGRA register becomes the period and the value established in the TGRB register becomes the duty cycle.

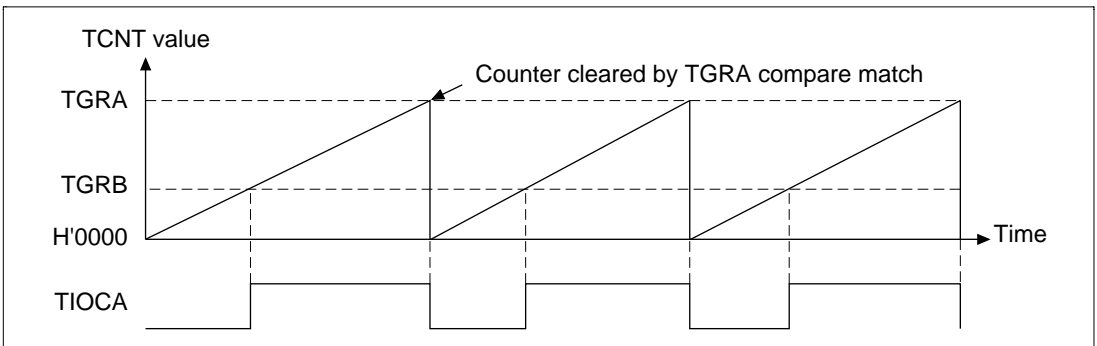


Figure 10.24 PWM Mode Operation Example (Mode 1)

PWM Mode Operation Examples—PWM Mode 2 (Figure 10.25): Channels 0 and 1 are set for synchronous operation, TGR1B register compare-match is used as a TCNT counter clear source, the other TGR register initial output value is 0 and output compare output value is 1, and a 5-phase PWM waveform is output. In this example, the value established in the TGR1B register becomes the period and the value established in the other TGR register becomes the duty cycle.

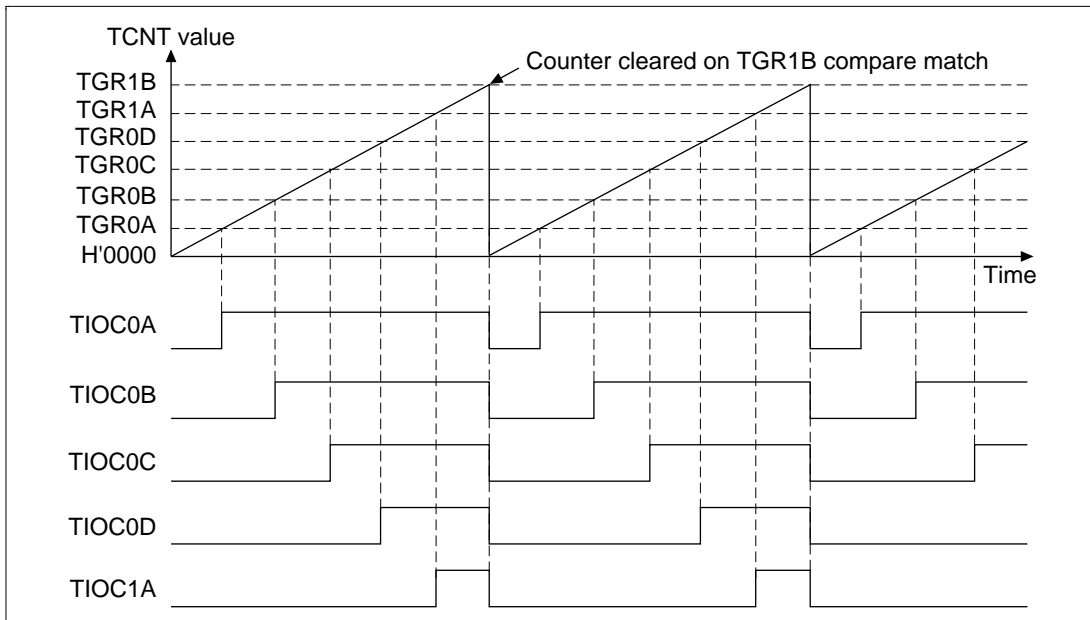


Figure 10.25 PWM Mode Operation Example (Mode 2)

0% Duty Cycle: Figure 10.26 shows an example of a 0% duty cycle PWM waveform output in PWM mode.

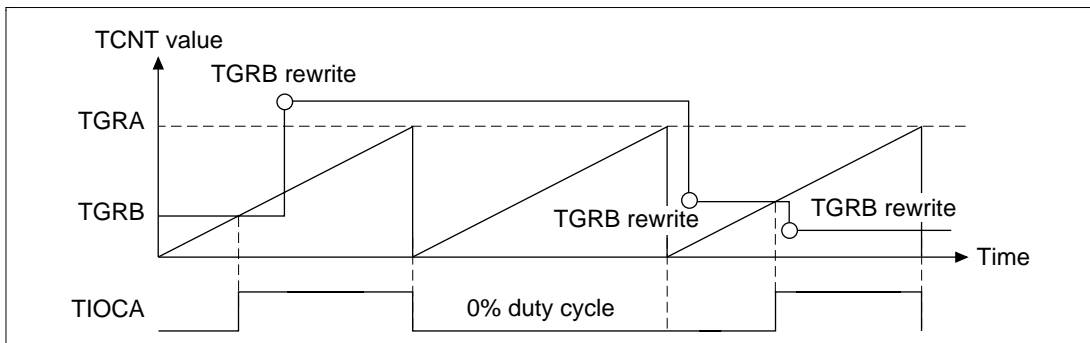


Figure 10.26 PWM Mode Operation Example (0% Duty Cycle)

100% Duty Cycle: Figure 10.27 shows an example of a 100% duty cycle PWM waveform output in PWM mode.

In PWM mode, when setting cycle = duty cycle the output waveform does not change, nor is there a change of waveform for the first pulse immediately after clearing the counter.

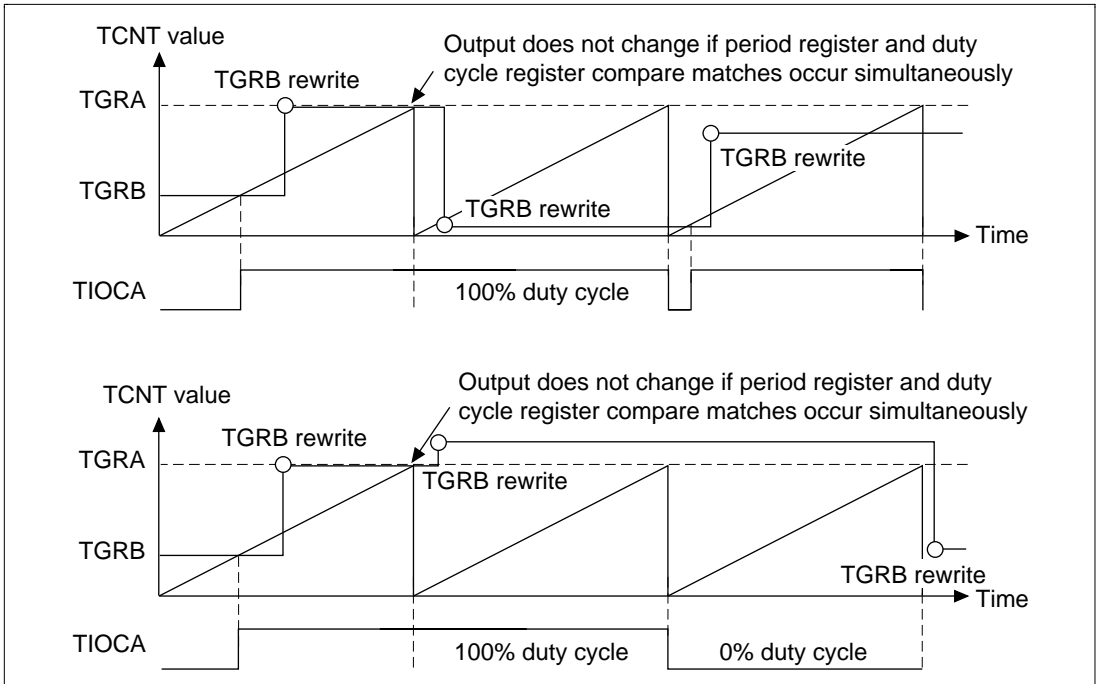


Figure 10.27 PWM Mode Operation Example (100% Duty Cycle)

10.4.7 Phase Counting Mode

The phase counting mode detects the phase differential of two external clock inputs and counts the TCNT counter up or down. This mode can be set for channels 1 and 2.

When set in the phase counting mode, an external clock is selected for the counter input clock, regardless of the settings of the TPSC2–TPSC0 bits of TCR or the CKEG1 and CKEG0 bits. TCNT also becomes an up/down counter. Since the TCR CCLR1/CCLR0 bits, TIOR, TIER, and TGR functions are all enabled, input capture and compare-match functions and interrupt sources can be used.

When the TCNT counter is incrementing, an overflow sets the TSR register TCFV (overflow flag). When it is decrementing, an underflow sets the TCFU (underflow flag).

The TSR register TCFD bit is a count direction flag. Read the TCFD flag to confirm whether the TCNT is incrementing or decrementing.

Table 10.8 shows the correspondence between channels and external clock pins.

Table 10.8 Phase Counting Mode Clock Input Pins

| Channel | A Phase Input Pin | B Phase Input Pin |
|---------|-------------------|-------------------|
| 1 | TCLKA | TCLKB |
| 2 | TCLKC | TCLKD |

Procedure for Selecting the Phase Counting Mode (Figure 10.28):

1. Set the MD3–MD0 bits of the timer mode register (TMDR) to select the phase counting mode.
2. Set the CST bit of the timer start register (TSTR) to 1 to start the count.

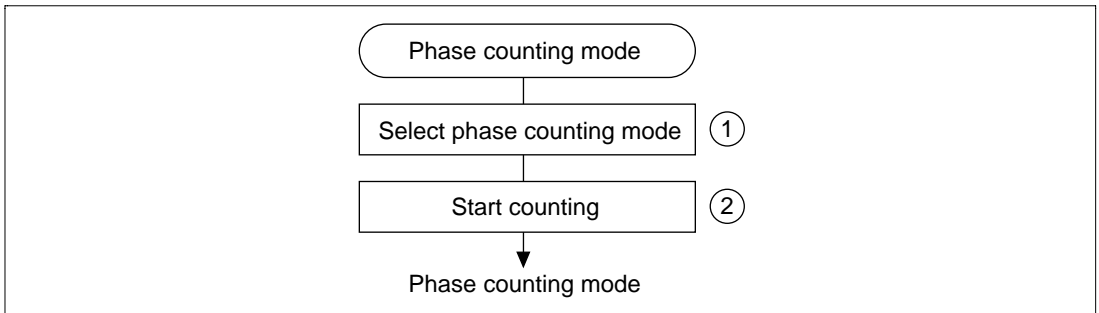


Figure 10.28 Procedure for Selecting the Phase Counting Mode

Phase Counting Operation Examples: The phase counting mode uses the phase difference between two external clocks to increment/decrement the TCNT counter. There are 4 modes, depending on the count conditions.

Phase Counting Mode 1: Figure 10.29 shows an example of phase counting mode 1 operation. Table 10.9 lists the up counting and down counting conditions for the TCNT.

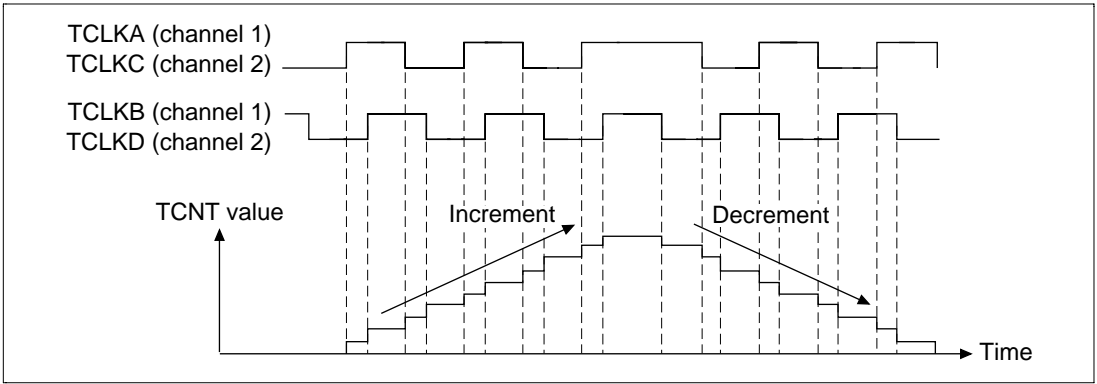


Figure 10.29 Phase Counting Mode 1 Operation

Table 10.9 Phase Count Mode 1 Up/Down Counting Conditions

| TCLKA (Channel 1) TCLKC (Channel 2) | TCLKB (Channel 1) TCLKD (Channel 2) | Operation |
|--|--|-----------|
| 1 (high level) | Rising edge | Increment |
| 0 (low level) | Falling edge | |
| Rising edge | 0 (low level) | Decrement |
| Falling edge | 1 (high level) | |
| 1 (high level) | Falling edge | Decrement |
| 0 (low level) | Rising edge | |
| Rising edge | 1 (high level) | Decrement |
| Falling edge | 0 (low level) | |

Phase Count Mode 2: Figure 10.30 shows an example of phase counting mode 2 operation. Table 10.10 lists the up counting and down counting conditions for the TCNT.

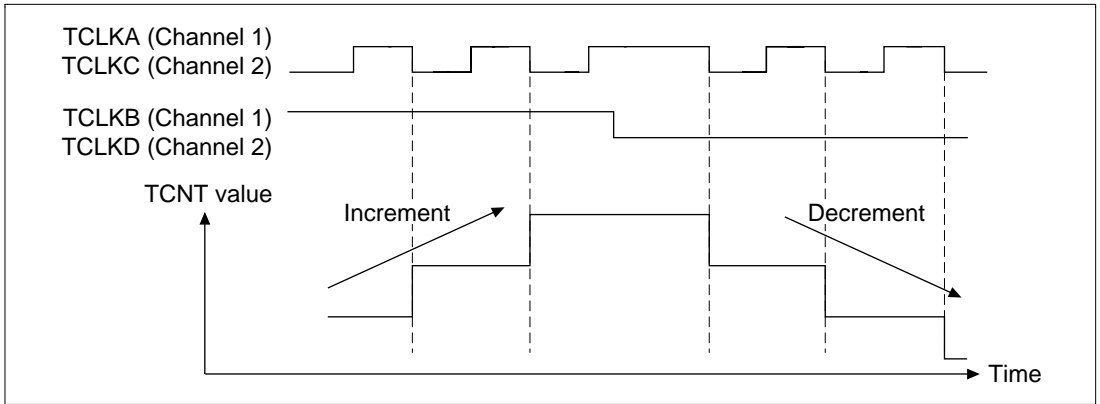


Figure 10.30 Phase Counting Mode 2 Operation

Table 10.10 Phase Count Mode 2 Up/Down Counting Conditions

| TCLKA (Channel 1) TCLKC (Channel 2) | TCLKB (Channel 1) TCLKD (Channel 2) | Operation |
|--|--|-----------------------------|
| 1 (high level) | Rising edge | Does not count (don't care) |
| 0 (low level) | Falling edge | Does not count (don't care) |
| Rising edge | 0 (low level) | Does not count (don't care) |
| Falling edge | 1 (high level) | Increment |
| 1 (high level) | Falling edge | Does not count (don't care) |
| 0 (low level) | Rising edge | Does not count (don't care) |
| Rising edge | 1 (high level) | Does not count (don't care) |
| Falling edge | 0 (low level) | Decrement |

Phase Count Mode 3: Figure 10.31 shows an example of phase counting mode 3 operation. Table 10.11 lists the up counting and down counting conditions for the TCNT.

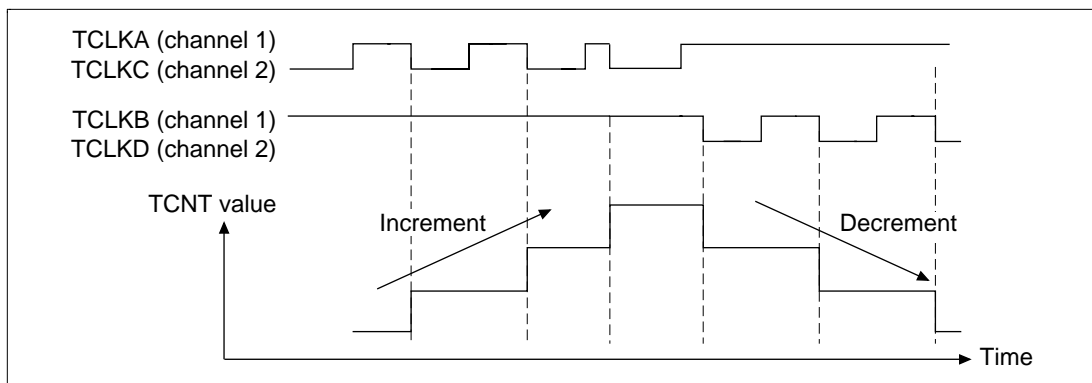


Figure 10.31 Phase Counting Mode 3 Operation

Table 10.11 Phase Count Mode 3 Up/Down Counting Conditions

| TCLKA (Channel 1) TCLKC (Channel 2) | TCLKB (Channel 1) TCLKD (Channel 2) | Operation |
|--|--|-----------------------------|
| 1 (high level) | Rising edge | Does not count (don't care) |
| 0 (low level) | Falling edge | Does not count (don't care) |
| Rising edge | 0 (low level) | Does not count (don't care) |
| Falling edge | 1 (high level) | Increment |
| 1 (high level) | Falling edge | Decrement |
| 0 (low level) | Rising edge | Does not count (don't care) |
| Rising edge | 1 (high level) | Does not count (don't care) |
| Falling edge | 0 (low level) | Does not count (don't care) |

Phase Count Mode 4: Figure 10.32 shows an example of phase counting mode 4 operation. Table 10.12 lists the up counting and down counting conditions for the TCNT.

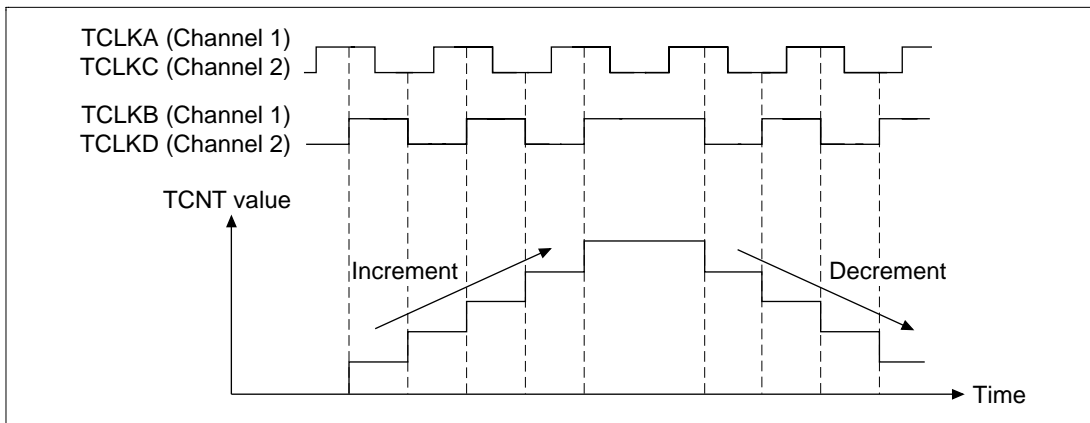


Figure 10.32 Phase Counting Mode 4 Operation

Table 10.12 Phase Count Mode 4 Up/Down Counting Conditions

| TCLKA (Channel 1) TCLKC (Channel 2) | TCLKB (Channel 1) TCLKD (Channel 2) | Operation |
|--|--|-----------------------------|
| 1 (high level) | Rising edge | Increment |
| 0 (low level) | Falling edge | |
| Rising edge | 0 (low level) | Does not count (don't care) |
| Falling edge | 1 (high level) | |
| 1 (high level) | Falling edge | Decrement |
| 0 (low level) | Rising edge | |
| Rising edge | 1 (high level) | Does not count (don't care) |
| Falling edge | 0 (low level) | |

Phase Counting Mode Application Example: Figure 10.33 shows an example where channel 1 is set to phase counting mode and is teamed with channel 0 to input a two-phase encoder pulse for a servo motor to accurately detect position and speed.

Channel 1 is set to phase counting mode 1, and the encoder pulse A phase and B phase are input to the TCLKA and TCLKB pins.

Channel 0 is set so that the TCNT counter is cleared on a TGR0C register compare-match, and the TGR0A and TGR0C registers are used with the compare-match function to establish the speed control and position control periods. The TGR0B register is used with the input capture function, and the TGR0B and TGR0D registers are employed for buffer operation. The channel 1 counter

input clock is used as the TGR0B register input capture source, and a pulse width of four times the 2-phase encoder pulse is detected.

The channel 1 TGR1A and TGR1B registers are set for the input capture function, the channel 0 TGR0A and TGR0C register compare-match is used as an input capture source, and all of the control period increment and decrement values are stored.

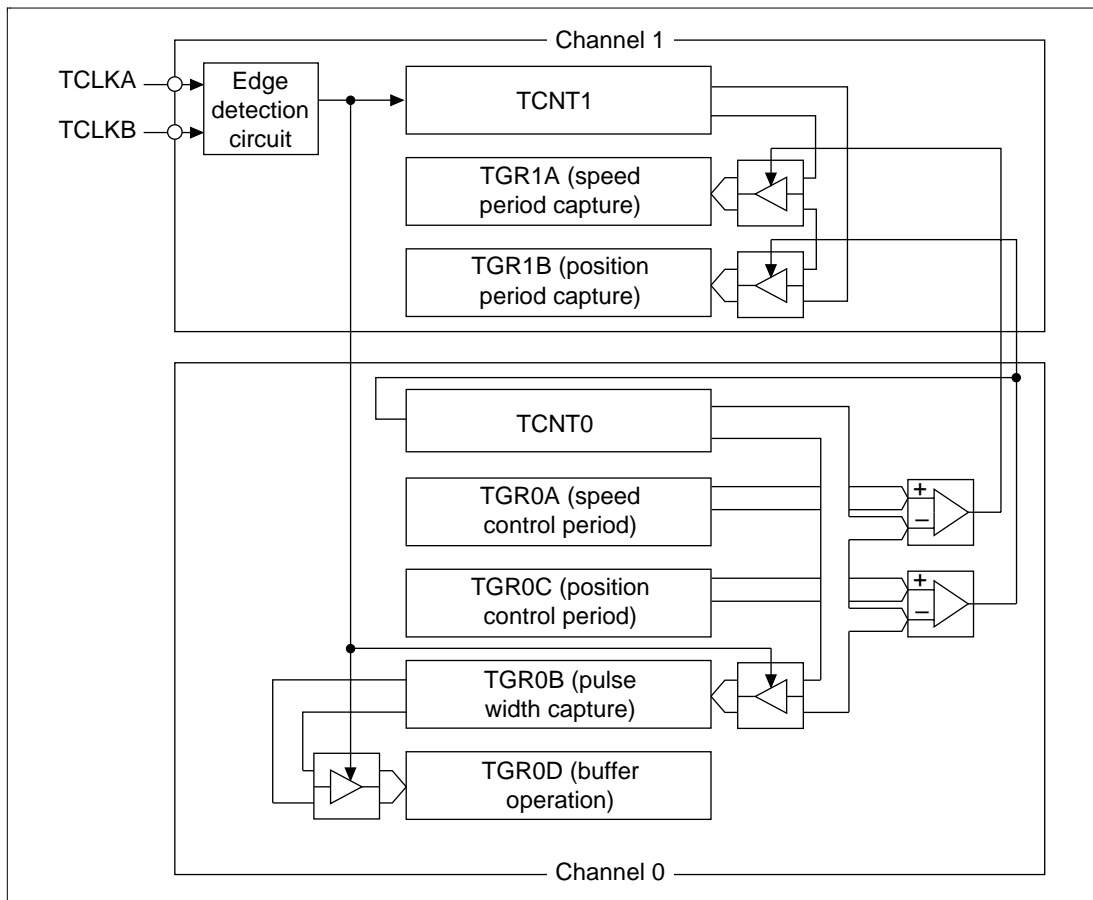


Figure 10.33 Phase Count Mode Application Example

10.5 Interrupts

10.5.1 Interrupt Sources and Priority Ranking

The MTU has three interrupt sources: TGR register compare-match/input captures, TCNT counter overflows and TCNT counter underflows. Because each of these three types of interrupts are allocated its own dedicated status flag and enable/disable bit, the issuing of interrupt request signals to the interrupt controller can be independently enabled or disabled.

When an interrupt source is generated, the corresponding status flag in the timer status register (TSR) is set to 1. If the corresponding enable/disable bit in the timer input enable register (TIER) is set to 1 at this time, the MTU makes an interrupt request of the interrupt controller. The interrupt request is canceled by clearing the status flag to 0.

The channel priority order can be changed with the interrupt controller. The priority ranking within a channel is fixed. For more information, see section 6, Interrupt Controller.

Table 10.13 lists the MTU interrupt sources.

Input Capture/Compare Match Interrupts: If the TGIE bit of the timer input enable register (TIER) is already set to 1 when the TGF flag in the timer status register (TSR) is set to 1 by a TGR register input capture/compare-match of any channel, an interrupt request is sent to the interrupt controller. The interrupt request is canceled by clearing the TGF flag to 0. The MTU has 8 input capture/compare-match interrupts; four for channel 0, and two each for channels 1 and 2.

Overflow Interrupts: If the TCIEV bit of the TIER is already set to 1 when the TCFV flag in the TSR is set to 1 by a TCNT counter overflow of any channel, an interrupt request is sent to the interrupt controller. The interrupt request is canceled by clearing the TCFV flag to 0. The MTU has three overflow interrupts, one for each channel.

Underflow Interrupts: If the TCIEU bit of the TIER is already set to 1 when the TCFU flag in the TSR is set to 1 by a TCNT counter underflow of any channel, an interrupt request is sent to the interrupt controller. The interrupt request is canceled by clearing the TCFU flag to 0. The MTU has two underflow interrupts, one each for channels 1 and 2.

Table 10.13 MTU Interrupt Sources

| Channel | Interrupt Source | Description | DMAC Activation | Priority* |
|---------|------------------|-----------------------------------|-----------------|-----------|
| 0 | TGI0A | TGR0A input capture/compare-match | Yes | High |
| | TGI0B | TGR0B input capture/compare-match | No | |
| | TGI0C | TGR0C input capture/compare-match | No | |
| | TGI0D | TGR0D input capture/compare-match | No | |
| | TCI0V | TCNT0 overflow | No | |
| 1 | TGI1A | TGR1A input capture/compare-match | Yes | |
| | TGI1B | TGR1B input capture/compare-match | No | |
| | TCI1V | TCNT1 overflow | No | |
| | TCI1U | TCNT1 underflow | No | |
| 2 | TGI2A | TGR2A input capture/compare-match | Yes | |
| | TGI2B | TGR2B input capture/compare-match | No | |
| | TCI2V | TCNT2 overflow | No | |
| | TCI2U | TCNT2 underflow | No | |

Note: Indicates the initial status following reset. The ranking of channels can be altered using the interrupt controller.

10.5.2 DMAC Activation

The TGRA register input capture/compare-match interrupt of any channel can be used as a source to activate the on-chip DMAC. For details, refer to section 11, DMAC.

The MTU has three TGRA register input capture/compare-match interrupts, one for any channel, that can be used as DMAC activation sources.

10.5.3 A/D Converter Activation

The TGRA register input capture/compare-match of any channel can be used to activate the on-chip A/D converter.

If the TTGE bit of the TIER is already set to 1 when the TGFA flag in the TSR is set to 1 by a TGRA register input capture/compare-match of any of the channels, an A/D conversion start request is sent to the A/D converter. If the MTU conversion start trigger is selected at such a time on the A/D converter side when this happens, the A/D conversion starts.

The MTU has three TGRA register input capture/compare-match interrupts, one for each channel, that can be used as A/D converter activation sources.

10.6 Operation Timing

10.6.1 Input/Output Timing

TCNT Count Timing: Count timing for the TCNT counter with internal clock operation is shown in figure 10.34. Count timing with external clock operation (normal mode) is shown in figure 10.35, and figure 10.36 shows count timing with external clock operation (phase counting mode).

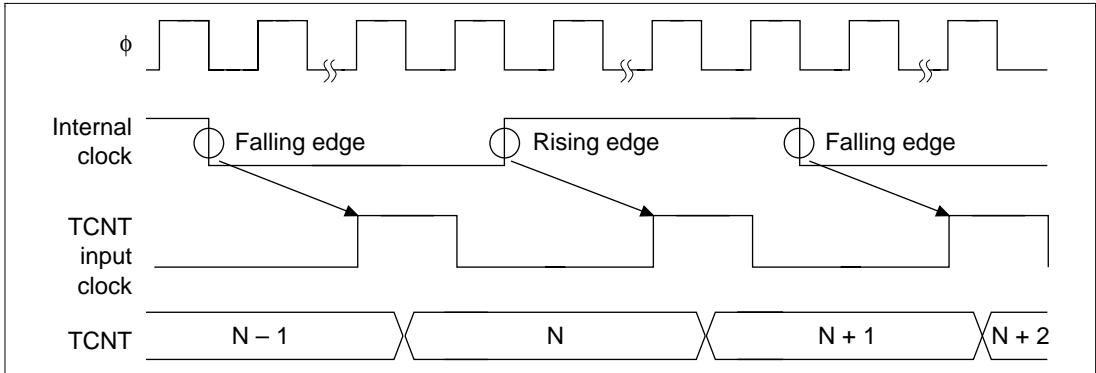


Figure 10.34 TCNT Count Timing during Internal Clock Operation

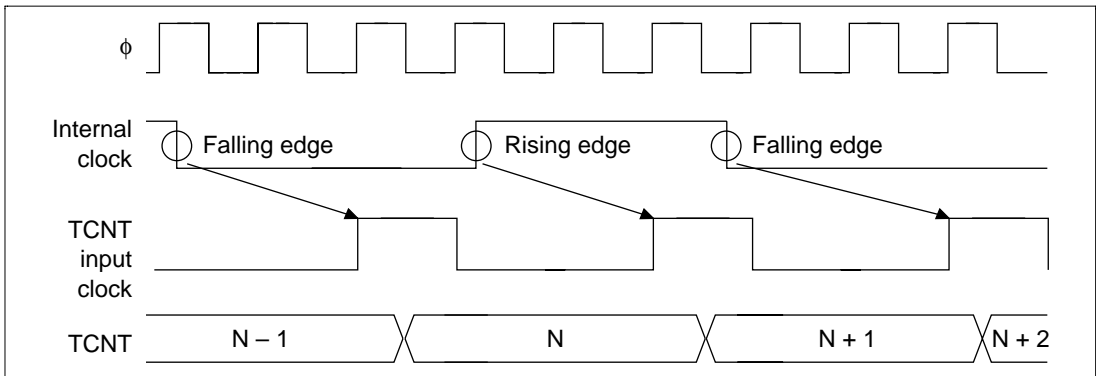


Figure 10.35 TCNT Count Timing during External Clock Operation (Normal Mode)

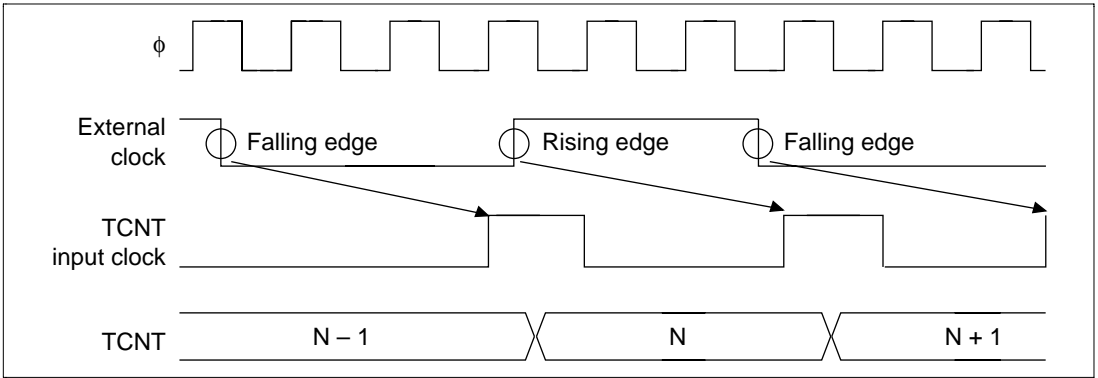


Figure 10.36 TCNT Count Timing during External Clock Operation (Phase Counting Mode)

Output Compare Output Timing: The compare-match signal is generated at the final state of TCNT and TGR matching. When a compare-match signal is issued, the output value set in TIOR or TOCR is output to the output compare output pin (TIOC pin). After TCNT and TGR matching, a compare-match signal is not issued until immediately before the TCNT input clock.

Output compare output timing (normal mode and PWM mode) is shown in figure 10.37.

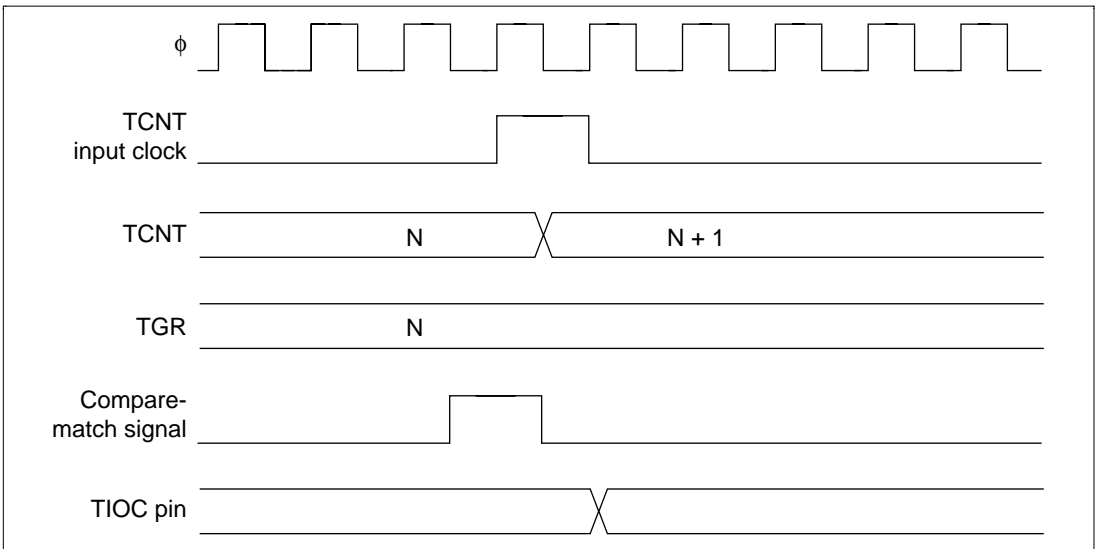


Figure 10.37 Output Compare Output Timing (Normal Mode/PWM Mode)

Input Capture Signal Timing: Figure 10.38 illustrates input capture timing.

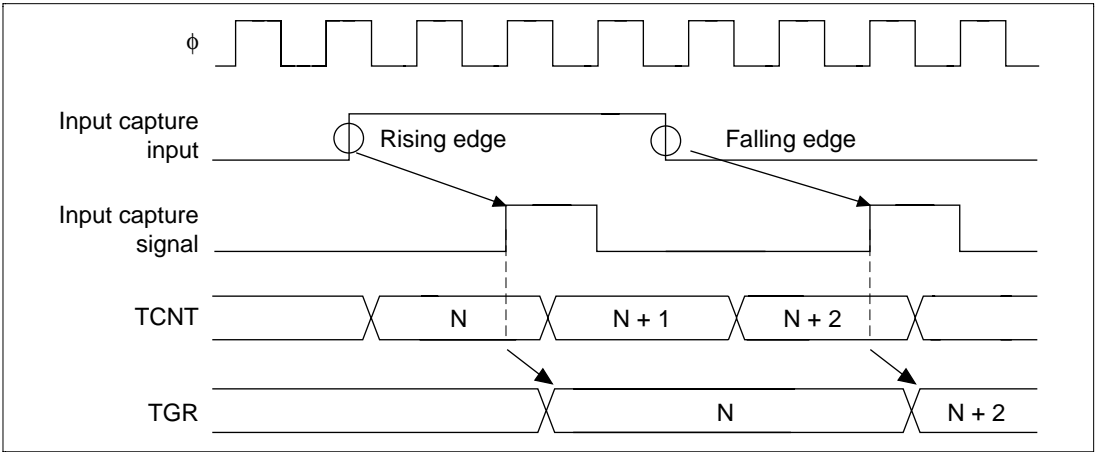


Figure 10.38 Input Capture Input Signal Timing

Counter Clearing Timing Due to Compare-Match/Input Capture: Timing for counter clearing due to compare-match is shown in figure 10.39. Figure 10.40 shows the timing for counter clearing due to input capture.

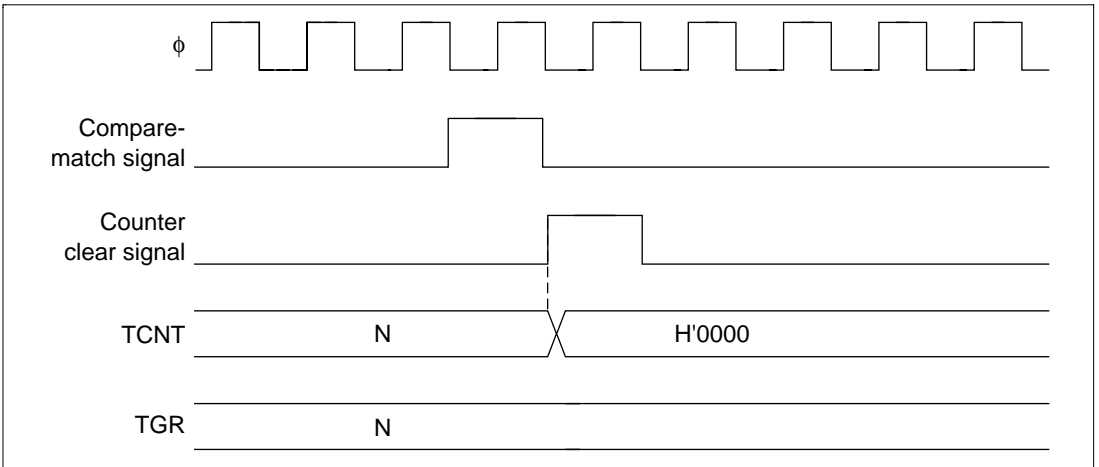


Figure 10.39 Counter Clearing Timing (Compare-Match)

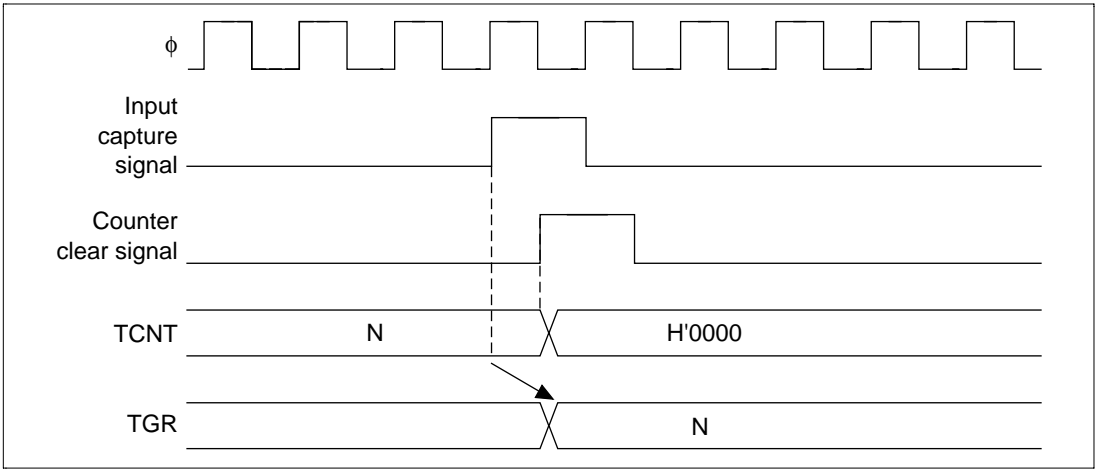


Figure 10.40 Counter Clearing Timing (Input Capture)

Buffer Operation Timing: Compare-match buffer operation timing is shown in figure 10.41. Figure 10.42 shows input capture buffer operation timing.

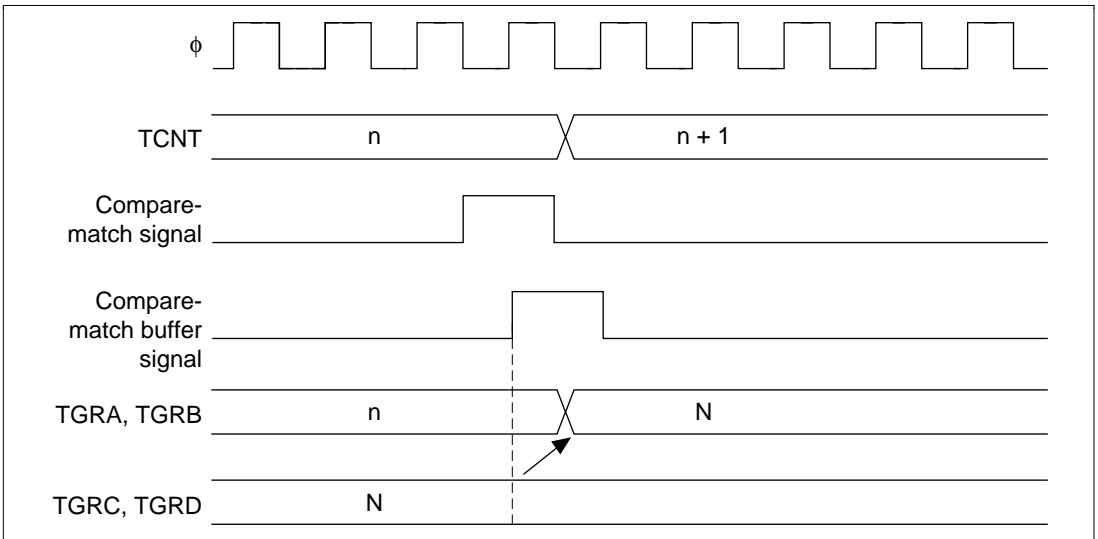


Figure 10.41 Buffer Operation Timing (Compare-Match)

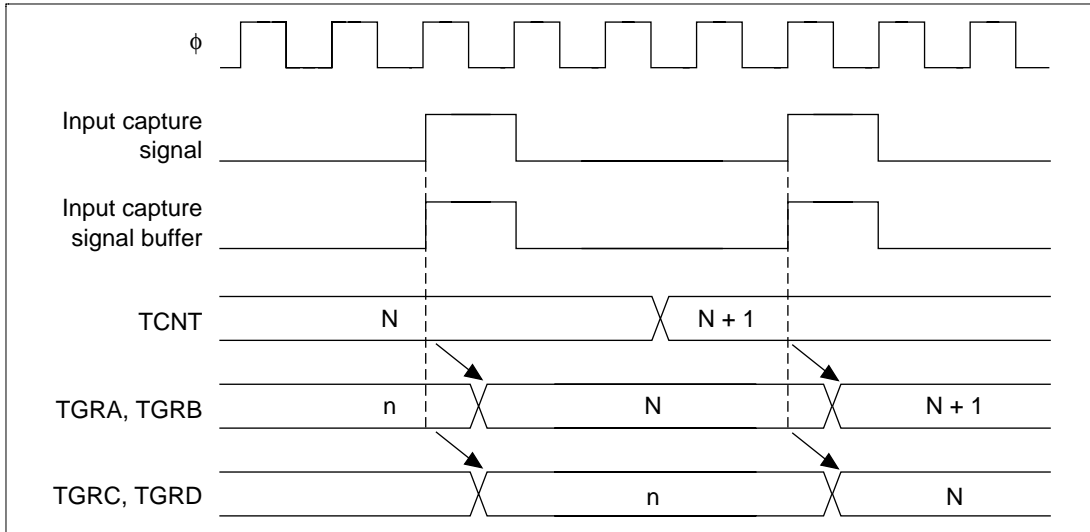


Figure 10.42 Buffer Operation Timing (Input Capture)

10.6.2 Interrupt Signal Timing

Setting TGF Flag Timing during Compare-Match: Figure 10.43 shows timing for the TGF flag of the timer status register (TSR) due to compare-match, as well as TGI interrupt request signal timing.

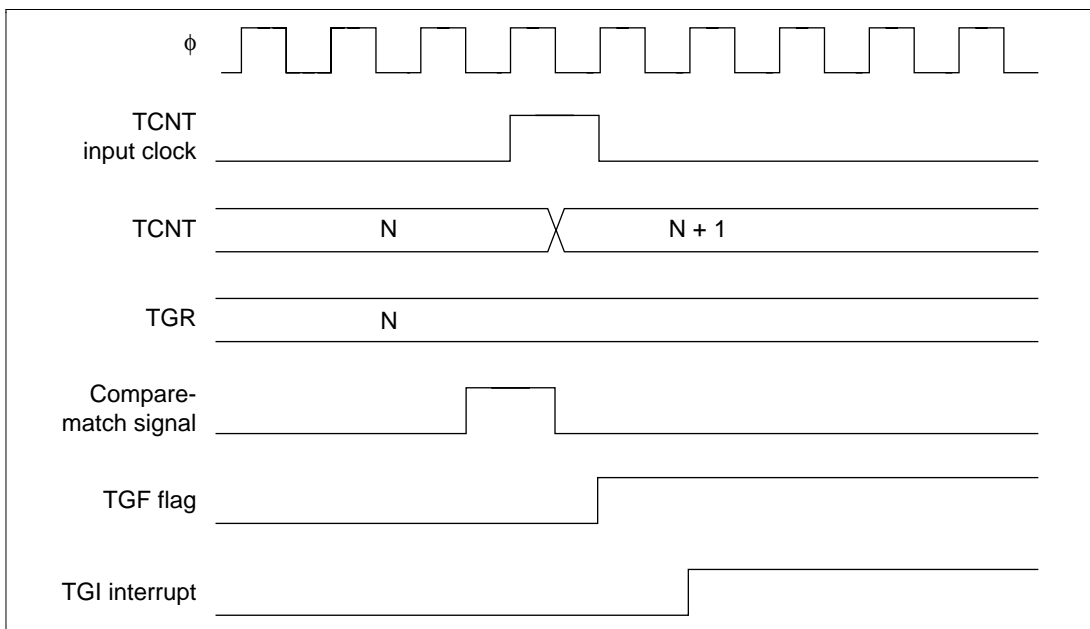


Figure 10.43 TGI Interrupt Timing (Compare Match)

Setting TGF Flag Timing during Input Capture: Figure 10.44 shows timing for the TGF flag of the timer status register (TSR) due to input capture, as well as TGI interrupt request signal timing.

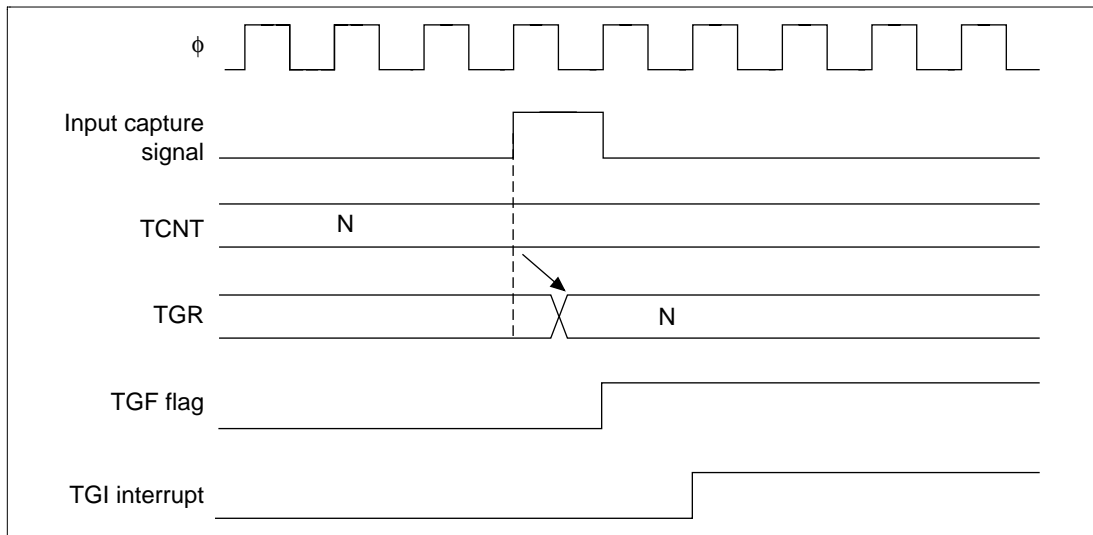


Figure 10.44 TGI Interrupt Timing (Input Capture)

Setting Timing for Overflow Flag (TCFV)/Underflow Flag (TCFU): Figure 10.45 shows timing for the TCFV flag of the timer status register (TSR) due to overflow, as well as TCIV interrupt request signal timing. Figure 10.46 shows timing for the TCFU flag of the timer status register (TSR) due to underflow, as well as TCIU interrupt request signal timing.

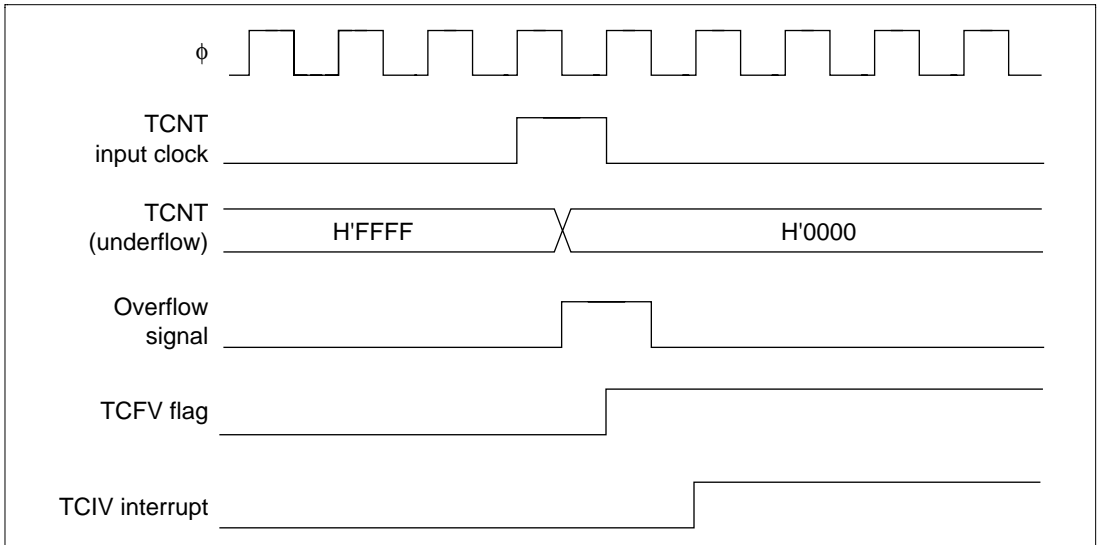


Figure 10.45 TCIV Interrupt Setting Timing

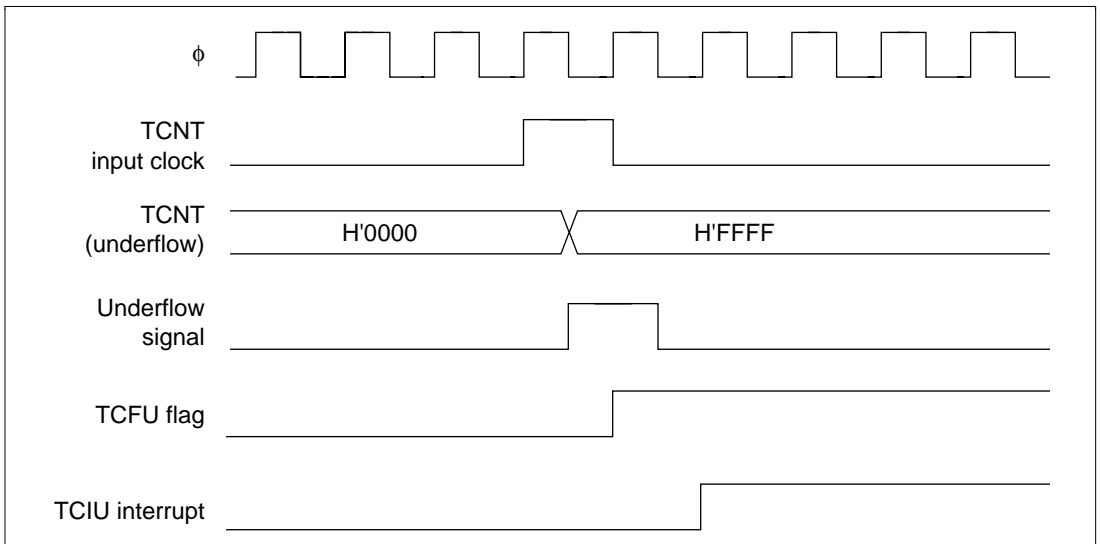


Figure 10.46 TCIU Interrupt Setting Timing

Status Flag Clearing Timing: The status flag is cleared when the CPU reads a 1 status followed by a 0 write. For DMA controller activation, clearing can also be done automatically. Figure 10.47 shows the timing for status flag clearing by the CPU. Figure 10.48 shows timing for clearing due to the DMA controller.

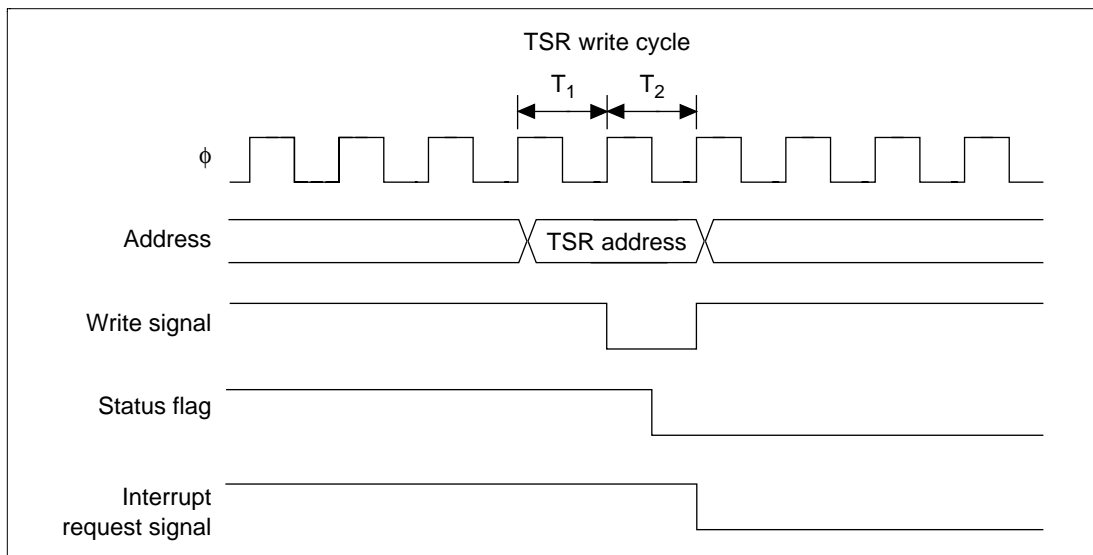


Figure 10.47 Timing of Status Flag Clearing by the CPU

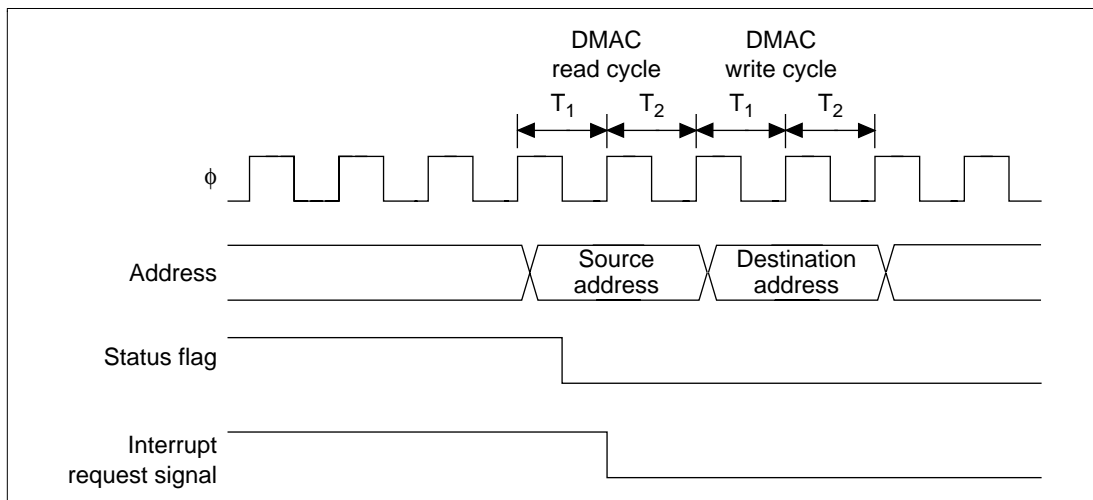


Figure 10.48 Timing of Status Flag Clearing by DMAC Activation

10.7 Notes and Precautions

This section describes contention and other matters requiring special attention during MTU operations.

10.7.1 Input Clock Limitations

The input clock pulse width, in the case of single edge, must be 1.5 states or greater, and 2.5 states or greater for both edges. Normal operation cannot be guaranteed with lesser pulse widths.

In phase counting mode, the phase difference between the two input clocks and the overlap must be 1.5 states or greater for each, and the pulse width must be 2.5 states or greater. Input clock conditions for phase counting mode are shown in figure 10.49.

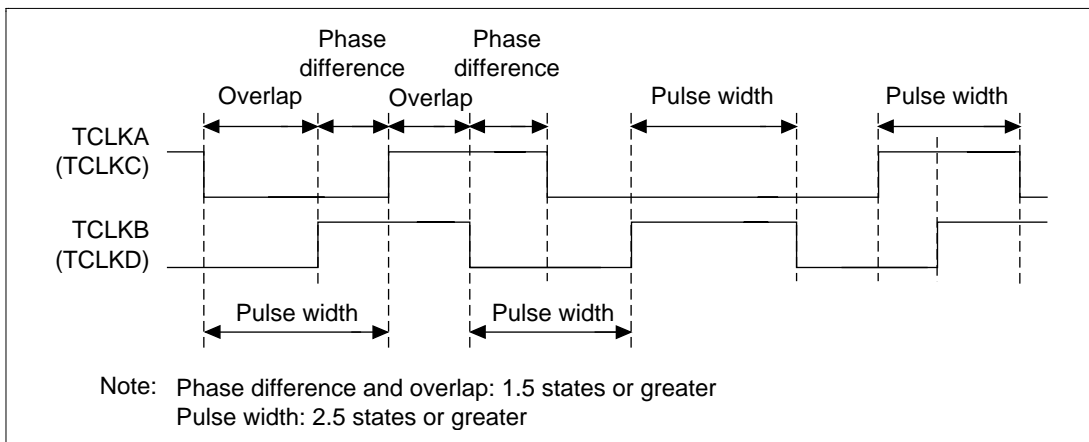


Figure 10.49 Phase Difference, Overlap, and Pulse Width in Phase Count Mode

10.7.2 Note on Cycle Setting

When setting a counter clearing by compare-match, clearing is done in the final state when TCNT matches the TGR value (update timing for count value on TCNT match). The actual number of states set in the counter is given by the following equation:

$$f = \frac{\phi}{(N + 1)}$$

(f: counter frequency, ϕ : operating frequency, N: value set in the TGR)

10.7.3 Contention between TCNT Write and Clear

If a counter clear signal is issued in the T_2 state during the TCNT write cycle, TCNT clearing has priority, and TCNT write is not conducted (figure 10.50).

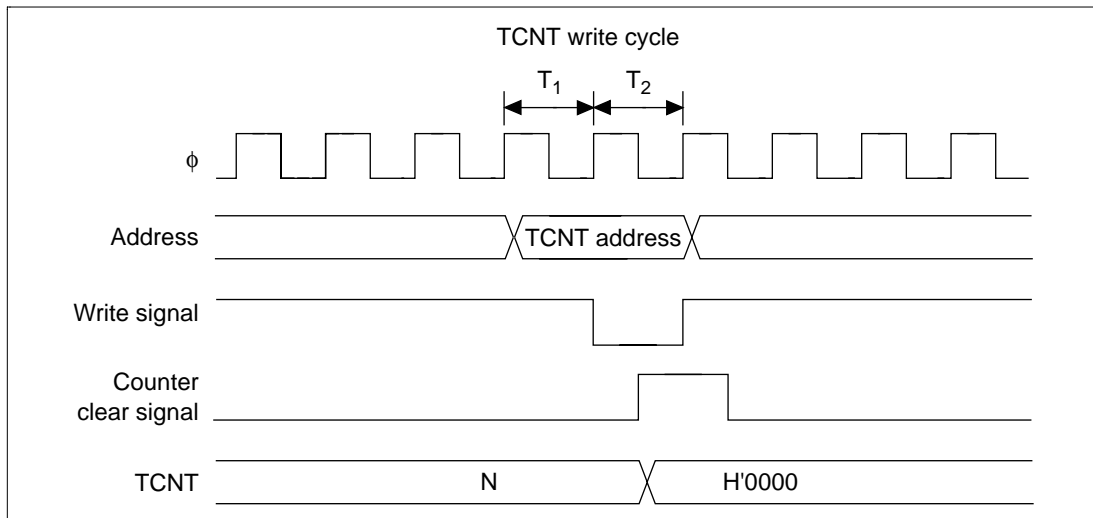


Figure 10.50 TCNT Write and Clear Contention

10.7.4 Contention between TCNT Write and Increment

If a count-up signal is issued in the T_2 state during the TCNT write cycle, TCNT write has priority, and the counter is not incremented (figure 10.51).

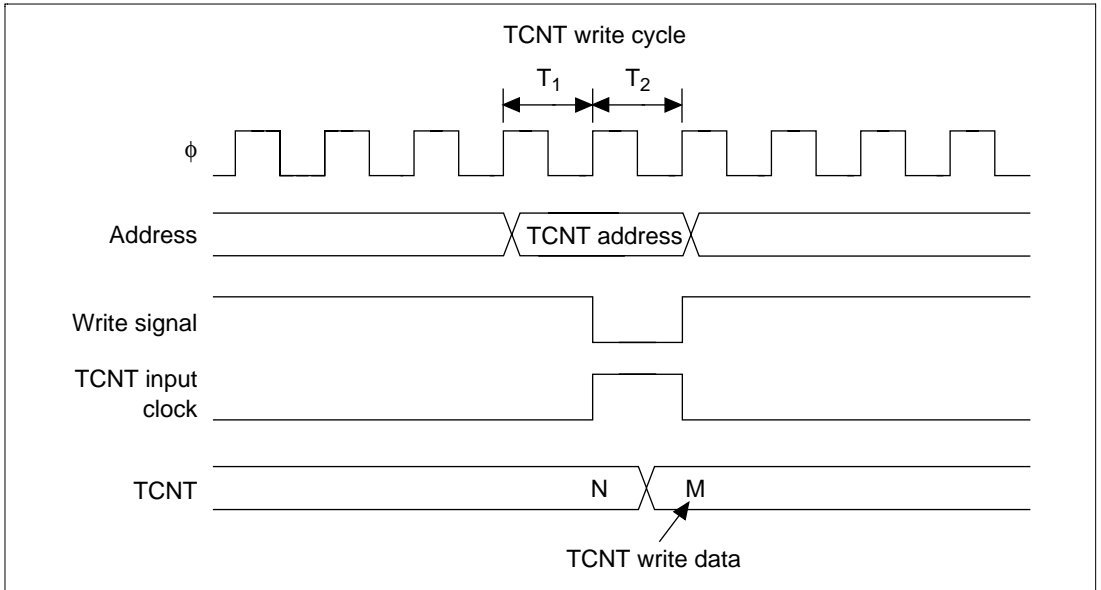


Figure 10.51 TCNT Write and Increment Contention

10.7.5 Contention between Buffer Register Write and Compare Match

If a compare-match occurs in the T_2 state of the TGR write cycle, data is transferred by the buffer operation from the buffer register to the TGR. Data to be transferred on channel 0 is that after write (figure 10.52).

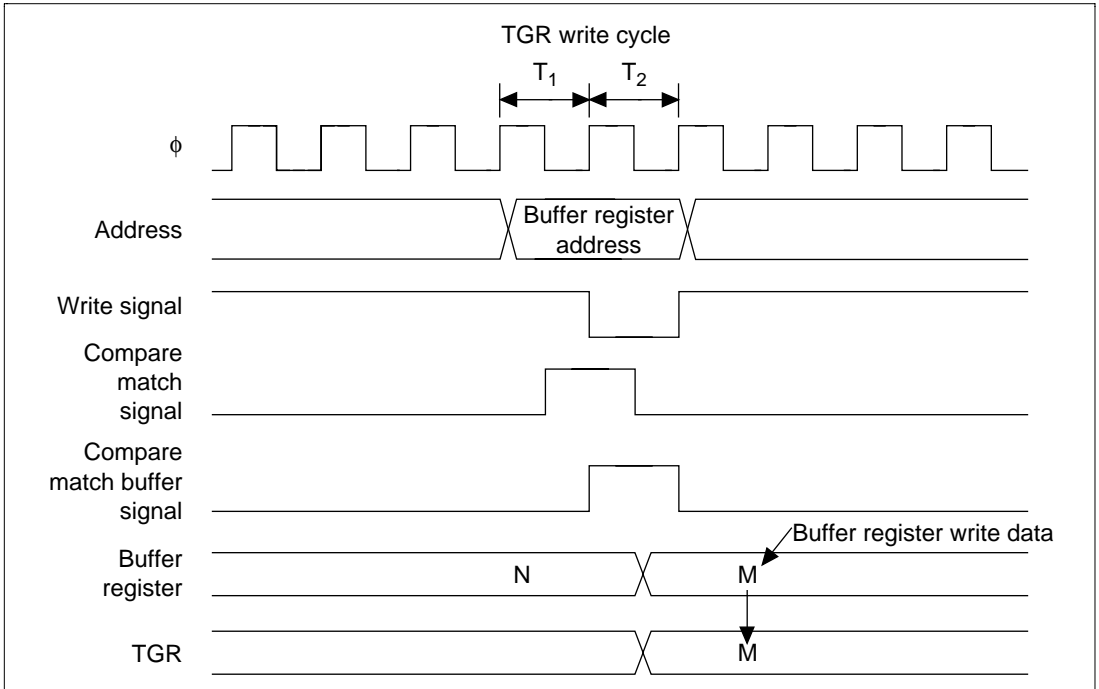


Figure 10.52 TGR Write and Compare-Match Contention (Channel 0)

10.7.6 Contention between TGR Read and Input Capture

If an input capture signal is issued in the T_1 state of the TGR read cycle, the read data is that after input capture transfer (figure 10.53).

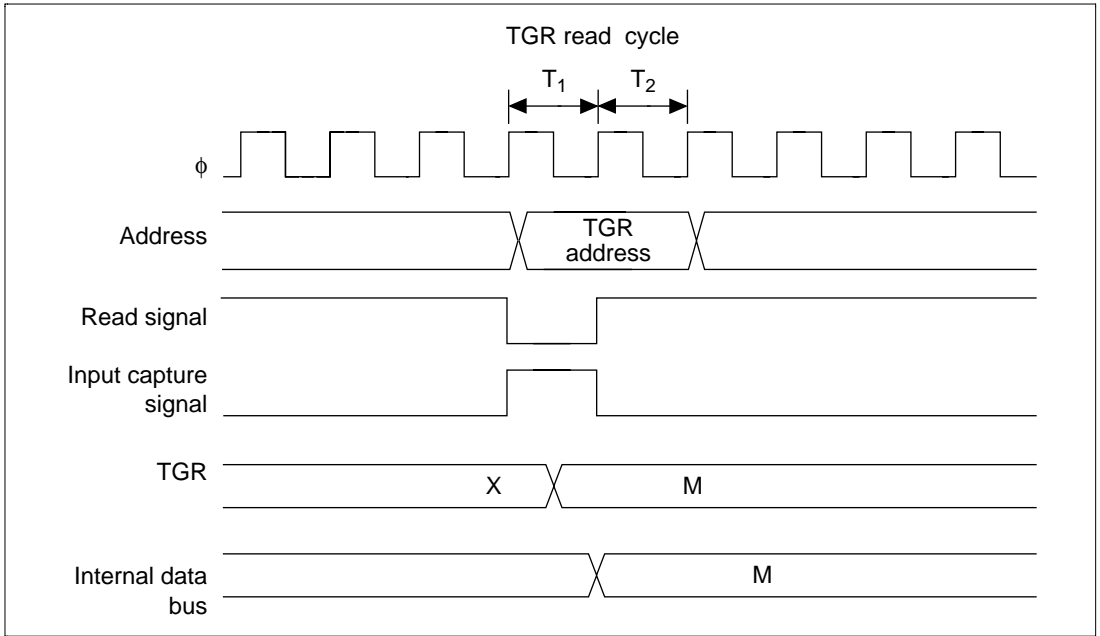


Figure 10.53 TGR Read and Input Capture Contention

10.7.7 Contention between TGR Write and Input Capture

If an input capture signal is issued in the T_2 state of the TGR read cycle, input capture has priority, and TGR write does not occur (figure 10.54).

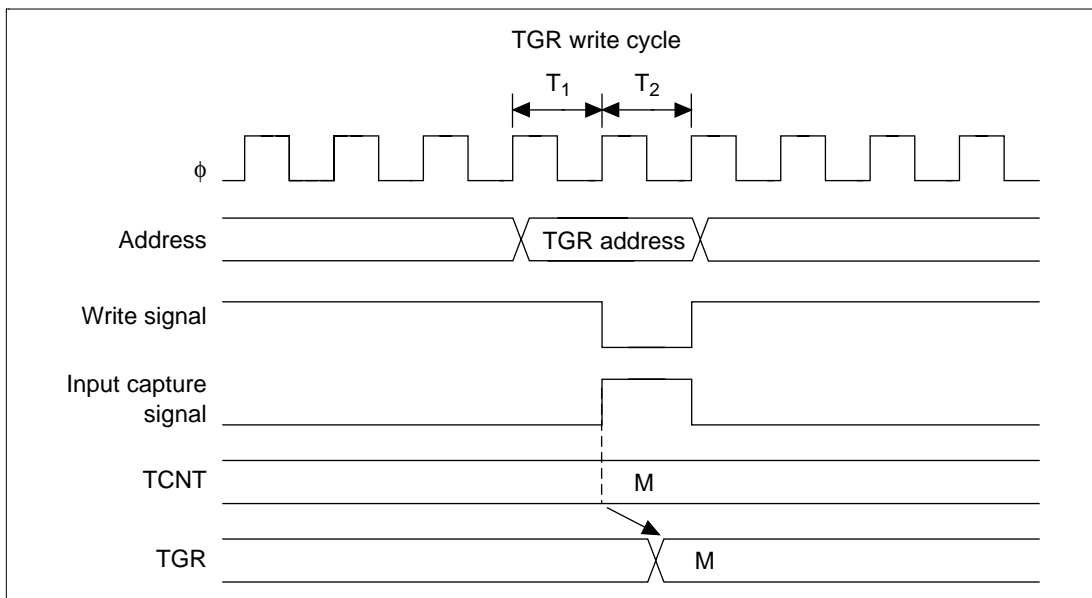


Figure 10.54 TGR Write and Input Capture Contention

10.7.8 Contention between Buffer Register Write and Input Capture

If an input capture signal is issued in the T_2 state of the buffer write cycle, write to the buffer register does not occur, and buffer operation takes priority (figure 10.55).

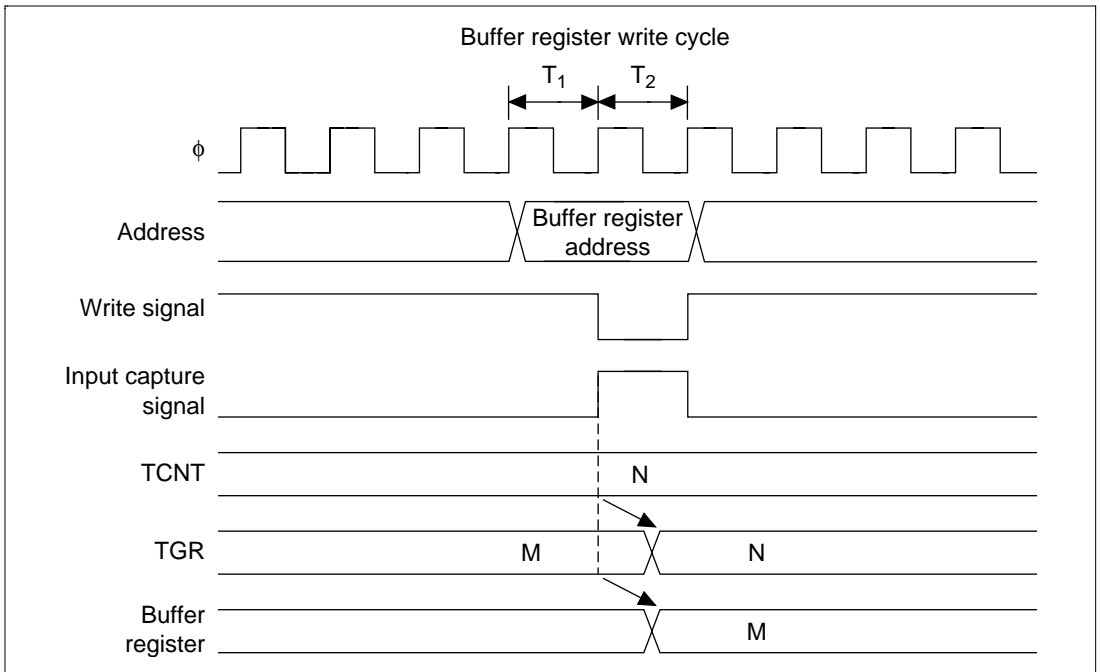


Figure 10.55 Buffer Register Write and Input Capture Contention

10.7.9 Contention between TGR Write and Compare Match

If a compare-match occurs in the T_2 state of the TGR write cycle, data is written to the TGR and a compare-match signal is issued (figure 10.56).

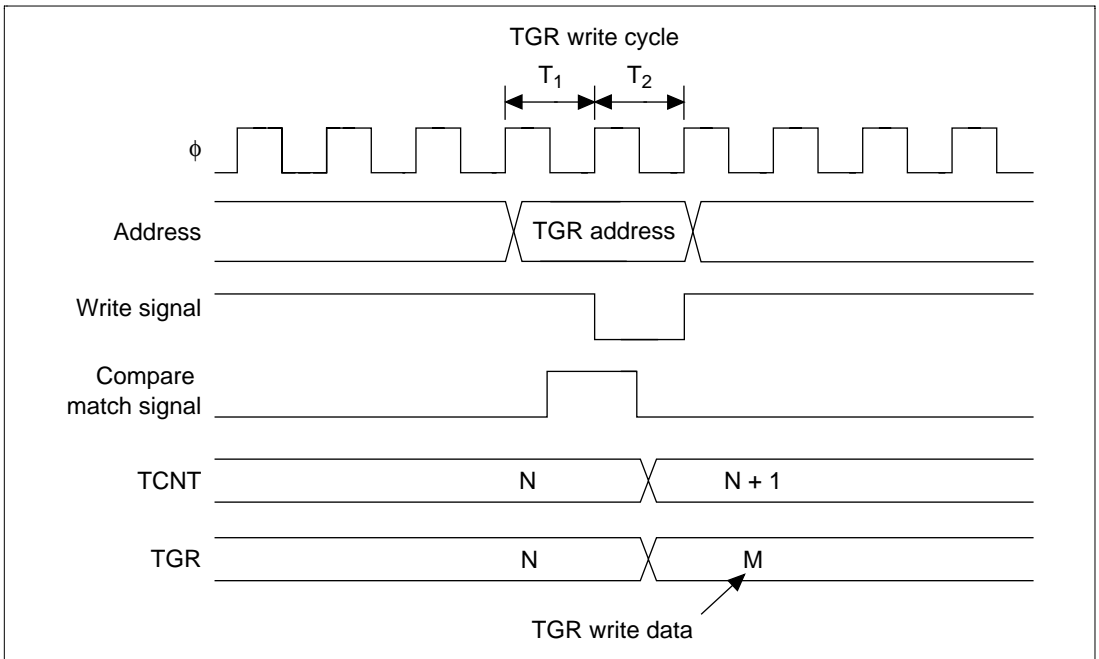


Figure 10.56 TGR Write and Compare Match Contention

10.7.10 TCNT2 Write and Overflow/Underflow Contention in Cascade Connection

With timer counters TCNT1 and TCNT2 in a cascade connection, when a contention occurs during TCNT1 count (during a TCNT2 overflow/underflow) in the T_2 state of the TCNT2 write cycle, the write to TCNT2 is conducted, and the TCNT1 count signal is prohibited. At this point, if there is match with TGR1A or TGR1B and the TCNT1 value, a compare signal is issued. The timing is shown in figure 10.57.

For cascade connections, be sure to synchronize settings for channels 1 and 2 when setting TCNT clearing.

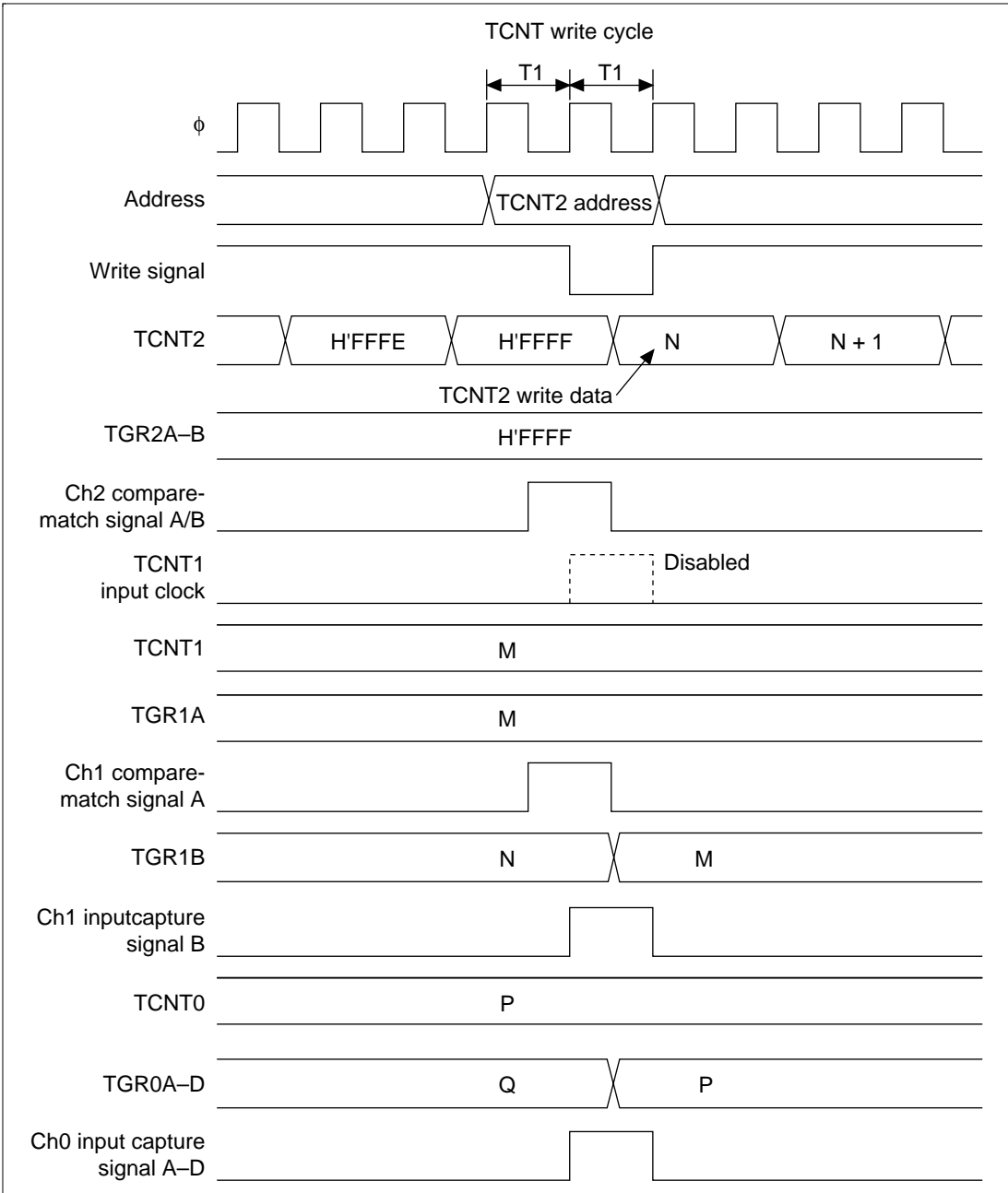


Figure 10.57 TCNT2 Write and Overflow/Underflow Contention with Cascade Connection

10.7.11 Contention between Overflow/Underflow and Counter Clearing

If overflow/underflow and counter clearing occur simultaneously, the TCFV/TCFU flag in TSR is not set and TCNT clearing takes precedence.

Figure 10.58 shows the operation timing when a TGR compare-match is specified as the clearing source, and H'FFFF is set in TGR.

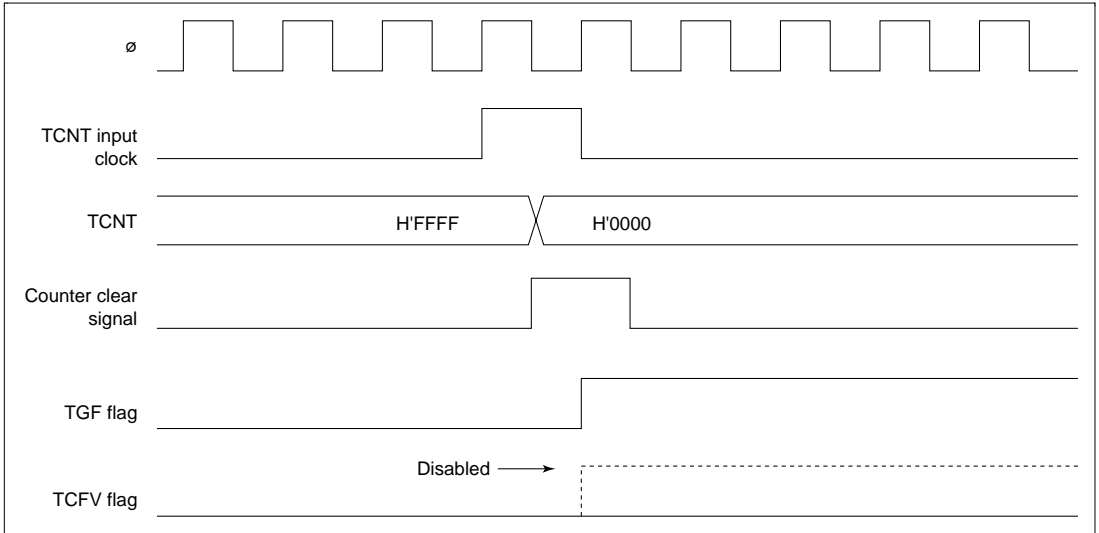


Figure 10.58 Contention between Overflow and Counter Clearing

10.7.12 Contention between TCNT Write and Overflow/Underflow

If there is an up-count or down-count in the T2 state of a TCNT write cycle, and overflow/underflow occurs, the TCNT write takes precedence and the TCFV/TCFU flag in TSR is not set.

Figure 10.59 shows the operation timing in this case.

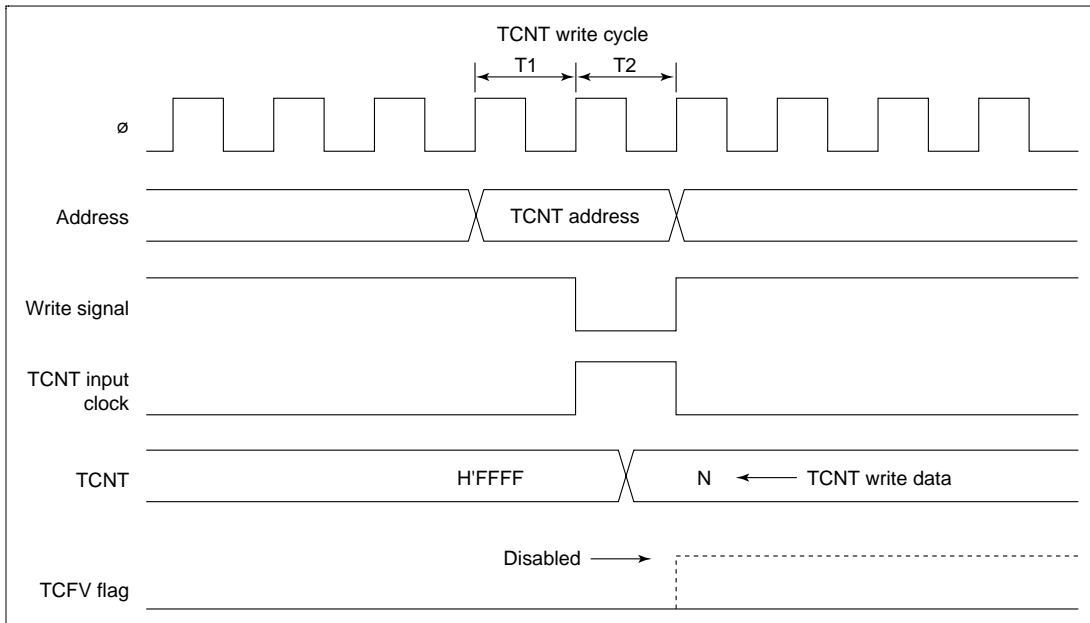


Figure 10.59 Contention between TCNT Write and Overflow

10.7.13 Cautions on Carrying Out Buffer Operation of Channel 0 in PWM Mode

In PWM mode 1, the TGRA and TGRB registers are used in pairs and PWM waveform is output to the TIOCA pin. In the same manner, the TGRC and TGRD registers are used in pairs and PWM waveform is output to the TIOCC pin. If either the TGRC or TGRD register is operating as a buffer register, the TIOCC pin cannot execute default output setting or PWM waveform output with the I/O control register (TIOR).

Note that for channel 0, the TIOCC pin allows both default output setting by TIOR and PWM output when setting buffer operation only for the TGRD register in PWM mode.

When using channel 0 in PWM mode 1 and setting buffer operation, use both the TGRC and TGRD registers as buffer registers.

10.8 MTU Output Pin Initialization

10.8.1 Operating Modes

The MTU has the following four operating modes. Waveform output is possible in all of these modes.

- Normal mode (channels 0 to 2)
- PWM mode 1 (channels 0 to 2)
- PWM mode 2 (channels 0 to 2)
- Reset-synchronous PWM mode 1 to 4 (channels 1 and 2)

The MTU output pin initialization method for each of these modes is described in this section.

10.8.2 Reset Start Operation

The MTU output pins (TIOC*) are initialized low by a reset and in standby mode. Since MTU pin function selection is performed by the pin function controller (PFC), when the PFC is set, the MTU pin states at that point are output to the ports. When MTU output is selected by the PFC immediately after a reset, the MTU output initial level, low, is output directly at the port. When the active level is low, the system will operate at this point, and therefore the PFC setting should be made after initialization of the MTU output pins is completed.

Note: Channel number and port notation are substituted for *.

10.8.3 Operation in Case of Re-Setting Due to Error during Operation, Etc.

If an error occurs during MTU operation, MTU output should be cut by the system. Cutoff is performed by switching the pin output to port output with the PFC and outputting the inverse of the active level. The pin initialization procedures for re-setting due to an error during operation, etc., and the procedures for restarting in a different mode after re-setting, are shown below.

The MTU has four operating modes, as stated above. There are thus 16 mode transition combinations, but some transitions are not available with certain channel and mode combinations. Possible mode transition combinations are shown in table 10.14.

Table 10.14 Mode Transition Combinations

| Before | After | | | |
|--------|--------|------|------|------|
| | Normal | PWM1 | PWM2 | PCM |
| Normal | (1) | (2) | (3) | (4) |
| PWM1 | (5) | (6) | (7) | (8) |
| PWM2 | (9) | (10) | (11) | (12) |
| PCM | (13) | (14) | (15) | (16) |

Legend:

Normal: Normal mode

PWM1: PWM mode 1

PWM2: PWM mode 2

PCM: Phase counting modes 1 to 4

The above abbreviations are used in some places in following descriptions.

10.8.4 Overview of Initialization Procedures and Mode Transitions in Case of Error during Operation, etc.

- When making a transition to a mode (Normal, PWM1, PWM2, PCM) in which the pin output level is selected by the timer I/O control register (TIOR) setting, initialize the pins by means of a TIOR setting.
- In PWM mode 1, since a waveform is not output to the TIOC*B (TIOC*D) pin, setting TIOR will not initialize the pins. If initialization is required, carry it out in normal mode, then switch to PWM mode 1.
- In PWM mode 2, since a waveform is not output to the cycle register pin, setting TIOR will not initialize the pins. If initialization is required, carry it out in normal mode, then switch to PWM mode 2.
- In normal mode or PWM mode 2, if TGRC and TGRD operate as buffer registers, setting TIOR will not initialize the buffer register pins. If initialization is required, clear buffer mode, carry out initialization, then set buffer mode again.
- In PWM mode 1, if either TGRC or TGRD operates as a buffer register, setting TIOR will not initialize the TGRC pin. To initialize the TGRC pin, clear buffer mode, carry out initialization, then set buffer mode again.

Note: Channel number is substituted for * indicated in this article.

Pin initialization procedures are described below for the numbered combinations in table 10.14. The active level is assumed to be low.

(1) Operation when Error Occurs during Normal Mode Operation, and Operation is Restarted in Normal Mode: Figure 10.60 shows an explanatory diagram of the case where an error occurs in normal mode and operation is restarted in normal mode after re-setting.

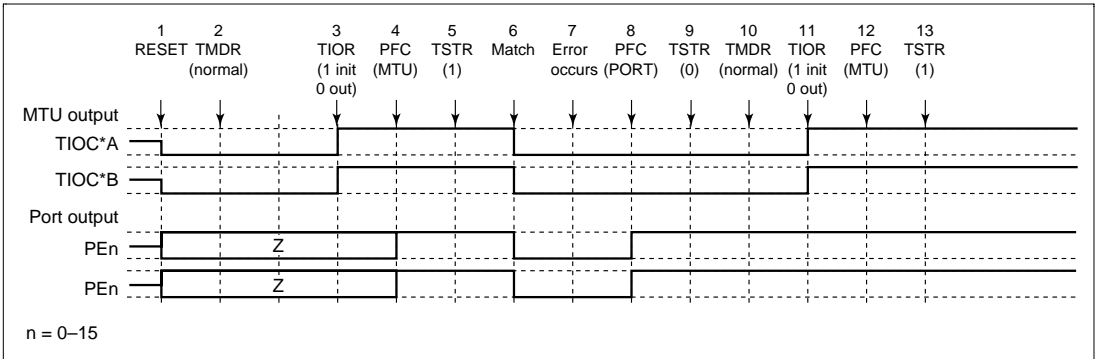


Figure 10.60 Error Occurrence in Normal Mode, Recovery in Normal Mode

1. After a reset, MTU output is low and ports are in the high-impedance state.
2. After a reset, the TMDR setting is for normal mode.
3. Initialize the pins with TIOR. (The example shows initial high output, with low output on compare-match occurrence.)
4. Set MTU output with the PFC.
5. The count operation is started by TSTR.
6. Output goes low on compare-match occurrence.
7. An error occurs.
8. Set port output with the PFC and output the inverse of the active level.
9. The count operation is stopped by TSTR.
10. Not necessary when restarting in normal mode.
11. Initialize the pins with TIOR.
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(2) Operation when Error Occurs during Normal Mode Operation, and Operation is Restarted in PWM Mode 1: Figure 10.61 shows an explanatory diagram of the case where an error occurs in normal mode and operation is restarted in PWM mode 1 after re-setting.

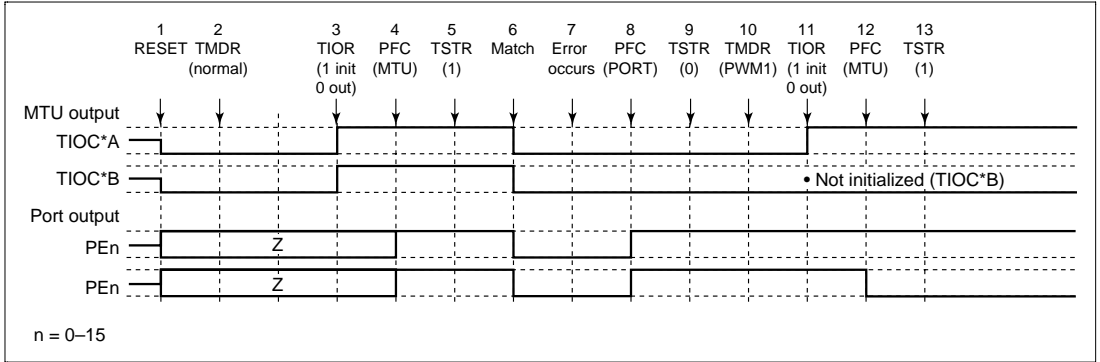


Figure 10.61 Error Occurrence in Normal Mode, Recovery in PWM Mode 1

1 to 9 are the same as in figure 10.60.

10. Set PWM mode 1.

11. Initialize the pins with TIOR. (In PWM mode 1, the TIOC*B side is not initialized. If initialization is required, initialize in normal mode, then switch to PWM mode 1.)

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

(3) Operation when Error Occurs during Normal Mode Operation, and Operation is Restarted in PWM Mode 2: Figure 10.62 shows an explanatory diagram of the case where an error occurs in normal mode and operation is restarted in PWM mode 2 after re-setting.

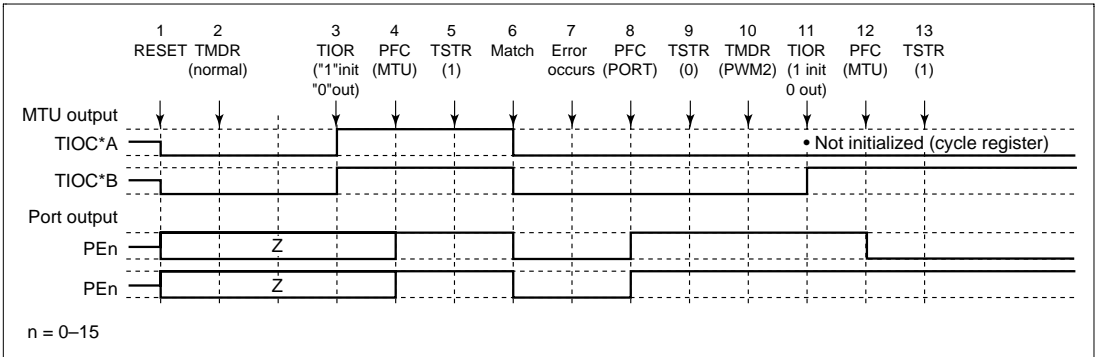


Figure 10.62 Error Occurrence in Normal Mode, Recovery in PWM Mode 2

1 to 9 are the same as in figure 10.60.

10. Set PWM mode 2.

11. Initialize the pins with TIOR. (In PWM mode 2, the cycle register pins are not initialized. If initialization is required, initialize in normal mode, then switch to PWM mode 2.)

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

Note: PWM mode 2 can only be set for channels 0 to 2, and therefore TOER setting is not necessary.

(4) Operation when Error Occurs during Normal Mode Operation, and Operation is Restarted in Phase Counting Mode: Figure 10.63 shows an explanatory diagram of the case where an error occurs in normal mode and operation is restarted in phase counting mode after re-setting.

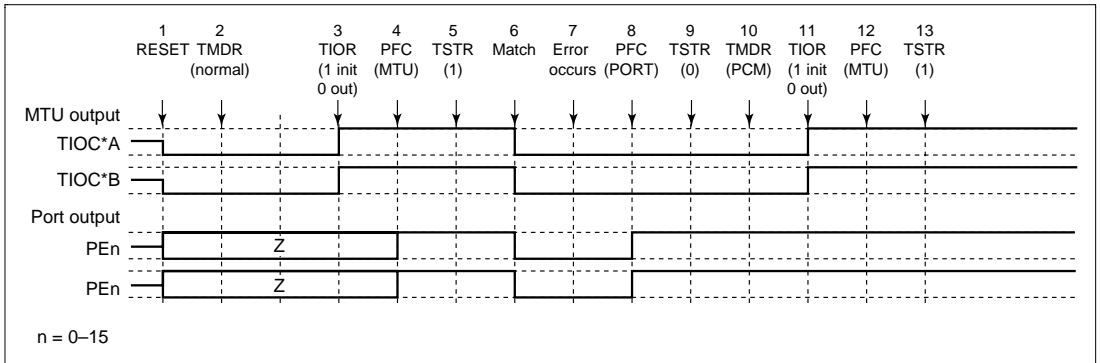


Figure 10.63 Error Occurrence in Normal Mode, Recovery in Phase Counting Mode

1 to 9 are the same as in figure 10.60.

10. Set phase counting mode.

11. Initialize the pins with TIOR.

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

Note: Phase counting mode can only be set for channels 1 and 2, and therefore TOER setting is not necessary.

(5) Operation when Error Occurs during PWM Mode 1 Operation, and Operation is Restarted in Normal Mode: Figure 10.64 shows an explanatory diagram of the case where an error occurs in PWM mode 1 and operation is restarted in normal mode after re-setting.

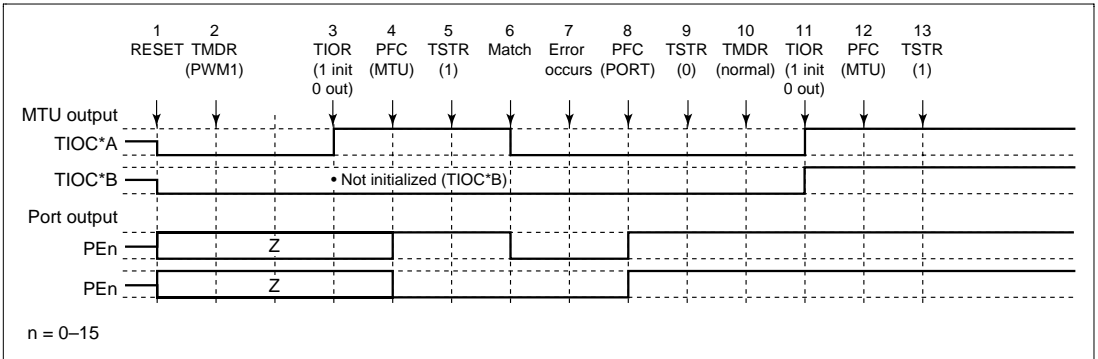


Figure 10.64 Error Occurrence in PWM Mode 1, Recovery in Normal Mode

1. After a reset, MTU output is low and ports are in the high-impedance state.
2. Set PWM mode 1.
3. Initialize the pins with TIOR. (The example shows initial high output, with low output on compare-match occurrence. In PWM mode 1, the TIOC*B side is not initialized.)
4. Set MTU output with the PFC.
5. The count operation is started by TSTR.
6. Output goes low on compare-match occurrence.
7. An error occurs.
8. Set port output with the PFC and output the inverse of the active level.
9. The count operation is stopped by TSTR.
10. Set normal mode.
11. Initialize the pins with TIOR.
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(6) Operation when Error Occurs during PWM Mode 1 Operation, and Operation is Restarted in PWM Mode 1: Figure 10.65 shows an explanatory diagram of the case where an error occurs in PWM mode 1 and operation is restarted in PWM mode 1 after re-setting.

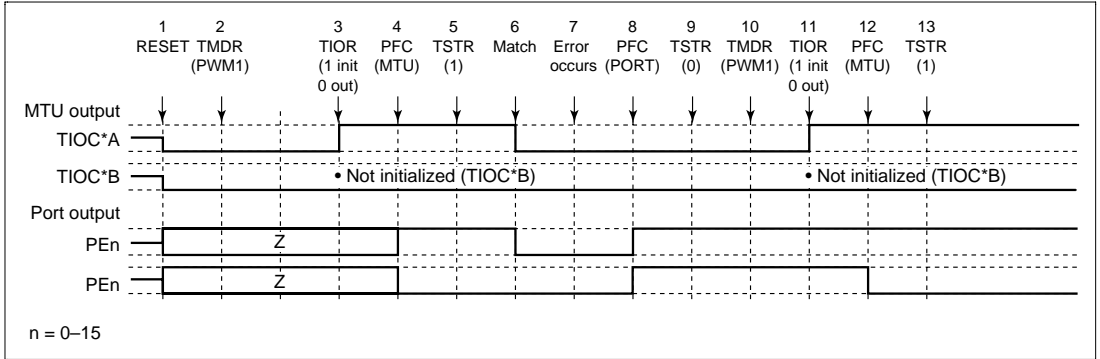


Figure 10.65 Error Occurrence in PWM Mode 1, Recovery in PWM Mode 1

1 to 9 are the same as in figure 10.64.

10. Not necessary when restarting in PWM mode 1.

11. Initialize the pins with TIOR. (In PWM mode 1, the TIOC*B side is not initialized.)

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

(7) Operation when Error Occurs during PWM Mode 1 Operation, and Operation is Restarted in PWM Mode 2: Figure 10.66 shows an explanatory diagram of the case where an error occurs in PWM mode 1 and operation is restarted in PWM mode 2 after re-setting.

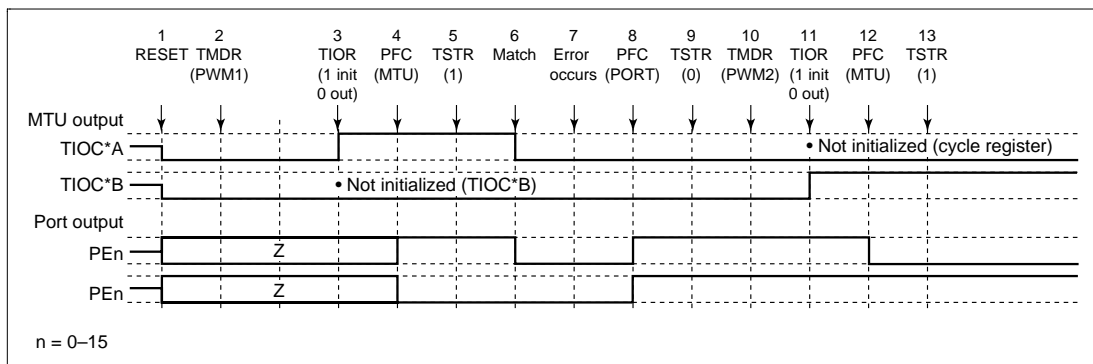


Figure 10.66 Error Occurrence in PWM Mode 1, Recovery in PWM Mode 2

1 to 9 are the same as in figure 10.64.

10. Set PWM mode 2.

11. Initialize the pins with TIOR. (In PWM mode 2, the cycle register pins are not initialized.)

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

Note: PWM mode 2 can only be set for channels 0 to 2, and therefore TOER setting is not necessary.

(8) Operation when Error Occurs during PWM Mode 1 Operation, and Operation is Restarted in Phase Counting Mode: Figure 10.67 shows an explanatory diagram of the case where an error occurs in PWM mode 1 and operation is restarted in phase counting mode after re-setting.

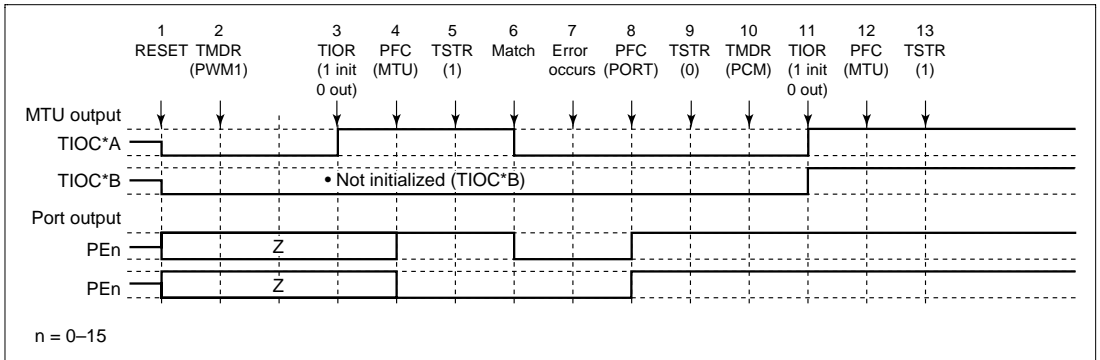


Figure 10.67 Error Occurrence in PWM Mode 1, Recovery in Phase Counting Mode

1 to 9 are the same as in figure 10.64.

10. Set phase counting mode.

11. Initialize the pins with TIOR.

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

Note: Phase counting mode can only be set for channels 1 and 2, and therefore TOER setting is not necessary.

(9) Operation when Error Occurs during PWM Mode 2 Operation, and Operation is Restarted in Normal Mode: Figure 10.68 shows an explanatory diagram of the case where an error occurs in PWM mode 2 and operation is restarted in normal mode after re-setting.

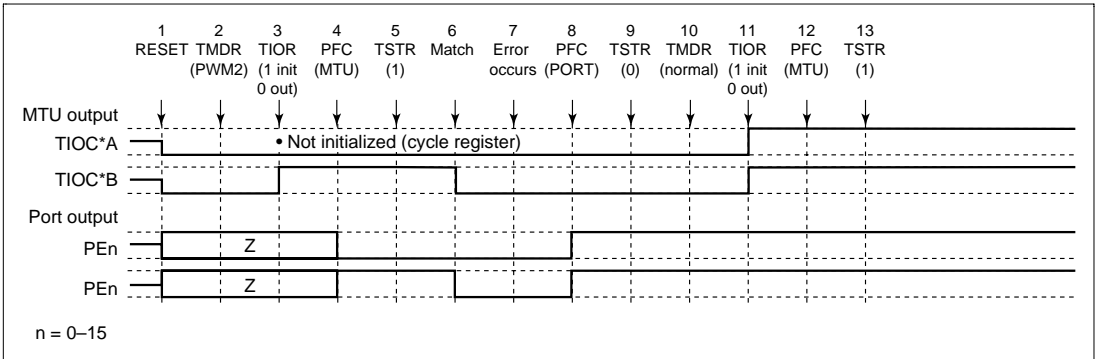


Figure 10.68 Error Occurrence in PWM Mode 2, Recovery in Normal Mode

1. After a reset, MTU output is low and ports are in the high-impedance state.
2. Set PWM mode 2.
3. Initialize the pins with TIOR. (The example shows initial high output, with low output on compare-match occurrence. In PWM mode 2, the cycle register pins are not initialized. In the example, TIOC*A is the cycle register.)
4. Set MTU output with the PFC.
5. The count operation is started by TSTR.
6. Output goes low on compare-match occurrence.
7. An error occurs.
8. Set port output with the PFC and output the inverse of the active level.
9. The count operation is stopped by TSTR.
10. Set normal mode.
11. Initialize the pins with TIOR.
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(10) Operation when Error Occurs during PWM Mode 2 Operation, and Operation is Restarted in PWM Mode 1: Figure 10.69 shows an explanatory diagram of the case where an error occurs in PWM mode 2 and operation is restarted in PWM mode 1 after re-setting.

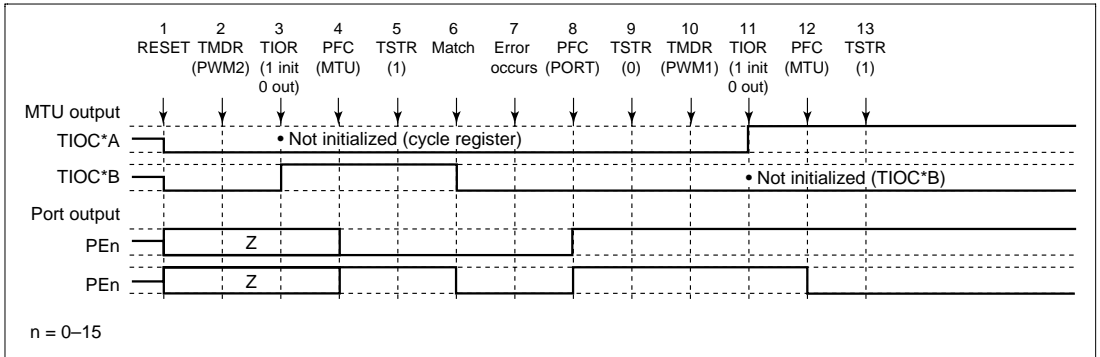


Figure 10.69 Error Occurrence in PWM Mode 2, Recovery in PWM Mode 1

1 to 9 are the same as in figure 10.68.

10. Set PWM mode 1.

11. Initialize the pins with TIOR. (In PWM mode 1, the TIOC*B side is not initialized.)

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

(11) Operation when Error Occurs during PWM Mode 2 Operation, and Operation is Restarted in PWM Mode 2: Figure 10.70 shows an explanatory diagram of the case where an error occurs in PWM mode 2 and operation is restarted in PWM mode 2 after re-setting.

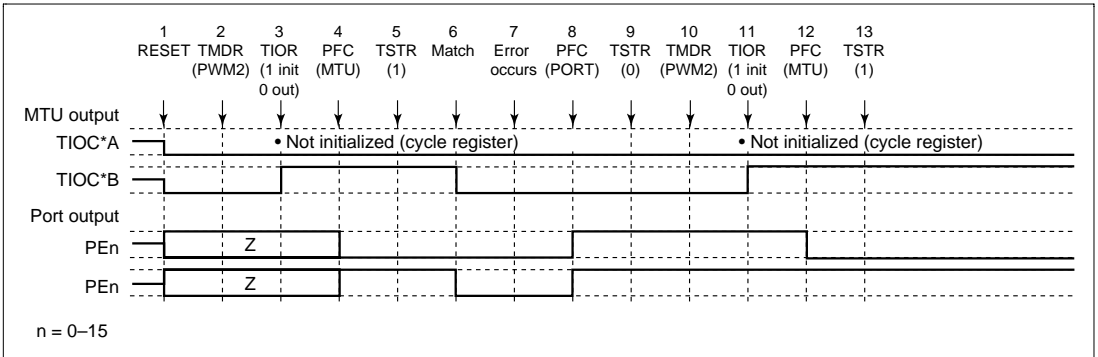


Figure 10.70 Error Occurrence in PWM Mode 2, Recovery in PWM Mode 2

1 to 9 are the same as in figure 10.68.

10. Not necessary when restarting in PWM mode 2.

11. Initialize the pins with TIOR. (In PWM mode 2, the cycle register pins are not initialized.)

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

(12) Operation when Error Occurs during PWM Mode 2 Operation, and Operation is Restarted in Phase Counting Mode: Figure 10.71 shows an explanatory diagram of the case where an error occurs in PWM mode 2 and operation is restarted in phase counting mode after re-setting.

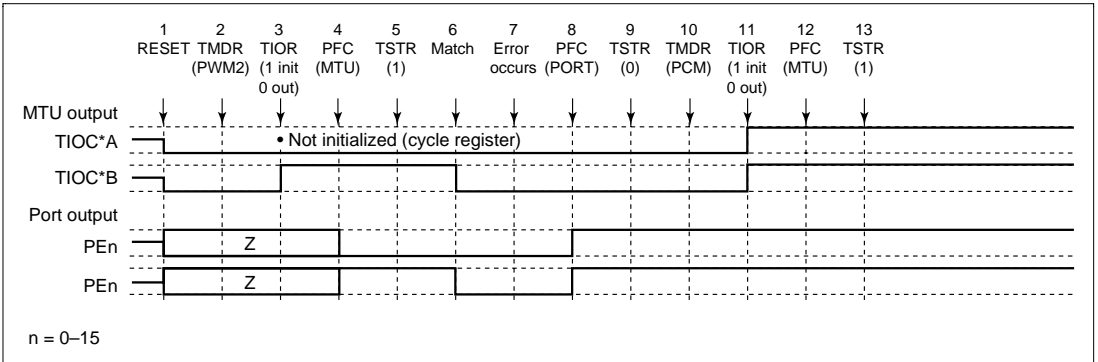


Figure 10.71 Error Occurrence in PWM Mode 2, Recovery in Phase Counting Mode

- 1 to 9 are the same as in figure 10.68.
10. Set phase counting mode.
 11. Initialize the pins with TIOR.
 12. Set MTU output with the PFC.
 13. Operation is restarted by TSTR.

(13) Operation when Error Occurs during Phase Counting Mode Operation, and Operation is Restarted in Normal Mode: Figure 10.72 shows an explanatory diagram of the case where an error occurs in phase counting mode and operation is restarted in normal mode after re-setting.

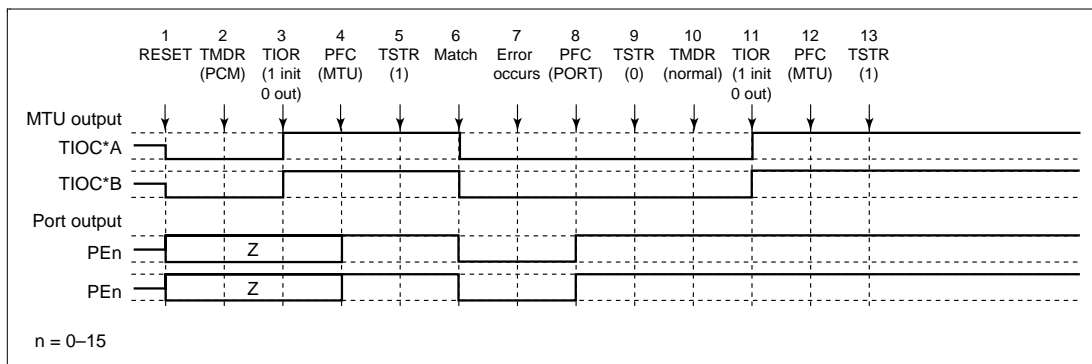


Figure 10.72 Error Occurrence in Phase Counting Mode, Recovery in Normal Mode

1. After a reset, MTU output is low and ports are in the high-impedance state.
2. Set phase counting mode.
3. Initialize the pins with TIOR. (The example shows initial high output, with low output on compare-match occurrence.)
4. Set MTU output with the PFC.
5. The count operation is started by TSTR.
6. Output goes low on compare-match occurrence.
7. An error occurs.
8. Set port output with the PFC and output the inverse of the active level.
9. The count operation is stopped by TSTR.
10. Set in normal mode.
11. Initialize the pins with TIOR.
12. Set MTU output with the PFC.
13. Operation is restarted by TSTR.

(14) Operation when Error Occurs during Phase Counting Mode Operation, and Operation is Restarted in PWM Mode 1: Figure 10.73 shows an explanatory diagram of the case where an error occurs in phase counting mode and operation is restarted in PWM mode 1 after re-setting.

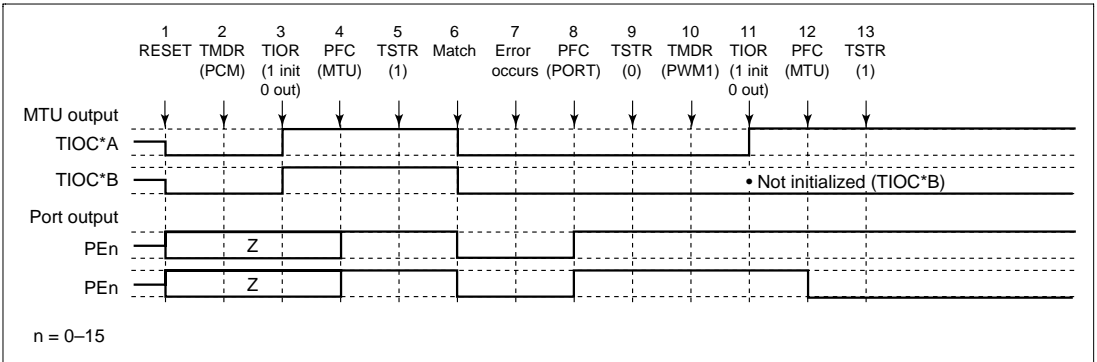


Figure 10.73 Error Occurrence in Phase Counting Mode, Recovery in PWM Mode 1

1 to 9 are the same as in figure 10.72.

10. Set PWM mode 1.

11. Initialize the pins with TIOR. (In PWM mode 1, the TIOC*B side is not initialized.)

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

(15) Operation when Error Occurs during Phase Counting Mode Operation, and Operation is Restarted in PWM Mode 2: Figure 10.74 shows an explanatory diagram of the case where an error occurs in phase counting mode and operation is restarted in PWM mode 2 after re-setting.

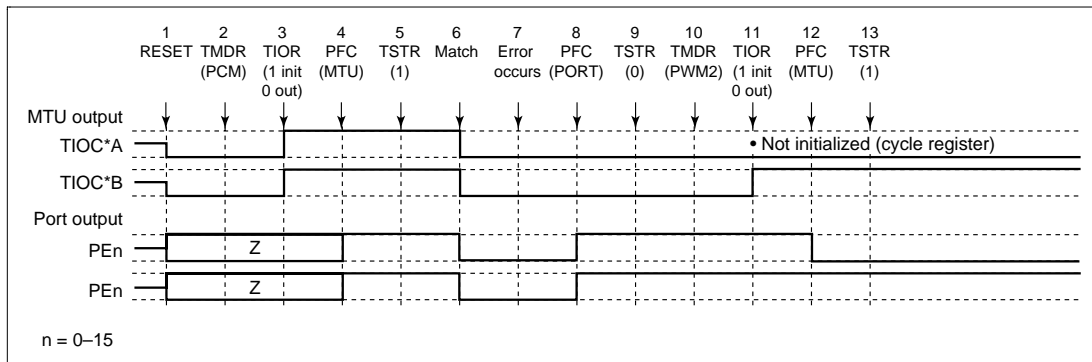


Figure 10.74 Error Occurrence in Phase Counting Mode, Recovery in PWM Mode 2

1 to 9 are the same as in figure 10.72.

10. Set PWM mode 2.

11. Initialize the pins with TIOR. (In PWM mode 2, the cycle register pins are not initialized.)

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

(16) Operation when Error Occurs during Phase Counting Mode Operation, and Operation is Restarted in Phase Counting Mode: Figure 10.75 shows an explanatory diagram of the case where an error occurs in phase counting mode and operation is restarted in phase counting mode after re-setting.

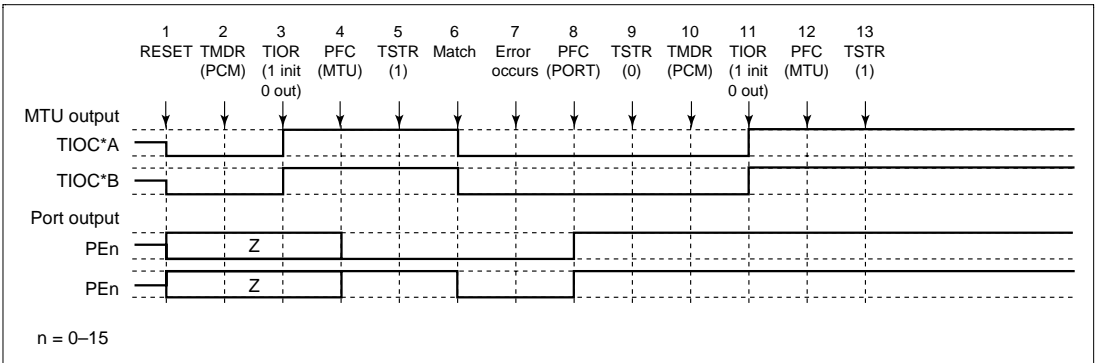


Figure 10.75 Error Occurrence in Phase Counting Mode, Recovery in Phase Counting Mode

1 to 9 are the same as in figure 10.72.

10. Not necessary when restarting in phase counting mode.

11. Initialize the pins with TIOR.

12. Set MTU output with the PFC.

13. Operation is restarted by TSTR.

Section 11 Watchdog Timer (WDT)

11.1 Overview

The watchdog timer (WDT) is a 1-channel timer for monitoring system operations. If a system encounters a problem (crashes, for example) and the timer counter overflows without being rewritten correctly by the CPU, an overflow signal ($\overline{\text{WDTOVF}}$) is output externally. The WDT can simultaneously generate an internal reset signal for the entire chip.

When the watchdog function is not needed, the WDT can be used as an interval timer. In the interval timer operation, an interval timer interrupt is generated at each counter overflow. The WDT is also used in recovering from the standby mode.

11.1.1 Features

- Works in watchdog timer mode or interval timer mode.
- Outputs $\overline{\text{WDTOVF}}$ in the watchdog timer mode. When the counter overflows in the watchdog timer mode, overflow signal $\overline{\text{WDTOVF}}$ is output externally. You can select whether to reset the chip internally when this happens.
- Generates interrupts in the interval timer mode. When the counter overflows, it generates an interval timer interrupt.
- Clears standby mode.
- Works with eight counter input clocks.

11.1.2 Block Diagram

Figure 11.1 is the block diagram of the WDT.

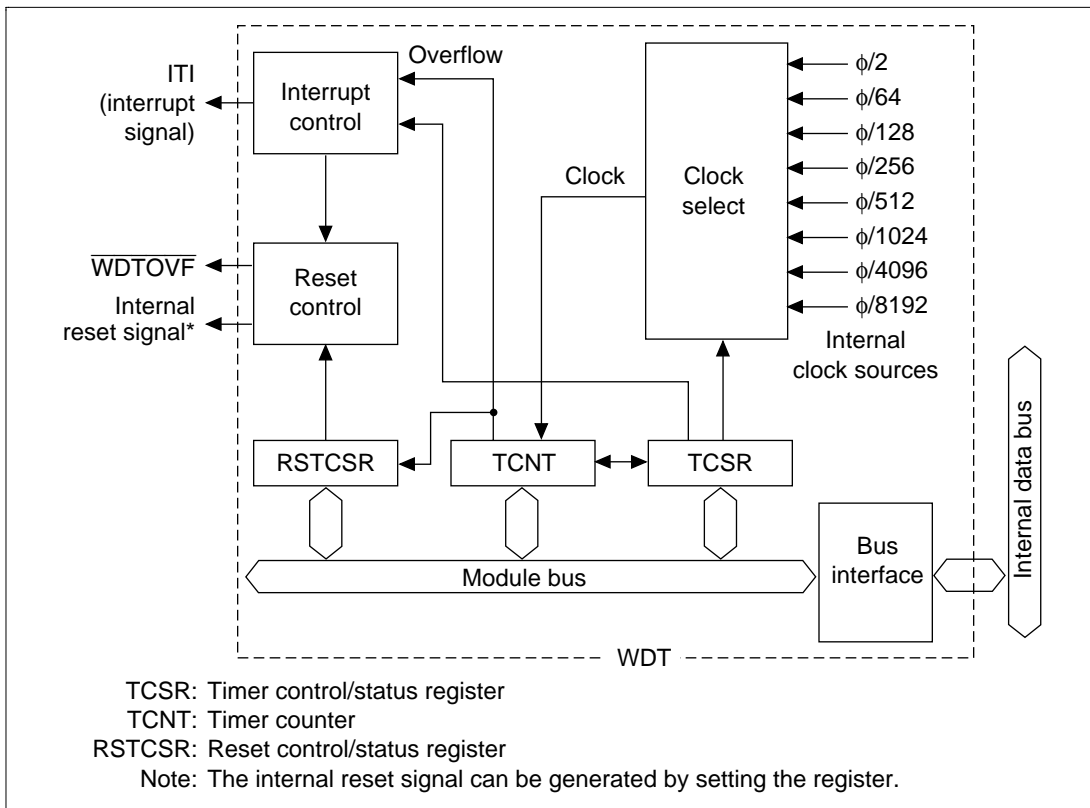


Figure 11.1 WDT Block Diagram

11.1.3 Pin Configuration

Table 11.1 shows the pin configuration.

Table 11.1 Pin Configuration

| Pin | Abbreviation | I/O | Function |
|-------------------------|----------------------------|-----|--|
| Watchdog timer overflow | $\overline{\text{WDTOVF}}$ | O | Outputs the counter overflow signal in the watchdog timer mode |

11.1.4 Register Configuration

Table 11.2 summarizes the three WDT registers. They are used to select the clock, switch the WDT mode, and control the reset signal.

Table 11.2 WDT Registers

| Name | Abbreviation | R/W | Initial Value | Address | |
|-------------------------------|--------------|---------------------|---------------|---------------------|--------------------|
| | | | | Write* ¹ | Read* ² |
| Timer control/status register | TCSR | R/(W)* ³ | H'18 | H'FFFF8610 | H'FFFF8610 |
| Timer counter | TCNT | R/W | H'00 | | H'FFFF8611 |
| Reset control/status register | RSTCSR | R/(W)* ³ | H'1F | H'FFFF8612 | H'FFFF8613 |

- Notes:
1. Write by word transfer. It cannot be written in byte or longword.
 2. Read by byte transfer. It cannot be read in word or longword.
 3. Only 0 can be written in bit 7 to clear the flag.

11.2 Register Descriptions

11.2.1 Timer Counter (TCNT)

The TCNT is an 8-bit read/write upcounter. (The TCNT differs from other registers in that it is more difficult to write to. See section 11.2.4, Register Access, for details.) When the timer enable bit (TME) in the timer control/status register (TCSR) is set to 1, the watchdog timer counter starts counting pulses of an internal clock selected by clock select bits 2 to 0 (CKS2 to CKS0) in the TCSR. When the value of the TCNT overflows (changes from H'FF to H'00), a watchdog timer overflow signal ($\overline{\text{WDTOVF}}$) or interval timer interrupt (ITI) is generated, depending on the mode selected in the $\overline{\text{WT/IT}}$ bit of the TCSR.

The TCNT is initialized to H'00 by a power-on reset and when the TME bit is cleared to 0. It is not initialized in the standby mode.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| 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 |

11.2.2 Timer Control/Status Register (TCSR)

The timer control/status register (TCSR) is an 8-bit read/write register. (The TCSR differs from other registers in that it is more difficult to write to. See section 11.2.4, Register Access, for details.) Its functions include selecting the timer mode and clock source.

Bits 7 to 5 are initialized to 000 by a power-on reset or in standby mode. Bits 2 to 0 are initialized to 000 by a power-on reset, but retain their values in the standby mode.

| | | | | | | | | |
|----------------|-------|----------------------------|-----|---|---|------|------|------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | OVF | WT/ $\overline{\text{IT}}$ | TME | — | — | CKS2 | CKS1 | CKS0 |
| Initial value: | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| R/W: | R/(W) | R/W | R/W | R | R | R/W | R/W | R/W |

- Bit 7—Overflow Flag (OVF): Indicates that the TCNT has overflowed from H'FF to H'00 in the interval timer mode. It is not set in the watchdog timer mode.

Bit 7: OVF

Description

| | |
|---|---|
| 0 | No overflow of TCNT in interval timer mode (initial value) Cleared by reading OVF, then writing 0 in OVF |
|---|---|

| | |
|---|--|
| 1 | TCNT overflow in the interval timer mode |
|---|--|

- Bit 6—Timer Mode Select ($\overline{\text{WT/IT}}$): Selects whether to use the WDT as a watchdog timer or interval timer. When the TCNT overflows, the WDT either generates an interval timer interrupt (ITI) or generates a $\overline{\text{WDTOVF}}$ signal, depending on the mode selected.

Bit 6: $\overline{\text{WT/IT}}$

Description

| | |
|---|--|
| 0 | Interval timer mode: interval timer interrupt request to the CPU when TCNT overflows (initial value) |
|---|--|

| | |
|---|--|
| 1 | Watchdog timer mode: $\overline{\text{WDTOVF}}$ signal output externally when TCNT overflows. (Section 11.2.3, Reset Control/Status Register (RSTCSR), describes in detail what happens when TCNT overflows in the watchdog timer mode.) |
|---|--|

- Bit 5—Timer Enable (TME): Enables or disables the timer.

| Bit 5: TME | Description |
|------------|---|
| 0 | Timer disabled: TCNT is initialized to H'00 and count-up stops (initial value) |
| 1 | Timer enabled: TCNT starts counting. A $\overline{\text{WDTOVF}}$ signal or interrupt is generated when TCNT overflows. |

- Bits 4 and 3—Reserved: These bits always read as 1. The write value should always be 1.
- Bits 2 to 0: Clock Select 2 to 0 (CKS2 to CKS0): These bits select one of eight internal clock sources for input to the TCNT. The clock signals are obtained by dividing the frequency of the system clock (ϕ).

| Bit 2: CKS2 | Bit 1: CKS1 | Bit 0: CKS0 | Clock Source | Description |
|-------------|-------------|-------------|--------------------------|---|
| | | | | Overflow Interval* ($\phi = 28.7 \text{ MHz}$) |
| 0 | 0 | 0 | $\phi/2$ (initial value) | 17.9 μs |
| 0 | 0 | 1 | $\phi/64$ | 573.4 μs |
| 0 | 1 | 0 | $\phi/128$ | 1.1 ms |
| 0 | 1 | 1 | $\phi/256$ | 2.3 ms |
| 1 | 0 | 0 | $\phi/512$ | 4.6 ms |
| 1 | 0 | 1 | $\phi/1024$ | 9.2 ms |
| 1 | 1 | 0 | $\phi/4096$ | 36.7 ms |
| 1 | 1 | 1 | $\phi/8192$ | 73.4 ms |

Note: The overflow interval listed is the time from when the TCNT begins counting at H'00 until an overflow occurs.

11.2.3 Reset Control/Status Register (RSTCSR)

The RSTCSR is an 8-bit readable and writable register. (The RSTCSR differs from other registers in that it is more difficult to write. See section 11.2.4, Register Access, for details.) It controls output of the internal reset signal generated by timer counter (TCNT) overflow. RSTCSR is initialized to H'1F by input of a reset signal from the $\overline{\text{RES}}$ pin, but is not initialized by the internal reset signal generated by the overflow of the WDT. It is initialized to H'1F in standby mode.

| | | | | | | | | |
|----------------|--------|------|---|---|---|---|---|---|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | WOVF | RSTE | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| R/W: | R/(W)* | R/W | R | R | R | R | R | R |

Note: Only 0 can be written in bit 7 to clear the flag.

- Bit 7—Watchdog Timer Overflow Flag (WOVF): Indicates that the TCNT has overflowed (H'FF to H'00) in the watchdog timer mode. It is not set in the interval timer mode.

| Bit 7: WOVF | Description |
|-------------|--|
| 0 | No TCNT overflow in watchdog timer mode (initial value) Cleared when software reads WOVF, then writes 0 in WOVF |
| 1 | Set by TCNT overflow in watchdog timer mode |

- Bit 6—Reset Enable (RSTE): Selects whether to reset the chip internally if the TCNT overflows in the watchdog timer mode.

| Bit 6: RSTE | Description |
|-------------|--|
| 0 | Not reset when TCNT overflows (initial value). LSI not reset internally, but TCNT and TCSR reset within WDT. |
| 1 | Reset when TCNT overflows |

- Bit 5—Reserved: This bit always read as 0. The write value should always be 0.
- Bits 4 to 0—Reserved: These bits always read as 1. The write value should always be 1.

11.2.4 Register Access

The watchdog timer's TCNT, TCSR, and RSTCSR registers differ from other registers in that they are more difficult to write to. The procedures for writing and reading these registers are given below.

Writing to the TCNT and TCSR: These registers must be written by a word transfer instruction. They cannot be written by byte transfer instructions.

The TCNT and TCSR both have the same write address. The write data must be contained in the lower byte of the written word. The upper byte must be H'5A (for the TCNT) or H'A5 (for the TCSR) (figure 11.2). This transfers the write data from the lower byte to the TCNT or TCSR.

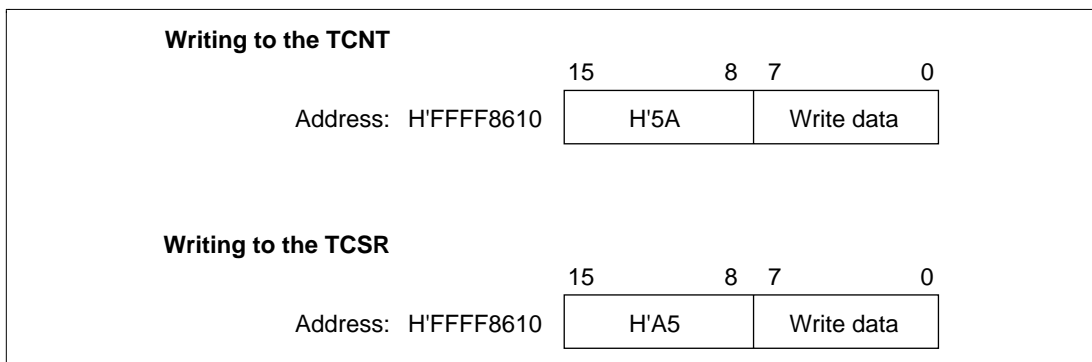


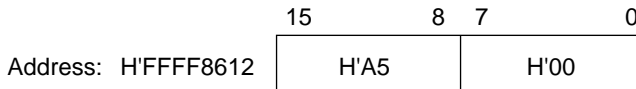
Figure 11.2 Writing to the TCNT and TCSR

Writing to the RSTCSR: The RSTCSR must be written by a word access to address H'FFFF8612. It cannot be written by byte transfer instructions.

Procedures for writing 0 in WOVF (bit 7) and for writing to RSTE (bit 6) are different, as shown in figure 11.3.

To write 0 in the WOVF bit, the write data must be H'A5 in the upper byte and H'00 in the lower byte. This clears the WOVF bit to 0. The RSTE bit is not affected. To write to the RSTE bit, the upper byte must be H'5A and the lower byte must be the write data. The values of bit 6 of the lower byte is transferred to the RSTE bit, respectively. The WOVF bit is not affected.

Writing 0 to the WOVF bit



Writing to the RSTE bit

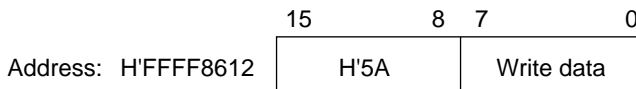


Figure 11.3 Writing to the RSTCSR

Reading from the TCNT, TCSR, and RSTCSR: TCNT, TCSR, and RSTCSR are read like other registers. Use byte transfer instructions. The read addresses are H'FFFF8610 for the TCSR, H'FFFF8611 for the TCNT, and H'FFFF8613 for the RSTCSR.

11.3 Operation

11.3.1 Watchdog Timer Mode

To use the WDT as a watchdog timer, set the $\overline{WT/IT}$ and TME bits of the TCSR to 1. Software must prevent TCNT overflow by rewriting the TCNT value (normally by writing H'00) before overflow occurs. No TCNT overflows will occur while the system is operating normally, but if the TCNT fails to be rewritten and overflows occur due to a system crash or the like, a \overline{WDTOVF} signal is output externally (figure 11.4). The \overline{WDTOVF} signal can be used to reset the system. The \overline{WDTOVF} signal is output for 128 ϕ clock cycles.

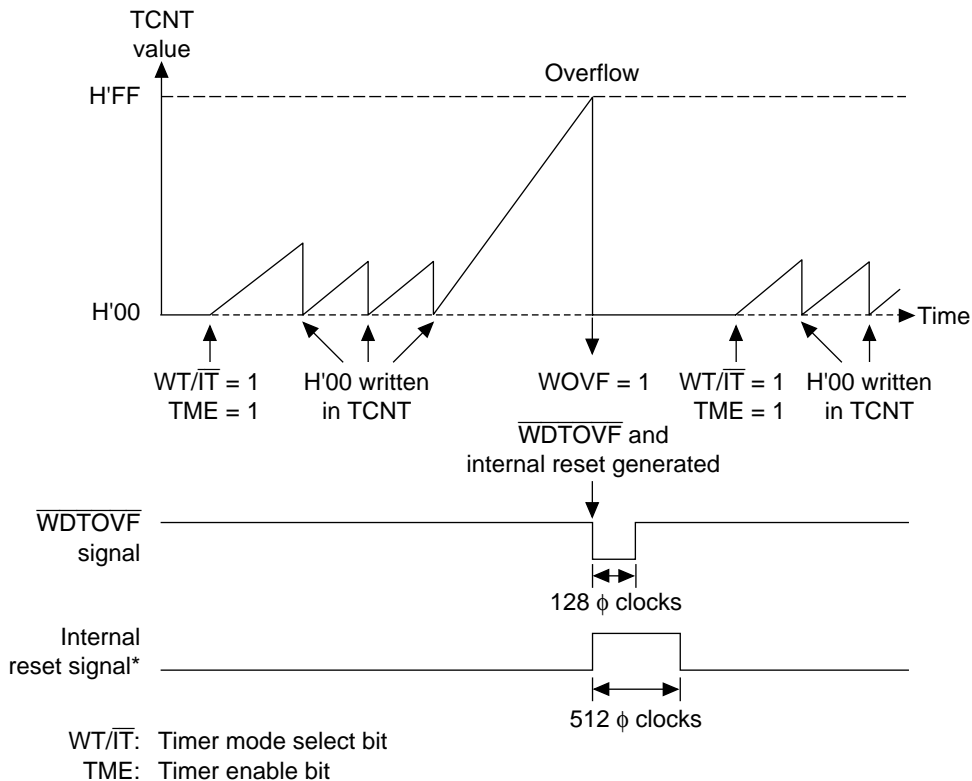
If the RSTE bit in the RSTCSR is set to 1, a signal to reset the chip will be generated internally simultaneous to the \overline{WDTOVF} signal when TCNT overflows. The internal reset signal is output for 512 ϕ clock cycles.

When a watchdog overflow reset is generated simultaneously with a reset input at the \overline{RES} pin, the \overline{RES} reset takes priority, and the WOVF bit is cleared to 0.

The following are not initialized a WDT reset signal:

- PFC (Pin Function Controller) function register
- I/O port register

Initializing is only possible by external power-on reset.



Note: Internal reset signal occurs only when the RSTE bit is set to 1.

Figure 11.4 Operation in the Watchdog Timer Mode

11.3.2 Interval Timer Mode

To use the WDT as an interval timer, clear WT/\overline{IT} to 0 and set TME to 1. An interval timer interrupt (ITI) is generated each time the timer counter overflows. This function can be used to generate interval timer interrupts at regular intervals (figure 11.5).

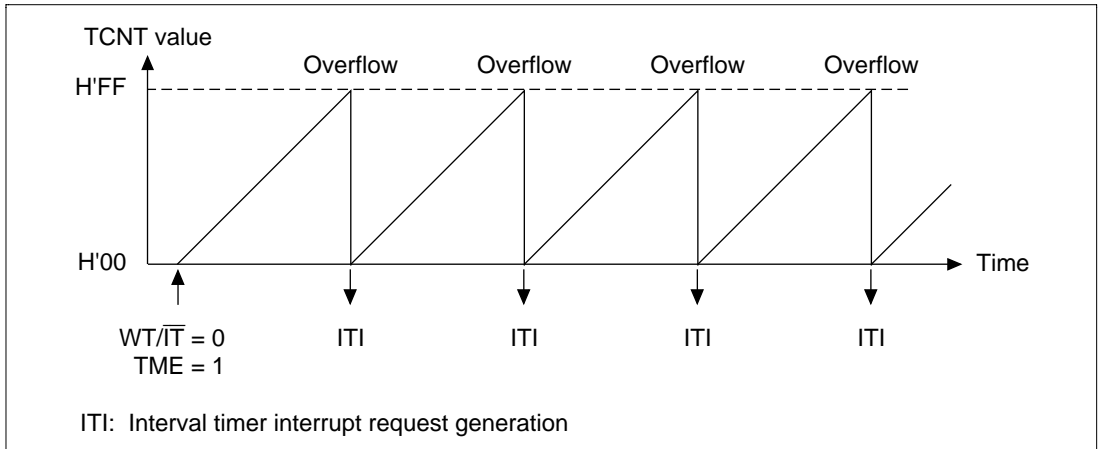


Figure 11.5 Operation in the Interval Timer Mode

11.3.3 Clearing the Standby Mode

The watchdog timer has a special function to clear the standby mode with an NMI interrupt. When using the standby mode, set the WDT as described below.

Before Transition to the Standby Mode: The TME bit in the TCSR must be cleared to 0 to stop the watchdog timer counter before it enters the standby mode. The chip cannot enter the standby mode while the TME bit is set to 1. Set bits CKS2 to CKS0 so that the counter overflow interval is equal to or longer than the oscillation settling time. See section 20.3, AC Characteristics, for the oscillation settling time.

Recovery from the Standby Mode: When an NMI request signal is received in standby mode, the clock oscillator starts running and the watchdog timer starts incrementing at the rate selected by bits CKS2 to CKS0 before the standby mode was entered. When the TCNT overflows (changes from H'FF to H'00), the clock is presumed to be stable and usable; clock signals are supplied to the entire chip and the standby mode ends.

For details on the standby mode, see section 19, Power Down State.

11.3.4 Timing of Setting the Overflow Flag (OVF)

In the interval timer mode, when the TCNT overflows, the OVF flag of the TCSR is set to 1 and an interval timer interrupt is simultaneously requested (figure 11.6).

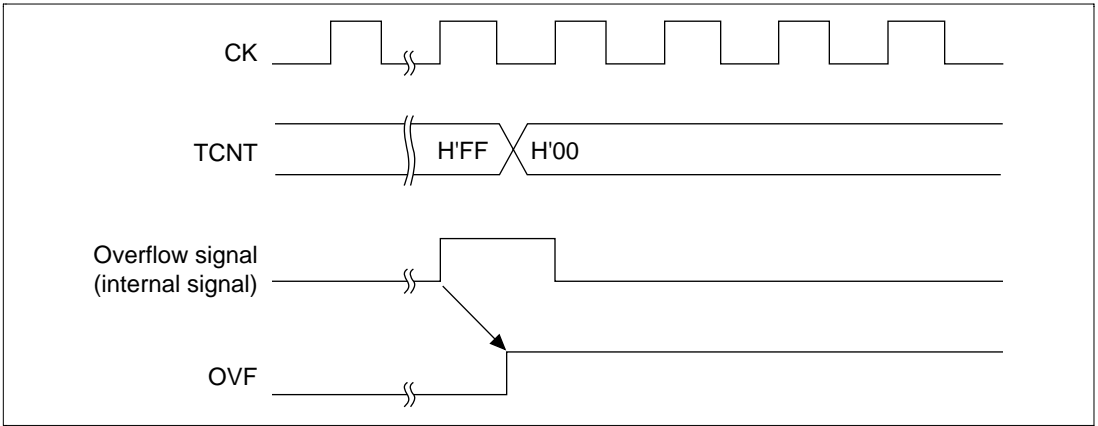


Figure 11.6 Timing of Setting the OVF

11.3.5 Timing of Setting the Watchdog Timer Overflow Flag (WOVF)

When the TCNT overflows in the watchdog timer mode, the WOVF bit of the RSTCSR is set to 1 and a WDTOVF signal is output. When the RSTE bit is set to 1, TCNT overflow enables an internal reset signal to be generated for the entire chip (figure 11.7).

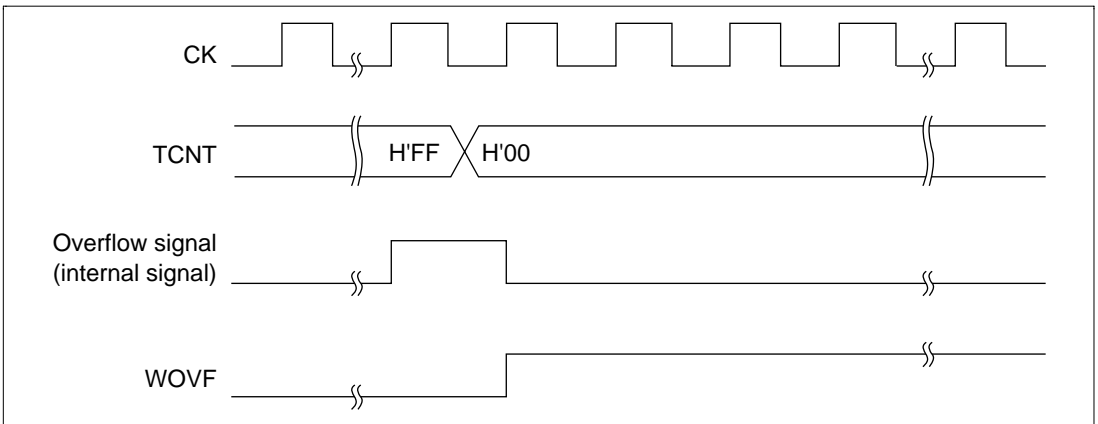


Figure 11.7 Timing of Setting the WOVF Bit

11.4 Notes on Use

11.4.1 TCNT Write and Increment Contention

If a timer counter increment clock pulse is generated during the T_3 state of a write cycle to the TCNT, the write takes priority and the timer counter is not incremented (figure 11.8).

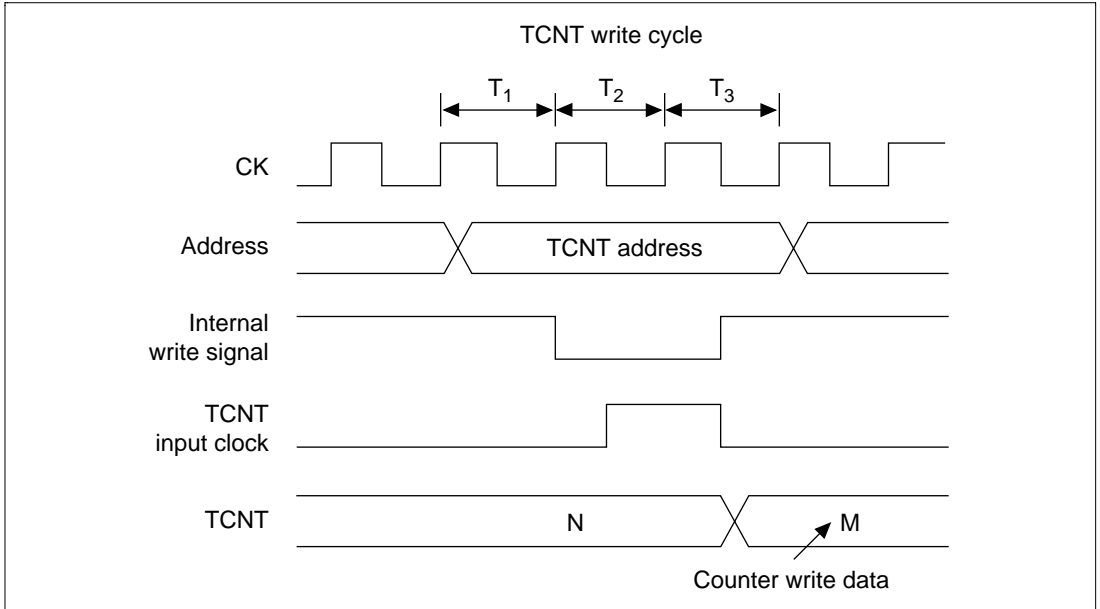


Figure 11.8 Contention between TCNT Write and Increment

11.4.2 Changing CKS2 to CKS0 Bit Values

If the values of bits CKS2 to CKS0 are altered while the WDT is running, the count may increment incorrectly. Always stop the watchdog timer (by clearing the TME bit to 0) before changing the values of bits CKS2 to CKS0.

11.4.3 Changing between Watchdog Timer/Interval Timer Modes

To prevent incorrect operation, always stop the watchdog timer (by clearing the TME bit to 0) before switching between interval timer mode and watchdog timer mode.

11.4.4 System Reset with $\overline{\text{WDTOVF}}$

If a $\overline{\text{WDTOVF}}$ signal is input to the $\overline{\text{RES}}$ pin, the LSI cannot initialize correctly.

Avoid logical input of the $\overline{\text{WDTOVF}}$ output signal to the $\overline{\text{RES}}$ input pin. To reset the entire system with the $\overline{\text{WDTOVF}}$ signal, use the circuit shown in figure 11.9.

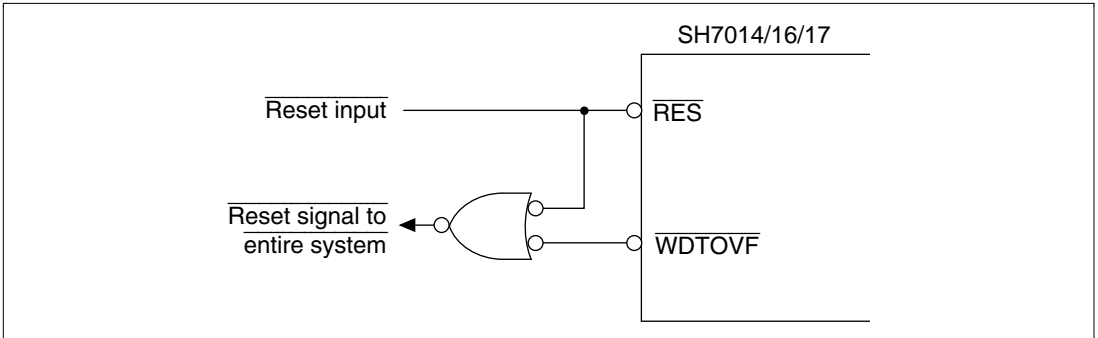


Figure 11.9 Example of a System Reset Circuit with a $\overline{\text{WDTOVF}}$ Signal

11.4.5 Internal Reset with the Watchdog Timer

If the RSTE bit is cleared to 0 in the watchdog timer mode, the LSI will not reset internally when a TCNT overflow occurs, but the TCNT and TCSR in the WDT will reset.

Section 12 Serial Communication Interface (SCI)

12.1 Overview

This LSI has a serial communication interface (SCI) with two independent channels, both of which possess the same functions.

The SCI supports both asynchronous and clock synchronous serial communication. It also has a multiprocessor communication function for serial communication among two or more processors.

12.1.1 Features

- Select asynchronous or clock synchronous as the serial communications mode.
 - Asynchronous mode: Serial data communications are synched by start-stop in character units. The SCI can communicate with a universal asynchronous receiver/transmitter (UART), an asynchronous communication interface adapter (ACIA), or any other chip that employs a standard asynchronous serial communication. It can also communicate with two or more other processors using the multiprocessor communication function. There are twelve selectable serial data communication formats.
 - Data length: seven or eight bits
 - Stop bit length: one or two bits
 - Parity: even, odd, or none
 - Multiprocessor bit: one or none
 - Receive error detection: parity, overrun, and framing errors
 - Break detection: by reading the RxD level directly when a framing error occurs
 - Clocked synchronous mode: Serial data communication is synchronized with a clock signal. The SCI can communicate with other chips having a clock synchronous communication function. There is one serial data communication format.
 - Data length: eight bits
 - Receive error detection: overrun errors
- Full duplex communication: The transmitting and receiving sections are independent, so the SCI can transmit and receive simultaneously. Both sections use double buffering, so continuous data transfer is possible in both the transmit and receive directions.
- On-chip baud rate generator with selectable bit rates.
- Internal or external transmit/receive clock source: baud rate generator (internal) or SCK pin (external).
- Four types of interrupts: Transmit-data-empty, transmit-end, receive-data-full, and receive-error interrupts are requested independently. The transmit-data-empty and receive-data-full interrupts can start the direct memory access controller (DMAC) to transfer data.

12.1.2 Block Diagram

Figure 12.1 shows a block diagram of the SCI.

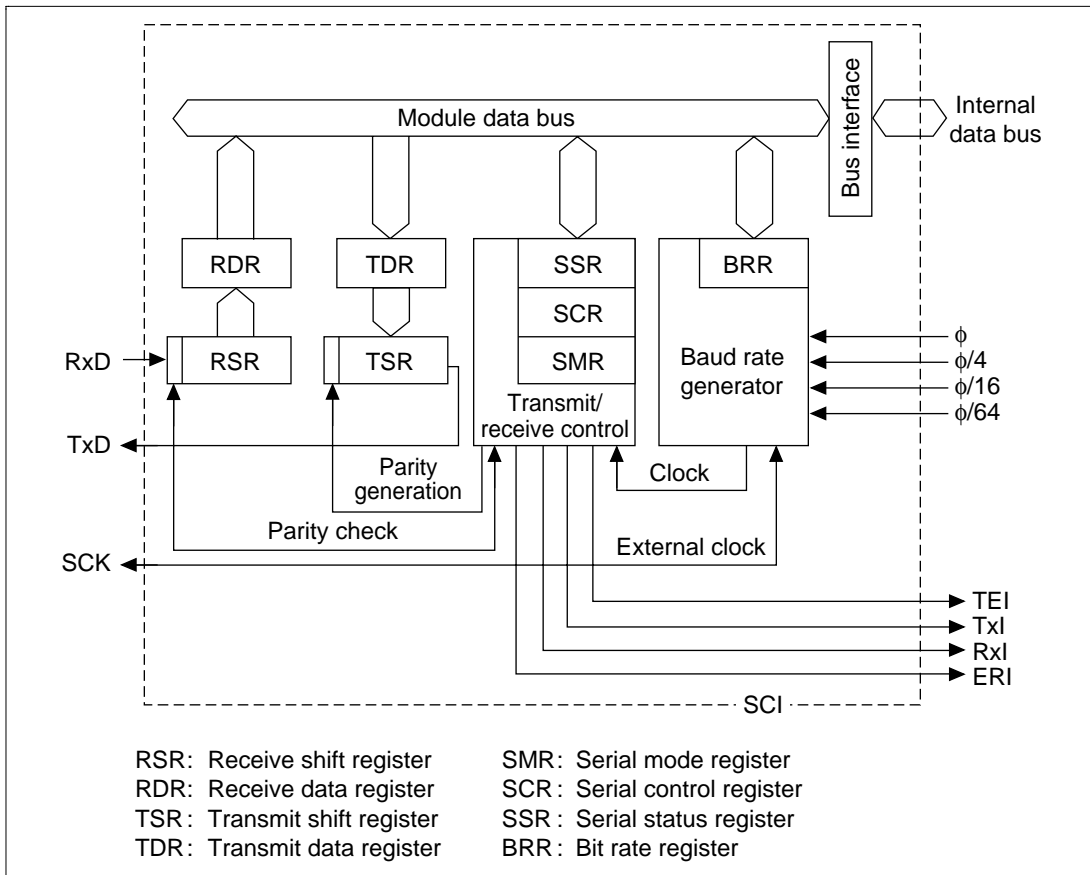


Figure 12.1 SCI Block Diagram

12.1.3 Pin Configuration

Table 12.1 summarizes the SCI pins by channel.

Table 12.1 SCI Pins

| Channel | Pin Name | Abbreviation | Input/Output | Function |
|---------|-------------------|--------------|--------------|---------------------------|
| 0 | Serial clock pin | SCK0 | Input/output | SCI0 clock input/output |
| | Receive data pin | RxD0 | Input | SCI0 receive data input |
| | Transmit data pin | TxD0 | Output | SCI0 transmit data output |
| 1 | Serial clock pin | SCK1 | Input/output | SCI1 clock input/output |
| | Receive data pin | RxD1 | Input | SCI1 receive data input |
| | Transmit data pin | TxD1 | Output | SCI1 transmit data output |

12.1.4 Register Configuration

Table 12.2 summarizes the SCI internal registers. These registers select the communication mode (asynchronous or clock synchronous), specify the data format and bit rate, and control the transmitter and receiver sections.

Table 12.2 Registers

| Channel | Name | Abbreviation | R/W | Initial Value | Address* ² | Access Size |
|---------|-------------------------|--------------|----------------------|---------------|-----------------------|-------------|
| 0 | Serial mode register | SMR0 | R/W | H'00 | H'FFFF81A0 | 8, 16 |
| | Bit rate register | BRR0 | R/W | H'FF | H'FFFF81A1 | 8, 16 |
| | Serial control register | SCR0 | R/W | H'00 | H'FFFF81A2 | 8, 16 |
| | Transmit data register | TDR0 | R/W | H'FF | H'FFFF81A3 | 8, 16 |
| | Serial status register | SSR0 | R/(W) * ¹ | H'84 | H'FFFF81A4 | 8, 16 |
| | Receive data register | RDR0 | R | H'00 | H'FFFF81A5 | 8, 16 |
| 1 | Serial mode register | SMR1 | R/W | H'00 | H'FFFF81B0 | 8, 16 |
| | Bit rate register | BRR1 | R/W | H'FF | H'FFFF81B1 | 8, 16 |
| | Serial control register | SCR1 | R/W | H'00 | H'FFFF81B2 | 8, 16 |
| | Transmit data register | TDR1 | R/W | H'FF | H'FFFF81B3 | 8, 16 |
| | Serial status register | SSR1 | R/(W) * ¹ | H'84 | H'FFFF81B4 | 8, 16 |
| | Receive data register | RDR1 | R | H'00 | H'FFFF81B5 | 8, 16 |

Notes: 1. The only value that can be written is a 0 to clear the flags.
2. Do not access empty addresses.

12.2 Register Descriptions

12.2.1 Receive Shift Register (RSR)

The receive shift register (RSR) receives serial data. Data input at the RxD pin is loaded into the RSR in the order received, LSB (bit 0) first, converting the data to parallel form. When one byte has been received, it is automatically transferred to the RDR.

The CPU cannot read or write the RSR directly.

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

12.2.2 Receive Data Register (RDR)

The receive data register (RDR) stores serial receive data. The SCI completes the reception of one byte of serial data by moving the received data from the receive shift register (RSR) into the RDR for storage. The RSR is then ready to receive the next data. This double buffering allows the SCI to receive data continuously.

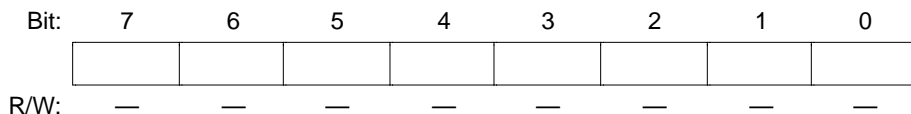
The CPU can read but not write the RDR. The RDR is initialized to H'00 by a power-on reset or in standby mode.

| | | | | | | | | |
|----------------|---|---|---|---|---|---|---|---|
| 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 |

12.2.3 Transmit Shift Register (TSR)

The transmit shift register (TSR) transmits serial data. The SCI loads transmit data from the transmit data register (TDR) into the TSR, then transmits the data serially from the TxD pin, LSB (bit 0) first. After transmitting one data byte, the SCI automatically loads the next transmit data from the TDR into the TSR and starts transmitting again. If the TDRE bit of the SSR is 1, however, the SCI does not load the TDR contents into the TSR.

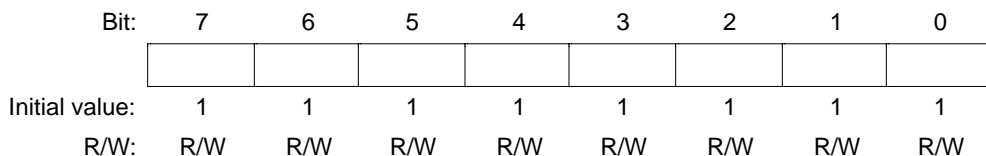
The CPU cannot read or write the TSR directly.



12.2.4 Transmit Data Register (TDR)

The transmit data register (TDR) is an 8-bit register that stores data for serial transmission. When the SCI detects that the transmit shift register (TSR) is empty, it moves transmit data written in the TDR into the TSR and starts serial transmission. Continuous serial transmission is possible by writing the next transmit data in the TDR during serial transmission from the TSR.

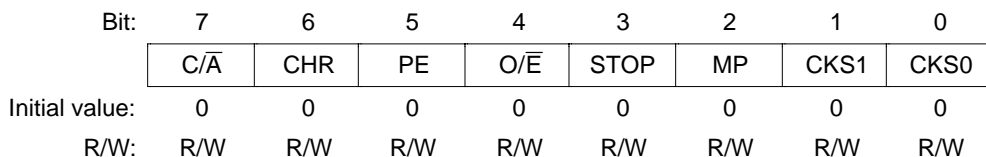
The CPU can always read and write the TDR. The TDR is initialized to H'FF by a power-on reset or in standby mode.



12.2.5 Serial Mode Register (SMR)

The serial mode register (SMR) is an 8-bit register that specifies the SCI serial communication format and selects the clock source for the baud rate generator.

The CPU can always read and write the SMR. The SMR is initialized to H'00 by a power-on reset or in standby mode.



- Bit 7—Communication Mode (C/\bar{A}): Selects whether the SCI operates in the asynchronous or clock synchronous mode.

| Bit 7: C/\bar{A} | Description |
|--------------------|-----------------------------------|
| 0 | Asynchronous mode (initial value) |
| 1 | Clocked synchronous mode |

- Bit 6—Character Length (CHR): Selects 7-bit or 8-bit data in the asynchronous mode. In the clock synchronous mode, the data length is always eight bits, regardless of the CHR setting.

| Bit 6: CHR | Description |
|------------|--|
| 0 | Eight-bit data (initial value) |
| 1 | Seven-bit data. (When 7-bit data is selected, the MSB (bit 7) of the transmit data register is not transmitted.) |

- Bit 5—Parity Enable (PE): Selects whether to add a parity bit to transmit data and to check the parity of receive data, in the asynchronous mode. In the clock synchronous mode, a parity bit is neither added nor checked, regardless of the PE setting.

| Bit 5: PE | Description |
|-----------|--|
| 0 | Parity bit not added or checked (initial value) |
| 1 | Parity bit added and checked. When PE is set to 1, an even or odd parity bit is added to transmit data, depending on the parity mode (O/\bar{E}) setting. Receive data parity is checked according to the even/odd (O/\bar{E}) mode setting. |

- Bit 4—Parity Mode (O/\bar{E}): Selects even or odd parity when parity bits are added and checked. The O/\bar{E} setting is used only in asynchronous mode and only when the parity enable bit (PE) is set to 1 to enable parity addition and check. The O/\bar{E} setting is ignored in the clock synchronous mode, or in the asynchronous mode when parity addition and check is disabled.

| Bit 4: O/\bar{E} | Description |
|--------------------|---|
| 0 | Even parity (initial value). If even parity is selected, the parity bit is added to transmit data to make an even number of 1s in the transmitted character and parity bit combined. Receive data is checked to see if it has an even number of 1s in the received character and parity bit combined. |
| 1 | Odd parity. If odd parity is selected, the parity bit is added to transmit data to make an odd number of 1s in the transmitted character and parity bit combined. Receive data is checked to see if it has an odd number of 1s in the received character and parity bit combined. |

- **Bit 3—Stop Bit Length (STOP):** Selects one or two bits as the stop bit length in the asynchronous mode. This setting is used only in the asynchronous mode. It is ignored in the clock synchronous mode because no stop bits are added.

In receiving, 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, but if the second stop bit is 0, it is treated as the start bit of the next incoming character.

| Bit 3: STOP | Description |
|-------------|---|
| 0 | One stop bit (initial value). In transmitting, a single bit of 1 is added at the end of each transmitted character. |
| 1 | Two stop bits. In transmitting, two bits of 1 are added at the end of each transmitted character. |

- **Bit 2—Multiprocessor Mode (MP):** Selects multiprocessor format. When multiprocessor format is selected, settings of the parity enable (PE) and parity mode (O/\bar{E}) bits are ignored. The MP bit setting is used only in the asynchronous mode; it is ignored in the clock synchronous mode. For the multiprocessor communication function, see section 12.3.3, Multiprocessor Communication.

| Bit 2: MP | Description |
|-----------|--|
| 0 | Multiprocessor function disabled (initial value) |
| 1 | Multiprocessor format selected |

- **Bits 1 and 0—Clock Select 1 and 0 (CKS1 and CKS0):** These bits select the internal clock source of the on-chip baud rate generator. Four clock sources are available; ϕ , $\phi/4$, $\phi/16$, or $\phi/64$. For further information on the clock source, bit rate register settings, and baud rate, see section 12.2.8, Bit Rate Register.

| Bit 1: CKS1 | Bit 0: CKS0 | Description |
|-------------|-------------|------------------------|
| 0 | 0 | ϕ (initial value) |
| | 1 | $\phi/4$ |
| 1 | 0 | $\phi/16$ |
| | 1 | $\phi/64$ |

12.2.6 Serial Control Register (SCR)

The serial control register (SCR) operates the SCI transmitter/receiver, selects the serial clock output in the asynchronous mode, enables/disables interrupt requests, and selects the transmit/receive clock source. The CPU can always read and write the SCR. The SCR is initialized to H'00 by a power-on reset or in standby mode.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|------|------|------|------|
| 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 |

- Bit 7—Transmit Interrupt Enable (TIE): Enables or disables the transmit-data-empty interrupt (TxI) requested when the transmit data register empty bit (TDRE) in the serial status register (SSR) is set to 1 by transfer of serial transmit data from the TDR to the TSR.

| Bit 7: TIE | Description |
|------------|---|
| 0 | Transmit-data-empty interrupt request (TxI) is disabled (initial value). The TxI interrupt request can be cleared by reading TDRE after it has been set to 1, then clearing TDRE to 0, or by clearing TIE to 0. |
| 1 | Transmit-data-empty interrupt request (TxI) is enabled |

- Bit 6—Receive Interrupt Enable (RIE): Enables or disables the receive-data-full interrupt (RxI) requested when the receive data register full bit (RDRF) in the serial status register (SSR) is set to 1 by transfer of serial receive data from the RSR to the RDR. It also enables or disables receive-error interrupt (ERI) requests.

| Bit 6: RIE | Description |
|------------|---|
| 0 | Receive-data-full interrupt (RxI) and receive-error interrupt (ERI) requests are disabled (initial value). RxI and ERI interrupt requests can be cleared by reading the RDRF flag or error flag (FER, PER, or ORER) after it has been set to 1, then clearing the flag to 0, or by clearing RIE to 0. |
| 1 | Receive-data-full interrupt (RxI) and receive-error interrupt (ERI) requests are enabled. |

- Bit 5—Transmit Enable (TE): Enables or disables the SCI serial transmitter.

| Bit 5: TE | Description |
|-----------|---|
| 0 | Transmitter disabled (initial value). The transmit data register empty bit (TDRE) in the serial status register (SSR) is locked at 1. |
| 1 | Transmitter enabled. Serial transmission starts when the transmit data register empty (TDRE) bit in the serial status register (SSR) is cleared to 0 after writing of transmit data into the TDR. Select the transmit format in the SMR before setting TE to 1. |

- Bit 4—Receive Enable (RE): Enables or disables the SCI serial receiver.

| Bit 4: RE | Description |
|-----------|---|
| 0 | Receiver disabled (initial value). Clearing RE to 0 does not affect the receive flags (RDRF, FER, PER, ORER). These flags retain their previous values. |
| 1 | Receiver enabled. Serial reception starts when a start bit is detected in the asynchronous mode, or synchronous clock input is detected in the clock synchronous mode. Select the receive format in the SMR before setting RE to 1. |

- Bit 3—Multiprocessor Interrupt Enable (MPIE): Enables or disables multiprocessor interrupts. The MPIE setting is used only in the asynchronous mode, and only if the multiprocessor mode bit (MP) in the serial mode register (SMR) is set to 1 during reception. The MPIE setting is ignored in the clock synchronous mode or when the MP bit is cleared to 0.

| Bit 3: MPIE | Description |
|-------------|---|
| 0 | Multiprocessor interrupts are disabled (normal receive operation) (initial value). MPIE is cleared when the MPIE bit is cleared to 0, or the multiprocessor bit (MPB) is set to 1 in receive data. |
| 1 | Multiprocessor interrupts are enabled. Receive-data-full interrupt requests (RxI), receive-error interrupt requests (ERI), and setting of the RDRF, FER, and ORER status flags in the serial status register (SSR) are disabled until data with the multiprocessor bit set to 1 is received. The SCI does not transfer receive data from the RSR to the RDR, does not detect receive errors, and does not set the RDRF, FER, and ORER flags in the serial status register (SSR). When it receives data that includes MPB = 1, MPB is set to 1, and the SCI automatically clears MPIE to 0, generates RxI and ERI interrupts (if the TIE and RIE bits in the SCR are set to 1), and allows the FER and ORER bits to be set. |

- Bit 2—Transmit-End Interrupt Enable (TEIE): Enables or disables the transmit-end interrupt (TEI) requested if TDR does not contain valid transmit data when the MSB is transmitted.

| Bit 2: TEIE | Description |
|-------------|---|
| 0 | Transmit-end interrupt (TEI) requests are disabled* (initial value) |
| 1 | Transmit-end interrupt (TEI) requests are enabled.* |

Note: The TEI request can be cleared by reading the TDRE bit in the serial status register (SSR) after it has been set to 1, then clearing TDRE to 0 and clearing the transmit end (TEND) bit to 0; or by clearing the TEIE bit to 0.

- Bits 1 and 0—Clock Enable 1 and 0 (CKE1 and CKE0): These bits select the SCI clock source and enable or disable clock output from the SCK pin. Depending on the combination of CKE1 and CKE0, the SCK pin can be used for serial clock output, or serial clock input. Select the SCK pin function by using the pin function controller (PFC).

The CKE0 setting is valid only in the asynchronous mode, and only when the SCI is internally clocked (CKE1 = 0). The CKE0 setting is ignored in the clock synchronous mode, or when an external clock source is selected (CKE1 = 1). Select the SCI operating mode in the serial mode register (SMR) before setting CKE1 and CKE0. For further details on selection of the SCI clock source, see table 12.10 in section 12.3, Operation.

Bit 1: Bit 0:

| CKE1 | CKE0 | Description ¹ | |
|------|------|--------------------------|---|
| 0 | 0 | Asynchronous mode | Internal clock, SCK pin used for input pin (input signal is ignored) or output pin (output level is undefined) ² |
| | | Clock synchronous mode | Internal clock, SCK pin used for synchronous clock output ² |
| 0 | 1 | Asynchronous mode | Internal clock, SCK pin used for clock output ³ |
| | | Clock synchronous mode | Internal clock, SCK pin used for synchronous clock output |
| 1 | 0 | Asynchronous mode | External clock, SCK pin used for clock input ⁴ |
| | | Clock synchronous mode | External clock, SCK pin used for synchronous clock input |
| 1 | 1 | Asynchronous mode | External clock, SCK pin used for clock input ⁴ |
| | | Clock synchronous mode | External clock, SCK pin used for synchronous clock input |

Notes: 1. The SCK pin is multiplexed with other functions. Use the pin function controller (PFC) to select the SCK function for this pin, as well as the I/O direction.

2. Initial value.

3. The output clock frequency is the same as the bit rate.

4. The input clock frequency is 16 times the bit rate.

12.2.7 Serial Status Register (SSR)

The serial status register (SSR) is an 8-bit register containing multiprocessor bit values, and status flags that indicate SCI operating status.

The CPU can always read and write the SSR, but cannot write 1 in the status flags (TDRE, RDRF, ORER, PER, and FER). These flags can be cleared to 0 only if they have first been read (after being set to 1). Bits 2 (TEND) and 1 (MPB) are read-only bits that cannot be written. The SSR is initialized to H'84 by a power-on reset or in standby mode.

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|------|-----|------|
| 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: The only value that can be written is a 0 to clear the flag.

- Bit 7—Transmit Data Register Empty (TDRE): Indicates that the SCI has loaded transmit data from the TDR into the TSR and new serial transmit data can be written in the TDR.

Bit 7: TDRE Description

| | |
|---|--|
| 0 | TDR contains valid transmit data TDRE is cleared to 0 when software reads TDRE after it has been set to 1, then writes 0 in TDRE or the DMAC writes data in TDR |
| 1 | TDR does not contain valid transmit data (initial value) TDRE is set to 1 when the chip is power-on reset or enters standby mode, the TE bit in the serial control register (SCR) is cleared to 0, or TDR contents are loaded into TSR, so new data can be written in TDR |

- Bit 6—Receive Data Register Full (RDRF): Indicates that RDR contains received data.

| Bit 6: RDRF | Description |
|-------------|---|
| 0 | RDR does not contain valid received data (initial value) RDRF is cleared to 0 when the chip is power-on reset or enters standby mode, software reads RDRF after it has been set to 1, then writes 0 in RDRF, or the DMAC reads data from RDR |
| 1 | RDR contains valid received data RDRF is set to 1 when serial data is received normally and transferred from RSR to RDR |

Note: The RDR and RDRF are not affected by detection of receive errors or by clearing of the RE bit to 0 in the serial control register. They retain their previous contents. If RDRF is still set to 1 when reception of the next data ends, an overrun error (ORER) occurs and the received data is lost.

- Bit 5—Overrun Error (ORER): Indicates that data reception ended abnormally due to an overrun error.

| Bit 5: ORER | Description |
|-------------|--|
| 0 | Receiving is in progress or has ended normally (initial value). Clearing the RE bit to 0 in the serial control register does not affect the ORER bit, which retains its previous value. ORER is cleared to 0 when the chip is power-on reset or enters standby mode or software reads ORER after it has been set to 1, then writes 0 in ORER |
| 1 | A receive overrun error occurred. RDR continues to hold the data received before the overrun error, so subsequent receive data is lost. Serial receiving cannot continue while ORER is set to 1. In the clock synchronous mode, serial transmitting is disabled. ORER is set to 1 if reception of the next serial data ends when RDRF is set to 1 |

- Bit 4—Framing Error (FER): Indicates that data reception ended abnormally due to a framing error in the asynchronous mode.

| Bit 4: FER | Description |
|------------|---|
| 0 | <p>Receiving is in progress or has ended normally (initial value). Clearing the RE bit to 0 in the serial control register does not affect the FER bit, which retains its previous value.</p> <p>FER is cleared to 0 when the chip is power-on reset or enters standby mode or software reads FER after it has been set to 1, then writes 0 in FER</p> |
| 1 | <p>A receive framing error occurred. When the stop bit length is two bits, only the first bit is checked to see if it is a 1. The second stop bit is not checked. When a framing error occurs, the SCI transfers the receive data into the RDR but does not set RDRF. Serial receiving cannot continue while FER is set to 1. In the clock synchronous mode, serial transmitting is also disabled.</p> <p>FER is set to 1 if the stop bit at the end of receive data is checked and found to be 0</p> |

- Bit 3—Parity Error (PER): Indicates that data reception (with parity) ended abnormally due to a parity error in the asynchronous mode.

| Bit 3: PER | Description |
|------------|---|
| 0 | <p>Receiving is in progress or has ended normally (initial value). Clearing the RE bit to 0 in the serial control register does not affect the PER bit, which retains its previous value.</p> <p>PER is cleared to 0 when the chip is power-on reset or enters standby mode or software reads PER after it has been set to 1, then writes 0 in PER</p> |
| 1 | <p>A receive parity error occurred. When a parity error occurs, the SCI transfers the receive data into the RDR but does not set RDRF. Serial receiving cannot continue while PER is set to 1. In the clock synchronous mode, serial transmitting is also disabled.</p> <p>PER is set to 1 if the number of 1s in receive data, including the parity bit, does not match the even or odd parity setting of the parity mode bit (O/\bar{E}) in the serial mode register (SMR)</p> |

- **Bit 2—Transmit End (TEND):** Indicates that when the last bit of a serial character was transmitted, the TDR did not contain valid data, so transmission has ended. TEND is a read-only bit and cannot be written.

| Bit 2: TEND | Description |
|--------------------|--|
| 0 | Transmission is in progress TEND is cleared to 0 when software reads TDRE after it has been set to 1, then writes 0 in TDRE, or the DMAC writes data in TDR |
| 1 | End of transmission (initial value) TEND is set to 1 when the chip is power-on reset or enters standby mode, TE is cleared to 0 in the serial control register (SCR), or TDRE is 1 when the last bit of a one-byte serial character is transmitted. |

- **Bit 1—Multiprocessor Bit (MPB):** Stores the value of the multiprocessor bit in receive data when a multiprocessor format is selected for receiving in the asynchronous mode. The MPB is a read-only bit and cannot be written.

| Bit 1: MPB | Description |
|-------------------|--|
| 0 | Multiprocessor bit value in receive data is 0 (initial value). If RE is cleared to 0 when a multiprocessor format is selected, the MPB retains its previous value. |
| 1 | Multiprocessor bit value in receive data is 1 |

- **Bit 0—Multiprocessor Bit Transfer (MPBT):** Stores the value of the multiprocessor bit added to transmit data when a multiprocessor format is selected for transmitting in the asynchronous mode. The MPBT setting is ignored in the clock synchronous mode, when a multiprocessor format is not selected, or when the SCI is not transmitting.

| Bit 0: MPBT | Description |
|--------------------|--|
| 0 | Multiprocessor bit value in transmit data is 0 (initial value) |
| 1 | Multiprocessor bit value in transmit data is 1 |

12.2.8 Bit Rate Register (BRR)

The bit rate register (BRR) is an 8-bit register that, together with the baud rate generator clock source selected by the CKS1 and CKS0 bits in the serial mode register (SMR), determines the serial transmit/receive bit rate.

The CPU can always read and write the BRR. The BRR is initialized to H'FF by a power-on reset or in standby mode. Each channel has independent baud rate generator control, so different values can be set in the two channels.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| 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 |

Table 12.3 lists examples of BRR settings in the asynchronous mode; table 12.4 lists examples of BRR settings in the clock synchronous mode.

Table 12.3 Bit Rates and BRR Settings in Asynchronous Mode

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | | | | |
|----------------------|--------------|-----|-----------|--------|-----|-----------|---|-----|-----------|
| | 4 | | | 4.9152 | | | 6 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 70 | 0.03 | 2 | 86 | 0.31 | 2 | 106 | -0.44 |
| 150 | 1 | 207 | 0.16 | 1 | 255 | 0.00 | 2 | 77 | 0.16 |
| 300 | 1 | 103 | 0.16 | 1 | 127 | 0.00 | 1 | 155 | 0.16 |
| 600 | 0 | 207 | 0.16 | 0 | 255 | 0.00 | 1 | 77 | 0.16 |
| 1200 | 0 | 103 | 0.16 | 0 | 127 | 0.00 | 0 | 155 | 0.16 |
| 2400 | 0 | 51 | 0.16 | 0 | 63 | 0.00 | 0 | 77 | 0.16 |
| 4800 | 0 | 25 | 0.16 | 0 | 31 | 0.00 | 0 | 38 | 0.16 |
| 9600 | 0 | 12 | 0.16 | 0 | 15 | 0.00 | 0 | 19 | -2.34 |
| 14400 | 0 | 8 | -3.55 | 0 | 10 | -3.03 | 0 | 12 | 0.16 |
| 19200 | 0 | 6 | -6.99 | 0 | 7 | 0.00 | 0 | 9 | -2.34 |
| 28800 | 0 | 3 | 8.51 | 0 | 4 | 6.67 | 0 | 6 | -6.99 |
| 31250 | 0 | 3 | 0.00 | 0 | 4 | -1.70 | 0 | 5 | 0.00 |
| 38400 | 0 | 2 | 8.51 | 0 | 3 | 0.00 | 0 | 4 | -2.34 |

Table 12.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | | | | |
|----------------------|--------------|-----|-----------|---|-----|-----------|--------|-----|-----------|
| | 7.3728 | | | 8 | | | 9.8304 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 130 | -0.07 | 2 | 141 | 0.03 | 2 | 174 | -0.26 |
| 150 | 2 | 95 | 0.00 | 2 | 103 | 0.16 | 2 | 127 | 0.00 |
| 300 | 1 | 191 | 0.00 | 1 | 207 | 0.16 | 1 | 255 | 0.00 |
| 600 | 1 | 95 | 0.00 | 1 | 103 | 0.16 | 1 | 127 | 0.00 |
| 1200 | 0 | 191 | 0.00 | 0 | 207 | 0.16 | 0 | 255 | 0.00 |
| 2400 | 0 | 95 | 0.00 | 0 | 103 | 0.16 | 0 | 127 | 0.00 |
| 4800 | 0 | 47 | 0.00 | 0 | 51 | 0.16 | 0 | 63 | 0.00 |
| 9600 | 0 | 23 | 0.00 | 0 | 25 | 0.16 | 0 | 31 | 0.00 |
| 14400 | 0 | 15 | 0.00 | 0 | 16 | 2.12 | 0 | 20 | 1.59 |
| 19200 | 0 | 11 | 0.00 | 0 | 12 | 0.16 | 0 | 15 | 0.00 |
| 28800 | 0 | 7 | 0.00 | 0 | 8 | -3.55 | 0 | 10 | -3.03 |
| 31250 | 0 | 6 | 5.33 | 0 | 7 | 0.00 | 0 | 9 | -1.70 |
| 38400 | 0 | 5 | 0.00 | 0 | 6 | -6.99 | 0 | 7 | 0.00 |

Table 12.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | | | | |
|----------------------|--------------|-----|-----------|---------|-----|-----------|----|-----|-----------|
| | 10 | | | 11.0592 | | | 12 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 177 | -0.25 | 2 | 195 | 0.19 | 2 | 212 | 0.03 |
| 150 | 2 | 129 | 0.16 | 2 | 143 | 0.00 | 2 | 155 | 0.16 |
| 300 | 2 | 64 | 0.16 | 2 | 71 | 0.00 | 2 | 77 | 0.16 |
| 600 | 1 | 129 | 0.16 | 1 | 143 | 0.00 | 1 | 155 | 0.16 |
| 1200 | 1 | 64 | 0.16 | 1 | 71 | 0.00 | 1 | 77 | 0.16 |
| 2400 | 0 | 129 | 0.16 | 0 | 143 | 0.00 | 0 | 155 | 0.16 |
| 4800 | 0 | 64 | 0.16 | 0 | 71 | 0.00 | 0 | 77 | 0.16 |
| 9600 | 0 | 32 | -1.36 | 0 | 35 | 0.00 | 0 | 38 | 0.16 |
| 14400 | 0 | 21 | -1.36 | 0 | 23 | 0.00 | 0 | 25 | 0.16 |
| 19200 | 0 | 15 | 1.73 | 0 | 17 | 0.00 | 0 | 19 | -2.34 |
| 28800 | 0 | 10 | -1.36 | 0 | 11 | 0.00 | 0 | 12 | 0.16 |
| 31250 | 0 | 9 | 0.00 | 0 | 10 | 0.54 | 0 | 11 | 0.00 |
| 38400 | 0 | 7 | 1.73 | 0 | 8 | 0.00 | 0 | 9 | -2.34 |

Table 12.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | | | | |
|----------------------|--------------|-----|-----------|----|-----|-----------|---------|-----|-----------|
| | 12.288 | | | 14 | | | 14.7456 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 2 | 217 | 0.08 | 2 | 248 | -0.17 | 3 | 64 | 0.70 |
| 150 | 2 | 159 | 0.00 | 2 | 181 | 0.16 | 2 | 191 | 0.00 |
| 300 | 2 | 79 | 0.00 | 2 | 90 | 0.16 | 2 | 95 | 0.00 |
| 600 | 1 | 159 | 0.00 | 1 | 181 | 0.16 | 1 | 191 | 0.00 |
| 1200 | 1 | 79 | 0.00 | 1 | 90 | 0.16 | 1 | 95 | 0.00 |
| 2400 | 0 | 159 | 0.00 | 0 | 181 | 0.16 | 0 | 191 | 0.00 |
| 4800 | 0 | 79 | 0.00 | 0 | 90 | 0.16 | 0 | 95 | 0.00 |
| 9600 | 0 | 39 | 0.00 | 0 | 45 | -0.93 | 0 | 47 | 0.00 |
| 14400 | 0 | 26 | -1.23 | 0 | 29 | 1.27 | 0 | 31 | 0.00 |
| 19200 | 0 | 19 | 0.00 | 0 | 22 | -0.93 | 0 | 23 | 0.00 |
| 28800 | 0 | 12 | 2.56 | 0 | 14 | 1.27 | 0 | 15 | 0.00 |
| 31250 | 0 | 11 | 2.40 | 0 | 13 | 0.00 | 0 | 14 | -1.70 |
| 38400 | 0 | 9 | 0.00 | 0 | 10 | 3.57 | 0 | 11 | 0.00 |

Table 12.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | | | | |
|----------------------|--------------|-----|-----------|---------|-----|-----------|----|-----|-----------|
| | 16 | | | 17.2032 | | | 18 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 70 | 0.03 | 3 | 75 | 0.48 | 3 | 79 | -0.12 |
| 150 | 2 | 207 | 0.16 | 2 | 223 | 0.00 | 2 | 233 | 0.16 |
| 300 | 2 | 103 | 0.16 | 2 | 111 | 0.00 | 2 | 116 | 0.16 |
| 600 | 1 | 207 | 0.16 | 1 | 223 | 0.00 | 1 | 233 | 0.16 |
| 1200 | 1 | 103 | 0.16 | 1 | 111 | 0.00 | 1 | 116 | 0.16 |
| 2400 | 0 | 207 | 0.16 | 0 | 223 | 0.00 | 0 | 233 | 0.16 |
| 4800 | 0 | 103 | 0.16 | 0 | 111 | 0.00 | 0 | 116 | 0.16 |
| 9600 | 0 | 51 | 0.16 | 0 | 55 | 0.00 | 0 | 58 | -0.69 |
| 14400 | 0 | 34 | -0.79 | 0 | 36 | 0.90 | 0 | 38 | 0.16 |
| 19200 | 0 | 25 | 0.16 | 0 | 27 | 0.00 | 0 | 28 | 1.02 |
| 28800 | 0 | 16 | 2.12 | 0 | 18 | -1.75 | 0 | 19 | -2.34 |
| 31250 | 0 | 15 | 0.00 | 0 | 16 | 1.20 | 0 | 17 | 0.00 |
| 38400 | 0 | 12 | 0.16 | 0 | 13 | 0.00 | 0 | 14 | -2.34 |

Table 12.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | | | | |
|----------------------|--------------|-----|-----------|---------|-----|-----------|----|-----|-----------|
| | 18.432 | | | 19.6608 | | | 20 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 81 | -0.22 | 3 | 86 | 0.31 | 3 | 88 | -0.25 |
| 150 | 2 | 239 | 0.00 | 2 | 255 | 0.00 | 3 | 64 | 0.16 |
| 300 | 2 | 119 | 0.00 | 2 | 127 | 0.00 | 2 | 129 | 0.16 |
| 600 | 1 | 239 | 0.00 | 1 | 255 | 0.00 | 2 | 64 | 0.16 |
| 1200 | 1 | 119 | 0.00 | 1 | 127 | 0.00 | 1 | 129 | 0.16 |
| 2400 | 0 | 239 | 0.00 | 0 | 255 | 0.00 | 1 | 64 | 0.16 |
| 4800 | 0 | 119 | 0.00 | 0 | 127 | 0.00 | 0 | 129 | 0.16 |
| 9600 | 0 | 59 | 0.00 | 0 | 63 | 0.00 | 0 | 64 | 0.16 |
| 14400 | 0 | 39 | 0.00 | 0 | 42 | -0.78 | 0 | 42 | 0.94 |
| 19200 | 0 | 29 | 0.00 | 0 | 31 | 0.00 | 0 | 32 | -1.36 |
| 28800 | 0 | 19 | 0.00 | 0 | 20 | 1.59 | 0 | 21 | -1.36 |
| 31250 | 0 | 17 | 2.40 | 0 | 19 | -1.70 | 0 | 19 | 0.00 |
| 38400 | 0 | 14 | 0.00 | 0 | 15 | 0.00 | 0 | 15 | 1.73 |

Table 12.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | | | | |
|----------------------|--------------|-----|-----------|---------|-----|-----------|----|-----|-----------|
| | 22 | | | 22.1184 | | | 24 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 97 | -0.35 | 3 | 97 | 0.19 | 3 | 106 | -0.44 |
| 150 | 3 | 71 | -0.54 | 3 | 71 | 0.00 | 3 | 77 | 0.16 |
| 300 | 2 | 142 | 0.16 | 2 | 143 | 0.00 | 2 | 155 | 0.16 |
| 600 | 2 | 71 | -0.54 | 2 | 71 | 0.00 | 2 | 77 | 0.16 |
| 1200 | 1 | 142 | 0.16 | 1 | 143 | 0.00 | 1 | 155 | 0.16 |
| 2400 | 1 | 71 | -0.54 | 1 | 71 | 0.00 | 1 | 77 | 0.16 |
| 4800 | 0 | 142 | 0.16 | 0 | 143 | 0.00 | 0 | 155 | 0.16 |
| 9600 | 0 | 71 | -0.54 | 0 | 71 | 0.00 | 0 | 77 | 0.16 |
| 14400 | 0 | 47 | -0.54 | 0 | 47 | 0.00 | 0 | 51 | 0.16 |
| 19200 | 0 | 35 | -0.54 | 0 | 35 | 0.00 | 0 | 38 | 0.16 |
| 28800 | 0 | 23 | -0.54 | 0 | 23 | 0.00 | 0 | 25 | 0.16 |
| 31250 | 0 | 21 | 0.00 | 0 | 21 | 0.54 | 0 | 23 | 0.00 |
| 38400 | 0 | 17 | -0.54 | 0 | 17 | 0.00 | 0 | 19 | -2.34 |

Table 12.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | | | | |
|----------------------|--------------|-----|-----------|---------|-----|-----------|----|-----|-----------|
| | 24.576 | | | 25.8048 | | | 26 | | |
| | n | N | Error (%) | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 108 | 0.08 | 3 | 114 | -0.40 | 3 | 114 | 0.36 |
| 150 | 3 | 79 | 0.00 | 3 | 83 | 0.00 | 3 | 84 | -0.43 |
| 300 | 2 | 159 | 0.00 | 2 | 167 | 0.00 | 2 | 168 | 0.16 |
| 600 | 2 | 79 | 0.00 | 2 | 83 | 0.00 | 2 | 84 | -0.43 |
| 1200 | 1 | 159 | 0.00 | 1 | 167 | 0.00 | 1 | 168 | 0.16 |
| 2400 | 1 | 79 | 0.00 | 1 | 83 | 0.00 | 1 | 84 | -0.43 |
| 4800 | 0 | 159 | 0.00 | 0 | 167 | 0.00 | 0 | 168 | 0.16 |
| 9600 | 0 | 79 | 0.00 | 0 | 83 | 0.00 | 0 | 84 | -0.43 |
| 14400 | 0 | 52 | 0.63 | 0 | 55 | 0.00 | 0 | 55 | 0.76 |
| 19200 | 0 | 39 | 0.00 | 0 | 41 | 0.00 | 0 | 41 | 0.76 |
| 28800 | 0 | 26 | -1.23 | 0 | 27 | 0.00 | 0 | 27 | 0.76 |
| 31250 | 0 | 24 | -1.70 | 0 | 25 | -0.75 | 0 | 25 | 0.00 |
| 38400 | 0 | 19 | 0.00 | 0 | 20 | 0.00 | 0 | 20 | 0.76 |

Table 12.3 Bit Rates and BRR Settings in Asynchronous Mode (cont)

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | |
|----------------------|--------------|-----|-----------|----|-----|-----------|
| | 27.0336 | | | 28 | | |
| | n | N | Error (%) | n | N | Error (%) |
| 110 | 3 | 119 | 0.00 | 3 | 123 | 0.23 |
| 150 | 3 | 87 | 0.00 | 3 | 90 | 0.16 |
| 300 | 2 | 175 | 0.00 | 2 | 181 | 0.16 |
| 600 | 2 | 87 | 0.00 | 2 | 90 | 0.16 |
| 1200 | 1 | 175 | 0.00 | 1 | 181 | 0.16 |
| 2400 | 1 | 87 | 0.00 | 1 | 90 | 0.16 |
| 4800 | 0 | 175 | 0.00 | 0 | 181 | 0.16 |
| 9600 | 0 | 87 | 0.00 | 0 | 90 | 0.16 |
| 14400 | 0 | 58 | -0.56 | 0 | 60 | -0.39 |
| 19200 | 0 | 43 | 0.00 | 0 | 45 | -0.93 |
| 28800 | 0 | 28 | 1.15 | 0 | 29 | 1.27 |
| 31250 | 0 | 26 | 0.12 | 0 | 27 | 0.00 |
| 38400 | 0 | 21 | 0.00 | 0 | 22 | -0.93 |

Table 12.4 Bit Rates and BRR Settings in Clocked Synchronous Mode

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | | | |
|----------------------|--------------|-----|---|-----|----|-----|----|-----|
| | 4 | | 8 | | 10 | | 12 | |
| | n | N | n | N | n | N | n | N |
| 110 | 3 | 141 | | | | | | |
| 250 | 2 | 249 | 3 | 124 | 3 | 155 | 3 | 187 |
| 500 | 2 | 124 | 2 | 249 | 3 | 77 | 3 | 93 |
| 1k | 1 | 249 | 2 | 124 | 2 | 155 | 2 | 187 |
| 2.5k | 1 | 99 | 1 | 199 | 1 | 249 | 2 | 74 |
| 5k | 0 | 199 | 1 | 99 | 1 | 124 | 1 | 149 |
| 10k | 0 | 99 | 0 | 199 | 0 | 249 | 1 | 74 |
| 25k | 0 | 39 | 0 | 79 | 0 | 99 | 0 | 119 |
| 50k | 0 | 19 | 0 | 39 | 0 | 49 | 0 | 59 |
| 100k | 0 | 9 | 0 | 19 | 0 | 24 | 0 | 29 |
| 250k | 0 | 3 | 0 | 7 | 0 | 9 | 0 | 11 |
| 500k | 0 | 1 | 0 | 3 | 0 | 4 | 0 | 5 |
| 1M | 0 | 0* | 0 | 1 | — | — | 0 | 2 |
| 2.5M | | | | | 0 | 0* | 0 | 0* |
| 5M | | | | | | | | |

Table 12.4 Bit Rates and BRR Settings in Clocked Synchronous Mode (cont)

| Bit Rate (Bits/s) | ϕ (MHz) | | | | | | | |
|----------------------|--------------|-----|----|-----|----|-----|----|-----|
| | 16 | | 20 | | 24 | | 28 | |
| | n | N | n | N | n | N | n | N |
| 110 | | | | | | | | |
| 250 | 3 | 249 | | | | | | |
| 500 | 3 | 124 | 3 | 155 | 3 | 187 | 3 | 218 |
| 1k | 2 | 249 | 3 | 77 | 3 | 93 | 3 | 108 |
| 2.5k | 2 | 99 | 2 | 124 | 2 | 149 | 2 | 174 |
| 5k | 1 | 199 | 1 | 249 | 2 | 74 | 2 | 87 |
| 10k | 1 | 99 | 1 | 124 | 1 | 149 | 1 | 174 |
| 25k | 0 | 159 | 0 | 199 | 0 | 239 | 1 | 69 |
| 50k | 0 | 79 | 0 | 99 | 0 | 119 | 0 | 139 |
| 100k | 0 | 39 | 0 | 49 | 0 | 59 | 0 | 69 |
| 250k | 0 | 15 | 0 | 19 | 0 | 23 | 0 | 27 |
| 500k | 0 | 7 | 0 | 9 | 0 | 11 | 0 | 13 |
| 1M | 0 | 3 | 0 | 4 | 0 | 5 | 0 | 6 |
| 2.5M | — | — | 0 | 1 | — | — | 0 | 2 |
| 5M | | | 0 | 0* | — | — | 0 | 1 |
| 7M | | | | | — | — | 0 | 0* |

Note: Settings with an error of 1% or less are recommended.

Legend

Blank: No setting available

—: Setting possible, but error occurs

*: Continuous transmission/reception is not possible.

The BRR setting is calculated as follows:

Asynchronous mode:

$$N = \frac{\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Synchronous mode:

$$N = \frac{\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

B: Bit rate (bit/s)

N: Baud rate generator BRR setting ($0 \leq N \leq 255$)

ϕ : Operating frequency (MHz)

n: Baud rate generator input clock ($n = 0$ to 3)

(See the following table for the clock sources and value of n.)

| n | Clock Source | SMR Settings | |
|---|--------------|--------------|------|
| | | CKS1 | CKS2 |
| 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 calculated as follows:

$$\text{Error (\%)} = \left\{ \frac{\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

Table 12.5 indicates the maximum bit rates in the asynchronous mode when the baud rate generator is being used for various frequencies. Tables 12.6 and 12.7 show the maximum rates for external clock input.

Table 12.5 Maximum Bit Rates for Various Frequencies with Baud Rate Generator (Asynchronous Mode)

| ϕ (MHz) | Maximum Bit Rate (Bits/s) | Settings | |
|--------------|---------------------------|----------|---|
| | | n | N |
| 4 | 125000 | 0 | 0 |
| 4.9152 | 153600 | 0 | 0 |
| 6 | 187500 | 0 | 0 |
| 7.3728 | 230400 | 0 | 0 |
| 8 | 250000 | 0 | 0 |
| 9.8304 | 307200 | 0 | 0 |
| 10 | 312500 | 0 | 0 |
| 11.0592 | 345600 | 0 | 0 |
| 12 | 375000 | 0 | 0 |
| 12.288 | 384000 | 0 | 0 |
| 14 | 437500 | 0 | 0 |
| 14.7456 | 460800 | 0 | 0 |
| 16 | 500000 | 0 | 0 |
| 17.2032 | 537600 | 0 | 0 |
| 18 | 562500 | 0 | 0 |
| 18.432 | 576000 | 0 | 0 |
| 19.6608 | 614400 | 0 | 0 |
| 20 | 625000 | 0 | 0 |
| 22 | 687500 | 0 | 0 |
| 22.1184 | 691200 | 0 | 0 |
| 24 | 750000 | 0 | 0 |
| 24.576 | 768000 | 0 | 0 |
| 25.8048 | 806400 | 0 | 0 |
| 26 | 812500 | 0 | 0 |
| 27.0336 | 844800 | 0 | 0 |
| 28 | 875000 | 0 | 0 |

Table 12.6 Maximum Bit Rates during External Clock Input (Asynchronous Mode)

| ϕ (MHz) | External Input Clock (MHz) | Maximum Bit Rate (Bits/s) |
|--------------|----------------------------|---------------------------|
| 4 | 1.0000 | 62500 |
| 4.9152 | 1.2288 | 76800 |
| 6 | 1.5000 | 93750 |
| 7.3728 | 1.8432 | 115200 |
| 8 | 2.0000 | 125000 |
| 9.8304 | 2.4576 | 153600 |
| 10 | 2.5000 | 156250 |
| 11.0592 | 2.7648 | 172800 |
| 12 | 3.0000 | 187500 |
| 12.288 | 3.0720 | 192000 |
| 14 | 3.5000 | 218750 |
| 14.7456 | 3.6864 | 230400 |
| 16 | 4.0000 | 250000 |
| 17.2032 | 4.3008 | 268800 |
| 18 | 4.5000 | 281250 |
| 18.432 | 4.6080 | 288000 |
| 19.6608 | 4.9152 | 307200 |
| 20 | 5.0000 | 312500 |
| 22 | 5.5000 | 343750 |
| 22.1184 | 5.5296 | 345600 |
| 24 | 6.0000 | 375000 |
| 24.576 | 6.1440 | 384000 |
| 25.8048 | 6.4512 | 403200 |
| 26 | 6.5000 | 406250 |
| 27.0336 | 6.7584 | 422400 |
| 28 | 7.0000 | 437500 |

Table 12.7 Maximum Bit Rates during External Clock Input (Clock Synchronous Mode)

| ϕ (MHz) | External Input Clock (MHz) | Maximum Bit Rate (Bits/s) |
|--------------|----------------------------|---------------------------|
| 4 | 0.6667 | 666666.7 |
| 6 | 1.0000 | 1000000.0 |
| 8 | 1.3333 | 1333333.3 |
| 10 | 1.6667 | 1666666.7 |
| 12 | 2.0000 | 2000000.0 |
| 14 | 2.3333 | 2333333.3 |
| 16 | 2.6667 | 2666666.7 |
| 18 | 3.0000 | 3000000.0 |
| 20 | 3.3333 | 3333333.3 |
| 22 | 3.6667 | 3666666.7 |
| 24 | 4.0000 | 4000000.0 |
| 26 | 4.3333 | 4333333.3 |
| 28 | 4.6667 | 4666666.7 |

12.3 Operation

12.3.1 Overview

For serial communication, the SCI has an asynchronous mode in which characters are synchronized individually, and a clock synchronous mode in which communication is synchronized with clock pulses. Asynchronous/clock synchronous mode and the transmission format are selected in the serial mode register (SMR), as shown in table 12.8. The SCI clock source is selected by the C/\bar{A} bit in the serial mode register (SMR) and the CKE1 and CKE0 bits in the serial control register (SCR), as shown in table 12.9.

Asynchronous Mode:

- Data length is selectable: seven or eight bits.
- Parity and multiprocessor bits are selectable, as well as the stop bit length (one or two bits). These selections determine the transmit/receive format and character length.
- In receiving, it is possible to detect framing errors (FER), parity errors (PER), overrun errors (ORER), and the break state.
- An internal or external clock can be selected as the SCI clock source.
 - When an internal clock is selected, the SCI operates using the on-chip baud rate generator clock, and can output a clock with a frequency matching the bit rate.
 - When an external clock is selected, the external clock input must have a frequency 16 times the bit rate. (The on-chip baud rate generator is not used.)

Clock Synchronous Mode:

- The communication format has a fixed 8-bit data length.
- In receiving, it is possible to detect overrun errors (ORER).
- An internal or external clock can be selected as the SCI clock source.
 - When an internal clock is selected, the SCI operates using the on-chip baud rate generator clock, and outputs a synchronous clock signal to external devices.
 - When an external clock is selected, the SCI operates on the input synchronous clock. The on-chip baud rate generator is not used.

Table 12.8 SMR Settings and SCI Communication Formats

| Mode | SMR Settings | | | | | SCI Communication Format | | | |
|--|--------------|--------------|-------------|-------------|---------------|--------------------------|---------------|-------------------------|--------------------|
| | Bit 7 C/A | Bit 6 CHR | Bit 5 PE | Bit 2 MP | Bit 3 STOP | Data Length | Parity Bit | Multipro- cessor Bit | Stop Bit Length |
| Asynchronous | 0 | 0 | 0 | 0 | 0 | 8-bit | Not set | Not set | 1 bit |
| | | | | | 1 | | | | 2 bits |
| | | | 1 | 0 | 0 | 7-bit | Not set | 1 bit | |
| | | | | | 1 | | | 2 bits | |
| | 1 | 0 | 1 | 0 | 0 | 7-bit | Not set | Not set | 1 bit |
| | | | | | 1 | | | | 2 bits |
| | | | 1 | 0 | 0 | 7-bit | Set | 1 bit | |
| | | | | | 1 | | | 2 bits | |
| Asynchronous (multiprocessor format) | 0 | * | 1 | 0 | 8-bit | Not set | Set | 1 bit | |
| | | | | 1 | | | | 2 bits | |
| | | | 1 | * | 0 | 7-bit | Not set | 1 bit | |
| | | | | | 1 | | | 2 bits | |
| Clock synchronous | 1 | * | * | * | 8-bit | Not set | Not set | None | |

Note: Asterisks (*) in the table indicate don't-care bits.

Table 12.9 SMR and SCR Settings and SCI Clock Source Selection

| Mode | SMR | SCR Settings | | SCI Transmit/Receive Clock | | | |
|------------------------|--------------|---------------|---------------|----------------------------|--|----------|---|
| | Bit 7 C/A | Bit 1 CKE1 | Bit 0 CKE0 | Clock Source | SCK Pin Function* | | |
| Asynchronous | 0 | 0 | 0 | Internal | SCI does not use the SCK pin | | |
| | | | 1 | | Outputs a clock with frequency matching the bit rate | | |
| | | | 1 | 0 | 0 | External | Inputs a clock with frequency 16 times the bit rate |
| | | | | | 1 | | |
| Clock synch- ronous | 1 | 0 | 0 | Internal | Outputs the synchronous clock | | |
| | | | 1 | | | | |
| | | | 1 | 0 | 0 | External | Inputs the synchronous clock |
| | | | | | 1 | | |

Note: Select the function in combination with the pin function controller (PFC).

12.3.2 Operation in Asynchronous Mode

In the asynchronous mode, each transmitted or received character begins with a start bit and ends with a stop bit. Serial communication is synchronized one character at a time.

The transmitting and receiving sections of the SCI are independent, so full duplex communication is possible. The transmitter and receiver are both double buffered, so data can be written and read while transmitting and receiving are in progress, enabling continuous transmitting and receiving.

Figure 12.2 shows the general format of asynchronous serial communication. In asynchronous serial communication, the communication line is normally held in the marking (high) state. The SCI monitors the line and starts serial communication when the line goes to the space (low) state, indicating a start bit. One serial character consists of a start bit (low), data (LSB first), parity bit (high or low), and stop bit (high), in that order.

When receiving in the asynchronous mode, the SCI synchronizes on the falling edge of the start bit. The SCI samples each data bit on the eighth pulse of a clock with a frequency 16 times the bit rate. Receive data is latched at the center of each bit.

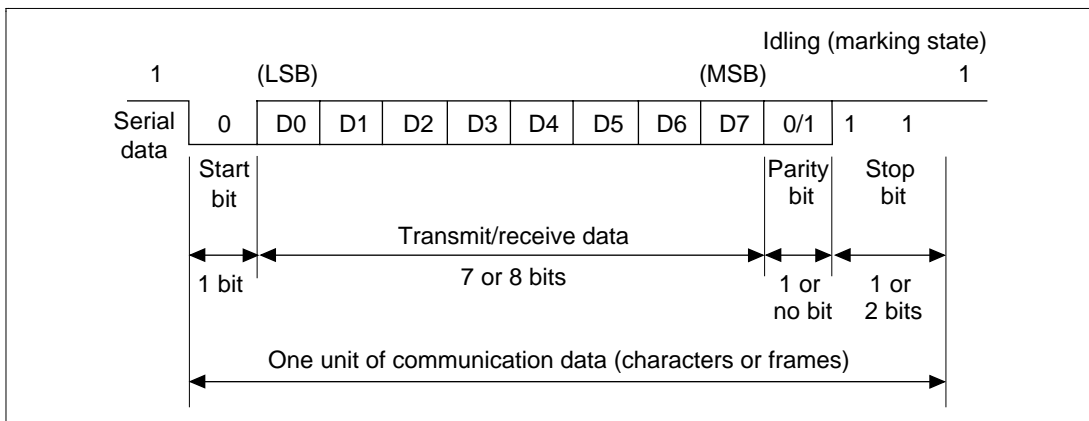


Figure 12.2 Data Format in Asynchronous Communication (Example: 8-bit Data with Parity and Two Stop Bits)

Transmit/Receive Formats: Table 12.10 shows the 12 communication formats that can be selected in the asynchronous mode. The format is selected by settings in the serial mode register (SMR).

Table 12.10 Serial Communication Formats (Asynchronous Mode)

| SMR Bits | | | | Serial Transmit/Receive Format and Frame Length | | | | | | | | | | | | |
|----------|----|----|------|---|------------|---|---|---|---|---|------|------|------|------|----|--|
| CHR | PE | MP | STOP | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 0 | 0 | 0 | 0 | START | 8-Bit data | | | | | | | STOP | | | | |
| 0 | 0 | 0 | 1 | START | 8-Bit data | | | | | | | STOP | STOP | | | |
| 0 | 1 | 0 | 0 | START | 8-Bit data | | | | | | | P | STOP | | | |
| 0 | 1 | 0 | 1 | START | 8-Bit data | | | | | | | P | STOP | STOP | | |
| 1 | 0 | 0 | 0 | START | 7-Bit data | | | | | | STOP | | | | | |
| 1 | 0 | 0 | 1 | START | 7-Bit data | | | | | | STOP | STOP | | | | |
| 1 | 1 | 0 | 0 | START | 7-Bit data | | | | | | P | STOP | | | | |
| 1 | 1 | 0 | 1 | START | 7-Bit data | | | | | | P | STOP | STOP | | | |
| 0 | — | 1 | 0 | START | 8-Bit data | | | | | | | MPB | STOP | | | |
| 0 | — | 1 | 1 | START | 8-Bit data | | | | | | | MPB | STOP | STOP | | |
| 1 | — | 1 | 0 | START | 7-Bit data | | | | | | MPB | STOP | | | | |
| 1 | — | 1 | 1 | START | 7-Bit data | | | | | | MPB | STOP | STOP | | | |

—: Don't care bits.

Note: START: Start bit

STOP: Stop bit

P: Parity bit

MPB: Multiprocessor bit

Clock: An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected as the SCI transmit/receive clock. The clock source is selected by the $\overline{C/A}$ bit in the serial mode register (SMR) and bits CKE1 and CKE0 in the serial control register (SCR) (table 12.10).

When an external clock is input at the SCK pin, it must have a frequency equal to 16 times the desired bit rate.

When the SCI operates on an internal clock, it can output a clock signal at the SCK pin. The frequency of this output clock is equal to the bit rate. The phase is aligned as in figure 12.3 so that the rising edge of the clock occurs at the center of each transmit data bit.

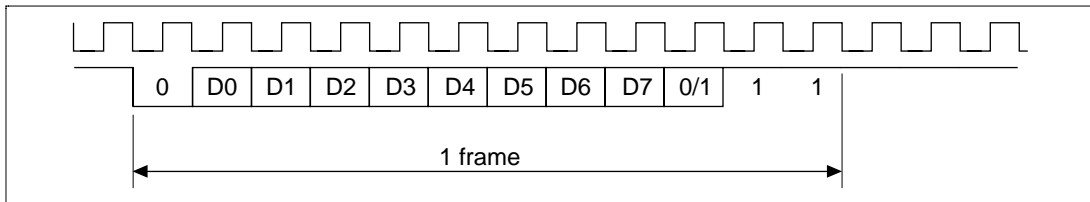


Figure 12.3 Output Clock and Communication Data Phase Relationship (Asynchronous Mode)

SCI Initialization (Asynchronous Mode): Before transmitting or receiving, clear the TE and RE bits to 0 in the serial control register (SCR), then initialize the SCI as follows.

When changing the operation mode or communication format, always clear the TE and RE bits to 0 before following the procedure given below. Clearing TE to 0 sets TDRE to 1 and initializes the transmit shift register (TSR). Clearing RE to 0, however, does not initialize the RDRF, PER, FER, and ORER flags and receive data register (RDR), which retain their previous contents.

When an external clock is used, the clock should not be stopped during initialization or subsequent operation. SCI operation becomes unreliable if the clock is stopped.

Figure 12.4 is a sample flowchart for initializing the SCI. The procedure is as follows (the steps correspond to the numbers in the flowchart):

1. Select the clock source in the serial control register (SCR). Leave RIE, TIE, TEIE, MPIE, TE and RE cleared to 0. If clock output is selected in asynchronous mode, clock output starts immediately after the setting is made to SCR.
2. Select the communication format in the serial mode register (SMR).
3. Write the value corresponding to the bit rate in the bit rate register (BRR) unless an external clock is used.
4. Wait for at least the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCR) to 1. Also set RIE, TIE, TEIE and MPIE as necessary. Setting TE or RE enables the SCI to use the TxD or RxD pin. The initial states are the marking transmit state, and the idle receive state (waiting for a start bit).

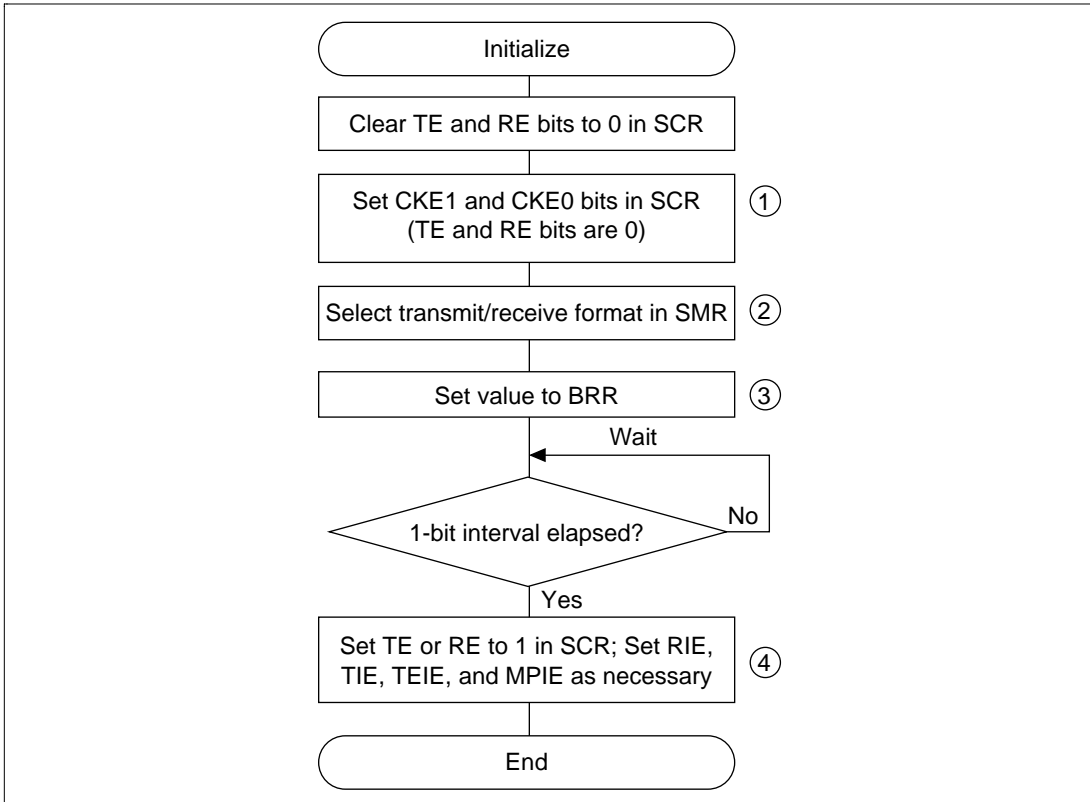


Figure 12.4 Sample Flowchart for SCI Initialization

Transmitting Serial Data (Asynchronous Mode): Figure 12.5 shows a sample flowchart for transmitting serial data. The procedure is as follows (the steps correspond to the numbers in the flowchart):

1. SCI initialization: Set the TxD pin using the PFC.
2. SCI status check and transmit data write: Read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR) and clear TDRE to 0.
3. Continue transmitting serial data: Read the TDRE bit to check whether it is safe to write (if it reads 1); if so, write data in TDR, then clear TDRE to 0. When the DMAC or the DTC is started by a transmit-data-empty interrupt request (TxI) in order to write data in TDR, the TDRE bit is checked and cleared automatically.
4. To output a break at the end of serial transmission, first clear the port data register (DR) to 0, then clear the TE to 0 in SCR and use the PFC to establish the TxD pin as an output port.

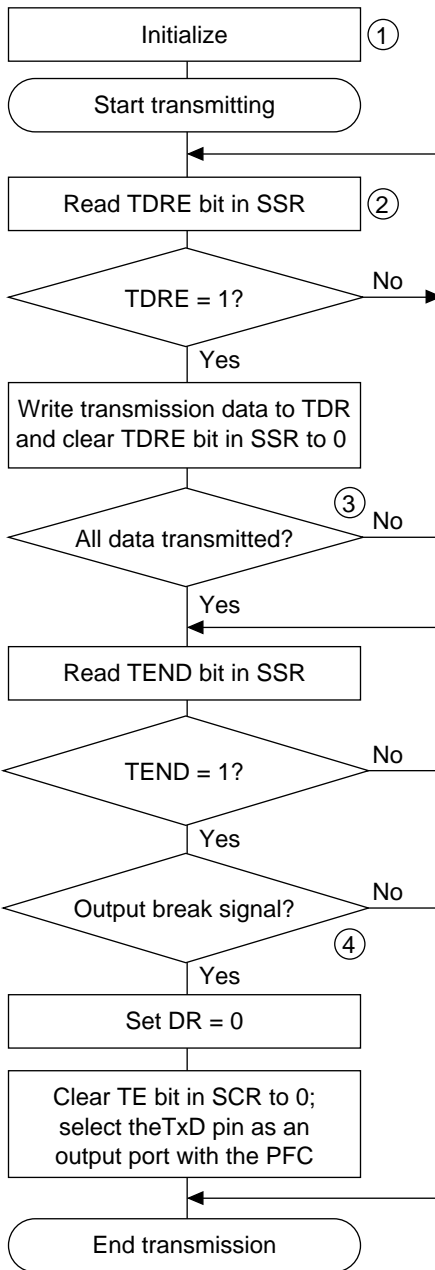


Figure 12.5 Sample Flowchart for Transmitting Serial Data

In transmitting serial data, the SCI operates as follows:

1. The SCI monitors the TDRE bit in the SSR. When TDRE is cleared to 0, the SCI recognizes that the transmit data register (TDR) contains new data, and loads this data from the TDR into the transmit shift register (TSR).
2. After loading the data from the TDR into the TSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) is set to 1 in the SCR, the SCI requests a transmit-data-empty interrupt (TxI) at this time.

Serial transmit data is transmitted in the following order from the TxD pin:

- a. Start bit: one 0 bit is output.
 - b. Transmit data: seven or eight bits of data are output, LSB first.
 - c. Parity bit or multiprocessor bit: one parity bit (even or odd parity) or one multiprocessor bit is output. Formats in which neither a parity bit nor a multiprocessor bit is output can also be selected.
 - d. Stop bit: one or two 1 bits (stop bits) are output.
 - e. Marking: output of 1 bits continues until the start bit of the next transmit data.
3. The SCI checks the TDRE bit when it outputs the stop bit. If TDRE is 0, the SCI loads new data from the TDR into the TSR, outputs the stop bit, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit to 1 in the SSR, outputs the stop bit, then continues output of 1 bits (marking). If the transmit-end interrupt enable bit (TEIE) in the SCR is set to 1, a transmit-end interrupt (TEI) is requested.

Figure 12.6 shows an example of SCI transmit operation in the asynchronous mode.

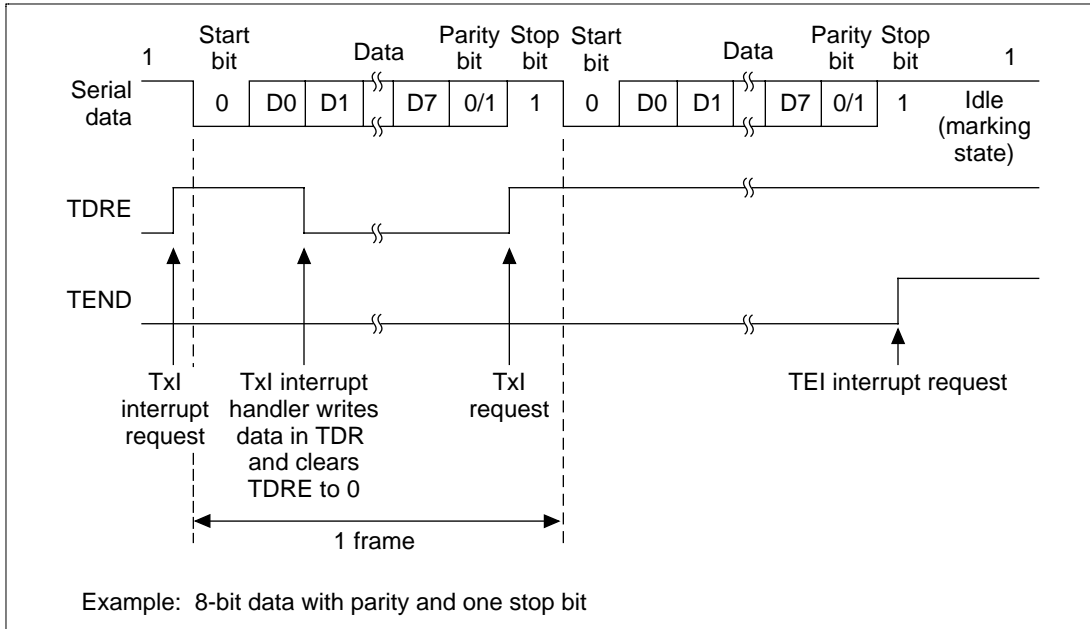


Figure 12.6 SCI Transmit Operation in Asynchronous Mode

Receiving Serial Data (Asynchronous Mode): Figures 12.7 and 12.8 show a sample flowchart for receiving serial data. The procedure is as follows (the steps correspond to the numbers in the flowchart).

1. SCI initialization: Set the RxD pin using the PFC.
2. Receive error handling and break detection: If a receive error occurs, read the ORER, PER, and FER bits of the SSR to identify the error. After executing the necessary error handling, clear ORER, PER, and FER all to 0. Receiving cannot resume if ORER, PER or FER remain set to 1. When a framing error occurs, the RxD pin can be read to detect the break state.
3. SCI status check and receive-data read: Read the serial status register (SSR), check that RDRF is set to 1, then read receive data from the receive data register (RDR) and clear RDRF to 0. The RxI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
4. Continue receiving serial data: Read the RDR and RDRF bit and clear RDRF to 0 before the stop bit of the current frame is received. If the DMAC is started by a receive-data-full interrupt (RxI) to read RDR, the RDRF bit is cleared automatically so this step is unnecessary.

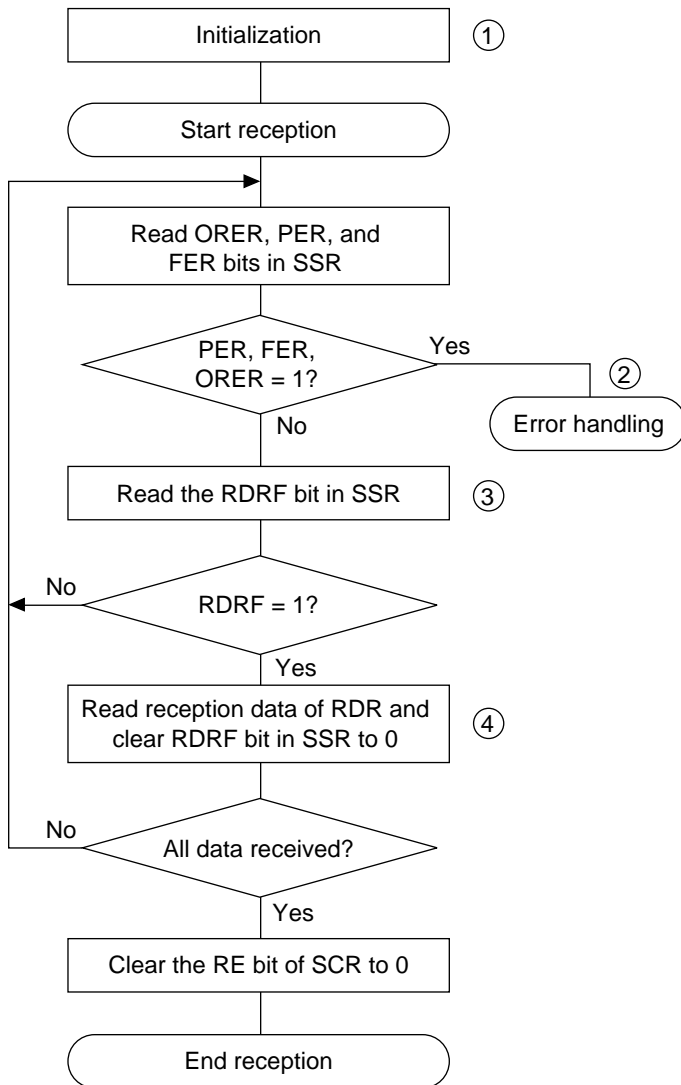


Figure 12.7 Sample Flowchart for Receiving Serial Data (1)

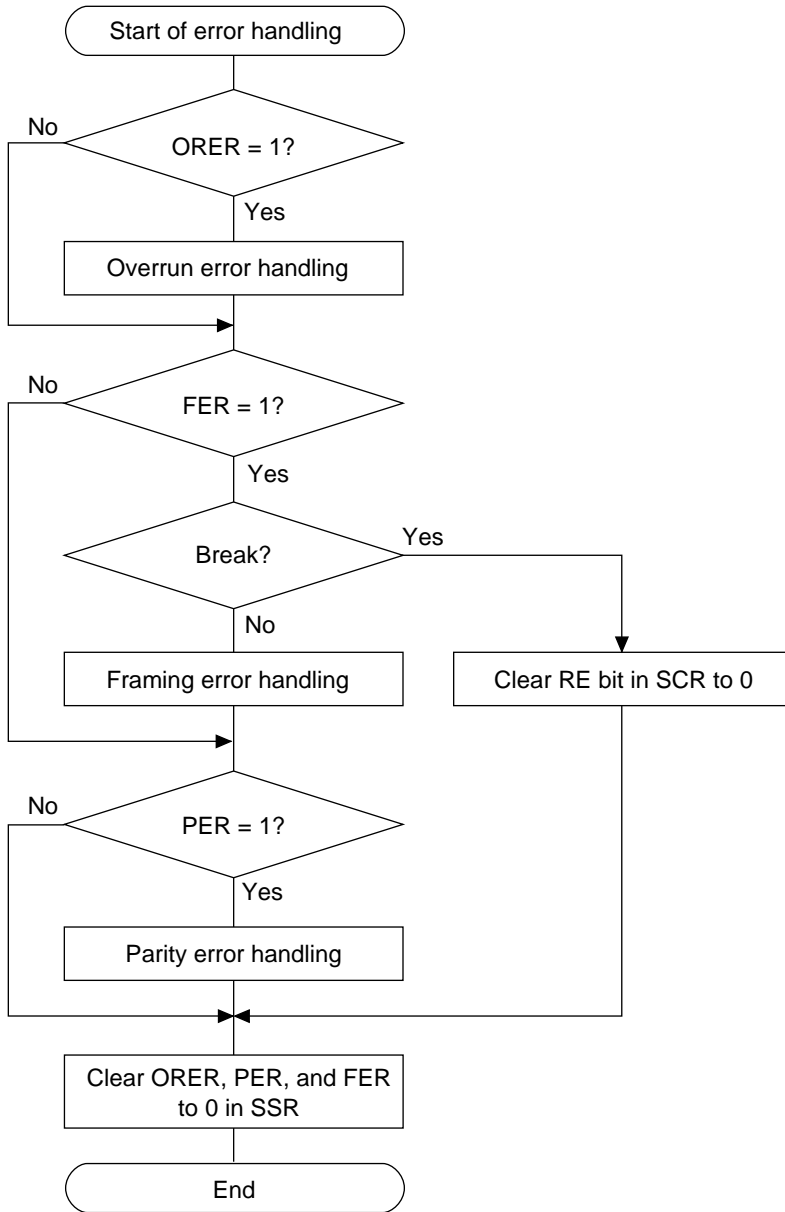


Figure 12.8 Sample Flowchart for Receiving Serial Data (2)

In receiving, the SCI operates as follows:

1. The SCI monitors the communication line. When it detects a start bit (0), the SCI synchronizes internally and starts receiving.
2. Receive data is shifted into the RSR in order from the LSB to the MSB.
3. The parity bit and stop bit are received. After receiving these bits, the SCI makes the following checks:
 - a. Parity check. The number of 1s in the receive data must match the even or odd parity setting of the O/E bit in the SMR.
 - b. Stop bit check. The stop bit value must be 1. If there are two stop bits, only the first stop bit is checked.
 - c. Status check. RDRF must be 0 so that receive data can be loaded from the RSR into the RDR.

If the data passes these checks, the SCI sets RDRF to 1 and stores the received data in the RDR. If one of the checks fails (receive error), the SCI operates as indicated in table 12.11.

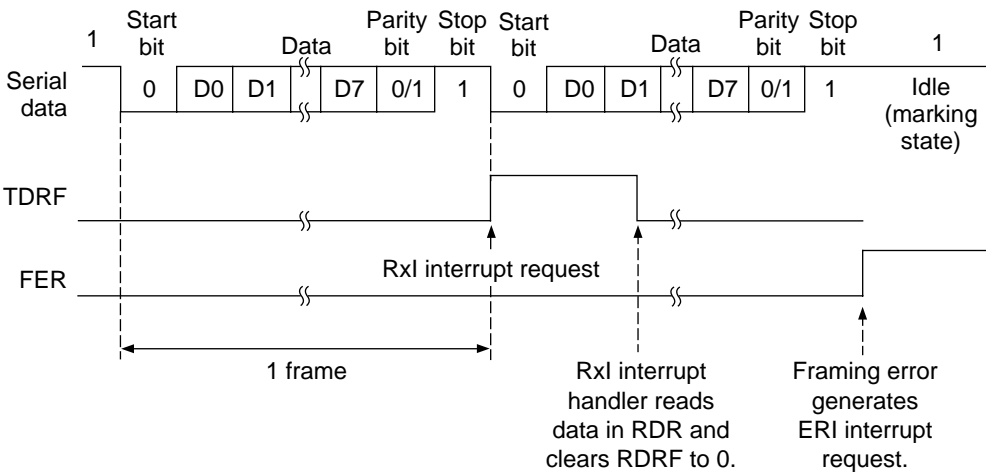
Note: When a receive error occurs, further receiving is disabled. While receiving, the RDRF bit is not set to 1, so be sure to clear the error flags.

4. After setting RDRF to 1, if the receive-data-full interrupt enable bit (RIE) is set to 1 in the SCR, the SCI requests a receive-data-full interrupt (RxI). If one of the error flags (ORER, PER, or FER) is set to 1 and the receive-data-full interrupt enable bit (RIE) in the SCR is also set to 1, the SCI requests a receive-error interrupt (ERI).

Figure 12.9 shows an example of SCI receive operation in the asynchronous mode.

Table 12.11 Receive Error Conditions and SCI Operation

| Receive Error | Abbreviation | Condition | Data Transfer |
|----------------------|---------------------|--|---|
| Overrun error | ORER | Receiving of next data ends while RDRF is still set to 1 in SSR | Receive data not loaded from RSR into RDR |
| Framing error | FER | Stop bit is 0 | Receive data loaded from RSR into RDR |
| Parity error | PER | Parity of receive data differs from even/odd parity setting in SMR | Receive data loaded from RSR into RDR |



Example: 8-bit data with parity and one stop bit.

Figure 12.9 SCI Receive Operation

12.3.3 Multiprocessor Communication

The multiprocessor communication function enables several processors to share a single serial communication line for sending and receiving data. The processors communicate in the asynchronous mode using a format with an additional multiprocessor bit (multiprocessor format).

In multiprocessor communication, each receiving processor is addressed by a unique ID. A serial communication cycle consists of an ID-sending cycle that identifies the receiving processor, and a data-sending cycle. The multiprocessor bit distinguishes ID-sending cycles from data-sending cycles. The transmitting processor starts by sending the ID of the receiving processor with which it wants to communicate as data with the multiprocessor bit set to 1. Next the transmitting processor sends transmit data with the multiprocessor bit cleared to 0.

Receiving processors skip incoming data until they receive data with the multiprocessor bit set to 1. When they receive data with the multiprocessor bit set to 1, receiving processors compare the data with their IDs. The receiving processor with a matching ID continues to receive further incoming data. Processors with IDs not matching the received data skip further incoming data until they again receive data with the multiprocessor bit set to 1. Multiple processors can send and receive data in this way.

Figure 12.10 shows the example of communication among processors using the multiprocessor format.

Communication Formats: Four formats are available. Parity-bit settings are ignored when the multiprocessor format is selected. For details see table 12.8.

Clock: See the description in the asynchronous mode section.

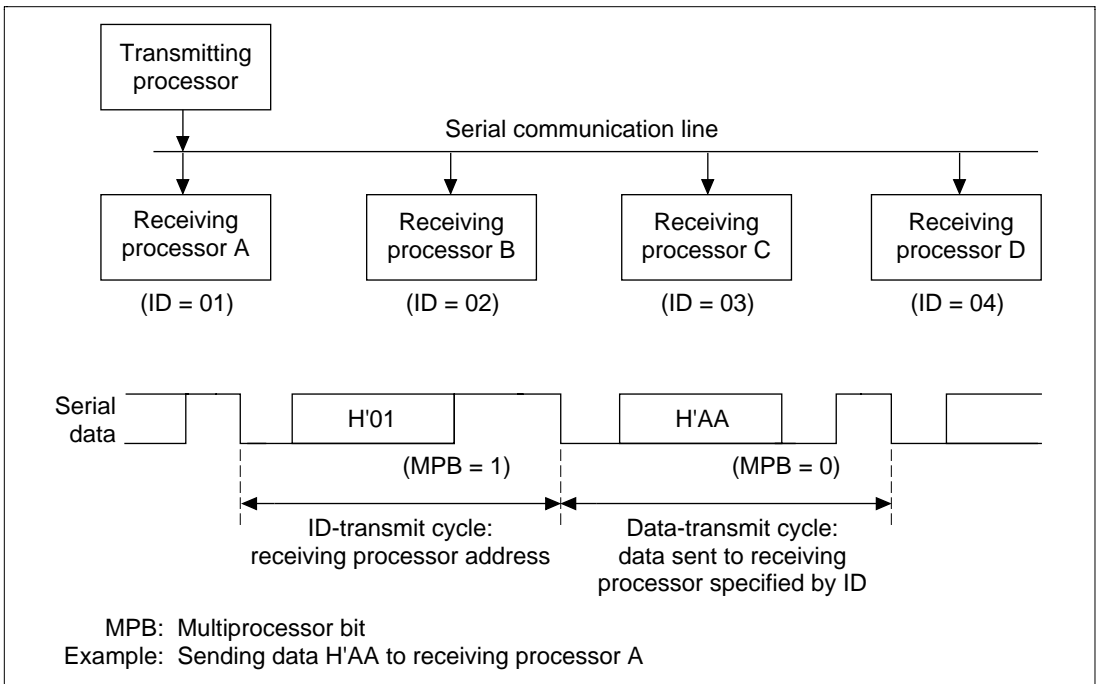


Figure 12.10 Communication Among Processors Using Multiprocessor Format

Transmitting Multiprocessor Serial Data: Figure 12.11 shows a sample flowchart for transmitting multiprocessor serial data. The procedure is as follows (the steps correspond to the numbers in the flowchart):

1. SCI initialization: Set the TxD pin using the PFC.
2. SCI status check and transmit data write: Read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR). Also set MPBT (multiprocessor bit transfer) to 0 or 1 in SSR. Finally, clear TDRE to 0.
3. Continue transmitting serial data: Read the TDRE bit to check whether it is safe to write (if it reads 1); if so, write data in TDR, then clear TDRE to 0. When the DMAC is started by a transmit-data-empty interrupt request (TxI) to write data in TDR, the TDRE bit is checked and cleared automatically.
4. Output a break at the end of serial transmission: Set the data register (DR) of the port to 0, then clear TE to 0 in SCR and set the TxD pin function as output port with the PFC.

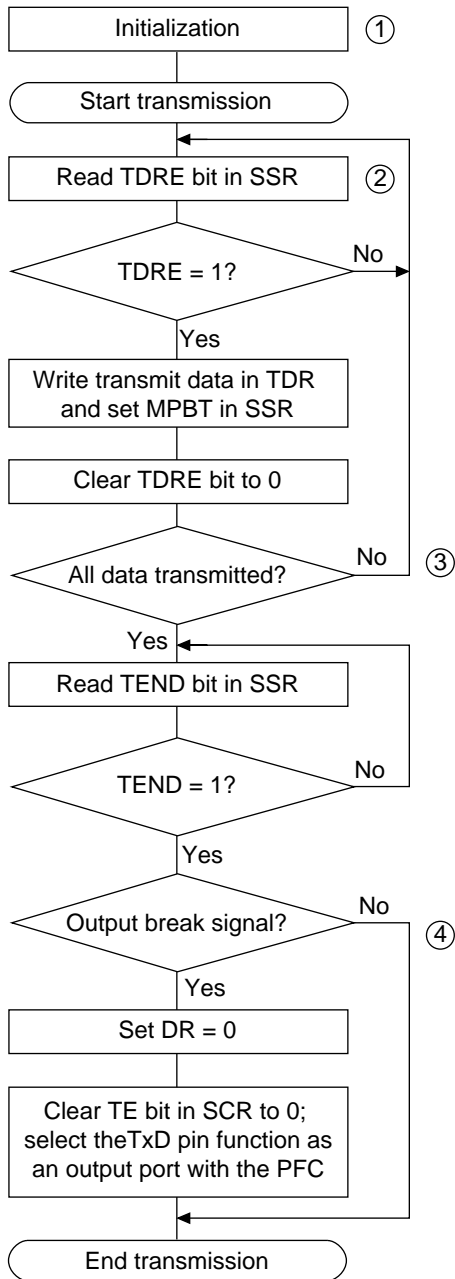


Figure 12.11 Sample Flowchart for Transmitting Multiprocessor Serial Data

In transmitting serial data, the SCI operates as follows:

1. The SCI monitors the TDRE bit in the SSR. When TDRE is cleared to 0 the SCI recognizes that the transmit data register (TDR) contains new data, and loads this data from the TDR into the transmit shift register (TSR).
2. After loading the data from the TDR into the TSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) in the SCR is set to 1, the SCI requests a transmit-data-empty interrupt (TxI) at this time.

Serial transmit data is transmitted in the following order from the TxD pin:

- a. Start bit: one 0 bit is output.
 - b. Transmit data: seven or eight bits are output, LSB first.
 - c. Multiprocessor bit: one multiprocessor bit (MPBT value) is output.
 - d. Stop bit: one or two 1 bits (stop bits) are output.
 - e. Marking: output of 1 bits continues until the start bit of the next transmit data.
3. The SCI checks the TDRE bit when it outputs the stop bit. If TDRE is 0, the SCI loads data from the TDR into the TSR, outputs the stop bit, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit in the SSR to 1, outputs the stop bit, then continues output of 1 bits in the marking state. If the transmit-end interrupt enable bit (TEIE) in the SCR is set to 1, a transmit-end interrupt (TEI) is requested at this time.

Figure 12.12 shows an example of SCI receive operation in the multiprocessor format.

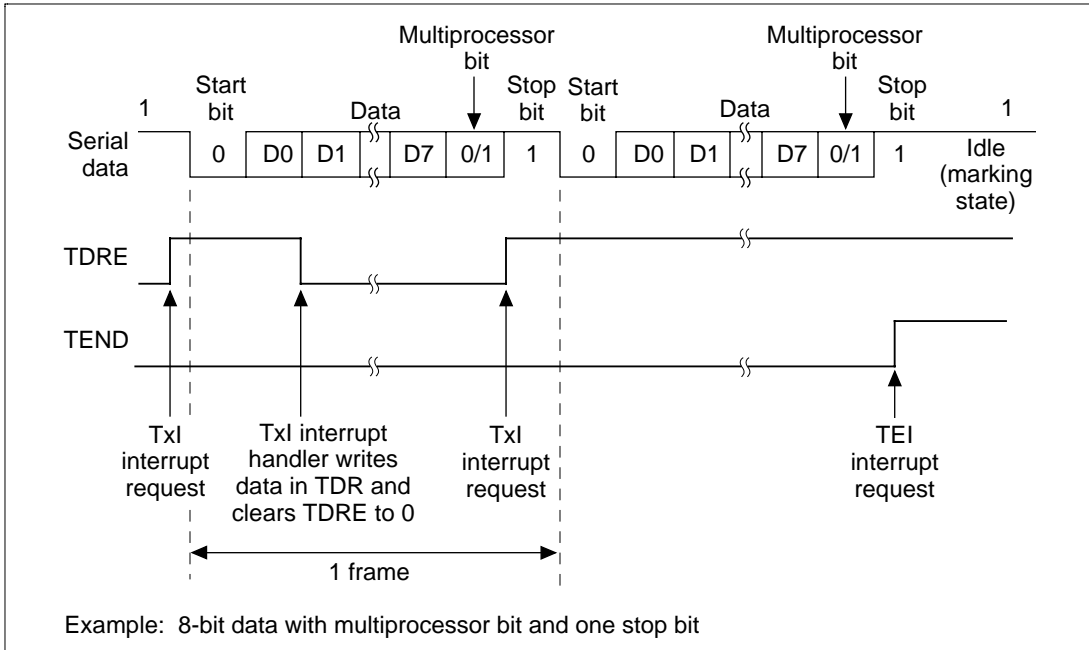


Figure 12.12 SCI Multiprocessor Transmit Operation

Receiving Multiprocessor Serial Data: Figures 12.13 and 12.14 show a sample flowchart for receiving multiprocessor serial data. The procedure for receiving multiprocessor serial data is listed below.

1. SCI initialization: Set the RxD pin using the PFC.
2. ID receive cycle: Set the MPIE bit in the serial control register (SCR) to 1.
3. SCI status check and compare to ID reception: Read the serial status register (SSR), check that RDRF is set to 1, then read data from the receive data register (RDR) and compare with the processor's own ID. If the ID does not match the receive data, set MPIE to 1 again and clear RDRF to 0. If the ID matches the receive data, clear RDRF to 0.
4. Receive error handling and break detection: If a receive error occurs, read the ORER and FER bits in SSR to identify the error. After executing the necessary error processing, clear both ORER and FER to 0. Receiving cannot resume if ORER or FER remain set to 1. When a framing error occurs, the RxD pin can be read to detect the break state.
5. SCI status check and data receiving: Read SSR, check that RDRF is set to 1, then read data from the receive data register (RDR).

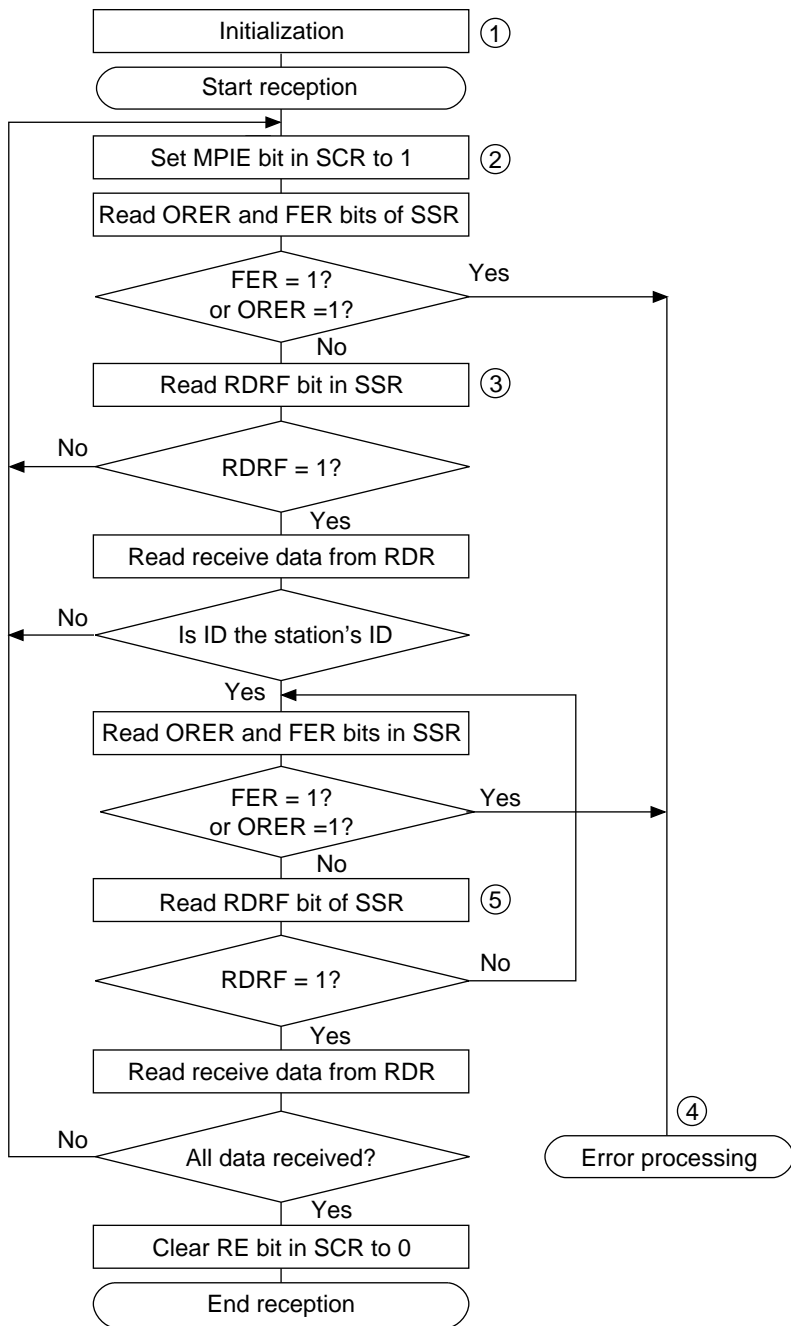


Figure 12.13 Sample Flowchart for Receiving Multiprocessor Serial Data (1)

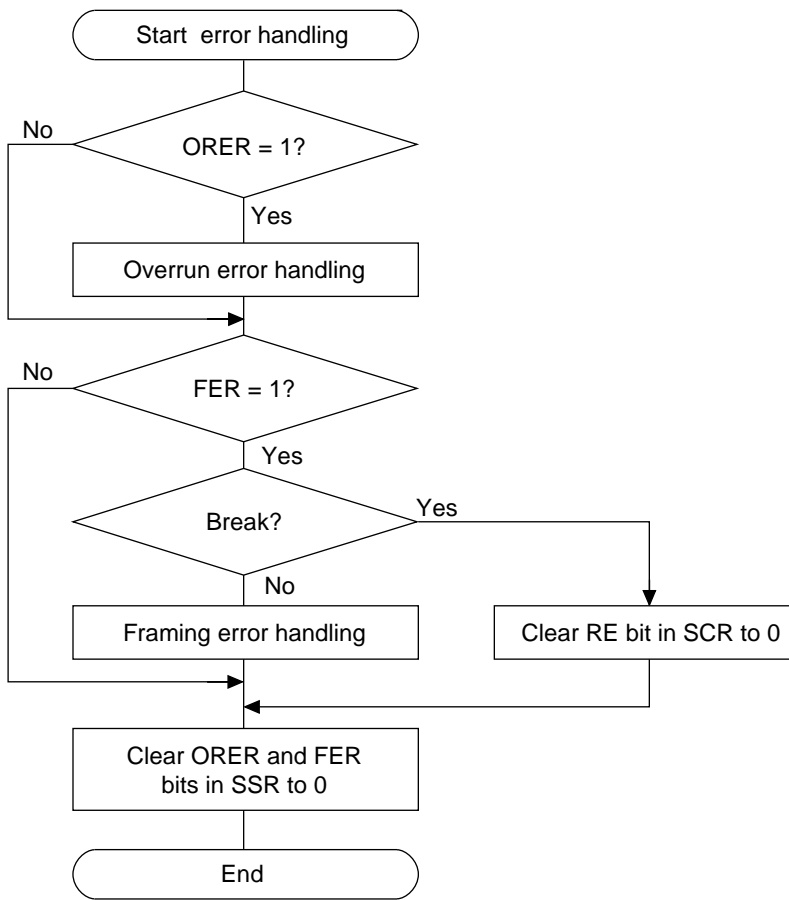


Figure 12.14 Sample Flowchart for Receiving Multiprocessor Serial Data (2)

Figures 12.15 and 12.16 show examples of SCI receive operation using a multiprocessor format.

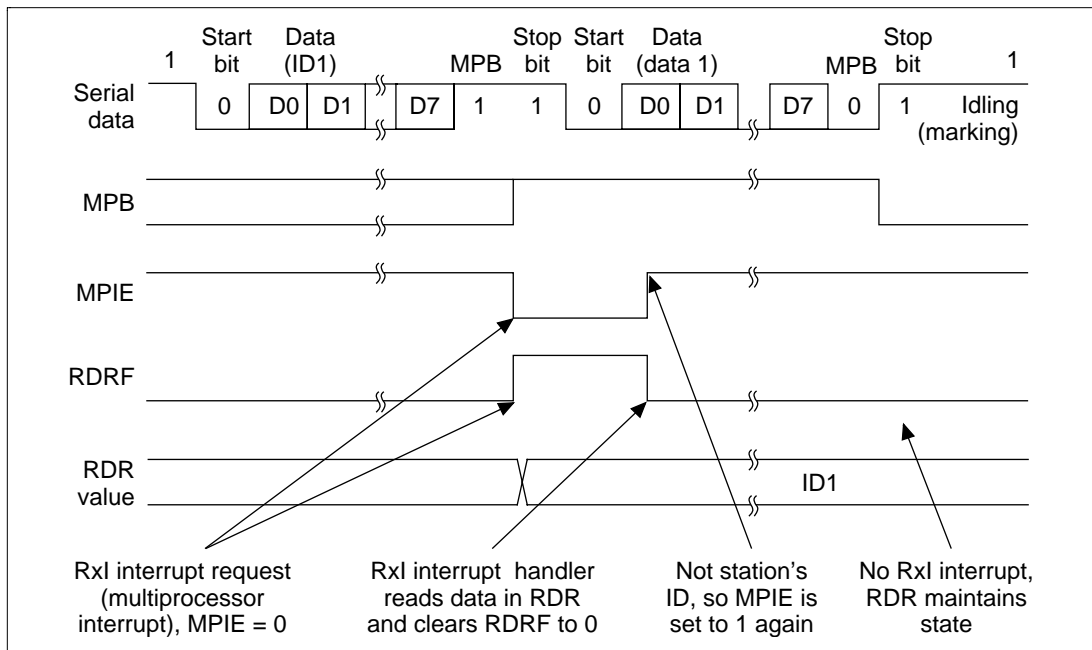
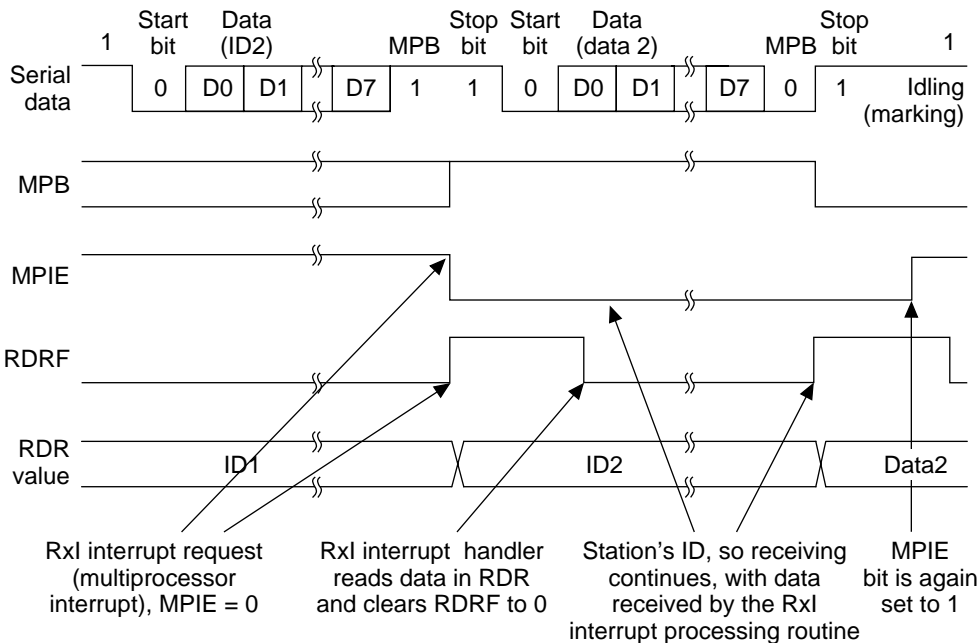


Figure 12.15 SCI Receive Operation (ID Does Not Match)



Example: Own ID matches data, 8-bit data with multiprocessor bit and one stop bit

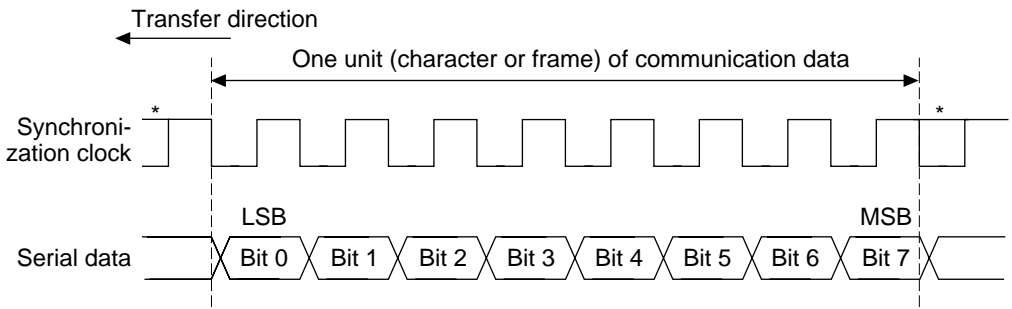
Figure 12.16 Example of SCI Receive Operation (ID Matches)

12.3.4 Clock Synchronous Operation

In the clock synchronous mode, the SCI transmits and receives data in synchronization with clock pulses. This mode is suitable for high-speed serial communication.

The SCI transmitter and receiver are independent, so full duplex communication is possible while sharing the same clock. The transmitter and receiver are also double buffered, so continuous transmitting or receiving is possible by reading or writing data while transmitting or receiving is in progress.

Figure 12.17 shows the general format in clock synchronous serial communication.



Note: High except in continuous transmitting or receiving.

Figure 12.17 Data Format in Clock Synchronous Communication

In clock synchronous serial communication, each data bit is output on the communication line from one falling edge of the serial clock to the next. Data are guaranteed valid at the rising edge of the serial clock. In each character, the serial data bits are transmitted in order from the LSB (first) to the MSB (last). After output of the MSB, the communication line remains in the state of the MSB. In the clock synchronous mode, the SCI transmits or receives data by synchronizing with the falling edge of the synchronization clock.

Communication Format: The data length is fixed at eight bits. No parity bit or multiprocessor bit can be added.

Clock: An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected as the SCI transmit/receive clock. The clock source is selected by the C/\bar{A} bit in the serial mode register (SMR) and bits CKE1 and CKE0 in the serial control register (SCR). See table 12.9.

When the SCI operates on an internal clock, it outputs the clock signal at the SCK pin. Eight clock pulses are output per transmitted or received character. When the SCI is not transmitting or receiving, the clock signal remains in the high state.

Note: An overrun error occurs only during the receive operation, and the sync clock is output until the RE bit is cleared to 0. When you want to perform a receive operation in one-character units, select external clock for the clock source.

SCI Initialization (Clock Synchronous Mode): Before transmitting or receiving, software must clear the TE and RE bits to 0 in the serial control register (SCR), then initialize the SCI as follows.

When changing the mode or communication format, always clear the TE and RE bits to 0 before following the procedure given below. Clearing TE to 0 sets TDRE to 1 and initializes the transmit shift register (TSR). Clearing RE to 0, however, does not initialize the RDRF, PER, FER, and ORER flags and receive data register (RDR), which retain their previous contents.

Figure 12.18 is a sample flowchart for initializing the SCI.

1. Select the clock source in the serial control register (SCR). Leave RIE, TIE, TEIE, MPIE, TE, and RE cleared to 0.
2. Select the communication format in the serial mode register (SMR).
3. Write the value corresponding to the bit rate in the bit rate register (BRR) unless an external clock is used.
4. Wait for at least the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCR) to 1. Also set RIE, TIE, TEIE, and MPIE. The Tx/D, Rx/D pins becomes usable in response to the PFC corresponding bits and the TE, RE bit settings.

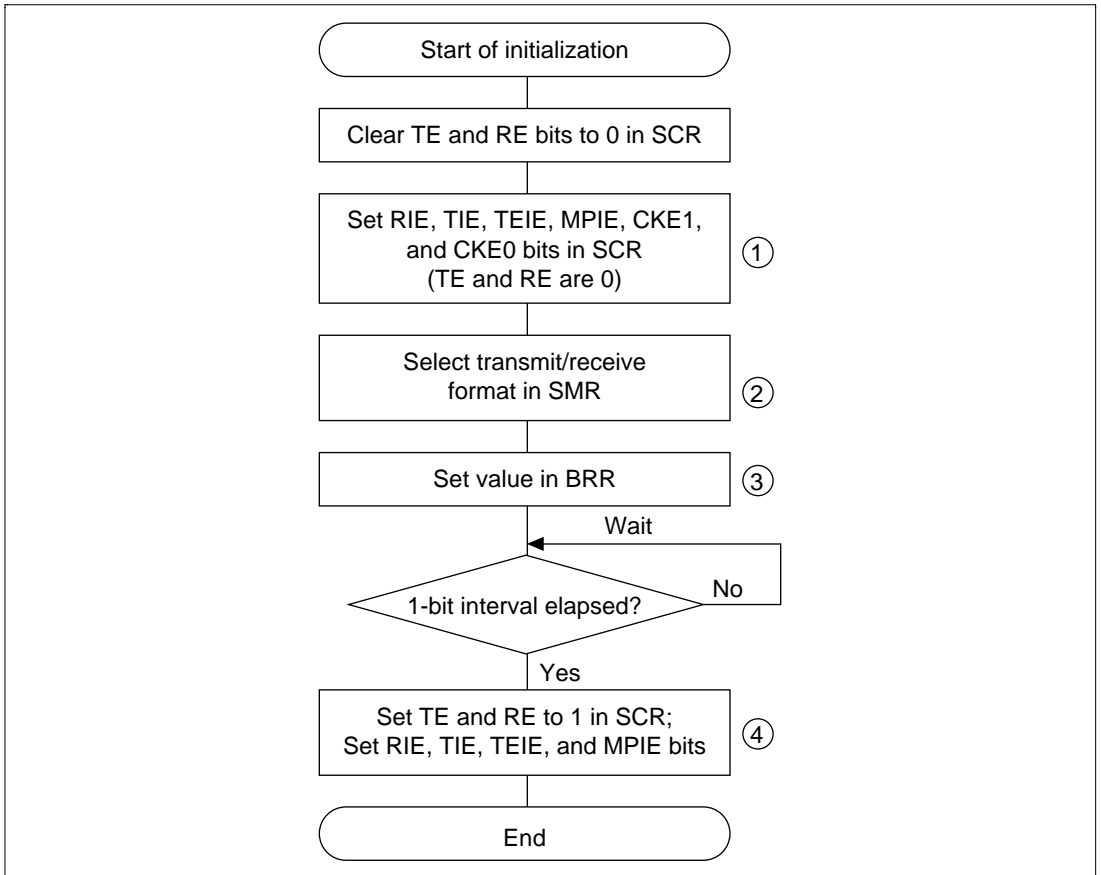


Figure 12.18 Sample Flowchart for SCI Initialization

Transmitting Serial Data (Synchronous Mode): Figure 12.19 shows a sample flowchart for transmitting serial data and indicates the procedure to follow.

1. SCI initialization: Set the TxD pin function with the PFC.
2. SCI status check and transmit data write: Read SSR, check that the TDRE flag is 1, then write transmit data in TDR and clear the TDRE flag to 0.
3. To continue transmitting serial data: After checking that the TDRE flag is 1, indicating that data can be written, write data in TDR, then clear the TDRE flag to 0. When the DMAC or DTC is activated by a transmit-data-empty interrupt request (TxI) to write data in TDR, the TDRE flag is checked and cleared automatically.

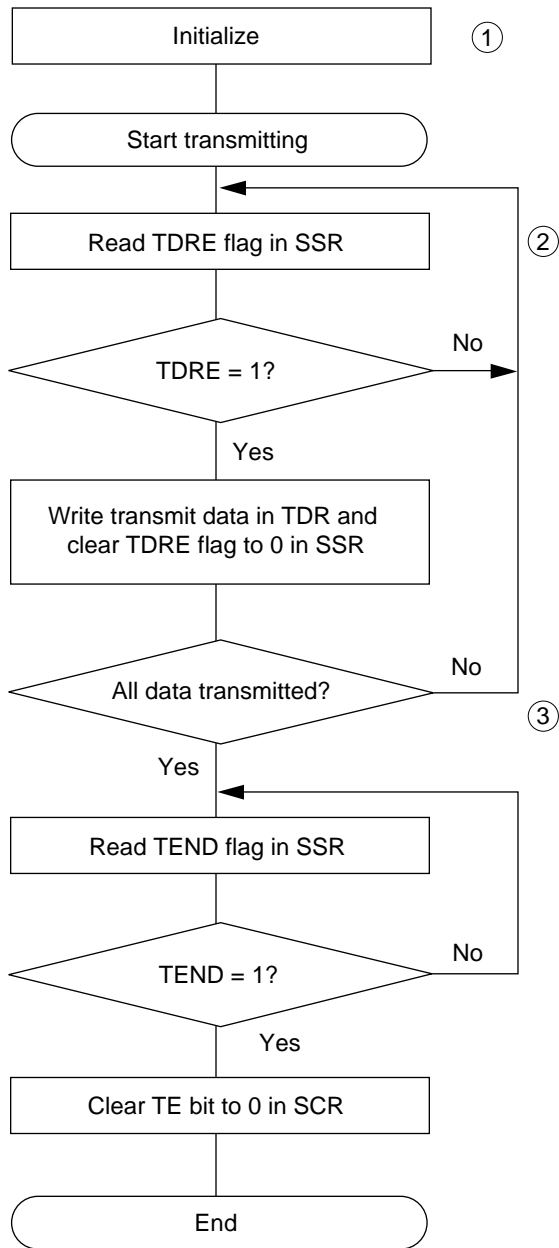


Figure 12.19 Sample Flowchart for Serial Transmitting

Figure 12.20 shows an example of SCI transmit operation.

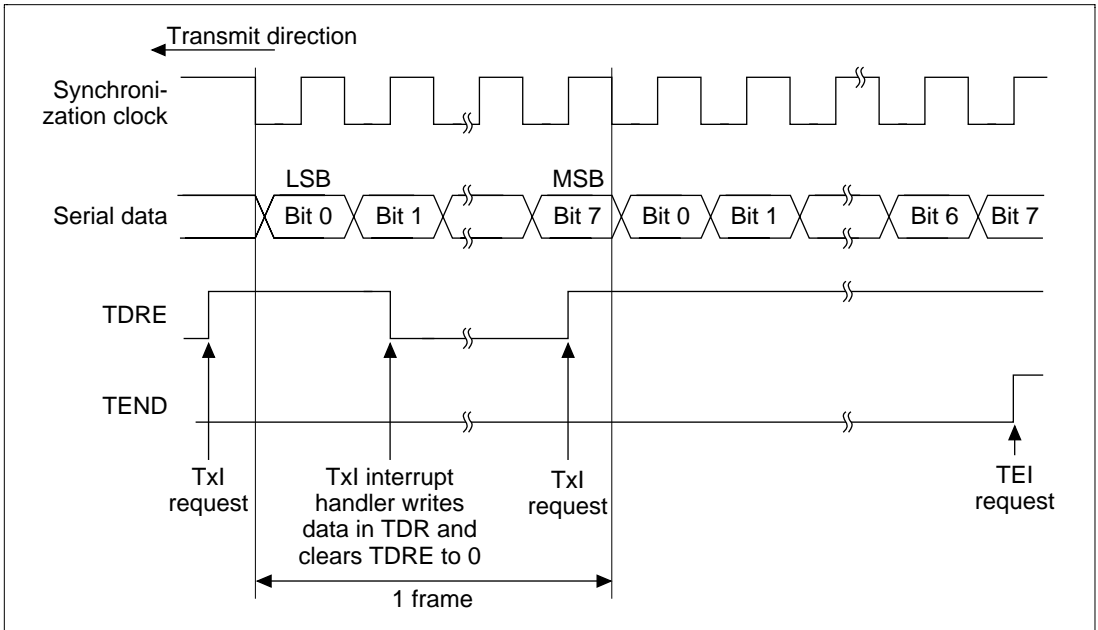


Figure 12.20 Example of SCI Transmit Operation

SCI serial transmission operates as follows.

1. The SCI monitors the TDRE bit in the SSR. When TDRE is cleared to 0 the SCI recognizes that the transmit data register (TDR) contains new data and loads this data from the TDR into the transmit shift register (TSR).
2. After loading the data from the TDR into the TSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) in the SCR is set to 1, the SCI requests a transmit-data-empty interrupt (TxI) at this time.
If clock output mode is selected, the SCI outputs eight synchronous clock pulses. If an external clock source is selected, the SCI outputs data in synchronization with the input clock. Data are output from the TxD pin in order from the LSB (bit 0) to the MSB (bit 7).
3. The SCI checks the TDRE bit when it outputs the MSB (bit 7). If TDRE is 0, the SCI loads data from the TDR into the TSR, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit in the SSR to 1, transmits the MSB, then holds the transmit data pin (TxD) in the MSB state. If the transmit-end interrupt enable bit (TEIE) in the SCR is set to 1, a transmit-end interrupt (TEI) is requested at this time.
4. After the end of serial transmission, the SCK pin is held in the high state.

Receiving Serial Data (Clock Synchronous Mode): Figures 12.21 and 12.22 show a sample flowchart for receiving serial data. When switching from the asynchronous mode to the clock synchronous mode, make sure that ORER, PER, and FER are cleared to 0. If PER or FER is set to 1, the RDRF bit will not be set and both transmitting and receiving will be disabled.

The procedure for receiving serial data is listed below:

1. SCI initialization: Set the RxD pin using the PFC.
2. Receive error handling: If a receive error occurs, read the ORER bit in SSR to identify the error. After executing the necessary error handling, clear ORER to 0. Transmitting/receiving cannot resume if ORER remains set to 1.
3. SCI status check and receive data read: Read the serial status register (SSR), check that RDRF is set to 1, then read receive data from the receive data register (RDR) and clear RDRF to 0. The RxI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
4. Continue receiving serial data: Read RDR, and clear RDRF to 0 before the frame MSB (bit 7) of the current frame is received. If the DMAC is started by a receive-data-full interrupt (RxI) to read RDR, the RDRF bit is cleared automatically so this step is unnecessary.

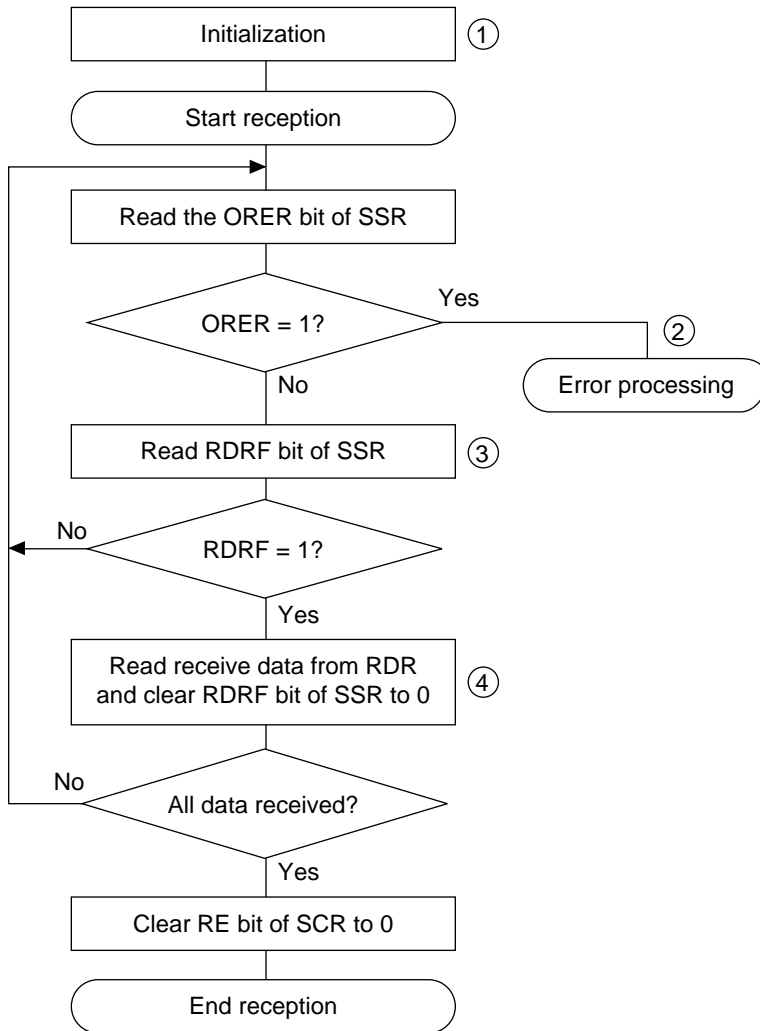


Figure 12.21 Sample Flowchart for Serial Receiving (1)

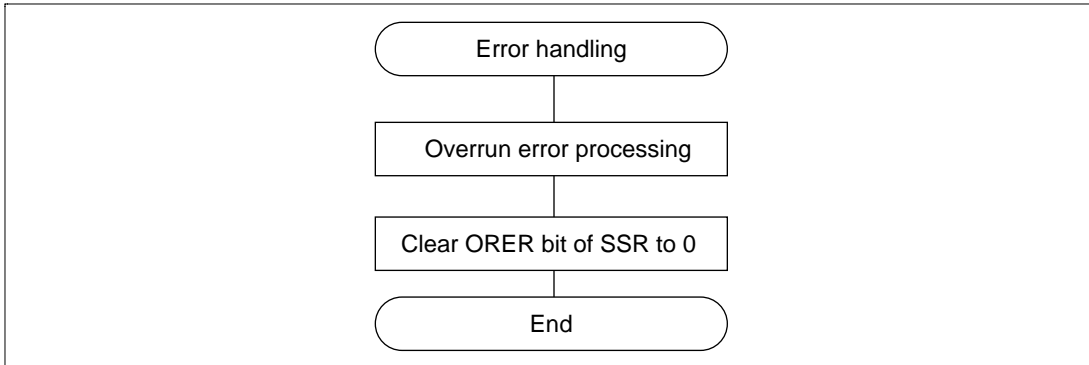


Figure 12.22 Sample Flowchart for Serial Receiving (2)

Figure 12.23 shows an example of the SCI receive operation.

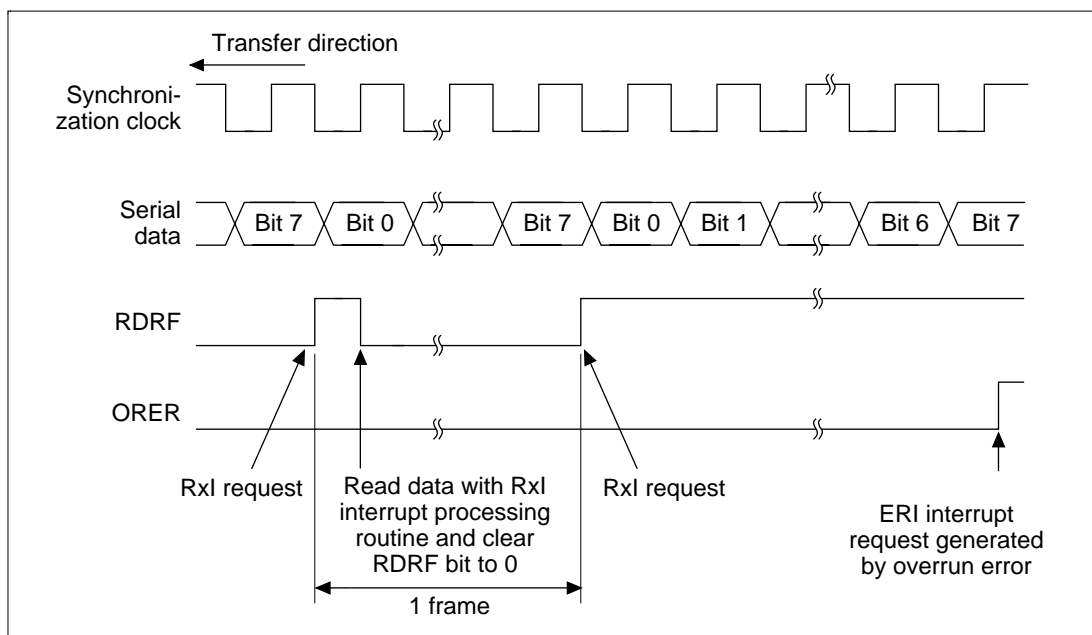


Figure 12.23 Example of SCI Receive Operation

In receiving, the SCI operates as follows:

1. The SCI synchronizes with serial clock input or output and initializes internally.
2. Receive data is shifted into the RSR in order from the LSB to the MSB. After receiving the data, the SCI checks that RDRF is 0 so that receive data can be loaded from the RSR into the RDR. If this check passes, the SCI sets RDRF to 1 and stores the received data in the RDR. If the check does not pass (receive error), the SCI operates as indicated in table 12.11 and no further transmission or reception is possible. If the error flag is set to 1, the RDRF bit is not set

to 1 during reception, even if the RDRF bit is 0 cleared. When restarting reception, be sure to clear the error flag.

3. After setting RDRF to 1, if the receive-data-full interrupt enable bit (RIE) is set to 1 in the SCR, the SCI requests a receive-data-full interrupt (RxI). If the ORER bit is set to 1 and the receive-data-full interrupt enable bit (RIE) in the SCR is also set to 1, the SCI requests a receive-error interrupt (ERI).

Transmitting and Receiving Serial Data Simultaneously (Clock Synchronous Mode): Figure 12.24 shows a sample flowchart for transmitting and receiving serial data simultaneously. The procedure is as follows (the steps correspond to the numbers in the flowchart):

1. SCI initialization: Set the TxD and RxD pins using the PFC.
2. SCI status check and transmit data write: Read the serial status register (SSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (TDR) and clear TDRE to 0. The TxI interrupt can also be used to determine if the TDRE bit has changed from 0 to 1.
3. Receive error handling: If a receive error occurs, read the ORER bit in SSR to identify the error. After executing the necessary error processing, clear ORER to 0. Transmitting/receiving cannot resume if ORER remains set to 1.
4. SCI status check and receive data read: Read the serial status register (SSR), check that RDRF is set to 1, then read receive data from the receive data register (RDR) and clear RDRF to 0. The RxI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
5. Continue transmitting and receiving serial data: Read the RDRF bit and RDR, and clear RDRF to 0 before the frame MSB (bit 7) of the current frame is received. Also read the TDRE bit to check whether it is safe to write (if it reads 1); if so, write data in TDR, then clear TDRE to 0 before the MSB (bit 7) of the current frame is transmitted. When the DMAC is started by a transmit-data-empty interrupt request (TxI) to write data in TDR, the TDRE bit is checked and cleared automatically. When the DMAC is started by a receive-data-full interrupt (RxI) to read RDR, the RDRF bit is cleared automatically.

Note: In switching from transmitting or receiving to simultaneous transmitting and receiving, simultaneously clear the TE bit and RE bit to 0, then simultaneously set the TE bit and RE bit to 1.

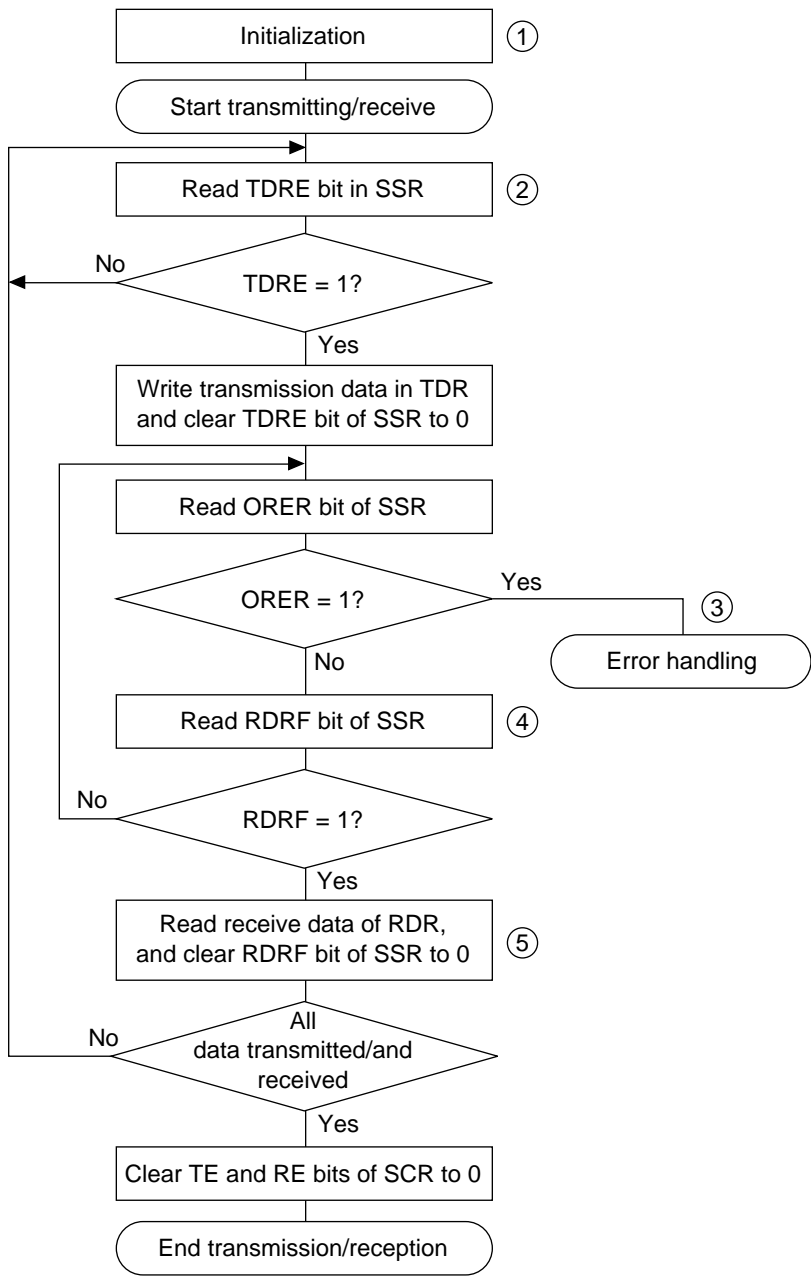


Figure 12.24 Sample Flowchart for Serial Transmission

12.4 SCI Interrupt Sources and the DMAC

The SCI has four interrupt sources: transmit-end (TEI), receive-error (ERI), receive-data-full (RxI), and transmit-data-empty (TxI). Table 12.12 lists the interrupt sources and indicates their priority. These interrupts can be enabled and disabled by the TIE, RIE, and TEIE bits in the serial control register (SCR). Each interrupt request is sent separately to the interrupt controller.

TxI is requested when the TDRE bit in the SSR is set to 1. TxI can start the direct memory access controller (DMAC) to transfer data. TDRE is automatically cleared to 0 when the DMAC writes data in the transmit data register (TDR).

RxI is requested when the RDRF bit in the SSR is set to 1. RxI can start the DMAC to transfer data. RDRF is automatically cleared to 0 when the DMAC reads the receive data register (RDR).

ERI is requested when the ORER, PER, or FER bit in the SSR is set to 1. ERI cannot start the DMAC.

TEI is requested when the TEND bit in the SSR is set to 1. TEI cannot start the DMAC. Where the TxI interrupt indicates that transmit data writing is enabled, the TEI interrupt indicates that the transmit operation is complete.

Table 12.12 SCI Interrupt Sources

| Interrupt Source | Description | DMAC Activation | Priority |
|-------------------------|-----------------------------------|------------------------|-----------------|
| ERI | Receive error (ORER, PER, or FER) | No | High |
| RxI | Receive data full (RDRF) | Yes | |
| TxI | Transmit data empty (TDRE) | Yes | |
| TEI | Transmit end (TEND) | No | Low |

12.5 Notes on Use

Sections 12.5.1 through 12.5.9 provide information for using the SCI.

12.5.1 TDR Write and TDRE Flags

The TDRE bit in the serial status register (SSR) is a status flag indicating loading of transmit data from TDR into TSR. The SCI sets TDRE to 1 when it transfers data from TDR to TSR. Data can be written to TDR regardless of the TDRE bit status. If new data is written in TDR when TDRE is 0, however, the old data stored in TDR will be lost because the data has not yet been transferred to the TSR. Before writing transmit data to the TDR, be sure to check that TDRE is set to 1.

12.5.2 Simultaneous Multiple Receive Errors

Table 12.13 indicates the state of the SSR status flags when multiple receive errors occur simultaneously. When an overrun error occurs, the RSR contents cannot be transferred to the RDR, so receive data is lost.

Table 12.13 SSR Status Flags and Transfer of Receive Data

| Receive Error Status | SSR Status Flags | | | | Receive Data Transfer |
|--|------------------|------|-----|-----|-----------------------|
| | RDRF | ORER | FER | PER | RSR → RDR |
| Overrun error | 1 | 1 | 0 | 0 | X |
| Framing error | 0 | 0 | 1 | 0 | O |
| Parity error | 0 | 0 | 0 | 1 | O |
| Overrun error + framing error | 1 | 1 | 1 | 0 | X |
| Overrun error + parity error | 1 | 1 | 0 | 1 | X |
| Framing error + parity error | 0 | 0 | 1 | 1 | O |
| Overrun error + framing error + parity error | 1 | 1 | 1 | 1 | X |

Note: O = Receive data is transferred from RSR to RDR.

X = Receive data is not transferred from RSR to RDR.

12.5.3 Break Detection and Processing

Break signals can be detected by reading the RxD pin directly when a framing error (FER) is detected. In the break state, the input from the RxD pin consists of all 0s, so FER is set and the parity error flag (PER) may also be set. In the break state, the SCI receiver continues to operate, so if the FER bit is cleared to 0, it will be set to 1 again.

12.5.4 Sending a Break Signal

The TxD pin becomes a general I/O pin with the I/O direction and level determined by the I/O port data register (DR) and pin function controller (PFC) control register (CR). These conditions allow break signals to be sent. The DR value is substituted for the marking status until the PFC is set. Consequently, the output port is set to initially output a 1. To send a break in serial transmission, first clear the DR to 0, then establish the TxD pin as an output port using the PFC. When TE is cleared to 0, the transmission section is initialized regardless of the present transmission status.

12.5.5 Receive Error Flags and Transmitter Operation (Clock Synchronous Mode Only)

When a receive error flag (ORER, PER, or FER) is set to 1, the SCI will not start transmitting even if TDRE is set to 1. Be sure to clear the receive error flags to 0 before starting to transmit. Note that clearing RE to 0 does not clear the receive error flags.

12.5.6 Receive Data Sampling Timing and Receive Margin in the Asynchronous Mode

In the asynchronous mode, the SCI operates on a base clock of 16 times the bit rate frequency. In receiving, the SCI synchronizes internally with the falling edge of the start bit, which it samples on the base clock. Receive data is latched on the rising edge of the eighth base clock pulse (figure 12.25).

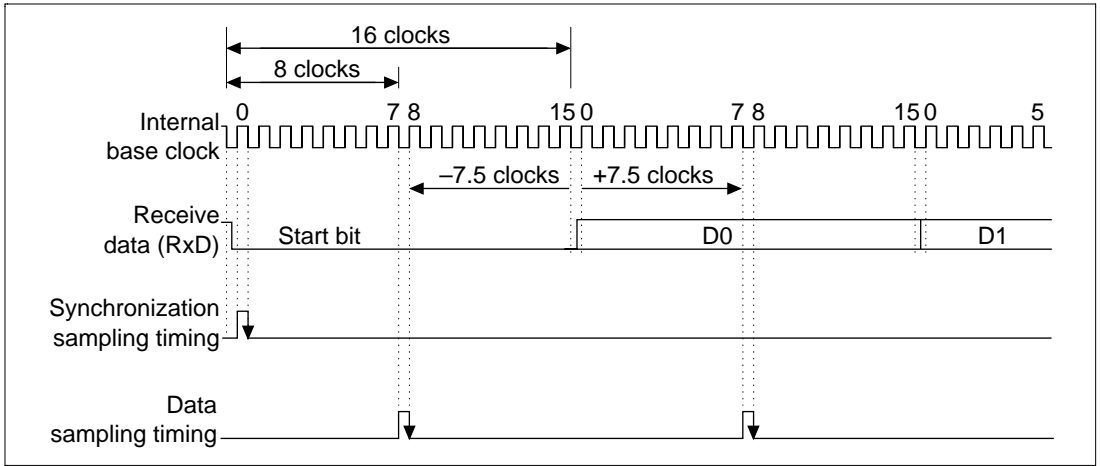


Figure 12.25 Receive Data Sampling Timing in Asynchronous Mode

The receive margin in the asynchronous mode can therefore be expressed as:

$$M = \left| \left(0.5 - \frac{1}{2N} \right) - (L - 0.5)F - \frac{|D - 0.5|}{N} (1 + F) \right| \times 100\%$$

M : Receive margin (%)

N : Ratio of clock frequency to bit rate (N = 16)

D : Clock duty cycle (D = 0–1.0)

L : Frame length (L = 9–12)

F : Absolute deviation of clock frequency

From the equation above, if F = 0 and D = 0.5 the receive margin is 46.875%:

$$D = 0.5, F = 0$$

$$M = (0.5 - 1/(2 \times 16)) \times 100\%$$

$$= 46.875\%$$

This is a theoretical value. A reasonable margin to allow in system designs is 20–30%.

12.5.7 Constraints on DMAC Use

- When using an external clock source for the synchronization clock, update the TDR with the DMAC or the DTC, and then after five system clocks or more elapse, input a transmit clock. If a transmit clock is input in the first four system clocks after the TDR is written, an error may occur (figure 12.26).
- Before reading the receive data register (RDR) with the DMAC, select the receive-data-full interrupt of the SCI as a start-up source.

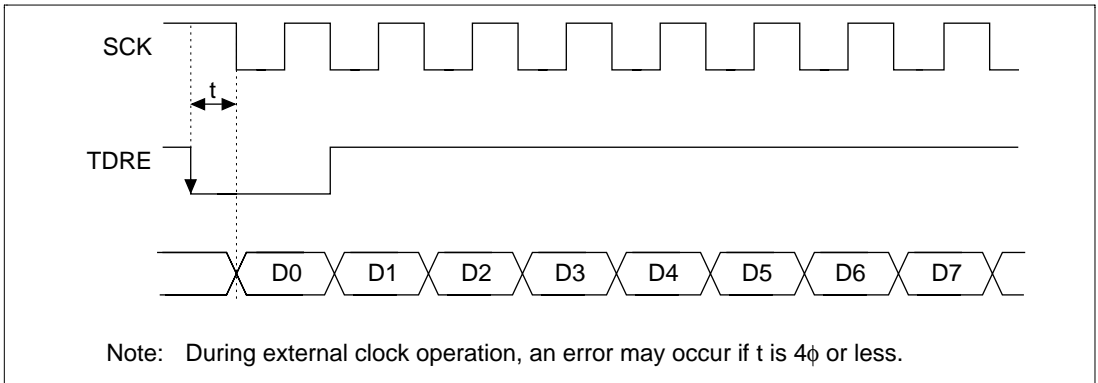


Figure 12.26 Example of Clock Synchronous Transmission with DMAC

12.5.8 Cautions for Clock Synchronous External Clock Mode

- Set $TE = RE = 1$ only when the external clock SCK is 1.
- Do not set $TE = RE = 1$ until at least four clocks after the external clock SCK has changed from 0 to 1.
- When receiving, RDRF is 1 when RE is set to zero 2.5–3.5 clocks after the rising edge of the RxD D7 bit SCK input, but it cannot be copied to RDR.

12.5.9 Caution for Clock Synchronous Internal Clock Mode

When receiving, RDRF is 1 when RE is set to zero 1.5 clocks after the rising edge of the RxD D7 bit SCK output, but it cannot be copied to RDR.

13.1 Overview

The high speed A/D converter has 10-bit resolution, and can select from a maximum of eight channels of analog inputs.

13.1.1 Features

The high speed A/D converter has the following features:

- 10-bit resolution
- Eight input channels
- High-speed conversion
 - Minimum conversion time: 2.9 μ s per channel (for 28-MHz operation)
- Multiple conversion modes
 - Select mode/group mode
 - Single mode/scan mode
 - Buffered operation possible
 - 2 channel simultaneous sampling possible
- Two types of conversion start
 - Software, or timer conversion start trigger (MTU) can be selected.
- Eight data registers
 - Conversion results stored in 16-bit data registers corresponding to each channel.
- Sample and hold function
- A/D conversion end interrupt generation
 - An A/D conversion end interrupt (ADI) request can be generated on completion of A/D conversions

13.1.2 Block Diagram

Figure 13.1 is the block diagram of the high speed A/D converter.

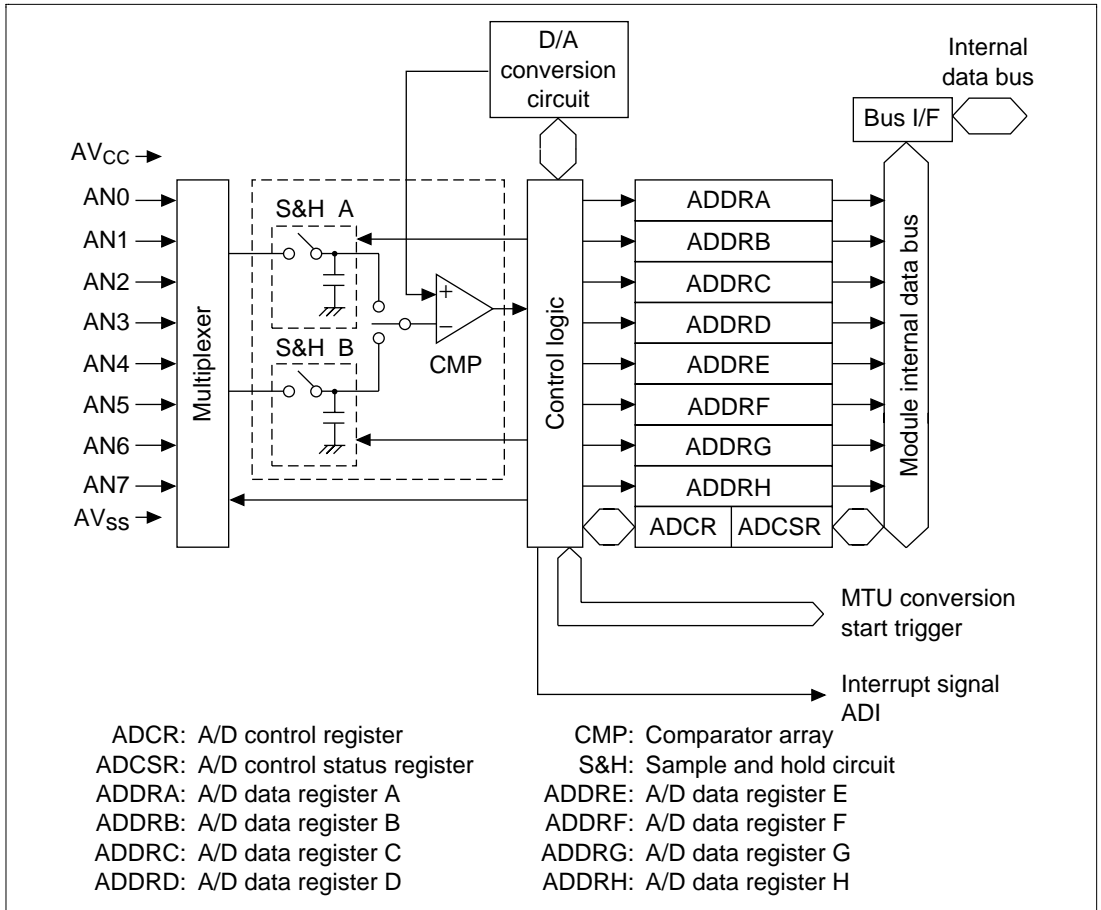


Figure 13.1 High Speed A/D Converter Block Diagram

13.1.3 Pin Configuration

Table 13.1 shows the input pins used by the high speed A/D converter.

The AV_{CC} and AV_{SS} pins are for the high speed A/D converter internal analog section power supply.

Table 13.1 Pin Configuration

| Pin | Abbreviation | I/O | Function |
|----------------|--------------|-----|--|
| Analog supply | AV_{CC} | I | Analog section power supply |
| Analog ground | AV_{SS} | I | Analog section ground and A/D conversion reference voltage |
| Analog input 0 | AN0 | I | Analog input channel 0 |
| Analog input 1 | AN1 | I | Analog input channel 1 |
| Analog input 2 | AN2 | I | Analog input channel 2 |
| Analog input 3 | AN3 | I | Analog input channel 3 |
| Analog input 4 | AN4 | I | Analog input channel 4 |
| Analog input 5 | AN5 | I | Analog input channel 5 |
| Analog input 6 | AN6 | I | Analog input channel 6 |
| Analog input 7 | AN7 | I | Analog input channel 7 |

13.1.4 Register Configuration

Table 13.2 shows the configuration of the high speed A/D converter registers.

Table 13.2 Register Configuration

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|-----------------------------|--------------|--------|---------------|-----------|-------------|
| A/D data register A | ADDRA | R | H'0000 | H'FFF83F0 | 8,16 |
| A/D data register B | ADDRB | R | H'0000 | H'FFF83F2 | |
| A/D data register C | ADDRC | R | H'0000 | H'FFF83F4 | |
| A/D data register D | ADDRD | R | H'0000 | H'FFF83F6 | |
| A/D data register E | ADDRE | R | H'0000 | H'FFF83F8 | |
| A/D data register F | ADDRF | R | H'0000 | H'FFF83FA | |
| A/D data register G | ADDRG | R | H'0000 | H'FFF83FC | |
| A/D data register H | ADDRH | R | H'0000 | H'FFF83FE | |
| A/D control/status register | ADCSR | R/(W)* | H'00 | H'FFF83E0 | |
| A/D control register | ADCR | R/W | H'00 | H'FFF83E1 | |

Note: Only 0 can be written to bit 7 to clear the flag.

13.2 Register Descriptions

13.2.1 A/D Data Registers A–H (ADDRA–ADDRH)

The ADDR are 16-bit read only registers for storing A/D conversion results. There are eight of these registers, ADDRA through ADDRH.

The A/D converted data is 10-bit data which is sent to the ADDR for the corresponding converted channel for storage. The lower 8 bits of the A/D converted data are transferred to and stored in the lower byte (bits 7–0) of the ADDR, and the upper 2 bits are stored into the upper byte (bits 9, 8). Bits 15–10 always read as 0. Data reads can be either byte or word. The upper 8 bits of the converted data are transferred upon byte data reads. Additionally, buffered operation is possible by combining ADDRA–ADDRD.

Table 13.3 shows the correspondence between the analog input channels and the ADDR.

The ADDR are initialized to H'0000 by power-on reset or in standby mode.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | AD9 | AD8 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |

Table 13.3 Analog Input Channel and ADDR Correspondence

| Analog Input Channel | A/D Data Register |
|----------------------|-------------------|
| AN0 | ADDRA* |
| AN1 | ADDRB* |
| AN2 | ADDRC* |
| AN3 | ADDRD* |
| AN4 | ADDRE |
| AN5 | ADDRF |
| AN6 | ADDRG |
| AN7 | ADDRH |

Note: Except during buffer operation

13.2.2 A/D Control/Status Register (ADCSR)

The ADCSR is an 8-bit read/write register used for A/D conversion operation control and to indicate status.

The ADCSR is initialized to H'00 by power-on reset or in standby mode.

| | | | | | | | | |
|----------------|--------|------|------|-----|-----|-----|-----|-----|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ADF | ADIE | ADST | CKS | GRP | CH2 | CH1 | CH0 |
| 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: The only value that can be written is a 0 to clear the flag.

- Bit 7—A/D End Flag (ADF): This status flag indicates that A/D conversion has ended.

| Bit 7: ADF | Description |
|------------|---|
| 0 | Clear conditions (initial value) <ul style="list-style-type: none"> • With ADF = 1, by reading the ADF flag then writing 0 in ADF • When the DMAC is activated by an ADI interrupt |
| 1 | Set conditions <ul style="list-style-type: none"> • Single mode: When A/D conversion ends after conversion for all designated channels (during buffer operation, this is not set until operation of the specified buffer has ended) • Scan mode: After one round of A/D conversion for all specified channels |

- Bit 6—A/D Interrupt Enable (ADIE): Enables or disables interrupt requests (ADI) after A/D conversion ends. Set the ADIE bit while conversion is suspended.

| Bit 6: ADIE | Description |
|-------------|---|
| 0 | Disables interrupt requests (ADI) after A/D conversion ends (initial value) |
| 1 | Enables interrupt requests (ADI) after A/D conversion ends |

- Bit 5—A/D Start (ADST): Selects start or stop for A/D conversion. A 1 is maintained during A/D conversions.

The ADST bit can be set to 1 by software, or timer conversion start triggers.

| Bit 5: ADST | Description |
|-------------|--|
| 0 | A/D conversion halted (initial value) |
| 1 | Single mode: Start A/D conversion. Automatically cleared to 0 after conversion for the designated channel ends. Scan mode: Start A/D conversion. Continuous conversion until 0 cleared by software. |

- Bit 4—Clock Select (CKS): Sets the A/D conversion time. Set so that the conversion time is 2 μ s or greater in proportion to the operating frequency.

Make conversion time changes only while conversion is halted.

| Bit 4: CKS | Description |
|------------|---|
| 0 | Conversion time = 40 states (high speed A/D converter standard clock = $\phi/2$) (initial value) |
| 1 | Conversion time = 80 states (when $\phi/4$ is selected) |

- Bit 3—Group Mode (GRP): Designates either select mode or group mode for the A/D conversion channel selection.

Set the GRP bit only while conversion is halted.

| Bit 3: GRP | Description |
|------------|-----------------------------|
| 0 | Select mode (initial value) |
| 1 | Group mode |

- Bits 2–0—Channel Select 2–0 (CH2–CH0): These bits, along with the GRP bit, select the analog input channel.

Set the input channel only while conversion is halted.

| Bit 2: CH2 | Bit 1: CH1 | Bit 0: CH0 | Description | |
|------------|------------|------------|-----------------------|----------------------|
| | | | Select Mode (GRP = 0) | Group Mode (GRP = 1) |
| 0 | 0 | 0 | AN0 (initial value) | AN0 |
| 0 | 0 | 1 | AN1 | AN0–AN1 |
| 0 | 1 | 0 | AN2 | AN0–AN2 |
| 0 | 1 | 1 | AN3 | AN0–AN3 |
| 1 | 0 | 0 | AN4 | AN0–AN4 |
| 1 | 0 | 1 | AN5 | AN0–AN5 |
| 1 | 1 | 0 | AN6 | AN0–AN6 |
| 1 | 1 | 1 | AN7 | AN0–AN7 |

13.2.3 A/D Control Register (ADCR)

The ADCR is an 8-bit read/write register used for A/D conversion operation control. The ADCR is initialized to H'00 by power-on reset or in standby mode.

| | | | | | | | | |
|----------------|---|-----|-------|-------|------|------|-------|-------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | PWR | TRGS1 | TRGS0 | SCAN | DSMP | BUFE1 | BUFE0 |
| 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 | R/W |

- Bit 7—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 6—Power (PWR): Designates the conversion start mode for the high speed A/D converter. Setting the PWR bit to 1 sets high speed start mode, and a 0 sets to low power conversion mode. See section 13.4.7, Conversion Start Modes for details on the conversion start operation. Set the PWR bit only while conversion is halted.

| Bit 6: PWR | Description |
|------------|---|
| 0 | Low power conversion mode (initial value) |
| 1 | High speed start mode |

- Bits 5–4—Timer Trigger Select 1, 0 (TRGS1, TRGS0): These bits enable or prohibit A/D conversion starts by trigger signals. Set the TRGS1, TRGS0 bits only while conversion is halted.

| Bit 5: TRGS1 | Bit 4: TRGS0 | Description |
|--------------|--------------|--|
| 0 | 0 | Enable A/D conversion start by software (initial value) |
| 0 | 1 | Enables A/D conversion start by MTU conversion start trigger |
| 1 | 0 | Reserved |
| 1 | 1 | Reserved |

- Bit 3—Scan Mode (SCAN): Selects either single mode or scan mode for the A/D conversion operation mode. See section 13.4, Operation, for details on single mode and scan mode operation. Set the SCAN bit only while conversion is halted.

| Bit 3: SCAN | Description |
|-------------|-----------------------------|
| 0 | Single mode (initial value) |
| 1 | Scan mode |

- Bit 2—Simultaneous Sampling (DSMP): Enables or disables the simultaneous sampling of two channels. See section 13.4.6, Simultaneous Sampling Operation, for details on simultaneous sampling.

Set the DSMP bit only while conversion is halted.

| Bit 2: DSMP | Description |
|-------------|---|
| 0 | Normal sampling operation (initial value) |
| 1 | Simultaneous sampling operation |

- Bits 1–0—Buffer Enable 1, 0 (BUFE1, BUFE0): These bits select whether to use the ADDR_B–ADDR_D as buffer registers.

Set the BUFE1 and BUFE0 bits only while conversion is halted.

| Bit 1: BUFE1 | Bit 0: BUFE0 | Description |
|--------------|--------------|---|
| 0 | 0 | Normal operation (initial value) |
| 0 | 1 | ADDRA and ADDR _B buffer operation: conversion result → ADDRA → ADDR _B (ADDR _B is the buffer register) |
| 1 | 0 | ADDRA and ADDR _C , also ADDR _B and ADDR _D buffer operation: conversion result 1 → ADDRA → ADDR _C , conversion result 2 → ADDR _B → ADDR _D (ADDR _C and ADDR _D are buffer registers) |
| 1 | 1 | ADDRA to ADDR _D buffer operation: conversion result → ADDRA → ADDR _B → ADDR _C → ADDR _D (ADDR _B –ADDR _D are buffer registers) |

13.3 Bus Master Interface

The ADDR_A–ADDR_H are 16-bit registers with a 16-bit width data bus to the bus master. The bus master can read from ADDR_A–ADDR_H in either word or byte units.

When an ADDR is read in word units, the ADDR contents are transferred to the bus master 16 bits at a time. In byte unit reads, the contents of the most significant eight bits (AD₉–AD₂) of the converted data (AD₉–AD₀) are transferred to the bus master.

Figures 13.2 and 13.3 show an example of the ADDR read operation.

Word data read

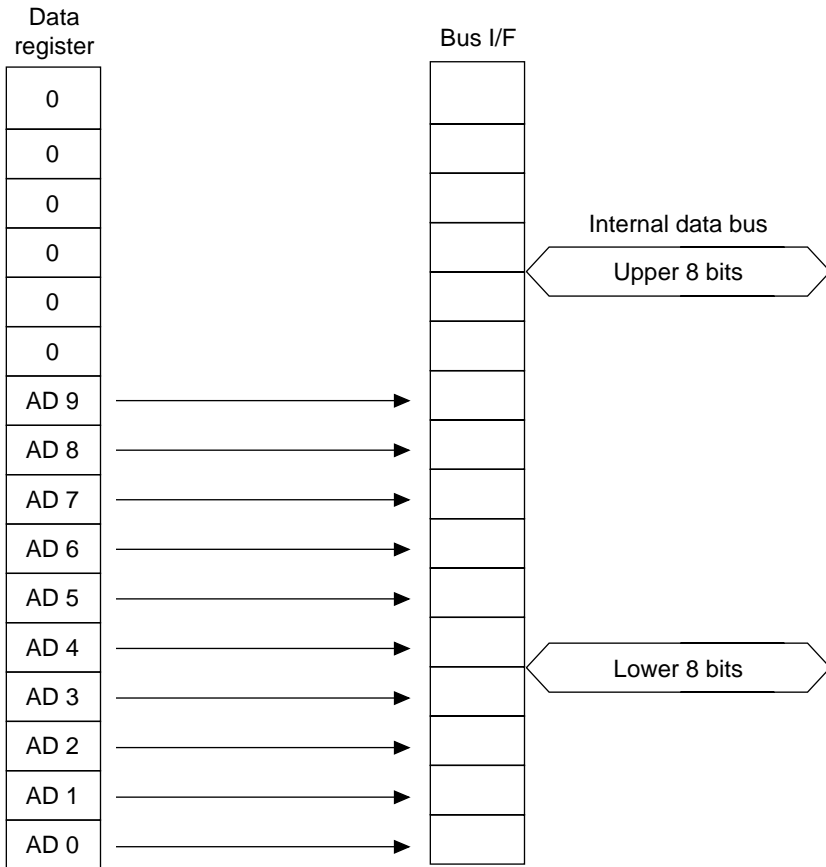
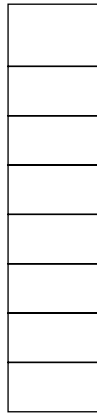


Figure 13.2 ADDR Read Operation (1)

Byte data read
Data register

| |
|------|
| 0 |
| 0 |
| 0 |
| 0 |
| 0 |
| 0 |
| AD 9 |
| AD 8 |
| AD 7 |
| AD 6 |
| AD 5 |
| AD 4 |
| AD 3 |
| AD 2 |
| AD 1 |
| AD 0 |

Bus I/F



Internal data bus

Upper 8 bits

Figure 13.3 ADDR Read Operation (2)

13.4 Operation

- The high speed A/D converter has 10-bit resolution.
- In addition to the four operating modes of select or group, and single or scan can be set in combination with buffer operation.
- Select mode uses one channel and group mode selects multiple channels.
- One start in the single mode performs conversions on all selected channels, and one start in the scan mode performs repeated conversions until stopped by software.
- In buffer operation, the previous conversion result is saved in a buffer register at the end of a conversion for the relevant channel.
- Software, a timer conversion start trigger (MTU) can be selected as the conversion start condition.
- High speed start mode or low power conversion mode can be selected for A/D conversion using the PWR bit setting.
- When changing the operation mode or input channel, rewrite the ADCSR, ADCR while the ADST bit is cleared to 0. After rewriting the ADCSR, ADCR, A/D conversion will be restarted when the ADST bit is set to 1. Operation mode or input channel changes can be made simultaneously with ADST bit setting. When stopping an A/D conversion before completion, 0 clear the ADST bit.

13.4.1 Select-Single Mode

Choose select-single mode when doing A/D conversions for one channel only.

When the ADST bit is set to 1, A/D conversion is started according to the designated conversion start conditions. The ADST bit is held to 1 during the A/D conversion and is automatically cleared to 0 upon completion.

The ADF flag is also set to 1 at the end of conversion. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. The ADF flag is cleared by reading the ADCSR, then writing a 0.

Figure 13.4 shows an example of operation in the select-single mode when AN1 is selected.

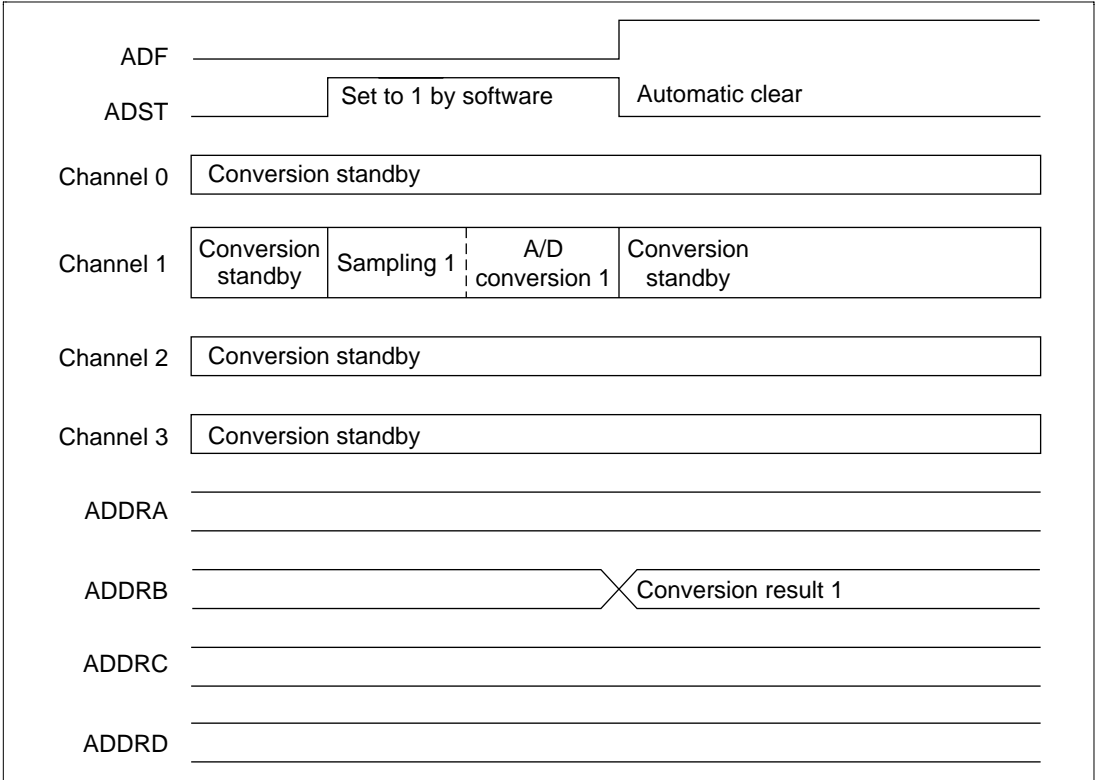


Figure 13.4 High Speed A/D Converter Operation Example (Select-Single Mode)

13.4.2 Select-Scan Mode

Choose select-scan mode when doing repeated A/D conversions for one channel. This is useful when doing continuous monitoring of the analog input of one channel.

When the ADST bit is set to 1, A/D conversion is started according to the designated conversion start conditions. The ADST bit is held to 1 until cleared by software. A/D conversion for the selected input channel is repeated during that interval.

The ADF flag is set to 1 at the end of the first conversion. At this point, if the ADIE bit is set, an ADI interrupt request is issued, and the high speed A/D converter is halted. With the high speed A/D converter in stop mode due to an ADI interrupt request, conversion is restarted when the ADF flag is cleared to 0. The ADF flag is cleared by reading the ADCSR then writing a 0.

Figure 13.5 shows an example of operation in the select-scan mode when AN1 is selected.

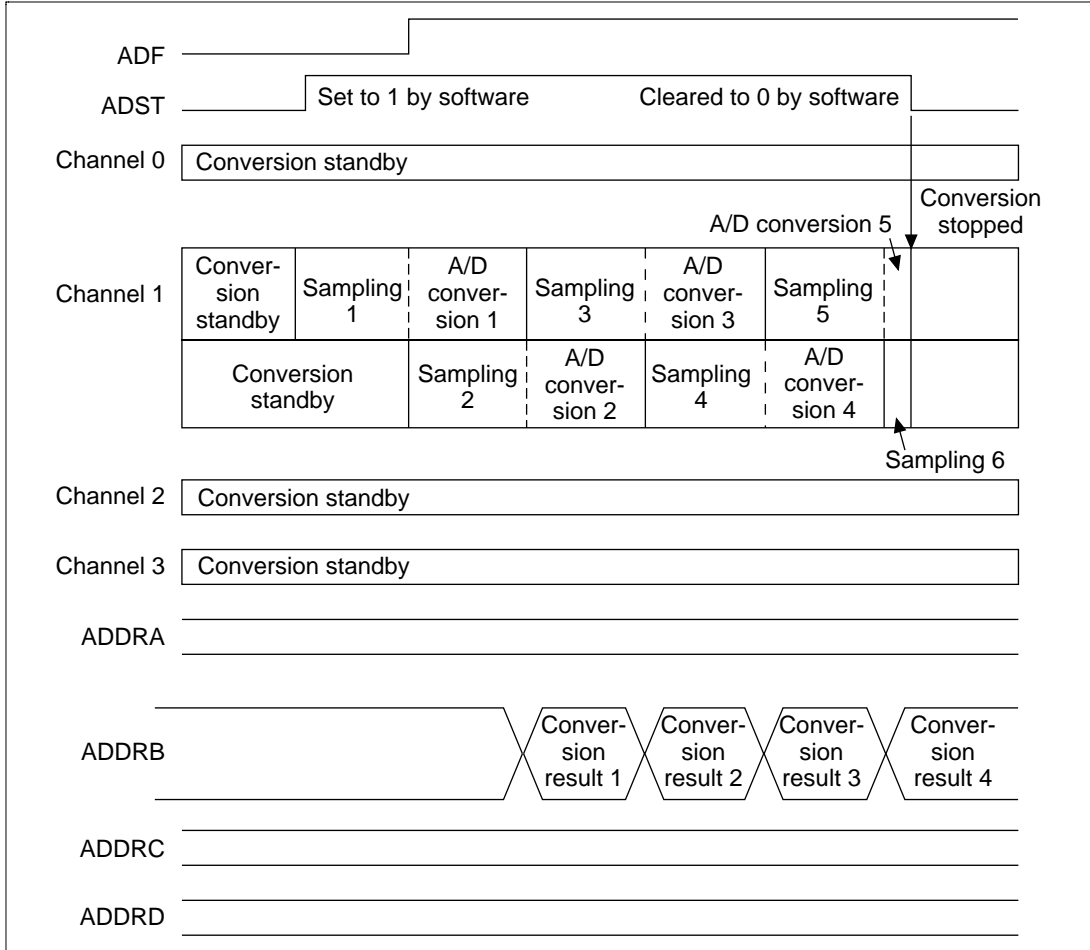


Figure 13.5 High Speed A/D Converter Operation Example (Select-Scan Mode)

13.4.3 Group-Single Mode

Choose group-single mode when doing A/D conversions for multiple channels.

When the ADST bit is set to 1, A/D conversion is started according to the designated conversion start conditions. The ADST bit is held to 1 during A/D conversion and is automatically cleared to 0 when all conversions for the designated input channels are completed.

The ADF flag is set to 1 when all conversions for the designated input channels are completed. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. The ADF flag is cleared by reading the ADCSR then writing a 0.

Figure 13.6 shows an example of operation in the group-single mode when AN0–AN2 are selected.

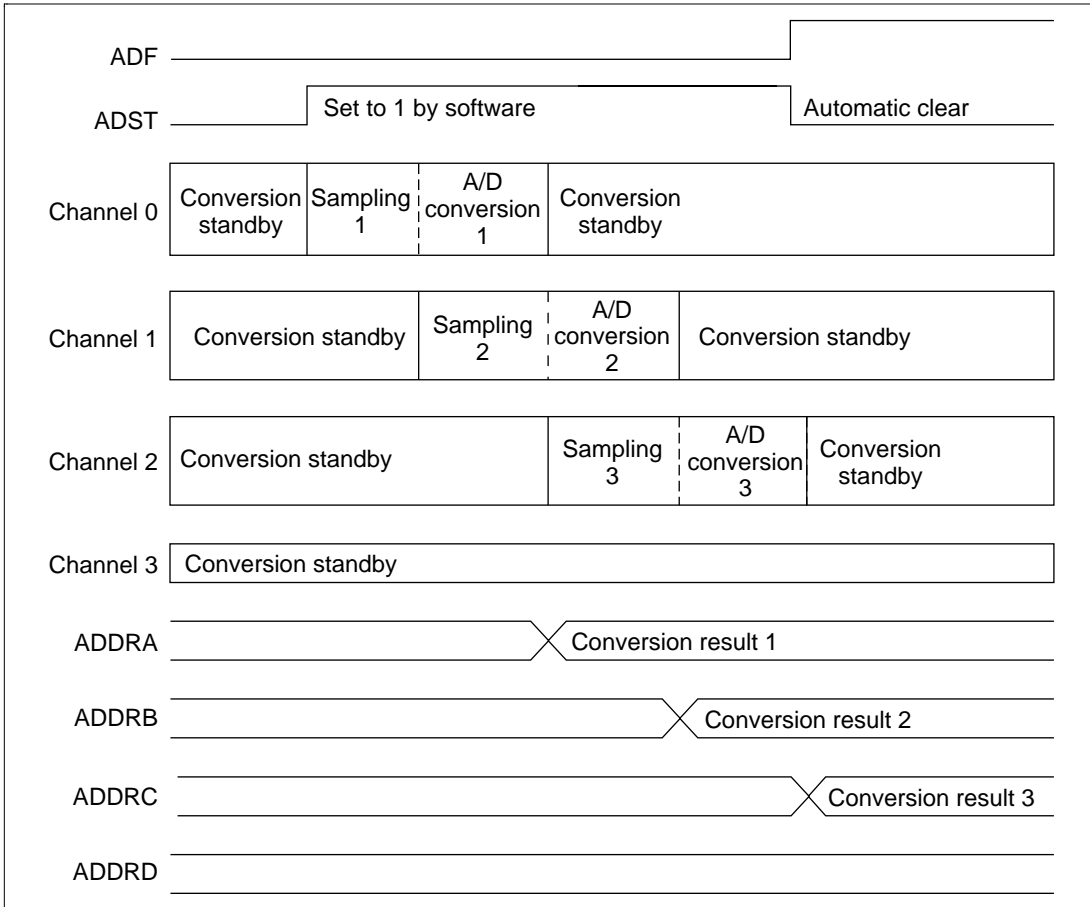


Figure 13.6 High Speed A/D Converter Operation Example (Group-Single Mode)

13.4.4 Group-Scan Mode

Choose group-scan mode when doing repeated A/D conversions for multiple channels. This is useful when doing continuous monitoring of the analog inputs of multiple channels.

When the ADST bit is set to 1, A/D conversion is started according to the designated conversion start conditions. The ADST bit is held to 1 until 0 cleared by software. A/D conversion for the selected input channels is repeated during that interval.

The ADF flag is set to 1 at the completion of the first conversions of all the designated input channels. At this point, if the ADIE bit is set to 1, an ADI interrupt request is issued, and the high speed A/D converter is temporarily halted. With the high speed A/D converter in stop mode due to an ADI interrupt request, conversion is restarted when the ADF flag is cleared to 0. The ADF flag is cleared by reading the ADCSR, then writing a 0.

Figure 13.7 shows an example of operation in the group-scan mode when AN0–AN2 are selected.

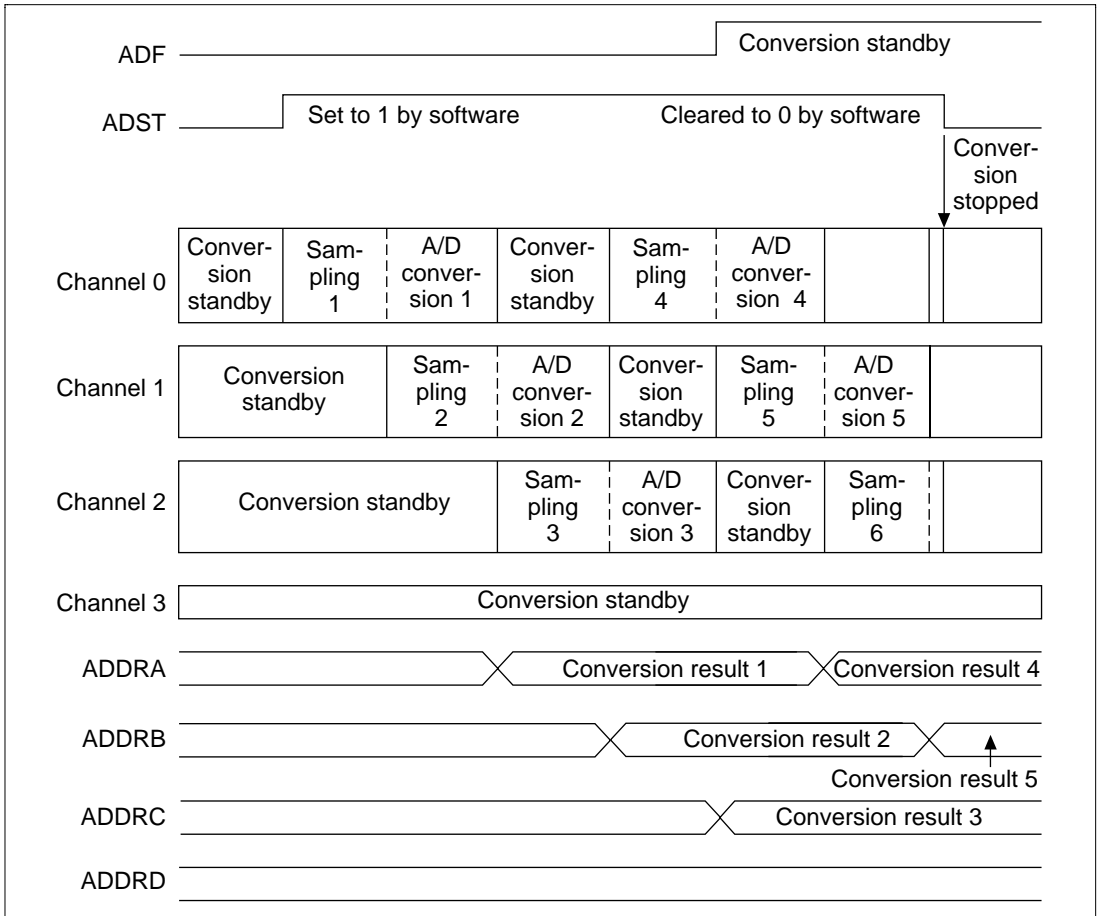


Figure 13.7 High Speed A/D Converter Operation Example (Group-Scan Mode)

13.4.5 Buffer Operation

When conversion ends on the relevant channel, the conversion result is stored in the ADDR, and simultaneously, the previously stored result is transferred to another ADDR. Buffer operation can be selected from the following:

- AN0 → ADDRA → ADDRb (Two-stage, one-group operation)
- AN0 → ADDRA → ADDRc, AN1 → ADDRb → ADDRd (Two-stage, two-group operation)
- AN0 → ADDRA → ADDRb → ADDRc → ADDRd (Four-stage, one-group operation)

To use in combination with simultaneous sampling, set GRP = 1, BUFE1, BUFE0 = B'10 and CH2 = 0. Buffer operation timing is shown in figure 13.8.

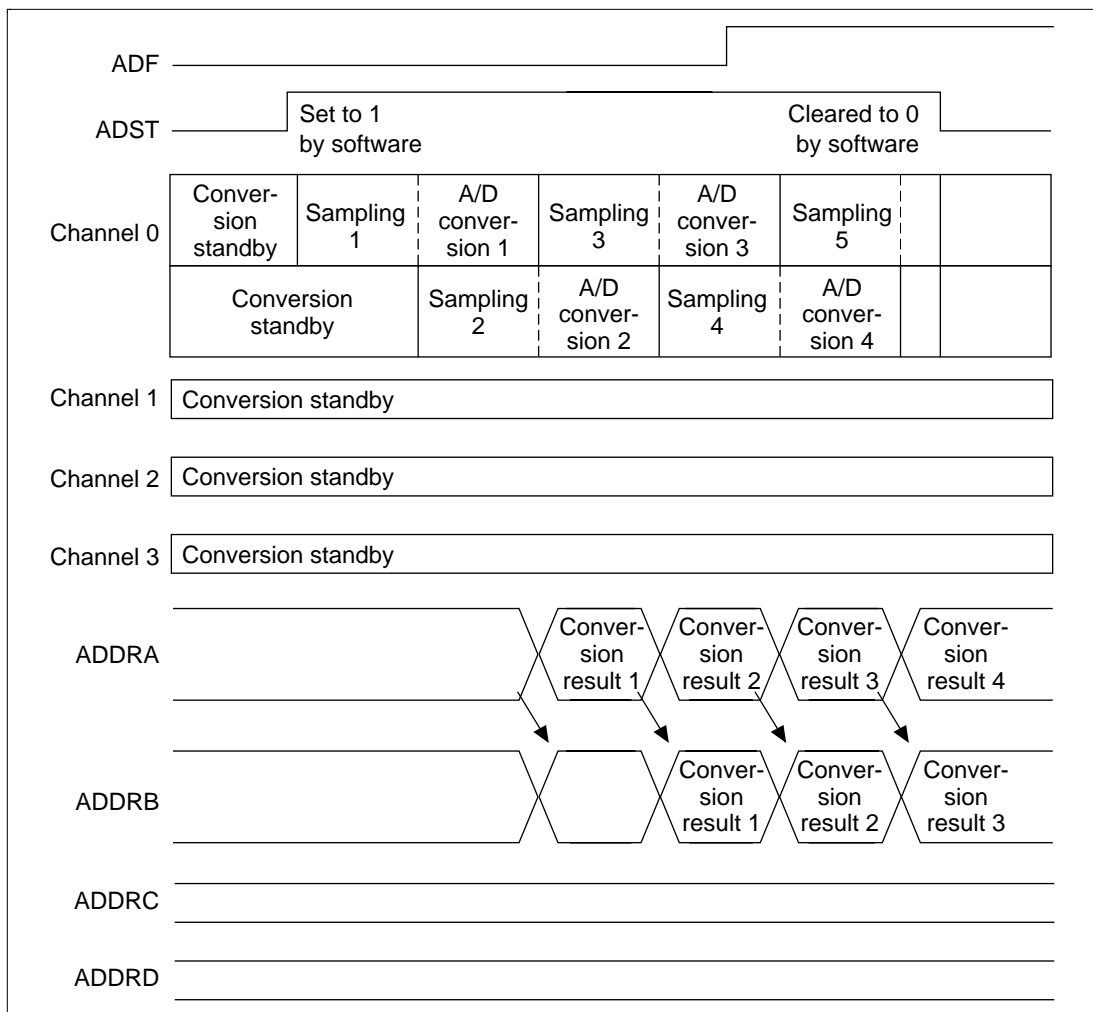


Figure 13.8 Buffer Operation Example (Select Scan Mode: Two-Stage One-Group Operation, When CH2–CH0 = B'001)

Buffer-Only Operation: When performing conversion only on the analog input channels specified by the BUFE1 and BUFE0 bits, select group mode, and you can select the ADF flag setting conditions with the CH2–CH0 bits.

Table 13.4 shows conversion during buffer operation and ADF flag setting conditions. The ADF flag is set at the point in the table when the final conversion has ended. In single mode, conversion is halted after the ADF flag is set to 1. In scan mode, conversion continues, and the converted data is stored in sequence in the buffer registers specified by the BUFE1 and BUFE0 bits.

When the ADF flag is set to 1, if the ADIE bit is also set to 1, an ADI interrupt is issued. After the ADCSR is read, the ADF flag is cleared by a 0 write.

With select single mode, the high speed A/D converter goes into standby mode at the end of every conversion cycle. The high speed A/D converter is restarted by software, or a timer trigger. When the number of conversion cycles shown in table 13.4 have ended, the ADF flag is set to 1.

Table 13.4 Conversion Channel and ADF Flag Setting/Clearing Conditions During Buffer Operation 1

| Channel Setting | | | Sampling Channel | | |
|-----------------|-----|-----|---------------------|--------------------------|---------------------|
| CH2 | CH1 | CH0 | BUFE1, BUFE0 = B'01 | BUFE1, BUFE0 = B'10 | BUFE1, BUFE0 = B'11 |
| 0 | 0 | 0 | AN0 1 time (ADDRA) | AN0, AN1 1 time (ADDRB) | AN0 1 time (ADDRA) |
| | | 1 | AN0 2 times (ADDRB) | | AN0 2 times (ADDRB) |
| | 1 | 0 | * | AN0, AN1 2 times (ADDRD) | AN0 3 times (ADDRC) |
| | | 1 | * | | AN0 4 times (ADDRD) |
| 1 | — | — | * | * | * |

Note: See table 13.5.

Combined Group Mode and Buffer Operation: Continuous conversion is possible on analog input channels (AN0 and AN1) specified by bits BUFE1 and BUFE0 as well as AN4–AN7 due to setting of bits CH2–CH0.

Table 13.5 shows conversion during buffer operation and ADF flag setting conditions. The ADF flag is set at the point in the table when the final conversion has ended. In this case, conversion is performed on the analog input corresponding with the ADDR specified in the buffer register. For example, when BUFE1 and BUFE0 = B'11 and CH2–CH0 = B'110, conversion results are stored in ADDRA and ADDRE–ADDRG. Also, contents of ADDRA–ADDRC before the start of conversion are transferred to ADDRb–ADDRD.

In single mode, conversion is halted after the ADF flag has been set to 1. Conversion continues in scan mode.

Table 13.5 Conversion Channel and ADF Flag Setting/Clearing Conditions During Buffer Operation 2

| Channel Setting | | | Sampling Channel | | |
|-----------------|-----|-----|-----------------------|----------------------------|-----------------------|
| CH2 | CH1 | CH0 | BUFE1, BUFE0 = B'01 | BUFE1, BUFE0 = B'10 | BUFE1, BUFE0 = B'11 |
| 0 | 0 | — | * | * | * |
| | 1 | 0 | AN0, AN2 (ADDRC) | | |
| | | 1 | AN0, AN2, AN3 (ADDRD) | | |
| 1 | 0 | 0 | AN0, AN2–AN4 (ADDRE) | AN0, AN1, AN4 (ADDRE) | AN0, AN4 (ADDRE) |
| | | 1 | AN0, AN2–AN5 (ADDRF) | AN0, AN1, AN4, AN5 (ADDRF) | AN0, AN4, AN5 (ADDRF) |
| | 1 | 0 | AN0, AN2–AN6 (ADDRG) | AN0, AN1, AN4–AN6 (ADDRG) | AN0, AN4–AN6 (ADDRG) |
| | | 1 | AN0, AN2–AN7 (ADDRH) | AN0, AN1, AN4–AN7 (ADDRH) | AN0, AN4–AN7 (ADDRH) |

Note: See table 13.4.

ADF Flag Clearing: When the DTC and DMAC are started up due to an A/D conversion end interrupt, the ADF flag is cleared when the ADDR specified in tables 13.4 or 13.5 has been read.

Resetting the Number of Buffer Operations: Clear the BUFE1 and BUFE0 bits to B'00 in conversion standby mode or when the converter has been halted. The number of buffer operations is cleared to 0.

Updating Buffer Operations: Clear the BUFE1 and BUFE0 bits to B'00 in conversion standby mode or when the converter has been halted. Thereafter, set BUFE1 and BUFE0, and the buffer operations shown in tables 13.4 and 13.5 are performed when conversion is resumed.

13.4.6 Simultaneous Sampling Operation

With simultaneous sampling, continuous conversion is conducted with sampling of the input voltages on two channels at the same time. Simultaneous sampling is valid in group mode. Channels for sampling are determined by the CH2 and CH1 bits of the RDSCR. The combinations are shown in table 13.6. For example, if GRP = 1 when CH2 and CH1 = B'11, sampling occurs in order in the following pairs: AN0, AN1 → AN2, AN3 → AN4, AN5 → AN6, AN7. Sampling timing is shown in figure 13.9.

Table 13.6 Simultaneous Sampling Channels

| Channel Setting | | |
|-----------------|-----|-------------------------------------|
| CH2 | CH1 | Sampling Channels, GRP 1 |
| 0 | 0 | AN0, AN1 |
| | 1 | AN0, AN1→AN2, AN3 |
| 1 | 0 | AN0, AN1→AN2, AN3→AN4, AN5 |
| | 1 | AN0, AN1→AN2, AN3→AN4, AN5→AN6, AN7 |

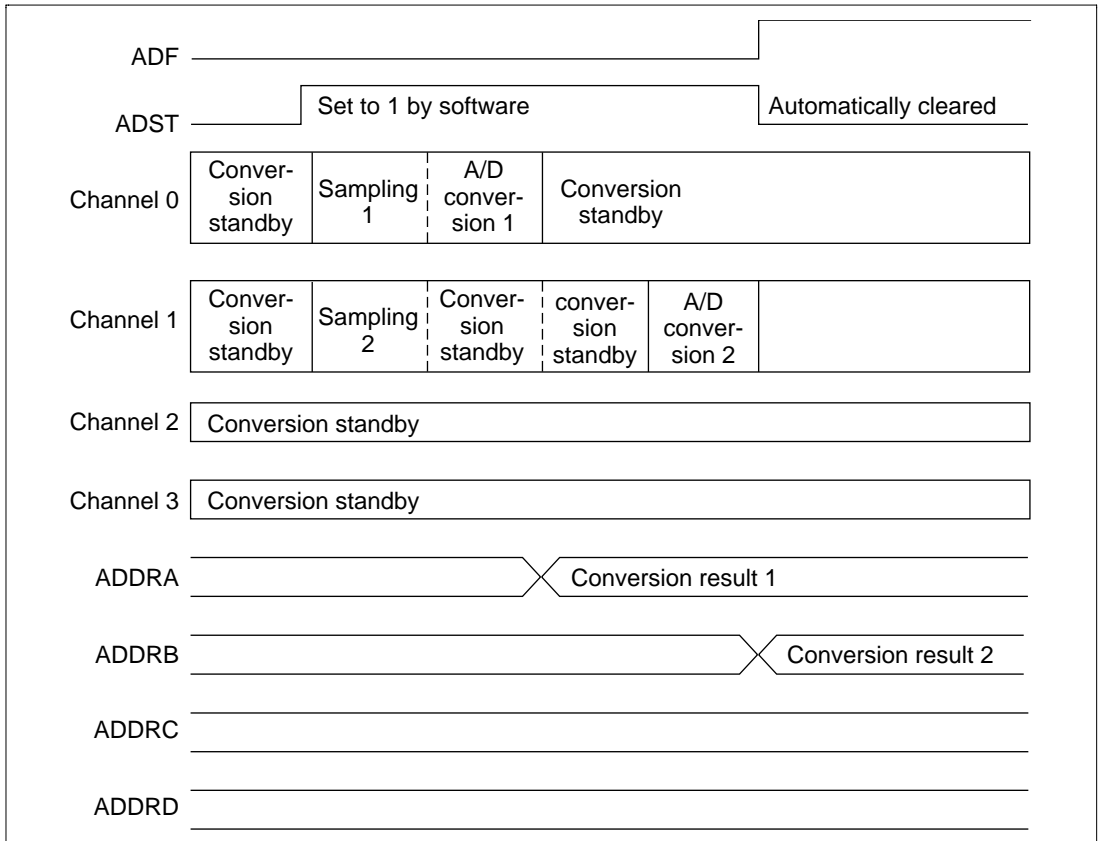


Figure 13.9 Simultaneous Sampling Operation (Group Single Mode)

13.4.7 Conversion Start Modes

The conversion start mode of the high speed A/D converter is set by the PWR bit of the ADCSR. When the PWR bit is cleared to 0, low-power conversion mode is set and the internal analog circuit becomes inactive. High-speed start mode is set by setting the PWR bit to 1, and the analog circuit becomes active.

In the low-power conversion mode, power is applied to the analog circuitry simultaneous to the conversion start (ADST set). When 200 cycles of the reference clock have elapsed, conversion becomes possible for the analog circuit and the first A/D conversion begins. When performing consecutive conversions, the second and later conversions are executed in 10 cycles. Select the basic clock with the CKS bit of the ADCSR. When the A/D conversion ends, ADST is cleared to 0 and the analog circuit power supply is automatically cut off. Because the analog circuit is only active during the A/D conversion operation period in this mode, current consumption can be reduced.

In high-speed start mode, ADST is cleared to 0 when A/D conversion ends. Power continues to be supplied to the analog circuitry, and conversion-ready status is maintained. Conversion is restarted immediately by resetting ADST to 1. However, the first conversion after power-on begins 200 cycles after setting ADST. Clear the PWR bit to 0 to switch off the analog power supply. When performing consecutive conversions, the second and later conversions are executed in 20 cycles. Because the analog circuit is always active in this mode, A/D conversion can be executed at high speed.

When A/D conversion is forcibly halted by clearing the ADST bit to 0 during conversion in high-speed start mode (when ADST = 1), the first conversion following a restart may not be performed normally. The second and subsequent conversions are performed normally.

Figures 13.10 and 13.11 show the timing of the conversion start operation.

Figures 13.10 and 13.11 show examples of conversion start operation timing.

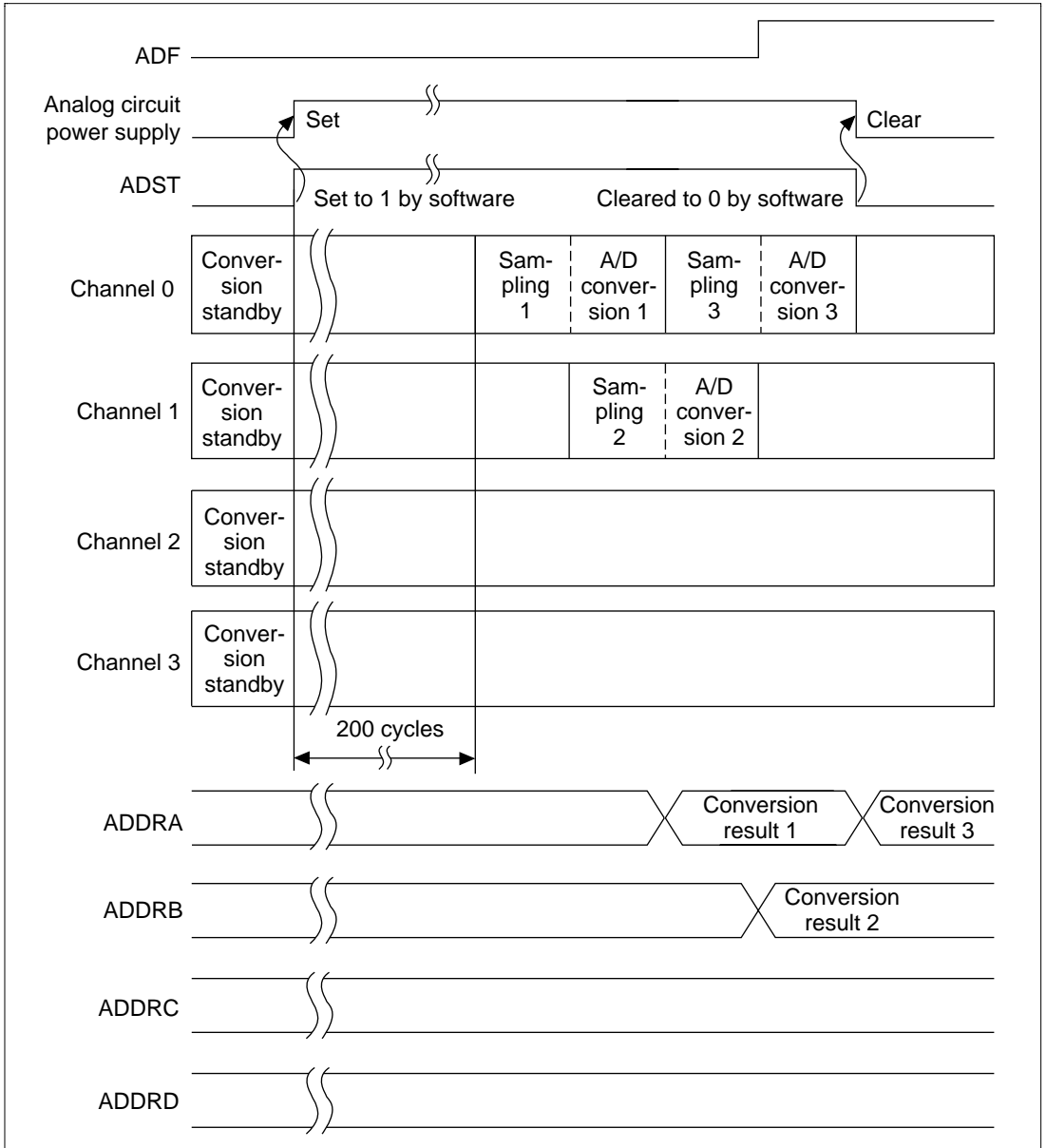


Figure 13.10 Conversion Start Operation (Low-Power Conversion Mode)

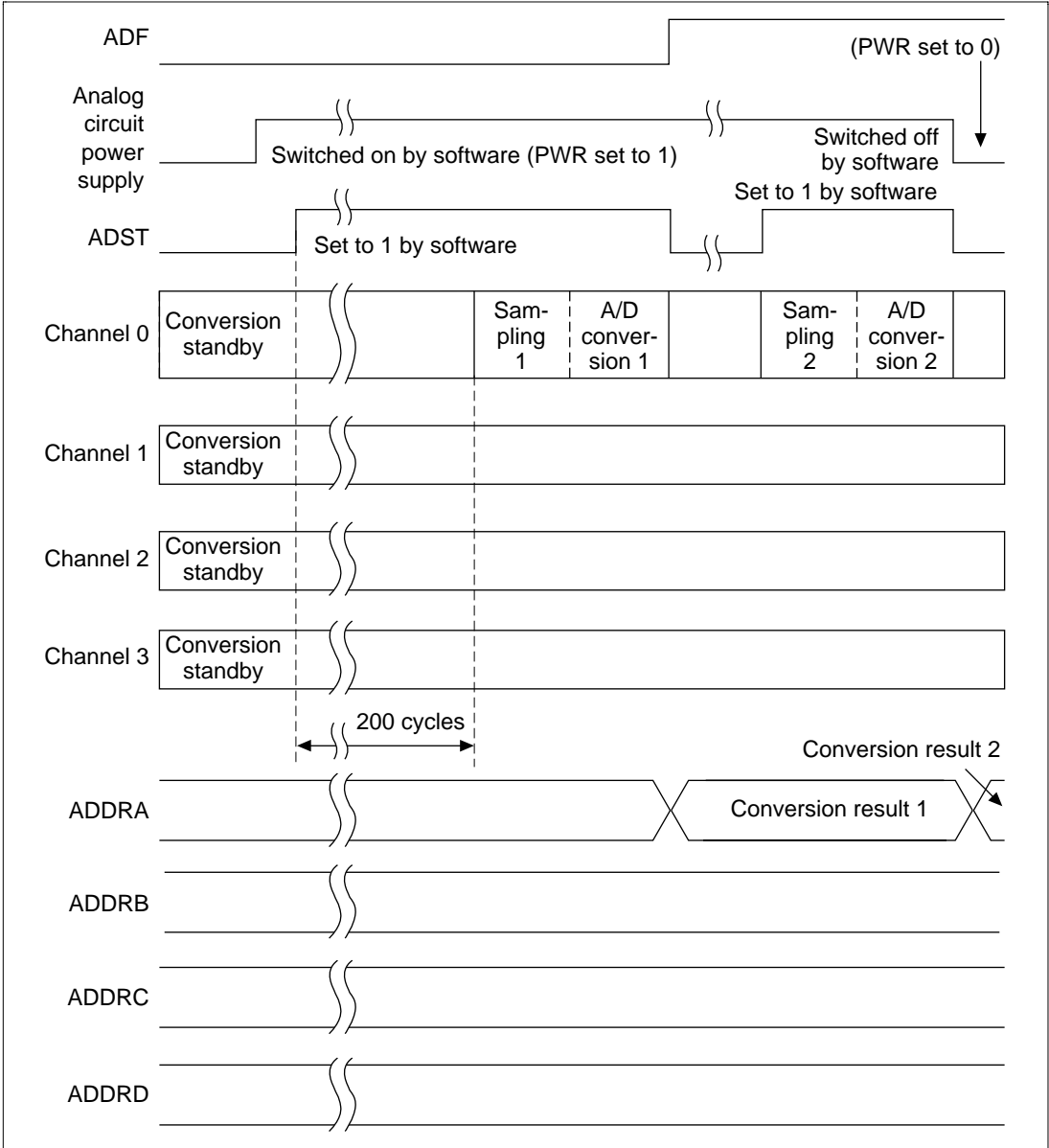


Figure 13.11 Conversion Start Operation (High-Speed Start Mode)

13.4.8 A/D Conversion Time

The high speed A/D converter has an on-chip sample and hold circuit. The high speed A/D converter samples the input at time t_D after the ADST bit is set to 1, and then starts the conversion.

The A/D conversion time t_{CONV} is the sum of the conversion start delay time t_D , the input sampling time t_{SPL} , and the operating time t_{CP} . This conversion time is not a set value, but is decided by the t_D ADCSR write timing, or the timer conversion start trigger generation timing.

Figure 13.12 shows an example of A/D conversion timing. Table 13.7 lists A/D conversion times.

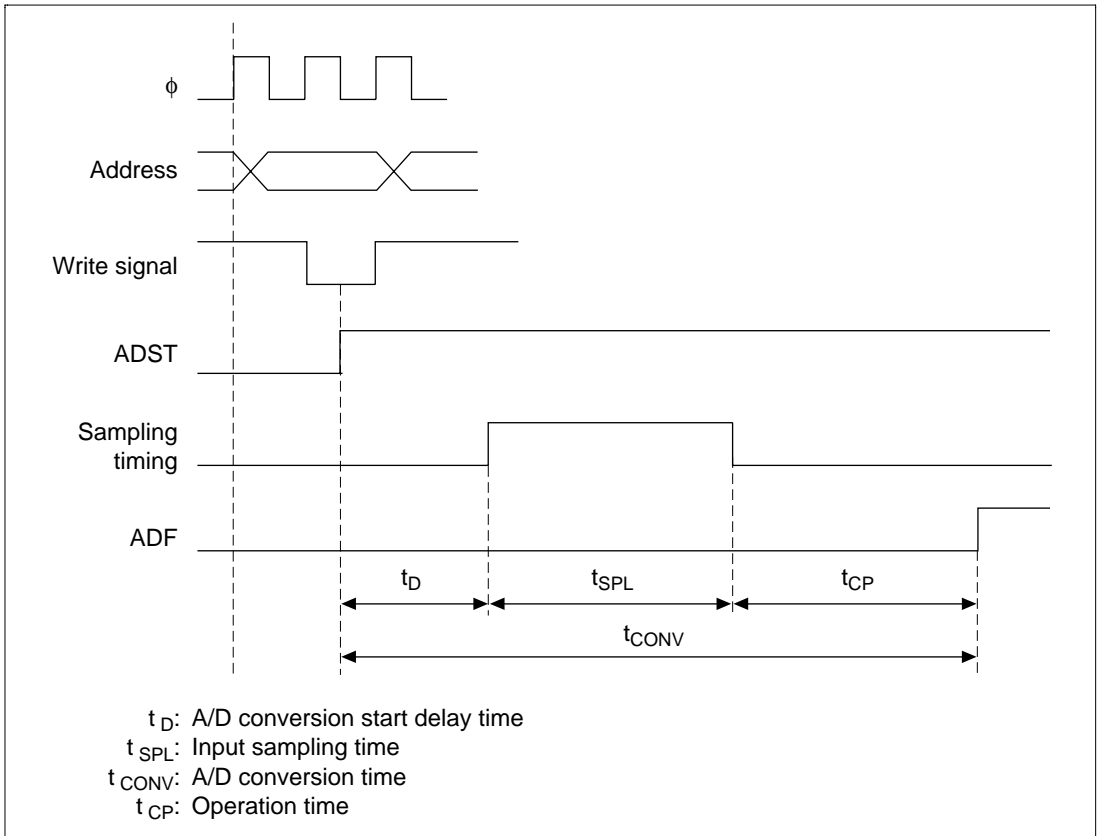


Figure 13.12 A/D Conversion Timing

Table 13.7 A/D Conversion Times

| Time | Symbol | CKS = 0 | | | CKS = 1 | | |
|---------------------------------|------------|---------|------|------|---------|------|------|
| | | Min | Typ | Max | Min | Typ | Max |
| A/D conversion start delay time | t_D | 1.5 | 1.5 | 1.5 | 1.5 | 1.5 | 1.5 |
| Input sampling time | t_{SPL} | 20 | 20 | 20 | 40 | 40 | 40 |
| A/D conversion time | t_{CONV} | 42.5 | 42.5 | 42.5 | 82.5 | 82.5 | 82.5 |

Notes: 1. Unit: states

2. Table entries are for when ADST = 1. If 200 states have not elapsed since the PWR bit has been set, no conversions are done until after those 200 states have occurred. When PWR = 0, add 200 states to the first A/D conversion start delay time. When two or more conversions are performed in succession, tcp is 20 cycles when CKS = 0, and 40 cycles when CKS = 1.

The CKS bit of the ADCSR is the operation time t_{CONV} , but set so that this is 2 μ s or greater. Table 13.8 shows the operating frequency and CKS bit settings.

Table 13.8 Operating Frequency and CKS Bit Settings

| CKS | Conversion Time (States) | Minimum Conversion Time (μ s) | | | | |
|-----|--------------------------|------------------------------------|--------|--------|--------|-------|
| | | 28 MHz | 20 MHz | 16 MHz | 10 MHz | 8 MHz |
| 0 | 42.5 | — | 2.1 | 2.6 | 4.3 | 5.3 |
| 1 | 82.5 | 2.9 | 4.2 | 5.0 | 8.3 | 10.3 |

13.5 Interrupts

The high speed A/D converter generates an A/D conversion end interrupt (ADI) upon completion of A/D conversions. The ADI interrupt request can be enabled or disabled by the ADIE bit of the ADCSR.

The DMAC can be activated by ADI interrupts. When converted data is read by the DMAC upon an ADI interrupt, consecutive conversions can be done without software responsibility.

Table 13.9 lists the high speed A/D converter interrupt sources.

During scan mode, if the ADIE bit is set to 1, A/D conversion is temporarily suspended immediately when the ADF flag is set to 1. A/D conversion is restarted when the ADF flag is cleared to 0.

When the DMAC is activated by an ADI interrupt, the ADF flag is cleared to 0 when the final specified data register is read.

Table 13.9 High Speed A/D Converter Interrupt Sources

| Interrupt Source | Description | DMAC Activation |
|------------------|------------------------------------|-----------------|
| ADI | Interrupt caused by conversion end | Possible |

13.6 Notes on Use

Note the following points concerning the high speed A/D converter.

13.6.1 Analog Input Voltage Range

During A/D conversions, see that the voltage applied to the analog input pins AN0–AN7 is within the range of $AV_{SS} \leq AN0\text{--}AN7 \leq AV_{CC}$.

13.6.2 AV_{CC} , AV_{SS} Input Voltages

The AV_{CC} , AV_{SS} input voltages should be $AV_{CC} = V_{CC} \pm 10\%$, and $AV_{SS} = V_{SS}$. When not using the high speed A/D converter, make $AV_{CC} = V_{CC}$ and $AV_{SS} = V_{SS}$. When in standby mode, make $V_{RAM} \leq AV_{CC} \leq 5.5\text{ V}$, and $AV_{SS} = V_{SS} \cdot V_{RAM}$ is the RAM standby voltage.

13.6.3 Input Ports

Make the time constant of circuitry connected to an input port shorter than the high speed A/D converter sampling time. If the time constant of the circuit is longer, there will be occasions where the input voltage is not adequately sampled.

13.6.4 Conversion Start Modes

Current consumption will be different for the A/D conversion operation in high speed start mode and low-power conversion mode according to the PWR bit setting.

13.6.5 A/D Conversion Termination

If conversion is terminated while in progress (when $ADST = 1$) in high-speed start mode ($PWR = 1$) by clearing the $ADST$ bit, there may be a large degree of error in the first conversion following the restart (on channel 1 in normal mode, or channel 2 in simultaneous sampling mode).

Subsequent conversions will be performed normally

If A/D conversion is terminated while in progress, use one of the following methods to solve this problem:

(a) Ignore the first data following a restart after termination.

- (b) If the first data after a restart is used, perform a single dummy conversion in single mode after the termination, terminate A/D conversion, then restart.
- (c) After termination, write 0 to the PWR bit before restarting. (In this case, the first conversion after the restart will begin after the elapse of 200 A/D reference clock cycles.)

13.6.6 Handling of Analog Input Pins

To prevent damage from surges and other abnormal voltages at the analog input pins (AN0 to AN7), connect a protection circuit such as that shown in figure 13.13. This circuit also includes a CR filter function that suppresses error due to noise. The circuit shown here is only a design example; circuit constants must be decided on the basis of the actual operating conditions.

Figure 13.14 shows an equivalent circuit for the analog input pins, and table 13.10 summarizes the analog input pin specifications.

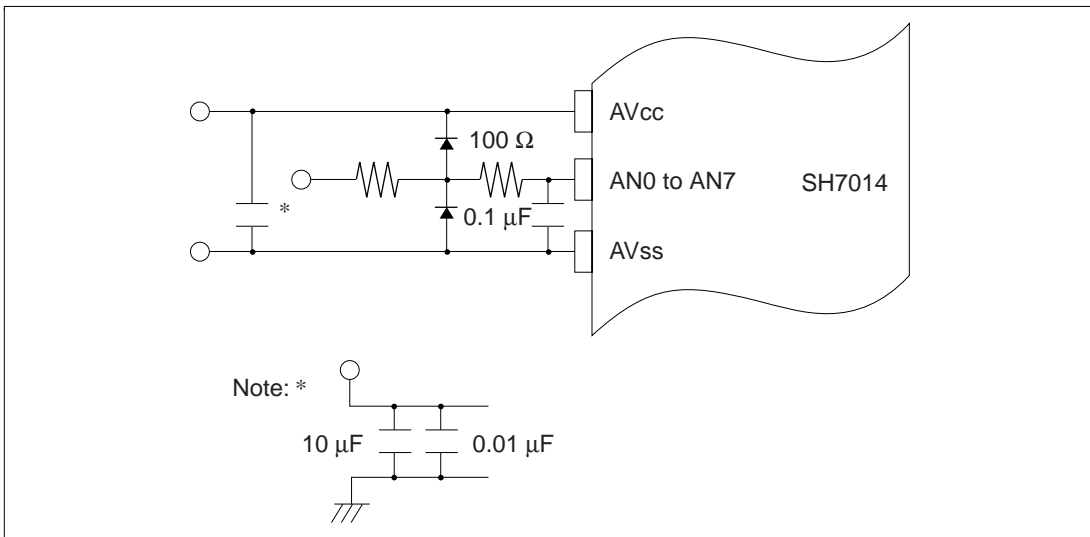
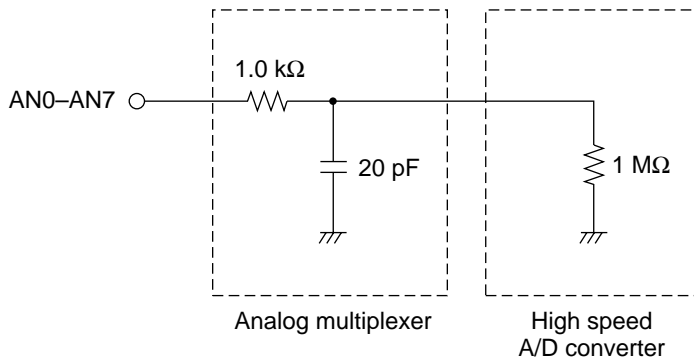


Figure 13.13 Example of Analog Input Pin Protection Circuit



Note: Numeric values are reference values.

Figure 13.14 Analog Input Pin Equivalent Circuit

Table 13.10 Analog Input Pin Specifications

| Item | Min | Max | Unit |
|-------------------------------------|-----|-----|------|
| Analog input capacitance | — | 20 | pF |
| Permissible signal source impedance | — | 1 | kΩ |

Section 14 Mid-Speed A/D Converter –SH7016, SH7017–

14.1 Overview

The mid-speed A/D converter has 10 bit resolution, and can select from a maximum of eight channels of analog input.

14.1.1 Features

The mid-speed A/D converter has the following features:

- 10-bit resolution
- Eight input channels
- Conversion time
 - Minimum conversion time: per channel: 6.7 μ s (20MHz)
- Multiple conversion modes: Single mode/scan mode
 - Single mode: 1 channel A/D conversion
 - 1 to 4 channel simultaneous conversion
- Four 16-bit data registers
 - The results of A/D conversion are transferred to and held in the data register for the relevant channel.
- Sample and hold function
- A/D conversion end interrupt generation
 - An A/D conversion end interrupt (ADI) can be generated on completion of A/D conversion.
- A/D conversion can be started by MTU trigger input.

14.1.2 Block Diagram

Figure 14.1 is the block diagram of the mid-speed A/D converter.

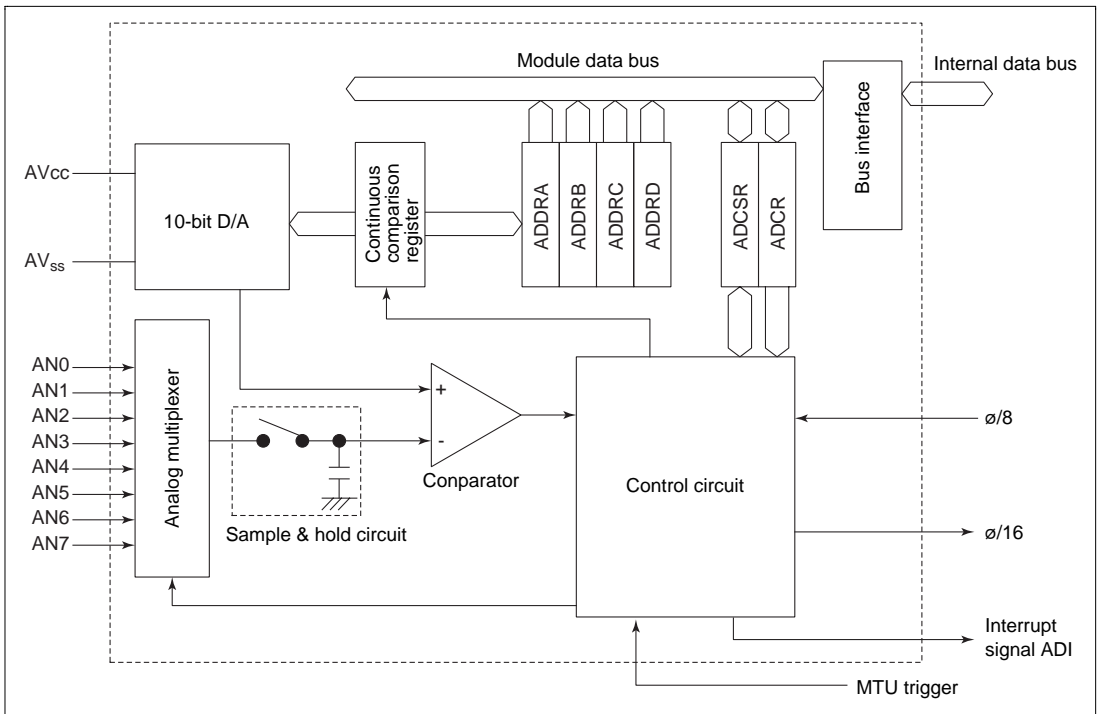


Figure 14.1 Mid-speed A/D converter block diagram

14.1.3 Pin Configuration

Table 14.1 shows the input pins used with the mid-speed A/D converter.

The eight analog input pins are divided into two groups, with analog input pins 0 to 3 (AN0 to AN3) comprising group 0, and analog input pins 4 to 7 (AN4 to AN7) comprising group 1.

Table 14.1 Pin Configuration

| Pin | Abbreviation | I/O | Function |
|----------------|---------------------|------------|---|
| Analog supply | $A_{V_{CC}}$ | I | Analog section power supply |
| Analog ground | $A_{V_{SS}}$ | I | Analog section ground and A/D conversion standard voltage |
| Analog input 0 | AN0 | I | Analog input group 0 |
| Analog input 1 | AN1 | I | |
| Analog input 2 | AN2 | I | |
| Analog input 3 | AN3 | I | |
| Analog input 4 | AN4 | I | Analog input group 1 |
| Analog input 5 | AN5 | I | |
| Analog input 6 | AN6 | I | |
| Analog input 7 | AN7 | I | |

14.1.4 Register Configuration

Table 14.2 shows the register configuration of the mid-speed A/D converter.

Table 14.2 Register Configuration

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|-----------------------------|---------------------|------------|----------------------|----------------|--------------------|
| A/D data register AH | ADDRAH | R | H'00 | H'FFFF8420 | 8, 16 |
| A/D data register AL | ADDRAL | R | H'00 | H'FFFF8421 | 8 |
| A/D data register BH | ADDRBH | R | H'00 | H'FFFF8422 | 8, 16 |
| A/D data register BL | ADDRBL | R | H'00 | H'FFFF8423 | 8 |
| A/D data register CH | ADDRCH | R | H'00 | H'FFFF8424 | 8, 16 |
| A/D data register CL | ADDRCL | R | H'00 | H'FFFF8425 | 8 |
| A/D data register DH | ADDRDH | R | H'00 | H'FFFF8426 | 8, 16 |
| A/D data register DL | ADDRDL | R | H'00 | H'FFFF8427 | 8 |
| A/D control/status register | ADCSR | R/(W)* | H'00 | H'FFFF8428 | 8, 16 |
| A/D control register | ADCR | R/W | H'7F | H'FFFF8429 | 8, 16 |

Note: Only 0 can be written to bit 7 to clear the flag.

14.2 Register Descriptions

14.2.1 A/D Data Register A to D (ADDRA to ADDRD)

A/D registers are special registers that read stored results of A/D conversion in 16 bits. There are eight registers: ADDRA to ADDRD.

The A/D converted data is 10 bit data which is to the ADDR of the corresponding converted channel for storage. The upper 8 bits of the A/D converted data correspond to the upper byte of the ADDR and the lower 2 bits correspond to the lower byte. Bits 5 to 0 of the lower byte of ADDR are reserved and always read 0. Analog input channels and correspondence to ADDR are shown in table 14.3.

ADDR can always be read from the CPU. The upper byte may be read directly. The lower byte is transferred through the temporary register (TEMP). For details, see 14.3 Interface with CPU.

ADDR is initialized to H'0000 during power-on reset.

| | | | | | | | | | | | | | | | | |
|-----------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|---|---|---|---|---|
| Bit : | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADDRn : | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | — | — | — | — | — | — |
| 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 |

(n=A to D)

Table 14.3 Analog Input Channel and ADDRA to ADDRD Correspondence

| Analog Input Channel | | |
|----------------------|---------|-------------------|
| Group | Group 1 | A/D Data Register |
| AN0 | AN4 | ADDRA |
| AN1 | AN5 | ADDRB |
| AN2 | AN6 | ADDRC |
| AN3 | AN7 | ADDRD |

14.2.2 A/D Control/Status Register (ADCSR)

The A/D control/status register (ADCSR) is register that can read/write in 8 bits and control mid-speed A/D converter operations such as mode selection.

The ADCSR is initialized to H'00 during power-on reset.

| | | | | | | | | |
|-----------------|--------|------|------|------|-----|-----|-----|-----|
| Bit : | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | ADF | ADIE | ADST | SCAN | CKS | CH2 | CH1 | CH0 |
| Initial value : | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 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.

- Bit 7—A/D End Flag (ADF): Status flag that indicates end of A/D conversion.

Bit 7:

| ADF | Description |
|-----|-------------|
|-----|-------------|

| | |
|---|---|
| 0 | [Clear conditions] (Initial value) 1. Writing 0 to ADF after reading ADF with ADF=1 2. When registers of the mid-speed converter are accessed after the DMAC is activated by ADI interrupt. |
| 1 | [Set conditions] 1. Single mode: When A/D conversion is complete 2. Scan mode: When A/D conversion of all designated channels are complete |

- Bit 6—A/D Interrupt Enable (ADIE): Enables or disables interrupt request (ADI) due to completion of A/D conversion.

Bit 6:

| ADIE | Description |
|------|-------------|
|------|-------------|

| | |
|---|--|
| 0 | Disables interrupt request (ADI) due to completion of A/D conversion (Initial value) |
| 1 | Enables interrupt request (ADI) due to completion of A/D conversion |

- Bit 5—A/D Start (ADST): Selects start/end of A/D conversion. A 1 is maintained during A/D conversion start. It is also possible to set a 1 by the MTU trigger input.

Bit 5:

| ADST | Description | |
|------|---|-----------------|
| 0 | A/D conversion halted | (Initial value) |
| 1 | 1. Single mode: Starts A/D conversion. Automatically clears to 0 when conversion of the designated channel is complete 2. Scan mode: Starts A/D conversion. Continuous conversion until cleared to 0 by the software | |

- Bit 4—Scan Mode (SCAN): Selects the A/D conversion mode from single mode and scan mode. For operations during single/scan mode, see 14.4 Operation. When switching modes, proceed while ADST=0.

Bit 4:

| SCAN | Description | |
|------|-------------|-----------------|
| 0 | Single mode | (Initial value) |
| 1 | Scan mode | |

- Bit 3—Clock Select (CKS): Sets the A/D conversion time. Proceed conversion time switch while ADST=0. Always set CKS=0 when operating frequency exceeds 20MHz.

Bit 3:

| CKS | Description | |
|-----|------------------------------------|-----------------|
| 0 | Conversion time = 266 states (max) | (Initial value) |
| 1 | Conversion time = 134 states (max) | |

- Bits 2–0—Channel select 2–0 (CH2–CH0): Selects the analog input channel along with the SCAN bit. Switch channels while ADST=0.

| Group Selection | Channel Selection | | Description | |
|------------------------|--------------------------|------------|---------------------|---------------------|
| CH2 | CH1 | CH0 | Single Mode | Scan Mode |
| 0 | 0 | 0 | AN0 (Initial value) | AN0 (Initial value) |
| | | 1 | AN1 | AN0, AN1 |
| | 1 | 0 | AN2 | AN0 to AN2 |
| | | 1 | AN3 | AN0 to AN3 |
| 1 | 0 | 0 | AN4 | AN4 |
| | | 1 | AN5 | AN4, AN5 |
| | 1 | 0 | AN6 | AN4 to AN6 |
| | | 1 | AN7 | AN4 to AN7 |

14.2.3 A/D Control Register (ADCR)

A/D control register (ADCR) is register that can read/write in 8 bits and enables or disables A/D conversion start of the MTU trigger input.

ADCR is initialized to H'7F during power-on reset.

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

- Bit 7—Trigger Enable (TRGE): Enables or disables A/D conversion start of input from MTU trigger.

Bit 7:

| TRGE | Description |
|------|--|
| 0 | Disables A/D conversion start of MTU trigger (Initial value) |
| 1 | Starts A/D conversion on MTU trigger. |

- Bits 6 to 0—Reserved bits: These bits always read as 1. The write value should always be 1.

14.3 Interface with CPU

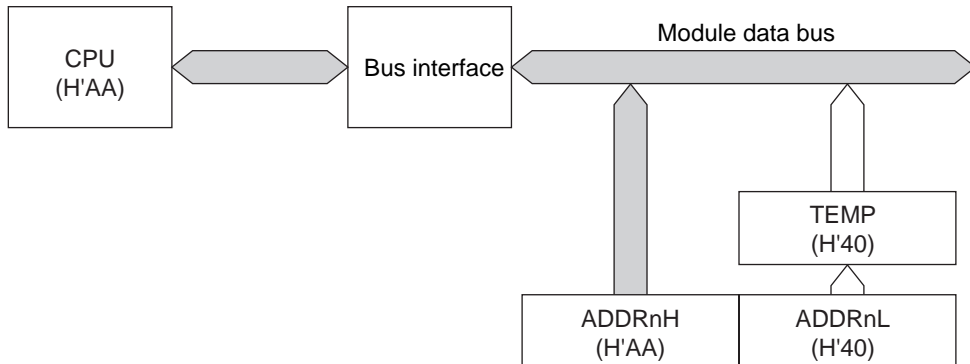
Although A/D data register ADDRA–ADDRD are 16-bit registers, the bus width within the chip that integrates with the CPU is 8-bits. So, the lower byte data is read through the temporary register (TEMP). The upper byte data can be read directly.

The procedure for reading data from ADDR is as follows: First, read the upper byte data from ADDR. At this time, the upper byte data is read directly into the CPU and the lower byte data is transferred to TEMP of the mid-speed A/D converter. Next, read the lower byte to read the TEMP contents into the CPU.

When reading the ADDR in byte size, read the upper byte before the lower byte. Furthermore, it is possible to read only the upper byte, however, please note that contents are not guaranteed when reading only the lower byte. In this case, ADDR can be read from the upper-byte address with a word transfer instruction (such as MOV.W).

Figure 14.2 shows the data flow when reading from ADDR.

<Reading the upper byte>



<Reading of the lower byte>

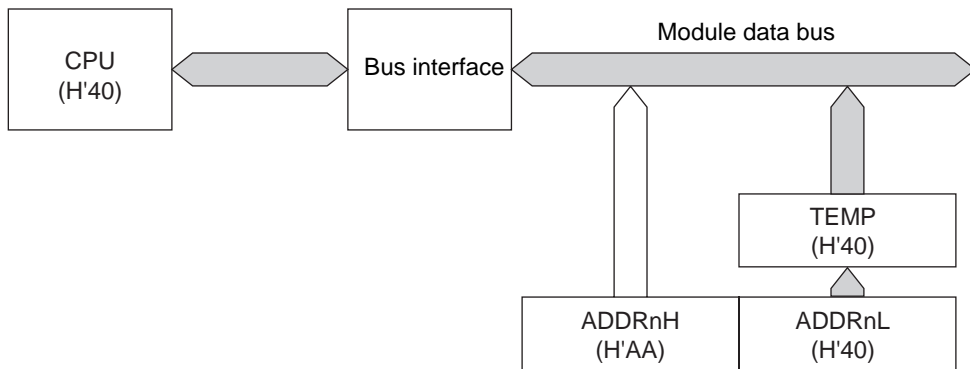


Figure 14.2 ADDR Access Operation (During Reading of (H'AA40))

14.4 Operation

The mid-speed converter operates using the continuous comparison method and is equipped with 10-bit resolution. Operations for the single and scan modes are explained below.

14.4.1 Single Mode (SCAN=0)

The single mode is selected when executing A/D conversion for one channel only. A/D conversion is initiated when the ADST bit of the A/D control/status register is set to 1 by the software or external trigger input. The ADST bit is held to 1 during the A/D conversion and is automatically cleared to 0 upon completion.

When conversion is complete, the ADF bit of ADCSR is set to 1. At this time, if the ADIE bit of ADCSR is 1, ADI interrupt request occurs.

The ADF bit can be cleared by writing 0 after reading ADF=1.

To switch modes or analog input channels during A/D conversion, clear the ADST bit to 0 and stop A/D conversion to avoid malfunction. After switching (mode/channel change and ADST bit setting can be made at the same time), set the ADST bit to 1 to restart A/D conversion.

An example of operation when channel 1 (AN1) is selected in the single mode is shown in figure 14.3.

1. Set operation mode to single mode (SCAN=0), input channel to AN1 (CH2=CH1=0, CH0=1) and A/D interrupt request to enable (ADIE=1) then start A/D conversion (ADST=1).
2. When A/D conversion is complete, A/D conversion result is transferred to ADDR0. At the same time, ADF=1 will become ADF=0 and the mid-speed converter will standby for conversion.
3. Since ADF=1 and ADIE=1, ADI interrupt request will occur.
4. The A/D interrupt process routine will start.
5. After reading ADF=1, write 0 to ADF.
6. Read the A/D conversion result (ADDR0) and process.
7. End A/D interrupt process routine execution. When ADST bit is set to 1, A/D conversion starts, following steps (2) to (7) above.

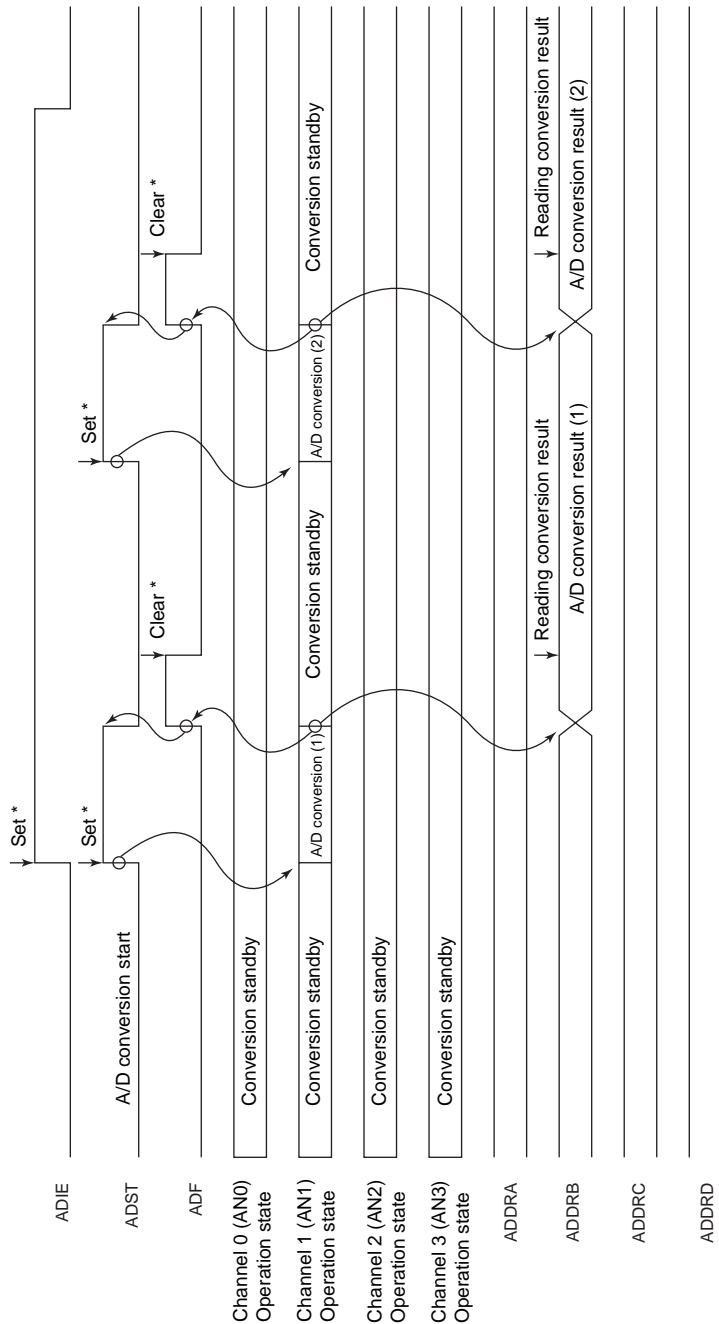


Figure 14.3 Operation Example of Mid-speed A/D Converter (Single Mode, Channel 1 Selected)

14.4.2 Scan Mode (SCAN=1)

The scan mode is optimal for monitoring analog input of multiple channels (including channel 1). A/D conversion is started from channel 1 (AN0 for CH2=0 and AN4 for CH1=1) of the group when the ADST bit of the A/D control/status register (ADCSR) is set to 1 by the software or external trigger input.

When multiple channels are selected, A/D conversion of channel 2 (AN1 or AN5) is initiated immediately after completion of the channel 1 conversion.

A/D conversion is performed repeatedly on the selected channels until the ADST bit is cleared to 0. The results of conversion are transferred to and held in the ADDR registers for the relevant channels.

To switch modes or analog input channels during A/D conversion, clear the ADST bit to 0 and stop A/D conversion to avoid malfunction. After switching (mode/channel change and ADST bit setting can be made at the same time), set ADST bit to 1 to restart A/D conversion from channel 1.

An example of operation when three channels of A/D0 (AN0 to 2) are selected for A/D conversion is shown in figure 14.4.

1. Set operation mode to scan mode (SCAN=1), set analog channels to AN0–2 (CH1=1, CH0=0) then start A/D conversion (ADST=1).
2. When A/D conversion for channel 1 (AN0) is complete, A/D conversion result is transferred to ADDR0.

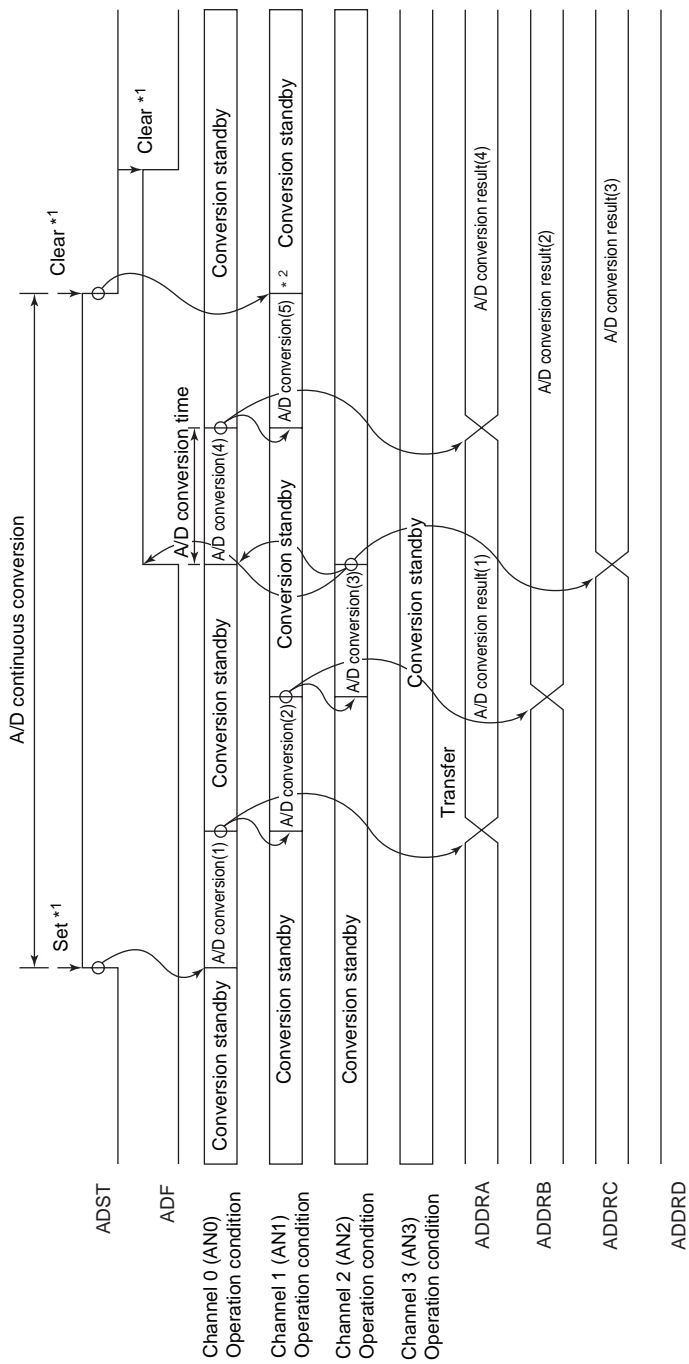
Next, channel 2 (AN1) will automatically be selected and conversion will begin.

3. In the same manner, channel 3 (AN2) will be converted.
4. When conversion of all of the selected channels (AN0 to AN2) are complete, ADF will become 1 and channel 1 (AN0) will again be selected and conversion will begin.

At this time, if the ADIE bit is set to 1, ADI interrupt request will occur after completing A/D conversion.

5. Steps (2) to (4) will be repeated while ADST bit is set to 1.

A/D conversion will stop when setting the ADST bit to 0. When setting the ADST bit to 1, A/D conversion will start again from channel 1 (AN0).



- Note: 1. ↓ indicates command execution by the software
 2. Data is ignored during conversion

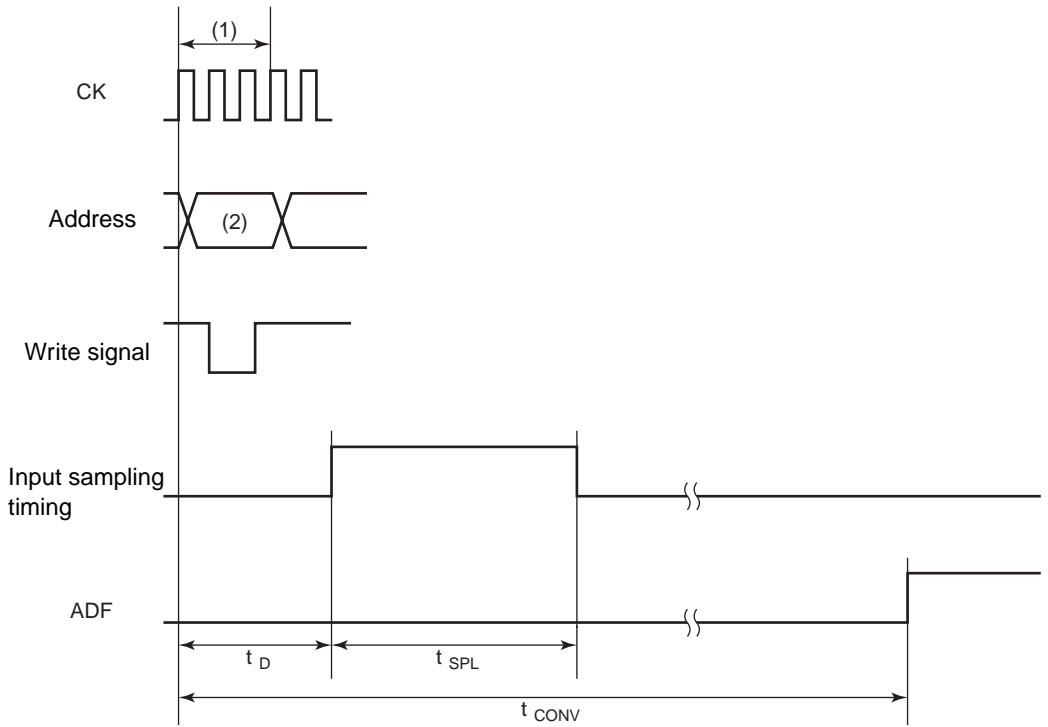
Figure 14.4 Operation Example of Mid-speed A/D Converter (Scan Mode, Three Channels Selected) (AN0 to AN2)

14.4.3 Input Sampling and A/D Conversion Time

The mid-speed A/D converter is equipped with a sample and hold circuit. The mid-speed A/D converter samples input after t_p hours has elapsed since setting the ADST bit of the A/D control/status register (ADCSR) to 1, then begins conversion. The A/D conversion timing is shown in table 14.4.

The A/D conversion time, as shown in figure 14.5, includes both t_p and input sampling time. Here, t_p is determined by the write timing to ADCSR and is not constant. Thus the conversion time changes in the range shown in table 14.4.

The conversion time shown in table 14.4 is the time for the first conversion. For the second conversion and after, the time will be 256 state (fixed) for CKS=0 and 128 state (fixed) for CKS=1.



- (1): Write cycle of ADCSR
- (2): Address of ADCSR
- t_D : A/D conversion start delay time
- t_{SPL} : Input sampling time
- t_{CONV} : A/D conversion time

Figure 14.5 A/D Conversion Timing

Table 14.4 A/D Conversion Time (Single Mode)

| | Notation | CKS=0 | | | CKS=1 | | |
|---------------------------------------|------------|-------|-----|-----|-------|-----|-----|
| | | Min | Typ | Max | Min | Typ | Max |
| A/D conversion start delay time t_D | | 10 | — | 17 | 6 | — | 9 |
| Input sampling time | t_{SPL} | — | 64 | — | — | 32 | — |
| A/D conversion time | t_{CCNV} | 259 | — | 266 | 131 | — | 134 |

Note: Numbers in the table are in states (t_{cyc}).

14.4.4 MTU Trigger Input Timing

It is possible to start A/D conversion from an MTU trigger input. MTU trigger input is input from MTU when the TRGE bit of the A/D control register (ADCR) is set to 1.

A/D conversion is started when the ADST bit of the A/D control/status register (ADCSR) is set to 1 by MTU trigger. Other operations, regardless of whether in the single or scan mode, are the same as when setting the ADST bit to 1 with the software.

Figure 14.6 shows an example of external trigger input timing.

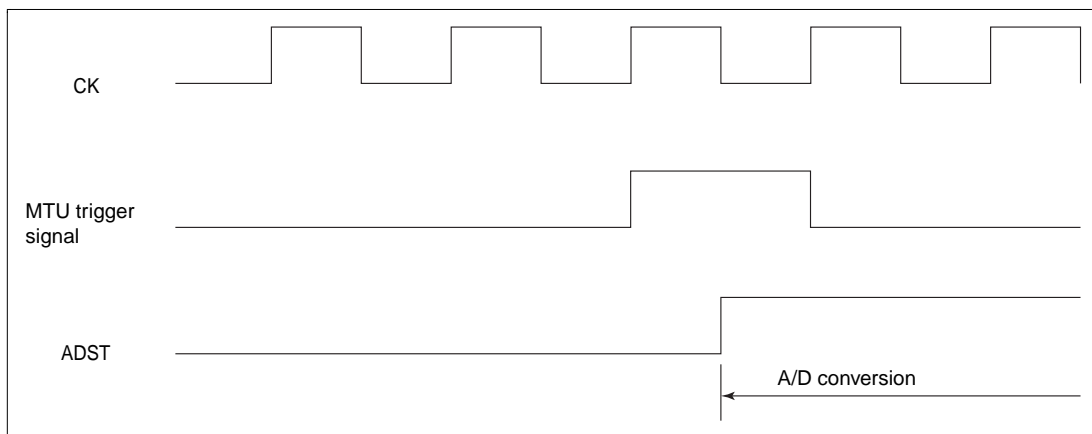


Figure 14.6 External Trigger Input Timing

14.5 Interrupt

The mid-speed A/D converter generates an A/D conversion end interrupt (ADI) on completion of A/D conversion. ADI interrupt requests can be enabled or disabled by means of the ADIE bit in ADCSR.

The DMAC can be activated by an ADI interrupt. Having converted data read by the DMAC by means of ADI interrupts allows continuous conversion to be carried out without imposing a load on the software.

The mid-speed A/D converter's interrupt source is summarized in table 14.5.

In scan mode, if the ADIE bit is set to 1 beforehand, A/D conversion will be temporarily halted when the ADF flag is set to 1, and resumed when the ADF flag is cleared to 0.

When the DMAC is activated by an ADI interrupt, the ADF flag is cleared to 0 when the last of the specified data registers is read.

Table 14.5 Mid-speed A/D converter interrupt factors

| Interrupt Factor | Content | DMAC |
|-------------------------|----------------------------------|--------------------|
| ADI | Interrupt by conversion complete | Activation enabled |

14.6 A/D Conversion Precision Definitions

The mid-speed A/D converter converts analog values input from analog input channels to 10-bit digital values by comparing them with an analog reference voltage. In this operation, the absolute precision of the A/D conversion (i.e. the deviation between the input analog value and the output digital value) includes the following kinds of error.

- (1) Offset error
- (2) Full-scale error
- (3) Quantization error
- (4) Nonlinearity error

The above four kinds of error are described below with reference to figure 14.7. For the sake of clarity, this figure shows 3-bit mid-speed A/D conversion rather than 10-bit mid-speed A/D conversion. Offset error (see figure 14.7 (1)) is the deviation between the actual A/D conversion characteristic and the ideal A/D conversion characteristic when the digital output value changes from the minimum value (zero voltage) of 0000000000 (000 in the figure) to 0000000001 (001 in the figure). Full-scale error (see figure 14.7 (2)) is the deviation between the actual A/D conversion characteristic and the ideal A/D conversion characteristic when the digital output value changes from 1111111110 (110 in the figure) to the maximum value (full-scale voltage) of 1111111111 (111 in the figure). Quantization error is the deviation inherent in the mid-speed A/D converter, given by $1/2$ LSB (see figure 14.7 (3)). Nonlinearity error is the deviation between the actual A/D conversion characteristic and the ideal A/D conversion characteristic from zero voltage to full-scale voltage (see figure 14.7 (4)). This does not include offset error, full-scale error, and quantization error.

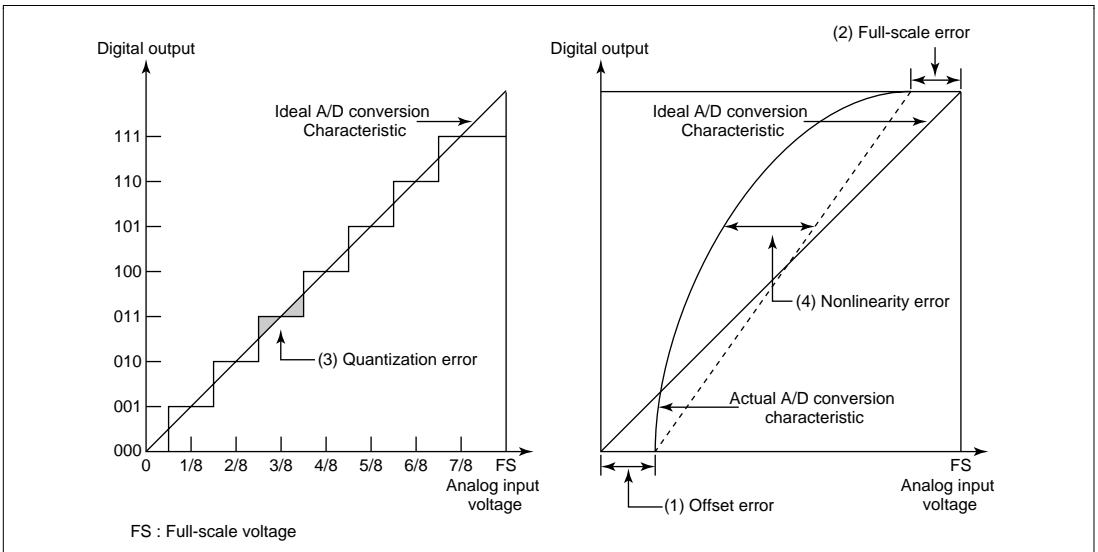


Figure 14.7 A/D Conversion Precision Definitions

14.7 Usage Notes

The following points should be noted when using the mid-speed A/D converter.

14.7.1 Analog Voltage Settings

(1) Analog input voltage range

The voltage applied to analog input pins during A/D conversion should be in the range $AV_{SS} \leq AN_n \leq AV_{CC}$ ($n = 0$ to 7).

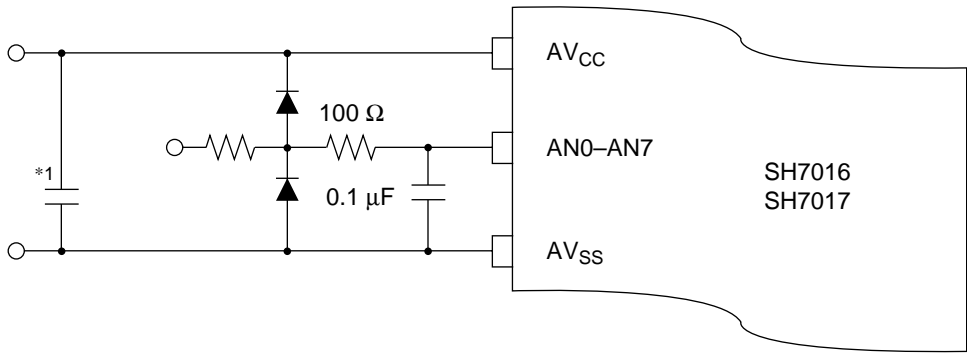
(2) AV_{CC} and AV_{SS} input voltages

For the AV_{CC} and AV_{SS} input voltages, set $AV_{CC} = V_{CC} = 5V \pm 10\%$ and $AV_{SS} = V_{SS}$.
When the mid-speed A/D converter is not used, set $AV_{CC} = V_{CC}$ and $AV_{SS} = V_{SS}$.

14.7.2 Handling of Analog Input Pins

To prevent damage from surges and other abnormal voltages at the analog input pins (AN0-AN7), connect a protection circuit such as that shown in figure 14.8. This circuit also includes a CR filter function that suppresses error due to noise. The circuit shown here is only a design example; circuit constants must be decided on the basis of the actual operating conditions.

Figure 14.9 shows an equivalent circuit for the analog input pins, and table 14.6 summarizes the analog input pin specifications.



Note:

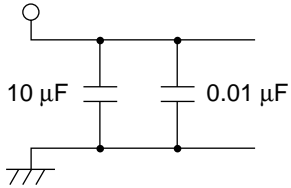


Figure 14.8 Example of Analog Input Pin Protection Circuit

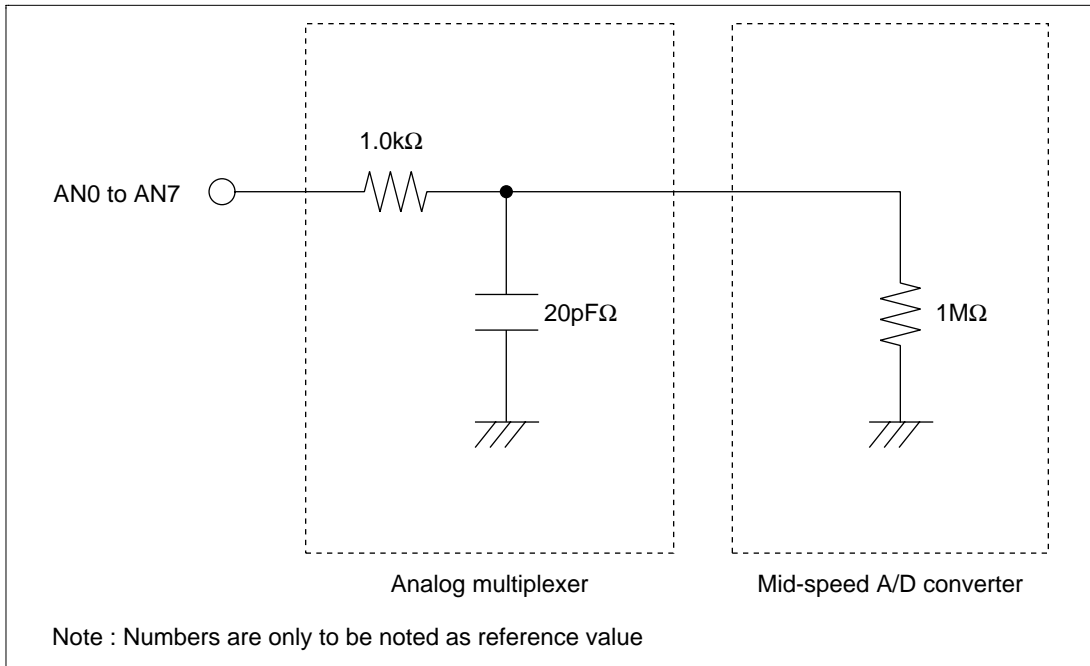


Figure 14.9 Equivalent Circuit for the Analog Input Pins

Table 14.6 Analog Pin Specifications

| Item | Min | Max | Unit |
|-------------------------------------|-----|-----|------|
| Analog input capacitance | — | 20 | pF |
| Permissible signal source impedance | — | 1 | kΩ |

Section 15 Compare Match Timer (CMT)

15.1 Overview

This LSI has an on-chip compare match timer (CMT) configured of 16-bit timers for two channels. The CMT has 16-bit counters and can generate interrupts at set intervals.

15.1.1 Features

The CMT has the following features:

- Four types of counter input clock can be selected
 - One of four internal clocks ($\phi/8$, $\phi/32$, $\phi/128$, $\phi/512$) can be selected independently for each channel.
- Interrupt sources
 - A compare match interrupt can be requested independently for each channel.

15.1.2 Block Diagram

Figure 15.1 shows a block diagram of the CMT.

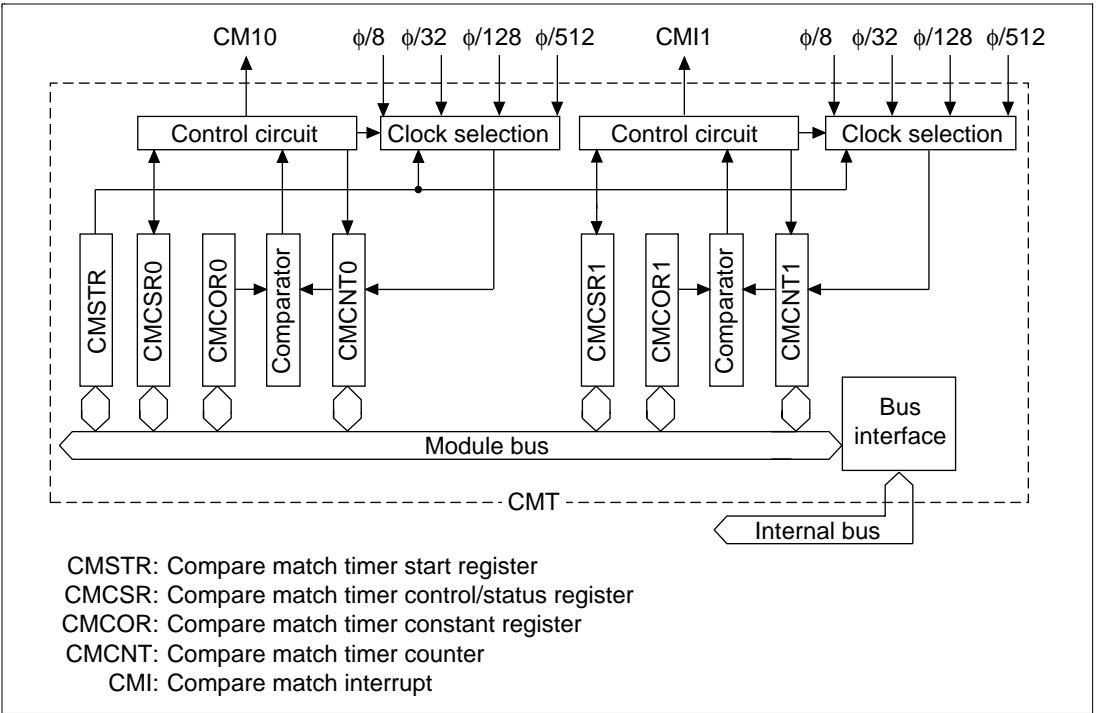


Figure 15.1 CMT Block Diagram

15.1.3 Register Configuration

Table 15.1 summarizes the CMT register configuration.

Table 15.1 Register Configuration

| Channel Name | Abbreviation | R/W | Initial Value | Address | Access Size (Bits) |
|--------------|---|--------|---------------|---------|----------------------|
| Shared | Compare match timer start register | CMSTR | R/W | H'0000 | H'FFFF83D0 8, 16, 32 |
| 0 | Compare match timer control/status register 0 | CMCSR0 | R/(W)* | H'0000 | H'FFFF83D2 8, 16, 32 |
| | Compare match timer counter 0 | CMCNT0 | R/W | H'0000 | H'FFFF83D4 8, 16, 32 |
| | Compare match timer constant register 0 | CMCOR0 | R/W | H'FFFF | H'FFFF83D6 8, 16, 32 |
| 1 | Compare match timer control/status register 1 | CMCSR1 | R/(W)* | H'0000 | H'FFFF83D8 8, 16, 32 |
| | Compare match timer counter 1 | CMCNT1 | R/W | H'0000 | H'FFFF83DA 8, 16, 32 |
| | Compare match timer constant register 1 | CMCOR1 | R/W | H'FFFF | H'FFFF83DC 8, 16, 32 |

Note: The only value that can be written to the CMCSR0 and CMCSR1 CMF bits is a 0 to clear the flags.

15.2 Register Descriptions

15.2.1 Compare Match Timer Start Register (CMSTR)

The compare match timer start register (CMSTR) is a 16-bit register that selects whether to operate or halt the channel 0 and channel 1 counters (CMCNT). It is initialized to H'0000 by power-on resets and by standby mode.

| | | | | | | | | |
|----------------|----|----|----|----|----|----|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | — | STR1 | STR0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R/W | R/W |

- Bits 15–2—Reserved: These bits always read as 0. The write value should always be 0.
- Bit 1—Count Start 1 (STR1): Selects whether to operate or halt compare match timer counter 1.

| Bit 1: STR1 | Description |
|-------------|---|
| 0 | CMCNT1 count operation halted (initial value) |
| 1 | CMCNT1 count operation |

- Bit 0—Count Start 0 (STR0): Selects whether to operate or halt compare match timer counter 0.

| Bit 0: STR0 | Description |
|-------------|---|
| 0 | CMCNT0 count operation halted (initial value) |
| 1 | CMCNT0 count operation |

15.2.2 Compare Match Timer Control/Status Register (CMCSR)

The compare match timer control/status register (CMCSR) is a 16-bit register that indicates the occurrence of compare matches, sets the enable/disable of interrupts, and establishes the clock used for incrementation. It is initialized to H'0000 by power-on resets and by standby mode.

| | | | | | | | | |
|----------------|--------|------|----|----|----|----|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| | | | | | | | | |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | CMF | CMIE | — | — | — | — | CKS1 | CKS0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/(W)* | R/W | R | R | R | R | R/W | R/W |

Note: The only value that can be written is a 0 to clear the flag.

- Bits 15–8 and 5–2—Reserved: These bits always read as 0. The write value should always be 0.
- Bit 7—Compare Match Flag (CMF): This flag indicates whether or not the CMCNT and CMCOR values have matched.

| Bit 7: CMF | Description |
|------------|---|
| 0 | CMCNT and CMCOR values have not matched (initial status) Clear condition: Write a 0 to CMF after reading a 1 from it |
| 1 | CMCNT and CMCOR values have matched |

- Bit 6—Compare Match Interrupt Enable (CMIE): Selects whether to enable or disable a compare match interrupt (CMI) when the CMCNT and CMCOR values have matched (CMF = 1).

| Bit 6: CMIE | Description |
|-------------|--|
| 0 | Compare match interrupts (CMI) disabled (initial status) |
| 1 | Compare match interrupts (CMI) enabled |

- Bits 1, 0—Clock Select 1, 0 (CKS1, CKS0): These bits select the clock input to the CMCNT from among the four internal clocks obtained by dividing the system clock (ϕ). When the STR bit of the CMSTR is set to 1, the CMCNT begins incrementing with the clock selected by CKS1 and CKS0.

| Bit 1: CKS1 | Bit 0: CKS0 | Description |
|-------------|-------------|---------------------------|
| 0 | 0 | $\phi/8$ (initial status) |
| | 1 | $\phi/32$ |
| 1 | 0 | $\phi/128$ |
| | 1 | $\phi/512$ |

15.2.3 Compare Match Timer Counter (CMCNT)

The compare match timer counter (CMCNT) is a 16-bit register used as an upcounter for generating interrupt requests.

When an internal clock is selected with the CKS1, CKS0 bits of the CMCSR register and the STR bit of the CMSTR is set to 1, the CMCNT begins incrementing with that clock. When the CMCNT value matches that of the compare match timer constant register (CMCOR), the CMCNT is cleared to H'0000 and the CMF flag of the CMCSR is set to 1. If the CMIE bit of the CMCSR is set to 1 at this time, a compare match interrupt (CMI) is requested.

The CMCNT is initialized to H'0000 by power-on resets and by standby mode.

| | | | | | | | | |
|----------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> |
| 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 |
| | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> | <input type="text"/> |
| 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 |

15.2.4 Compare Match Timer Constant Register (CMCOR)

The compare match timer constant register (CMCOR) is a 16-bit register that sets the compare match period with the CMCNT.

The CMCOR is initialized to H'FFFF by power-on resets and by standby mode.

| | | | | | | | | |
|----------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | <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 | 1 | 1 | 1 | 1 |
| 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 |
| | <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 | 1 | 1 | 1 | 1 |
| R/W: | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

15.3 Operation

15.3.1 Period Count Operation

When an internal clock is selected with the CKS1, CKS0 bits of the CMCSR register and the STR bit of the CMSTR is set to 1, the CMCNT begins incrementing with the selected clock. When the CMCNT counter value matches that of the compare match constant register (CMCOR), the CMCNT counter is cleared to H'0000 and the CMF flag of the CMCSR register is set to 1. If the CMIE bit of the CMCSR register is set to 1 at this time, a compare match interrupt (CMI) is requested. The CMCNT counter begins counting up again from H'0000.

Figure 15.2 shows the compare match counter operation.

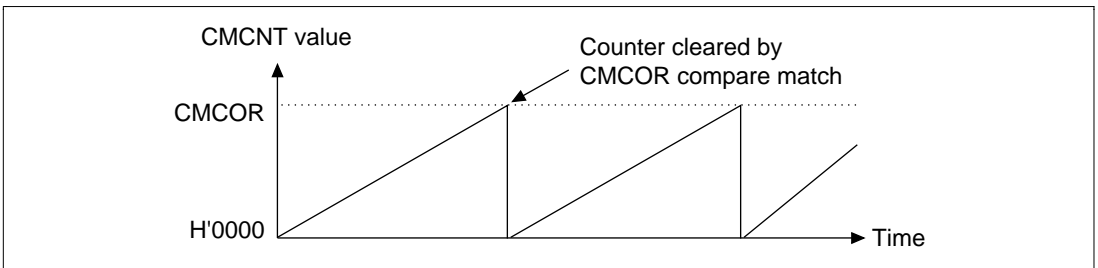


Figure 15.2 Counter Operation

15.3.2 CMCNT Count Timing

One of four clocks ($\phi/8$, $\phi/32$, $\phi/128$, $\phi/512$) obtained by dividing the system clock (CK) can be selected by the CKS1, CKS0 bits of the CMCSR. Figure 15.3 shows the timing.

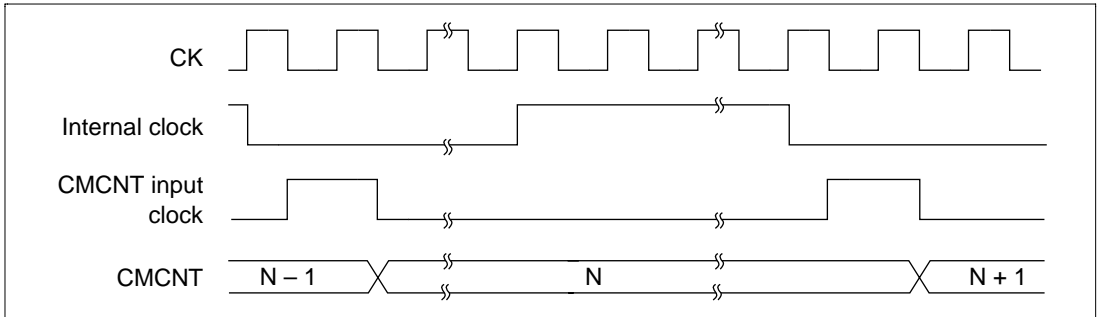


Figure 15.3 Count Timing

15.4 Interrupts

15.4.1 Interrupt Sources and DTC Activation

The CMT has a compare match interrupt for each channel, with independent vector addresses allocated to each of them. The corresponding interrupt request is output when the interrupt request flag CMF is set to 1 and the interrupt enable bit CMIE has also been set to 1.

When activating CPU interrupts by interrupt request, the priority between the channels can be changed by using the interrupt controller settings. See section 6, Interrupt Controller, for details.

15.4.2 Compare Match Flag Set Timing

The CMF bit of the CMCSR register is set to 1 by the compare match signal generated when the CMCOR register and the CMCNT counter match. The compare match signal is generated upon the final state of the match (timing at which the CMCNT counter matching count value is updated). Consequently, after the CMCOR register and the CMCNT counter match, a compare match signal will not be generated until a CMCNT counter input clock occurs. Figure 15.4 shows the CMF bit set timing.

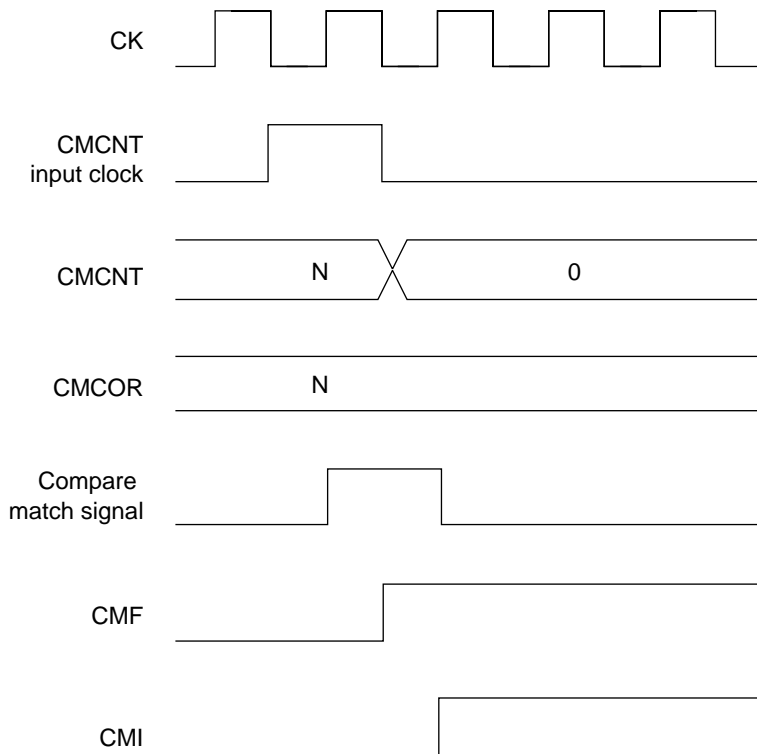


Figure 15.4 CMF Set Timing

15.4.3 Compare Match Flag Clear Timing

The CMF bit of the CMCSR register is cleared either by writing a 0 to it after reading a 1, or by a clear signal after a DTC transfer. Figure 15.5 shows the timing when the CMF bit is cleared by the CPU.

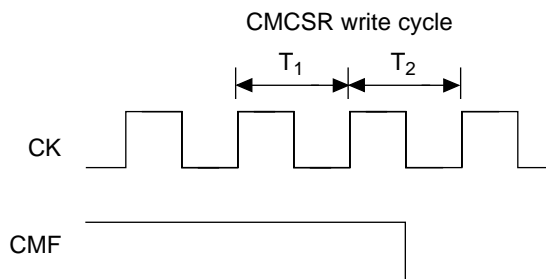


Figure 15.5 Timing of CMF Clear by the CPU

15.5 Notes on Use

Take care that the contentions described in sections 15.5.1–15.5.3 do not arise during CMT operation.

15.5.1 Contention between CMCNT Write and Compare Match

If a compare match signal is generated during the T_2 state of the CMCNT counter write cycle, the CMCNT counter clear has priority, so the write to the CMCNT counter is not performed. Figure 15.6 shows the timing.

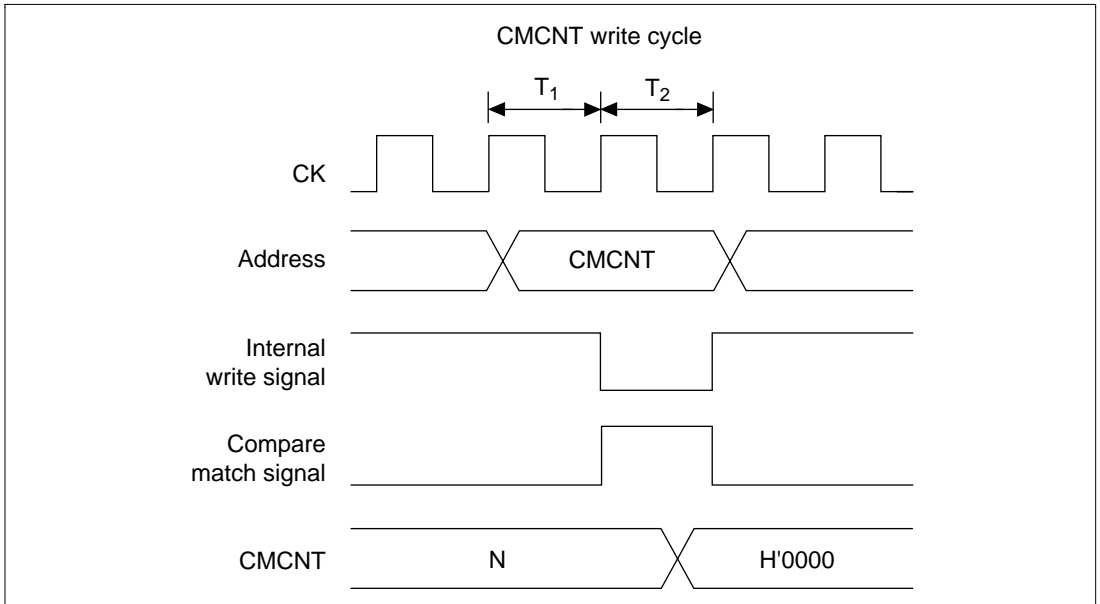


Figure 15.6 CMCNT Write and Compare Match Contention

15.5.2 Contention between CMCNT Word Write and Incrementation

If an increment occurs during the T_2 state of the CMCNT counter word write cycle, the counter write has priority, so no increment occurs. Figure 15.7 shows the timing.

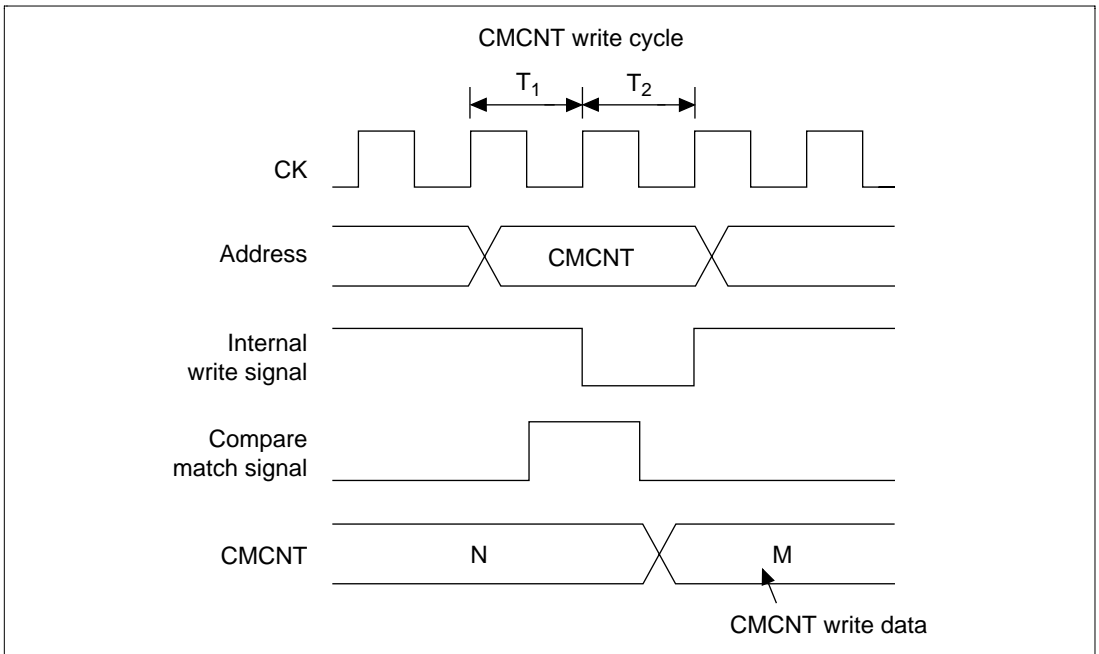


Figure 15.7 CMCNT Word Write and Increment Contention

15.5.3 Contention between CMCNT Byte Write and Incrementation

If an increment occurs during the T_2 state of the CMCNT byte write cycle, the counter write has priority, so no increment of the write data results on the writing side. The byte data on the side not performing the writing is also not incremented, so the contents are those before the write.

Figure 15.8 shows the timing when an increment occurs during the T_2 state of the CMCNTH write cycle.

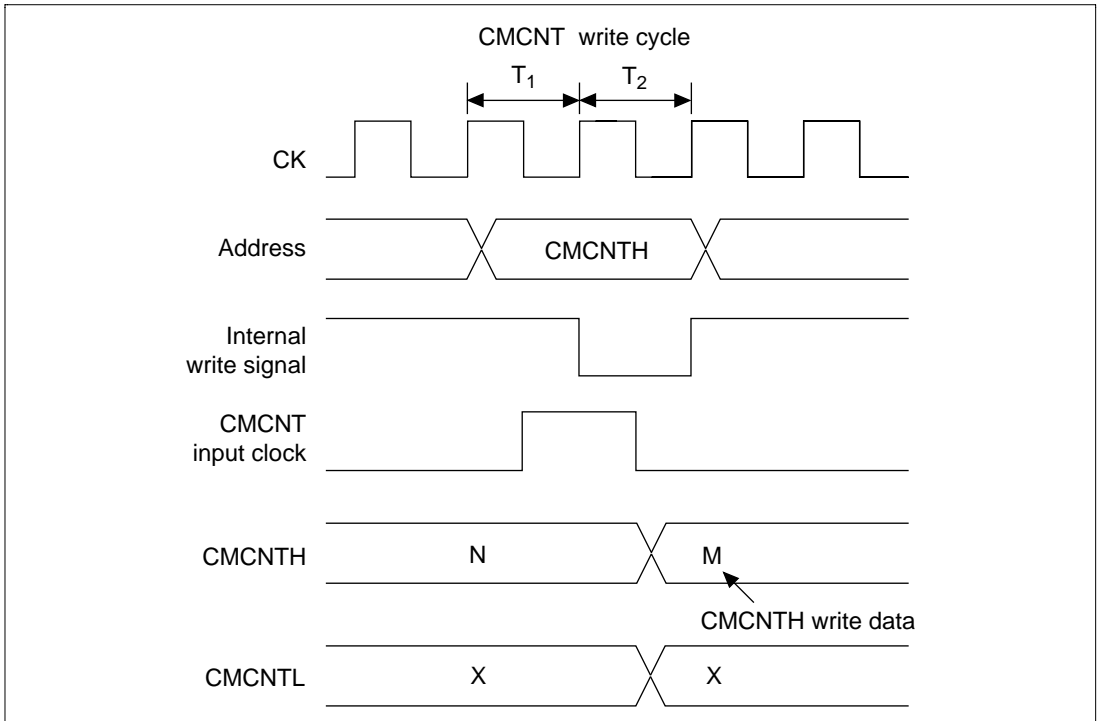


Figure 15.8 CMCNT Byte Write and Increment Contention

Section 16 Pin Function Controller

16.1 Overview

The pin function controller (PFC) is composed of registers for selecting the function of multiplexed pins and the direction of input/output. Table 16.1 lists this LSI's multiplexed pins. The multiplex pin functions have restrictions dependent on the operating mode. Table 16.2 lists the pin functions and initial values for each operating mode.

Table 16.1 Multiplexed Pins

| Port | Function 1 (Related Module) | Function 2 (Related Module) | Function 3 (Related Module) | Function 4 (Related Module) | Pin No. |
|------|--------------------------------|--------------------------------|------------------------------------|-----------------------------------|---------|
| A | PA15 I/O (port) | CK output (CPG) | — | — | 83 |
| A | PA14 I/O (port)* | \overline{RD} output (BSC) | — | — | 34 |
| A | PA13 I/O (port)* | \overline{WRH} output (BSC) | — | — | 36 |
| A | PA12 I/O (port)* | \overline{WRL} output (BSC) | — | — | 38 |
| A | PA11 I/O (port)* | $\overline{CS1}$ output (BSC) | — | — | 40 |
| A | PA10 I/O (port)* | $\overline{CS0}$ output (BSC) | — | — | 41 |
| A | PA9 I/O (port) | TCLKD input (MTU) | $\overline{IRQ3}$ (INTC) | — | 42 |
| A | PA8 I/O (port) | TCLKC input (MTU) | $\overline{IRQ2}$ (INTC) | — | 43 |
| A | PA7 I/O (port) | TCLKB input (MTU) | $\overline{CS3}$ output (BSC) | — | 44 |
| A | PA6 I/O (port) | TCLKA input (MTU) | $\overline{CS2}$ output (BSC) | — | 45 |
| A | PA5 I/O (port) | SCK1 I/O (SCI) | $\overline{DREQ1}$ input (DMAC) | $\overline{IRQ1}$ input (INTC) | 46 |
| A | PA4 I/O (port) | TxD1 output (SCI) | — | — | 47 |
| A | PA3 I/O (port) | RxD1 input (SCI) | — | — | 48 |
| A | PA2 I/O (port) | SCK0 I/O (SCI) | $\overline{DREQ0}$ input (DMAC) | $\overline{IRQ0}$ input (INTC) | 49 |
| A | PA1 I/O (port) | TxD0 output (SCI) | — | — | 50 |
| A | PA0 I/O (port) | RxD0 input (SCI) | — | — | 51 |
| B | PB9 I/O (port) | $\overline{IRQ7}$ input (INTC) | A21 output (BSC) | — | 32 |
| B | PB8 I/O (port) | $\overline{IRQ6}$ input (INTC) | A20 output (BSC) | \overline{WAIT} input (BSC) | 31 |
| B | PB7 I/O (port) | — | A19 output (BSC) | — | 30 |
| B | PB6 I/O (port) | — | A18 output (BSC) | — | 29 |

Table 16.1 Multiplexed Pins (cont)

| Port | Function 1 (Related Module) | Function 2 (Related Module) | Function 3 (Related Module) | Function 4 (Related Module) | Pin No. |
|-------------|--|--|--|--|----------------|
| B | PB5 I/O (port) | $\overline{\text{IRQ3}}$ input (INTC) | — | RDWR output (BSC) | 28 |
| B | PB4 I/O (port) | $\overline{\text{IRQ2}}$ input (INTC) | — | $\overline{\text{CASH}}$ output (BSC) | 26 |
| B | PB3 I/O (port) | $\overline{\text{IRQ1}}$ input (INTC) | — | $\overline{\text{CASL}}$ output (BSC) | 25 |
| B | PB2 I/O (port) | $\overline{\text{IRQ0}}$ input (INTC) | — | $\overline{\text{RAS}}$ output (BSC) | 24 |
| B | PB1 I/O (port)* | A17 input (BSC) | — | — | 22 |
| B | PB0 I/O (port)* | A16 output (BSC) | — | — | 20 |
| C | PC15 I/O (port)* | A15 output (BSC) | — | — | 19 |
| C | PC14 I/O (port)* | A14 output (BSC) | — | — | 18 |
| C | PC13 I/O (port)* | A13 output (BSC) | — | — | 17 |
| C | PC12 I/O (port)* | A12 output (BSC) | — | — | 16 |
| C | PC11 I/O (port)* | A11 output (BSC) | — | — | 15 |
| C | PC10 I/O (port)* | A10 output (BSC) | — | — | 14 |
| C | PC9 I/O (port)* | A9 output (BSC) | — | — | 13 |
| C | PC8 I/O (port)* | A8 output (BSC) | — | — | 12 |
| C | PC7 I/O (port)* | A7 output (BSC) | — | — | 11 |
| C | PC6 I/O (port)* | A6 output (BSC) | — | — | 10 |
| C | PC5 I/O (port)* | A5 output (BSC) | — | — | 9 |
| C | PC4 I/O (port)* | A4 output (BSC) | — | — | 8 |
| C | PC3 I/O (port)* | A3 output (BSC) | — | — | 7 |
| C | PC2 I/O (port)* | A2 output (BSC) | — | — | 6 |
| C | PC1 I/O (port)* | A1 output (BSC) | — | — | 5 |
| C | PC0 I/O (port)* | A0 output (BSC) | — | — | 4 |
| D | PD15 I/O (port)* | D15 I/O (BSC) | — | — | 52 |
| D | PD14 I/O (port)* | D14 I/O (BSC) | — | — | 53 |
| D | PD13 I/O (port)* | D13 I/O (BSC) | — | — | 54 |
| D | PD12 I/O (port)* | D12 I/O (BSC) | — | — | 56 |
| D | PD11 I/O (port)* | D11 I/O (BSC) | — | — | 57 |
| D | PD10 I/O (port)* | D10 I/O (BSC) | — | — | 58 |
| D | PD9 I/O (port)* | D9 I/O (BSC) | — | — | 59 |

Table 16.1 Multiplexed Pins (cont)

| Port | Function 1 (Related Module) | Function 2 (Related Module) | Function 3 (Related Module) | Function 4 (Related Module) | Pin No. |
|------|--------------------------------|--------------------------------|---|-------------------------------------|---------|
| D | PD8 I/O (port)* | D8 I/O (BSC) | — | — | 60 |
| D | PD7 I/O (port)* | D7 I/O (BSC) | — | — | 62 |
| D | PD6 I/O (port)* | D6 I/O (BSC) | — | — | 63 |
| D | PD5 I/O (port)* | D5 I/O (BSC) | — | — | 64 |
| D | PD4 I/O (port)* | D4 I/O (BSC) | — | — | 66 |
| D | PD3 I/O (port)* | D3 I/O (BSC) | — | — | 67 |
| D | PD2 I/O (port)* | D2 I/O (BSC) | — | — | 68 |
| D | PD1 I/O (port)* | D1 I/O (BSC) | — | — | 69 |
| D | PD0 I/O (port)* | D0 I/O (BSC) | — | — | 70 |
| E | PE15 I/O (port) | — | DACK1 output (DMAC) | — | 2 |
| E | PE14 I/O (port) | — | DACK0 output (DMAC) | $\overline{\text{AH}}$ output (BSC) | 1 |
| E | PE13 I/O (port) | — | — | — | 112 |
| E | PE12 I/O (port) | — | — | — | 111 |
| E | PE11 I/O (port) | — | — | — | 110 |
| E | PE10 I/O (port) | — | — | — | 108 |
| E | PE9 I/O (port) | — | — | — | 107 |
| E | PE8 I/O (port) | — | — | — | 106 |
| E | PE7 I/O (port) | TIOC2B I/O (MTU) | — | — | 105 |
| E | PE6 I/O (port) | TIOC2A I/O (MTU) | — | — | 104 |
| E | PE5 I/O (port) | TIOC1B I/O (MTU) | — | — | 102 |
| E | PE4 I/O (port) | TIOC1A I/O (MTU) | — | — | 89 |
| E | PE3 I/O (port) | TIOC0D I/O (MTU) | DRAK1 output (DMAC) | — | 88 |
| E | PE2 I/O (port) | TIOC0C I/O (MTU) | DREQ1 input (DMAC) | — | 87 |
| E | PE1 I/O (port) | TIOC0B I/O (MTU) | DRAK0 output (DMAC) | — | 86 |
| E | PE0 I/O (port) | TIOC0A I/O (MTU) | $\overline{\text{DREQ0}}$ input (DMAC) | — | 85 |

Table 16.1 Multiplexed Pins (cont)

| Port | Function 1 (Related Module) | Function 2 (Related Module) | Function 3 (Related Module) | Function 4 (Related Module) | Pin No. |
|-------------|--|--|--|--|----------------|
| F | PF7 input (port) | AN7 input (A/D) | — | — | 99 |
| F | PF6 input (port) | AN6 input (A/D) | — | — | 98 |
| F | PF5 input (port) | AN5 input (A/D) | — | — | 96 |
| F | PF4 input (port) | AN4 input (A/D) | — | — | 95 |
| F | PF3 input (port) | AN3 input (A/D) | — | — | 94 |
| F | PF2 input (port) | AN2 input (A/D) | — | — | 93 |
| F | PF1 input (port) | AN1 input (A/D) | — | — | 92 |
| F | PF0 input (port) | AN0 input (A/D) | — | — | 91 |

Note: SH7016, SH7017 only

Table 16.2 Pin Functions in Each Operating Mode (SH7014)

| Pin No. | Pin Name | | | |
|---|----------------------|--------------------------|------------------|--------------------------|
| | On-Chip ROM Disabled | | | |
| | MPU Mode 0 | | MPU Mode 1 | |
| FP-112 | Initial Function | Function Settable by PFC | Initial Function | Function Settable by PFC |
| 21, 37, 65, 103 | Vcc | Vcc | Vcc | Vcc |
| 3, 23, 27, 33, 39, 55, 61, 71, 90, 101, 109 | Vss | Vss | Vss | Vss |
| 70 | D0 | D0 | D0 | D0 |
| 69 | D1 | D1 | D1 | D1 |
| 68 | D2 | D2 | D2 | D2 |
| 67 | D3 | D3 | D3 | D3 |
| 66 | D4 | D4 | D4 | D4 |
| 64 | D5 | D5 | D5 | D5 |
| 63 | D6 | D6 | D6 | D6 |
| 62 | D7 | D7 | D7 | D7 |
| 60 | D8 | D8 | D8 | D8 |
| 59 | D9 | D9 | D9 | D9 |
| 58 | D10 | D10 | D10 | D10 |
| 57 | D11 | D11 | D11 | D11 |
| 56 | D12 | D12 | D12 | D12 |
| 54 | D13 | D13 | D13 | D13 |
| 53 | D14 | D14 | D14 | D14 |
| 52 | D15 | D15 | D15 | D15 |
| 4 | A0 | A0 | A0 | A0 |
| 5 | A1 | A1 | A1 | A1 |
| 6 | A2 | A2 | A2 | A2 |
| 7 | A3 | A3 | A3 | A3 |
| 8 | A4 | A4 | A4 | A4 |
| 9 | A5 | A5 | A5 | A5 |
| 10 | A6 | A6 | A6 | A6 |
| 11 | A7 | A7 | A7 | A7 |

Table 16.2 Pin Functions in Each Operating Mode (SH7014) (cont)

| Pin No. | Pin Name | | | |
|---------|----------------------|--|------------------|--|
| | On-Chip ROM Disabled | | | |
| | MPU Mode 0 | | MPU Mode 1 | |
| FP-112 | Initial Function | Function Settable by PFC | Initial Function | Function Settable by PFC |
| 12 | A8 | A8 | A8 | A8 |
| 13 | A9 | A9 | A9 | A9 |
| 14 | A10 | A10 | A10 | A10 |
| 15 | A11 | A11 | A11 | A11 |
| 16 | A12 | A12 | A12 | A12 |
| 17 | A13 | A13 | A13 | A13 |
| 18 | A14 | A14 | A14 | A14 |
| 19 | A15 | A15 | A15 | A15 |
| 20 | A16 | A16 | A16 | A16 |
| 22 | A17 | A17 | A17 | A17 |
| 24 | PB2 | PB2/ $\overline{\text{IRQ0}}$ /RAS | PB2 | PB2/ $\overline{\text{IRQ0}}$ /RAS |
| 25 | PB3 | PB3/ $\overline{\text{IRQ1}}$ /CASL | PB3 | PB3/ $\overline{\text{IRQ1}}$ /CASL |
| 26 | PB4 | PB4/ $\overline{\text{IRQ2}}$ /CASH | PB4 | PB4/ $\overline{\text{IRQ2}}$ /CASH |
| 28 | PB5 | PB5/ $\overline{\text{IRQ3}}$ /RDWR | PB5 | PB5/ $\overline{\text{IRQ3}}$ /RDWR |
| 29 | PB6 | PB6/A18 | PB6 | PB6/A18 |
| 30 | PB7 | PB7/A19 | PB7 | PB7/A19 |
| 31 | PB8 | PB8/ $\overline{\text{IRQ6}}$ /A20/WAIT | PB8 | PB8/ $\overline{\text{IRQ6}}$ /A20/WAIT |
| 32 | PB9 | PB9/ $\overline{\text{IRQ7}}$ /A21 | PB9 | PB9/ $\overline{\text{IRQ7}}$ /A21 |
| 51 | PA0 | PA0/RXD0 | PA0 | PA0/RXD0 |
| 50 | PA1 | PA1/TXD0 | PA1 | PA1/TXD0 |
| 49 | PA2 | PA2/SCK0/ $\overline{\text{DREQ0}}$ / $\overline{\text{IRQ0}}$ | PA2 | PA2/SCK0/ $\overline{\text{DREQ0}}$ / $\overline{\text{IRQ0}}$ |
| 48 | PA3 | PA3/RXD1 | PA3 | PA3/RXD1 |
| 47 | PA4 | PA4/TXD1 | PA4 | PA4/TXD1 |
| 46 | PA5 | PA5/SCK1/ $\overline{\text{DREQ1}}$ / $\overline{\text{IRQ1}}$ | PA5 | PA5/SCK1/ $\overline{\text{DREQ1}}$ / $\overline{\text{IRQ1}}$ |
| 45 | PA6 | PA6/TCLKA/ $\overline{\text{CS2}}$ | PA6 | PA6/TCLKA/ $\overline{\text{CS2}}$ |
| 44 | PA7 | PA7/TCLKB/ $\overline{\text{CS3}}$ | PA7 | PA7/TCLKB/ $\overline{\text{CS3}}$ |
| 43 | PA8 | PA8/TCLKC/ $\overline{\text{IRQ2}}$ | PA8 | PA8/TCLKC/ $\overline{\text{IRQ2}}$ |

Table 16.2 Pin Functions in Each Operating Mode (SH7014) (cont)

| Pin No. | Pin Name | | | |
|---------|----------------------------|-------------------------------------|----------------------------|-------------------------------------|
| | On-Chip ROM Disabled | | | |
| | MPU Mode 0 | | MPU Mode 1 | |
| FP-112 | Initial Function | Function Settable by PFC | Initial Function | Function Settable by PFC |
| 42 | PA9 | PA9/TCLKD/ $\overline{\text{IRQ3}}$ | PA9 | PA9/TCLKD/ $\overline{\text{IRQ3}}$ |
| 41 | $\overline{\text{CS0}}$ | $\overline{\text{CS0}}$ | $\overline{\text{CS0}}$ | $\overline{\text{CS0}}$ |
| 40 | $\overline{\text{CS1}}$ | $\overline{\text{CS1}}$ | $\overline{\text{CS1}}$ | $\overline{\text{CS1}}$ |
| 38 | $\overline{\text{WRL}}$ | $\overline{\text{WRL}}$ | $\overline{\text{WRL}}$ | $\overline{\text{WRL}}$ |
| 36 | $\overline{\text{WRH}}$ | $\overline{\text{WRH}}$ | $\overline{\text{WRH}}$ | $\overline{\text{WRH}}$ |
| 34 | $\overline{\text{RD}}$ | $\overline{\text{RD}}$ | $\overline{\text{RD}}$ | $\overline{\text{RD}}$ |
| 83 | CK | PA15/CK | CK | PA15/CK |
| 80 | PLLVCC | PLLVCC | PLLVCC | PLLVCC |
| 82 | PLLSS | PLLSS | PLLSS | PLLSS |
| 74 | EXTAL | EXTAL | EXTAL | EXTAL |
| 72 | XTAL | XTAL | XTAL | XTAL |
| 81 | PLLCAP | PLLCAP | PLLCAP | PLLCAP |
| 76 | NMI | NMI | NMI | NMI |
| 84 | $\overline{\text{RES}}$ | $\overline{\text{RES}}$ | $\overline{\text{RES}}$ | $\overline{\text{RES}}$ |
| 35 | $\overline{\text{WDTOVF}}$ | $\overline{\text{WDTOVF}}$ | $\overline{\text{WDTOVF}}$ | $\overline{\text{WDTOVF}}$ |
| 79 | MD0 | MD0 | MD0 | MD0 |
| 78 | MD1 | MD1 | MD1 | MD1 |
| 75 | MD2 | MD2 | MD2 | MD2 |
| 73 | MD3 | MD3 | MD3 | MD3 |
| 77 | VCC | VCC | VCC | VCC |
| 100 | AVCC | AVCC | AVCC | AVCC |
| 97 | AVSS | AVSS | AVSS | AVSS |
| 91 | PF0/AN0 | PF0/AN0 | PF0/AN0 | PF0/AN0 |
| 92 | PF1/AN1 | PF1/AN1 | PF1/AN1 | PF1/AN1 |
| 93 | PF2/AN2 | PF2/AN2 | PF2/AN2 | PF2/AN2 |
| 94 | PF3/AN3 | PF3/AN3 | PF3/AN3 | PF3/AN3 |
| 95 | PF4/AN4 | PF4/AN4 | PF4/AN4 | PF4/AN4 |

Table 16.2 Pin Functions in Each Operating Mode (SH7014) (cont)

| Pin No. | Pin Name | | | |
|---------|----------------------|--------------------------|------------------|--------------------------|
| | On-Chip ROM Disabled | | | |
| | MPU Mode 0 | | MPU Mode 1 | |
| FP-112 | Initial Function | Function Settable by PFC | Initial Function | Function Settable by PFC |
| 96 | PF5/AN5 | PF5/AN5 | PF5/AN5 | PF5/AN5 |
| 98 | PF6/AN6 | PF6/AN6 | PF6/AN6 | PF6/AN6 |
| 99 | PF7/AN7 | PF7/AN7 | PF7/AN7 | PF7/AN7 |
| 85 | PE0 | PE0/TIOC0A/DREQ0 | PE0 | PE0/TIOC0A/DREQ0 |
| 86 | PE1 | PE1/TIOC0B/DRAK0 | PE1 | PE1/TIOC0B/DRAK0 |
| 87 | PE2 | PE2/TIOC0C/DREQ1 | PE2 | PE2/TIOC0C/DREQ1 |
| 88 | PE3 | PE3/TIOC0D/DRAK1 | PE3 | PE3/TIOC0D/DRAK1 |
| 89 | PE4 | PE4/TIOC1A | PE4 | PE4/TIOC1A |
| 102 | PE5 | PE5/TIOC1B | PE5 | PE5/TIOC1B |
| 104 | PE6 | PE6/TIOC2A | PE6 | PE6/TIOC2A |
| 105 | PE7 | PE7/TIOC2B | PE7 | PE7/TIOC2B |
| 106 | PE8 | PE8 | PE8 | PE8 |
| 107 | PE9 | PE9 | PE9 | PE9 |
| 108 | PE10 | PE10 | PE10 | PE10 |
| 110 | PE11 | PE11 | PE11 | PE11 |
| 111 | PE12 | PE12 | PE12 | PE12 |
| 112 | PE13 | PE13 | PE13 | PE13 |
| 1 | PE14 | PE14/DACK0/AH | PE14 | PE14/DACK0/AH |
| 2 | PE15 | PE15/DACK1 | PE15 | PE15/DACK1 |

Table 16.3 Pin Arrangement in Each Operating Mode (SH7016, SH7017)

| Pin NO. | Pin Name | | | | On-Chip ROM Disabled | | | | On-Chip ROM Enabled | | | | Single Chip Mode | | | |
|---|----------------------|-------------------------------------|------------------|-------------------------------------|----------------------|-------------------------------------|------------------|-------------------------------------|---------------------|-------------------------------------|------------------|-------------------------------------|------------------|-------------------------------------|------------------|-------------------------------------|
| | On-Chip ROM Disabled | | MPU Mode 1 | | MPU Mode 2 | | MPU Mode 2 | | On-Chip ROM Enabled | | MPU Mode 2 | | Single Chip Mode | | Single Chip Mode | |
| | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities |
| FP-112 | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc | Vcc |
| 21,37,65 103 | Vss | Vss | Vss | Vss | Vss | Vss | Vss | Vss | Vss | Vss | Vss | Vss | Vss | Vss | Vss | Vss |
| 3,23,27,33 39,55,61,71 90,101,109 | D0 | D0 | D0 | D0 | D0 | D0 | D0 | D0 | D0 | D0 | D0 | D0 | D0 | D0 | D0 | D0 |
| 70 | D1 | D1 | D1 | D1 | D1 | D1 | D1 | D1 | D1 | D1 | D1 | D1 | D1 | D1 | D1 | D1 |
| 69 | D2 | D2 | D2 | D2 | D2 | D2 | D2 | D2 | D2 | D2 | D2 | D2 | D2 | D2 | D2 | D2 |
| 68 | D3 | D3 | D3 | D3 | D3 | D3 | D3 | D3 | D3 | D3 | D3 | D3 | D3 | D3 | D3 | D3 |
| 67 | D4 | D4 | D4 | D4 | D4 | D4 | D4 | D4 | D4 | D4 | D4 | D4 | D4 | D4 | D4 | D4 |
| 66 | D5 | D5 | D5 | D5 | D5 | D5 | D5 | D5 | D5 | D5 | D5 | D5 | D5 | D5 | D5 | D5 |
| 64 | D6 | D6 | D6 | D6 | D6 | D6 | D6 | D6 | D6 | D6 | D6 | D6 | D6 | D6 | D6 | D6 |
| 63 | D7 | D7 | D7 | D7 | D7 | D7 | D7 | D7 | D7 | D7 | D7 | D7 | D7 | D7 | D7 | D7 |
| 62 | D8 | D8 | D8 | D8 | D8 | D8 | D8 | D8 | D8 | D8 | D8 | D8 | D8 | D8 | D8 | D8 |
| 60 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 | D9 |
| 59 | D10 | D10 | D10 | D10 | D10 | D10 | D10 | D10 | D10 | D10 | D10 | D10 | D10 | D10 | D10 | D10 |
| 58 | D11 | D11 | D11 | D11 | D11 | D11 | D11 | D11 | D11 | D11 | D11 | D11 | D11 | D11 | D11 | D11 |
| 57 | D12 | D12 | D12 | D12 | D12 | D12 | D12 | D12 | D12 | D12 | D12 | D12 | D12 | D12 | D12 | D12 |
| 56 | D13 | D13 | D13 | D13 | D13 | D13 | D13 | D13 | D13 | D13 | D13 | D13 | D13 | D13 | D13 | D13 |
| 54 | D14 | D14 | D14 | D14 | D14 | D14 | D14 | D14 | D14 | D14 | D14 | D14 | D14 | D14 | D14 | D14 |
| 53 | D15 | D15 | D15 | D15 | D15 | D15 | D15 | D15 | D15 | D15 | D15 | D15 | D15 | D15 | D15 | D15 |
| 52 | A0 | A0 | A0 | A0 | A0 | A0 | A0 | A0 | A0 | A0 | A0 | A0 | A0 | A0 | A0 | A0 |
| 4 | A1 | A1 | A1 | A1 | A1 | A1 | A1 | A1 | A1 | A1 | A1 | A1 | A1 | A1 | A1 | A1 |
| 5 | A2 | A2 | A2 | A2 | A2 | A2 | A2 | A2 | A2 | A2 | A2 | A2 | A2 | A2 | A2 | A2 |
| 6 | A3 | A3 | A3 | A3 | A3 | A3 | A3 | A3 | A3 | A3 | A3 | A3 | A3 | A3 | A3 | A3 |
| 7 | A4 | A4 | A4 | A4 | A4 | A4 | A4 | A4 | A4 | A4 | A4 | A4 | A4 | A4 | A4 | A4 |
| 8 | A5 | A5 | A5 | A5 | A5 | A5 | A5 | A5 | A5 | A5 | A5 | A5 | A5 | A5 | A5 | A5 |
| 9 | A6 | A6 | A6 | A6 | A6 | A6 | A6 | A6 | A6 | A6 | A6 | A6 | A6 | A6 | A6 | A6 |
| 10 | A7 | A7 | A7 | A7 | A7 | A7 | A7 | A7 | A7 | A7 | A7 | A7 | A7 | A7 | A7 | A7 |
| 11 | A8 | A8 | A8 | A8 | A8 | A8 | A8 | A8 | A8 | A8 | A8 | A8 | A8 | A8 | A8 | A8 |
| 12 | A9 | A9 | A9 | A9 | A9 | A9 | A9 | A9 | A9 | A9 | A9 | A9 | A9 | A9 | A9 | A9 |
| 13 | A10 | A10 | A10 | A10 | A10 | A10 | A10 | A10 | A10 | A10 | A10 | A10 | A10 | A10 | A10 | A10 |
| 14 | A11 | A11 | A11 | A11 | A11 | A11 | A11 | A11 | A11 | A11 | A11 | A11 | A11 | A11 | A11 | A11 |
| 15 | A12 | A12 | A12 | A12 | A12 | A12 | A12 | A12 | A12 | A12 | A12 | A12 | A12 | A12 | A12 | A12 |
| 16 | | | | | | | | | | | | | | | | |

Table 16.3 Pin Arrangement in Each Operating Mode (SH7016, SH7017) (cont)

| Pin NO. | Pin Name | On-Chip ROM Disabled | | On-Chip ROM Enabled | | Single Chip Mode | |
|---------|-------------------------------|-------------------------------|-------------------------------------|-------------------------------|-------------------------------------|-------------------------------|-------------------------------------|
| | | MPU Mode0 | MPU Mode1 | MPU Mode2 | MPU Mode3 | Initial Function | PFC Selected Function Possibilities |
| FP-112 | | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities |
| 17 | A13 | A13 | A13 | PC13 | PC13/A13 | PC13 | PC13 |
| 18 | A14 | A14 | A14 | PC14 | PC14/A14 | PC14 | PC14 |
| 19 | A15 | A15 | A15 | PC15 | PC15/A15 | PC15 | PC15 |
| 20 | A16 | A16 | A16 | PB0 | PB0/A16 | PB0 | PB0 |
| 22 | A17 | A17 | A17 | PB1 | PB1/A17 | PB1 | PB1 |
| 24 | PB2 | PB2/IRQ0/POE0/RAS | PB2 | PB2/IRQ0/RAS | PB2 | PB2 | PB2/IRQ0 |
| 25 | PB3 | PB3/IRQ1/POE1/CASL | PB3 | PB3/IRQ1/CASL | PB3 | PB3 | PB3/IRQ1 |
| 26 | PB4 | PB4/IRQ2/POE2/CASH | PB4 | PB4/IRQ2/CASH | PB4 | PB4 | PB4/IRQ2 |
| 28 | PB5 | PB5/IRQ3/POE3/RDWR | PB5 | PB5/IRQ3/RDWR | PB5 | PB5 | PB5/IRQ3 |
| 29 | PB6 | PB6/A18 | PB6 | PB6/A18 | PB6 | PB6 | PB6 |
| 30 | PB7 | PB7/A19 | PB7 | PB7/A19 | PB7 | PB7 | PB7 |
| 31 | PB8 | PB8/IRQ6/A20/WAIT | PB8 | PB8/IRQ6/A20/WAIT | PB8 | PB8 | PB8/IRQ6 |
| 32 | PB9 | PB9/IRQ7/A21 | PB9 | PB9/IRQ7/A21 | PB9 | PB9 | PB9/IRQ7 |
| 51 | PA0 | PA0/RXD0 | PA0 | PA0/RXD0 | PA0 | PA0 | PA0/RXD0 |
| 50 | PA1 | PA1/TXD0 | PA1 | PA1/TXD0 | PA1 | PA1 | PA1/TXD0 |
| 49 | PA2 | PA2/SCK0/DREQ0/IRQ0 | PA2 | PA2/SCK0/DREQ0/IRQ0 | PA2 | PA2 | PA2/SCK0/IRQ0 |
| 48 | PA3 | PA3/RXD1 | PA3 | PA3/RXD1 | PA3 | PA3 | PA3/RXD1 |
| 47 | PA4 | PA4/TXD1 | PA4 | PA4/TXD1 | PA4 | PA4 | PA4/TXD1 |
| 46 | PA5 | PA5/SCK1/DREQ1/IRQ1 | PA5 | PA5/SCK1/DREQ1/IRQ1 | PA5 | PA5 | PA5/SCK1/IRQ1 |
| 45 | PA6 | PA6/TCLKA/CS2 | PA6 | PA6/TCLKA/CS2 | PA6 | PA6 | PA6/TCLKA |
| 44 | PA7 | PA7/TCLKB/CS3 | PA7 | PA7/TCLKB/CS3 | PA7 | PA7 | PA7/TCLKB |
| 43 | PA8 | PA8/TCLKC/IRQ2 | PA8 | PA8/TCLKC/IRQ2 | PA8 | PA8 | PA8/TCLKC/IRQ2 |
| 42 | PA9 | PA9/TCLKD/IRQ3 | PA9 | PA9/TCLKD/IRQ3 | PA9 | PA9 | PA9/TCLKD/IRQ3 |
| 41 | CS0 | CS0 | CS0 | PA10/CS0 | PA10 | PA10 | PA10 |
| 40 | CS1 | CS1 | CS1 | PA11/CS1 | PA11 | PA11 | PA11 |
| 38 | WRL | WRL | WRL | PA12/WRL | PA12 | PA12 | PA12 |
| 36 | WRH | WRH | WRH | PA13/WRH | PA13 | PA13 | PA13 |
| 34 | RD | RD | RD | PA14/RD | PA14 | PA14 | PA14 |
| 83 | CK | PA15/CK | CK | PA15/CK | PA15 | PA15 | PA15/CK |
| 80 | PLL _{V_{CC}} | PLL _{V_{CC}} | PLL _{V_{CC}} | PLL _{V_{CC}} | PLL _{V_{CC}} | PLL _{V_{CC}} | PLL _{V_{CC}} |
| 82 | PLL _{V_{SS}} | PLL _{V_{SS}} | PLL _{V_{SS}} | PLL _{V_{SS}} | PLL _{V_{SS}} | PLL _{V_{SS}} | PLL _{V_{SS}} |
| 74 | EXTAL | EXTAL | EXTAL | EXTAL | EXTAL | EXTAL | EXTAL |
| 72 | XTAL | XTAL | XTAL | XTAL | XTAL | XTAL | XTAL |
| 81 | PLL _{CAP} | PLL _{CAP} | PLL _{CAP} | PLL _{CAP} | PLL _{CAP} | PLL _{CAP} | PLL _{CAP} |
| 76 | NMI | NMI | NMI | NMI | NMI | NMI | NMI |
| 84 | RES | RES | RES | RES | RES | RES | RES |
| 35 | WDTOVF | WDTOVF | WDTOVF | WDTOVF | WDTOVF | WDTOVF | WDTOVF |
| 79 | MD0 | MD0 | MD0 | MD0 | MD0 | MD0 | MD0 |
| 78 | MD1 | MD1 | MD1 | MD1 | MD1 | MD1 | MD1 |
| 75 | MD2 | MD2 | MD2 | MD2 | MD2 | MD2 | MD2 |
| 73 | MD3 | MD3 | MD3 | MD3 | MD3 | MD3 | MD3 |
| 77 | V _{CC} | V _{CC} | V _{CC} | V _{CC} | V _{CC} | V _{CC} | V _{CC} |
| 100 | AV _{CC} | AV _{CC} | AV _{CC} | AV _{CC} | AV _{CC} | AV _{CC} | AV _{CC} |
| 97 | AV _{SS} | AV _{SS} | AV _{SS} | AV _{SS} | AV _{SS} | AV _{SS} | AV _{SS} |

Table 16.3 Pin Arrangement in Each Operating Mode (SH7016, SH7017) (cont)

| Pin No. | Pin Name | On-Chip ROM Disabled | | | | On-Chip ROM Enabled | | | | Single Chip Mode | |
|---------|----------|----------------------|-------------------------------------|------------------|-------------------------------------|---------------------|-------------------------------------|------------------|-------------------------------------|------------------|-------------------------------------|
| | | MPU Mode 0 | | MPU Mode 1 | | MPU Mode 2 | | MPU Mode 3 | | Initial Function | PFC Selected Function Possibilities |
| | | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities | Initial Function | PFC Selected Function Possibilities | PF0/AN0 | PF0/AND |
| FF-112 | | | | | | | | | | | |
| 91 | | PF0/AND | PF0/AND | PF0/AND | PF0/AND | PF0/AND | PF0/AND | PF0/AND | PF0/AND | PF0/AND | PF0/AND |
| 92 | | PF1/AN1 | PF1/AN1 | PF1/AN1 | PF1/AN1 | PF1/AN1 | PF1/AN1 | PF1/AN1 | PF1/AN1 | PF1/AN1 | PF1/AN1 |
| 93 | | PF2/AN2 | PF2/AN2 | PF2/AN2 | PF2/AN2 | PF2/AN2 | PF2/AN2 | PF2/AN2 | PF2/AN2 | PF2/AN2 | PF2/AN2 |
| 94 | | PF3/AN3 | PF3/AN3 | PF3/AN3 | PF3/AN3 | PF3/AN3 | PF3/AN3 | PF3/AN3 | PF3/AN3 | PF3/AN3 | PF3/AN3 |
| 95 | | PF4/AN4 | PF4/AN4 | PF4/AN4 | PF4/AN4 | PF4/AN4 | PF4/AN4 | PF4/AN4 | PF4/AN4 | PF4/AN4 | PF4/AN4 |
| 96 | | PF5/AN5 | PF5/AN5 | PF5/AN5 | PF5/AN5 | PF5/AN5 | PF5/AN5 | PF5/AN5 | PF5/AN5 | PF5/AN5 | PF5/AN5 |
| 98 | | PF6/AN6 | PF6/AN6 | PF6/AN6 | PF6/AN6 | PF6/AN6 | PF6/AN6 | PF6/AN6 | PF6/AN6 | PF6/AN6 | PF6/AN6 |
| 99 | | PF7/AN7 | PF7/AN7 | PF7/AN7 | PF7/AN7 | PF7/AN7 | PF7/AN7 | PF7/AN7 | PF7/AN7 | PF7/AN7 | PF7/AN7 |
| 85 | PE0 | PE0/TIOC0A/DREQ0 | PE0/TIOC0A/DREQ0 | PE0 | PE0/TIOC0A/DREQ0 | PE0 | PE0/TIOC0A/DREQ0 | PE0 | PE0/TIOC0A | PE0 | PE0/TIOC0A |
| 86 | PE1 | PE1/TIOC0B/DRAK0 | PE1/TIOC0B/DRAK0 | PE1 | PE1/TIOC0B/DRAK0 | PE1 | PE1/TIOC0B/DRAK0 | PE1 | PE1/TIOC0B | PE1 | PE1/TIOC0B |
| 87 | PE2 | PE2/TIOC0C/DREQ1 | PE2/TIOC0C/DREQ1 | PE2 | PE2/TIOC0C/DREQ1 | PE2 | PE2/TIOC0C/DREQ1 | PE2 | PE2/TIOC0C | PE2 | PE2/TIOC0C |
| 88 | PE3 | PE3/TIOC0D/DRAK1 | PE3/TIOC0D/DRAK1 | PE3 | PE3/TIOC0D/DRAK1 | PE3 | PE3/TIOC0D/DRAK1 | PE3 | PE3/TIOC0D | PE3 | PE3/TIOC0D |
| 89 | PE4 | PE4/TIOC1A | PE4/TIOC1A | PE4 | PE4/TIOC1A | PE4 | PE4/TIOC1A | PE4 | PE4/TIOC1A | PE4 | PE4/TIOC1A |
| 102 | PE5 | PE5/TIOC1B | PE5/TIOC1B | PE5 | PE5/TIOC1B | PE5 | PE5/TIOC1B | PE5 | PE5/TIOC1B | PE5 | PE5/TIOC1B |
| 104 | PE6 | PE6/TIOC2A | PE6/TIOC2A | PE6 | PE6/TIOC2A | PE6 | PE6/TIOC2A | PE6 | PE6/TIOC2A | PE6 | PE6/TIOC2A |
| 105 | PE7 | PE7/TIOC2B | PE7/TIOC2B | PE7 | PE7/TIOC2B | PE7 | PE7/TIOC2B | PE7 | PE7/TIOC2B | PE7 | PE7/TIOC2B |
| 106 | PE8 | PE8 | PE8 | PE8 | PE8 | PE8 | PE8 | PE8 | PE8 | PE8 | PE8 |
| 107 | PE9 | PE9 | PE9 | PE9 | PE9 | PE9 | PE9 | PE9 | PE9 | PE9 | PE9 |
| 108 | PE10 | PE10 | PE10 | PE10 | PE10 | PE10 | PE10 | PE10 | PE10 | PE10 | PE10 |
| 110 | PE11 | PE11 | PE11 | PE11 | PE11 | PE11 | PE11 | PE11 | PE11 | PE11 | PE11 |
| 111 | PE12 | PE12 | PE12 | PE12 | PE12 | PE12 | PE12 | PE12 | PE12 | PE12 | PE12 |
| 112 | PE13 | PE13 | PE13 | PE13 | PE13 | PE13 | PE13 | PE13 | PE13 | PE13 | PE13 |
| 1 | PE14 | PE14/DACK0/AH | PE14/DACK0/AH | PE14 | PE14/DACK0/AH | PE14 | PE14/DACK0/AH | PE14 | PE14/DACK0/AH | PE14 | PE14 |
| 2 | PE15 | PE15/DACK1 | PE15/DACK1 | PE15 | PE15/DACK1 | PE15 | PE15/DACK1 | PE15 | PE15/DACK1 | PE15 | PE15 |

16.2 Register Configuration

Table 16.4 summarizes the registers of the pin function controller.

Table 16.4 Pin Function Controller Registers

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|---|--------------|-----|--------------------------------|--------------------------|-------------|
| Port A I/O register L | PAIORL | R/W | H'0000 | H'FFFF8386 H'FFFF8387 | 8, 16, 32 |
| Port A control register L1 | PACRL1 | R/W | H'0000* ¹ H'4000 | H'FFFF838C H'FFFF838D | 8, 16, 32 |
| Port A control register L2 | PACRL2 | R/W | H'0000 | H'FFFF838E H'FFFF838F | 8, 16, 32 |
| Port B I/O register | PBIOR | R/W | H'0000 | H'FFFF8394 H'FFFF8395 | 8, 16, 32 |
| Port B control register 1 | PBCR1 | R/W | H'0000 | H'FFFF8398 H'FFFF8399 | 8, 16, 32 |
| Port B control register 2 | PBCR2 | R/W | H'0000 | H'FFFF839A H'FFFF839B | 8, 16, 32 |
| Port C I/O register* ² | PCIOR | R/W | H'0000 | H'FFFF8396 H'FFFF8397 | 8, 16, 32 |
| Port C control register* ² | PCCR | R/W | H'0000 | H'FFFF839C H'FFFF839D | 8, 16, 32 |
| Port D I/O register L* ² | PDIORL | R/W | H'0000 | H'FFFF83A6 H'FFFF83A7 | 8, 16, 32 |
| Port D control register L* ² | PDCRL | R/W | H'0000 | H'FFFF83AC H'FFFF83AD | 8, 16, 32 |
| Port E I/O register | PEIOR | R/W | H'0000 | H'FFFF83B4 H'FFFF83B5 | 8, 16, 32 |
| Port E control register 1 | PECR1 | R/W | H'0000 | H'FFFF83B8 H'FFFF83B9 | 8, 16, 32 |
| Port E control register 2 | PECR2 | R/W | H'0000 | H'FFFF83BA H'FFFF83BB | 8, 16, 32 |

Notes: 1. The port A control register L1 initial value varies depending on the operating mode.

2. SH7016, SH7017 only.

16.3 Register Descriptions

16.3.1 Port A I/O Register L (PAIORL)

The port A I/O register L (PAIORL) is a 16-bit read/write register that selects input or output for the 16 pins of port A (11 pins in the SH7014). Bits PA15IOR to PA0IOR correspond to pins PA15/CK to PA0/RXD0. PAIORL is enabled when the port A pins function as general input/outputs (PA15–PA0), or with the serial clock (SCK1, SCK0). For other functions, it is disabled.

When the port A pin functions PA15–PA0 are SCK1, SCK0, a given pin in port A is an output pin if its corresponding PAIORL bit is set to 1, and an input pin if the bit is cleared to 0.

PAIORL is initialized to H'0000 by external power-on reset; however, it is not initialized for reset by WDT, standby mode, or sleep mode, so the previous data is maintained.

The settings of bits 14 to 10 of this register are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to bits 14 to 10; these bits are always read as 0, and their write value should always be 0.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|-------------|-------------|-------------|-------------|-------------|-------------|------------|------------|
| | PA15 IOR | PA14 IOR | PA13 IOR | PA12 IOR | PA11 IOR | PA10 IOR | PA9 IOR | PA8 IOR |
| 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 |
|----------------|------------|------------|------------|------------|------------|------------|------------|------------|
| | PA7 IOR | PA6 IOR | PA5 IOR | PA4 IOR | PA3 IOR | PA2 IOR | PA1 IOR | PA0 IOR |
| 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: * R only in the SH7014

16.3.2 Port A Control Registers L1, L2 (PACRL1 and PACRL2)

PACRL1 and PACRL2 are 16-bit read/write registers that select the functions of the 16 multiplexed pins of port A (11 pins in the SH7014). PACRL1 selects the function of the PA15/CK to PA8/TCLKC/ $\overline{\text{IRQ}}2$ pins of port A; PACRL2 selects the function of the PA7/TCLKB/ $\overline{\text{CS}}3$ to PA0/RXD0 pins of port A.

Port A includes bus control signals ($\overline{\text{RD}}$, $\overline{\text{WRH}}$, $\overline{\text{WRL}}$, $\overline{\text{CS}}0$ to $\overline{\text{CS}}3$, and $\overline{\text{AH}}$) and DMAC control signals ($\overline{\text{DREQ}}0$ and $\overline{\text{DREQ}}1$), but in the SH7016 and SH7017, register settings relating to the selection of these pin functions may be invalid depending on the operating mode. For details, see table 16.2, Pin Functions in Each Operating Mode.

PACRL1 is initialized by external power-on reset to H'4000. PACRL2 is initialized by external power-on reset to H'0000. Neither register is initialized by reset by WDT, standby mode, or sleep mode, so the previous data is maintained.

The settings of bits 12, 10, 8, 6, and 4 of PACRL1 are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to these bits; they are always read as 0, and their write value should always be 0.

Port A Control Register L1 (PACRL1):

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|----|---------------------|----|-------------------|----|-------------------|---|-------------------|
| | — | PA15MD | — | PA14MD | — | PA13MD | — | PA12MD |
| Initial value: | 0 | 0 (1)* ¹ | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R/W | R | R/W* ² | R | R/W* ² | R | R/W* ² |

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|---|-------------------|---|-------------------|--------|--------|--------|--------|
| | — | PA11MD | — | PA10MD | PA9MD1 | PA9MD0 | PA8MD1 | PA8MD0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R/W* ² | R | R/W* ² | R/W | R/W | R/W | R/W |

- Notes: 1. Bit 14 is initialized to 1 in extended mode.
 2. R only in the SH7014.

- Bit 15—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 14—PA15 Mode (PA15MD): Selects the function of the PA15/CK pin.

Bit 14: PA15MD Description

| 0 | General input/output (PA15) |
|---|-----------------------------------|
| 1 | Clock output (CK) (initial value) |

- Bits 13—Reserved: This bit always read as 0. The write value should always be 0.
- Bit 12—PA14 Mode (PA14MD) –**SH7016, SH7017**–: Selects the function of the PA14/ $\overline{\text{RD}}$ pin.

Bit 12: PA14MD Description

| | |
|---|---|
| 0 | General input/output (PA14) (initial value) ($\overline{\text{RD}}$ in on-chip ROM invalid mode) |
| 1 | Read output ($\overline{\text{RD}}$) (PA14 in single chip mode) |

- Bit 11—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 10—PA13 Mode (PA13MD) –**SH7016, SH7017**–: Selects the function of the PA13/ $\overline{\text{WRH}}$ pin.

Bit 10: PA13MD Description

| | |
|---|--|
| 0 | General input/output (PA13) (initial value) ($\overline{\text{WRH}}$ in on-chip ROM invalid mode) |
| 1 | Most significant side write output ($\overline{\text{WRH}}$) (PA13 in single chip mode) |

- Bit 9—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 8—PA12 Mode (PA12MD) –**SH7016, SH7017**–: Selects the function of the PA12/ $\overline{\text{WRL}}$ pin.

Bit 8: PA12MD Description

| | |
|---|--|
| 0 | General input/output (PA12) (initial value) ($\overline{\text{WRL}}$ in on-chip ROM invalid mode) |
| 1 | Least significant side write output ($\overline{\text{WRL}}$) (PA12 in single chip mode) |

- Bit 7—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 6—PA11 Mode (PA11MD) –**SH7016, SH7017**–: Selects the function of the PA11/ $\overline{\text{CS1}}$ pin.

Bit 6: PA11MD Description

| | |
|---|--|
| 0 | General input/output (PA11) (initial value) ($\overline{\text{CS1}}$ in on-chip ROM invalid mode) |
| 1 | Chip select output ($\overline{\text{CS1}}$) (PA11 in single chip mode) |

- Bit 5—Reserved: This bit always reads as 0. The write value should always be 0.

- Bit 4—PA10 Mode (PA10MD) –**SH7016, SH7017**–: Selects the function of the PA10/ $\overline{\text{CS0}}$ pin.

Bit 4: PA10MD **Description**

| | |
|---|--|
| 0 | General input/output (PA10) (initial value) ($\overline{\text{CS0}}$ in on-chip ROM invalid mode) |
| 1 | Chip select output ($\overline{\text{CS0}}$) (PA10 in single chip mode) |

- Bits 3 and 2—PA9 Mode 1, 0 (PA9MD1 and PA9MD0): These bits select the function of the PA9/TCLKD/ $\overline{\text{IRQ3}}$ pin.

| Bit 3: PA9MD1 | Bit 2: PA9MD0 | Description |
|--------------------------|--------------------------|--|
| 0 | 0 | General input/output (PA9) (initial value) |
| | 1 | MTU timer clock input (TCLKD) |
| 1 | 0 | Interrupt request input ($\overline{\text{IRQ3}}$) |
| | 1 | Reserved |

- Bits 1 and 0—PA8 Mode 1, 0 (PA8MD1 and PA8MD0): These bits select the function of the PA8/TCLKC/ $\overline{\text{IRQ2}}$ pin.

| Bit 1: PA8MD1 | Bit 0: PA8MD0 | Description |
|--------------------------|--------------------------|--|
| 0 | 0 | General input/output (PA8) (initial value) |
| | 1 | MTU timer clock input (TCLKC) |
| 1 | 0 | Interrupt request input ($\overline{\text{IRQ2}}$) |
| | 1 | Reserved |

Port A Control Register L2 (PACRL2):

| | | | | | | | | |
|----------------|---------|---------|---------|---------|---------|---------|---|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PA7 MD1 | PA7 MD0 | PA6 MD1 | PA6 MD0 | PA5 MD1 | PA5 MD0 | — | PA4MD |
| 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 |

| | | | | | | | | |
|----------------|---|-------|---------|---------|---|-------|---|-------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | PA3MD | PA2 MD1 | PA2 MD0 | — | PA1MD | — | PA0MD |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R/W | R/W | R/W | R | R/W | R | R/W |

- Bits 15 and 14—PA7 Mode 1, 0 (PA7MD1 and PA7MD0): These bits select the function of the PA7/TCLKB/ $\overline{\text{CS3}}$ pin.

| Bit 15: PA7MD1 | Bit 14: PA7MD0 | Description |
|-------------------|-------------------|--|
| 0 | 0 | General input/output (PA7) (initial value) |
| | 1 | MTU timer clock input (TCLKB) |
| 1 | 0 | Chip select output ($\overline{\text{CS3}}$) (PA7 in single-chip mode) |
| | 1 | Reserved |

- Bits 13 and 12—PA6 Mode 1, 0 (PA6MD1 and PA6MD0): These bits select the function of the PA6/TCLKA/ $\overline{\text{CS2}}$ pin.

| Bit 13: PA6MD1 | Bit 12: PA6MD0 | Description |
|-------------------|-------------------|--|
| 0 | 0 | General input/output (PA6) (initial value) |
| | 1 | MTU timer clock input (TCLKA) |
| 1 | 0 | Chip select output ($\overline{\text{CS2}}$) (PA6 in single-chip mode) |
| | 1 | Reserved |

- Bits 11 and 10—PA5 Mode 1, 0 (PA5MD1 and PA5MD0): These bits select the function of the PA5/SCK1/DREQ1/IRQ1 pin.

| Bit 11: PA5MD1 | Bit 10: PA5MD0 | Description |
|-------------------|-------------------|--|
| 0 | 0 | General input/output (PA5) (initial value) |
| | 1 | Serial clock input/output (SCK1) |
| 1 | 0 | DMA transfer request received input (DREQ1) (PA5 in single-chip mode) |
| | 1 | Interrupt request input (IRQ1) |

- Bit 9—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 8—PA4 Mode (PA4MD): Selects the function of the PA4/TxD1 pin.

| Bit 8: PA4MD | Description |
|--------------|--|
| 0 | General input/output (PA4) (initial value) |
| 1 | Transmit data output (TxD1) |

- Bit 7—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 6—PA3 Mode (PA3MD): Selects the function of the PA3/RxD1 pin.

| Bit 6: PA3MD | Description |
|--------------|--|
| 0 | General input/output (PA3) (initial value) |
| 1 | Receive data input (RxD1) |

- Bits 5 and 4—PA2 Mode 1, 0 (PA2MD1 and PA2MD0): These bits select the function of the PA2/SCK0/DREQ0/IRQ0 pin.

| Bit 5: PA2MD1 | Bit 4: PA2MD0 | Description |
|------------------|------------------|--|
| 0 | 0 | General input/output (PA2) (initial value) |
| | 1 | Serial clock input/output (SCK0) |
| 1 | 0 | DMA transfer request received input (DREQ0) (PA2 in single-chip mode) |
| | 1 | Interrupt request input (IRQ0) |

- Bit 3—Reserved: This bit always reads as 0. The write value should always be 0.

- Bit 2—PA1 Mode (PA1MD): Selects the function of the PA1/TxD0 pin.

| Bit 2: PA1MD | Description |
|--------------|--|
| 0 | General input/output (PA1) (initial value) |
| 1 | Transmit data output (TxD0) |

- Bit 1—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 0—PA0 Mode (PA0MD): Selects the function of the PA0/RxD0 pin.

| Bit 0: PA0MD | Description |
|--------------|--|
| 0 | General input/output (PA0) (initial value) |
| 1 | Receive data input (RxD0) |

16.3.3 Port B I/O Register (PBIOR)

The port B I/O register L (PBIOR) is a 16-bit read/write register that selects input or output for the ten pins of port B (eight pins in the SH7014). Bits PB9IOR–PB2IOR correspond to the PB9/ $\overline{\text{IRQ7}}$ /A21 pin to PB2/ $\overline{\text{IRQ6}}$ /RAS pin. PBIOR is enabled when the port B pins function as input/outputs (PB9–PB2). For other functions, it is disabled.

For port B pin functions PB9–PB0, a given pin in port B is an output pin if its corresponding PBIOR bit is set to 1, and an input pin if the bit is cleared to 0.

PBIOR is initialized to H'0000 by external power-on reset; however, it is not initialized for reset by WDT, standby mode, or sleep mode, so the previous data is maintained.

The settings of bits 1 and 0 of PBIOR are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to these bits; they are always read as 0, and their write value should always be 0.

| | | | | | | | | |
|----------------|------------|------------|------------|------------|------------|------------|------------|------------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | PB9 IOR | PB8 IOR |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R/W | R/W |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PB7 IOR | PB6 IOR | PB5 IOR | PB4 IOR | PB3 IOR | PB2 IOR | PB1 IOR | PB0 IOR |
| 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: * R only in the SH7014.

16.3.4 Port B Control Registers (PBCR1 and PBCR2)

PBCR1 and PBCR2 are 16-bit read/write registers that select the functions of the ten multiplexed pins of port B (eight pins in the SH7014). PBCR1 selects the functions of the top two bits of port B; PBCR2 selects the functions of the bottom eight bits of port B (six bits in the SH7014).

Port B includes bus control signals ($\overline{\text{RDWR}}$, $\overline{\text{RAS}}$, $\overline{\text{CASH}}$, $\overline{\text{CASL}}$, and $\overline{\text{WAIT}}$) and address outputs (A21 to A16), but in the SH7016 and SH7017, settings in these registers relating to the selection of these pin functions may be invalid in single-chip mode. For details, see table 16.3, Pin Functions in Each Operating Mode.

PBCR1 and PBCR2 are both initialized to H'0000 by external power-on reset but are not initialized for reset by WDT, standby mode, or sleep mode, so the previous data is maintained.

The settings of bits 2 and 0 of PBCR2 are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to these bits; they are always read as 0, and their write value should always be 0.

Port B Control Register 1 (PBCR1):

| | | | | | | | | |
|----------------|----|----|----|----|------------|------------|------------|------------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | PB9 MD1 | PB9 MD0 | PB8 MD1 | PB8 MD0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R/W | R/W | R/W | R/W |

- Bits 15-4—Reserved: These bits always read as 0. The write value should always be 0.
- Bits 3 and 2—PB9 Mode (PB9MD1 and PB9MD0): PB9MD1 and PB9MD0 select the function of the PB9/ $\overline{\text{IRQ7}}$ /A21 pin.

| Bit 3: PB9MD1 | Bit 2: PB9MD0 | Description |
|---------------|---------------|--|
| 0 | 0 | General input/output (PB9) (initial value) |
| | 1 | Interrupt request input ($\overline{\text{IRQ7}}$) |
| 1 | 0 | Address output (A21) (PB9 in single-chip mode) |
| | 1 | Reserved |

- Bits 1 and 0—PB8 Mode (PB8MD1 and PB8MD0): PB8MD1 and PB8MD0 select the function of the PB8/ $\overline{\text{IRQ6}}$ /A20/ $\overline{\text{WAIT}}$ pin.

| Bit 1: PB8MD1 | Bit 0: PB8MD0 | Description |
|---------------|---------------|--|
| 0 | 0 | General input/output (PB8) (initial value) |
| | 1 | Interrupt request input ($\overline{\text{IRQ6}}$) |
| 1 | 0 | Address output (A20) (PB8 in single-chip mode) |
| | 1 | Wait state request input ($\overline{\text{WAIT}}$) (PB8 in single-chip mode) |

Port B Control Register 2 (PBCR2):

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|--------|--------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PB7MD1 | PB7MD0 | PB6MD1 | PB6MD0 | PB5MD1 | PB5MD0 | PB4MD1 | PB4MD0 |
| 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 |
| | PB3MD1 | PB3MD0 | PB2MD1 | PB2MD0 | — | PB1MD | — | PB0MD |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R | R/W* | R | R/W* |

Note: * R only in the SH7014.

- Bits 15 and 14—PB7 Mode (PB7MD1 and PB7MD0): PB7MD1 and PB7MD0 select the function of the PB7/A19 pin.

| Bit 15: PB7MD1 | Bit 14: PB7MD0 | Description |
|-------------------|-------------------|--|
| 0 | 0 | General input/output (PB7) (initial value) |
| | 1 | Reserved |
| 1 | 0 | Address output (A19) (PB7 in single-chip mode) |
| | 1 | Reserved |

- Bits 13 and 12—PB6 Mode (PB6MD1 and PB6MD0): PB6MD1 and PB6MD0 select the function of the PB6/A18 pin.

| Bit 13: PB6MD1 | Bit 12: PB6MD0 | Description |
|-------------------|-------------------|--|
| 0 | 0 | General input/output (PB6) (initial value) |
| | 1 | Reserved |
| 1 | 0 | Address output (A18) (PB6 in single-chip mode) |
| | 1 | Reserved |

- Bits 11 and 10—PB5 Mode (PB5MD1 and PB5MD0): PB5MD1 and PB5MD0 select the function of the PB5/ $\overline{\text{IRQ3}}$ /RDWR pin.

| Bit 11: PB5MD1 | Bit 10: PB5MD0 | Description |
|-------------------|-------------------|--|
| 0 | 0 | General input/output (PB5) (initial value) |
| | 1 | Interrupt request input ($\overline{\text{IRQ3}}$) |
| 1 | 0 | Reserved |
| | 1 | Read/write output (RDWR) (PB5 in single-chip mode) |

- Bits 9 and 8—PB4 Mode (PB4MD1 and PB4MD0): PB4MD1 and PB4MD0 select the function of the PB4/ $\overline{\text{IRQ2}}$ / $\overline{\text{CASH}}$ pin.

| Bit 9: PB4MD1 | Bit 8: PB4MD0 | Description |
|---------------|---------------|--|
| 0 | 0 | General input/output (PB4) (initial value) |
| | 1 | Interrupt request input ($\overline{\text{IRQ2}}$) |
| 1 | 0 | Reserved |
| | 1 | Column address strobe ($\overline{\text{CASH}}$) (PB4 in single-chip mode) |

- Bits 7 and 6—PB3 Mode (PB3MD1 and PB3MD0): PB3MD1 and PB3MD0 select the function of the PB3/ $\overline{\text{IRQ1}}$ / $\overline{\text{CASL}}$ pin.

| Bit 7: PB3MD1 | Bit 6: PB3MD0 | Description |
|---------------|---------------|--|
| 0 | 0 | General input/output (PB3) (initial value) |
| | 1 | Interrupt request input ($\overline{\text{IRQ1}}$) |
| 1 | 0 | Reserved |
| | 1 | Column address strobe ($\overline{\text{CASL}}$) (PB3 in single-chip mode) |

- Bits 5 and 4—PB2 Mode (PB2MD1 and PB2MD0): PB2MD1 and PB2MD0 select the function of the PB2/ $\overline{\text{IRQ0}}$ /RAS pin.

| Bit 5: PB2MD1 | Bit 4: PB2MD0 | Description |
|---------------|---------------|--|
| 0 | 0 | General input/output (PB2) (initial value) |
| | 1 | Interrupt request input ($\overline{\text{IRQ0}}$) |
| 1 | 0 | Reserved |
| | 1 | Row address strobe ($\overline{\text{RAS}}$) (PB2 in single-chip mode) |

- Bit 3—Reserved: This bit always reads as 0. The write value should always be 0.

- Bit 2—PB1 Mode (PB1MD) –**SH7016, SH7017**–: Selects the function of the PB1/A17 pin.

| Bit 2: PB1MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|--|
| 0 | General input/output (PB1) (initial value) (A17 in on-chip ROM invalid mode) |
| 1 | Address output (A17) (PB1 in single chip mode) |

- Bit 1—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 0—PB0 Mode (PB0MD) –**SH7016, SH7017**–: Selects the function of the PB0/A16 pin.

| Bit 0: PA0MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|--|
| 0 | General input/output (PB0) (initial value) (A16 in on-chip ROM invalid mode) |
| 1 | Address output (A16) (PB0 in single chip mode) |

16.3.5 Port C I/O Register (PCIOR) –SH7016, SH7017–

The port C I/O register (PCIOR) is a 16-bit read/write register that selects input or output for the 16 port C pins. Bits PC15IOR–PC0IOR correspond to pins PC15/A15 to PC0/A0. PCIOR is enabled when the port C pins function as general input/outputs (PC15–PC0). For other functions, it is disabled.

When the port C pin functions are as PC15–PC0, a given pin in port C is an output pin if its corresponding PCIOR bit is set to 1, and an input pin if the bit is cleared to 0.

PCIOR is initialized to H'0000 by external power-on reset; however, it is not initialized for reset by WDT, standby mode, or sleep mode, so the previous values are maintained.

The settings in this register are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to this register, and it should not be read or written to.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|-------------|-------------|-------------|-------------|-------------|-------------|------------|------------|
| | PC15 IOR | PC14 IOR | PC13 IOR | PC12 IOR | PC11 IOR | PC10 IOR | PC9 IOR | PC8 IOR |
| 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 |
|----------------|------------|------------|------------|------------|------------|------------|------------|------------|
| | PC7 IOR | PC6 IOR | PC5 IOR | PC4 IOR | PC3 IOR | PC2 IOR | PC1 IOR | PC0 IOR |
| 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 |

16.3.6 Port C Control Register (PCCR) –SH7016, SH7017–

PCCR is a 16-bit read/write register that selects the functions for the sixteen port C multiplexed pins. There are instances when these register settings will be ignored, depending on the operation mode. Refer to table 16.3, Pin Arrangement by Mode, for details.

PCCR is initialized to H'0000 by power-on resets but is not initialized for reset by WDT, standby mode, or sleep mode, so the previous data is maintained.

The settings in this register are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to this register, and it should not be read or written to.

| | | | | | | | | |
|----------------|------------|------------|------------|------------|------------|------------|-----------|-----------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PC15 MD | PC14 MD | PC13 MD | PC12 MD | PC11 MD | PC10 MD | PC9 MD | PC8 MD |
| 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 |
| | PC7 MD | PC6 MD | PC5 MD | PC4 MD | PC3 MD | PC2 MD | PC1 MD | PC0 MD |
| 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 15—PC15 Mode (PC15MD): Selects the function of the PC15/A15 pin.

Bit 15: PC15MD Description

| | |
|---|---|
| 0 | General input/output (PC15) (initial value) (A15 in on-chip ROM invalid mode) |
| 1 | Address output (A15) (PC15 in single chip mode) |

- Bit 14—PC14 Mode (PC14MD): Selects the function of the PC14/A14 pin.

Bit 14: PC14MD Description

| | |
|---|---|
| 0 | General input/output (PC14) (initial value) (A14 in on-chip ROM invalid mode) |
| 1 | Address output (A14) (PC14 in single chip mode) |

- Bit 13—PC13 Mode (PC13MD): Selects the function of the PC13/A13 pin.

| Bit 13: PC13MD | Description |
|----------------|-------------|
|----------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC13) (initial value) (A13 in on-chip ROM invalid mode) |
| 1 | Address output (A13) (PC13 in single chip mode) |

- Bit 12—PC12 Mode (PC12MD): Selects the function of the PC12/A12 pin.

| Bit 12: PC12MD | Description |
|----------------|-------------|
|----------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC12) (initial value) (A12 in on-chip ROM invalid mode) |
| 1 | Address output (A12) (PC12 in single chip mode) |

- Bit 11—PC11 Mode (PC11MD): Selects the function of the PC11/A11 pin.

| Bit 11: PC11MD | Description |
|----------------|-------------|
|----------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC11) (initial value) (A11 in on-chip ROM invalid mode) |
| 1 | Address output (A11) (PC11 in single chip mode) |

- Bit 10—PC10 Mode (PC10MD): Selects the function of the PC10/A10 pin.

| Bit 10: PC10MD | Description |
|----------------|-------------|
|----------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC10) (initial value) (A10 in on-chip ROM invalid mode) |
| 1 | Address output (A10) (PC10 in single chip mode) |

- Bit 9—PC9 Mode (PC9MD): Selects the function of the PC9/A9 pin.

| Bit 9: PC9MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC9) (initial value) (A9 in on-chip ROM invalid mode) |
| 1 | Address output (A9) (PC9 in single chip mode) |

- Bit 8—PC8 Mode (PC8MD): Selects the function of the PC8/A8 pin.

| Bit 8: PC8MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC8) (initial value) (A8 in on-chip ROM invalid mode) |
| 1 | Address output (A8) (PC8 in single chip mode) |

- Bit 7—PC7 Mode (PC7MD): Selects the function of the PC7/A7 pin.

| Bit 7: PC7MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC7) (initial value) (A7 in on-chip ROM invalid mode) |
| 1 | Address output (A7) (PC7 in single chip mode) |

- Bit 6—PC6 Mode (PC6MD): Selects the function of the PC6/A6 pin.

| Bit 6: PC6MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC6) (initial value) (A6 in on-chip ROM invalid mode) |
| 1 | Address output (A6) (PC6 in single chip mode) |

- Bit 5—PC5 Mode (PC5MD): Selects the function of the PC5/A5 pin.

| Bit 5: PC5MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC5) (initial value) (A5 in on-chip ROM invalid mode) |
| 1 | Address output (A5) (PC5 in single chip mode) |

- Bit 4—PC4 Mode (PC4MD): Selects the function of the PC4/A4 pin.

| Bit 4: PC4MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC4) (initial value) (A4 in on-chip ROM invalid mode) |
| 1 | Address output (A4) (PC4 in single chip mode) |

- Bit 3—PC3 Mode (PC3MD): Selects the function of the PC3/A3 pin.

| Bit 3: PC3MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC3) (initial value) (A3 in on-chip ROM invalid mode) |
| 1 | Address output (A3) (PC3 in single chip mode) |

- Bit 2—PC2 Mode (PC2MD): Selects the function of the PC2/A2 pin.

| Bit 2: PC2MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PC2) (initial value) (A2 in on-chip ROM invalid mode) |
| 1 | Address output (A2) (PC2 in single chip mode) |

- Bit 1—PC1 Mode (PC1MD): Selects the function of the PC1/A1 pin.

| Bit 1: PC1MD | Description |
|--------------|---|
| 0 | General input/output (PC1) (initial value) (A1 in on-chip ROM invalid mode) |
| 1 | Address output (A1) (PC1 in single chip mode) |

- Bit 0—PC0 Mode (PC0MD): Selects the function of the PC0/A0 pin.

| Bit 0: PC0MD | Description |
|--------------|---|
| 0 | General input/output (PC0) (initial value) (A0 in on-chip ROM invalid mode) |
| 1 | Address output (A0) (PC0 in single chip mode) |

16.3.7 Port D I/O Register L (PDIORL) –SH7016, SH7017–

The port D I/O register L (PDIORL) is a 16-bit read/write register that selects input or output for the sixteen port D pins. Bits PD15IOR–PD0IOR correspond to the PD15/D15 pin to PD0/D0 pin. PDIORL is enabled when the port D pins function as general input/outputs (PD15–PD0). For other functions, it is disabled.

For port D pin functions PD15–PD0, a given pin in port D is an output pin if its corresponding PDIORL bit is set to 1, and an input pin if the bit is cleared to 0.

PDIORL is initialized to H'0000 by external power-on reset; however, it is not initialized for reset by WDT, standby mode, or sleep mode, so the previous data is maintained.

The settings in this register are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to this register, and it should not be read or written to.

| | | | | | | | | |
|----------------|-------------|-------------|-------------|-------------|-------------|-------------|------------|------------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PD15 IOR | PD14 IOR | PD13 IOR | PD12 IOR | PD11 IOR | PD10 IOR | PD9 IOR | PD8 IOR |
| 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 |
| | PD7 IOR | PD6 IOR | PD5 IOR | PD4 IOR | PD3 IOR | PD2 IOR | PD1 IOR | PD0 IOR |
| 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 |

16.3.8 Port D Control Register L (PDCRL) –SH7016, SH7017–

PDCRL is a 16-bit read/write register that selects the multiplexed pin functions for the least significant sixteen port D pins. There are instances when these register settings will be ignored, depending on the operation mode.

On-Chip ROM-Disabled Extended Mode:

- Mode 0 (8-bit bus): Port D pins are data I/O pins; PDCRL settings are disabled.
- Mode 1 (16-bit bus): Port D pins are data I/O pins; PDCRL settings are disabled.

On-Chip ROM-Enabled Extended Mode: The port D pins are shared as data I/O pins and general I/O pins; PDCRL settings are enabled.

Single Chip Mode: The port D pins are general I/O pins; PDCRL settings are disabled.

PDCRL is initialized to H'0000 by external power-on reset but is not initialized for reset by WDT, standby mode, or sleep mode, so the previous data is maintained.

The settings in this register are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to this register, and it should not be read or written to.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|------------|------------|------------|------------|------------|------------|-----------|-----------|
| | PD15 MD | PD14 MD | PD13 MD | PD12 MD | PD11 MD | PD10 MD | PD9 MD | PD8 MD |
| 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 |
| | PD7 MD | PD6 MD | PD5 MD | PD4 MD | PD3 MD | PD2 MD | PD1 MD | PD0 MD |
| 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 15—PD15 Mode (PD15MD): Selects the function of the PD15/D15 pin.

Bit 15: PD15MD Description

| Bit | Description |
|-----|---|
| 0 | General input/output (PD15) (initial value) (D15 in on-chip ROM invalid mode) |
| 1 | Data input/output (D15) (PD15 in single chip mode) |

- Bit 14—PD14 Mode (PD14MD): Selects the function of the PD14/D14 pin.

Bit 14: PD14MD Description

| | |
|---|---|
| 0 | General input/output (PD14) (initial value) (D14 in on-chip ROM invalid mode) |
| 1 | Data input/output (D14) (PD14 in single chip mode) |

- Bit 13—PD13 Mode (PD13MD): Selects the function of the PD13/D13 pin.

Bit 13: PD13MD Description

| | |
|---|---|
| 0 | General input/output (PD13) (initial value) (D13 in on-chip ROM invalid mode) |
| 1 | Data input/output (D13) (PD13 in single chip mode) |

- Bit 12—PD12 Mode (PD12MD): Selects the function of the PD12/D12 pin.

Bit 12: PD12MD Description

| | |
|---|---|
| 0 | General input/output (PD12) (initial value) (D12 in on-chip ROM invalid mode) |
| 1 | Data input/output (D12) (PD12 in single chip mode) |

- Bit 11—PD11 Mode (PD11MD): Selects the function of the PD11/D11 pin.

Bit 11: PD11MD Description

| | |
|---|---|
| 0 | General input/output (PD11) (initial value) (D11 in on-chip ROM invalid mode) |
| 1 | Data input/output (D11) (PD11 in single chip mode) |

- Bit 10—PD10 Mode (PD10MD): Selects the function of the PD10/D10 pin.

Bit 10: PD10MD Description

| | |
|---|---|
| 0 | General input/output (PD10) (initial value) (D10 in on-chip ROM invalid mode) |
| 1 | Data input/output (D10) (PD10 in single chip mode) |

- Bit 9—PD9 Mode (PD9MD): Selects the function of the PD9/D9 pin.

Bit 9: PD9MD Description

| | |
|---|---|
| 0 | General input/output (PD9) (initial value) (D9 in on-chip ROM invalid mode) |
| 1 | Data input/output (D9) (PD9 in single chip mode) |

- Bit 8—PD8 Mode (PD8MD): Selects the function of the PD8/D8 pin.

| Bit 8: PD8MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PD8) (initial value) (D8 in on-chip ROM invalid mode) |
| 1 | Data input/output (D8) (PD8 in single chip mode) |

- Bit 7—PD7 Mode (PD7MD): Selects the function of the PD7/D7 pin.

| Bit 7: PD7MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PD7) (initial value) (D7 in on-chip ROM invalid mode) |
| 1 | Data input/output (D7) (PD7 in single chip mode) |

- Bit 6—PD6 Mode (PD6MD): Selects the function of the PD6/D6 pin.

| Bit 6: PD6MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PD6) (initial value) (D6 in on-chip ROM invalid mode) |
| 1 | Data input/output (D6) (PD6 in single chip mode) |

- Bit 5—PD5 Mode (PD5MD): Selects the function of the PD5/D5 pin.

| Bit 5: PD5MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PD5) (initial value) (D5 in on-chip ROM invalid mode) |
| 1 | Data input/output (D5) (PD5 in single chip mode) |

- Bit 4—PD4 Mode (PD4MD): Selects the function of the PD4/D4 pin.

| Bit 4: PD4MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PD4) (initial value) (D4 in on-chip ROM invalid mode) |
| 1 | Data input/output (D4) (PD4 in single chip mode) |

- Bit 3—PD3 Mode (PD3MD): Selects the function of the PD3/D3 pin.

| Bit 3: PD3MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|---|
| 0 | General input/output (PD3) (initial value) (D3 in on-chip ROM invalid mode) |
| 1 | Data input/output (D3) (PD3 in single chip mode) |

- Bit 2—PD2 Mode (PD2MD): Selects the function of the PD2/D2 pin.

| Bit 2: PD2MD | Description |
|---------------------|---|
| 0 | General input/output (PD2) (initial value) (D2 in on-chip ROM invalid mode) |
| 1 | Data input/output (D2) (PD2 in single chip mode) |

- Bit 1—PD1 Mode (PD1MD): Selects the function of the PD1/D1 pin.

| Bit 1: PD1MD | Description |
|---------------------|---|
| 0 | General input/output (PD1) (initial value) (D1 in on-chip ROM invalid mode) |
| 1 | Data input/output (D1) (PD1 in single chip mode) |

- Bit 0—PD0 Mode (PD0MD): Selects the function of the PD0/D0 pin.

| Bit 0: PD0MD | Description |
|---------------------|---|
| 0 | General input/output (PD0) (initial value) (D0 in on-chip ROM invalid mode) |
| 1 | Data input/output (D0) (PD0 in single chip mode) |

16.3.9 Port E I/O Register (PEIOR)

The port E I/O register (PEIOR) is a 16-bit read/write register that selects input or output for the 16 port E pins. Bits PE15IOR–PE0IOR correspond to pins PE15/DACK1 to PE0/TIOC0A/ $\overline{\text{DREQ}}_0$. PEIOR is enabled when the port E pins function as general input/outputs (PE15–PE0) or TIOC pin of the MTU. For other functions, it is disabled.

When the port E pin functions are as PE15–PE0, or TIOC pin of the MTU, a given pin in port E is an output pin if its corresponding PEIOR bit is set to 1, and an input pin if the bit is cleared to 0.

PEIOR is initialized to H'0000 by external power-on reset; however, it is not initialized for reset by WDT, standby mode, or sleep mode, so the previous data is maintained.

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|-------------|-------------|-------------|-------------|-------------|-------------|------------|------------|
| | PE15 IOR | PE14 IOR | PE13 IOR | PE12 IOR | PE11 IOR | PE10 IOR | PE9 IOR | PE8 IOR |
| 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 |
|----------------|------------|------------|------------|------------|------------|------------|------------|------------|
| | PE7 IOR | PE6 IOR | PE5 IOR | PE4 IOR | PE3 IOR | PE2 IOR | PE1 IOR | PE0 IOR |
| 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 |

16.3.10 Port E Control Registers 1, 2 (PECR1 and PECR2)

PECR1 and PECR2 are 16-bit read/write registers that select the functions of the sixteen multiplexed pins of port E. PECR1 selects the functions of the upper eight bit pins of port E; PECR2 selects the function of the lower eight bit pins of port E.

Port E includes a bus control signal (\overline{AH}) and DMAC control signals (DACK1, DACK0, DRAK1, and DRAK0), but in the SH7016 and SH7017, settings in these registers relating to the selection of these pin functions may be invalid in single-chip mode. For details, see table 16.2, Pin Functions in Each Operating Mode.

PECR1 and PECR2 are both initialized to H'0000 by external power-on reset but are not initialized for reset by WDT, standby mode, or sleep mode, so the previous data is maintained.

Port E Control Register 1 (PECR1):

| | | | | | | | | |
|----------------|-------------|-------------|-------------|-------------|----|----|---|---|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PE15 MD1 | PE15 MD0 | PE14 MD1 | PE14 MD0 | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R/W | R/W | R/W | R/W | R | R | R | R |
| | | | | | | | | |
| 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 |

- Bits 15 and 14—PE15 Mode 1, 0 (PE15MD1 and PE15MD0): These bits select the function of the PE15/DACK1 pin.

| Bit 15: PE15MD1 | Bit 14: PE15MD0 | Description |
|--------------------|--------------------|--|
| 0 | 0 | Input/output (PE15) (initial value) |
| | 1 | Reserved |
| 1 | 0 | DMAC request received output (DACK1) (PE15 in single-chip mode) |
| | 1 | Reserved |

- Bits 13 and 12—PE14 Mode 1, 0 (PE14MD1 and PE14MD0): These bits select the function of the PE14/DACK0/ $\overline{\text{AH}}$ pin.

| Bit 13: PE14MD1 | Bit 12: PE14MD0 | Description |
|--------------------|--------------------|---|
| 0 | 0 | Input/output (PE14) (initial value) |
| | 1 | Reserved |
| 1 | 0 | DMAC request received output (DACK0) (PE14 in single-chip mode) |
| | 1 | Address hold output ($\overline{\text{AH}}$) (PE14 in single-chip mode) |

- Bits 11–0—Reserved: These bits always reads as 0. The write values should always be 0.

Port E Control Register 2 (PECR2):

| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|----------------|----|-------|----|-------|----|-------|---|-------|
| | — | PE7MD | — | PE6MD | — | PE5MD | — | PE4MD |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R/W | R | R/W | R | R/W | R | R/W |

| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------|------------|------------|------------|------------|------------|------------|------------|------------|
| | PE3 MD1 | PE3 MD0 | PE2 MD1 | PE2 MD0 | PE1 MD1 | PE1 MD0 | PE0 MD1 | PE0 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 |

- Bit 15—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 14—PE7 Mode (PE7MD): Selects the function of the PE7/TIOC2B pin.

| Bit 14: PE7MD | Description |
|---------------|--|
| 0 | General input/output (PE7) (initial value) |
| 1 | MTU input capture input/output compare output (TIOC2B) |

- Bit 13 —Reserved: This bit always reads as 0. The write value should always be 0.

- Bit 12—PE6 Mode (PE6MD): Selects the function of the PE6/TIOC2A pin.

| Bit 12: PE6MD | Description |
|---------------|-------------|
|---------------|-------------|

| | |
|---|--|
| 0 | General input/output (PE6) (initial value) |
| 1 | MTU input capture input/output compare output (TIOC2A) |

- Bit 11—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 10—PE5 Mode (PE5MD): Selects the function of the PE5/TIOC1B pin.

| Bit 10: PE5MD | Description |
|---------------|-------------|
|---------------|-------------|

| | |
|---|--|
| 0 | General input/output (PE5) (initial value) |
| 1 | MTU input capture input/output compare output (TIOC1B) |

- Bit 9—Reserved: This bit always reads as 0. The write value should always be 0.
- Bit 8—PE4 Mode (PE4MD): Selects the function of the PE4/TIOC1A pin.

| Bit 8: PE4MD | Description |
|--------------|-------------|
|--------------|-------------|

| | |
|---|--|
| 0 | General input/output (PE4) (initial value) |
| 1 | MTU input capture input/output compare output (TIOC1A) |

- Bits 7 and 6—PE3 Mode 1, 0 (PE3MD1 and PE3MD0): These bits select the function of the PE3/TIOC0D/DRAK1 pin.

| Bit 7: PE3MD1 | Bit 6: PE3MD0 | Description |
|------------------|------------------|---|
| 0 | 0 | General input/output (PE3) (initial value) |
| | 1 | MTU input capture input/output compare output (TIOC0D) |
| 1 | 0 | DREQ1 request received output (DRAK1) (PE3 in single-chip mode) |
| | 1 | Reserved |

- Bits 5 and 4—PE2 Mode 1, 0 (PE2MD1 and PE2MD0): These bits select the function of the PE2/TIOC0C/DREQ1 pin.

| Bit 5: PE2MD1 | Bit 4: PE2MD0 | Description |
|------------------|------------------|---|
| 0 | 0 | General input/output (PE2) (initial value) |
| | 1 | MTU input capture input/output compare output (TIOC0C) |
| 1 | 0 | $\overline{\text{DREQ1}}$ request receive input (PE2 in single-chip mode) |
| | 1 | Reserved |

- Bits 3 and 2—PE1 Mode 1, 0 (PE1MD1 and PE1MD0): These bits select the function of the PE1/TIOC0B/DRAK0 pin.

| Bit 3: PE1MD1 | Bit 2: PE1MD0 | Description |
|------------------|------------------|---|
| 0 | 0 | General input/output (PE1) (initial value) |
| | 1 | MTU input capture input/output compare output (TIOC0B) |
| 1 | 0 | $\overline{\text{DREQ0}}$ request received output (DRAK0) (PE1 in single-chip mode) |
| | 1 | Reserved |

- Bits 1 and 0—PE0 Mode 1, 0 (PE0MD1 and PE0MD0): These bits select the function of the PE0/TIOC0A/DREQ0 pin.

| Bit 1: PE0MD1 | Bit 0: PE0MD0 | Description |
|------------------|------------------|---|
| 0 | 0 | General input/output (PE0) (initial value) |
| | 1 | MTU input capture input/output compare output (TIOC0A) |
| 1 | 0 | $\overline{\text{DREQ0}}$ request receive input (PE0 in single-chip mode) |
| | 1 | Reserved |

Section 17 I/O Ports (I/O)

17.1 Overview

The SH7016 and SH7017 have six ports, A to F, and the SH7014 has four ports, A, B, E, and F. The pins of the ports are multiplexed for use as general-purpose I/Os (the port F pins are general input) or for other functions. Use the pin function controller (PFC) to select the function of multiplexed pins. The ports each have one data register for storing pin data. The initialize function after power-on reset differs depending on the operating mode of each pin. See tables 16.2 and 16.3, Pin Arrangement by Mode, for details.

17.2 Port A

Port A is a 16-pin input/output port (11-pin in the SH7014) as shown in table 17.1.

Table 17.1 Port A

| ROM Disabled Extended Mode (Modes 0, 1) | ROM Enabled Extended Mode (Mode 2)* | Single Chip Mode* |
|---|---|--|
| PA15 (I/O)/CK (output) | PA15 (I/O)/CK (output) | PA15 (I/O)/CK (output) |
| \overline{RD} (output) | PA14 (I/O)/ \overline{RD} (output) | PA14 (I/O) |
| \overline{WRH} (output) | PA13 (I/O)/ \overline{WRH} (output) | PA13 (I/O) |
| \overline{WRL} (output) | PA12 (I/O)/ \overline{WRL} (output) | PA12 (I/O) |
| $\overline{CS1}$ (output) | PA11 (I/O)/ $\overline{CS1}$ (output) | PA11 (I/O) |
| $\overline{CS0}$ (output) | PA10 (I/O)/ $\overline{CS0}$ (output) | PA10 (I/O) |
| PA9 (I/O)/TCLKD (input)/ $\overline{IRQ3}$ (input) | PA9 (I/O)/TCLKD (input)/ $\overline{IRQ3}$ (input) | PA9 (I/O)/TCLKD (input)/ $\overline{IRQ3}$ (input) |
| PA8 (I/O)/TCLKC (input)/ $\overline{IRQ2}$ (input) | PA8 (I/O)/TCLKC (input)/ $\overline{IRQ2}$ (input) | PA8 (I/O)/TCLKC (input)/ $\overline{IRQ2}$ (input) |
| PA7 (I/O)/TCLKB (input)/ $\overline{CS3}$ (input) | PA7 (I/O)/TCLKB (input)/ $\overline{CS3}$ (input) | PA7 (I/O)/TCLKB (input) |
| PA6 (I/O)/TCLKA (input)/ $\overline{CS2}$ (input) | PA6 (I/O)/TCLKA (input)/ $\overline{CS2}$ (input) | PA6 (I/O)/TCLKA (input) |
| PA5 (I/O)/SCK1 (I/O)/ $\overline{DREQ1}$ (input)/ $\overline{IRQ1}$ (input) | PA5 (I/O)/SCK1 (I/O)/ $\overline{DREQ1}$ (input)/ $\overline{IRQ1}$ (input) | PA5 (I/O)/SCK1 (I/O)/ $\overline{IRQ1}$ (input) |
| PA4 (I/O)/TXD1 (output) | PA4 (I/O)/TXD1 (output) | PA4 (I/O)/TXD1 (output) |
| PA3 (I/O)/RXD1 (input) | PA3 (I/O)/RXD1 (input) | PA3 (I/O)/RXD1 (input) |
| PA2 (I/O)/SCK0 (I/O)/ $\overline{DREQ0}$ (input)/ $\overline{IRQ0}$ (input) | PA2 (I/O)/SCK0 (I/O)/ $\overline{DREQ0}$ (input)/ $\overline{IRQ0}$ (input) | PA2 (I/O)/SCK0 (I/O)/ $\overline{IRQ0}$ (input) |
| PA1 (I/O)/TXD0 (output) | PA1 (I/O)/TXD0 (output) | PA1 (I/O)/TXD0 (output) |
| PA0 (I/O)/RXD0 (input) | PA0 (I/O)/RXD0 (input) | PA0 (I/O)/RXD0 (input) |

Note: SH7016, SH7017 only.

17.2.1 Register Configuration

Table 17.2 summarizes the port A register.

Table 17.2 Port A Register

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|------------------------|---------------------|------------|----------------------|--------------------------|--------------------|
| Port A data register L | PADRL | R/W | H'0000 | H'FFFF8382 H'FFFF8383 | 8, 16, 32 |

17.2.2 Port A Data Register L (PADRL)

PADRL is a 16-bit read/write register that stores data for port A. The bits PA15DR–PA0DR correspond to the PA15/CK–PA0/RXD0 pins. When the pins are used as ordinary outputs, they will output whatever value is written in the PADRL; when PADRL is read, the register value will be output regardless of the pin status. When the pins are used as ordinary inputs, the pin status rather than the register value is read directly when PADRL is read. When a value is written to PADRL, that value can be written into PADRL, but it will not affect the pin status. Table 17.3 shows the read/write operations of the port A data register.

PADRL is initialized by an external power-on reset. However, PADRL is not initialized for manual reset, reset by WDT, standby mode, or sleep mode.

The settings of bits 14 to 10 of this register are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to bits 14 to 10; these bits are always read as 0, and their write value should always be 0.

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PA15DR | PA14DR | PA13DR | PA12DR | PA11DR | PA10DR | PA9DR | PA8DR |
| 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 |
| | PA7DR | PA6DR | PA5DR | PA4DR | PA3DR | PA2DR | PA1DR | PA0DR |
| 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: * R only in the SH7014.

Table 17.3 Read/Write Operation of the Port A Data Register (PADR)

| PAIOR | Pin Status | Read | Write |
|-------|-----------------|------------|---|
| 0 | Ordinary input | Pin status | Can write to PADR, but it has no effect on pin status |
| | Other function | Pin status | Can write to PADR, but it has no effect on pin status |
| 1 | Ordinary output | PADR value | Value written is output by pin |
| | Other function | PADR value | Can write to PADR, but it has no effect on pin status |

17.3 Port B

Port B is a 10-pin input/output port (8-pin in the SH7014) as shown in table 17.4.

Table 17.4 Port B

| ROM Disabled Extended Mode (Modes 0, 1) | ROM Enabled Extended Mode (Mode 2)* | Single Chip Mode* |
|--|--|---|
| PB9 (I/O)/ $\overline{\text{IRQ7}}$ (input)/A21 (output) | PB9 (I/O)/ $\overline{\text{IRQ7}}$ (input)/A21 (output) | PB9 (I/O)/ $\overline{\text{IRQ7}}$ (input) |
| PB8 (I/O)/ $\overline{\text{IRQ6}}$ (input)/A20 (output)/ $\overline{\text{WAIT}}$ (input) | PB8 (I/O)/ $\overline{\text{IRQ6}}$ (input)/A20 (output)/ $\overline{\text{WAIT}}$ (input) | PB8 (I/O)/ $\overline{\text{IRQ6}}$ (input) |
| PB7 (I/O)/A19 (output) | PB7 (I/O)/A19 (output) | PB7 (I/O) |
| PB6 (I/O)/A18 (output) | PB6 (I/O)/A18 (output) | PB6 (I/O) |
| PB5 (I/O)/ $\overline{\text{IRQ3}}$ (input)/RDWR (output) | PB5 (I/O)/ $\overline{\text{IRQ3}}$ (input)/RDWR (output) | PB5 (I/O)/ $\overline{\text{IRQ3}}$ (input) |
| PB4 (I/O)/ $\overline{\text{IRQ2}}$ (input)/ $\overline{\text{CASH}}$ (output) | PB4 (I/O)/ $\overline{\text{IRQ2}}$ (input)/ $\overline{\text{CASH}}$ (output) | PB4 (I/O)/ $\overline{\text{IRQ2}}$ (input) |
| PB3 (I/O)/ $\overline{\text{IRQ1}}$ (input)/ $\overline{\text{CASL}}$ (output) | PB3 (I/O)/ $\overline{\text{IRQ1}}$ (input)/ $\overline{\text{CASL}}$ (output) | PB3 (I/O)/ $\overline{\text{IRQ1}}$ (input) |
| PB2 (I/O)/ $\overline{\text{IRQ0}}$ (input)/ $\overline{\text{RAS}}$ (output) | PB2 (I/O)/ $\overline{\text{IRQ0}}$ (input)/ $\overline{\text{RAS}}$ (output) | PB2 (I/O)/ $\overline{\text{IRQ0}}$ (input) |
| A17 (output) | PB1 (I/O)/A17 (output) | PB1 (I/O) |
| A16 (output) | PB0 (I/O)/A16 (output) | PB0 (I/O) |

Note: SH7016 and SH7017 only.

17.3.1 Register Configuration

Table 17.5 summarizes the port B register.

Table 17.5 Port B Register

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|----------------------|---------------------|------------|----------------------|--------------------------|--------------------|
| Port B data register | PBDR | R/W | H'0000 | H'FFFF8390 H'FFFF8391 | 8, 16, 32 |

17.3.2 Port B Data Register (PBDR)

PBDR is a 16-bit read/write register that stores data for port B. The bits PB9DR–PB0DR correspond to the PB9/ $\overline{\text{IRQ7}}$ /A21–PB0/A16 pins. When the pins are used as ordinary outputs, they will output whatever value is written in the PBDR; when PBDR is read, the register value will be read regardless of the pin status. When the pins are used as ordinary inputs, the pin status rather than the register value is read directly when PBDR is read. When a value is written to PBDR, that value can be written into PBDR, but it will not affect the pin status. table 17.6 shows the read/write operations of the port B data register.

PBDR is initialized by an external power-on reset. However, PBDR is not initialized for a manual reset, reset by WDT, standby mode, or sleep mode.

The settings of bits 1 and 0 of this register are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to these bits; they are always read as 0, and their write value should always be 0.

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | PB9DR | PB8DR |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R/W | R/W |
| | | | | | | | | |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR | PB0DR |
| 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: * R only in the SH7014.

Table 17.6 Read/Write Operation of the Port B Data Register (PBDR)

| PBIOR | Pin Status | Read | Write |
|-------|-----------------|------------|---|
| 0 | Ordinary input | Pin status | Can write to PBDR, but it has no effect on pin status |
| | Other function | Pin status | Can write to PBDR, but it has no effect on pin status |
| 1 | Ordinary output | PBDR value | Value written is output by pin |
| | Other function | PBDR value | Can write to PBDR, but it has no effect on pin status |

17.4 Port C –SH7016, SH7017–

Port C is a 16 pin input/output port as listed in table 17.7. There is no port C in the SH7014.

Table 17.7 Port C

| ROM Disabled Extended Mode (Modes 0, 1) | ROM Enabled Extended Mode (Mode 2) | Single Chip Mode |
|--|---|-------------------------|
| A15 (output) | PC15 (I/O)/A15 (output) | PC15 (I/O) |
| A14 (output) | PC14 (I/O)/A14 (output) | PC14 (I/O) |
| A13 (output) | PC13 (I/O)/A13 (output) | PC13 (I/O) |
| A12 (output) | PC12 (I/O)/A12 (output) | PC12 (I/O) |
| A11 (output) | PC11 (I/O)/A11 (output) | PC11 (I/O) |
| A10 (output) | PC10 (I/O)/A10 (output) | PC10 (I/O) |
| A9 (output) | PC9 (I/O)/A9 (output) | PC9 (I/O) |
| A8 (output) | PC8 (I/O)/A8 (output) | PC8 (I/O) |
| A7 (output) | PC7 (I/O)/A7 (output) | PC7 (I/O) |
| A6 (output) | PC6 (I/O)/A6 (output) | PC6 (I/O) |
| A5 (output) | PC5 (I/O)/A5 (output) | PC5 (I/O) |
| A4 (output) | PC4 (I/O)/A4 (output) | PC4 (I/O) |
| A3 (output) | PC3 (I/O)/A3 (output) | PC3 (I/O) |
| A2 (output) | PC2 (I/O)/A2 (output) | PC2 (I/O) |
| A1 (output) | PC1 (I/O)/A1 (output) | PC1 (I/O) |
| A0 (output) | PC0 (I/O)/A0 (output) | PC0 (I/O) |

17.4.1 Register Configuration

Table 17.8 summarizes the port C register.

Table 17.8 Port C Register

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|----------------------|---------------------|------------|----------------------|--------------------------|--------------------|
| Port C data register | PCDR | R/W | H'0000 | H'FFFF8392 H'FFFF8393 | 8, 16, 32 |

17.4.2 Port C Data Register (PCDR)

PCDR is a 16-bit read/write register that stores data for port C. The bits PC15DR–PC0DR correspond to the PC15/A15–PC0/A0 pins. When the pins are used as ordinary outputs, they will output whatever value is written in the PCDR; when PCDR is read, the register value will be read regardless of the pin status. When the pins are used as ordinary inputs, the pin status rather than the register value is read directly when PCDR is read. When a value is written to PCDR, that value can be written into PCDR, but it will not affect the pin status. Table 17.9 shows the read/write operations of the port C data register.

PCDR is initialized by an external power-on reset. However, PCDR is not initialized for a manual reset, reset by WDT, standby mode, or sleep mode.

The settings in this register are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to this register, and it should not be read or written to.

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PC15DR | PC14DR | PC13DR | PC12DR | PC11DR | PC10DR | PC9DR | PC8DR |
| 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 |
| | PC7DR | PC6DR | PC5DR | PC4DR | PC3DR | PC2DR | PC1DR | PC0DR |
| 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 |

Table 17.9 Read/Write Operation of the Port C Data Register (PCDR)

| PCIOR | Pin Status | Read | Write |
|-------|-----------------|------------|---|
| 0 | Ordinary input | Pin status | Can write to PCDR, but it has no effect on pin status |
| | Other function | Pin status | Can write to PCDR, but it has no effect on pin status |
| 1 | Ordinary output | PCDR value | Value written is output by pin |
| | Other function | PCDR value | Can write to PCDR, but it has no effect on pin status |

17.5 Port D –SH7016, SH7017–

Port D is a 16-pin input/output port, as shown in table 17.10. There is no port D in the SH7014.

Table 17.10 Port D

| Extended Mode Without ROM (Mode 0) | Extended Mode Without ROM (Mode 1) | Extended Mode With ROM (Mode 2) | Single Chip Mode |
|------------------------------------|------------------------------------|---------------------------------|------------------|
| D15 (I/O) | D15 (I/O) | PD15 (I/O)/D15 (I/O) | PD15 (I/O) |
| D14 (I/O) | D14 (I/O) | PD14 (I/O)/D14 (I/O) | PD14 (I/O) |
| D13 (I/O) | D13 (I/O) | PD13 (I/O)/D13 (I/O) | PD13 (I/O) |
| D12 (I/O) | D12 (I/O) | PD12 (I/O)/D12 (I/O) | PD12 (I/O) |
| D11 (I/O) | D11 (I/O) | PD11 (I/O)/D11 (I/O) | PD11 (I/O) |
| D10 (I/O) | D10 (I/O) | PD10 (I/O)/D10 (I/O) | PD10 (I/O) |
| D9 (I/O) | D9 (I/O) | PD9 (I/O)/D9 (I/O) | PD9 (I/O) |
| D8 (I/O) | D8 (I/O) | PD8 (I/O)/D8 (I/O) | PD8 (I/O) |
| D7 (I/O) | D7 (I/O) | PD7 (I/O)/D7 (I/O) | PD7 (I/O) |
| D6 (I/O) | D6 (I/O) | PD6 (I/O)/D6 (I/O) | PD6 (I/O) |
| D5 (I/O) | D5 (I/O) | PD5 (I/O)/D5 (I/O) | PD5 (I/O) |
| D4 (I/O) | D4 (I/O) | PD4 (I/O)/D4 (I/O) | PD4 (I/O) |
| D3 (I/O) | D3 (I/O) | PD3 (I/O)/D3 (I/O) | PD3 (I/O) |
| D2 (I/O) | D2 (I/O) | PD2 (I/O)/D2 (I/O) | PD2 (I/O) |
| D1 (I/O) | D1 (I/O) | PD1 (I/O)/D1 (I/O) | PD1 (I/O) |
| D0 (I/O) | D0 (I/O) | PD0 (I/O)/D0 (I/O) | PD0 (I/O) |

17.5.1 Register Configuration

Table 17.11 summarizes the port D register.

Table 17.11 Port D Register

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|------------------------|--------------|-----|---------------|--------------------------|-------------|
| Port D data register L | PDDL | R/W | H'0000 | H'FFFF83A2 H'FFFF83A3 | 8, 16, 32 |

17.5.2 Port D Data Register L (PDDRL)

PDDRL is a 16-bit read/write register that stores data for port D. The bits PD15DR–PD0DR correspond to the PD15/D15–PD0/D0 pins. When the pins are used as ordinary outputs, they will output whatever value is written in the PDDRL; when PDDRL is read, the register value will be read regardless of the pin status. When the pins are used as ordinary inputs, the pin status rather than the register value is read directly when PDDRL is read. When a value is written to PDDRL, that value can be written into PDDRL, but it will not affect the pin status. Table 17.12 shows the read/write operations of the port D data register.

PDDRL is initialized by an external power-on reset. However, PDDRL is not initialized for a manual reset, reset by WDT, standby mode, or sleep mode.

The settings in this register are functional only in the SH7016 and SH7017. In the SH7014, there are no pins corresponding to this register, and it should not be read or written to.

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PD15DR | PD14DR | PD13DR | PD12DR | PD11DR | PD10DR | PD9DR | PD8DR |
| 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 |
| | PD7DR | PD6DR | PD5DR | PD4DR | PD3DR | PD2DR | PD1DR | PD0DR |
| 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 |

Table 17.12 Read/Write Operation of the Port D Data Register (PDDR)

| PDIOR | Pin Status | Read | Write |
|-------|-----------------|------------|---|
| 0 | Ordinary input | Pin status | Can write to PDDR, but it has no effect on pin status |
| | Other function | Pin status | Can write to PDDR, but it has no effect on pin status |
| 1 | Ordinary output | PDDR value | Value written is output by pin |
| | Other function | PDDR value | Can write to PDDR, but it has no effect on pin status |

17.6 Port E

Port E is a 16-pin input/output port, as listed in table 17.13.

Table 17.13 Port E

| Extended Modes (Modes 0, 1, 2)* ¹ | Single Chip Mode* ² |
|---|--------------------------------|
| PE15 (I/O)/ DACK1 (output) | PE15 (I/O) |
| PE14 (I/O)/DACK0 (output)/ \overline{AH} (output) | PE14 (I/O) |
| PE13 (I/O) | PE13 (I/O) |
| PE12 (I/O) | PE12 (I/O) |
| PE11 (I/O) | PE11 (I/O) |
| PE10 (I/O) | PE10 (I/O) |
| PE9 (I/O) | PE9 (I/O) |
| PE8 (I/O) | PE8 (I/O) |
| PE7 (I/O)/TIOC2B (I/O) | PE7 (I/O)/TIOC2B (I/O) |
| PE6 (I/O)/TIOC2A (I/O) | PE6 (I/O)/TIOC2A (I/O) |
| PE5 (I/O)/TIOC1B (I/O) | PE5 (I/O)/TIOC1B (I/O) |
| PE4 (I/O)/TIOC1A (I/O) | PE4 (I/O)/TIOC1A (I/O) |
| PE3 (I/O)/TIOC0D (I/O)/DRAK1 (output) | PE3 (I/O)/TIOC0D (I/O) |
| PE2 (I/O)/TIOC0C (I/O)/ $\overline{DREQ1}$ (input) | PE2 (I/O)/TIOC0C (I/O) |
| PE1 (I/O)/TIOC0B (I/O)/DRAK0 (output) | PE1 (I/O)/TIOC0B (I/O) |
| PE0 (I/O)/TIOC0A (I/O)/ $\overline{DREQ0}$ (input) | PE0 (I/O)/TIOC0A (I/O) |

Notes: 1. Modes 0 and 1 only in the SH7014.
2. SH7016 and SH7017 only.

17.6.1 Register Configuration

Table 17.14 summarizes the port E register.

Table 17.14 Port E Register

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|----------------------|--------------|-----|---------------|--------------------------|-------------|
| Port E data register | PEDR | R/W | H'0000 | H'FFFF83B0 H'FFFF83B1 | 8, 16, 32 |

17.6.2 Port E Data Register (PEDR)

PEDR is a 16-bit read/write register that stores data for port E. The bits PE15DR–PE0DR correspond to the PE15/DACK1–PE0/TIOC0A/ $\overline{\text{DREQ0}}$ pins. When the pins are used as ordinary outputs, they will output whatever value is written in the PEDR; when PEDR is read, the register value will be read regardless of the pin status. When the pins are used as ordinary inputs, the pin status rather than the register value is read directly when PEDR is read. When a value is written to PEDR, that value can be written into PEDR, but it will not affect the pin status. Table 17.15 shows the read/write operations of the port E data register.

PEDR is initialized by a external power-on reset. However, PEDR is not initialized for a manual reset, reset by WDT, standby mode, or sleep mode, so the previous data is retained.

| | | | | | | | | |
|----------------|--------|--------|--------|--------|--------|--------|-------|-------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | PE15DR | PE14DR | PE13DR | PE12DR | PE11DR | PE10DR | PE9DR | PE8DR |
| 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 |
| | PE7DR | PE6DR | PE5DR | PE4DR | PE3DR | PE2DR | PE1DR | PE0DR |
| 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 |

Table 17.15 Read/Write Operation of the Port E Data Register (PEDR)

| PEIOR | Pin Status | Read | Write |
|-------|-----------------|------------|---|
| 0 | Ordinary input | Pin status | Can write to PEDR, but it has no effect on pin status |
| | Other function | Pin status | Can write to PEDR, but it has no effect on pin status |
| 1 | Ordinary output | PEDR value | Value written is output by pin |
| | Other function | PEDR value | Can write to PEDR, but it has no effect on pin status |

17.7 Port F

Port F is an 8-pin input port. All modes are configured in the following way:

- PF7 (input)/AN7 (input)
- PF6 (input)/AN6 (input)
- PF5 (input)/AN5 (input)
- PF4 (input)/AN4 (input)
- PF3 (input)/AN3 (input)
- PF2 (input)/AN2 (input)
- PF1 (input)/AN1 (input)
- PF0 (input)/AN0 (input)

17.7.1 Register Configuration

Table 17.16 summarizes the port F register.

Table 17.16 Port F Register

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|----------------------|--------------|-----|------------------------|------------|-------------|
| Port F data register | PFDR | R | External pin dependent | H'FFFF83B3 | 8 |

17.7.2 Port F Data Register (PFDR)

PFDR is an 8-bit read-only register that stores data for port F. The bits PF7DR–PF0DR correspond to the PF7/AN7–PF0/AN0 pins. There are no bits 15 to 8, so always access as eight bits. Any value written into these bits is ignored, and there is no effect on the status of the pins. When any of the bits are read, the pin status rather than the bit value is read directly. However, when an A/D converter analog input is being sampled, values of 1 are read out. Table 17.17 shows the read/write operations of the port F data register.

PFDR is not initialized by power-on resets, manual resets, standby mode, or sleep mode (the bits always reflect the pin status).

| | | | | | | | | |
|----------------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | PF7DR | PF6DR | PF5DR | PF4DR | PF3DR | PF2DR | PF1DR | PF0DR |
| Initial value: | * | * | * | * | * | * | * | * |
| R/W: | R | R | R | R | R | R | R | R |

Note: * Initial values are dependent on the status of the pins at the time of the reads.

Table 17.17 Read/Write Operation of the Port F Data Register (PFDR)

| Pin I/O | Pin Function | Read | Write |
|----------------|---------------------|--------------------|-----------------------------------|
| Input | Ordinary | Pin status is read | Ignored (no effect on pin status) |
| | ANn: analog input | 1 is read | Ignored (no effect on pin status) |

n=7-0

Section 18 128 kB Flash Memory (F-ZTAT)

18.1 Features

The SH7017 has 128 kbytes of on-chip flash memory. 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. Block erase (in single-block units) can be performed. Erasing the entire memory requires erasure of each block in turn. Block erasing can be performed as required on 1 kB, 28 kB, and 32 kB 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

With data transfer in boot mode, the SH7017's bit rate can be automatically adjusted to match the transfer bit rate of the host.
- Flash memory emulation in RAM

flash memory programming can be emulated in real time by overlapping a part of RAM onto flash memory.
- Protect modes

There are two protect modes, hardware and software, 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.

18.2 Overview

18.2.1 Block Diagram

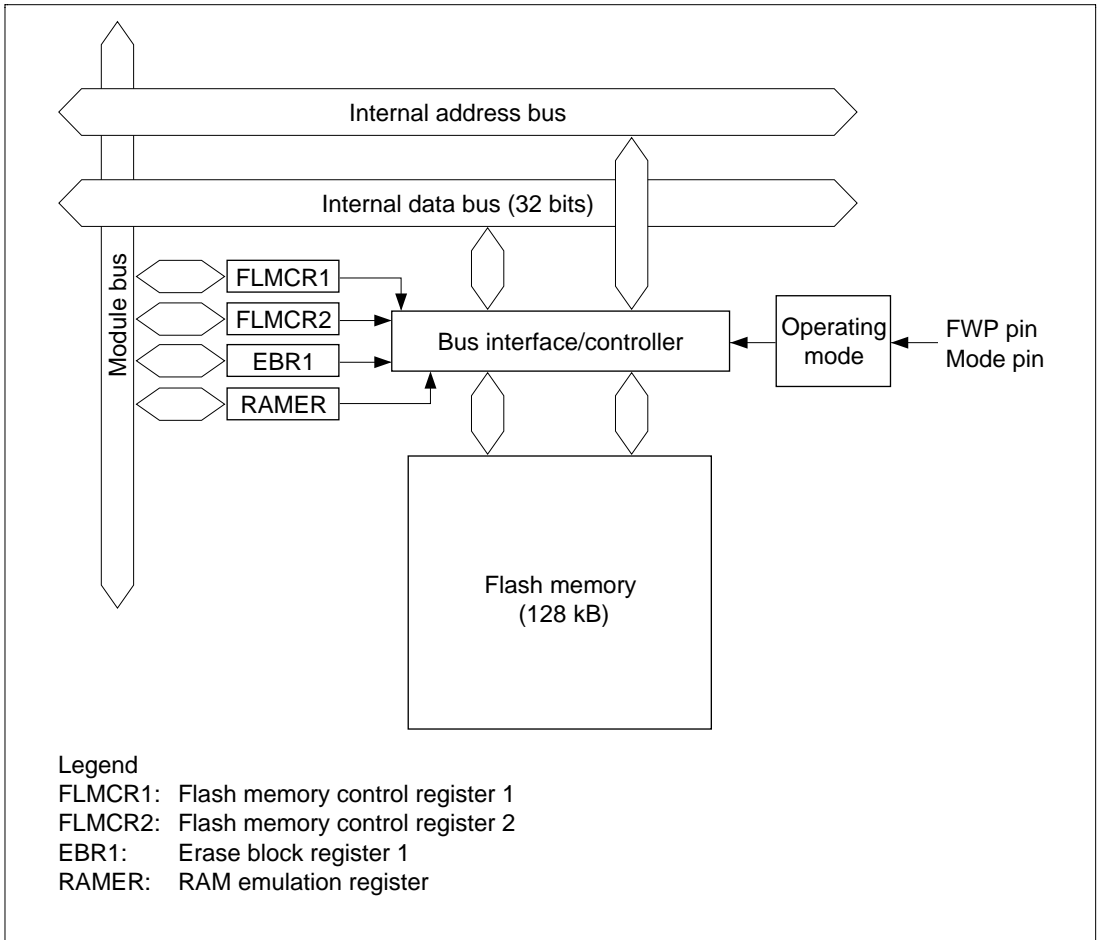


Figure 18.1 Block Diagram of Flash Memory

18.2.2 Mode Transitions

When the mode pins and the FWP pin are set in the reset state and a reset-start is executed, the SH7017 enters one of the operating modes shown in figure 18.2. 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.

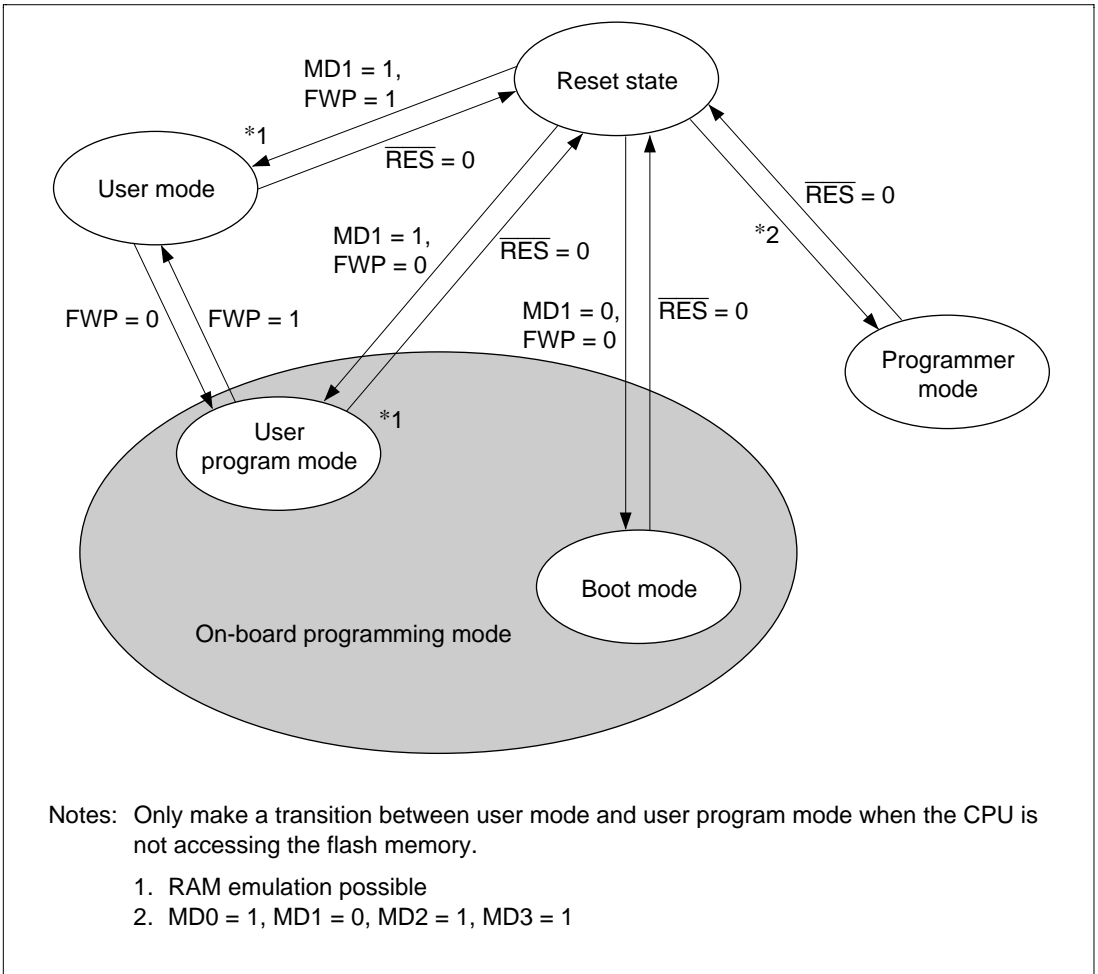


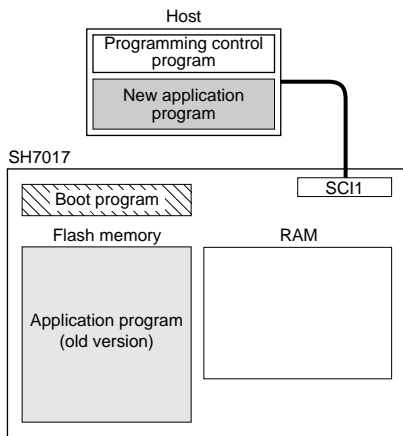
Figure 18.2 Flash Memory Mode Transitions

18.2.3 On-Board Programming Modes

Boot Mode

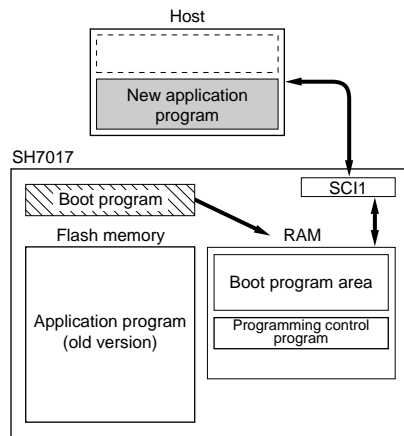
1. Initial state

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.



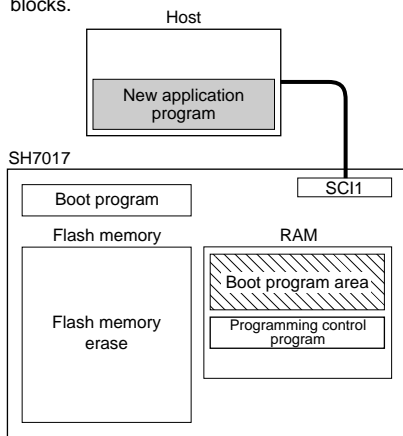
2. Programming control program transfer

When boot mode is entered, the boot program in the SH7050 (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.



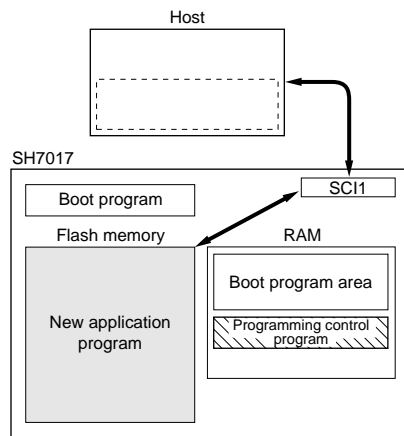
3. Flash memory initialization

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.




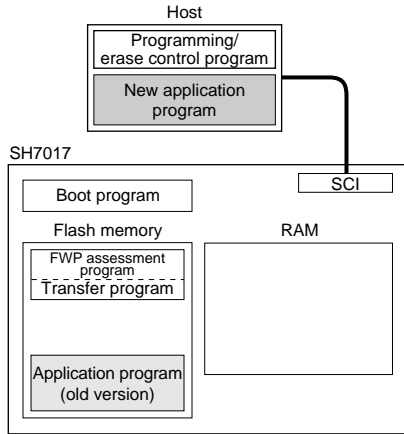
 Program execution state

Figure 18.3 Boot Mode

User Program Mode

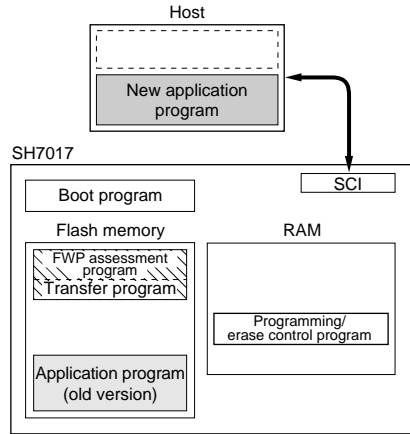
1. Initial state

The FWP assessment program that confirms that user program mode has been entered, and the program that will transfer the programming/erase control program from flash memory to on-chip RAM should be written into the flash memory by the user beforehand. The programming/erase control program should be prepared in the host or in the flash memory.



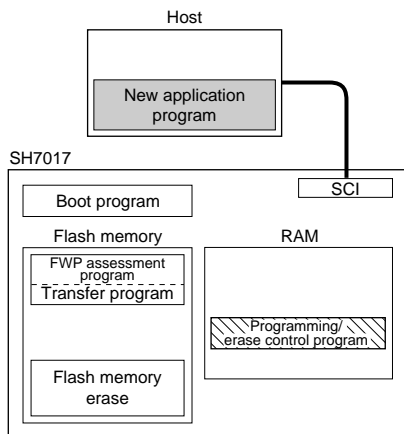
2. Programming/erase control program transfer

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



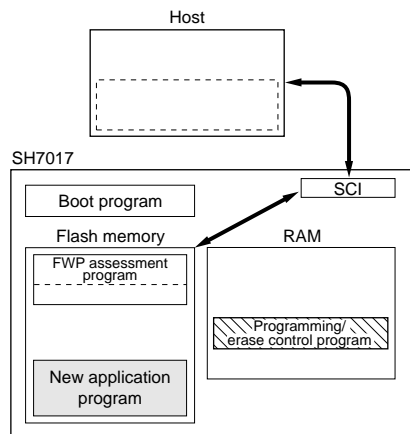
3. Flash memory initialization

The programming/erase 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 18.4 User Program Mode

18.2.4 Flash Memory Emulation in RAM

Emulation should be performed in user mode or user program mode. When the emulation block set in RAMER is accessed while the emulation function is being executed, data written in the overlap RAM is read.

- User Mode
- User Program Mode

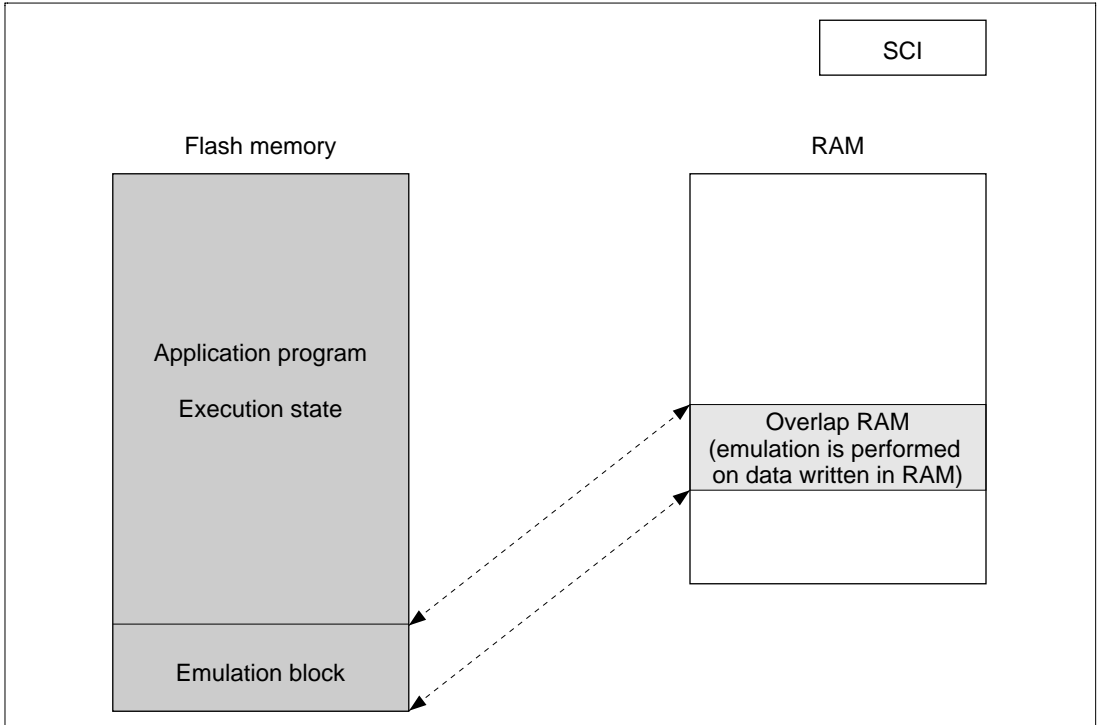


Figure 18.5 Emulation

When overlap RAM data is confirmed, the RAMS bit is cleared, RAM overlap is released, and writes should actually be performed to the flash memory.

When the programming control program is transferred to RAM, ensure that the transfer destination and the overlap RAM do not overlap, as this will cause data in the overlap RAM to be rewritten.

- User Program Mode

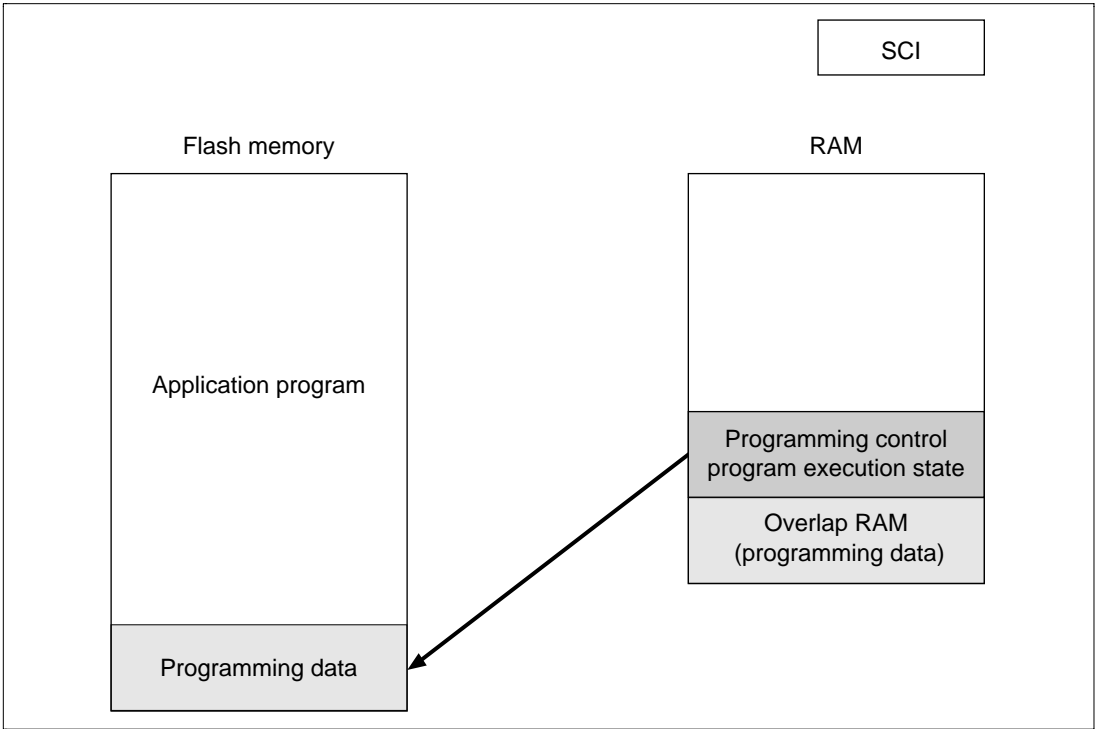


Figure 18.6 Programming to the Flash Memory

18.2.5 Differences between Boot Mode and User Program Mode

Table 18.1 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* | (2) | (1) (2) (3) |

(1) Erase/erase-verify

(2) Program/program-verify

(3) Emulation

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

18.2.6 Block Configuration

The flash memory is divided into three 32 kB blocks, one 28 kB blocks, and four 1 kB blocks.

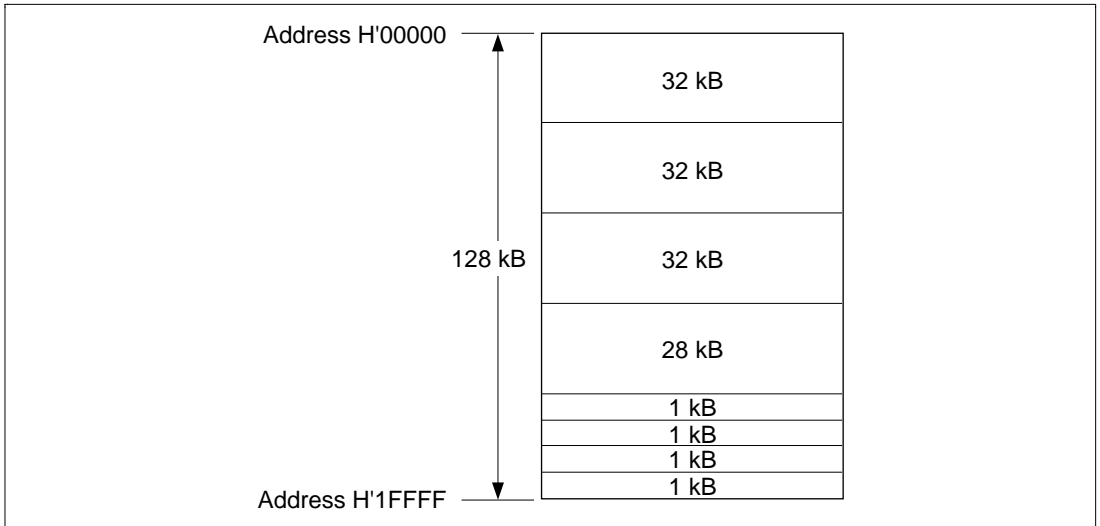


Figure 18.7 Block Configuration

18.3 Pin Configuration

The flash memory is controlled by means of the pins shown in table 18.2.

Table 18.2 Flash Memory Pins

| Pin Name | Abbreviation | I/O | Function |
|----------------------|--------------|--------|--|
| Power-on reset | RES | Input | Power-on reset |
| Flash memory protect | FWP | Input | Flash program/erase protection by hardware |
| Mode 3 | MD3 | Input | Sets SH7017 operating mode |
| Mode 2 | MD2 | Input | Sets SH7017 operating mode |
| Mode 1 | MD1 | Input | Sets SH7017 operating mode |
| Mode 0 | MD0 | Input | Sets SH7017 operating mode |
| Transmit data | TxD1 | Output | Serial transmit data output |
| Receive data | RxD1 | Input | Serial receive data input |

18.4 Register Configuration

The registers used to control the on-chip flash memory when enabled are shown in table 18.3.

Table 18.3 Flash Memory Registers

| Register Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|---------------------------------|--------------|-------------------|--------------------|------------|-------------|
| Flash memory control register 1 | FLMCR1 | R/W* ¹ | H'00* ³ | H'FFFF8580 | 8 |
| Flash memory control register 2 | FLMCR2 | R* ² | H'00 | H'FFFF8581 | 8 |
| Erase block register 1 | EBR1 | R/W* ¹ | H'00* ⁴ | H'FFFF8582 | 8 |
| RAM emulation register | RAMER | R/W | H'0000 | H'FFFF8628 | 8, 16, 32 |

- Notes:
1. In modes in which the on-chip flash memory is disabled, a read will return H'00, and writes are invalid. Writes are also disabled when the FWE bit is set to 1 in FLMCR1.
 2. A read in a mode in which on-chip flash memory is disabled will return H'00.
 3. When a high level is input to the FWP pin, the initial value is H'80.
 4. When a low level is input to the FWP pin, or if a high level is input and the SWE bit in FLMCR1 is not set, these registers are initialized to H'00.
 5. FLMCR1, FLMCR2, and EBR1 are 8-bit registers, and RAMER is a 16-bit register.
 6. Only byte accesses are valid for FLMCR1, FLMCR2, and EBR1, the access requiring 3 cycles. Three cycles are required for a byte or word access to RAMER, and 6 cycles for a longword access.
 7. When a longword write is performed on RAMER, 0 must always be written to the lower word (address H'FFFF8630). Operation is not guaranteed if any other value is written.

18.5 Register Descriptions

18.5.1 Flash Memory Control Register 1 (FLMCR1)

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, and finally setting the P bit. Erase mode is entered by setting SWE to 1 when FWE = 1, then setting the ESU bit, and finally setting the E bit. FLMCR1 is initialized by a reset, and in hardware standby mode and software standby mode. 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 bits SWE, ESU, PSU, EV, and PV are enabled 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 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | FWE | SWE | ESU | PSU | EV | PV | E | P |
| Initial value: | 1/0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

- Bit 7—Flash Write Enable Bit (FWE): Sets hardware protection against flash memory programming/erasing.

Bit 7:

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

- Bit 6—Software Write Enable Bit (SWE): Enables or disables the flash memory. This bit should be set before setting bits 5 to 0, and EBR1 bits 7 to 0.

Bit 6:

| SWE | Description |
|-----|---|
| 0 | Writes disabled (Initial value) |
| 1 | Writes enabled [Setting condition] When FWE = 1 |

- Bit 5—Erase Setup Bit (ESU): Prepares for a transition to erase mode. Do not set the SWE, PSU, EV, PV, E, or P bit at the same time.

Bit 5:

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

- Bit 4—Program Setup Bit (PSU): Prepares for a transition to program mode. Do not set the SWE, ESU, EV, PV, E, or P bit at the same time.

Bit 4:

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

- 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] When FWE = 1 and SWE = 1 |

- 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] When FWE = 1 and SWE = 1 |

- 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] When FWE = 1, SWE = 1, and ESU1 = 1 |

- 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] When FWE = 1, SWE = 1, and PSU1 = 1 |

18.5.2 Flash Memory Control Register 2 (FLMCR2)

FLMCR2 is an 8-bit register that monitors the presence or absence of flash memory program/erase protection (error protection). FLMCR2 is initialized to H'00 by a reset, and in hardware standby mode.

When on-chip flash memory is disabled, a read will return H'00.

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

- 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. (Initial value) Flash memory program/erase protection (error protection) is disabled. [Clearing condition] Power-on reset |
| 1 | An error has occurred during flash memory programming/erasing. Flash memory program/erase protection (error protection) is enabled. [Setting condition] See section 18.8.3, Error Protection. |

- Bits 6 to 0—Reserved: These bits are always read as 0.

18.5.3 Erase Block Register 1 (EBR1)

EBR1 is an 8-bit readable/writable register that specifies the flash memory erase area block by block. EBR1 is initialized to H'00 by a reset, in hardware standby mode and software standby mode, when a low level is input to the FWE pin, and 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 to 1, the corresponding block can be erased. Other blocks are erase-protected. Set only one bit in EBR1 (more than one bit cannot be set). When on-chip flash memory is disabled, a read will return H'00, and writes are invalid.

The flash memory block configuration is shown in table 18.4.

| | | | | | | | | |
|----------------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | 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 |

Table 18.4 Flash Memory Erase Blocks

| Block (Size) | Address |
|--------------|-------------------|
| EB0 (32 kB) | H'000000–H'007FFF |
| EB1 (32 kB) | H'008000–H'00FFFF |
| EB2 (32 kB) | H'010000–H'017FFF |
| EB3 (28 kB) | H'018000–H'01EFFF |
| EB4 (1 kB) | H'01F000–H'01F3FF |
| EB5 (1 kB) | H'01F400–H'01F7FF |
| EB6 (1 kB) | H'01F800–H'01FBFF |
| EB7 (1 kB) | H'01FC00–H'01FFFF |

18.5.4 RAM Emulation Register (RAMER)

RAMER specifies the area of flash memory to be overlapped with part of RAM when emulating real-time flash memory programming. RAMER is initialized to H'0000 by a reset and in hardware standby mode. It is not initialized in software standby mode. RAMER settings should be made in user mode or user program mode. (For details, see the description of the BSC.)

Flash memory area divisions are shown in table 18.5. To ensure correct operation of the emulation function, the ROM for which RAM emulation is performed should not be accessed immediately after this register has been modified. Normal execution of an access immediately after register modification is not guaranteed.

| | | | | | | | | |
|----------------|----|----|----|----|----|------|------|------|
| Bit: | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| | — | — | — | — | — | — | — | — |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R | R | R |
| Bit: | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | — | — | — | — | — | RAMS | RAM1 | RAM0 |
| Initial value: | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| R/W: | R | R | R | R | R | R/W | R/W | R/W |

- Bits 15 to 3—Reserved: These bits are always read as 0.
- Bit 2—RAM Select (RAMS): Specifies selection or non-selection of flash memory emulation in RAM. When RAMS = 1, all flash memory block are program/erase-protected.

Bit 2:

| RAMS | Description |
|------|---|
| 0 | Emulation not selected Program/erase-protection of all flash memory blocks is disabled |
| 1 | Emulation selected Program/erase-protection of all flash memory blocks is enabled |

- Bits 1 and 0—Flash Memory Area Selection (RAM1, RAM0): These bits are used together with bit 2 to select the flash memory area to be overlapped with RAM. (See table 18.5.)

Table 18.5 Flash Memory Area Divisions

| Addresses | Block Name | RAMS | RAM1 | RAM0 |
|-------------------|---------------|------|------|------|
| H'FFF800–H'FFFBF | RAM area 1 kB | 0 | * | * |
| H'01F000–H'01F3FF | EB4 (1 kB) | 1 | 0 | 0 |
| H'01F400–H'01F7FF | EB5 (1 kB) | 1 | 0 | 1 |
| H'01F800–H'01FBFF | EB6 (1 kB) | 1 | 1 | 0 |
| H'01FC00–H'01FFFF | EB7 (1 kB) | 1 | 1 | 1 |

18.6 On-Board Programming Modes

When pins are set to on-board programming mode and a reset-start is executed, a transition is made to the on-board programming state in which 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 18.6. For a diagram of the transitions to the various flash memory modes, see figure 18.2.

Table 18.6 Setting On-Board Programming Modes

| Mode | | PLL Multiple | FWP | MD3 | MD2 | MD1 | MD0 |
|-------------------|------------------|--------------|-----|-----|-----|-----|-----|
| Boot mode | Expanded mode | ×1 | 0 | 0 | 0 | 0 | 0 |
| | Single chip mode | | | 0 | 0 | 0 | 1 |
| | Expanded mode | ×2 | | 0 | 1 | 0 | 0 |
| | Single chip mode | | | 0 | 1 | 0 | 1 |
| | Expanded mode | ×4 | | 1 | 0 | 0 | 0 |
| | Single chip mode | | | 1 | 0 | 0 | 1 |
| User program mode | Expanded mode | ×1 | 0 | 0 | 0 | 1 | 0 |
| | Single chip mode | | | 0 | 0 | 1 | 1 |
| | Expanded mode | ×2 | | 0 | 1 | 1 | 0 |
| | Single chip mode | | | 0 | 1 | 1 | 1 |
| | Expanded mode | ×4 | | 1 | 0 | 1 | 0 |
| | Single chip mode | | | 1 | 0 | 1 | 1 |

18.6.1 Boot Mode

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

When a reset-start is executed after the SH7017 pins have been set to boot mode, the boot program built into the SH7017 is started and the programming control program prepared in the host is serially transmitted to the SH7017 via the SCI. In the SH7017, the programming control program received via the SCI 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 18.8, and the boot mode execution procedure in figure 18.9.

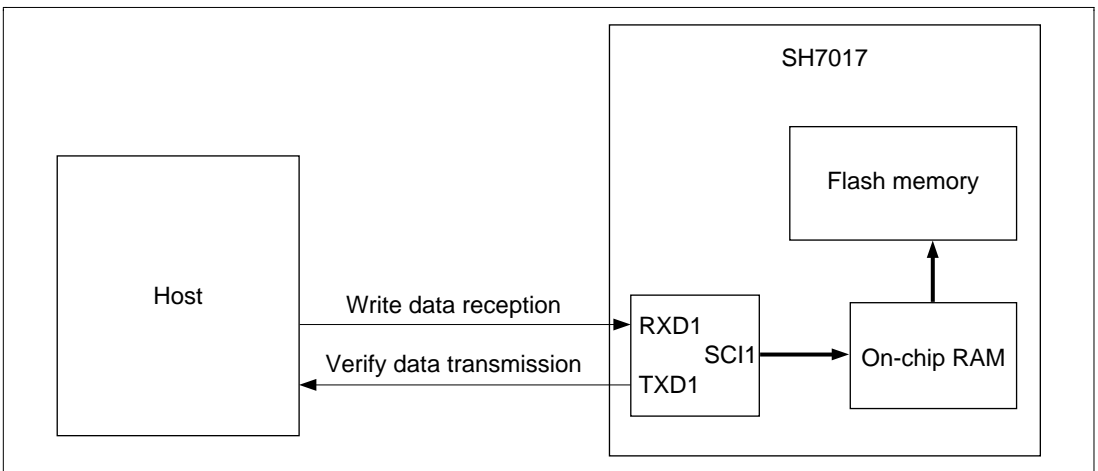
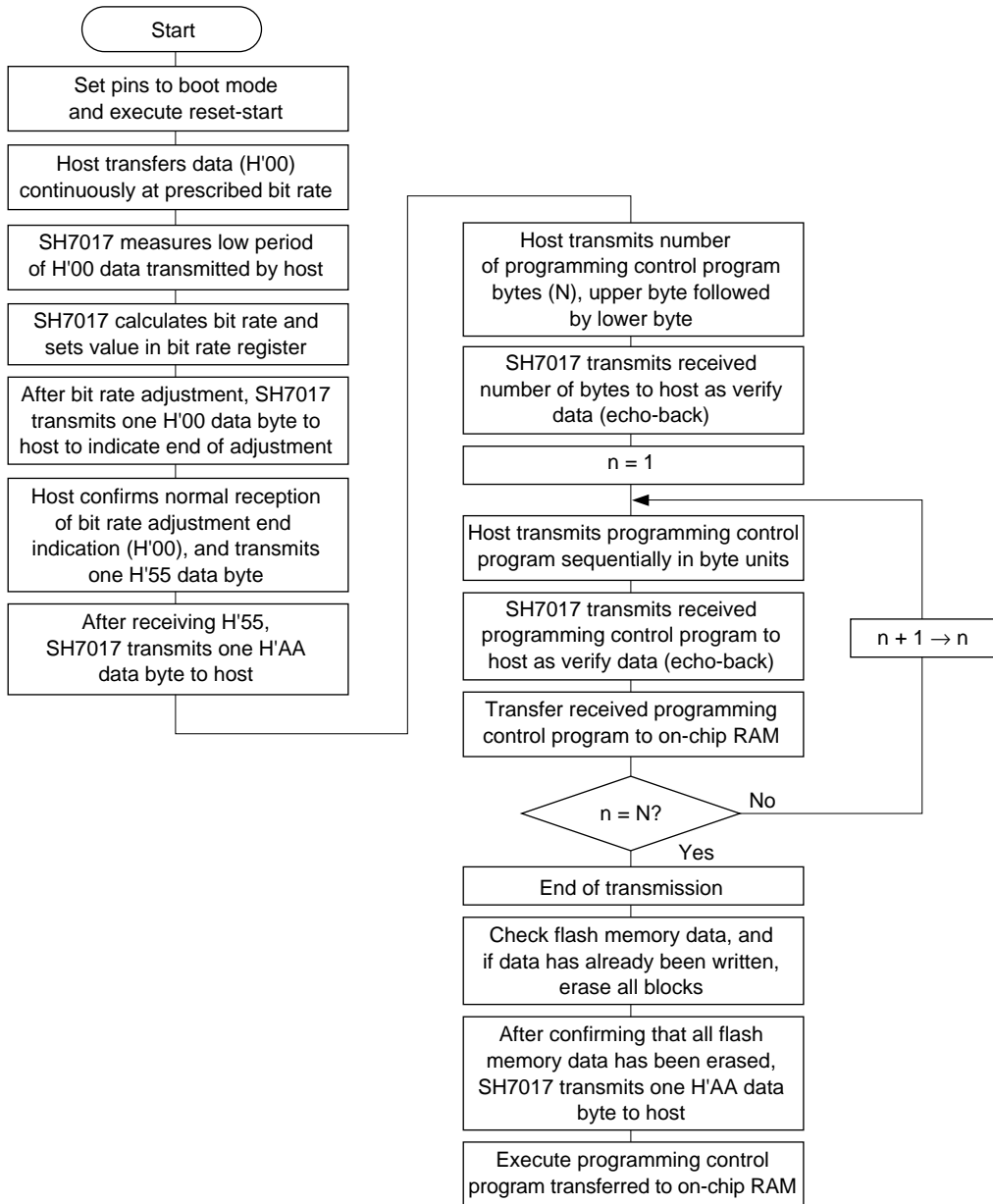


Figure 18.8 System Configuration in Boot Mode



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 18.9 Boot Mode Execution Procedure

Automatic SCI Bit Rate Adjustment

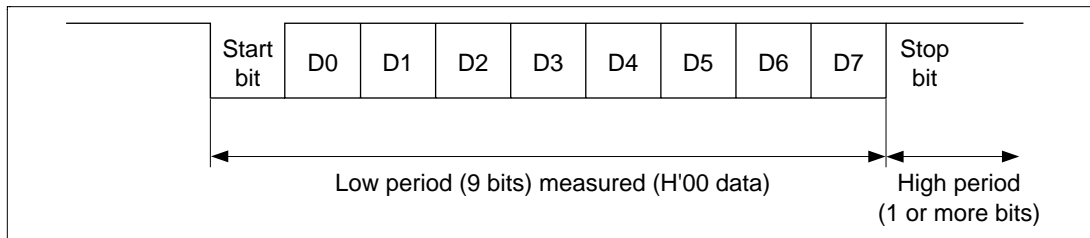


Figure 18.10 Automatic SCI Bit Rate Adjustment

When boot mode is initiated, the SH7017 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 SH7017 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 SH7017. 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 SH7017's system clock frequency, there will be a discrepancy between the bit rates of the host and the SH7017. To ensure correct SCI operation, the host's transfer bit rate should be set to 4800bps, 9600bps.

Table 18.7 shows host transfer bit rates and system clock frequencies for which automatic adjustment of the SH7017 bit rate is possible. The boot program should be executed within this system clock range.

Table 18.7 System Clock Frequencies for which Automatic Adjustment of SH7017 Bit Rate is Possible

| Host Bit Rate | System Clock Frequency for which Automatic Adjustment of SH7017 Bit Rate is Possible |
|---------------|--|
| 9600bps | 8 to 28.7MHz |
| 4800bps | 4 to 20MHz |

On-Chip RAM Area Divisions in Boot Mode: In boot mode, the RAM area is divided into an area used by the boot program and an area to which the programming control program is transferred via the SCI, as shown in figure 18.11. The boot program area cannot be used until the execution state in boot mode switches to the programming control program transferred from the host.

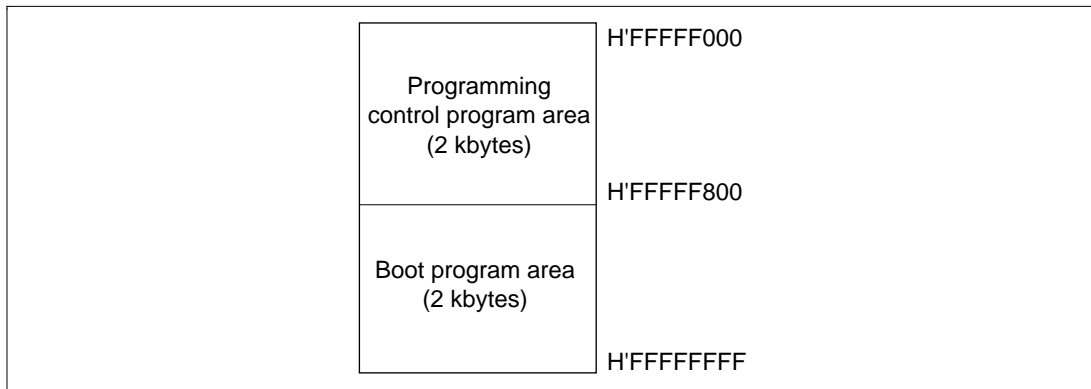


Figure 18.11 RAM Areas in Boot Mode

Note: The boot program area cannot be used until a transition is made to the execution state for the programming control program transferred to RAM. Note also that the boot program remains in this area of the on-chip RAM even after control branches to the programming control program.

18.6.2 User Program Mode

After setting FWP, the user should branch to, and execute, the previously prepared programming/erase control program.

As the flash memory itself cannot be read while flash memory programming/erasing is being executed, the control program that performs programming and erasing should be run in on-chip RAM or external memory.

Use the following procedure (figure 18.12) to execute the programming control program that writes to flash memory (when transferred to RAM).

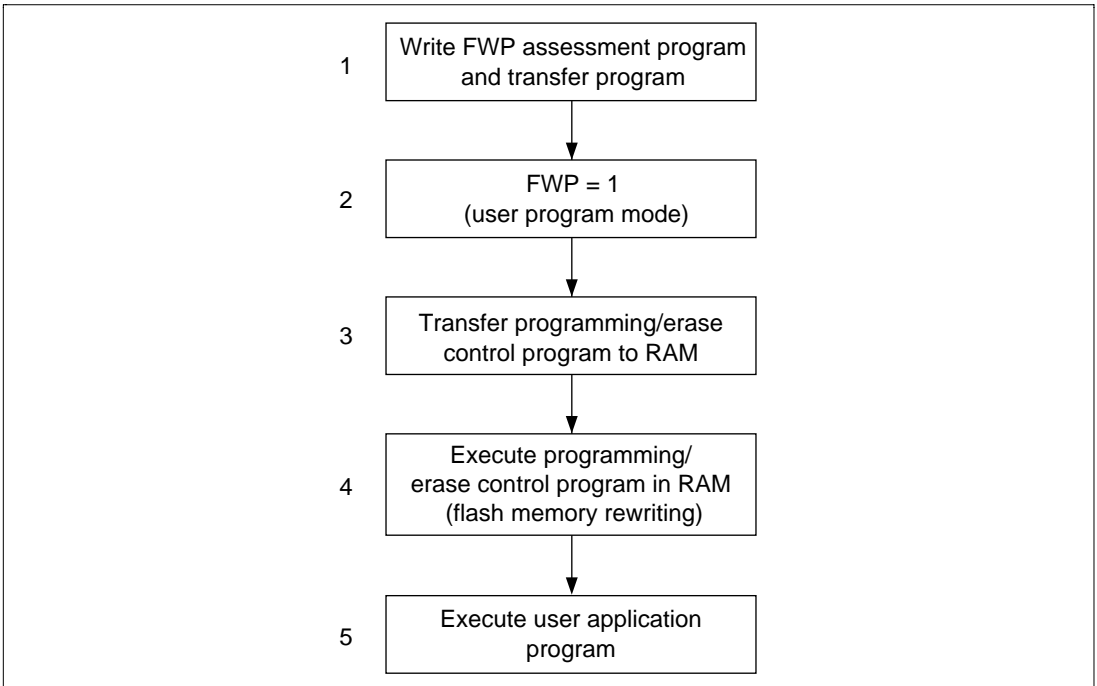


Figure 18.12 User Program Mode Execution Procedure

Note: When programming and erasing, start the watchdog timer so that measures can be taken to prevent program runaway, etc. Memory cells may not operate normally if overprogrammed or overerased due to program runaway.

18.7 Programming/Erasing Flash Memory

A software method, using the CPU, is employed to program and erase flash memory in the on-board programming modes. 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, ESU, P, E, PV, and EV bits in FLMCR1.

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

- Notes:
1. Operation is not guaranteed if setting/resetting of the SWE, ESU, PSU, EV, PV, E, and P bits in FLMCR1 is executed by a program in flash memory.
 2. When programming or erasing, a low level is input to the FWP pin (programming/erasing will not be executed if a high level is input to the FWP pin).
 3. Programming should be performed in the erased state. Do not perform additional programming on previously programmed addresses.

18.7.1 Program Mode

Follow the procedure shown in the program/program-verify flowchart in figure 18.7 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.

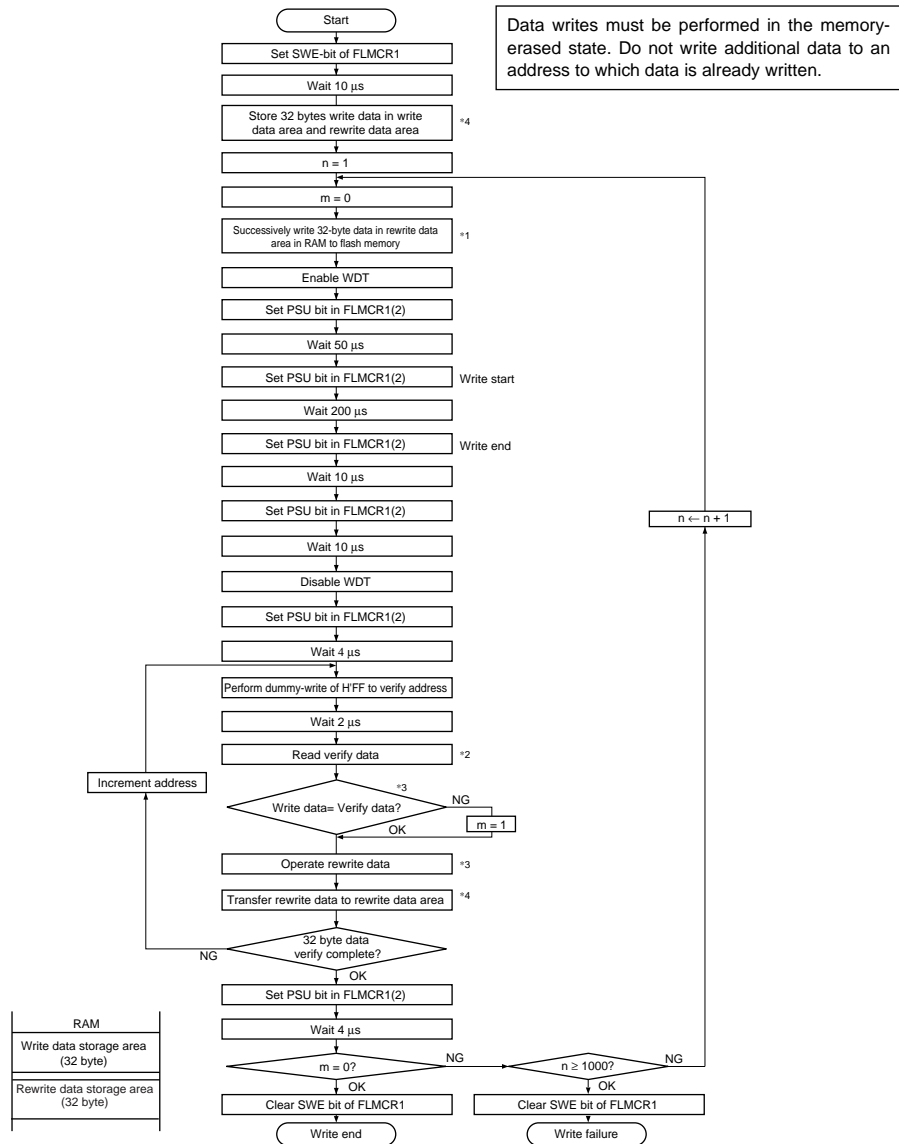
Following the elapse of 10 μ s or more after the SWE bit is set to 1 in flash memory control register 1 (FLMCR1), 32-byte program data is stored in the program data area and reprogram data area, and the 32-byte data in the program data area in RAM is written consecutively to the program address (the 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 a minimum value of 300 μ s or more as the WDT overflow period. After this, preparation for program mode (program setup) is carried out by setting the PSU bit in FLMCR1, and after the elapse of 50 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. Use a fixed 200 μ s pulse for the write time.

18.7.2 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 PSUn bit is cleared at least 10 μ s later). The watchdog timer is cleared after the elapse of 10 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 4 μ s or more. When the flash memory is read in this state (verify data is read in 32-bit units), the data at the latched address is read. Wait at least 2 μ s after the dummy write before performing this read operation. Next, the written data is compared with the verify data, and reprogram data is computed (see figure 18.13) and transferred to the reprogram data area. After 32 bytes of data have been verified, exit program-verify mode, wait for at least 4 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 1000 times on the same bits.



- Notes: 1. Transfer data in a byte unit. The lower eight bits of the start address to which data is written must be H'00, H'20, H'40, H'60, H'80, H'A0, H'C0, or H'E0. Transfer 32-byte data even when writing fewer than 32 bytes. In this case, Set H'FF in unused addresses.
 2. Read verify data in logword form (32 bits).
 3. Already programmed bits are not reprogrammed. Reprogram data is determined by the computation shown below.
 4. The write data storage area (32 bytes) and rewrite data storage area (32 bytes) must be located in RAM. The contents of the rewrite data storage area are rewritten as writing progresses.

| Previous write data | Verify data (V) | Rewrite data (X) | Description |
|---------------------|-----------------|------------------|---|
| 0 | 0 | 1 | Rewrite should not be performed to bits already written to. |
| 0 | 1 | 0 | Write is incomplete; rewrite should be performed. |
| 1 | 0 | 1 | — |
| 1 | 1 | 1 | Left in the erased state. |

Figure 18.13 Program/Program-Verify Flowchart

- Sample 32-byte programming program

The wait time set values (number of loops) are for the case where $f = 28.7$ MHz. For other frequencies, the set value is given by the following expression:

$$\text{Wait time } (\mu\text{S}) \times f \text{ (MHz)} \div 4$$

Registers Used

R4 (input): Program data storage address
 R5 (input): Programming destination address
 R7 (output): OK (normal) or NG (error)
 R0-3, 8-13: Work registers

```

FLMCR1      .EQU    H'80
FLMCR2      .EQU    H'81
OK           .EQU    H'0
NG           .EQU    H'1
Wait10u     .EQU    72
Wait50u     .EQU    359
Wait4u      .EQU    29
Wait2u      .EQU    14
Wait200u    .EQU    1435
WDT_TCSR    .EQU    H'FFFF8610
WDT_573u    .EQU    H'A579
SWESET      .EQU    B'01000000
PSU1SET     .EQU    B'00010000
P1SET       .EQU    B'00000001
PICLEAR     .EQU    B'11111110
PSU1CLEAR   .EQU    B'11101111
PVSET       .EQU    B'00000100
PVCLEAR     .EQU    B'11111011
SWECLEAR    .EQU    B'10111111
MAXVerify   .EQU    1000
;
FlashProgram .EQU    $
    MOV     #H'01,R2           ; R2 work register (1)
    MOV.L   #PdataBuff,R0     ; Save program data to work area
    MOV     R4,R12
    MOV     #8,R13
COPY_LOOP   .EQU    $
  
```

```

MOV.L    @R12+,R1
MOV.L    R1,@R0
ADD.L    #4,R0
ADD.L    #-1,R13
CMP/PL   R13
BT       COPY_LOOP
MOV.L    #H'FFFF8500,R0          ; Initialize GBR
LDC      R0,GBR
;
MOV.L    #Wait10u,R3
MOV.L    #FLMCR1,R0              ; Initialize R0 to FLCMR1 address
OR.B     #SWESET,@(R0,GBR)      ; Set SWE
Wait_1   SUBC    R2,R3            ; Wait 10 µs
BF       Wait_1
;
MOV.L    #H' 20000,R9
CMP/GT   R5,R9
BT       Program_Start
MOV.L    #FLMCR2,R0
Program_Start .EQU    $
MOV.L    #0,R9                  ; Initialize n (R9) to 0
;
Program_loop .EQU    $
MOV.L    #0,R10                 ; Initialize m (R10) to 0
MOV.L    #32,R3                 ; Write 32-byte data consecutively
MOV.L    #PdataBuff,R12
MOV.L    R5,R13
Write_Loop .EQU    $
MOV.B    @R12+,R1
MOV.B    R1,@R13
ADD.L    #1,R13
ADD.L    #-1,R3
CMP/PL   R3
BT       Write_Loop
;
MOV.L    #WDT_TCSR,R1           ; Enable WDT
MOV.W    #WDT_573u,R3          ; 573.4 µs cycle

```



```

        MOV.W      R3,@R1
;
        MOV.L      #Wait50u,R3
        OR.B       #PSU1SET,@(R0,GBR)      ; Set PSU
Wait_2  SUBC       R2,R3                    ; Wait 50 µs
        BF        Wait_2
;
        MOV.L      #Wait200u,R3
        OR.B       #P1SET,@(R0,GBR)       ; Set P
Wait_3  SUBC       R2,R3                    ; Wait 200 µs
        BF        Wait_3
;
        MOV.L      #Wait10u,R3
        AND.B      #P1CLEAR,@(R0,GBR)     ; Clear P
Wait_4  SUBC       R2,R3                    ; Wait 10 µs
        BF        Wait_4
;
        MOV.L      #Wait10u,R3
        AND.B      #PSU1CLEAR,@(R0,GBR)   ; Clear PSU
Wait_5  SUBC       R2,R3                    ; Wait 10 µs
        BF        Wait_5
;
        MOV.L      #WDT_TCSR,R1           ; Disable WDT
        MOV.W      #H'A55F,R3
        MOV.W      R3,@R1
;
        MOV.L      #Wait4u,R3
        OR.B       #PVSET,@(R0,GBR)       ; Set PV
Wait_6  SUBC       R2,R3                    ; Wait 4 µs
        BF        Wait_6
;
        MOV.L      PdataBuff,R3
        MOV.L      R4,R1
        MOV.L      R5,R12
        MOV.L      #8,R13
        MOV.L      #H'FFFFFFFF,R11
;

```

```

VerifyLoop      .EQU      $
                MOV.L     R11,@R12          ; Write H'FF to verify address
                MOV.L     R11,@R3          ; Reprogram data RAM (PdataBuff) initialization
                MOV.L     #Wait2u,R7
Wait_7          SUBC     R2,R7              ; Wait 2 μs
                BF        Wait_7
;
                MOV.L     @R12+,R7
                MOV.L     @R1+,R8
                CMP/EQ    R7,R8            ; Verify
                BT        Verify_OK
                MOV.L     #1,R10           ; Verify NG, m <- 1
                XOR       R8,R7            ; Program data computation
                NOT       R7,R7
                OR        R7,R8
                MOV.L     R8,@R3          ; Store in reprogram data RAM (PdataBuff)
Verify_OK       .EQU      $
                ADD.L     #4,R3
                ADD.L     #-1,R13
                CMP/PL    R13
                BT        VerifyLoop
;
                MOV.L     #Wait4u,R7
                AND.B     #PVCLEAR,@(R0,GBR) ; Clear PV
Wait_8          SUBC     R2,R7              ; Wait 4 μs
                BF        Wait_8
;
                CMP/PL    R10 ; if m=0 then GOTO Program_OK
                BF        Program_OK
                ADD      #1,R9
                MOV.L     #NG,R7           ; R7 <- NG (return value)
                MOV.L     #MAXVerify,R12  ; if n>=MAXVerify then Program NG
                CMP/EQ    R9,R12
                BT        Program_end
                BRA       Program_loop
                NOP
Program_OK      .EQU      $

```

```

        MOV.L    #OK,R7                ; R7 <- OK (return value)
Program_end .EQU    $
        MOV.B    #H'00,R0
        MOV.B    R0,@(FLMCR1,GBR)    ; Clear SWE
;
        RTS
        NOP
;
        .ALIGN    4
PdataBuff .RES.B 32

```

18.7.3 Erase Mode

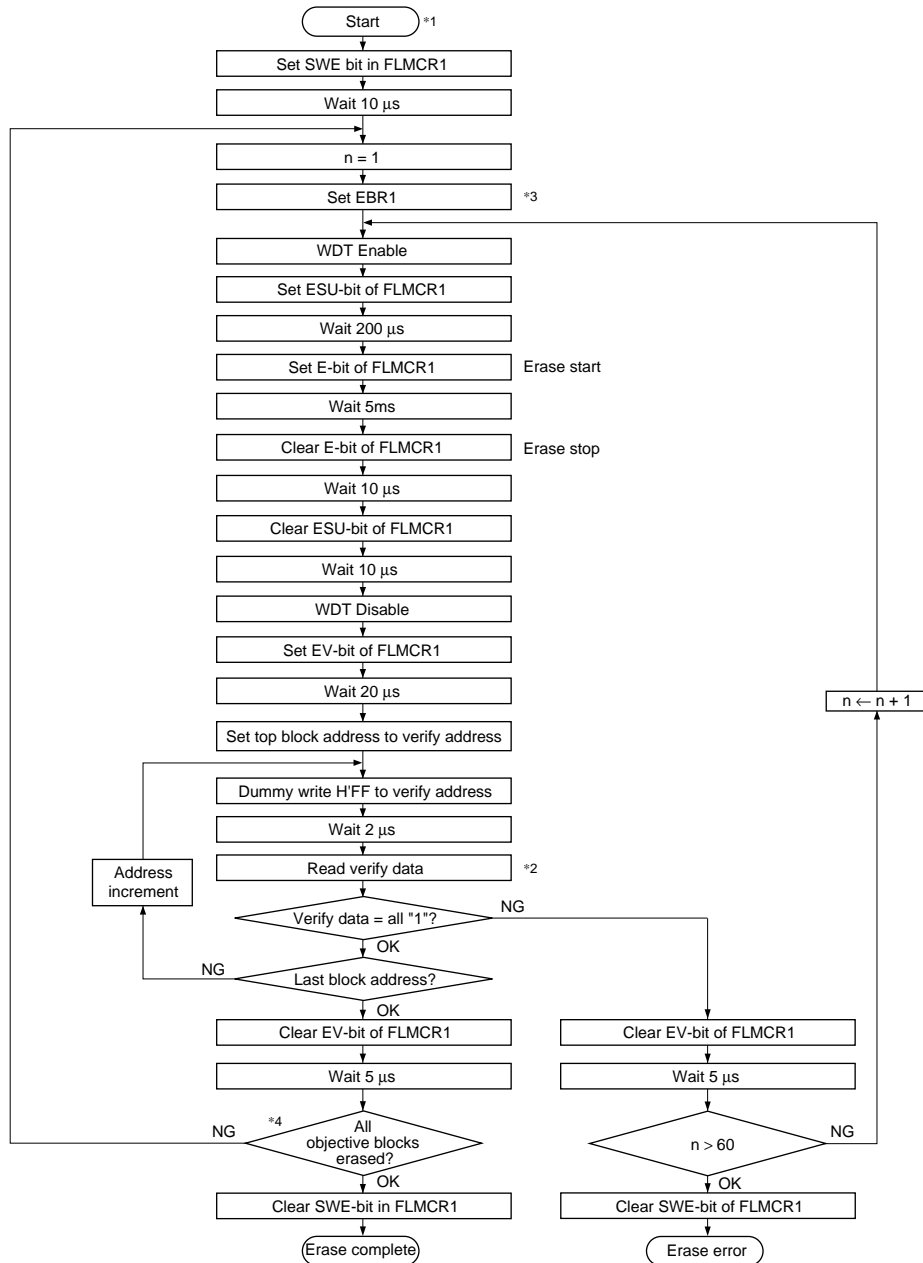
To perform data or program erasure, set the 1 bit flash memory area to be erased in erase block register 1 (EBR1) at least 10 μ s after setting the SWE bit to 1 in flash memory control register 1 (FLMCR1). Next, the watchdog timer is set to prevent overerasing in the event of program runaway, etc. Set 9.2 ms as the WDT overflow period. After this, preparation for erase mode (erase setup) is carried out by setting the ESU bit in FLMCR1, and after the elapse of 200 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 5 ms.

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

18.7.4 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 is cleared at least 10 μ s later), the watchdog timer is cleared after the elapse of 10 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 20 μ s or more. When the flash memory is read in this state (verify data is read in 32-bit units), the data at the latched address is read. Wait at least 2 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 60 times. When verification is completed, exit erase-verify mode, and wait for at least 5 μ s. If erasure has been completed on all the erase blocks, clear the SWE bit in FLMCR1. If there are any unerased blocks, set 1 bit 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. Verify data is read in 32-bit (longword) units.
 3. Set only one bit in EBR1. More than one bit cannot be set.
 4. Erasing is performed in block units. To erase a number of blocks, each block must be erased in turn.

Figure 18.14 Erase/Eraser-Verify Flowchart (Single-Block Erase)

- Sample one-block erase program

The wait time set values (number of loops) are for the case where $f = 28.7$ MHz. For other frequencies, the set value is given by the following expression:

$$\text{Wait time } (\mu\text{S}) \times f \text{ (MHz)} \div 4$$

The WDT overflow cycle set value is for the case where $f = 28.7$ MHz. For other frequencies, ensure that the overflow cycle is a minimum of 5.3 ms.

Registers Used

R5 (input): Memory block table pointer
 R7 (output): OK (normal) or NG (error)
 R0-3, 6, 8-9: Work registers

```

FLMCR1      .EQU    H'80
FLMCR2      .EQU    H'81
EBR1        .EQU    H'82
EBR2        .EQU    H'83
Wait10u     .EQU    72
Wait2u      .EQU    14
Wait200u    .EQU    1435
Wait5m      .EQU    35875
Wait20u     .EQU    144
Wait5u      .EQU    36
WDT_TCSR    .EQU    H'FFFF8610
WDT_9m      .EQU    H'A57D
SWESET      .EQU    B'01000000
ESUSET      .EQU    B'00100000
ESET        .EQU    B'00000010
ECLEAR      .EQU    B'11111101
ESUCLEAR    .EQU    B'11011111
EVSET       .EQU    B'00001000
EVCLEAR     .EQU    B'11110111
SWECLEAR    .EQU    B'10111111
MAXErase    .EQU    60
;
FlashErase  .EQU    $
    MOV.L    #H'FFFF8500,R0
    LDC     R0,GBR                ; Initialize GBR
    MOV.L    #1,R2

```

```

;
MOV.L    #Wait10u,R3
MOV.L    #FLMCR1,R0
OR.B     #SWESET,@(R0,GBR)      ; Set SWE
EWait_1  SUBC    R2,R3           ; Wait 10 µs
BF       EWait_1
;
MOV.L    #0,R9                  ; Initialize n (R9) to 0
;
MOV.B    @(6,R5),R0
MOV.B    R0,@(EBR1,GBR)        ; Erase memory block (EBR1) setting
MOV.B    @(7,R5),R0
MOV.B    R0,@(EBR2,GBR)        ; Erase memory block (EBR2) setting
;
MOV.L    #FLMCR1,R0
MOV.L    @R5,R6                 ; Erase memory block start address -> R6
MOV.L    #H'020000,R7
CMP/GT   R6,R7
BT       EraseLoop
MOV.L    #FLMCR2,R0
;
EraseLoop .EQU    $
MOV.L    #WDT_TCSR,R1          ; Enable WDT
MOV.W    #WDT_9m,R3           ; 9.2 ms cycle
MOV.W    R3,@R1
;
MOV.L    #Wait200u,R3
OR.B     #ESUSET,@(R0,GBR)    ; Set ESU
EWait_2  SUBC    R2,R3           ; Wait 200 µs
BF       EWait_2
;
MOV.L    #Wait5m,R3
OR.B     #ESET,@(R0,GBR)      ; Set E
EWait_3  SUBC    R2,R3           ; Wait 5 ms
BF       EWait_3
;
MOV.L    #Wait10u,R3

```

```

        AND.B    #ECLEAR,@(R0,GBR)      ; Clear E
EWait_4  SUBC    R2,R3                    ; Wait 10 µs
        BF      EWait_4
;
        MOV.L    #Wait10u,R3
        AND.B    #ESUCLEAR,@(R0,GBR)   ; Clear ESU
EWait_5  SUBC    R2,R3                    ; Wait 10 µs
        BF      EWait_5
;
        MOV.L    #WDT_TCSR,R1           ; Disable WDT
        MOV.W    #H'A55F,R3
        MOV.W    R3,@R1
;
        MOV.L    #Wait20u,R3
        OR.B     #EVSET,@(R0,GBR)      ; Set EV
EWait_6  SUBC    R2,R3                    ; Wait 20 µs
        BF      EWait_6
;
        MOV.L    @R5,R6                  ; Erase memory block start address -> R6
BlockVerify_1 .EQU    $                  ; Erase-verify
        MOV.L    #H'FFFFFFFF,R8
        MOV.L    R8,@R6                  ; H'FF dummy write
        MOV.L    #Wait2u,R3
EWait_7  SUBC    R2,R3
        BF      EWait_7
;
        MOV.L    @R6+,R1                 ; Read verify data
        CMP/EQ   R8,R1
        BF      BlockVerify_NG
        MOV.L    @(8,R5),R7
        CMP/EQ   R6,R7                   ; Check for last address of memory block
        BF      BlockVerify_1
        MOV.L    #Wait5u,R3
        AND.B    #EVCLEAR,@(R0,GBR)    ; Clear EV
EWait_8  SUBC    R2,R3                    ; Wait 5 µs
        BF      EWait_8
;

```



```

MOV.L    #OK,R7                ; R7 <- OK (return value)
BRA      FlashErase_end        ; Verify OK
NOP

;
BlockVerify_NG    .EQU    $
ADD.L    #1,R9                ; Verify NG, n <- n + 1
MOV.L    #Wait5u,R3
AND.B    #EVCLEAR,@(R0,GBR)    ; Clear EV
EWait_9 SUBC    R2,R3          ; Wait 5 µs
BF       EWait_9
MOV.L    #MAXErase,R7          ; If n > MAXErase then erase NG
CMP/EQ   R7,R9
BF       EraseLoop
MOV.L    #NG,R7                ; R7 <- NG (return value)
FlashErase_end    .EQU    $
MOV.L    #FLMCR1,R0
AND.B    #SWECLEAR,@(R0,GBR)    ; Clear SWE

;
RTS
NOP

;
; Memory block table    Memory block start address: EBR value
.ALIGN   4
Flash_BlockData    .EQU    $
EB0     .DATA.L    H'00000000,H'00000100
EB1     .DATA.L    H'00008000,H'00000200
EB2     .DATA.L    H'00010000,H'00000400
EB3     .DATA.L    H'00018000,H'00000800
EB4     .DATA.L    H'00020000,H'00000001
EB5     .DATA.L    H'00028000,H'00000002
EB6     .DATA.L    H'00030000,H'00000004
EB7     .DATA.L    H'00038000,H'00000008
EB8     .DATA.L    H'0003F000,H'00000010
EB9     .DATA.L    H'0003F400,H'00000020
EB10    .DATA.L    H'0003F800,H'00000040
EB11    .DATA.L    H'0003FC00,H'00000080
Dummy   .DATA.L    H'00040000

```

18.8 Protection

There are two kinds of flash memory program/erase protection, hardware protection and software protection.

18.8.1 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 register 1 (FLMCR1) and erase block register 1 (EBR1). The FLMCR1 and EBR1 settings are retained in the error-protected state. (See table 18.8.)

Table 18.8 Hardware Protection

| Item | Description | Functions | |
|--------------------------|---|-----------|-------|
| | | Program | Erase |
| FWP pin protection | <ul style="list-style-type: none">When a low level is input to the FWP pin, FLMCR1 and EBR1 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 standby mode, FLMCR1 and EBR1 are initialized, and the program/erase-protected state is entered.In a power-on 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 |

18.8.2 Software Protection

Software protection can be implemented by setting erase block register 1 (EBR1) and the RAMS bit in the RAM emulation register (RAMER). 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 18.9.)

Table 18.9 Software Protection

| Item | Description | Functions | |
|--------------------------------|--|-----------|-------|
| | | Program | Erase |
| SWE pin 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 register 1 (EBR1).Setting EBR1 to H'00 places all blocks in the erase-protected state. | — | Yes |
| Emulation protection | <ul style="list-style-type: none">Setting the RAMS bit to 1 in the RAM emulation register (RAMER) places all blocks in the program/erase-protected state. | Yes | Yes |

18.8.3 Error Protection

In error protection, an error is detected when SH7017 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 SH7017 malfunctions during flash memory programming/erasing, the FLER bit is set to 1 in FLMCR2 and the error protection state is entered. The FLMCR1 and EBR1 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 (including software standby) is executed during programming/erasing
4. When the bus is released during programming/erasing

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

Figure 18.15 shows the flash memory state transition diagram.

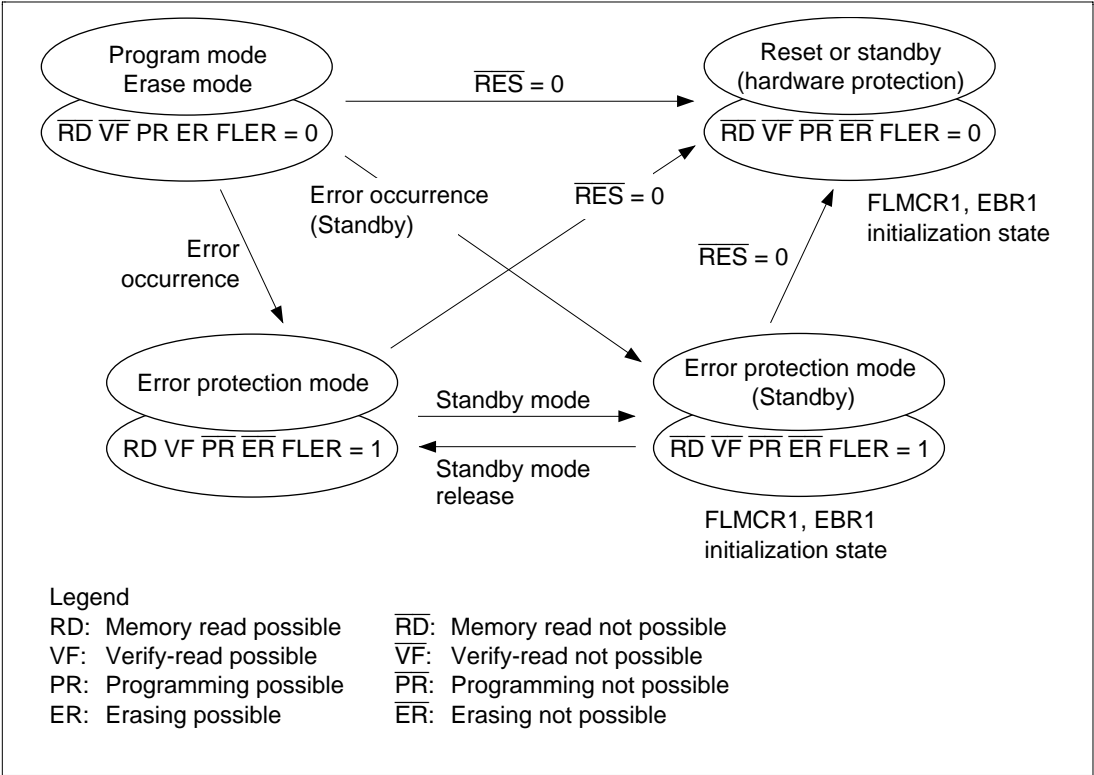


Figure 18.15 Flash Memory State Transitions

18.9 Flash Memory Emulation in RAM

Making a setting in the RAM emulation register (RAMER) enables part of RAM to be overlapped onto the flash memory area so that data to be written to flash memory can be emulated in RAM in real time. After the RAMER setting has been made, accesses cannot be made from the flash memory area or the RAM area overlapping flash memory. Emulation can be performed in user mode and user program mode. Figure 18.16 shows an example of emulation of real-time flash memory programming.

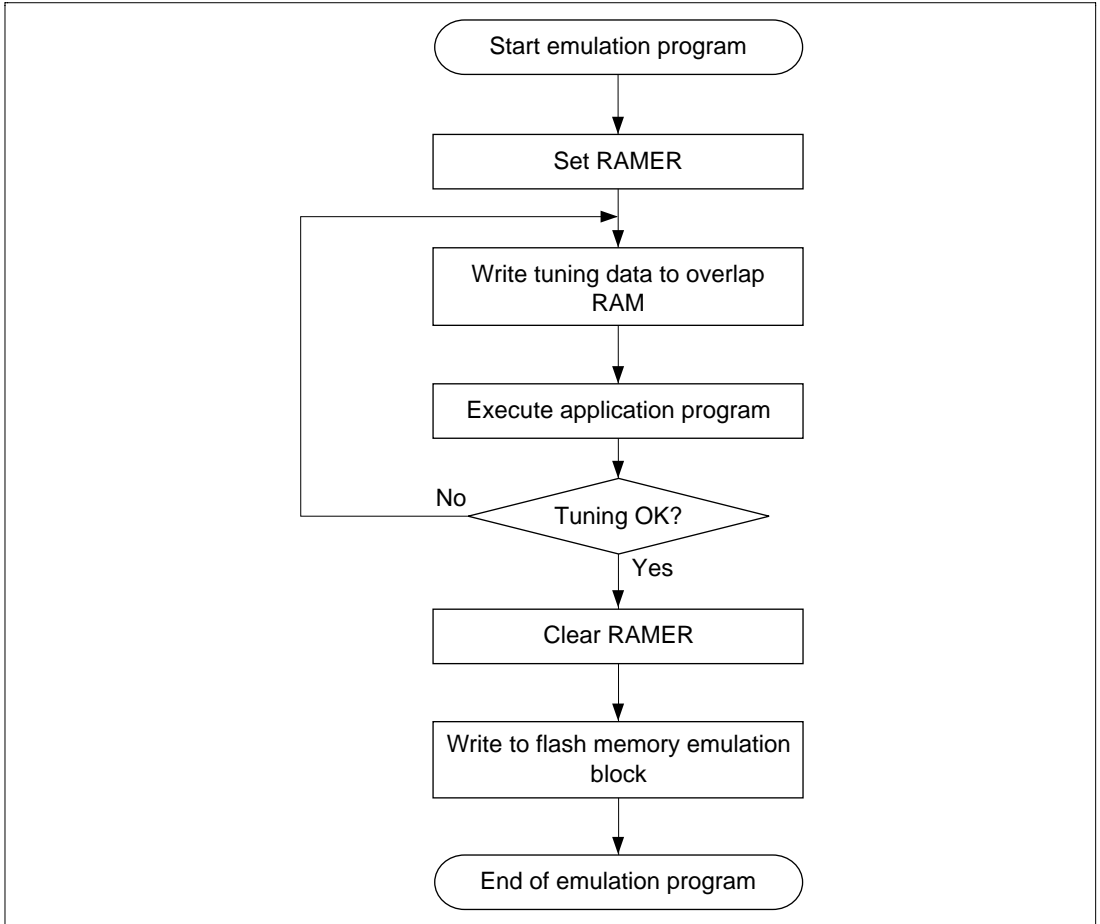


Figure 18.16 Flowchart for Flash Memory Emulation in RAM

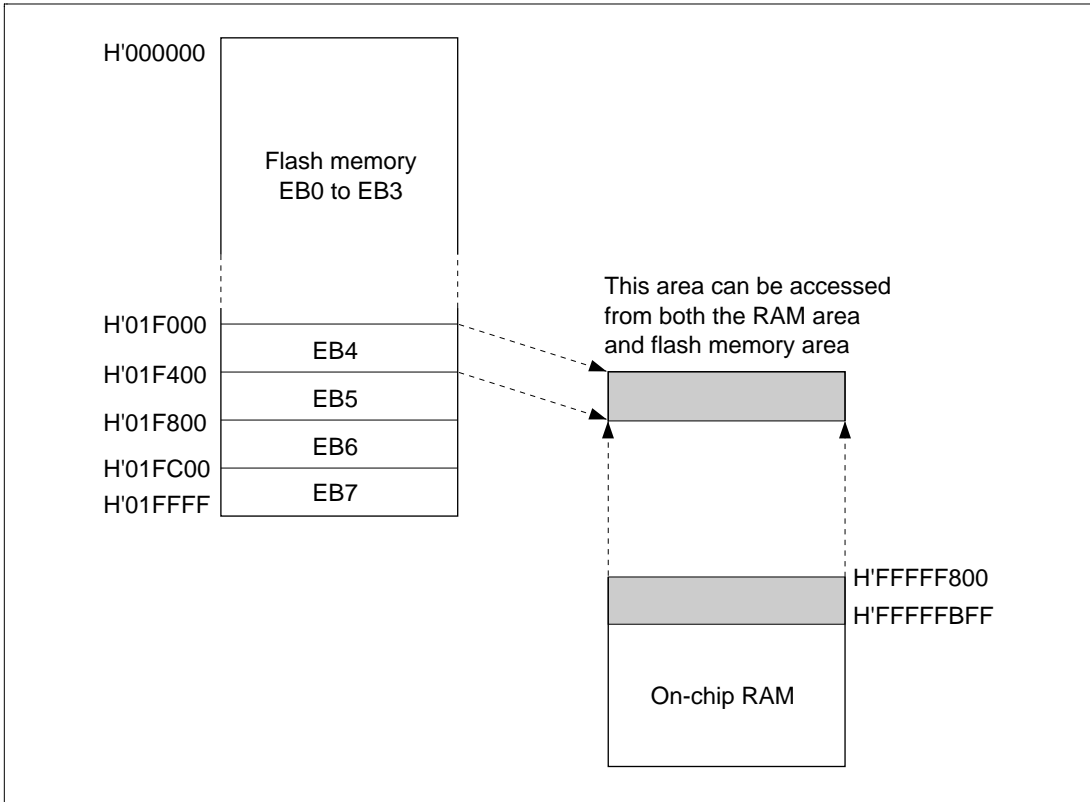


Figure 18.17 Example of RAM Overlap Operation

Example in Which Flash Memory Block Area (EB4) is Overlapped

1. Set bits RAMS, RAM1, and RAM0 in RAMER to 1, 0, 1, to overlap part of RAM onto the area (EB4) for which real-time programming is required.
2. Real-time programming is performed using the overlapping RAM.
3. After the program data has been confirmed, the RAMS bit is cleared, releasing RAM overlap.
4. The data written in the overlapping RAM is written into the flash memory space (EB4).

- Notes:
1. When the RAMS bit is set to 1, program/erase protection is enabled for all blocks regardless of the value of RAM1 and RAM0 (emulation protection). In this state, setting the P or E bit in flash memory control register 1 (FLMCR1) will not cause a transition to program mode or erase mode. When actually programming a flash memory area, the RAMS bit should be cleared to 0.
 2. A RAM area cannot be erased by execution of software in accordance with the erase algorithm while flash memory emulation in RAM is being used.

18.10 Note on Flash Memory Programming/Erasing

In the on-board programming modes (user mode and user program mode), NMI input should be disabled to give top priority to the program/erase operations (including RAM emulation).

18.11 Flash Memory Programmer Mode

Programs and data can be written and erased in programmer mode as well as in the on-board programming modes. In programmer mode, flash memory read mode, auto-program mode, auto-erase mode, and status read mode are supported. 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.

In programmer mode, set the mode pins to PLLx2 mode (see table 18.10) and input a 6 MHz input clock, so that the SH7017 runs at 12 MHz.

Table 18.10 shows the pin settings for programmer mode. For the pin names in programmer mode, see section 1.3.3, Pin Functions.)

Table 18.10 PROM Mode Pin Settings

| Pin Names | Settings |
|---|---|
| Mode pins: MD3, MD2, MD1, MD0 | 1101 (PLL × 2) |
| FWE pin | High level input (in auto-program and auto-erase modes) |
| $\overline{\text{RES}}$ pin | Power-on reset circuit |
| XTAL, EXTAL, PLLV _{CC} , PLLCAP, PLLV _{SS} pins | Oscillator circuit |

Note: In programmer mode, the FWP pin has its polarity reversed and functions as the FWE (flash write enable) pin.

18.11.1 Socket Adapter Pin Correspondence Diagram

Connect the socket adapter to the chip as shown in figure 18.19. This will enable conversion to a 32-pin arrangement. The on-chip ROM memory map is shown in figure 18.18, and the socket adapter pin correspondence diagram in figure 18.19.

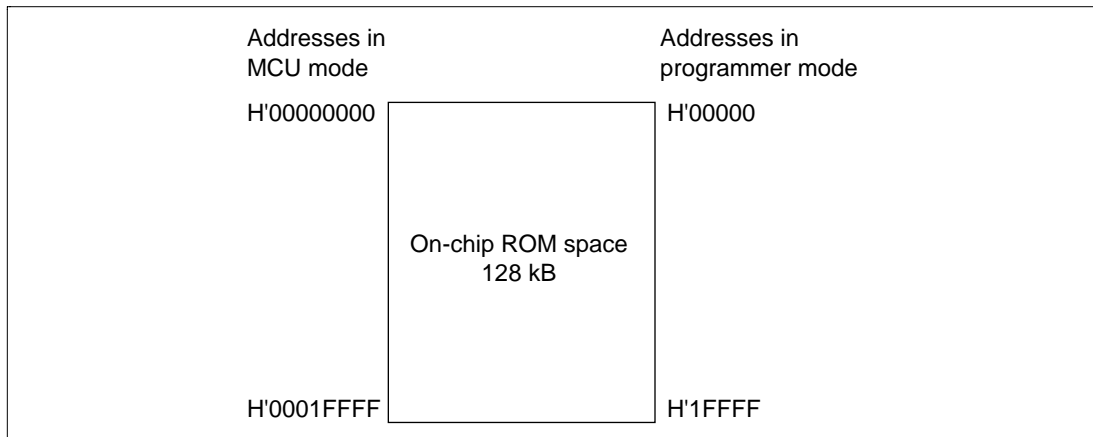
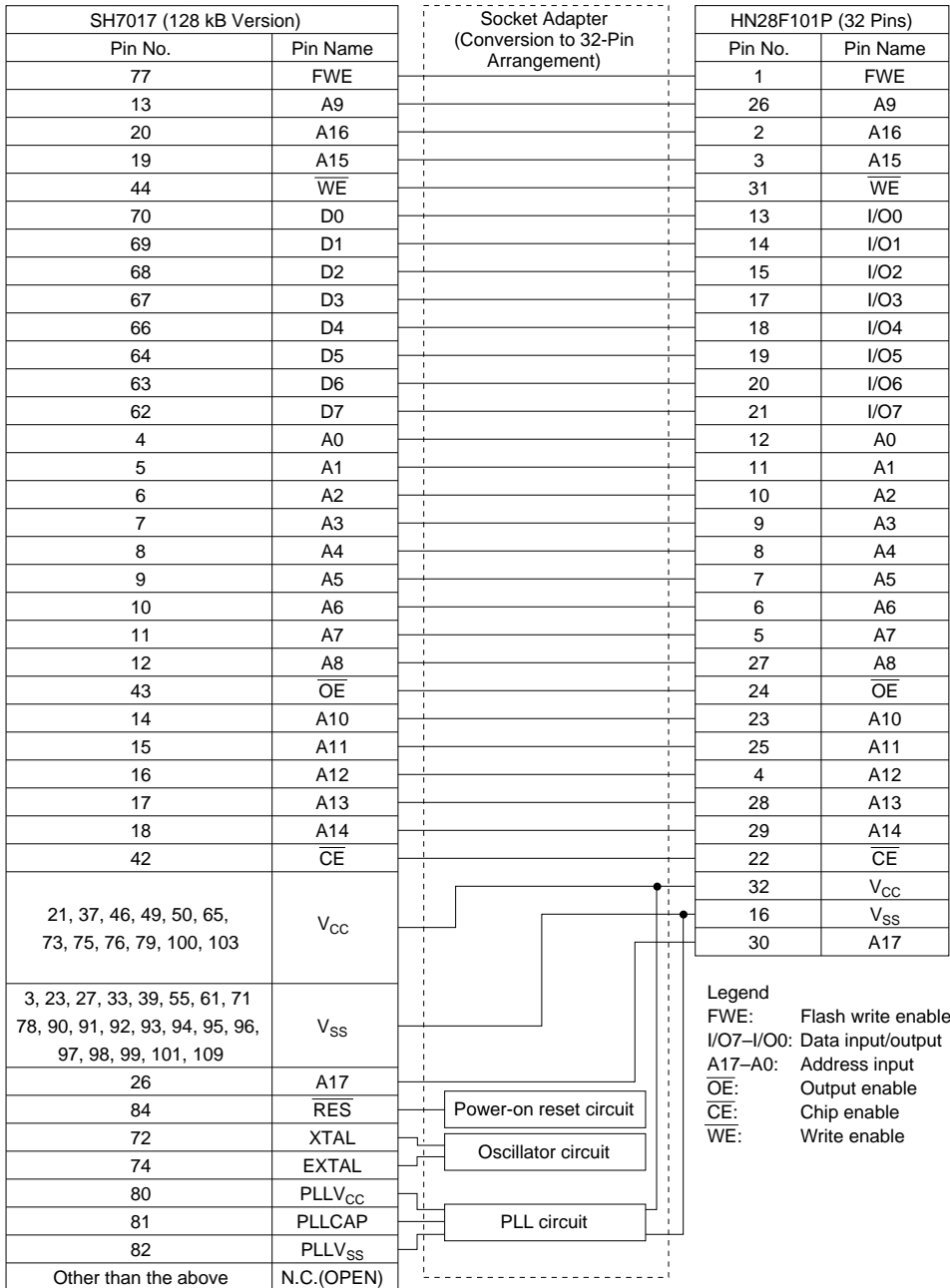


Figure 18.18 On-Chip ROM Memory Map



Note: Use address pin A17 as V_{SS}.

Figure 18.19 Socket Adapter Pin Correspondence Table (SH7017: FP-112)

18.11.2 Programmer Mode Operation

Table 18.11 shows how the different operating modes are set when using programmer mode, and table 18.12 lists the commands used in programmer mode. Details of each mode are given below.

- **Memory Read Mode**
Memory read mode supports byte reads.
- **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.
- **Auto-Erase Mode**
Auto-erase mode supports automatic erasing of the entire flash memory. Status polling is used to confirm the end of auto-programming.
- **Status Read Mode**
Status polling is used for auto-programming and auto-erasing, and normal termination can be confirmed by reading the I/O 6 signal. In status read mode, error information is output if an error occurs.

Table 18.11 Settings for Various Operating Modes In Programmer Mode

| Mode | Pin Names | | | | | |
|----------------|-----------|----|----|----|-------------|--------|
| | FWE | CE | OE | WE | I/O0–I/O7 | A0–A17 |
| 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 | L | H | L | Data input | *Ain |
| Chip disable | H or L | H | 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 in auto-program and auto-erase modes, input a high level to the FWE pin.

Table 18.12 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.

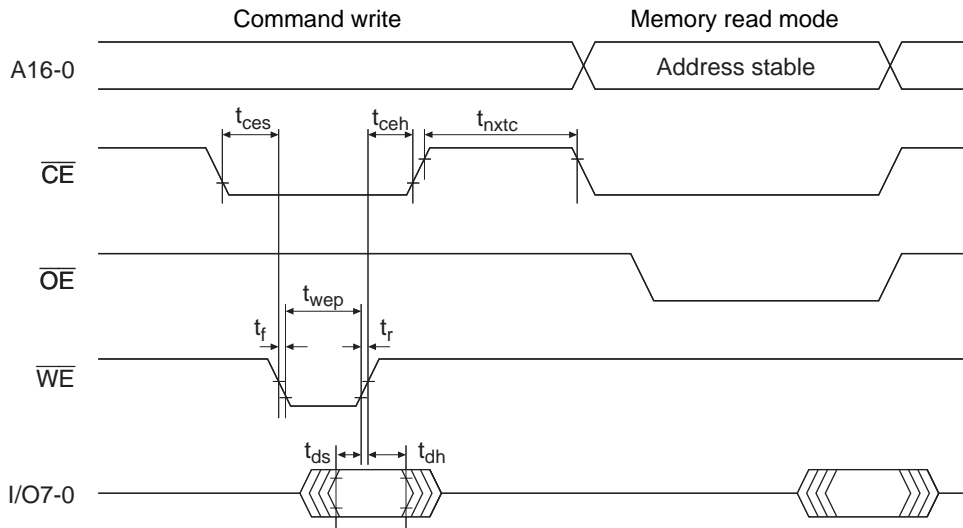
2. In memory read mode, the number of cycles depends on the number of address write cycles (n).

18.11.3 Memory Read Mode

1. After completion of auto-program/auto-erase/status read operations, a transition is made to the command wait state. When reading memory contents, a transition to memory read mode must first be made with a command write, after which the memory contents are read.
2. In memory read mode, command writes can be performed in the same way as in the command wait state.
3. Once memory read mode has been entered, consecutive reads can be performed.
4. After powering on, memory read mode is entered.

Table 18.13 AC Characteristics in Transition to 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 | Notes |
|-----------------------------------|------------|-----|-----|---------------|-------|
| 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 | |

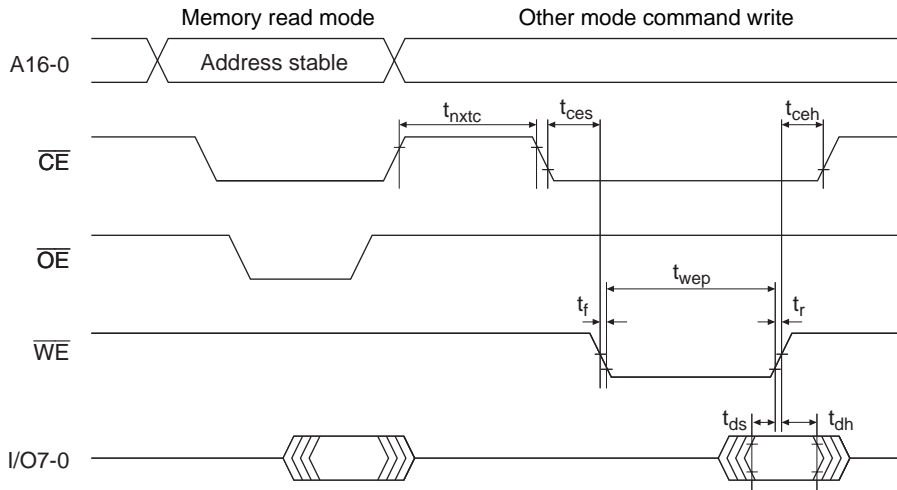


Note: Data is latched on the rising edge of \overline{WE} .

Figure 18.20 Timing Waveforms for Memory Read after Memory Write

Table 18.14 AC Characteristics in Transition from Memory Read Mode to Another 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 | Notes |
|----------------------------|------------|-----|-----|---------------|-------|
| 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 | |
| \overline{WE} rise time | t_r | | 30 | ns | |
| \overline{WE} fall time | t_f | | 30 | ns | |



Note: Do not enable \overline{WE} and \overline{OE} at the same time.

Figure 18.21 Timing Waveforms in Transition from Memory Read Mode to Another Mode

Table 18.15 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 | Notes |
|-----------------------------------|-----------|-----|-----|---------------|-------|
| Access time | t_{acc} | | 20 | μs | |
| \overline{CE} output delay time | t_{ce} | | 150 | ns | |
| \overline{OE} output delay time | t_{oe} | | 150 | ns | |
| Output disable delay time | t_{df} | | 100 | ns | |
| Data output hold time | t_{oh} | 5 | | ns | |

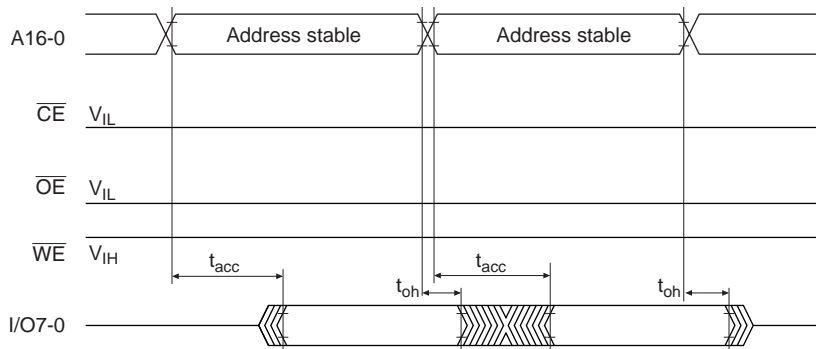


Figure 18.22 \overline{CE} and \overline{OE} Enable State Read Timing Waveforms

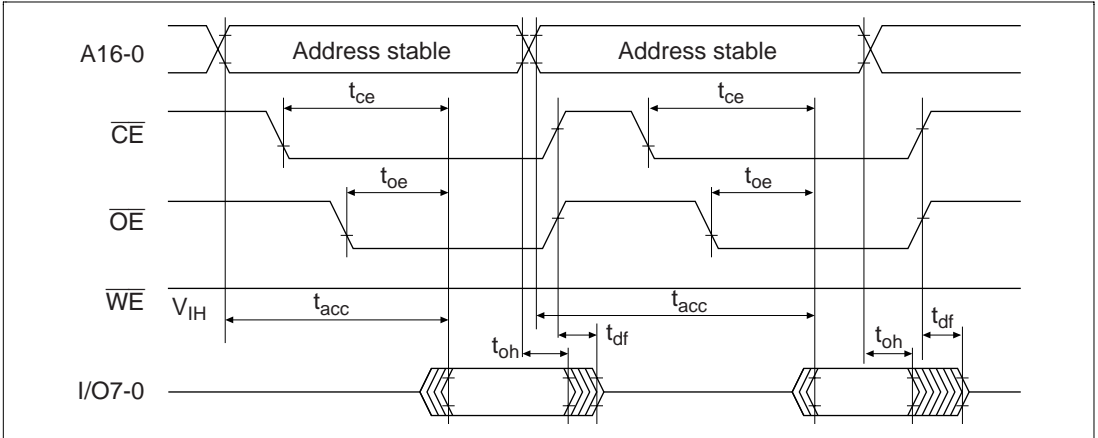


Figure 18.23 $\overline{\text{CE}}$ and $\overline{\text{OE}}$ Clock System Read Timing Waveforms

18.11.4 Auto-Program Mode

1. In auto-program mode, 128 bytes are programmed simultaneously. This should be carried out by executing 128 consecutive byte transfers.
2. 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.
3. The lower 7 bits of the transfer address must be low. 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.
4. Memory address transfer is performed in the third cycle (figure 18.24). Do not perform transfer after the second cycle.
5. Do not perform a command write during a programming operation.
6. Perform one auto-program operation for a 128-byte block for each address. Two or more additional programming operations cannot be performed on a previously programmed address block.
7. Confirm normal end of auto-programming by checking I/O6. Alternatively, status read mode can also be used for this purpose (I/O7 status polling uses the auto-program operation end identification pin).
8. Status polling I/O6 and I/O7 pin information is retained until the next command write. As long as the next command write has not been performed, reading is possible by enabling $\overline{\text{CE}}$ and $\overline{\text{OE}}$.

18.11.5 Auto-Erase Mode

1. Auto-erase mode supports only entire memory erasing.
2. Do not perform a command write during auto-erasing.
3. Confirm normal end of auto-erasing by checking I/O6. Alternatively, status read mode can also be used for this purpose (I/O7 status polling uses the auto-erase operation end identification pin).
4. Status polling I/O6 and I/O7 pin information is retained until the next command write. As long as the next command write has not been performed, reading is possible by enabling $\overline{\text{CE}}$ and $\overline{\text{OE}}$.

Table 18.17 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 | Notes |
|-----------------------------------|--------------------|-----|-------|---------------|-------|
| 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_{ests} | 1 | | ms | |
| Status polling access time | t_{spa} | | 150 | ns | |
| Memory erase time | t_{erase} | 100 | 40000 | ms | |
| Erase setup time | t_{ens} | 100 | | ns | |
| Erase end setup time | t_{enh} | 100 | | ns | |
| $\overline{\text{WE}}$ rise time | t_{r} | | 30 | ns | |
| $\overline{\text{WE}}$ fall time | t_{f} | | 30 | ns | |

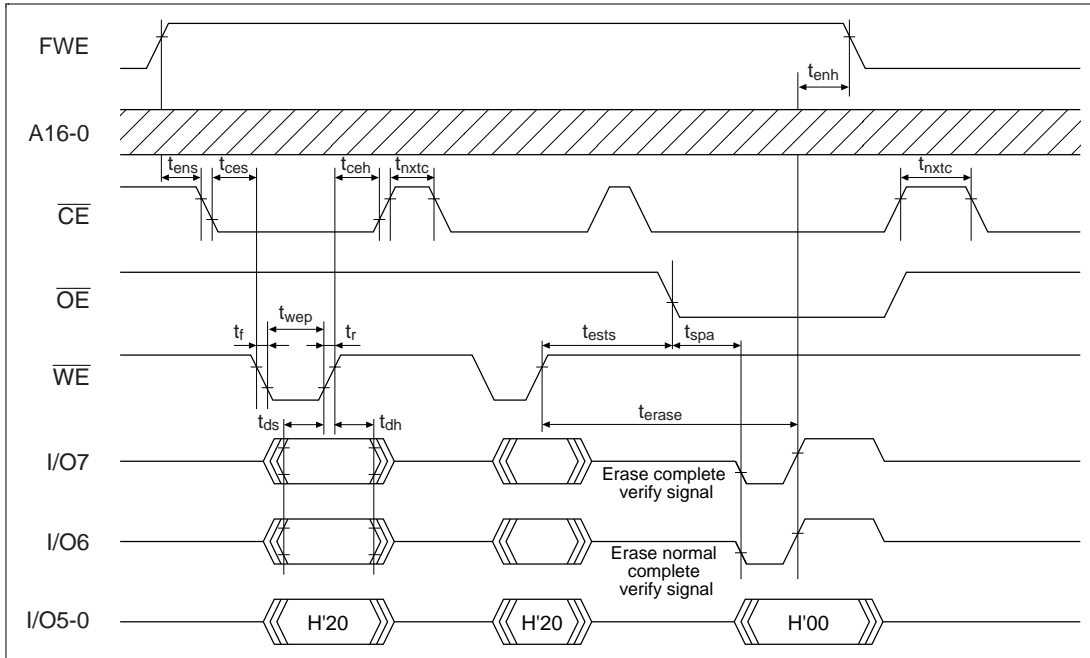


Figure 18.25 Auto-Erase Mode Timing Waveforms

18.11.6 Status Read Mode

1. Status read mode is provided to specify the kind of abnormal end. 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 other than a status read mode command write is executed.

Table 18.18 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 | Notes |
|--|------------|-----|-----|---------------|-------|
| Read time after command write | t_{strd} | 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 | |

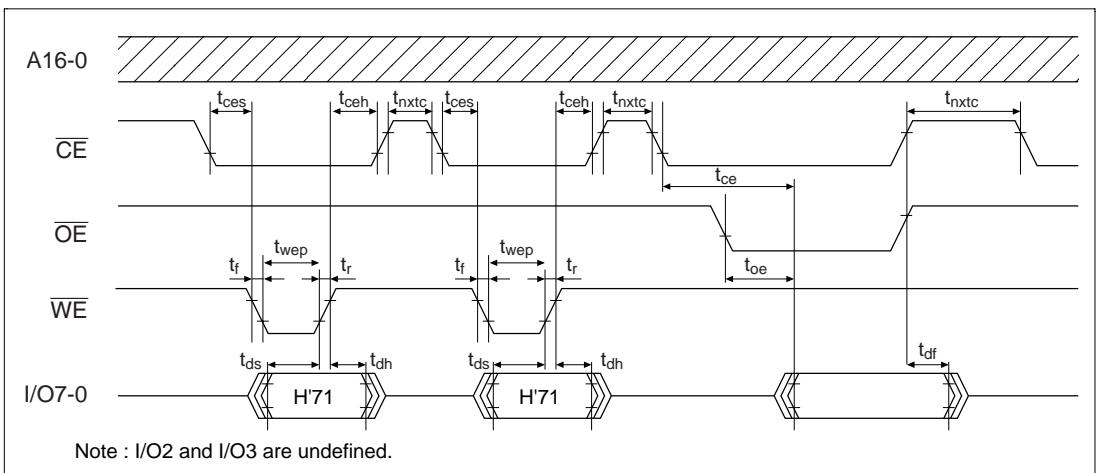


Figure 18.26 Status Read Mode Timing Waveforms

Table 18.19 Status Read Mode Return Commands

| Pin Name | I/O7 | I/O6 | I/O5 | I/O4 | I/O3 | I/O2 | I/O1 | I/O0 |
|---------------|----------------------------------|----------------------------------|--------------------------------------|----------------------------------|------|------|-------------------------------------|--|
| Attribute | Normal end identification | Command error | Programming error | Erase error | — | — | Programming or erase count exceeded | Effective address error |
| Initial value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Indications | Normal end: 0 Abnormal end: 1 | Command Error: 1 Otherwise: 0 | Programming Error: 1 Otherwise: 0 | Erasing Error: 1 Otherwise: 0 | — | — | Count exceeded: 1 Otherwise: 0 | Effective address Error: 1 Otherwise: 0 |

Note: D2 and D3 are undefined at present.

18.11.7 Status Polling

1. I/O7 status polling is a flag that indicates the operating status in auto-program/auto-erase mode.
2. I/O6 status polling is a flag that indicates a normal or abnormal end in auto-program/auto-erase mode.

Table 18.20 Status Polling Output Truth Table

| Pin Name | During Internal Operation | | Normal End | |
|------------|---------------------------|---|------------|---|
| | Abnormal End | | | |
| I/O7 | 0 | 1 | 0 | 1 |
| I/O6 | 0 | 0 | 1 | 1 |
| I/O0– I/O5 | 0 | 0 | 0 | 0 |

18.11.8 Programmer Mode Transition Time

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

Table 18.21 Stipulated Transition Times to Command Wait State

| Item | Symbol | Min | Max | Unit | Notes |
|--|------------|-----|-----|------|-------|
| 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 | |

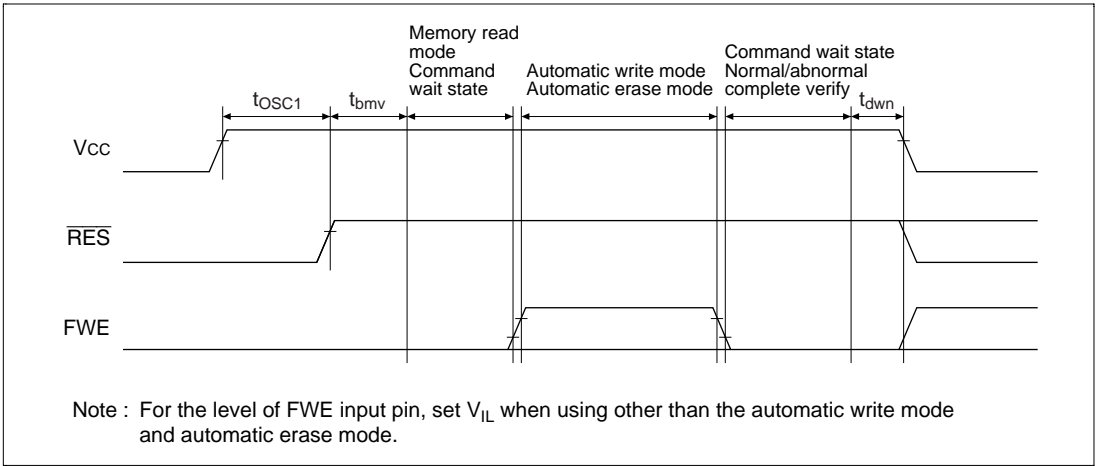


Figure 18.27 Oscillation Stabilization Time, Boot Program Transfer Time, and Power-Down Sequence

18.11.9 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 PROM 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. Additional programming cannot be performed on previously programmed address blocks.

Section 19 Mask ROM

19.1 Overview

This LSI is available with 64kbytes or 128kbytes of on-chip ROM. The on-chip ROM is connected to the CPU, direct memory access controller (DMAC) and data transfer controller (DTC) through a 32-bit data bus (figures 19.1 and 19.2). The CPU and DMAC can access the on-chip ROM in 8, 16 and 32-bit widths. Data in the on-chip ROM can always be accessed in one cycle.

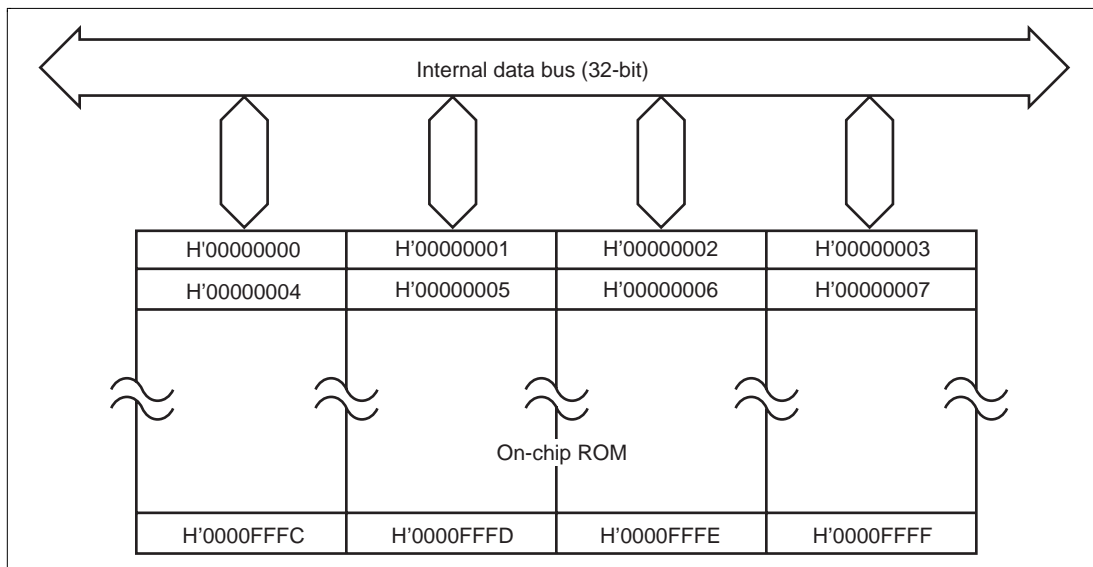


Figure 19.1 Mask ROM Block Diagram (64kbyte version)

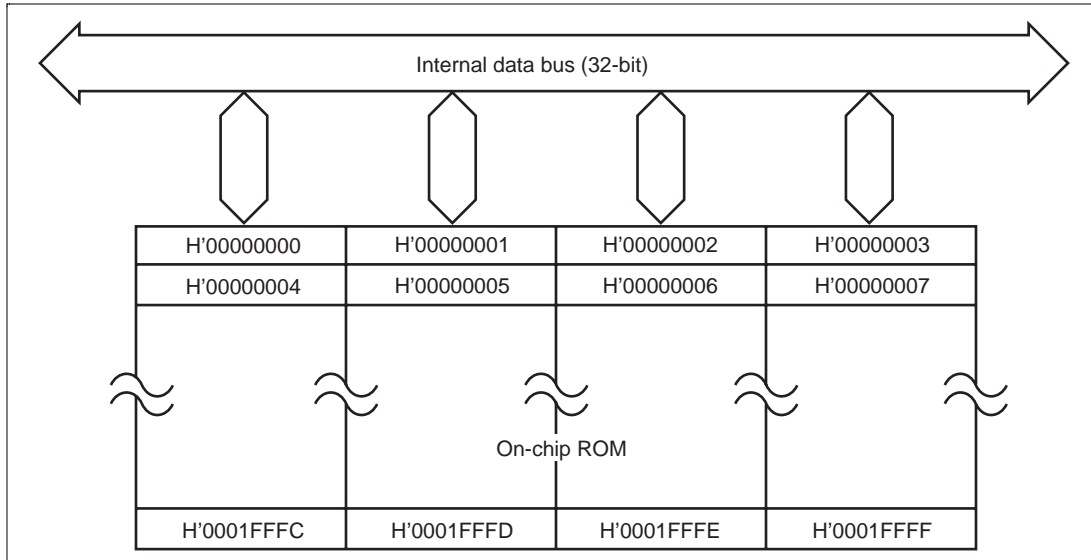


Figure 19.2 Mask ROM Block Diagram (128kbyte version)

The operating mode determines whether the on-chip ROM is valid or not. The operating mode is selected using mode-setting pins MD3–MD0 as shown in table 19.1. If you are using the on-chip ROM, select mode 2 or mode 3; if you are not, select mode 0 or 1. The on-chip ROM is allocated to addresses H'00000000–H'0000FFFF of memory area 0 for the 64kbyte version and H'00000000–H'0001FFFF of memory area 0 for the 128kbyte version.

Table 19.1 Operation Modes and ROM

| Operation Mode | Mode Setting Pin | | | | Area 0 |
|---------------------|------------------|-----|-----|-----|--|
| | MD3 | MD2 | MD1 | MD0 | |
| Mode 0 (MCU mode 0) | * | * | 0 | 0 | On-chip ROM invalid, external 8-bit space |
| Mode 1 (MCU mode 1) | * | * | 0 | 1 | On-chip ROM invalid, external 16-bit space |
| Mode 2 (MCU mode 2) | * | * | 1 | 0 | On-chip ROM valid, external space (bus width set with bus state controller) |
| Mode 3 (MCU mode 3) | * | * | 1 | 1 | On-chip ROM valid, single-chip mode |

0: Low

1: High

*: Refer to section 3, Operating Modes.

Section 20 RAM

20.1 Overview

The SH7014 and SH7016 have 3 kbytes of on-chip RAM, and the SH7017 has 4 kbytes. The on-chip RAM is linked to the CPU and direct memory access controller (DMAC) with a 32-bit data bus (figure 20.1). The CPU can access data in the on-chip RAM in 8, 16, or 32 bit widths. The DMAC can access 8 or 16 bit widths. On-chip RAM data can always be accessed in one state, making the RAM ideal for use as a program area, stack area, or data area, which require high-speed access. The contents of the on-chip RAM are held in both the sleep and standby modes. Memory area 0 addresses H'FFFFFF000 to H'FFFFFFBFF (SH7014, SH7016) or H'FFFFFF000 to H'FFFFFFFFF (SH7017) are allocated to the on-chip RAM.

The on-chip RAM is also used as cache memory. When the cache is used, 1 kbyte of on-chip RAM is available in the SH7014 and SH7016, and 2 kbytes in the SH7017. See section 7, Cache Memory, for details.

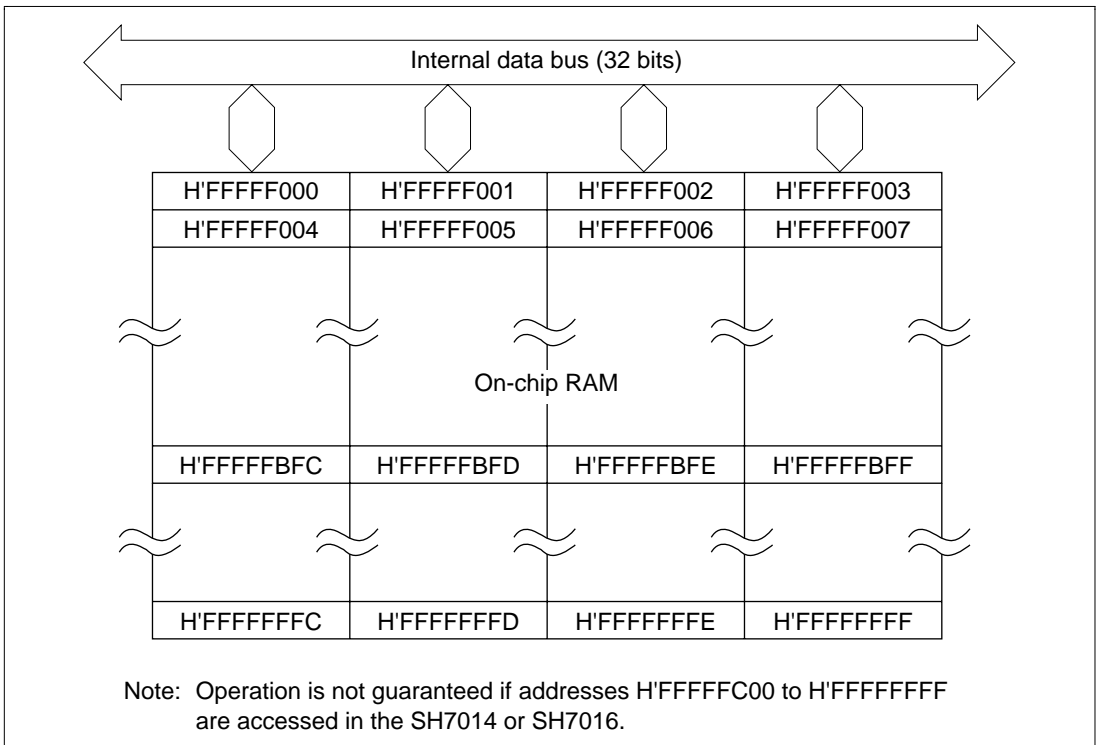


Figure 20.1 Block Diagram of RAM

Section 21 Power-Down State

21.1 Overview

In the power-down state, the CPU functions are halted. This enables a great reduction in power consumption.

21.1.1 Power-Down States

The power-down state is effected by the following two modes:

- Sleep mode
- Standby mode

Table 21.1 describes the transition conditions for entering the modes from the program execution state as well as the CPU and peripheral function status in each mode and the procedures for canceling each mode.

Table 21.1 Power-Down State Conditions

| Mode | Entering Procedure | State | | | | | | Canceling Procedure |
|----------|--|-------|------|----------------------------|---------------|------|--------------------------------------|---|
| | | Clock | CPU | On-Chip Peripheral Modules | CPU Registers | RAM | I/O Ports | |
| Sleep | Execute SLEEP instruction with SBY bit set to 0 in SBYCR | Run | Halt | Run | Held | Held | Held | <ul style="list-style-type: none"> • Interrupt • DMAC address error • Power-on reset |
| Stand-by | Execute SLEEP instruction with SBY bit set to 1 in SBYCR | Halt | Halt | Halt* ² | Held | Held | Held or high impedance* ³ | <ul style="list-style-type: none"> • NMI interrupt • Power-on reset |

- Notes:
1. SBYCR: standby control register. SBY: standby bit
 2. Some bits within on-chip peripheral module registers are initialized by the standby mode; some are not. Refer to table 21.3, Register States in the Standby Mode, in section 21.4.1, Transition to Standby Mode. Also refer to the register descriptions for each peripheral module.
 3. The status of the I/O port in standby mode is set by the port high impedance bit (HIZ) of the SBYCR. Refer to section 21.2, Standby Control Register. For pin status other than for the I/O port, refer to Appendix C, Pin Status.

21.1.2 Related Register

Table 21.2 shows the register used for power-down state control.

Table 21.2 Related Register

| Name | Abbreviation | R/W | Initial Value | Address | Access Size |
|--------------------------|--------------|-----|---------------|------------|-------------|
| Standby control register | SBYCR | R/W | H'1F | H'FFFF8614 | 8, 16, 32 |

21.2 Standby Control Register (SBYCR)

The standby control register (SBYCR) is a read/write 8-bit register that sets the transition to standby mode, and the port status in standby mode. The SBYCR is initialized to H'1F when reset.

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

- Bit 7—Standby (SBY): Specifies transition to the standby mode. The SBY bit cannot be set to 1 while the watchdog timer is running (when the timer enable bit (TME) of the WDT timer control/status register (TCSR) is set to 1). To enter the standby mode, always halt the WDT by 0 clearing the TME bit, then set the SBY bit.

| Bit 7: SBY | Description |
|------------|--|
| 0 | Executing SLEEP instruction puts the LSI into sleep mode (initial value) |
| 1 | Executing SLEEP instruction puts the LSI into standby mode |

- Bit 6—Port High Impedance (HIZ): In the standby mode, this bit selects whether to set the I/O port pin to high impedance or hold the pin status. The HIZ bit cannot be set to 1 when the TME bit of the WDT timer control/status register (TCSR) is set to 1. When making the I/O port pin status high impedance, always clear the TME bit to 0 before setting the HIZ bit.

| Bit 6: HIZ | Description |
|------------|--|
| 0 | Holds pin status while in standby mode (initial value) |
| 1 | Keeps pin at high impedance while in standby mode |

- Bits 5–0—Reserved: Bit 5 always reads as 0. Always write 0 to bit 5. Bits 4–0 always read as 1. Always write 1 to these bits.

21.3 Sleep Mode

21.3.1 Transition to Sleep Mode

Executing the SLEEP instruction when the SBY bit of SBYCR is 0 causes a transition from the program execution state to the sleep mode. Although the CPU halts immediately after executing the SLEEP instruction, the contents of its internal registers remain unchanged. The on-chip peripheral modules continue to run during the sleep mode.

21.3.2 Canceling Sleep Mode

Sleep mode is canceled by an interrupt, DMAC address error, or power-on reset.

Cancellation by an Interrupt: When an interrupt occurs, the sleep mode is canceled and interrupt exception processing is executed. The sleep mode is not canceled if the interrupt cannot be accepted because its priority level is equal to or less than the mask level set in the CPU's status register (SR) or if an interrupt by an on-chip peripheral module is disabled at the peripheral module.

Cancellation by a DMAC Address Error: If a DMAC address error occurs, the sleep mode is canceled and DMAC address error exception processing is executed.

Cancellation by a Power-On Reset: A power-on reset resulting from setting the $\overline{\text{RES}}$ pin to low level cancels the sleep mode.

21.4 Standby Mode

21.4.1 Transition to Standby Mode

To enter the standby mode, set the SBY bit to 1 in SBYCR, then execute the SLEEP instruction. The LSI moves from the program execution state to the standby mode. In the standby mode, power consumption is greatly reduced by halting not only the CPU, but the clock and on-chip peripheral modules as well. CPU register contents and on-chip RAM data are held as long as the prescribed voltages are applied. The register contents of some on-chip peripheral modules are initialized, but some are not (table 21.3). The I/O port status can be selected as held or high impedance by the port high impedance bit (HIZ) of the SBYCR. For pin status other than for the I/O port, refer to Appendix B, Pin Status.

Table 21.3 Register States in the Standby Mode

| Module | Registers Initialized | Registers that Retain Data | Registers with Undefined Contents |
|--|---|--|---|
| Interrupt controller (INTC) | — | All registers | — |
| Cache memory (CAC) | — | All registers | — |
| Bus state controller (BSC) | — | All registers | — |
| Direct memory access controller (DMAC) | <ul style="list-style-type: none"> • DMA channel control registers 0, 1 (CHCR0, CHCR1) • DMA operation register (DMAOR) | — | <ul style="list-style-type: none"> • DMA source address registers 0, 1 (SAR0, SAR1) • DMA destination address registers 0, 1 (DAR0, DAR1) • DMA transfer count registers 0, 1 (DMATCR0, DMATCR1) |
| Multifunction timer pulse unit (MTU) | MTU associated registers | — | — |
| Watchdog timer (WDT) | <ul style="list-style-type: none"> • Bits 7–5 (OVF, WT/\overline{IT}, TME) of the timer control status register (TCSR) • Reset control/status register (RSTCSR) | <ul style="list-style-type: none"> • Bits 2–0 (CKS2–CKS0) of the TCSR • Timer counter (TCNT) | — |
| Serial communication interface (SCI) | <ul style="list-style-type: none"> • Receive data register (RDR) • Transmit data register (TDR) • Serial mode register (SMR) • Serial control register (SCR) • Serial status register (SSR) • Bit rate register (BBR) | — | — |
| A/D converter (A/D) | All registers | — | — |
| Compare match timer (CMT) | All registers | — | — |

Table 21.3 Register States in the Standby Mode (cont)

| Module | Registers Initialized | Registers that Retain Data | Registers with Undefined Contents |
|-------------------------------|-----------------------|----------------------------------|-----------------------------------|
| Pin function controller (PFC) | — | All registers | — |
| I/O port (I/O) | — | All registers | — |
| Power-down state related | — | Standby control register (SBYCR) | — |

21.4.2 Canceling the Standby Mode

The standby mode is canceled by an NMI interrupt or a power-on reset.

Cancellation by an NMI: Clock oscillation starts when a rising edge or falling edge (selected by the NMI edge select bit (NMIE) of the interrupt control register (ICR) of the INTC) is detected in the NMI signal. This clock is supplied only to the watchdog timer (WDT). A WDT overflow occurs if the time established by the clock select bits (CKS2–CKS0) in the TCSR of the WDT elapses before transition to the standby mode. The occurrence of this overflow is used to indicate that the clock has stabilized, so the clock is supplied to the entire chip, the standby mode is canceled, and NMI exception processing begins.

When canceling standby mode with NMI interrupts, set the CKS2–CKS0 bits so that the WDT overflow period is longer than the oscillation stabilization time.

When canceling standby mode with an NMI pin set for falling edge, be sure that the NMI pin level upon entering standby (when the clock is halted) is high level, and that the NMI pin level upon returning from standby (when the clock starts after oscillation stabilization) is low level. When canceling standby mode with an NMI pin set for rising edge, be sure that the NMI pin level upon entering standby (when the clock is halted) is low level, and that the NMI pin level upon returning from standby (when the clock starts after oscillation stabilization) is high level.

Cancellation by a Power-On Reset: A power-on reset caused by setting the $\overline{\text{RES}}$ pin to low level cancels the standby mode.

21.4.3 Standby Mode Application Example

This example describes a transition to standby mode on the falling edge of an NMI signal, and a cancellation on the rising edge of the NMI signal. The timing is shown in figure 21.1.

When the NMI pin is changed from high to low level while the NMI edge select bit (NMIE) of the ICR is set to 0 (falling edge detection), the NMI interrupt is accepted. When the NMIE bit is set to 1 (rising edge detection) by an NMI exception service routine, the standby bit (SBY) of the SBYCR is set to 1, and a SLEEP instruction is executed, standby mode is entered. Thereafter, standby mode is canceled when the NMI pin is changed from low to high level.

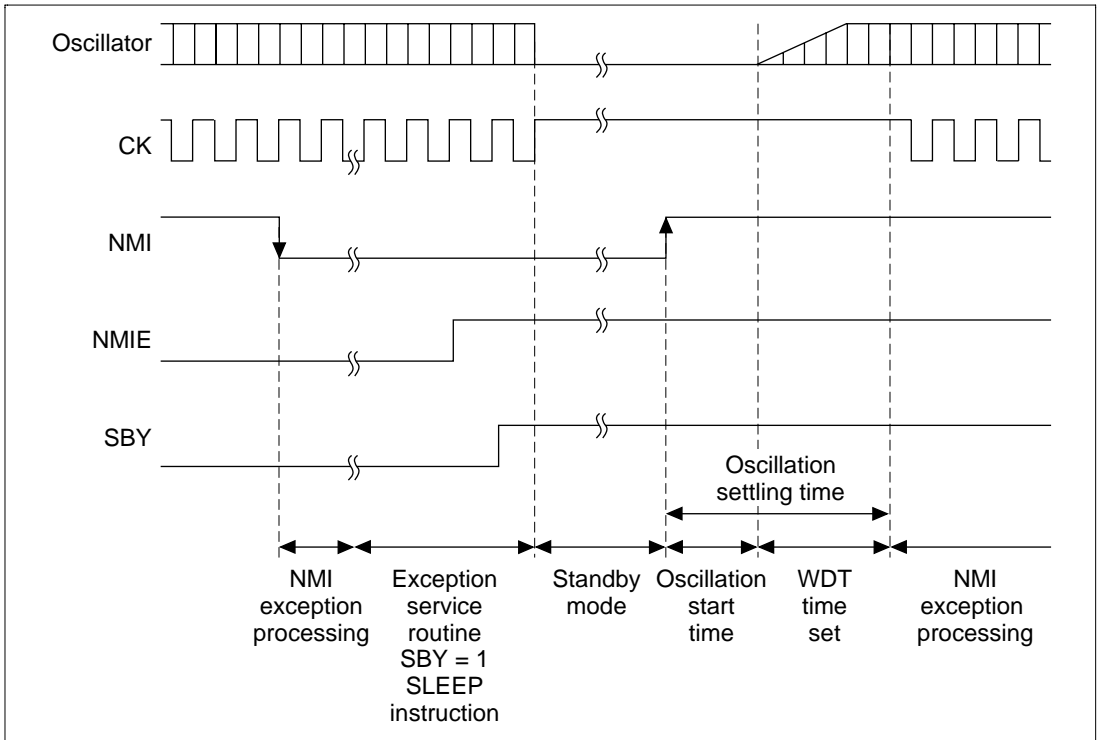


Figure 21.1 Standby Mode NMI Timing (Application Example)

Section 22 Electrical Characteristics (5 V 28.7 MHz)

22.1 Absolute Maximum Ratings

Table 22.1 shows the absolute maximum ratings.

Table 22.1 Absolute Maximum Ratings

| Item | Symbol | Rating | Unit |
|---|---------------|-------------------------|-------------|
| Power supply voltage | V_{CC} | -0.3 to +7.0 | V |
| Input voltage (other than A/D ports) | V_{in} | -0.3 to $V_{CC} + 0.3$ | V |
| Input voltage (A/D ports) | V_{in} | -0.3 to $AV_{CC} + 0.3$ | V |
| Analog supply voltage | AV_{CC} | -0.3 to +7.0 | V |
| Analog input voltage | V_{AN} | -0.3 to $AV_{CC} + 0.3$ | V |
| Operating temperature | T_{opr} | -20 to +75 | °C |
| Programming temperature (F-ZTAT version only) | T_{WE} | -20 to +75 | °C |
| Storage temperature | T_{stg} | -55 to +125 | °C |

Note: Operating the LSI in excess of the absolute maximum ratings may result in permanent damage.

22.2 DC Characteristics

Table 22.2 DC Characteristics (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{ C}$)

| Item | Pin | Symbol | Min | Typ | Max | Unit | Measurement Conditions |
|--------------------------------------|--|---------------|---------------------|-----|-----------------|---------------|---|
| Input high-level voltage | \overline{RES} , NMI, MD3– MD0, FWP, PA2, PA5, PA6 to PA9, PE0 to PR15 | V_{IH} | $V_{CC} - 0.7$ | — | $V_{CC} + 0.3$ | V | — |
| | EXTAL | | $V_{CC} \times 0.7$ | — | $V_{CC} + 0.3$ | V | — |
| | A/D port | | 2.2 | — | $AV_{CC} + 0.3$ | V | — |
| | Other input pins | | 2.2 | — | $V_{CC} + 0.3$ | V | — |
| Input low-level voltage | \overline{RES} , NMI, MD3– MD0, PA2, PA5, PA6 to PA9, PE0 to PE15 | V_{IL} | -0.3 | — | 0.5 | V | — |
| | Other input pins | | -0.3 | — | 0.8 | V | — |
| Schmitt trigger input voltage | PA2, PA5, PA6– PA9, PE0–PE15 | $VT^+ - VT^-$ | 0.4 | — | — | V | $VT^+ \geq V_{CC} - 0.7$ (max) $VT^- \leq 0.5$ (min) |
| Input leakage current | \overline{RES} , NMI, MD3– MD0, PA2, PA5, PA6–PA9, PE0– PE15 | $ I_{in} $ | — | — | 1.0 | μA | $V_{in} = 0.5$ to $V_{CC} - 0.5\text{ V}$ |
| | A/D port | | — | — | 1.0 | μA | $V_{in} = 0.5$ to $AV_{CC} - 0.5\text{ V}$ |
| | Other input pins | | — | — | 1.0 | μA | $V_{in} = 0.5$ to $V_{CC} - 0.5\text{ V}$ |
| Three-state leak current (while off) | A21–A0, D15– D0, $\overline{CS3}$ – $\overline{CS0}$, \overline{RDWR} , \overline{RAS} , \overline{CASxx} , \overline{WRxx} , \overline{RD} , ports A, B, E | $ I_{TSI} $ | — | — | 1.0 | μA | $V_{in} = 0.5$ to $V_{CC} - 0.5\text{ V}$ |
| Output high-level voltage | All output pins | V_{OH} | $V_{CC} - 0.5$ | — | — | V | $I_{OH} = -200\ \mu\text{A}$ |
| | | | 3.5 | — | — | V | $I_{OH} = -1\ \text{mA}$ |

Table 22.2 DC Characteristics (Conditions: $V_{CC} = 5.0 \text{ V} \pm 10\%$, $AV_{CC} = 5.0 \text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0 \text{ V}$, $T_a = -20 \text{ to } +75^\circ \text{ C}$) (cont)

| Item | Pin | Symbol | Min | Typ | Max | Unit | Measurement Conditions |
|--|-------------------------|-----------|-----|------|-----|---------------|--|
| Output low-level voltage | All output pins | V_{OL} | — | — | 0.4 | V | $I_{OL} = 1.6 \text{ mA}$ |
| Input capacitance | $\overline{\text{RES}}$ | C_{in} | — | — | 80 | pF | $V_{in} = 0 \text{ V}$, $f = 1 \text{ MHz}$, $T_a = 25^\circ \text{ C}$ |
| | NMI | | — | — | 50 | pF | |
| | All other input pins | | — | — | 20 | pF | |
| Current consumption (SH7014) | Ordinary operation | I_{CC} | — | 130 | 180 | mA | $f = 28 \text{ MHz}$ |
| | Sleep | | — | 100 | 150 | mA | $f = 28 \text{ MHz}$ |
| | Standby | | — | 0.01 | 5 | μA | $T_a \leq 50^\circ \text{ C}$ |
| | | | — | — | 20 | μA | $T_a > 50^\circ \text{ C}$ |
| Current consumption (SH7016, SH7017) | Ordinary operation | I_{CC} | — | 140 | 180 | mA | $f = 28 \text{ MHz}$ |
| | Sleep | | — | 110 | 150 | mA | $f = 28 \text{ MHz}$ |
| | Standby | | — | 0.01 | 5 | μA | $T_a \leq 50^\circ \text{ C}$ |
| | | | — | — | 20 | μA | $T_a > 50^\circ \text{ C}$ |
| Analog supply current (SH7014) | | AI_{CC} | — | 13 | 22 | mA | |
| Analog supply current (SH7016, SH7017) | | AI_{CC} | — | 5 | 10 | mA | |
| RAM standby voltage | | V_{RAM} | 2.0 | — | — | V | |

- Notes:
1. When the A/D converter is not used (including during standby), do not release the AV_{CC} and AV_{SS} pins. Connect the AV_{CC} pin to V_{CC} and the AV_{SS} pin to V_{SS} .
 2. The current consumption is measured when $V_{IH\text{min}} = V_{CC} - 0.5 \text{ V}$, $V_{IL\text{max}} = 0.5 \text{ V}$, with all output pins unloaded.
 3. The F-ZTAT and mask versions have the same functions, and the electrical characteristics of both are within specification, but characteristic-related performance values, operating margins, noise margins, noise emission, etc., are different. Caution is therefore required in carrying out system design, and when switching between F-ZTAT and mask versions.

Table 22.3 Permitted Output Current Values (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{ C}$)

| Item | Symbol | Min | Typ | Max | Unit |
|---|------------------|-----|-----|-----|------|
| Output low-level permissible current (per pin) | I_{OL} | — | — | 2.0 | mA |
| Output low-level permissible current (total) | $\sum I_{OL}$ | — | — | 80 | mA |
| Output high-level permissible current (per pin) | $-I_{OH}$ | — | — | 2.0 | mA |
| Output high-level permissible current (total) | $\sum (-I_{OH})$ | — | — | 25 | mA |

Note: To assure LSI reliability, do not exceed the output values listed in this table.

22.3 AC Characteristics

22.3.1 Clock Timing

Table 22.4 Clock Timing (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{ C}$)

| Item | Symbol | Min | Max | Unit | Figures |
|--|-------------|------|------|------|---------|
| Operating frequency | f_{OP} | 4 | 28.7 | MHz | 22.1 |
| Clock cycle time | t_{cyc} | 34.8 | 250 | ns | |
| Clock low-level pulse width | t_{CL} | 10 | — | ns | |
| Clock high-level pulse width | t_{CH} | 10 | — | ns | |
| Clock rise time | t_{CR} | — | 5 | ns | |
| Clock fall time | t_{CF} | — | 5 | ns | |
| EXTAL clock input frequency | f_{EX} | 4 | 10 | MHz | 22.2 |
| EXTAL clock input cycle time | t_{EXcyc} | 100 | 250 | ns | |
| EXTAL clock low-level input pulse width | t_{EXL} | 40 | — | ns | |
| EXTAL clock high-level input pulse width | t_{EXH} | 40 | — | ns | |
| EXTAL clock input rise time | t_{EXR} | — | 5 | ns | |
| EXTAL clock input fall time | t_{EXF} | — | 5 | ns | |
| Reset oscillation settling time | t_{OSC1} | 10 | — | ms | 22.3 |
| Standby return clock settling time | t_{OSC2} | 10 | — | ms | |

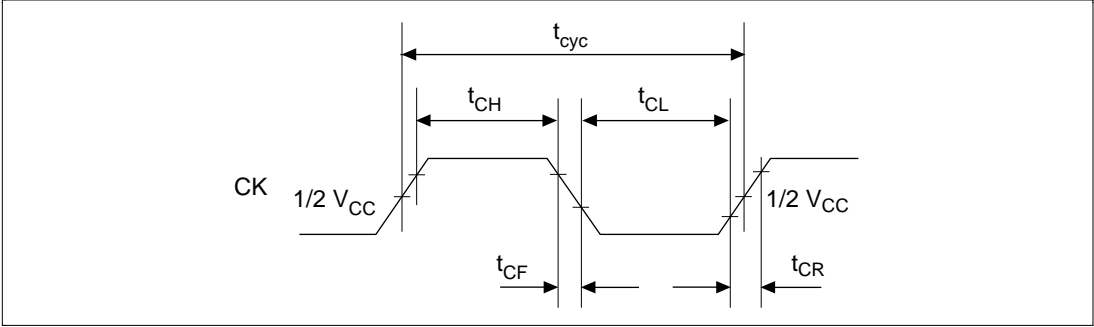


Figure 22.1 System Clock Timing

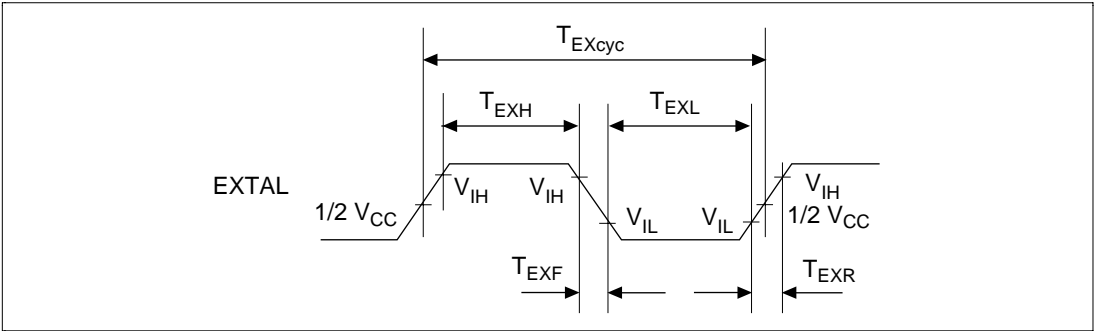


Figure 22.2 EXTAL Clock Input Timing

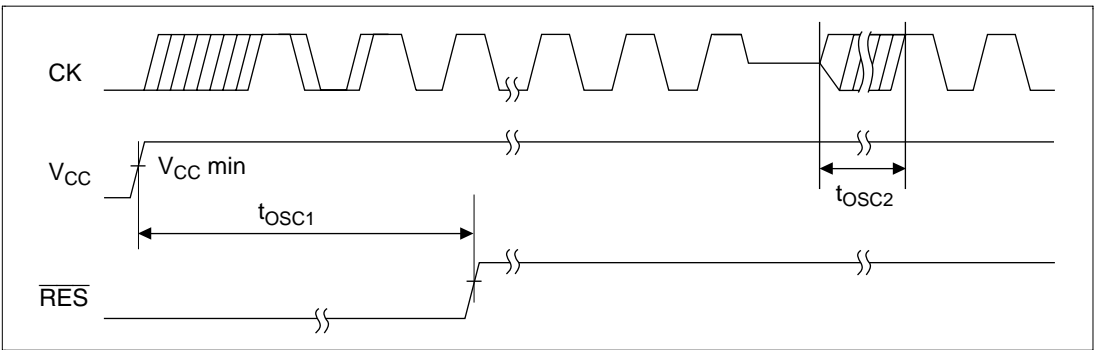


Figure 22.3 Oscillation Settling Time

22.3.2 Control Signal Timing

Table 22.5 Control Signal Timing (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{ C}$)

| Item | Symbol | Min | Max | Unit | Figure |
|---|------------------------------------|-----|-----|------------------|--------|
| $\overline{\text{RES}}$ rise/fall | $t_{\text{RESr}}, t_{\text{RESf}}$ | — | 200 | ns | 22.4 |
| $\overline{\text{RES}}$ pulse width | t_{RESW} | 20 | — | t_{cyc} | |
| NMI rise/fall | $t_{\text{NMIr}}, t_{\text{NMIf}}$ | — | 200 | ns | |
| $\overline{\text{RES}}$ setup time* | t_{RESS} | 35 | — | ns | 22.4, |
| NMI setup time* | t_{NMIS} | 35 | — | ns | 22.5 |
| $\overline{\text{IRQ7}}, \overline{\text{IRQ6}}, \overline{\text{IRQ3}}\text{--}\overline{\text{IRQ0}}$ setup time (edge detection)* | t_{IRQES} | 35 | — | ns | |
| $\overline{\text{IRQ7}}, \overline{\text{IRQ6}}, \overline{\text{IRQ3}}\text{--}\overline{\text{IRQ0}}$ setup time (level detection)* | t_{IRQLS} | 35 | — | ns | |
| NMI hold time | t_{NMIH} | 35 | — | ns | 22.5 |
| $\overline{\text{IRQ7}}, \overline{\text{IRQ6}}, \overline{\text{IRQ3}}\text{--}\overline{\text{IRQ0}}$ hold time | t_{IRQEH} | 35 | — | ns | |

Note: * The $\overline{\text{RES}}$, NMI, $\overline{\text{IRQ7}}$, $\overline{\text{IRQ6}}$, and $\overline{\text{IRQ3}}\text{--}\overline{\text{IRQ0}}$ signals are asynchronous inputs, but when the setup times shown here are provided, the signals are considered to have produced changes at clock rise (for $\overline{\text{RES}}$) or clock fall (for NMI, $\overline{\text{IRQ7}}$, $\overline{\text{IRQ6}}$, and $\overline{\text{IRQ3}}\text{--}\overline{\text{IRQ0}}$). If the setup times are not provided, recognition is delayed until the next clock rise or fall.

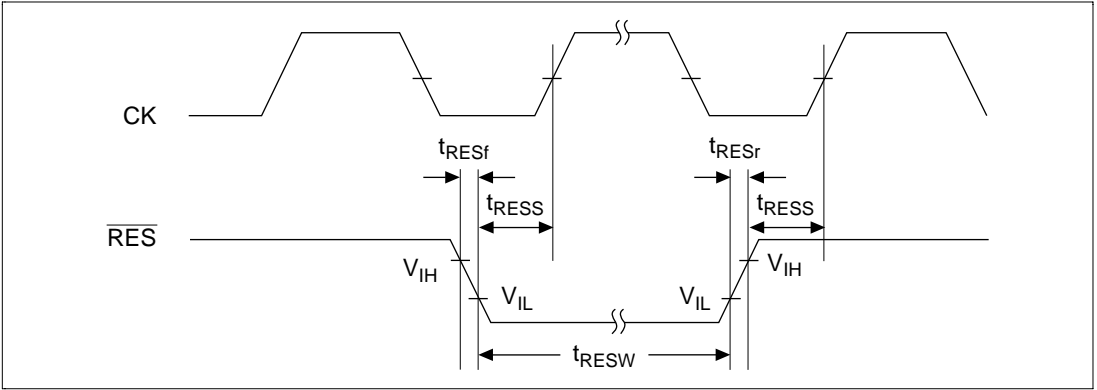


Figure 22.4 Reset Input Timing

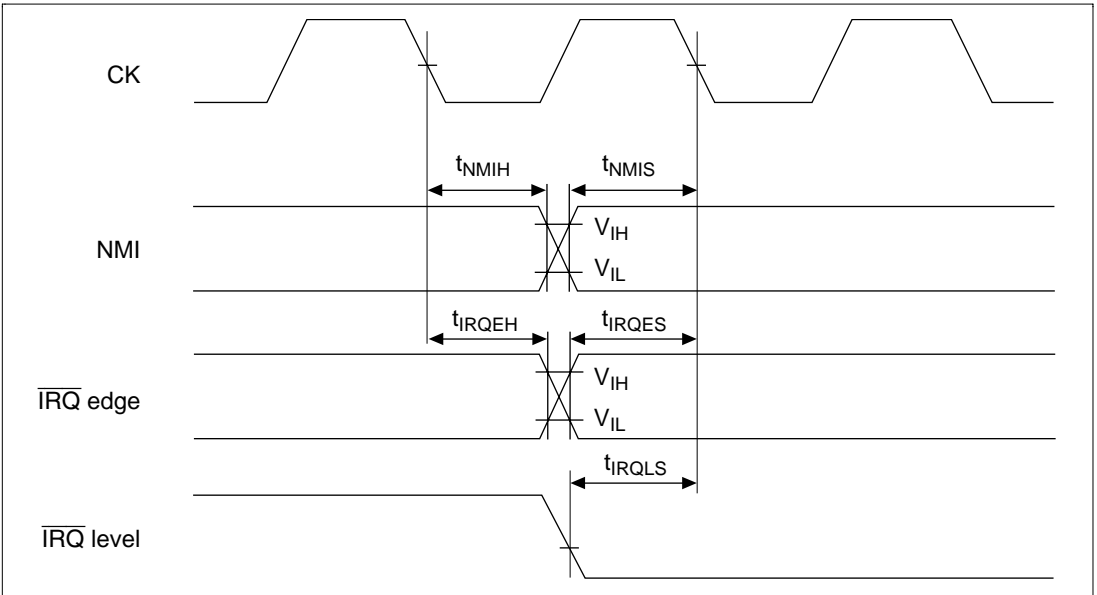


Figure 22.5 Interrupt Signal Input Timing

22.3.3 Bus Timing

Table 22.6 Bus Timing (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $AV_{CC} = 5.0\text{ V} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit | Figure |
|-----------------------------------|----------------|---------------------------------------|-----------|------|-------------------------------|
| Address delay time | t_{AD} | 2^{*3} | 18 | ns | 22.6, 22.7, 22.9–22.14, 22.17 |
| \overline{CS} delay time 1 | t_{CSD1} | 2^{*3} | 21 | ns | 22.6, 22.7, 22.17 |
| \overline{CS} delay time 2 | t_{CSD2} | 2^{*3} | 21 | ns | |
| Read strobe delay time 1 | t_{RSD1} | 2^{*3} | 18 | ns | 22.6, 22.7, 22.9–22.14, 22.17 |
| Read strobe delay time 2 | t_{RSD2} | 2^{*3} | 18 | ns | |
| Read data setup time | t_{RDS}^{*4} | 15 | — | ns | |
| Read data hold time | t_{RDH} | 0 | — | ns | |
| Write strobe delay time 1 | t_{WSD1} | 2^{*3} | 18 | ns | |
| Write strobe delay time 2 | t_{WSD2} | 2^{*3} | 18 | ns | |
| Write data delay time | t_{WDD} | — | 35 | ns | |
| Write data hold time | t_{WDH} | 0 | 10^{*2} | ns | |
| \overline{WAIT} setup time | t_{WTS} | 15 | — | ns | 22.8, 22.13, 22.17 |
| \overline{WAIT} hold time | t_{WTH} | 0 | — | ns | |
| \overline{RAS} delay time 1 | t_{RASD1} | 2^{*3} | 18 | ns | 22.9–22.16 |
| \overline{RAS} delay time 2 | t_{RASD2} | 2^{*3} | 18 | ns | |
| \overline{CAS} delay time 1 | t_{CASD1} | 2^{*3} | 18 | ns | |
| \overline{CAS} delay time 2 | t_{CASD2} | 2^{*3} | 18 | ns | |
| Read data access time | t_{ACC}^{*1} | $t_{cyc} \times (n + 2) - 40$ | — | ns | 22.6, 22.7 |
| Access time from read strobe | t_{OE}^{*1} | $t_{cyc} \times (n + 1.5) - 40$ | — | ns | |
| Access time from column address | t_{AA}^{*1} | $t_{cyc} \times (n + 2) - 40$ | — | ns | 22.9–22.14 |
| Access time from \overline{RAS} | t_{RAC}^{*1} | $t_{cyc} \times (n + RCD + 2.5) - 40$ | — | ns | |
| Access time from \overline{CAS} | t_{CAC}^{*1} | $t_{cyc} \times (n + 1) - 40$ | — | ns | |
| Row address hold time | t_{RAH} | $t_{cyc} \times (RCD + 0.5) - 15$ | — | ns | |
| Row address setup time | t_{ASR}^{*5} | $t_{cyc} \times 0.5 - 17.5$ | — | ns | |
| Data input setup time | t_{DS} | $t_{cyc} \times (m + 0.5) - 25$ | — | ns | |
| Data input hold time | t_{DH} | 20 | — | ns | |

Table 22.7 Bus Timing (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $\Delta V_{CC} = V_{CC} \pm 10\%$, $\Delta V_{CC} = V_{CC} \pm 10\%$, $V_{SS} = \Delta V_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit | Figure |
|---|--------------|-----------------------------------|-----|------|-------------------------------|
| Write address setup time | t_{AS} | 0 | — | ns | 22.6–22.7 |
| Write address hold time | t_{WR} | 5 | — | ns | |
| Write data hold time | t_{WRH} | 0 | — | ns | |
| Read/write strobe delay time 1 | t_{RWD1} | 2^{*3} | 18 | ns | 22.9–22.14 |
| Read/write strobe delay time 2 | t_{RWD2} | 2^{*3} | 18 | ns | |
| High-speed page mode CAS precharge time | t_{CP} | $t_{cyc} - 25$ | — | ns | 22.14 |
| RAS precharge time | t_{RP} | $t_{cyc} \times (TPC + 1.5) - 15$ | — | ns | 22.9–22.14 |
| CAS setup time | t_{CSR} | 10 | — | ns | 22.15, 22.16 |
| AH delay time 1 | t_{AHD1} | 2^{*3} | 18 | ns | 22.17 |
| AH delay time 2 | t_{AHD2} | 2^{*3} | 18 | ns | |
| Multiplex address delay time | t_{MAD} | 2^{*3} | 18 | ns | |
| Multiplex address hold time | t_{MAH} | 0 | — | ns | |
| DACK delay time | t_{DACKD1} | 2^{*3} | 21 | ns | 22.6, 22.7, 22.9–22.14, 22.17 |

Notes: n is the number of waits. m is 0 when the number of DRAM write cycle waits is 0, and 1 otherwise. RCD is the set value of the RCD bit in DCR.

1. If the access time is satisfied, t_{RDS} need not be satisfied.
2. t_{WDH} (max) is a reference value.
3. The delay time Min values are reference values (typ).
4. t_{RDS} is a reference value.
5. At 28.7 MHz, $t_{ASR} = 0\text{ ns}$ (min)

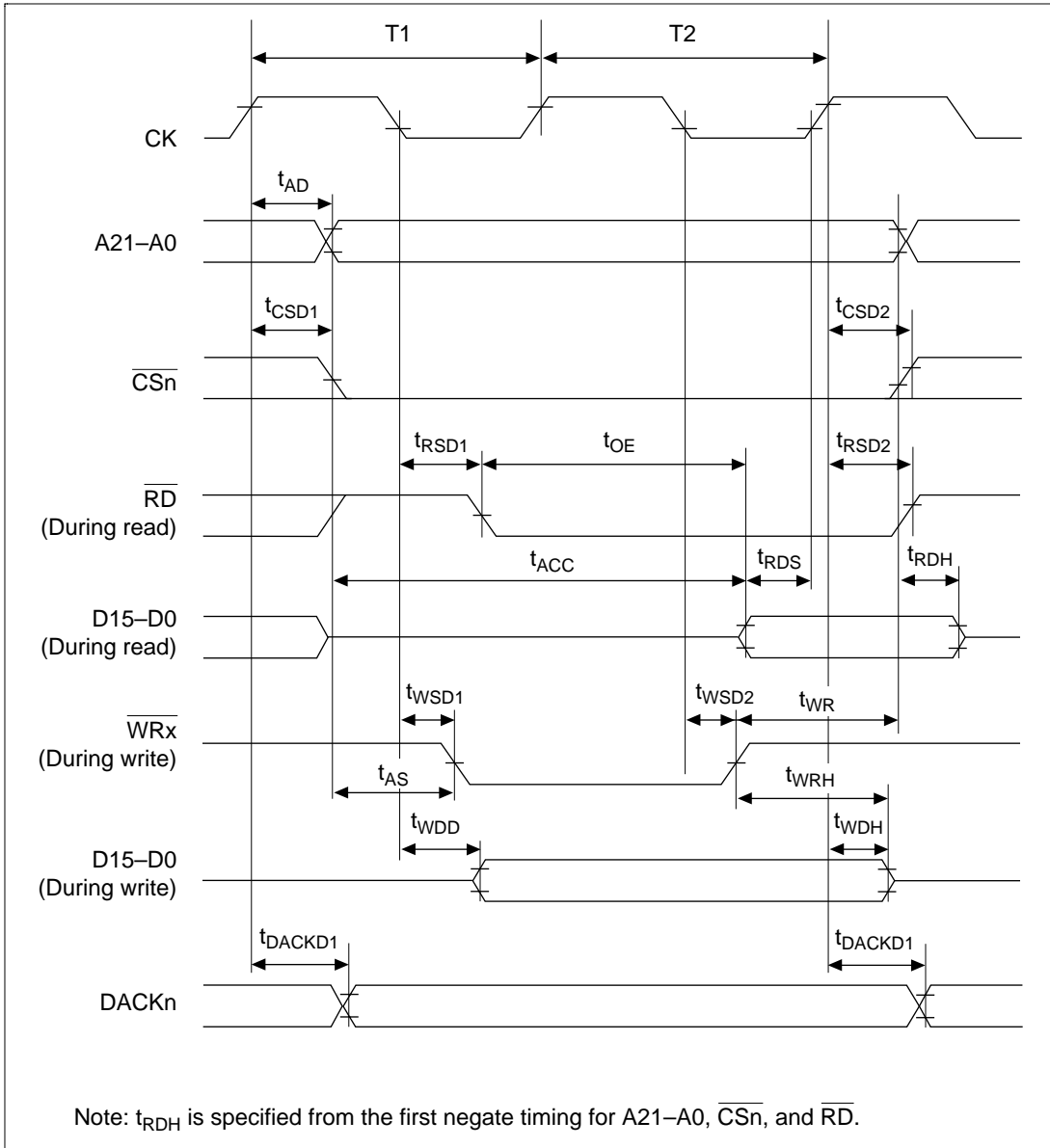
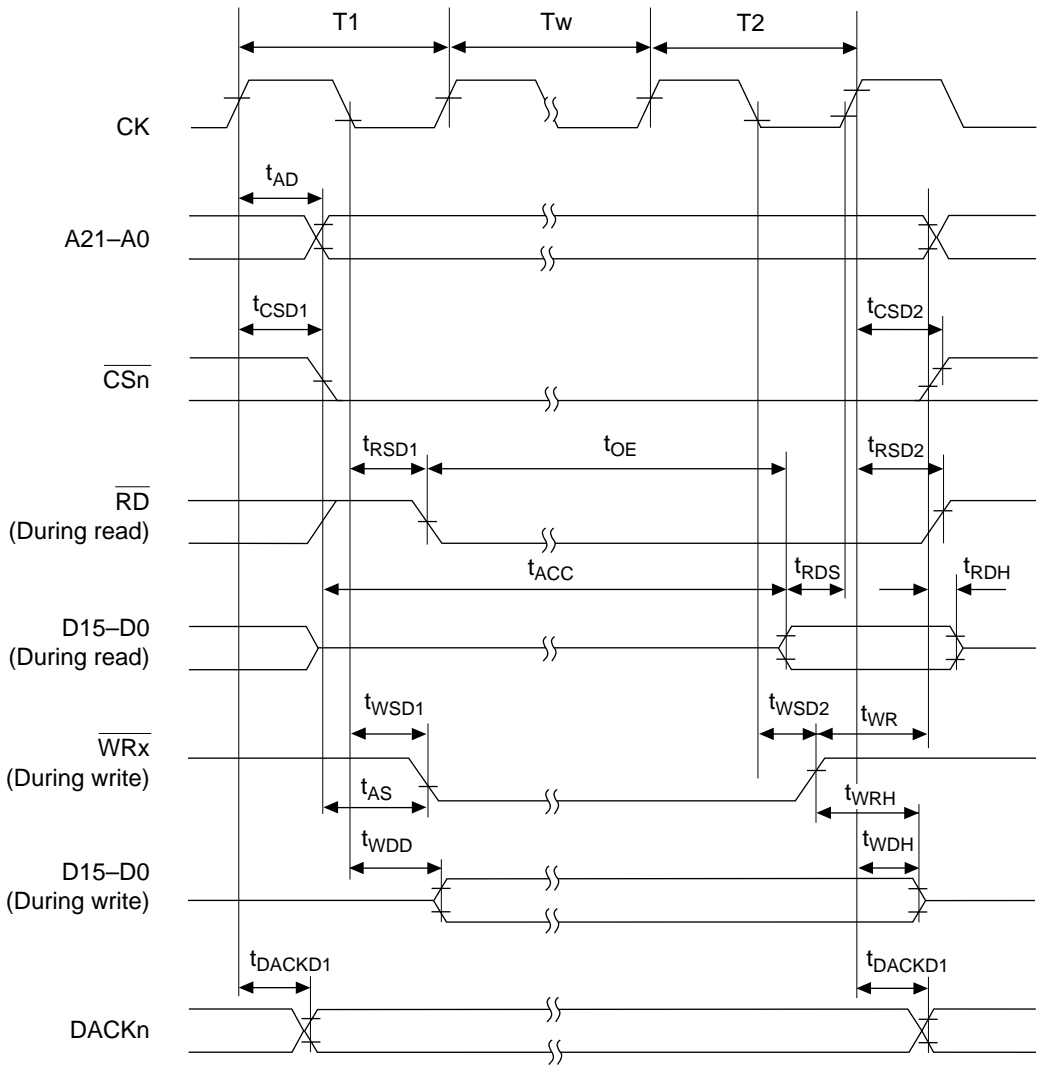


Figure 22.6 Basic Cycle (No Waits)



Note: t_{RDH} is specified from the first negate timing for A21-A0, \overline{CS}_n , and \overline{RD} .

Figure 22.7 Basic Cycle (Software Waits)

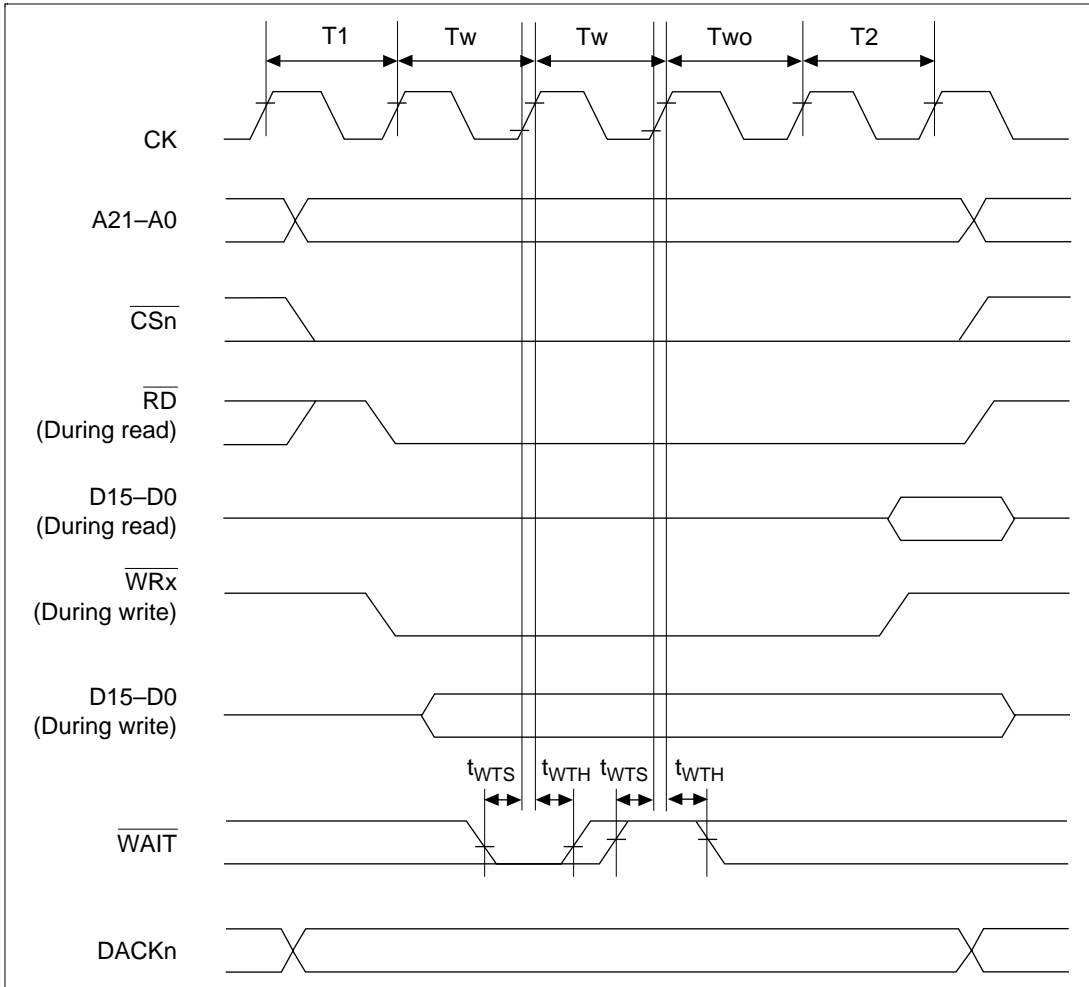
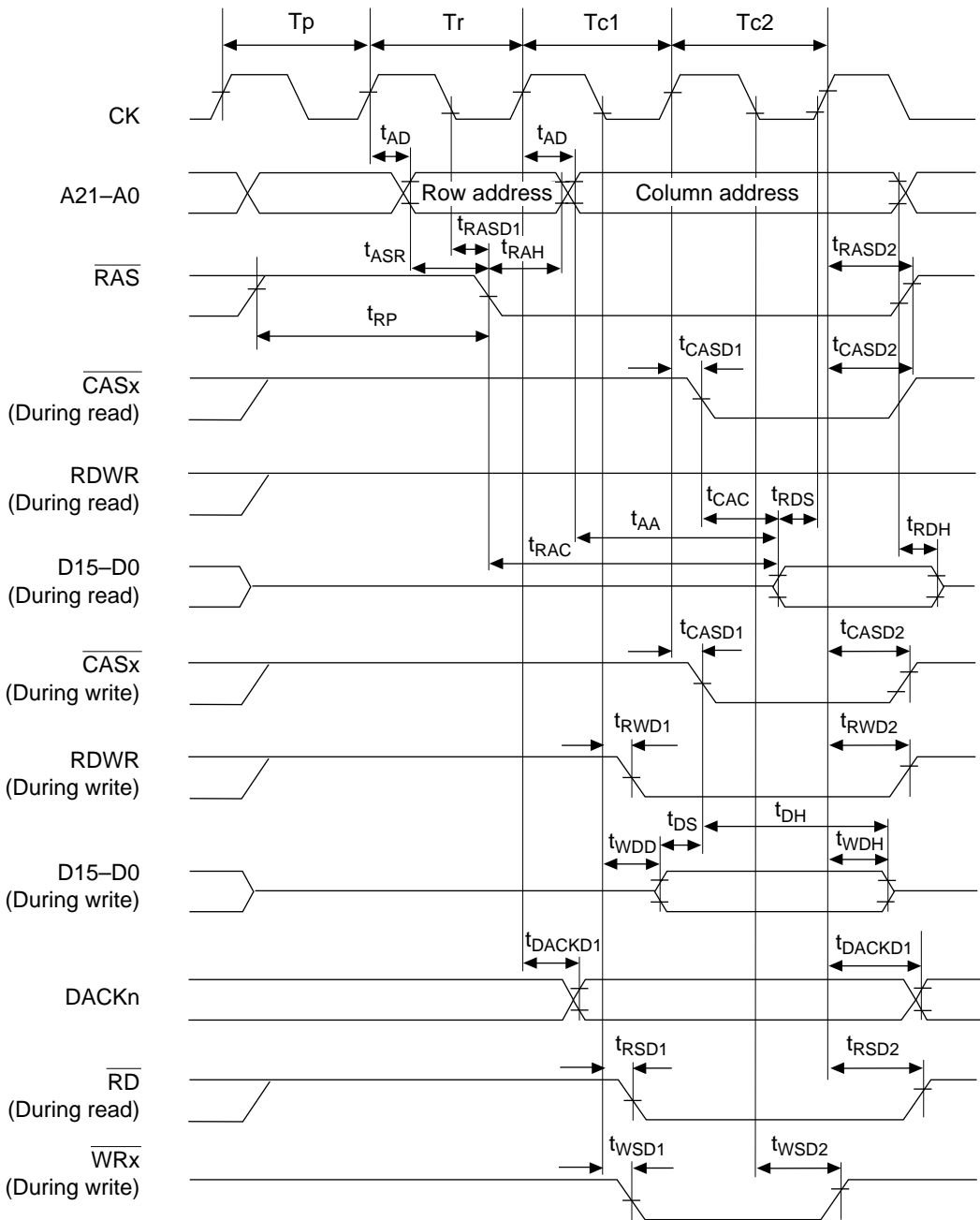


Figure 22.8 Basic Cycle (2 Software Waits + Wait due to \overline{WAIT} Signal)



Note: t_{RDH} is specified from the first negate timing for A21-A0, \overline{RAS} , and \overline{CAS} .

Figure 22.9 DRAM Cycle (Normal Mode, No Waits)

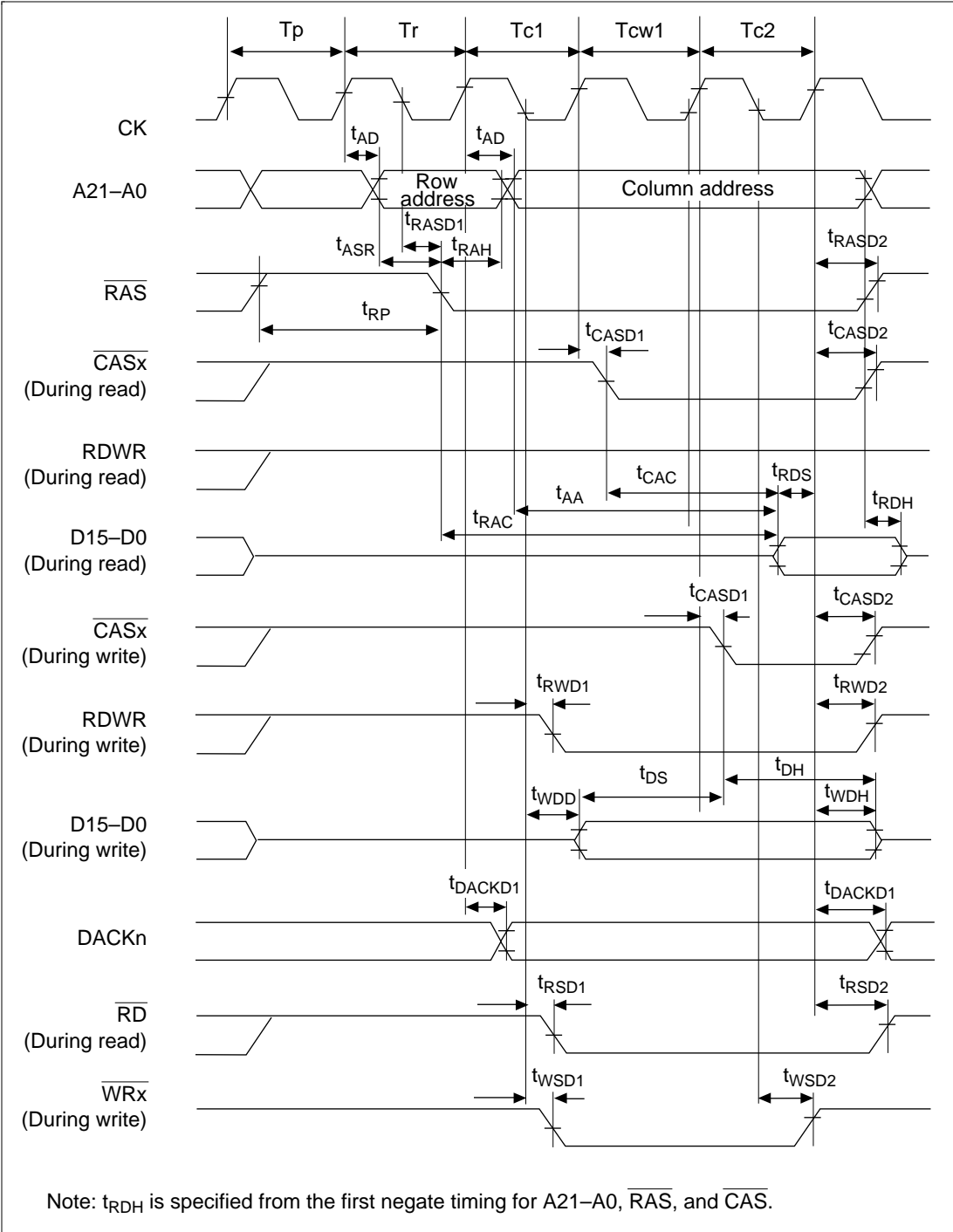
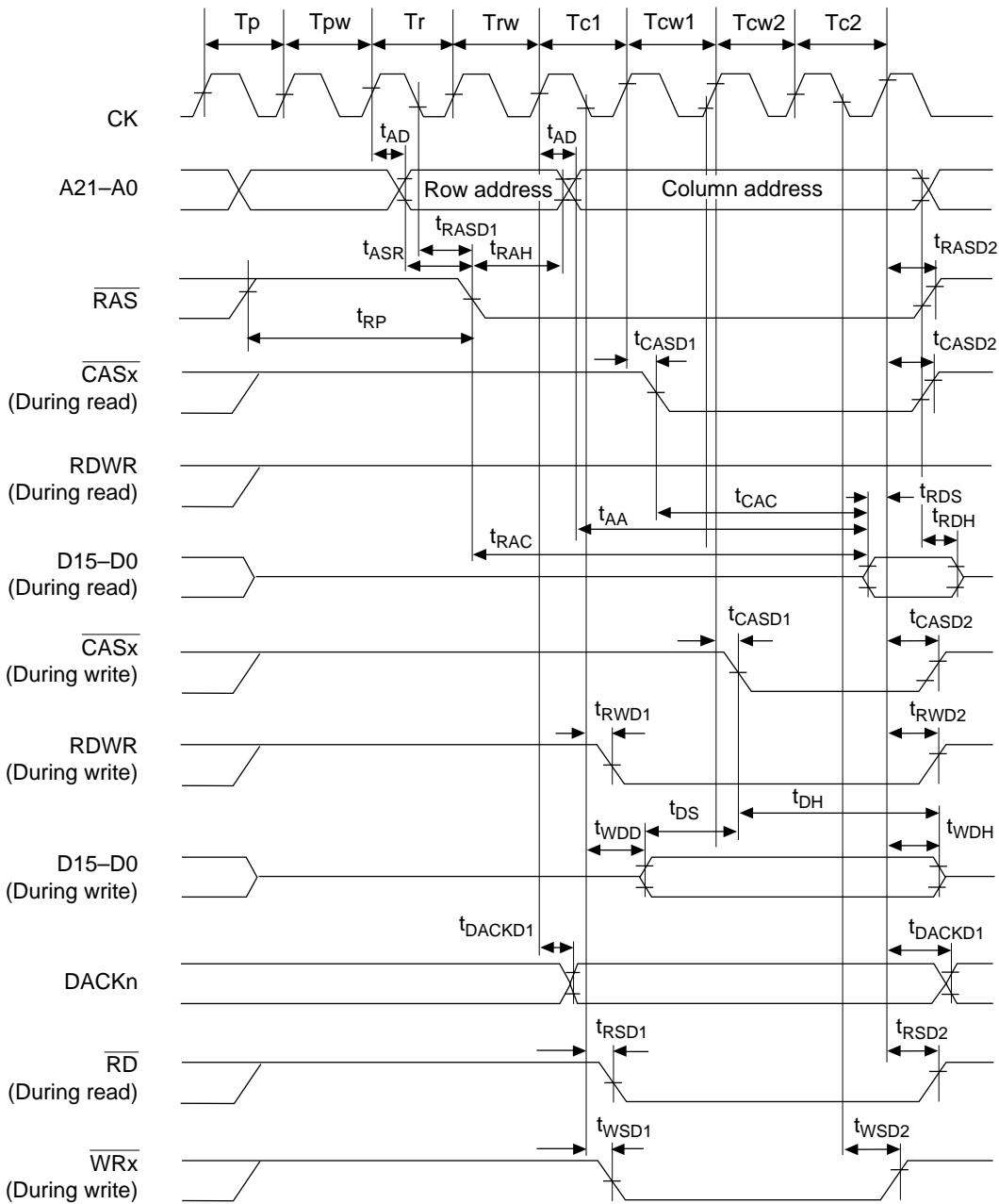
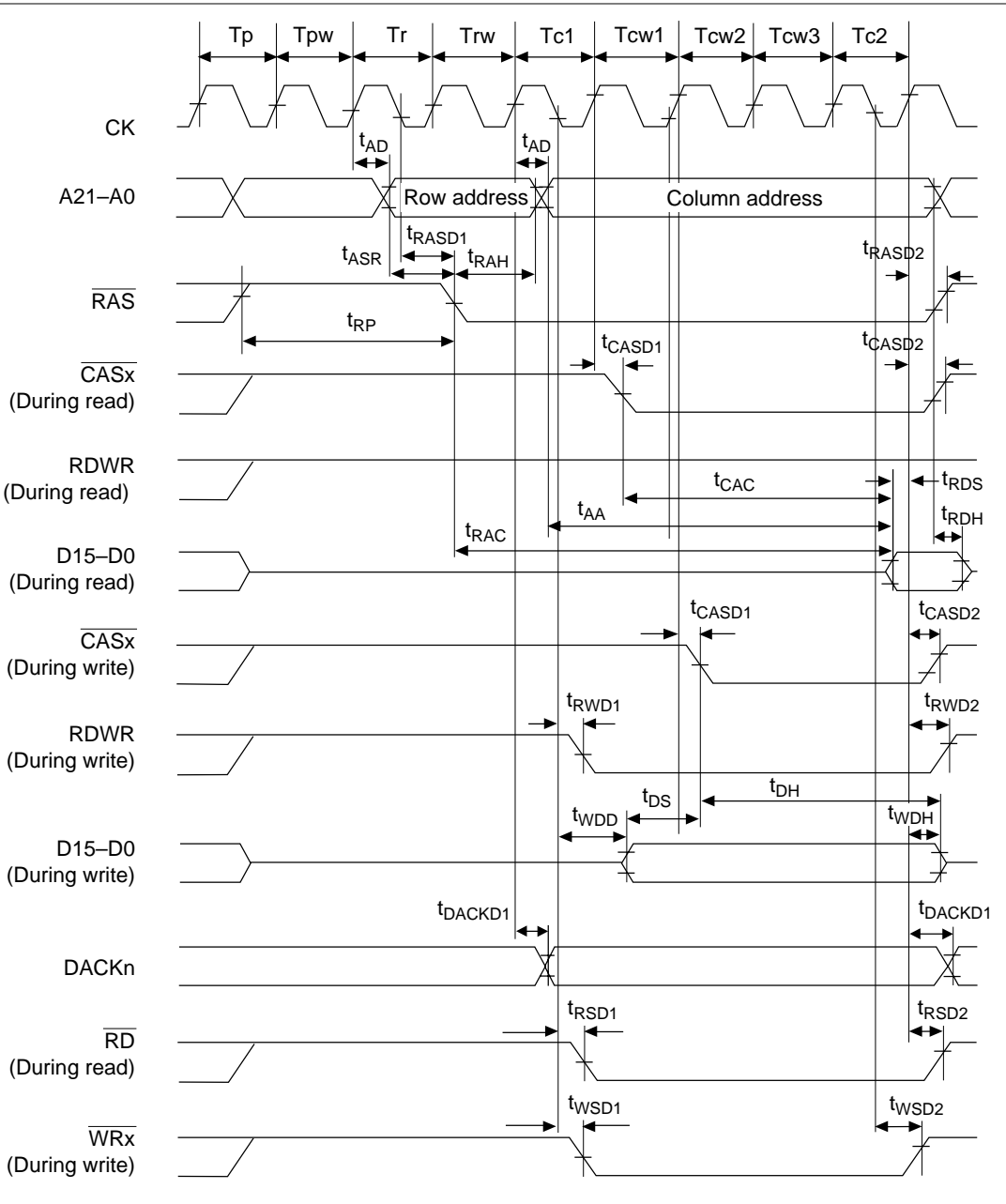


Figure 22.10 DRAM Cycle (Normal Mode, 1 Wait, TPC = 0, RCD = 0)



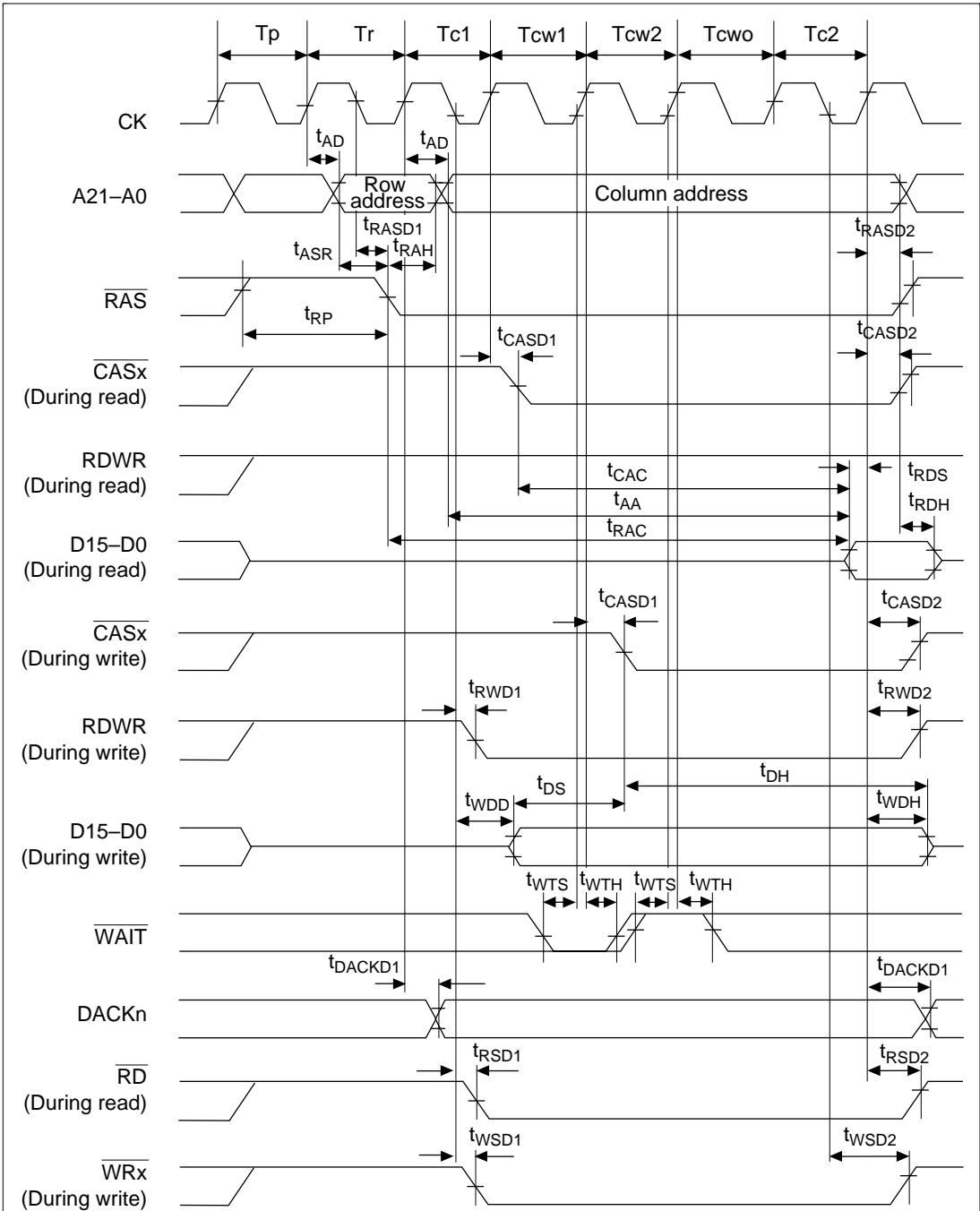
Note: t_{RDH} is specified from the first negate timing for A21-A0, \overline{RAS} , and \overline{CAS} .

Figure 22.11 DRAM Cycle (Normal Mode, 2 Waits, TPC = 1, RCD = 1)



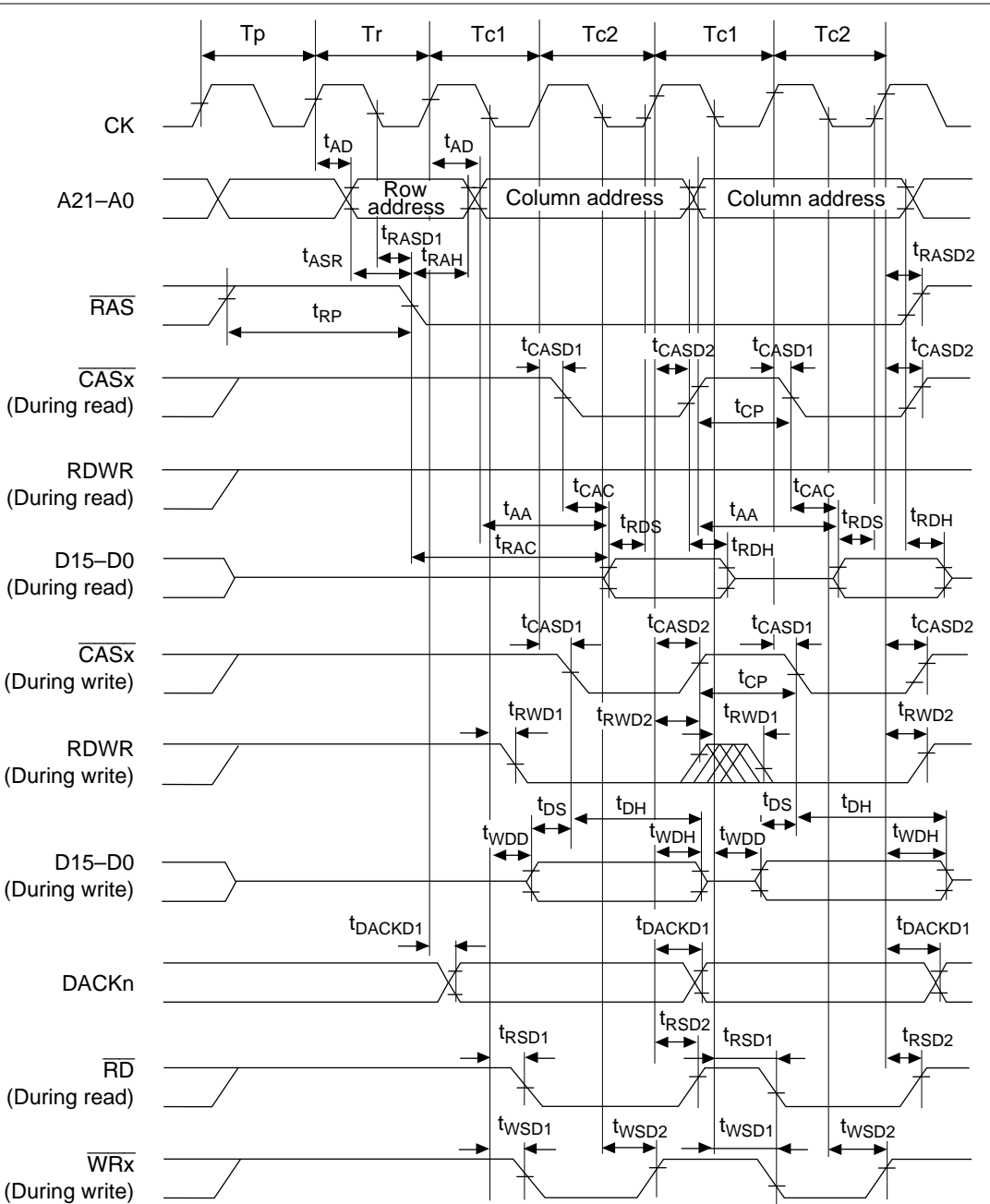
Note: t_{RDH} is specified from the first negate timing for A21-A0, \overline{RAS} , and \overline{CAS} .

Figure 22.12 DRAM Cycle (Normal Mode, 3 Waits, TPC = 1, RCD = 1)



Note: t_{RDH} is specified from the first negate timing for A21-A0, RAS, and CAS.

Figure 22.13 DRAM Cycle (Normal Mode, 2 Waits + Wait due to WAIT Signal)



Note: t_{RDH} is specified from the first negate timing for A21-A0, \overline{RAS} , and \overline{CAS} .

Figure 22.14 DRAM Cycle (High-Speed Page Mode)

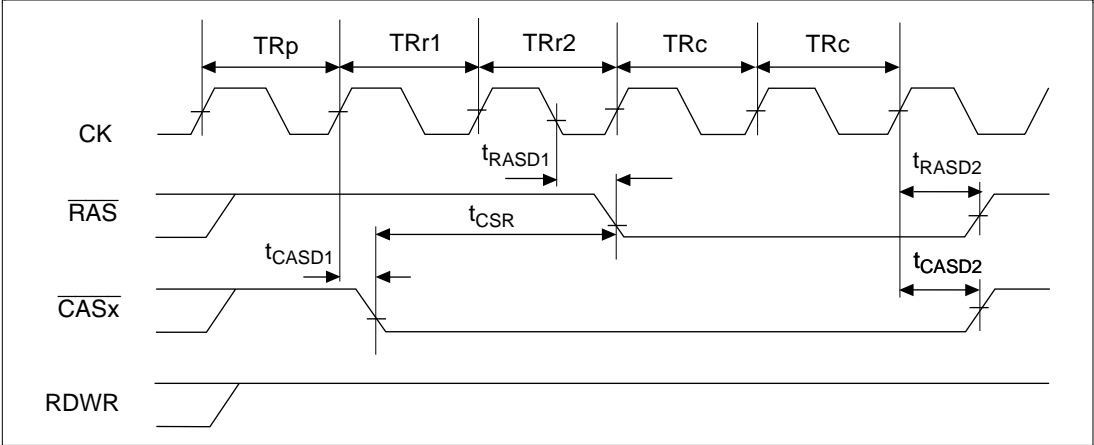


Figure 22.15 CAS Before RAS Refresh (TRAS1 = 0, TRAS0 = 0)

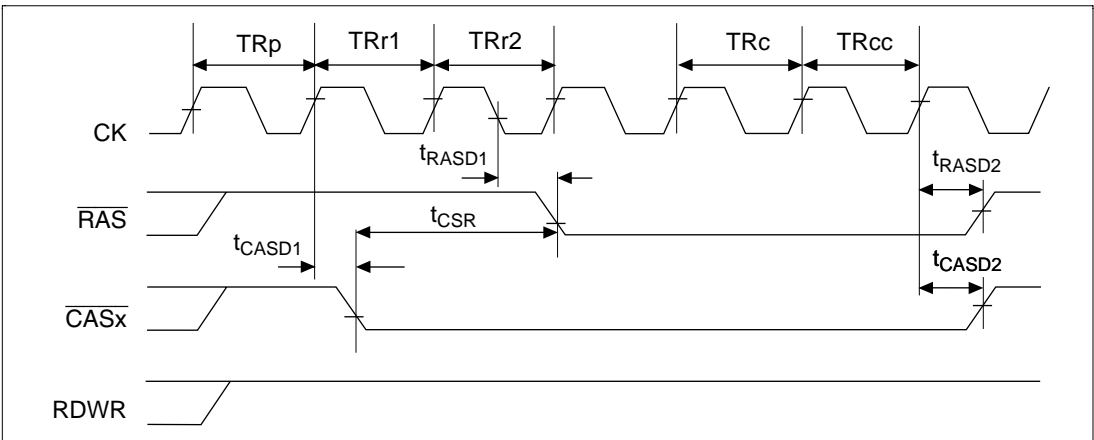
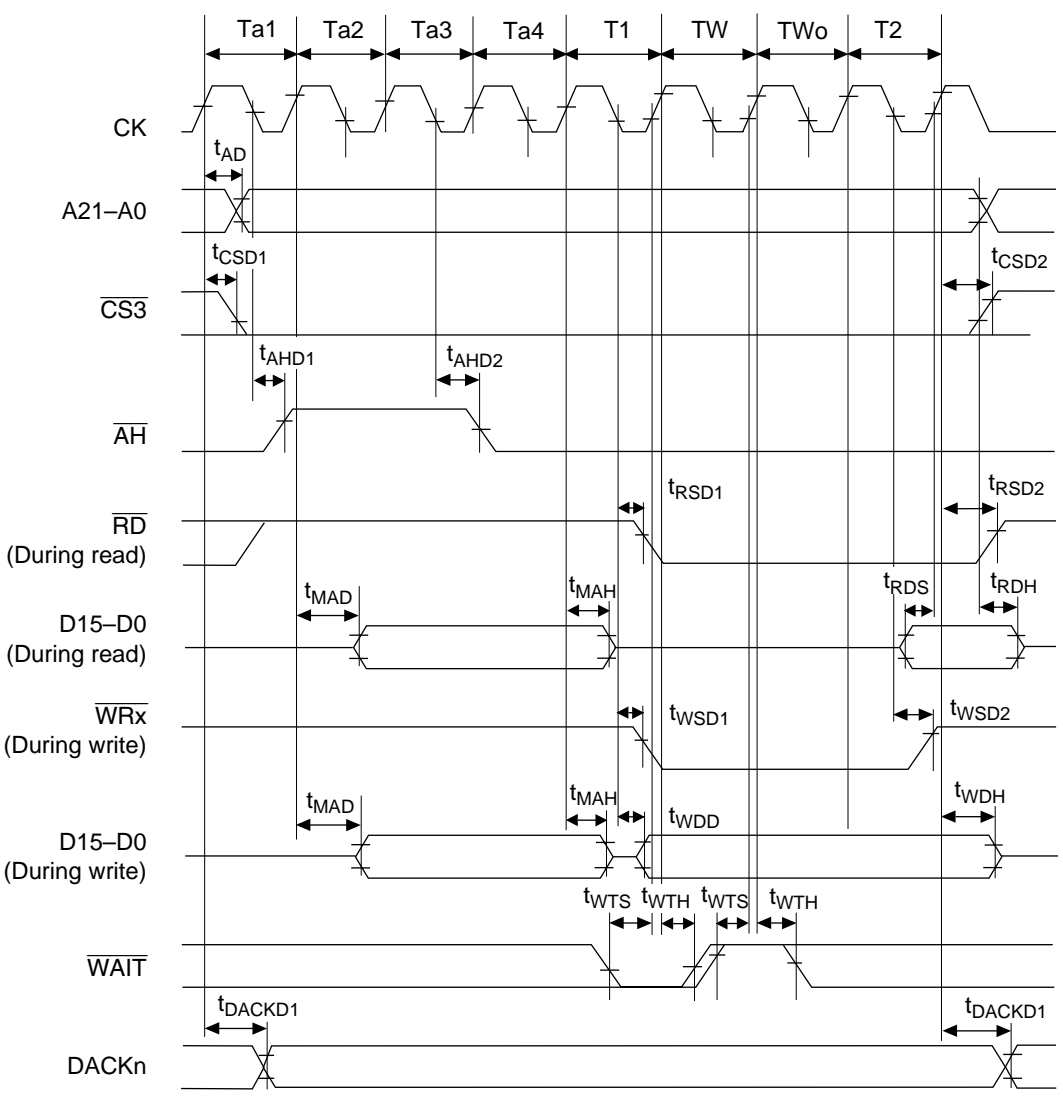


Figure 22.16 Self Refresh



Note: t_{RDH} is specified from the first negate timing for A21-A0, CS3, and RD.

Figure 22.17 Address Data Multiplex I/O Space Cycle (1 Software Wait + 1 External Wait)

22.3.4 Direct Memory Access Controller Timing

Table 22.8 Direct Memory Access Controller Timing (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit | Figure |
|---|--------------------|-----|-----|------------------|--------|
| $\overline{\text{DREQ0}}$ and $\overline{\text{DREQ1}}$ setup time | t_{DRQS} | 18 | — | ns | 22.18 |
| $\overline{\text{DREQ0}}$ and $\overline{\text{DREQ1}}$ hold time | t_{DRQH} | 18 | — | ns | |
| $\overline{\text{DREQ0}}$ and $\overline{\text{DREQ1}}$ pulse width | t_{DRQW} | 1.5 | — | t_{cyc} | 22.19 |
| DRAK output delay time | t_{DRAKD} | 18 | — | ns | 22.20 |

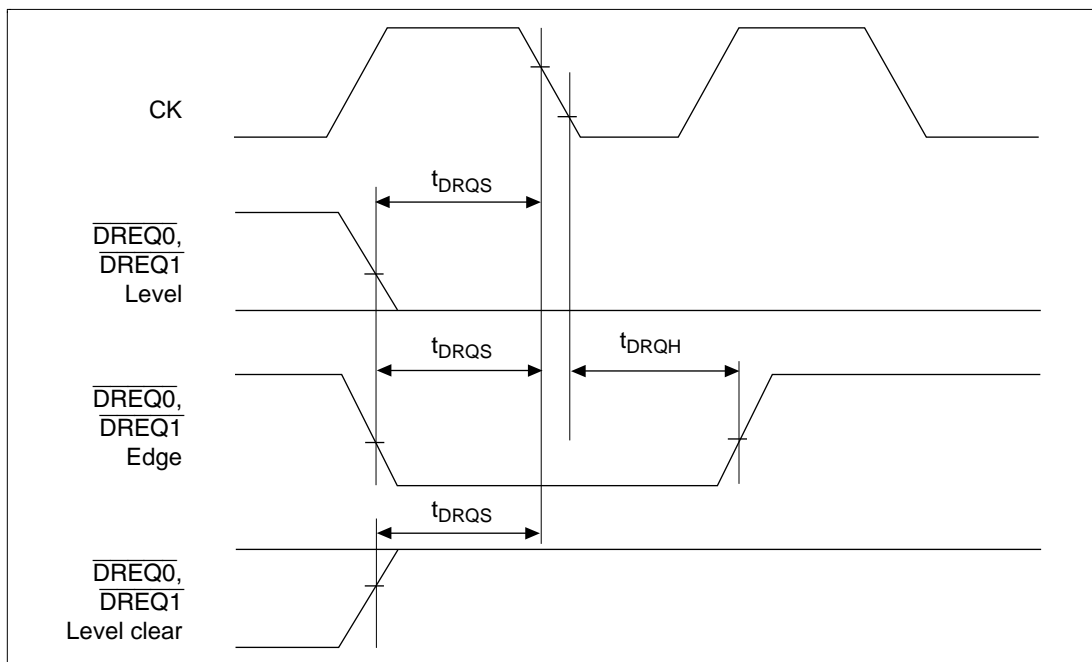


Figure 22.18 $\overline{\text{DREQ0}}$ and $\overline{\text{DREQ1}}$ Input Timing (1)

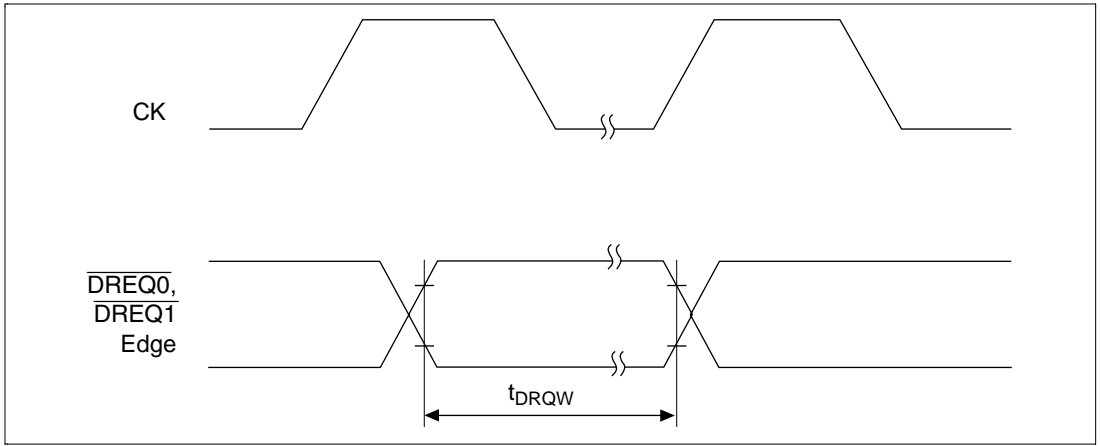


Figure 22.19 DREQ0 and DREQ1 Input Timing (2)

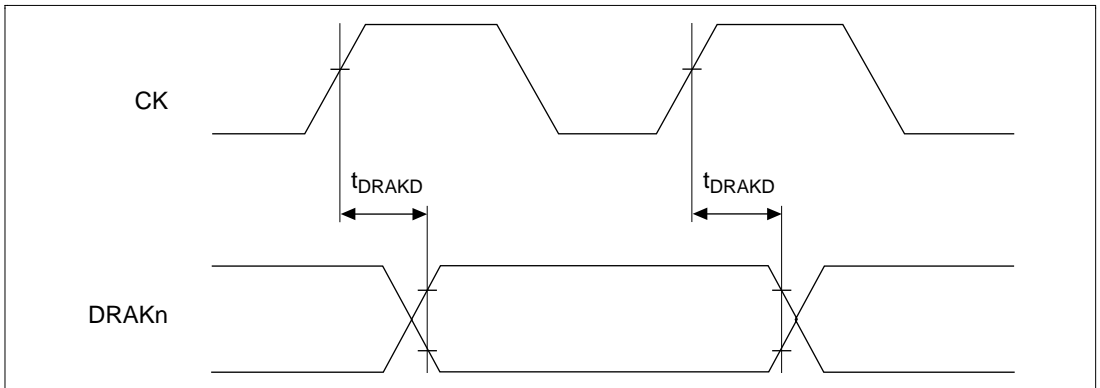


Figure 22.20 DRAK Output Delay Time

22.3.5 Multifunction Timer Pulse Unit Timing

Table 22.9 Multifunction Timer Pulse Unit Timing (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit | Figure |
|---|---------------|-----|-----|-----------|--------|
| Output compare output delay time | t_{TOCD} | — | 100 | ns | 22.21 |
| Input capture input setup time | t_{TICS} | 30 | — | ns | |
| Timer input setup time | t_{TCKS} | 35 | — | ns | |
| Timer clock pulse width (single edge specification) | $t_{TCKWH/L}$ | 1.5 | — | t_{cyc} | 22.22 |
| Timer clock pulse width (both edges specified) | $t_{TCKWH/L}$ | 2.5 | — | t_{cyc} | |
| Timer clock pulse width (phase measurement mode) | $t_{TCKWH/L}$ | 2.5 | — | t_{cyc} | |

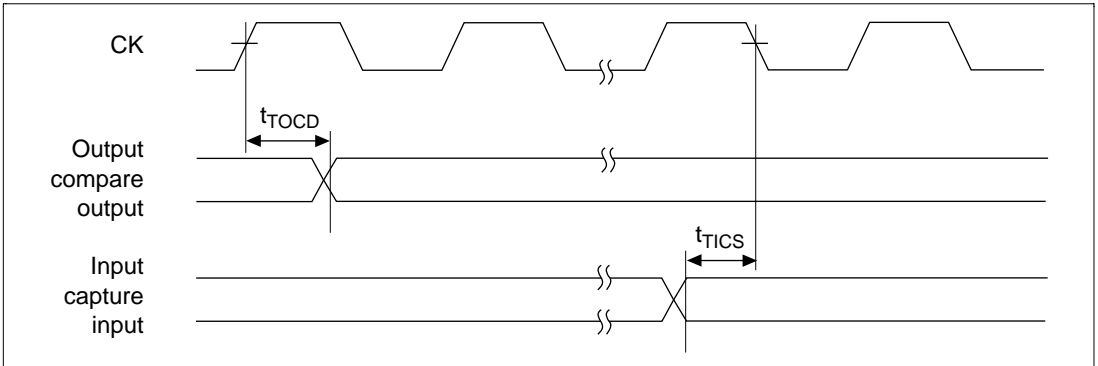


Figure 22.21 MTU I/O Timing

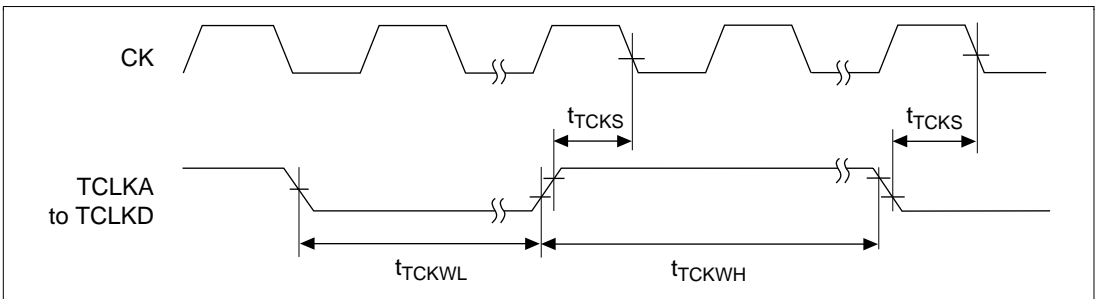


Figure 22.22 MTU Clock Input Timing

22.3.6 I/O Port Timing

Table 22.10 I/O Port Timing (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit | Figure |
|-----------------------------|-----------|-----|-----|------|--------|
| Port output data delay time | t_{PWD} | — | 100 | ns | 22.23 |
| Port input hold time | t_{PRH} | 35 | — | ns | |
| Port input setup time | t_{PRS} | 35 | — | ns | |

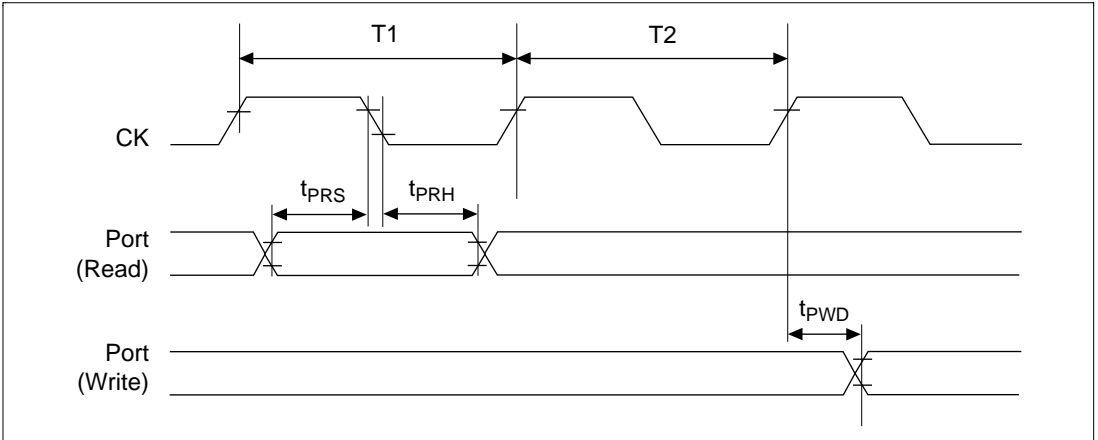


Figure 22.23 I/O Port I/O Timing

22.3.7 Watchdog Timer Timing

Table 22.11 Watchdog Timer Timing (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit | Figure |
|-------------------|------------|-----|-----|------|--------|
| WDTOVF delay time | t_{WOVD} | — | 100 | ns | 22.24 |

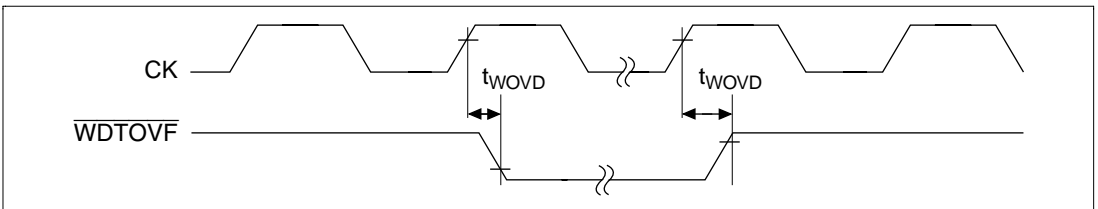


Figure 22.24 Watchdog Timer Timing

22.3.8 Serial Communication Interface Timing

Table 22.12 Serial Communication Interface Timing (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{C}$)

| Item | Symbol | Min | Max | Unit | Figure |
|---------------------------------------|------------|-----|-----|------------|--------|
| Input clock cycle | t_{scyc} | 4 | — | t_{cyc} | 22.25 |
| Input clock cycle (clock sync) | t_{scyc} | 6 | — | t_{cyc} | |
| Input clock pulse width | t_{sckw} | 0.4 | 0.6 | t_{scyc} | |
| Input clock rise time | t_{sckr} | — | 1.5 | t_{cyc} | |
| Input clock fall time | t_{sckf} | — | 1.5 | t_{cyc} | |
| Transmit data delay time (clock sync) | t_{TXD} | — | 100 | ns | 22.26 |
| Receive data setup time (clock sync) | t_{RXS} | 100 | — | ns | |
| Receive data hold time (clock sync) | t_{RXH} | 100 | — | ns | |

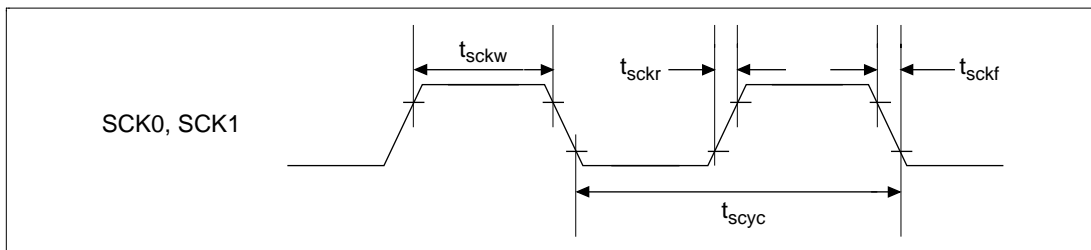


Figure 22.25 Input Clock Timing

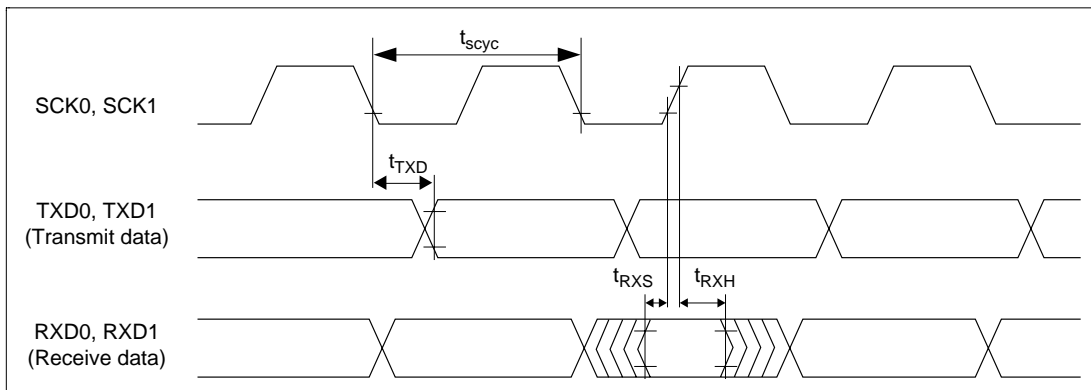


Figure 22.26 SCI I/O Timing (Clock Sync Mode)

22.3.9 High Speed A/D Converter Timing – SH7014–

Table 22.13 A/D Converter Timing (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{C}$)

| Item | | Symbol | Min | Typ | Max | Unit | Figure |
|---------------------------------|---------|-------------------|------|------|------|------------------|--------|
| A/D conversion start delay time | CKS = 0 | t_D | 1.5 | 1.5 | 1.5 | t_{cyc} | 22.27 |
| | CKS = 1 | | 1.5 | 1.5 | 1.5 | | |
| Input sampling time | CKS = 0 | t_{SPL} | 20 | 20 | 20 | | |
| | CKS = 1 | | 40 | 40 | 40 | | |
| A/D conversion time | CKS = 0 | t_{CONV} | 42.5 | 42.5 | 42.5 | | |
| | CKS = 1 | | 82.5 | 82.5 | 82.5 | | |

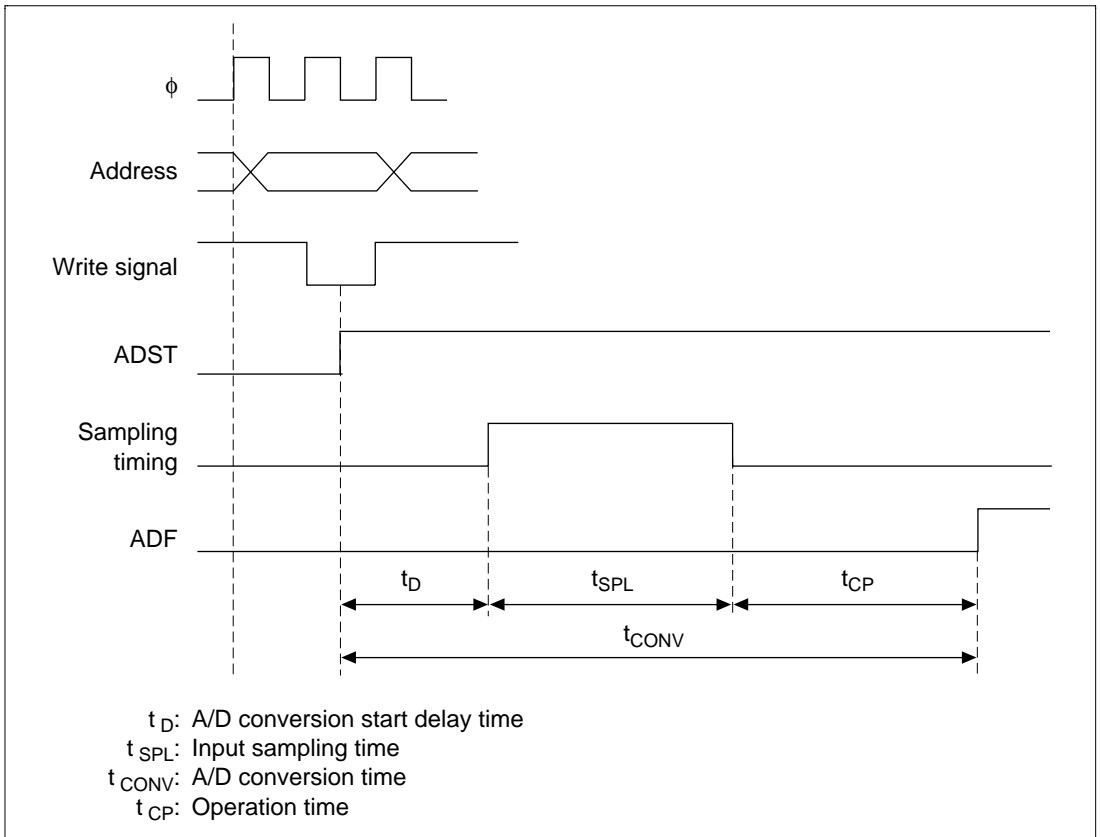


Figure 22.27 Analog Conversion Timing

22.3.10 Mid-speed Converter Timing –SH7016, SH7017–

Table 22.14 Mid-speed A/D Converter Timing (Conditions: $V_{CC} = 5.0V \pm 10\%$, $AV_{CC} = 5.0V \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0V$, $T_a = -20$ to $+75^\circ C$)

| Item | | Symbol | Min | Typ | Max | Unit | Figure |
|---------------------------------|---------|------------|-----|-----|-----|-----------|--------|
| A/D conversion start delay time | CKS = 0 | t_D | 10 | – | 17 | t_{cyc} | 22.28 |
| | CKS = 1 | | 6 | – | 9 | | |
| Input sampling time | CKS = 0 | t_{SPL} | – | 64 | – | | |
| | CKS = 1 | | – | 32 | – | | |
| A/D conversion time | CKS = 0 | t_{CONV} | 259 | – | 266 | | |
| | CKS = 1 | | 131 | – | 134 | | |

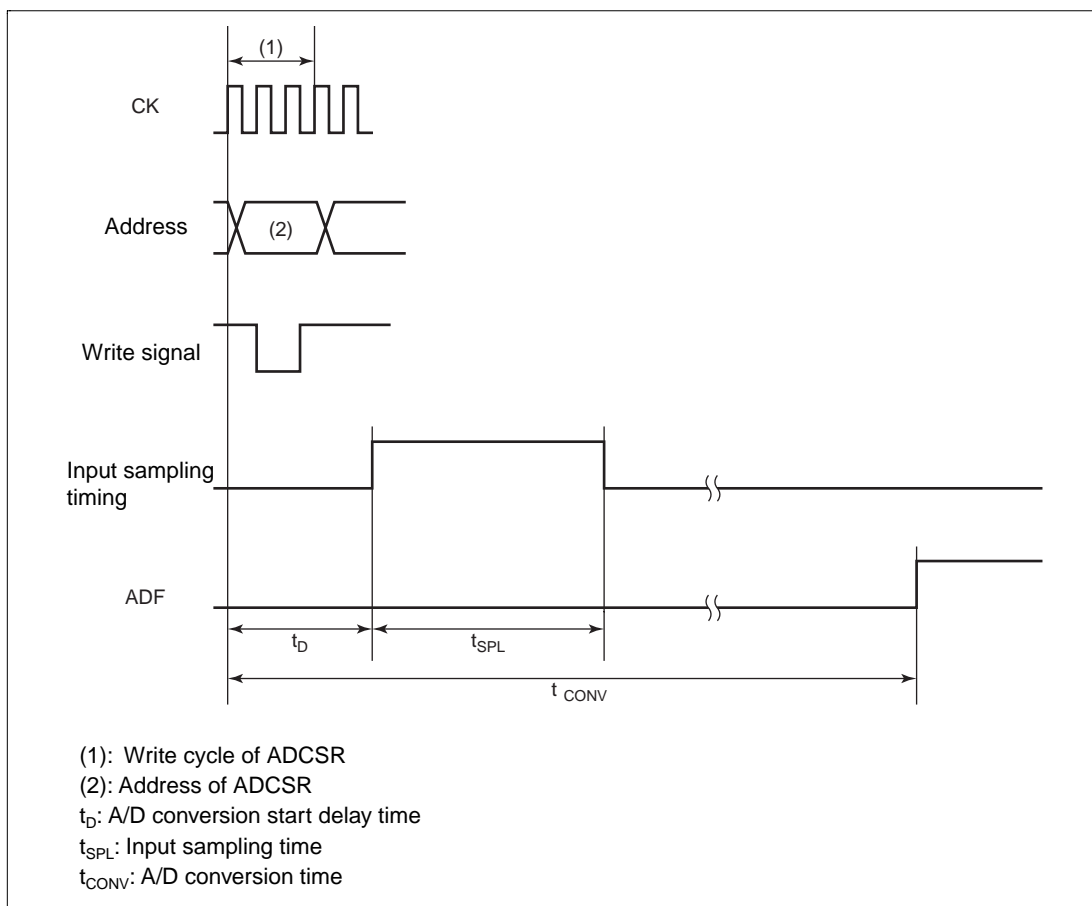
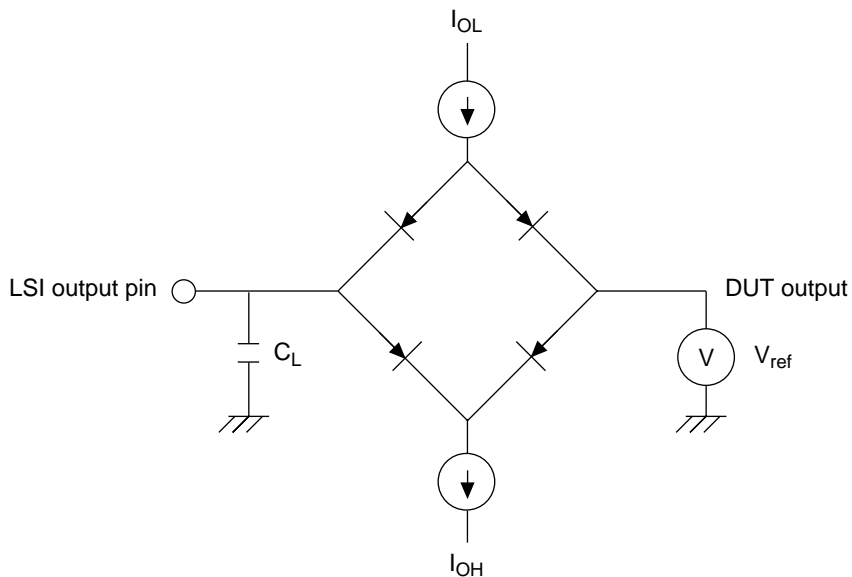


Figure 22.28 Analog Conversion Timing

22.3.11 Measuring Conditions for AC Characteristics

- Input reference levels:
 - High level: 2.2 V
 - Low level: 0.8 V
- Output reference levels:
 - High level: 2.0 V
 - Low level: 0.8 V



Note: C_L is set with the following pins, including the total capacitance of the measurement jig, etc:

30 pF: \overline{CK} , \overline{RAS} , \overline{CASx} , \overline{RDWR} , $\overline{CS0-CS3}$, \overline{AH} , $\overline{DACK0}$, $\overline{DACK1}$

50 pF: $A21-A0$, $D15-D0$, \overline{RD} , \overline{WRx}

70 pF: Port output and peripheral module output pins other than the above.

I_{OL} , I_{OH} : See table 22.3, Permitted Output Current Values.

Figure 22.29 Output Test Circuit

22.4 A/D Converter Characteristics

Table 22.15 A/D Converter Characteristics (HD6417014) (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{C}$)

| Item | 28.7MHz | | | Unit |
|-----------------------------------|---------|-----|-------|------|
| | Min | Typ | Max | |
| Resolution | 10 | 10 | 10 | Bits |
| Conversion time* ¹ | — | — | 2.9 | μs |
| Analog input capacitance | — | — | 20 | pF |
| Permitted signal source impedance | — | — | 1 | kΩ |
| Non-linear error* ² | — | — | ± 8 | LSB |
| Offset error* ² | — | — | ± 8 | LSB |
| Full-scale error* ² | — | — | ± 8 | LSB |
| Quantization error* ² | — | — | ± 0.5 | LSB |
| Absolute error* ¹ | — | — | ± 15 | LSB |

Notes: 1. CKS = 1

2. Reference values

Table 22.16 A/D Converter Characteristics (HD6417014R) (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20\text{ to }+75^\circ\text{C}$)

| Item | 28.7MHz | | | Unit |
|------------------------------------|---------|-----|-------|------|
| | Min | Typ | Max | |
| Resolution | 10 | 10 | 10 | bit |
| Conversion time* ¹ | — | — | 2.9 | μs |
| Analog input capacitance | — | — | 20 | pF |
| Permission signal source impedance | — | — | 1 | kΩ |
| Non-linear error* ² | — | — | ± 8 | LSB |
| Offset error* ² | — | — | ± 8 | LSB |
| Full scale error* ² | — | — | ± 8 | LSB |
| Quantize error* ² | — | — | ± 0.5 | LSB |
| Absolute error* ¹ | — | — | ± 8 | LSB |

Notes: 1. CKS = 1

2. Reference values

Table 22.17 A/D Converter Characteristics (SH7016, SH7017) (Conditions: $V_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = 5.0\text{ V} \pm 10\%$, $AV_{CC} = V_{CC} \pm 10\%$, $V_{SS} = AV_{SS} = 0\text{ V}$, $T_a = -20$ to $+75^\circ\text{C}$)

| Item | 28.7MHz | | | 20MHz | | | Unit |
|------------------------------------|---------|-----|-----------|-------|-----|--------------------|---------------|
| | min | typ | max | min | typ | max | |
| Resolution | 10 | 10 | 10 | 10 | 10 | 10 | bit |
| Conversion time (when CKS = 0) | — | — | 9.3 | — | — | 13.4* ² | μs |
| Analog input capacity | — | — | 20 | — | — | 20 | pF |
| Permission signal source impedance | — | — | 1 | — | — | 1 | k Ω |
| Non-linearity error* ¹ | — | — | ± 3 | — | — | ± 3 | LSB |
| Offset error* ¹ | — | — | ± 3 | — | — | ± 3 | LSB |
| Full scale error* ¹ | — | — | ± 3 | — | — | ± 3 | LSB |
| Quantize error* ¹ | — | — | ± 0.5 | — | — | ± 0.5 | LSB |
| Absolute error | — | — | ± 4 | — | — | ± 4 | LSB |

Notes: 1. Reference value
 2. 6.7 μs when CKS = 1.

Appendix A On-Chip Supporting Module Registers

A.1 Addresses

Table A.1 On-Chip I/O Register Addresses

| Address | Register Abbr. | Bit Names | | | | | | | | Module |
|--------------------------------|-------------------|-------------|-------|-------|-------------|-------|-------|-------|-------|--------|
| | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| H'FFFF81A0 | SMR0 | C \bar{A} | CHR | PE | O \bar{E} | STOP | MP | CKS1 | CKS0 | SCI |
| H'FFFF81A1 | BRR0 | | | | | | | | | |
| H'FFFF81A2 | SCR0 | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 | |
| H'FFFF81A3 | TDR0 | | | | | | | | | |
| H'FFFF81A4 | SSR0 | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT | |
| H'FFFF81A5 | RDR0 | | | | | | | | | |
| H'FFFF81A6 to H'FFFF81AF | — | — | — | — | — | — | — | — | — | |
| H'FFFF81B0 | SMR1 | C \bar{A} | CHR | PE | O \bar{E} | STOP | MP | CKS1 | CKS0 | |
| H'FFFF81B1 | BRR1 | | | | | | | | | |
| H'FFFF81B2 | SCR1 | TIE | RIE | TE | RE | MPIE | TEIE | CKE1 | CKE0 | |
| H'FFFF81B3 | TDR1 | | | | | | | | | |
| H'FFFF81B4 | SSR1 | TDRE | RDRF | ORER | FER | PER | TEND | MPB | MPBT | |
| H'FFFF81B5 | RDR1 | | | | | | | | | |
| H'FFFF81B6 to H'FFFF81FF | — | — | — | — | — | — | — | — | — | |
| H'FFFF8200 to H'FFFF823F | — | — | — | — | — | — | — | — | — | MTU |
| H'FFFF8240 | TSTR | — | — | — | — | — | CST2 | CST1 | CST0 | |
| H'FFFF8241 | TSYR | — | — | — | — | — | SYNC2 | SYNC1 | SYNC0 | |
| H'FFFF8242 to H'FFFF825F | — | — | — | — | — | — | — | — | — | |
| H'FFFF8260 | TCR0 | CCLR2 | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | |
| H'FFFF8261 | TMDR0 | — | — | BFB | BFA | MD3 | MD2 | MD1 | MD0 | |
| H'FFFF8262 | TIOR0H | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | |
| H'FFFF8263 | TIOR0L | IOD3 | IOD2 | IOD1 | IOD0 | IOC3 | IOC2 | IOC1 | IOC0 | |
| H'FFFF8264 | TIER0 | TTGE | — | — | TCIEV | TGIED | TGIEC | TGIEB | TGIEA | |
| H'FFFF8265 | TSR0 | — | — | — | TCFV | TGFD | TGFC | TGFB | TGFA | |
| H'FFFF8266 | TCNT0 | | | | | | | | | |
| H'FFFF8267 | | | | | | | | | | |

Table A.1 On-Chip I/O Register Addresses (cont)

| Address | Register Abbr. | Bit Names | | | | | | | | Module |
|--------------------------------|----------------|-----------|-------|-------|-------|-------|-------|-------|-------|--------|
| | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| H'FFFF8268 | TGR0A | | | | | | | | | MTU |
| H'FFFF8269 | | | | | | | | | | |
| H'FFFF826A | TGR0B | | | | | | | | | |
| H'FFFF826B | | | | | | | | | | |
| H'FFFF826C | TGR0C | | | | | | | | | |
| H'FFFF826D | | | | | | | | | | |
| H'FFFF826E | TGR0D | | | | | | | | | |
| H'FFFF826F | | | | | | | | | | |
| H'FFFF8270 to H'FFFF827F | — | — | — | — | — | — | — | — | — | |
| H'FFFF8280 | TCR1 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | |
| H'FFFF8281 | TMDR1 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | |
| H'FFFF8282 | TIOR1 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | |
| H'FFFF8283 | — | — | — | — | — | — | — | — | — | |
| H'FFFF8284 | TIER1 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | |
| H'FFFF8285 | TSR1 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | |
| H'FFFF8286 | TCNT1 | | | | | | | | | |
| H'FFFF8287 | | | | | | | | | | |
| H'FFFF8288 | TGR1A | | | | | | | | | |
| H'FFFF8289 | | | | | | | | | | |
| H'FFFF828A | TGR1B | | | | | | | | | |
| H'FFFF828B | | | | | | | | | | |
| H'FFFF828C to H'FFFF829F | — | — | — | — | — | — | — | — | — | |
| H'FFFF82A0 | TCR2 | — | CCLR1 | CCLR0 | CKEG1 | CKEG0 | TPSC2 | TPSC1 | TPSC0 | |
| H'FFFF82A1 | TMDR2 | — | — | — | — | MD3 | MD2 | MD1 | MD0 | |
| H'FFFF82A2 | TIOR2 | IOB3 | IOB2 | IOB1 | IOB0 | IOA3 | IOA2 | IOA1 | IOA0 | |
| H'FFFF82A3 | — | — | — | — | — | — | — | — | — | |
| H'FFFF82A4 | TIER2 | TTGE | — | TCIEU | TCIEV | — | — | TGIEB | TGIEA | |
| H'FFFF82A5 | TSR2 | TCFD | — | TCFU | TCFV | — | — | TGFB | TGFA | |
| H'FFFF82A6 | TCNT2 | | | | | | | | | |
| H'FFFF82A7 | | | | | | | | | | |
| H'FFFF82A8 | TGR2A | | | | | | | | | |
| H'FFFF82A9 | | | | | | | | | | |
| H'FFFF82AA | TGR2B | | | | | | | | | |
| H'FFFF82AB | | | | | | | | | | |

Table A.1 On-Chip I/O Register Addresses (cont)

| Address | Register Abbr. | Bit Names | | | | | | | | Module |
|--------------------------------|----------------|-----------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|----------------------|----------------------|--------|
| | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| H'FFFF82AC to H'FFFF8347 | — | — | — | — | — | — | — | — | — | MTU |
| H'FFFF8348 H'FFFF8349 | IPRA | | | | | | | | | INTC |
| H'FFFF834A H'FFFF834B | IPRB | | | | | | | | | |
| H'FFFF834C H'FFFF834D | IPRC | | | | | | | | | |
| H'FFFF834E H'FFFF834F | IPRD | | | | | | | | | |
| H'FFFF8350 H'FFFF8351 | IPRE | | | | | | | | | |
| H'FFFF8352 H'FFFF8353 | IPRF | | | | | | | | | |
| H'FFFF8354 H'FFFF8355 | IPRG | | | | | | | | | |
| H'FFFF8356 H'FFFF8357 | IPRH | | | | | | | | | |
| H'FFFF8358 | ICR | NMIL | — | — | — | — | — | — | NMIE | |
| H'FFFF8359 | | IRQ0S | IRQ1S | IRQ2S | IRQ3S | — | — | IRQ6S | IRQ7S | |
| H'FFFF835A H'FFFF835B | ISR | — | — | — | — | — | — | — | — | |
| H'FFFF835C to H'FFFF8381 | — | IRQ0F | IRQ1F | IRQ2F | IRQ3F | — | — | IRQ6F | IRQ7F | |
| H'FFFF8382 H'FFFF8383 | PADRL | PA15DR | PA14DR* ¹ | PA13DR* ¹ | PA12DR* ¹ | PA11DR* ¹ | PA10DR* ¹ | PA9DR | PA8DR | I/O |
| H'FFFF8384 H'FFFF8385 | — | PA7DR | PA6DR | PA5DR | PA4DR | PA3DR | PA2DR | PA1DR | PA0DR | |
| H'FFFF8386 H'FFFF8387 | PAIORL | PA15IOR | PA14IOR* ¹ | PA13IOR* ¹ | PA12IOR* ¹ | PA11IOR* ¹ | PA10IOR* ¹ | PA9IOR* ¹ | PA8IOR* ¹ | PFC |
| H'FFFF8388 H'FFFF8389 | — | PA7IOR | PA6IOR | PA5IOR | PA4IOR | PA3IOR | PA2IOR | PA1IOR | PA0IOR | |

Note: 1.Reserved bit in the SH7014.

Table A.1 On-Chip I/O Register Addresses (cont)

| Address | Register Abbr. | Bit Names | | | | | | | | Module |
|------------|----------------------|-----------|--------------|---------|--------------|---------|---------------------|----------------------|----------------------|--------|
| | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| H'FFFF838A | — | — | — | — | — | — | — | — | — | PFC |
| H'FFFF838B | — | — | — | — | — | — | — | — | — | |
| H'FFFF838C | PACRL1 | — | PA15MD | — | PA14MD *1 | — | PA13MD *1 | — | PA12MD* 1 | |
| H'FFFF838D | — | — | PA11MD *1 | — | PA10MD *1 | PA9MD1 | PA9MD0 | PA8MD1 | PA8MD0 | |
| H'FFFF838E | PACRL2 | PA7MD1 | PA7MD0 | PA6MD1 | PA6MD0 | PA5MD1 | PA5MD0 | — | PA4MD | |
| H'FFFF838F | — | — | PA3MD | PA2MD1 | PA2MD0 | — | PA1MD | — | PA0MD | |
| H'FFFF8390 | PBDR | — | — | — | — | — | — | PB9DR | PB8DR | I/O |
| H'FFFF8391 | — | PB7DR | PB6DR | PB5DR | PB4DR | PB3DR | PB2DR | PB1DR* ¹ | PB0DR* ¹ | |
| H'FFFF8392 | PCDR* ² | PC15DR | PC14DR | PC13DR | PC12DR | PC11DR | PC10DR | PC9DR | PC8DR | |
| H'FFFF8393 | — | PC7DR | PC6DR | PC5DR | PC4DR | PC3DR | PC2DR | PC1DR | PC0DR | |
| H'FFFF8394 | PBIOR | — | — | — | — | — | — | PB9IOR | PB8IOR | PFC |
| H'FFFF8395 | — | PB7IOR | PB6IOR | PB5IOR | PB4IOR | PB3IOR | PB2IOR | PB1IOR* ¹ | PB0IOR* ¹ | |
| H'FFFF8396 | PCIOR* ² | PC15IOR | PC14IOR | PC13IOR | PC12IOR | PC11IOR | PC10IOR | PC9IOR | PC8IOR | |
| H'FFFF8397 | — | PC7IOR | PC6IOR | PC5IOR | PC4IOR | PC3IOR | PC2IOR | PC1IOR | PC0IOR | |
| H'FFFF8398 | PBCR1 | — | — | — | — | — | — | — | — | |
| H'FFFF8399 | — | — | — | — | — | PB9MD1 | PB9MD0 | PB8MD1 | PB8MD0 | |
| H'FFFF839A | PBCR2 | PB7MD1 | PB7MD0 | PB6MD1 | PB6MD0 | PB5MD1 | PB5MD0 | PB4MD1 | PB4MD0 | |
| H'FFFF839B | — | PB3MD1 | PB3MD0 | PB2MD1 | PB2MD0 | — | PB1MD* ¹ | — | PB0MD* ¹ | |
| H'FFFF839C | PCCR* ² | PC15MD | PC14MD | PC13MD | PC12MD | PC11MD | PC10MD | PC9MD | PC8MD | |
| H'FFFF839D | — | PC7MD | PC6MD | PC5MD | PC4MD | PC3MD | PC2MD | PC1MD | PC0MD | |
| H'FFFF839E | — | — | — | — | — | — | — | — | — | |
| H'FFFF839F | — | — | — | — | — | — | — | — | — | |
| H'FFFF83A0 | — | — | — | — | — | — | — | — | — | I/O |
| H'FFFF83A1 | — | — | — | — | — | — | — | — | — | |
| H'FFFF83A2 | PDDR1* ² | PD15DR | PD14DR | PD13DR | PD12DR | PD11DR | PD10DR | PD9DR | PD8DR | |
| H'FFFF83A3 | — | PD7DR | PD6DR | PD5DR | PD4DR | PD3DR | PD2DR | PD1DR | PD0DR | |
| H'FFFF83A4 | — | — | — | — | — | — | — | — | — | PFC |
| H'FFFF83A5 | — | — | — | — | — | — | — | — | — | |
| H'FFFF83A6 | PDIOR1* ² | PD15IOR | PD14IOR | PD13IOR | PD12IOR | PD11IOR | PD10IOR | PD9IOR | PD8IOR | |
| H'FFFF83A7 | — | PD7IOR | PD6IOR | PD5IOR | PD4IOR | PD3IOR | PD2IOR | PD1IOR | PD0IOR | |
| H'FFFF83A8 | — | — | — | — | — | — | — | — | — | |
| H'FFFF83A9 | — | — | — | — | — | — | — | — | — | |

Note: 1. Reserved bit in the SH7014.

2. In the SH7014, this address is reserved and must not be accessed.

Table A.1 On-Chip I/O Register Addresses (cont)

| Address | Register Abbr. | Bit Names | | | | | | | | Module |
|------------|-------------------|-----------|---------|---------|---------|---------|---------|--------|--------|--------|
| | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| H'FFFF83AA | — | — | — | — | — | — | — | — | — | PFC |
| H'FFFF83AB | — | — | — | — | — | — | — | — | — | |
| H'FFFF83AC | PDCRL*2 | PD15MD | PD14MD | PD13MD | PD12MD | PD11MD | PD10MD | PD9MD | PD8MD | |
| H'FFFF83AD | | PD7MD | PD6MD | PD5MD | PD4MD | PD3MD | PD2MD | PD1MD | PD0MD | |
| H'FFFF83AE | — | — | — | — | — | — | — | — | — | |
| H'FFFF83AF | — | — | — | — | — | — | — | — | — | |
| H'FFFF83B0 | PEDR | PE15DR | PE14DR | PE13DR | PE12DR | PE11DR | PE10DR | PE9DR | PE8DR | I/O |
| H'FFFF83B1 | | PE7DR | PE6DR | PE5DR | PE4DR | PE3DR | PE2DR | PE1DR | PE0DR | |
| H'FFFF83B2 | PFDR | — | — | — | — | — | — | — | — | |
| H'FFFF83B3 | | PF7DR | PF6DR | PF5DR | PF4DR | PF3DR | PF2DR | PF1DR | PF0DR | |
| H'FFFF83B4 | PEIOR | PE15IOR | PE14IOR | PE13IOR | PE12IOR | PE11IOR | PE10IOR | PE9IOR | PE8IOR | PFC |
| H'FFFF83B5 | | PE7IOR | PE6IOR | PE5IOR | PE4IOR | PE3IOR | PE2IOR | PE1IOR | PE0IOR | |
| H'FFFF83B6 | — | — | — | — | — | — | — | — | — | |
| H'FFFF83B7 | — | — | — | — | — | — | — | — | — | |
| H'FFFF83B8 | PECR1 | PE15MD1 | PE15MD0 | PE14MD1 | PE14MD0 | — | — | — | — | |
| H'FFFF83B9 | | — | — | — | — | — | — | — | — | |
| H'FFFF83BA | PECR2 | — | PE7MD | — | PE6MD | — | PE5MD | — | PE4MD | |
| H'FFFF83BB | | PE3MD1 | PE3MD0 | PE2MD1 | PE2MD0 | PE1MD1 | PE1MD0 | PE0MD1 | PE0MD0 | |
| H'FFFF83BC | — | — | — | — | — | — | — | — | — | — |
| H'FFFF83CF | | | | | | | | | | |
| H'FFFF83D0 | CMSTR | — | — | — | — | — | — | — | — | CMT |
| H'FFFF83D1 | | — | — | — | — | — | — | STR1 | STR0 | |
| H'FFFF83D2 | CMCSR0 | — | — | — | — | — | — | — | — | |
| H'FFFF83D3 | | CMF | CMIE | — | — | — | — | CKS1 | CKS0 | |
| H'FFFF83D4 | CMCNT0 | | | | | | | | | |
| H'FFFF83D5 | | | | | | | | | | |
| H'FFFF83D6 | CMCOR0 | | | | | | | | | |
| H'FFFF83D7 | | | | | | | | | | |
| H'FFFF83D8 | CMCSR1 | — | — | — | — | — | — | — | — | |
| H'FFFF83D9 | | CMF | CMIE | — | — | — | — | CKS1 | CKS0 | |

Note: 2. In the SH7014, this address is reserved and must not be accessed.

Table A.1 On-Chip I/O Register Addresses (cont)

| Address | Register Abbr. | Bit Names | | | | | | | | Module |
|--------------------------|----------------|-----------|-------|-------|-------|-------|-------|-------|-------|-----------------------|
| | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| H'FFFF83DA | CMCNT1 | | | | | | | | | CMT |
| H'FFFF83DB | | | | | | | | | | |
| H'FFFF83DC | CMCOR1 | | | | | | | | | |
| H'FFFF83DD | | | | | | | | | | |
| H'FFFF83DE | — | — | — | — | — | — | — | — | — | |
| H'FFFF83DF | — | — | — | — | — | — | — | — | — | |
| H'FFFF83E0 | ADCSR | ADF | ADIE | ADST | CKS | GRP | CH2 | CH1 | CH0 | A/D |
| H'FFFF83E1 | ADCR | — | PWR | TRGS1 | TRGS0 | SCAN | DSMP | BUFE1 | BUFE0 | (High speed) (SH7014) |
| H'FFFF83E2 to H'FFFF83EF | — | — | — | — | — | — | — | — | — | |
| H'FFFF83F0 | ADDRA | — | — | — | — | — | — | AD9 | AD8 | |
| H'FFFF83F1 | | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | |
| H'FFFF83F2 | ADDRB | — | — | — | — | — | — | AD9 | AD8 | |
| H'FFFF83F3 | | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | |
| H'FFFF83F4 | ADDRC | — | — | — | — | — | — | AD9 | AD8 | |
| H'FFFF83F5 | | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | |
| H'FFFF83F6 | ADDRD | — | — | — | — | — | — | AD9 | AD8 | |
| H'FFFF83F7 | | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | |
| H'FFFF83F8 | ADDRE | — | — | — | — | — | — | AD9 | AD8 | |
| H'FFFF83F9 | | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | |
| H'FFFF83FA | ADDRF | — | — | — | — | — | — | AD9 | AD8 | |
| H'FFFF83FB | | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | |
| H'FFFF83FC | ADDRG | — | — | — | — | — | — | AD9 | AD8 | |
| H'FFFF83FD | | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | |
| H'FFFF83FE | ADDRH | — | — | — | — | — | — | AD9 | AD8 | |
| H'FFFF83FF | | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | AD1 | AD0 | |
| H'FFFF8400 to H'FFFF841F | — | — | — | — | — | — | — | — | — | |
| H'FFFF8420 | ADDRA | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | A/D(Mid-speed) |
| H'FFFF8421 | | AD1 | AD0 | — | — | — | — | — | — | |
| H'FFFF8422 | ADDRB | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | (SH7016 /17) |
| H'FFFF8423 | | AD1 | AD0 | — | — | — | — | — | — | |
| H'FFFF8424 | ADDRC | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | |
| H'FFFF8425 | | AD1 | AD0 | — | — | — | — | — | — | |

Table A.1 On-Chip I/O Register Addresses (cont)

| Address | Register Abbr. | Bit Names | | | | | | | | Module |
|--------------------------------|----------------------|-----------|-------|-------|-------|-------|-------|-------|-------|------------------|
| | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| H'FFFF8426 | ADDRD | AD9 | AD8 | AD7 | AD6 | AD5 | AD4 | AD3 | AD2 | A/D(Mid-speed) |
| H'FFFF8427 | | AD1 | AD0 | — | — | — | — | — | — | |
| H'FFFF8428 | ADCSR | ADF | ADIE | ADST | SCAN | CKS | CH2 | CH1 | CH0 | (SH7016 |
| H'FFFF8429 | ADCR | TRGE | — | — | — | — | — | — | — | /17) |
| H'FFFF842A to H'FFFF857F | — | — | — | — | — | — | — | — | — | |
| H'FFFF8580 | FLMCR1 | FWE | SWE | ESU1 | PSU1 | EV1 | PV1 | E1 | P1 | FLASH |
| H'FFFF8581 | FLMCR2 | FLER | — | — | — | — | — | — | — | (F-ZTAT |
| H'FFFF8582 | EBR1 | EB7 | EB6 | EB5 | EB4 | EB3 | EB2 | EB1 | EB0 | version- |
| H'FFFF8583 to H'FFFF860F | — | — | — | — | — | — | — | — | — | only) |
| H'FFFF8610 | TCSR | OVF | WT/IT | TME | — | — | CKS2 | CKS1 | CKS0 | WDT |
| H'FFFF8610 | TCNT* ³ | | | | | | | | | |
| H'FFFF8611 | TCNT* ⁴ | | | | | | | | | |
| H'FFFF8612 | RSTCSR* ³ | WOVF | RSTE | — | — | — | — | — | — | |
| H'FFFF8613 | RSTCSR* ⁴ | WOVF | RSTE | — | — | — | — | — | — | |
| H'FFFF8614 | SBYCR | SBY | HIZ | — | — | — | — | — | — | Power-down state |
| H'FFFF8615 to H'FFFF861F | — | — | — | — | — | — | — | — | — | BSC |
| H'FFFF8620 | BCR1 | — | — | — | — | — | — | — | IOE | |
| H'FFFF8621 | — | — | — | — | — | A3SZ | A2SZ | A1SZ | A0SZ | |
| H'FFFF8622 | BCR2 | IW31 | IW30 | IW21 | IW20 | IW11 | IW10 | IW01 | IW00 | |
| H'FFFF8623 | | CW3 | CW2 | CW1 | CW0 | SW3 | SW2 | SW1 | SW0 | |
| H'FFFF8624 | WCR1 | W33 | W32 | W31 | W30 | W23 | W22 | W21 | W20 | |
| H'FFFF8625 | | W13 | W12 | W11 | W10 | W03 | W02 | W01 | W00 | |
| H'FFFF8626 | WCR2 | — | — | — | — | — | — | — | — | |
| H'FFFF8627 | — | — | — | DDW1 | DDW0 | DSW3 | DSW2 | DSW1 | DSW0 | |
| H'FFFF8628 | RAMER | — | — | — | — | — | — | — | — | FLASH |
| H'FFFF8629 | — | — | — | — | — | — | RAMS | RAM1 | RAM0 | (F-ZTAT |
| | | | | | | | | | | version only) |

Notes: 3. Write address.

4. Read address. For details, see 13.2.4, Notes on Register Access, in section 13, Watchdog Timer.

Table A.1 On-Chip I/O Register Addresses (cont)

| Address | Register Abbr. | Bit Names | | | | | | | | Module |
|------------|-----------------------|-----------|-------|-------|-------|-------|-------|-------|-------|--------|
| | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| H'FFFF862A | DCR | TPC | RCD | TRAS1 | TRAS0 | DWW1 | DWW0 | DWR1 | DWR0 | BSC |
| H'FFFF862B | | DIW | — | BE | RASD | — | SZ0 | AMX1 | AMX0 | |
| H'FFFF862C | RTCSR | — | — | — | — | — | — | — | — | |
| H'FFFF862D | | — | CMF | CMIE | CKS2 | CKS1 | CKS0 | RFSH | RMD | |
| H'FFFF862E | RTCNT | — | — | — | — | — | — | — | — | |
| H'FFFF862F | | — | — | — | — | — | — | — | — | |
| H'FFFF8630 | RTCOR | — | — | — | — | — | — | — | — | |
| H'FFFF8631 | | — | — | — | — | — | — | — | — | |
| H'FFFF8632 | — to H'FFFF86AF | — | — | — | — | — | — | — | — | |
| H'FFFF86B0 | | DMAOR | — | — | — | — | — | — | — | DMAC |
| H'FFFF86B1 | — | | — | — | — | — | AE | NMIF | DME | |
| H'FFFF86B2 | — to H'FFFF86BF | — | — | — | — | — | — | — | — | |
| H'FFFF86C0 | | SAR0 | — | — | — | — | — | — | — | — |
| H'FFFF86C1 | — | | — | — | — | — | — | — | — | |
| H'FFFF86C2 | — | | — | — | — | — | — | — | — | |
| H'FFFF86C3 | — | | — | — | — | — | — | — | — | |
| H'FFFF86C4 | DAR0 | — | — | — | — | — | — | — | — | |
| H'FFFF86C5 | | — | — | — | — | — | — | — | — | |
| H'FFFF86C6 | | — | — | — | — | — | — | — | — | |
| H'FFFF86C7 | | — | — | — | — | — | — | — | — | |
| H'FFFF86C8 | | DMATCR0 | — | — | — | — | — | — | — | — |
| H'FFFF86C9 | — | | — | — | — | — | — | — | — | |
| H'FFFF86CA | — | | — | — | — | — | — | — | — | |
| H'FFFF86CB | — | | — | — | — | — | — | — | — | |
| H'FFFF86CC | CHCR0 | — | — | — | — | — | — | — | — | |
| H'FFFF86CD | | — | — | — | — | — | RL | AM | AL | |
| H'FFFF86CE | — | DM1 | DM0 | SM1 | SM0 | RS3 | RS2 | RS1 | RS0 | |
| H'FFFF86CF | — | DS | TM | TS1 | TS0 | IE | TE | DE | | |
| H'FFFF86D0 | SAR1 | — | — | — | — | — | — | — | — | |
| H'FFFF86D1 | | — | — | — | — | — | — | — | — | |
| H'FFFF86D2 | | — | — | — | — | — | — | — | — | |
| H'FFFF86D3 | | — | — | — | — | — | — | — | — | |

Table A.1 On-Chip I/O Register Addresses (cont)

| Address | Register Abbr. | Bit Names | | | | | | | | Module |
|--------------------------------|-------------------|-----------|-------|-------|--------|-------|-------|-------|-------|--------|
| | | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| H'FFFF86D4 | DAR1 | | | | | | | | | DMAC |
| H'FFFF86D5 | | | | | | | | | | |
| H'FFFF86D6 | | | | | | | | | | |
| H'FFFF86D7 | | | | | | | | | | |
| H'FFFF86D8 | DMATCR1 | | | | | | | | | |
| H'FFFF86D9 | | | | | | | | | | |
| H'FFFF86DA | | | | | | | | | | |
| H'FFFF86DB | | | | | | | | | | |
| H'FFFF86DC | CHCR1 | — | — | — | — | — | — | — | — | |
| H'FFFF86DD | | — | — | — | — | — | RL | AM | AL | |
| H'FFFF86DE | | DM1 | DM0 | SM1 | SM0 | RS3 | RS2 | RS1 | RS0 | |
| H'FFFF86DF | | — | DS | TM | TS1 | TS0 | IE | TE | DE | |
| H'FFFF87E0 to H'FFFF873F | — | — | — | — | — | — | — | — | — | DTC |
| H'FFFF8740 | CCR | — | — | — | — | — | — | — | — | CAC |
| H'FFFF8741 | | — | — | — | CEDRAM | CECS3 | CECS2 | CECS1 | CECS0 | |
| H'FFFF8742 to H'FFFF87FF | — | — | — | — | — | — | — | — | — | |

Appendix B I/O Port Block Diagrams

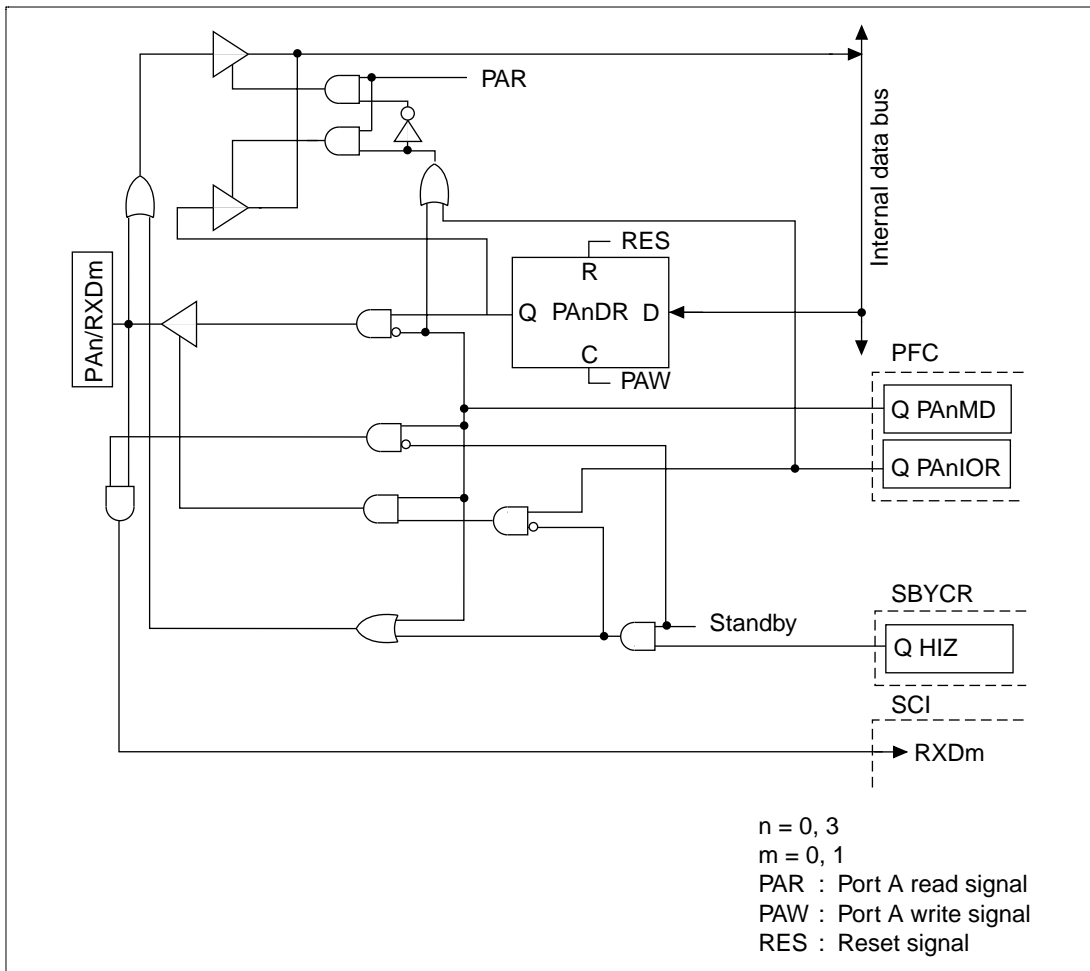


Figure B.1 PAn/RXm Block Diagram

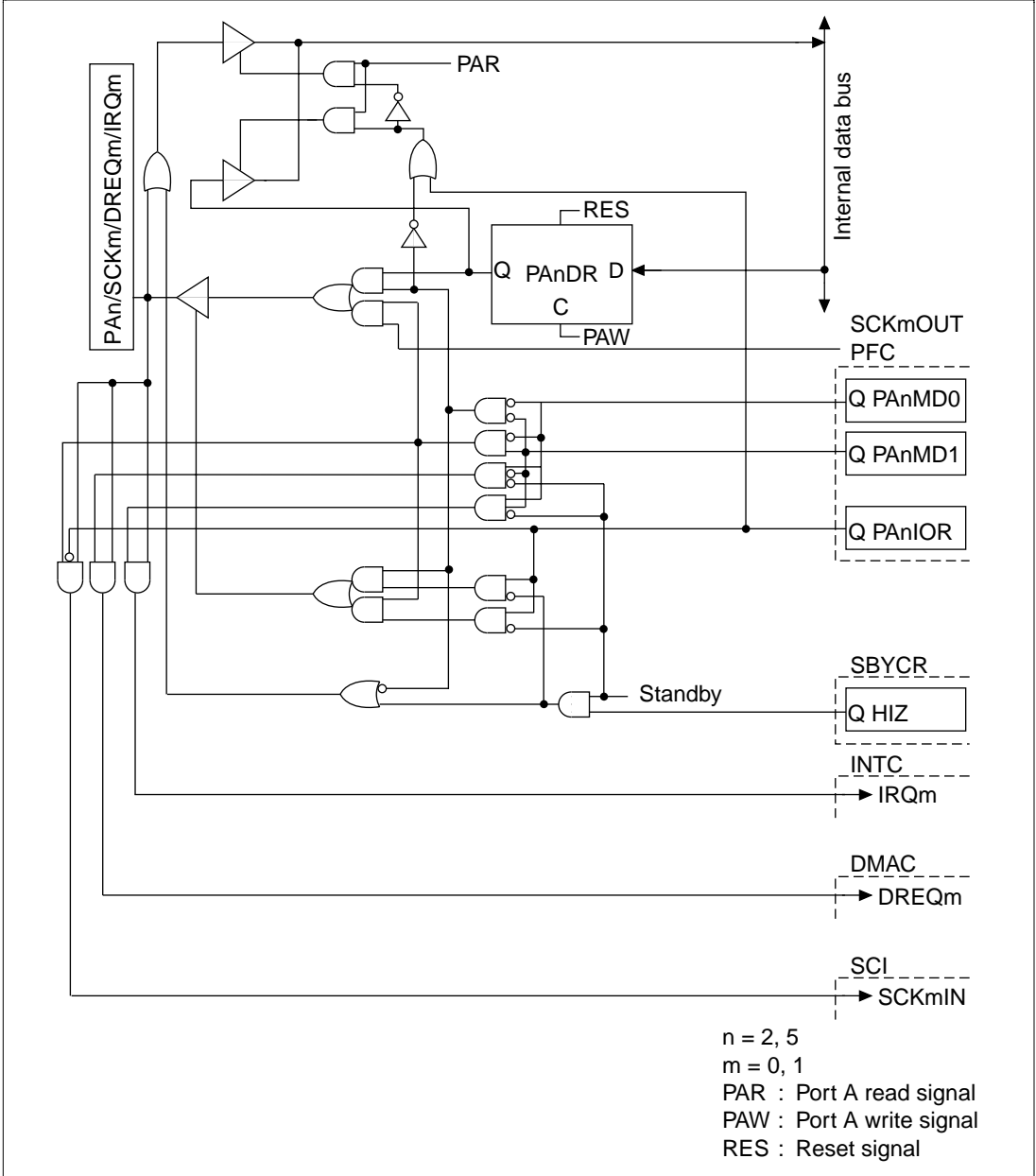


Figure B.2 PAn/SCKm/DREQm/IRQm Block Diagram

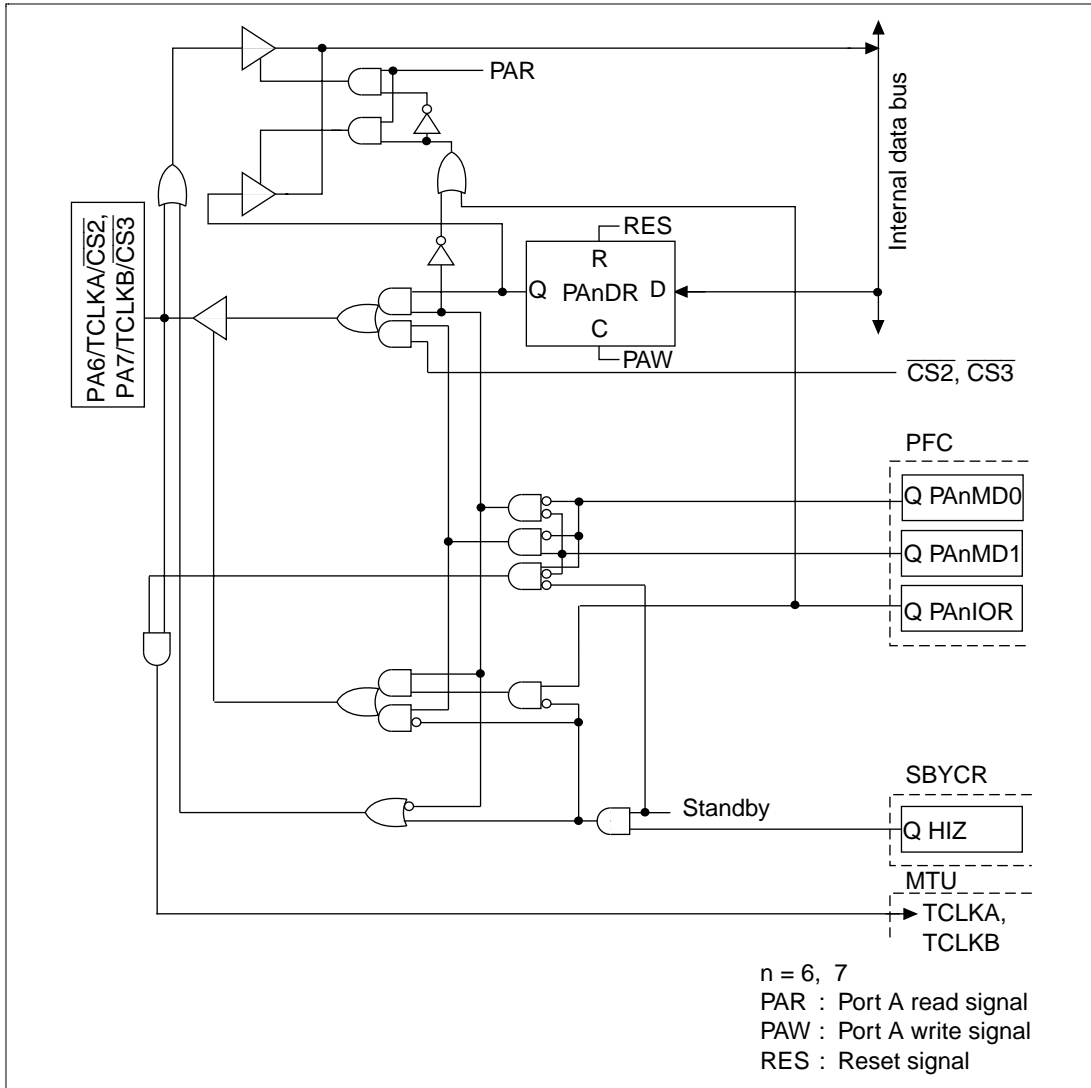


Figure B.3 PA6/TCLKA/CS2, PA7/TCLKB/CS3 (SH7014, Mask) Block Diagram

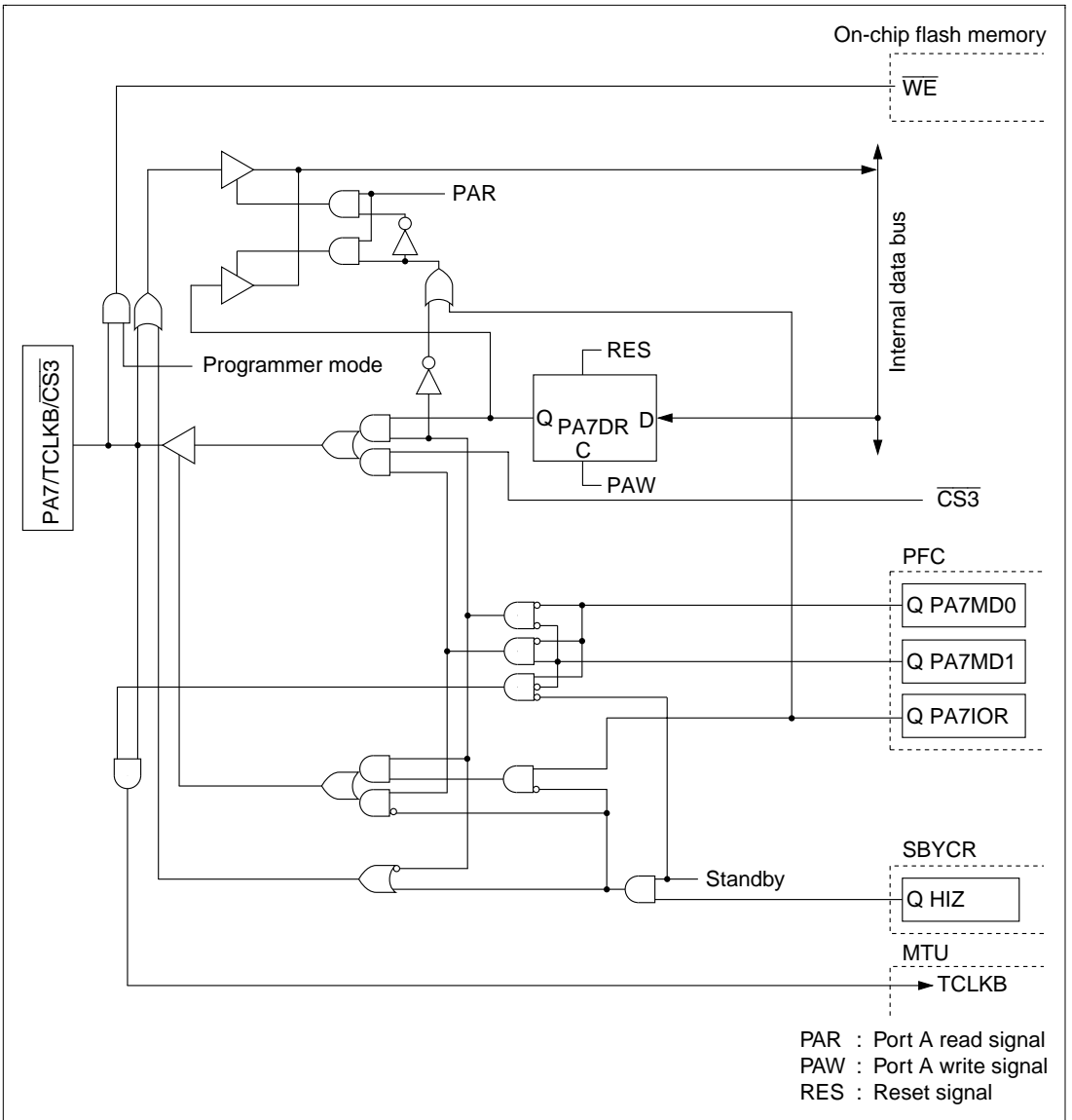


Figure B.4 PA7/TCLKB/ $\overline{\text{CS3}}$ Block Diagram (F-ZTAT Version)

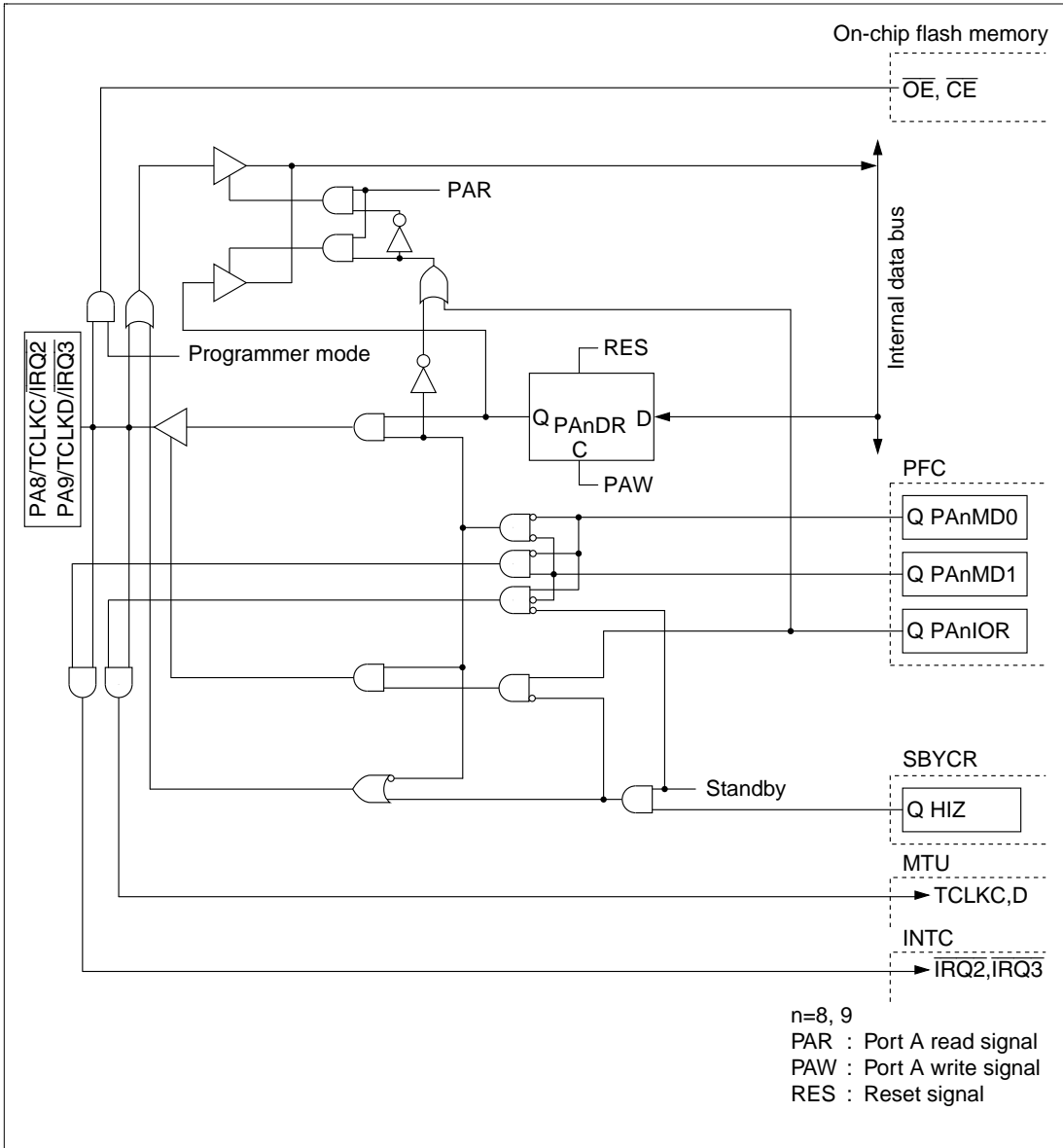


Figure B.6 PAn/TCLKm/IRQx Block Diagram (F-ZTAT Version)

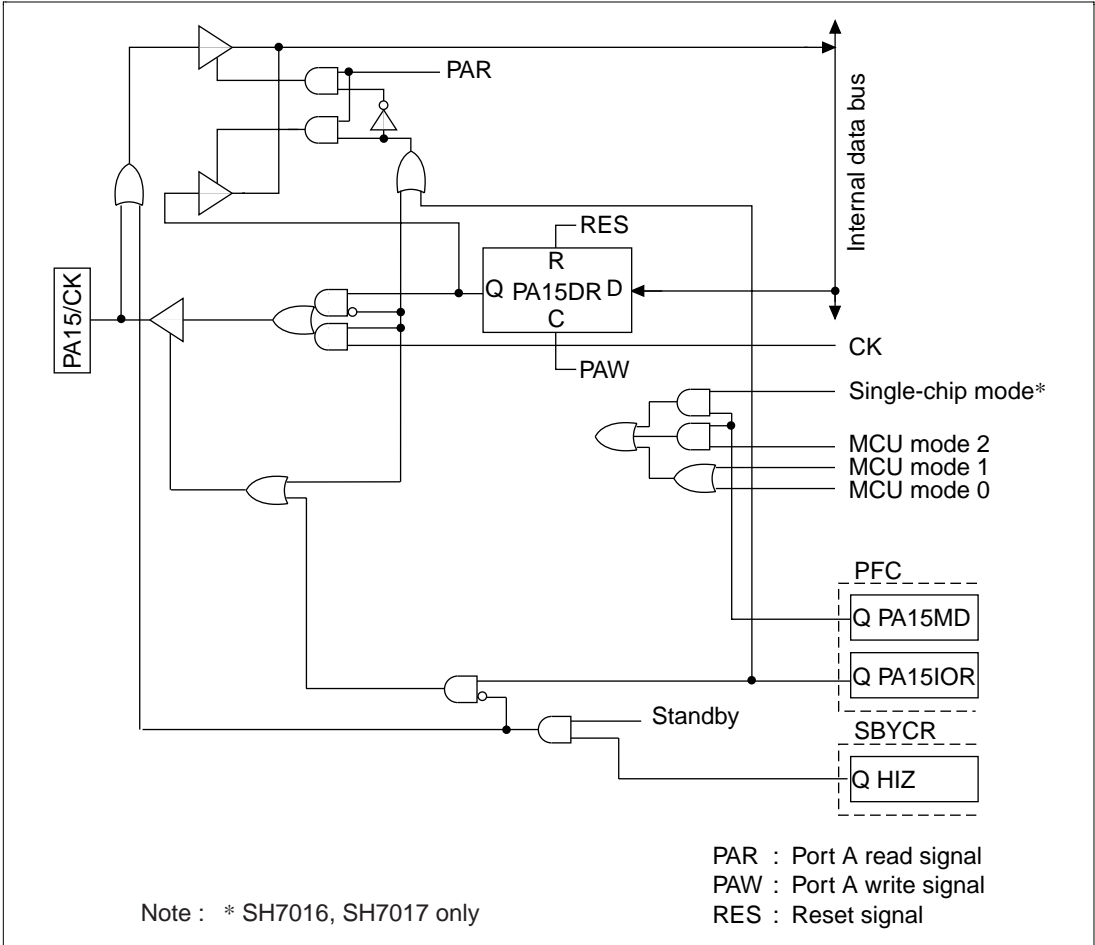


Figure B.8 PA15/CK Block Diagram

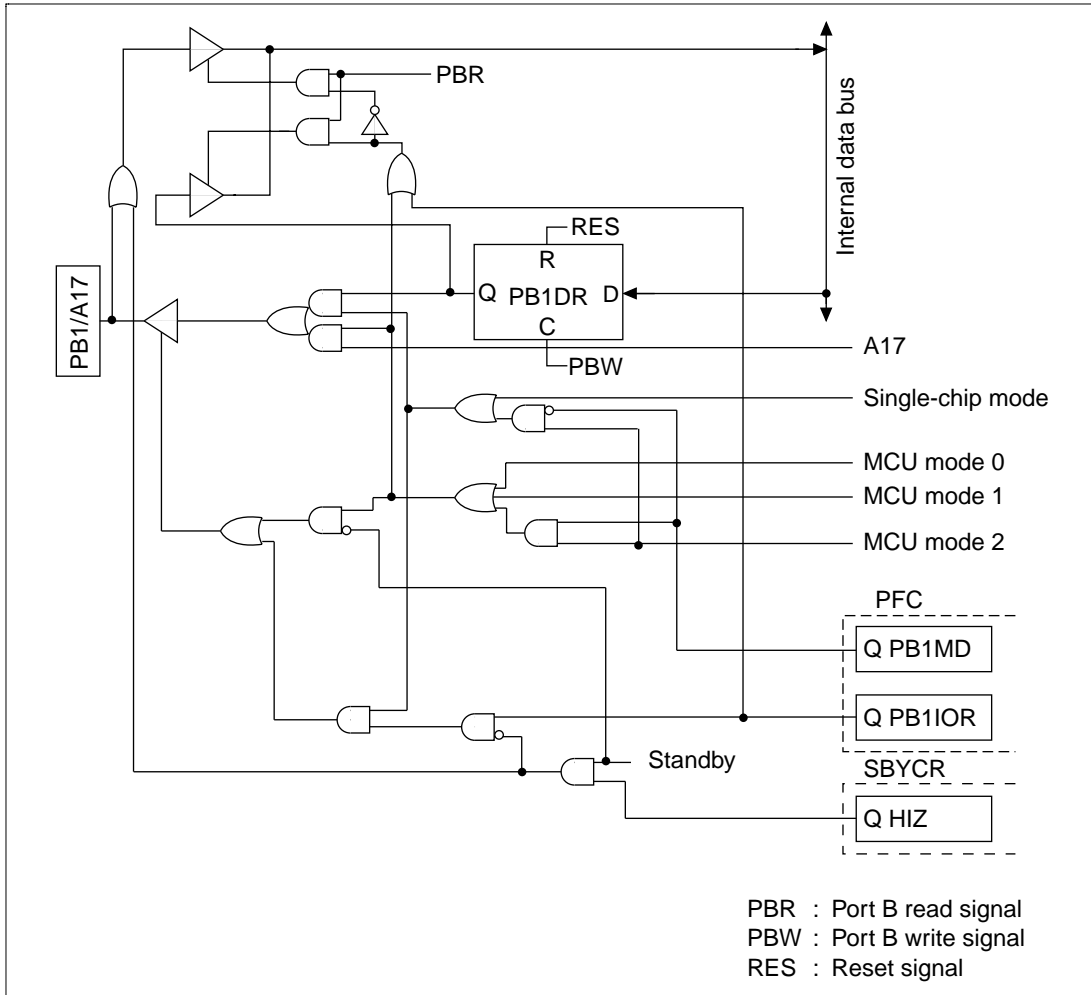


Figure B.11 PB1/A17 Block Diagram (SH7016, SH7017)

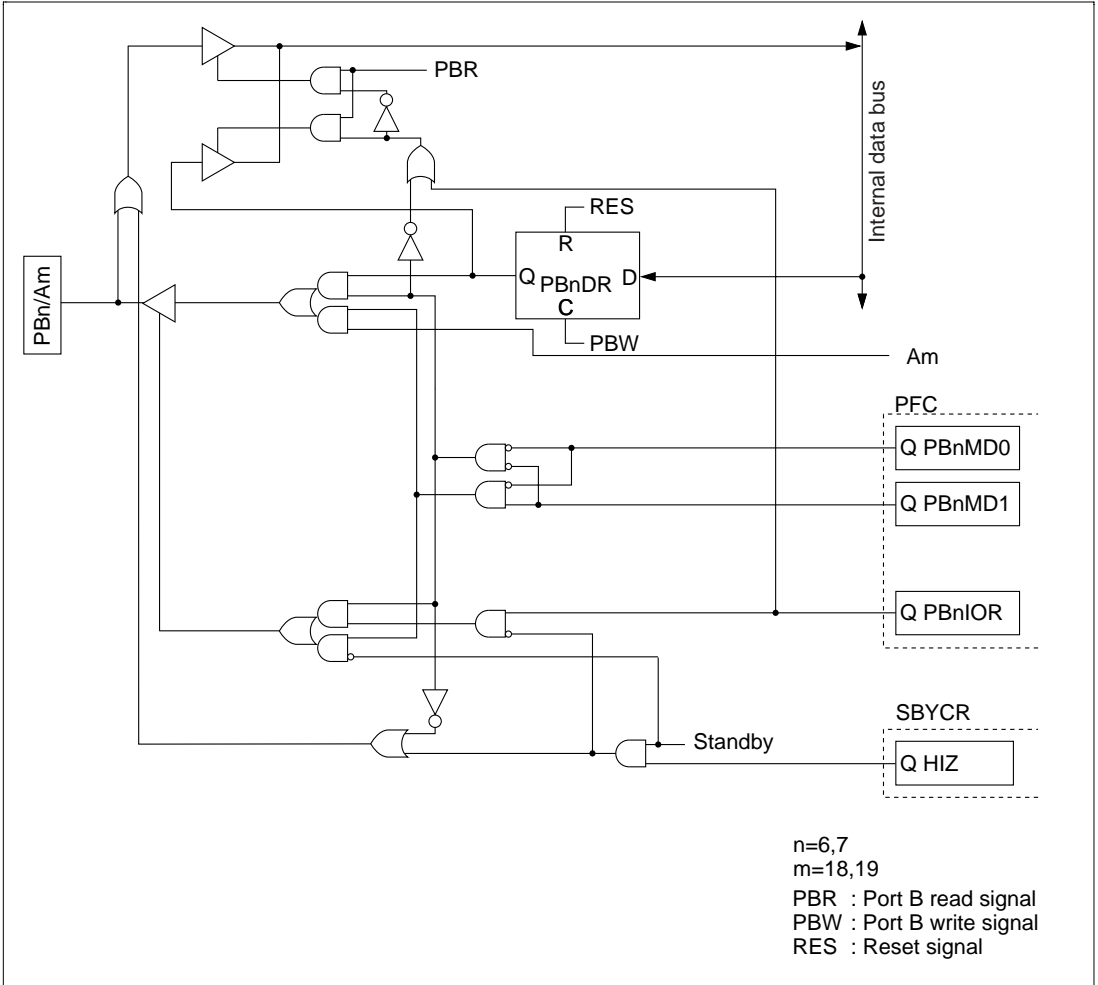


Figure B.12 PBn/Am Block Diagram

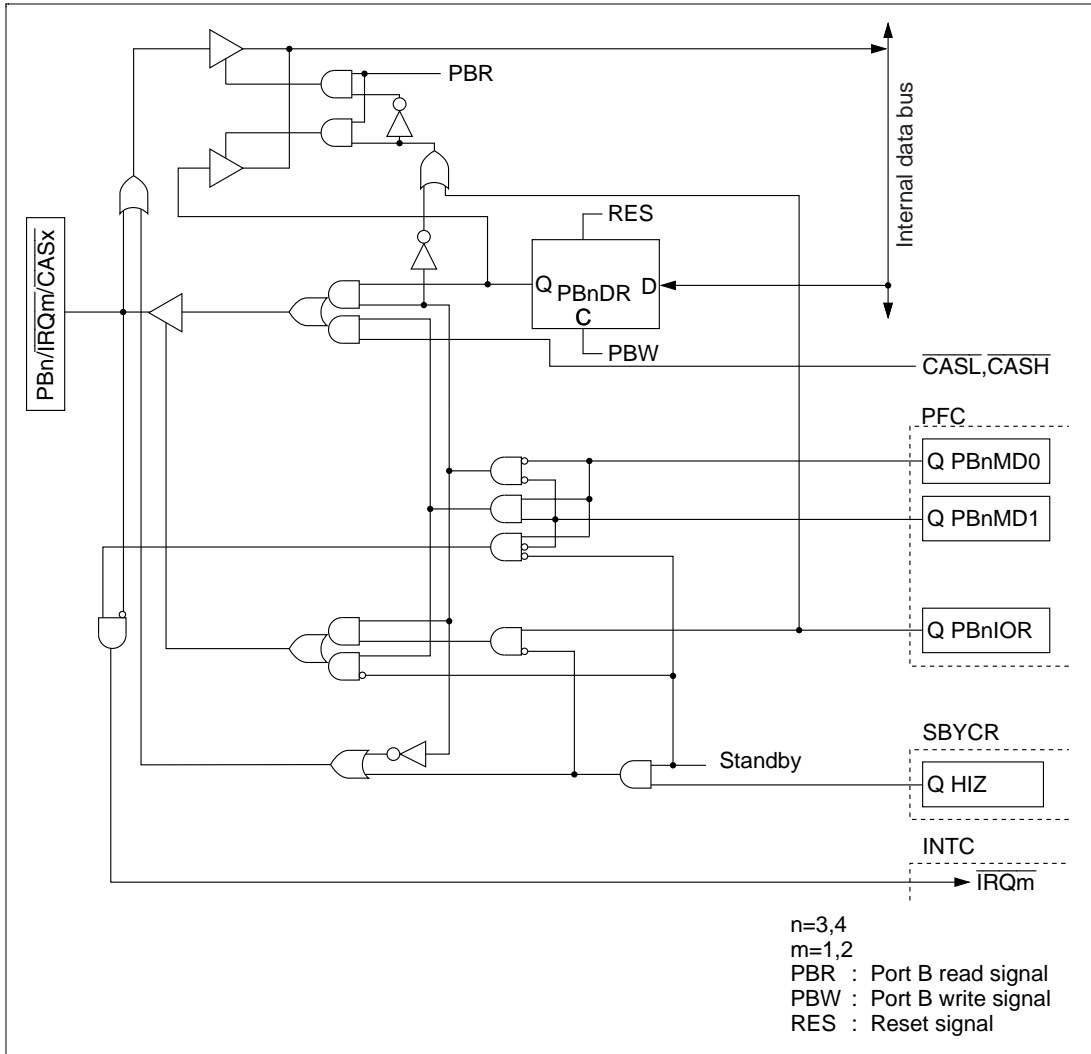


Figure B.13 $\overline{PBN}/\overline{IRQm}/\overline{CASx}$ Block Diagram

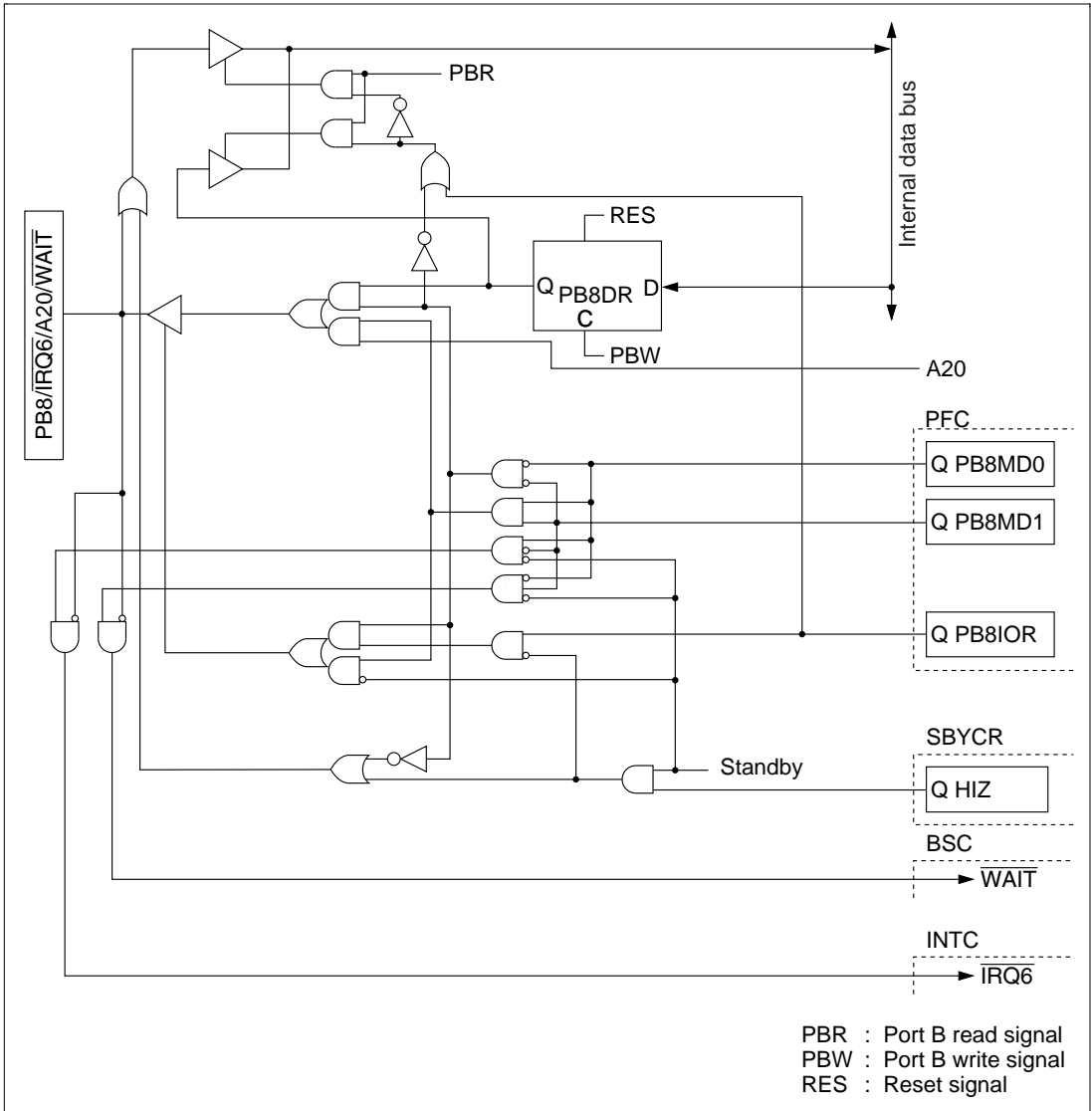


Figure B.14 PB8/IRQ6/A20/WAIT Block Diagram

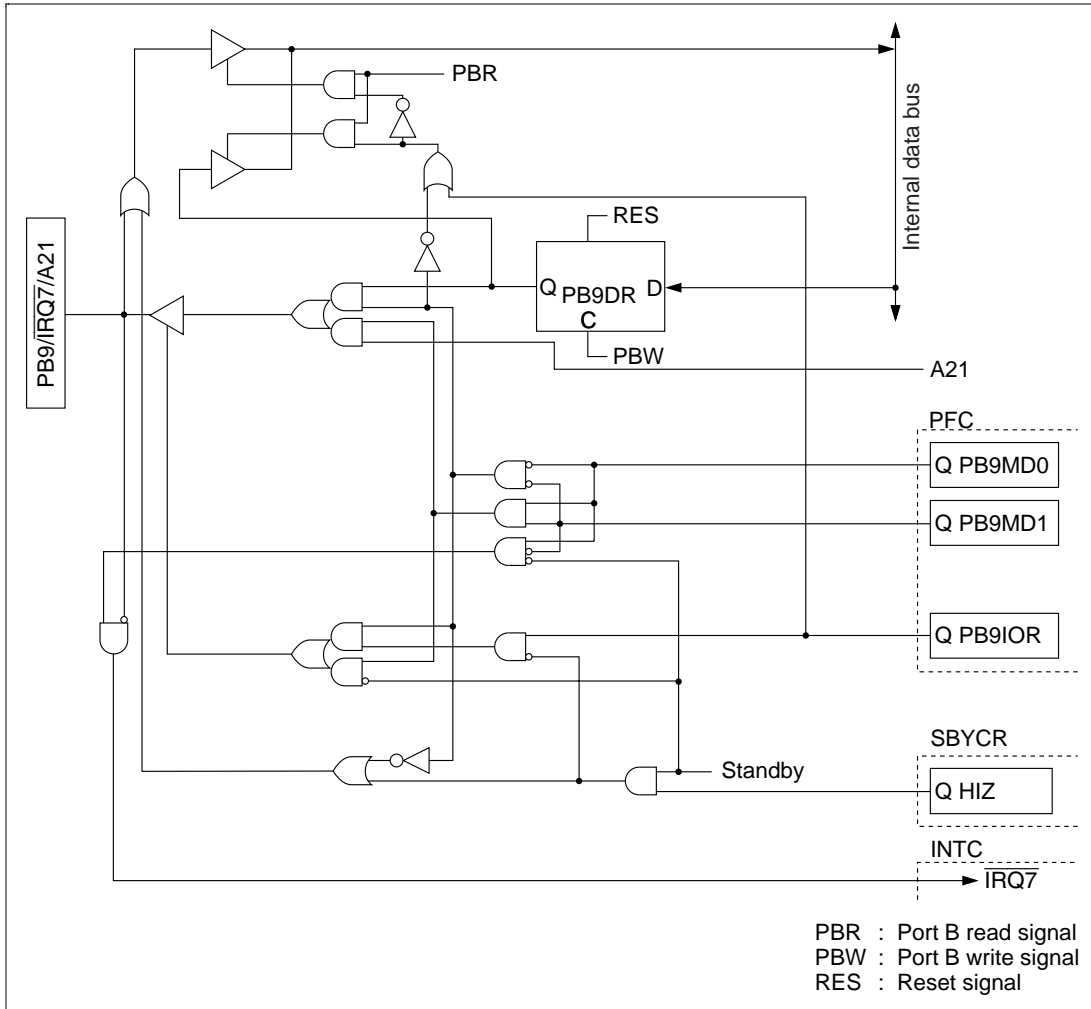


Figure B.15 PB9/IRQ7/A21 Block Diagram

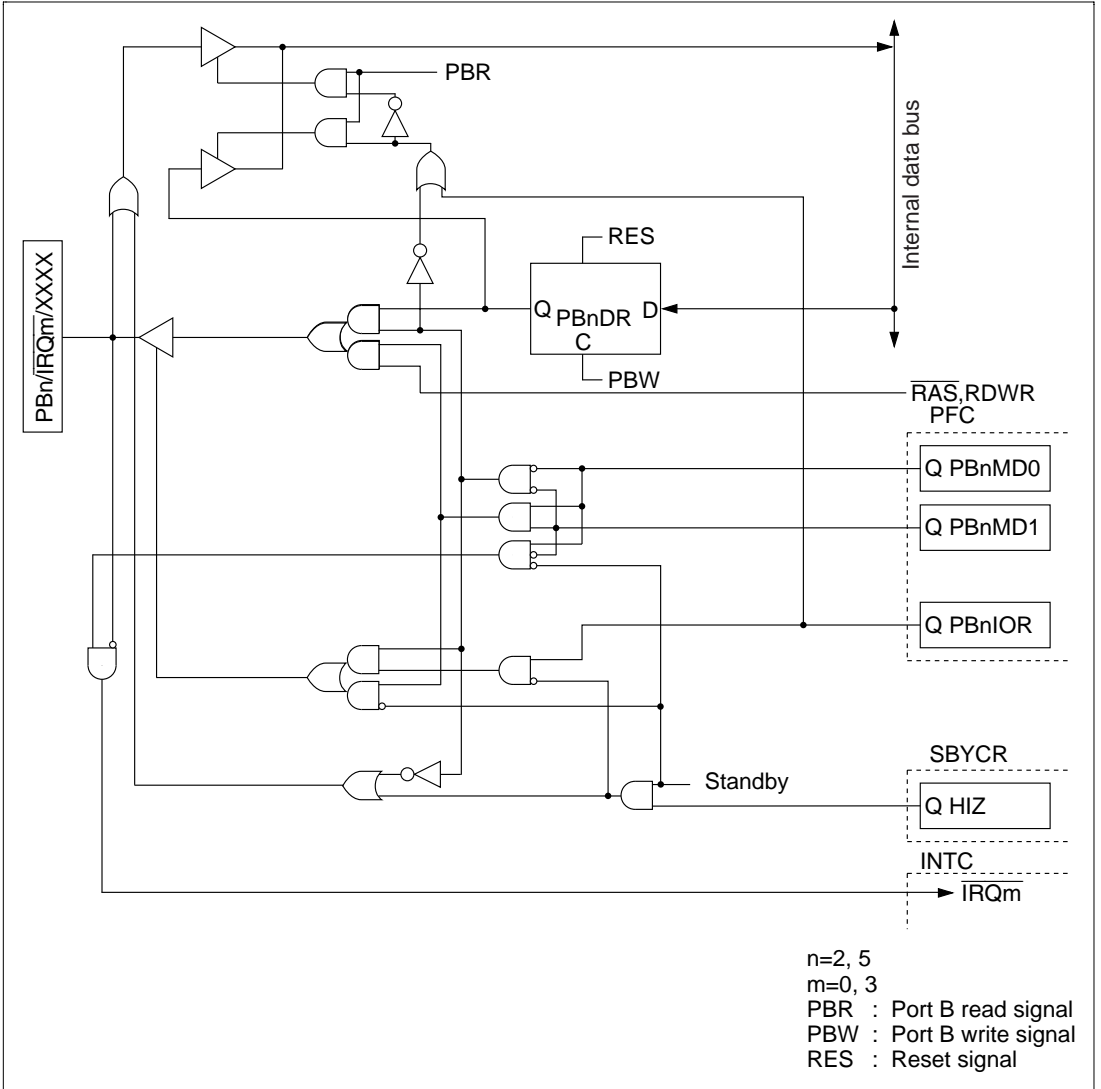


Figure B.16 $\overline{\text{PBn}}/\overline{\text{IRQm}}/\text{XXXX}$ Block Diagram

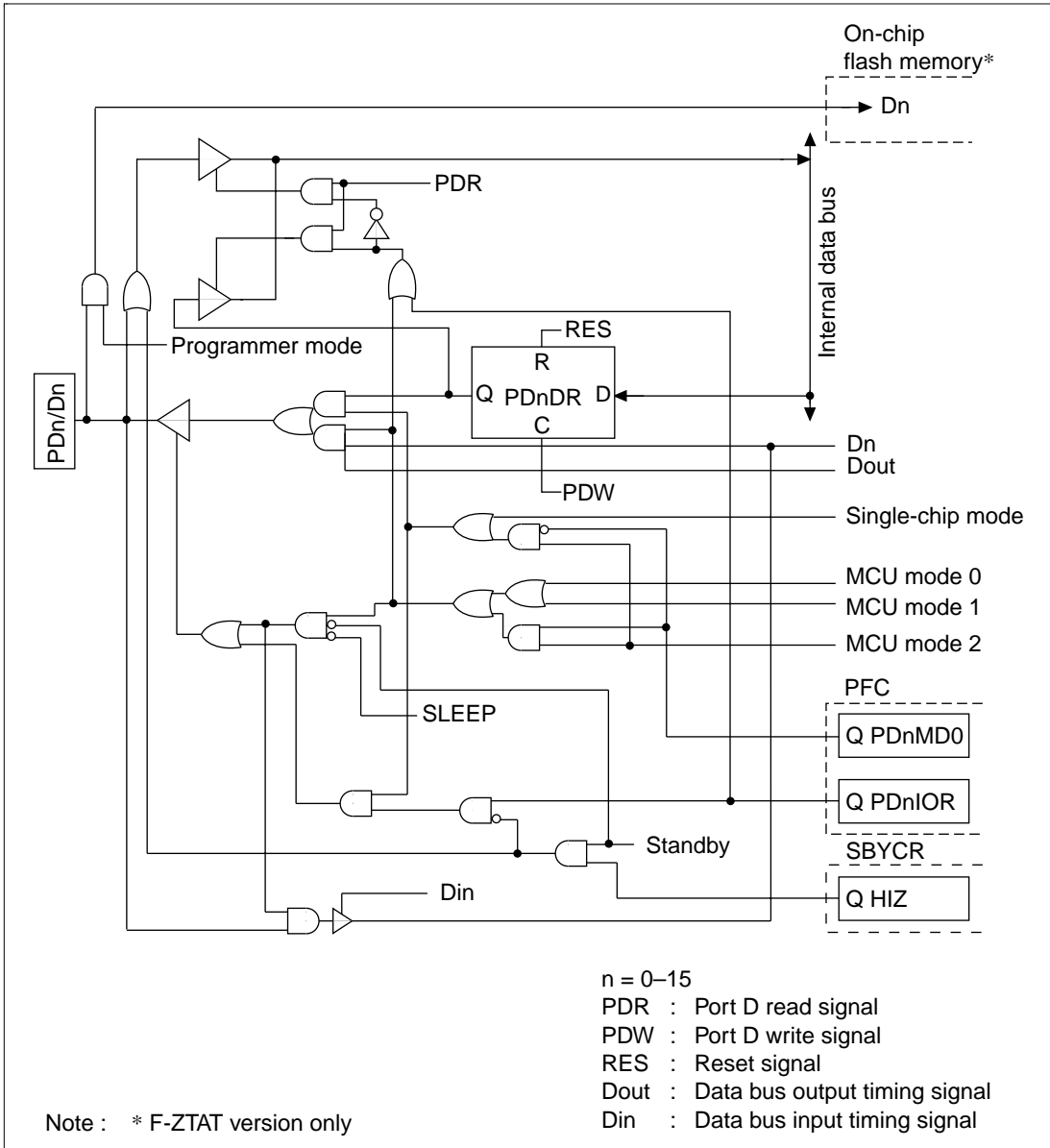


Figure B.18 PDn/Dn Block Diagram (SH7016, SH7017)

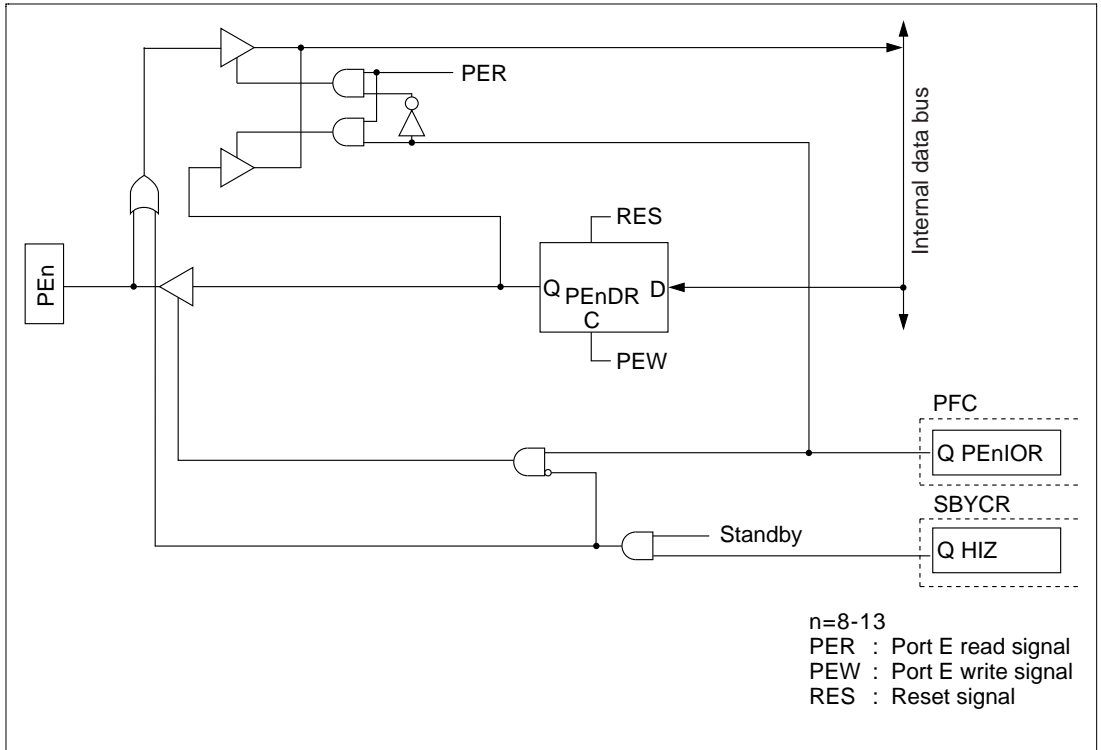


Figure B.19 PEn Block Diagram

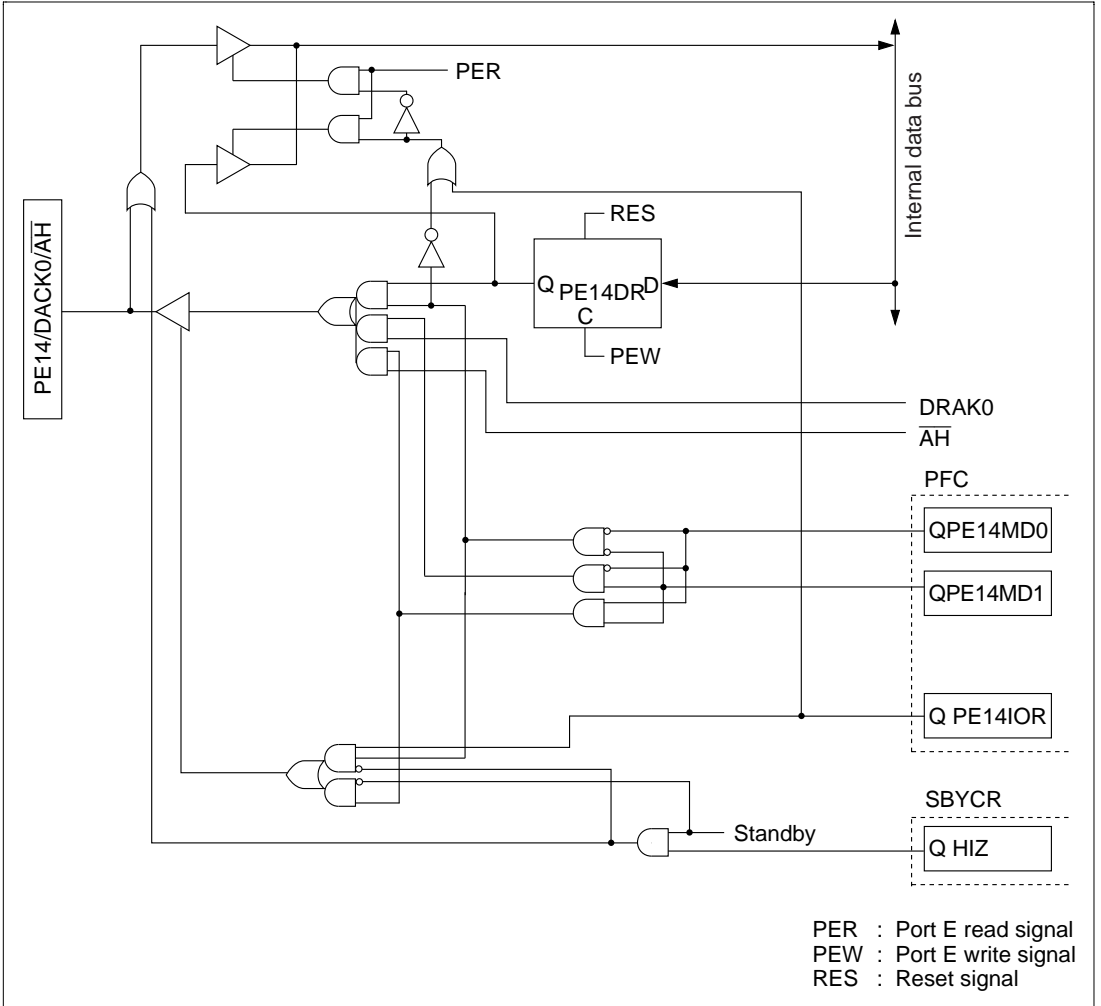


Figure B.20 PE14/DACK0/AH Block Diagram

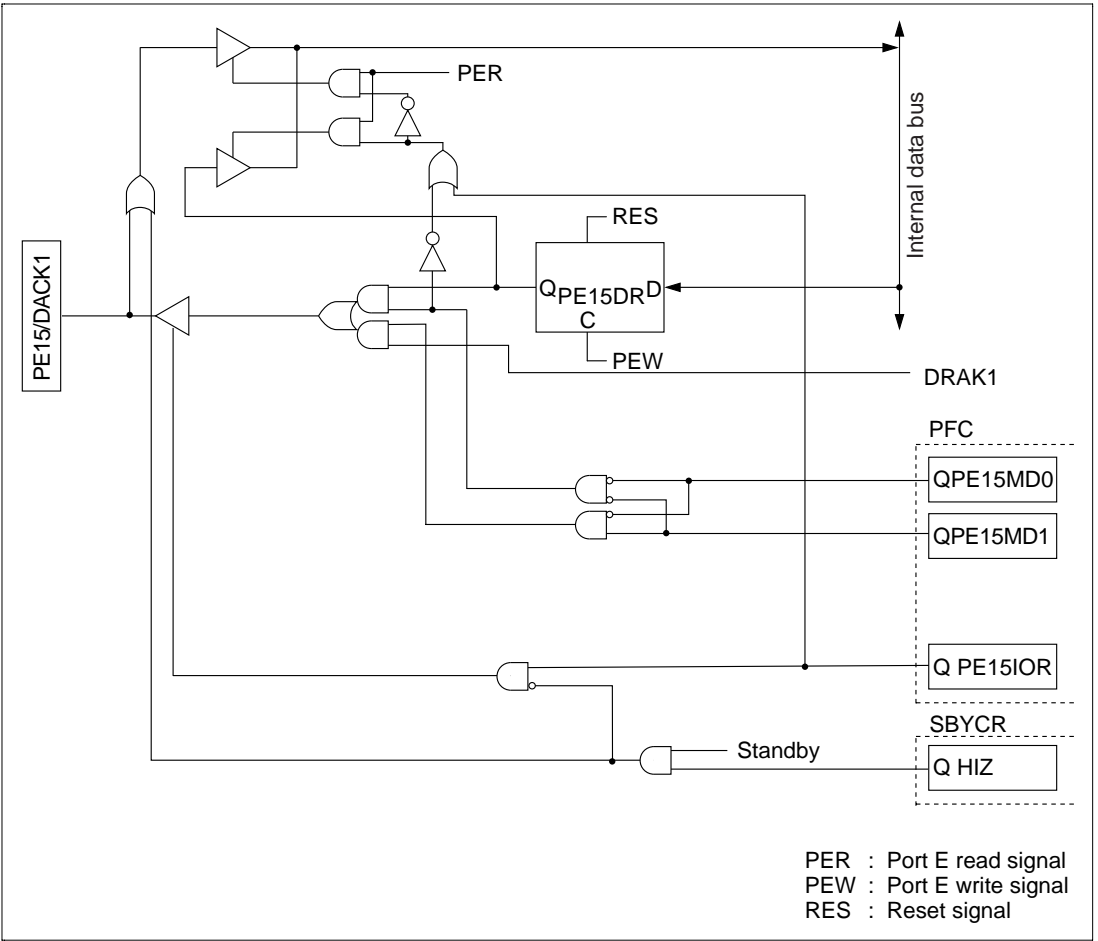
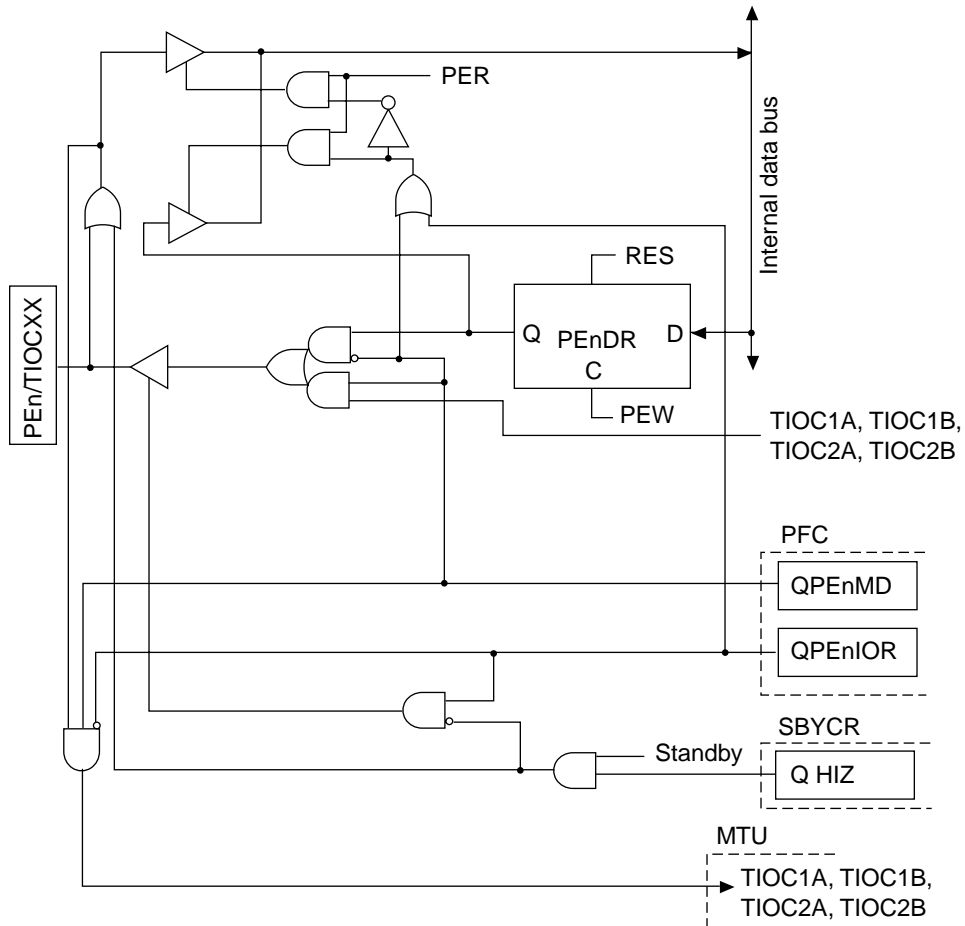
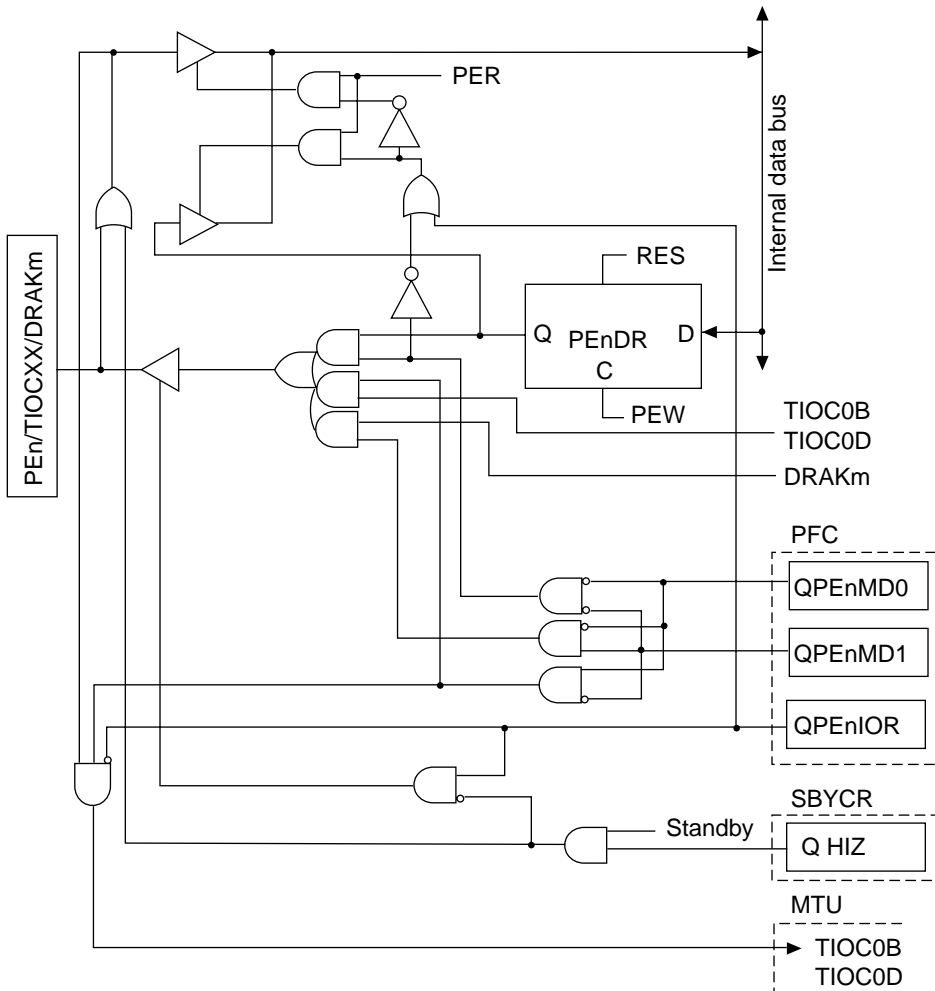


Figure B.21 PE15/DACK1 Block Diagram



n = 4-7
 PER : Port E read signal
 PEW : Port E write signal
 RES : Reset signal

Figure B.22 PEn/TIOCXX Block Diagram



$n = 1, 3$
 $m = 0, 1$
 PER : Port E read signal
 PEW : Port E write signal
 RES : Reset signal

Figure B.23 PEn/TIOCXX/DRAKm Block Diagram

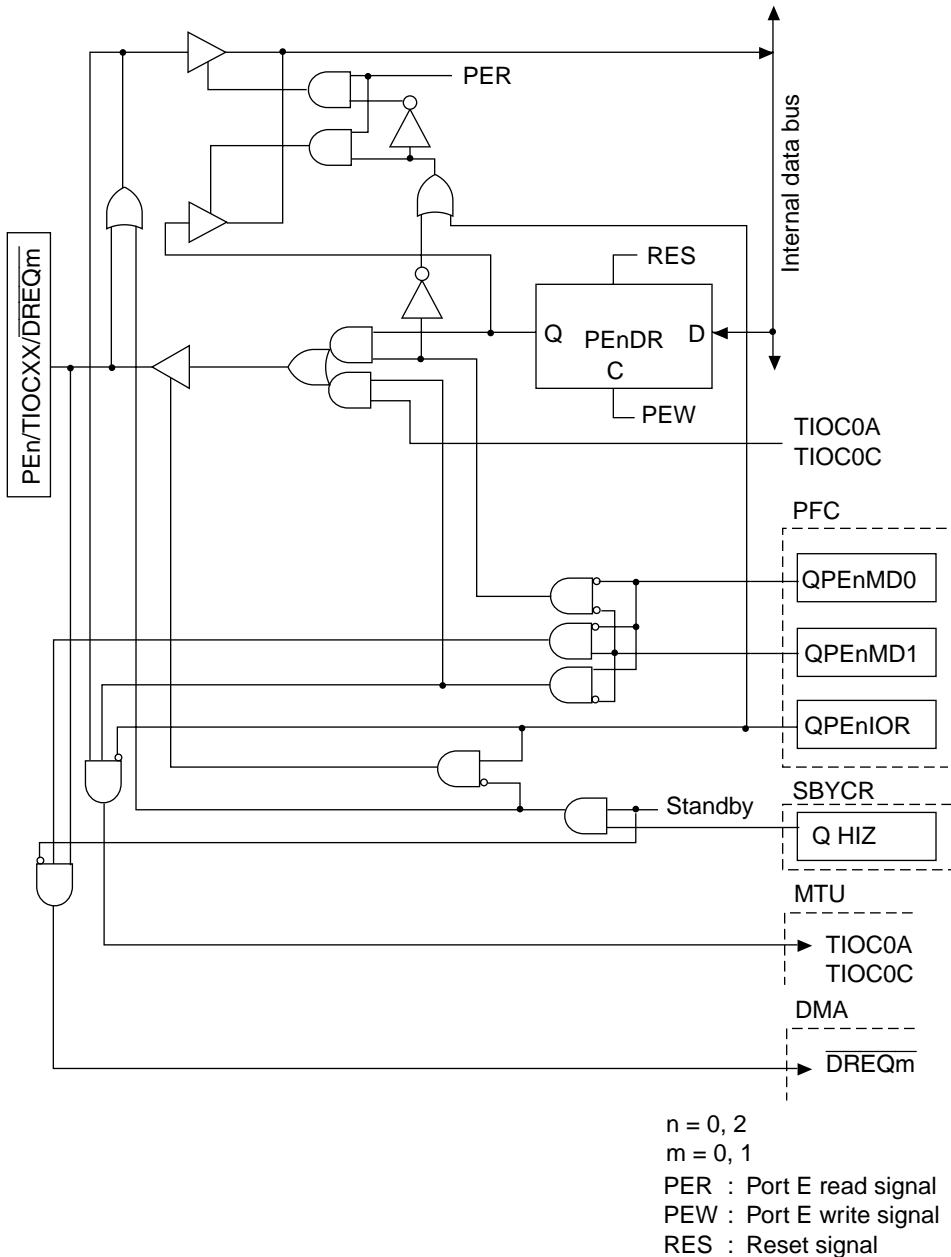


Figure B.24 PEn/TIOCXX/ \overline{DREQm} Block Diagram

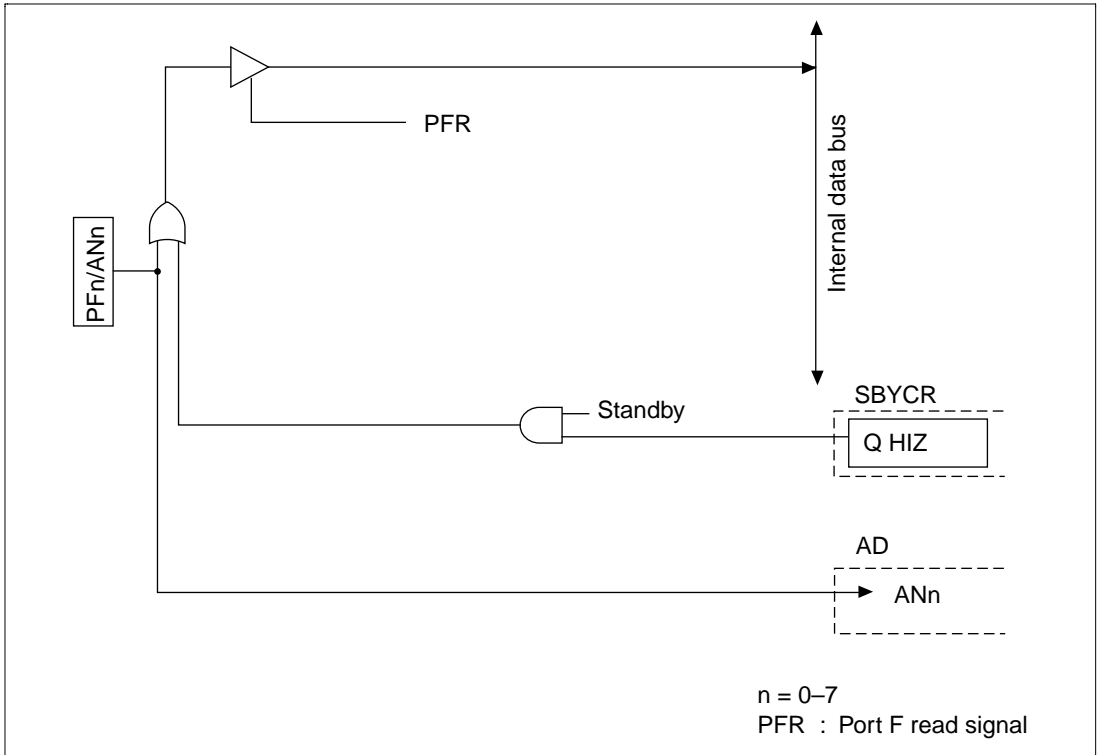


Figure B.25 PFn/ANn Block Diagram

Appendix C Pin States

Table C.1 Pin Modes During Reset, Power-Down, and Bus Right Release Modes

| Pin Function | | Pin modes | | |
|----------------|---|-----------|-----------------|-------|
| | | Reset | Power-Down | |
| Class | Pin Name | Power-On | Standby | Sleep |
| Clock | CK | O | H* ¹ | O |
| System control | $\overline{\text{RES}}$ | I | I | I |
| | $\overline{\text{WDTOVF}}$ | O | O | O |
| Interrupt | NMI | I | I | I |
| | $\overline{\text{IRQ0}}\text{--}\overline{\text{IRQ3}}$, $\overline{\text{IRQ6}}$, $\overline{\text{IRQ7}}$ | Z | Z | I |
| Address bus | A0–A21 | O | Z | O |
| Data bus | D0–D31 | Z | Z | I/O |
| Bus control | $\overline{\text{WAIT}}$ | Z | Z | I |
| | $\overline{\text{RDWR}}$, $\overline{\text{RAS}}$ | Z | O | H |
| | $\overline{\text{CASH}}$, $\overline{\text{CASL}}$ | Z | O | H |
| | $\overline{\text{RD}}$ | H | Z | H |
| | $\overline{\text{CS0}}$, $\overline{\text{CS1}}$ | H | Z | H |
| | $\overline{\text{CS2}}$, $\overline{\text{CS3}}$ | Z | Z | H |
| | $\overline{\text{WRH}}$, $\overline{\text{WRL}}$ | H | Z | H |
| | $\overline{\text{AH}}$ | Z | Z | L |
| DMAC | DACK0, DACK1 | Z | O* ¹ | O |
| | DRAK0, DRAK1 | Z | O* ¹ | O |
| | $\overline{\text{DREQ0}}$, $\overline{\text{DREQ1}}$ | Z | Z | I |
| MTU | TIOC0A–TIOC0D, TIOC1A, TIOC1B, TIOC2A, TIOC2B | Z | K* ¹ | I/O |
| | TCLKA–TCLKD | Z | Z | I |
| SCI | SCK0, SCK1 | Z | Z | I/O |
| | TXD0, TCD1 | Z | O* ¹ | O |
| | RXD0, RXD1 | Z | Z | I |
| A/D converter | AN0–AN7 | Z | Z | I |

Table C.1 Pin Modes During Reset, Power-Down, and Bus Right Release Modes (cont)

| Pin Function | | Pin modes | | |
|--------------|------------------------|-----------|-----------------|-------|
| | | Reset | Power-Down | |
| Class | Pin Name | Power-On | Standby | Sleep |
| I/O Port | PA0–PA9, PA15 | Z | K* ¹ | K |
| | PB0–PB9 | | | |
| | PC0–PC15 | | | |
| | PD0–PD15 | | | |
| | PE0–PE15* ² | | | |
| | PF0–PF7 | Z | Z | I |

- Notes:
1. If the standby control register port high-impedance bits are set to 1, output pins become high impedance.
 2. When an emulator is used, PE9 and P11 to P15 become high-impedance in standby mode.
 3. I: Input, O: Output, H: High-level output, L: Low-level output, Z: High impedance, K: Input pin with high impedance, output pin mode maintained.

Table C.2 Pin Settings for On-Chip Peripheral Modules

| Pin Name | On-Chip Peripheral Module | | | | | |
|---|---------------------------------|----------------|-------------|--------------|---------------|-------------------|
| | On-Chip ROM (SH7016/17 only) | On-Chip RAM | 8-Bit Space | 16-Bit Space | | |
| | | | | Upper Byte | Lower Byte | Word/ Longword |
| $\overline{\text{CS0}}\text{--}\overline{\text{CS3}}$ | H | H | H | H | H | H |
| $\overline{\text{RAS}}^{*1}$ | H | H | H | H | H | H |
| $\overline{\text{CASH}}^{*2}$ | H | H | H | H | H | H |
| $\overline{\text{CASL}}^{*2}$ | H | H | H | H | H | H |
| RDWR | H | H | H | H | H | H |
| $\overline{\text{AH}}$ | L | L | L | L | L | L |
| $\overline{\text{RD}}$ | R | H | H | H | H | H |
| | W | — | H | H | H | H |
| $\overline{\text{WRH}}$ | R | H | H | H | H | H |
| | W | — | H | H | H | H |
| $\overline{\text{WRL}}$ | R | H | H | H | H | H |
| | W | — | H | H | H | H |
| A21–A0 | Address | Address | Address | Address | Address | Address |
| D15–D8 | High-Z | High-Z | High-Z | High-Z | High-Z | High-Z |
| D7–D0 | High-Z | High-Z | High-Z | High-Z | High-Z | High-Z |

Notes: 1. L asserted in RAS down state or refresh state.

2. L asserted in refresh state.

3. R: Read W: Write.

Table C.3 Pin Settings for Normal External Space

| Pin Name | 16-Bit Space | | | |
|---|--------------|------------|------------|---------------|
| | 8-Bit Space | Upper Word | Lower Word | Word/Longword |
| $\overline{\text{CS0}}\text{--}\overline{\text{CS2}}$ | Valid | Valid | Valid | Valid |
| $\overline{\text{CS3}}$ | H | H | H | H |
| $\overline{\text{RAS}}^{*1}$ | H | H | H | H |
| $\overline{\text{CASH}}^{*2}$ | H | H | H | H |
| $\overline{\text{CASL}}^{*2}$ | H | H | H | H |
| RDWR | H | H | H | H |
| $\overline{\text{AH}}$ | L | L | L | L |
| $\overline{\text{RD}}$ | R | L | L | L |
| | W | H | H | H |
| $\overline{\text{WRH}}$ | R | H | H | H |
| | W | H | L | L |
| $\overline{\text{WRL}}$ | R | H | H | H |
| | W | L | H | L |
| A21–A0 | Address | Address | Address | Address |
| D15–D8 | High-Z | Data | High-Z | Data |
| D7–D0 | Data | High-Z | Data | Data |

Notes: 1. L asserted in RAS down state or refresh state.

2. L asserted in refresh state.

3. R: Read W: Write.

Table C.4 Pin Settings for Multiplex I/O Space

| Pin Name | 16-Bit Space | | | |
|---|--------------|--------------|--------------|---------------|
| | 8-Bit Space | Upper Byte | Lower Byte | Word/Longword |
| $\overline{\text{CS0}}\text{--}\overline{\text{CS2}}$ | H | H | H | H |
| $\overline{\text{CS3}}$ | L | L | L | L |
| $\overline{\text{RAS}}^{*1}$ | H | H | H | H |
| $\overline{\text{CASH}}^{*2}$ | H | H | H | H |
| $\overline{\text{CASL}}^{*2}$ | H | H | H | H |
| RDWR | H | H | H | H |
| $\overline{\text{AH}}$ | Valid | Valid | Valid | Valid |
| $\overline{\text{RD}}$ | R | L | L | L |
| | W | H | H | H |
| $\overline{\text{WRH}}$ | R | H | H | H |
| | W | H | L | L |
| $\overline{\text{WRL}}$ | R | H | H | H |
| | W | L | H | L |
| A21–A0 | Address | Address | Address | Address |
| D15–D8 | High-Z | Address/Data | Address | Address/Data |
| D7–D0 | Address/Data | Address | Address/Data | Address/Data |

- Notes:
1. L asserted in RAS down mode or refresh mode.
 2. L asserted in refresh mode.
 3. R: Read W: Write
 4. Valid: High output in accordance with $\overline{\text{AH}}$ timing.

Table C.5 Pin Settings for DRAM Space

| Pin Name | 8-Bit Space | 16-Bit Space | | |
|---|-------------|--------------|------------|---------------|
| | | Upper Word | Lower Word | Word/Longword |
| $\overline{\text{CS0}}\text{--}\overline{\text{CS3}}$ | H | H | H | H |
| $\overline{\text{RAS}}^{*1}$ | Valid | Valid | Valid | Valid |
| $\overline{\text{CASH}}^{*2}$ | H | Valid | H | Valid |
| $\overline{\text{CASL}}^{*2}$ | Valid | H | Valid | Valid |
| RD/WR | R | H | H | H |
| | W | L | L | L |
| $\overline{\text{AH}}$ | L | L | L | L |
| $\overline{\text{RD}}$ | R | L | L | L |
| | W | H | H | H |
| $\overline{\text{WRH}}$ | R | H | H | H |
| | W | H | L | L |
| $\overline{\text{WRL}}$ | R | H | H | H |
| | W | L | H | L |
| A21–A0 | Address | Address | Address | Address |
| D15–D8 | High-Z | Data | High-Z | Data |
| D7–D0 | Data | High-Z | Data | Data |

Notes: 1. L asserted in RAS down mode or refresh mode.

2. L asserted in refresh mode.

3. R: Read W: Write

4. Valid: Asserted (low) at a timing determined by the DRAM access strobe waveform.

Appendix D Notes when Converting the F-ZTAT Application Software to the Mask-ROM Versions

Please note the following when converting the F-ZTAT application software to the mask-ROM versions.

The values read from the internal registers for the flash ROM of the mask-ROM version and F-ZTAT version differ as follows.

| Register | Bit | Status | |
|----------|-----|---|---|
| | | F-ZTAT Version | Mask-ROM Version |
| FLMCR1 | FWE | 0: Application software running 1: Programming | 0: Is not read out 1: Application software running |

Note: This difference applies to all the F-ZTAT versions and all the mask-ROM versions that have different ROM size.

Appendix E Product Code Lineup

Table E.1 SH7014, SH7016, and SH7017 Product Code Lineup

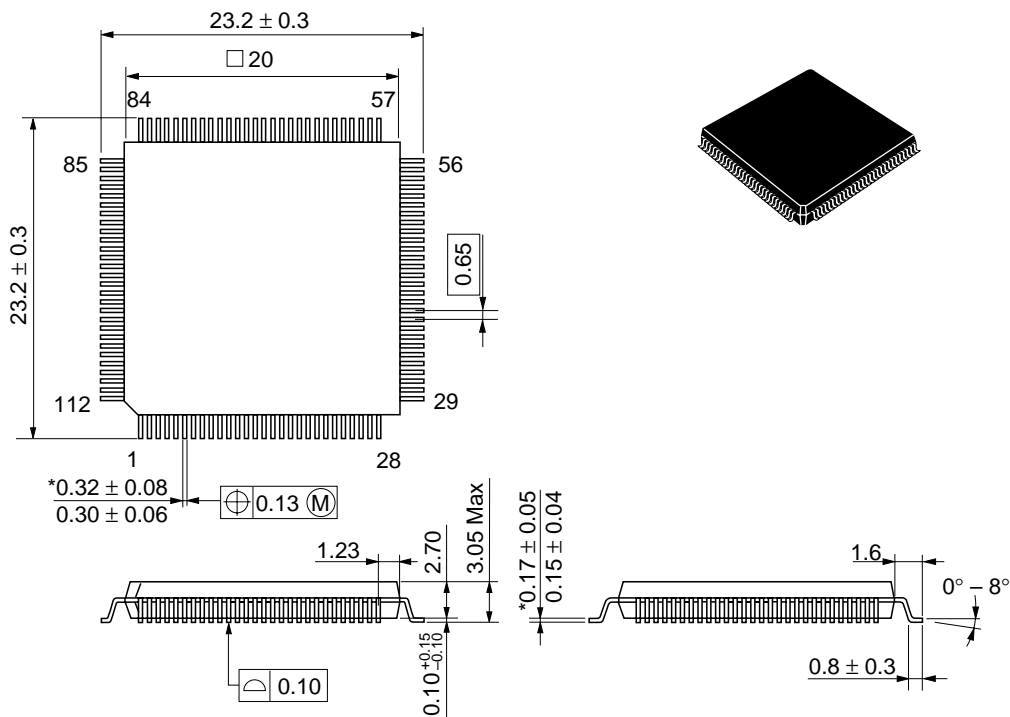
| Product Type | | Product Code | Mark Code | Package (Hitachi Package code) |
|---------------------|------------------|---------------------|-------------------|---------------------------------------|
| SH7014 | | HD6417014F28 | HD6417014F28 | 112-pin QFP (FP-112) |
| | | HD6417014RF28 | HD6417014RF28 | 112-pin QFP (FP-112) |
| SH7016 | Mask ROM version | HD6437016F28 | HD6437016(***)F28 | 112-pin QFP (FP-112) |
| SH7017 | F-ZTAT version | HD64F7017F28 | HD64F7017 | 112-pin QFP (FP-112) |

Note: (***) is the ROM code.

Appendix F Package Dimensions

Package dimensions of the SH7014, SH7016, SH7017 (FP-112) are shown in figure F.1.

Unit: mm



*Dimension including the plating thickness
Base material dimension

| | |
|--------------------------|----------|
| Hitachi Code | FP-112 |
| JEDEC | — |
| EIAJ | Conforms |
| Weight (reference value) | 2.4 g |

Figure F.1 Package Dimensions (FP-112)

SH7014, SH7016, SH7017F-ZTAT™ Hardware Manual

Publication Date: 1st Edition, September 1997
4th Edition, March 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., 1997. All rights reserved. Printed in Japan.