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 16-Bit Single-Chip Microcomputer

H8S/2172 Series

H8S/2170 F-ZTAT

HD64F2170

Hardware Manual

ADE-602-316

Rev. 1.0
03/10/03
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.

General Precautions on Handling of Product

1. Treatment of NC Pins

Note: Do not connect anything to the NC pins.
The NC (not connected) pins are either not connected to any of the internal circuitry or are used as test pins or to reduce noise. If something is connected to the NC pins, the operation of the LSI is not guaranteed.

2. Treatment of Unused Input Pins

Note: Fix all unused input pins to high or low level.
Generally, the input pins of CMOS products are high-impedance input pins. If unused pins are in their open states, intermediate levels are induced by noise in the vicinity, a pass-through current flows internally, and a malfunction may occur.

3. Processing before Initialization

Note: When power is first supplied, the product's state is undefined.
The states of internal circuits are undefined until full power is supplied throughout the chip and a low level is input on the reset pin. During the period where the states are undefined, the register settings and the output state of each pin are also undefined. Design your system so that it does not malfunction because of processing while it is in this undefined state. For those products which have a reset function, reset the LSI immediately after the power supply has been turned on.

4. Prohibition of Access to Undefined or Reserved Addresses

Note: Access to undefined or reserved addresses is prohibited.
The undefined or reserved addresses may be used to expand functions, or test registers may have been be allocated to these addresses. Do not access these registers; the system's operation is not guaranteed if they are accessed.

Configuration of This Manual

This manual comprises the following items:

1. General Precautions on Handling of Product
2. Configuration of This Manual
3. Preface
4. Contents
5. Overview
6. Description of Functional Modules
   - CPU and System-Control Modules
   - On-Chip Peripheral Modules

     The configuration of the functional description of each module differs according to the module. However, the generic style includes the following items:

   i) Feature
   ii) Input/Output Pin
   iii) Register Description
   iv) Operation
   v) Usage Note

When designing an application system that includes this LSI, take notes into account. Each section includes notes in relation to the descriptions given, and usage notes are given, as required, as the final part of each section.

7. List of Registers
8. Electrical Characteristics
9. Appendix
10. Main Revisions and Additions in this Edition (only for revised versions)

The list of revisions is a summary of points that have been revised or added to earlier versions. This does not include all of the revised contents. For details, see the actual locations in this manual.

11. Index

Preface

This LSI is a microcomputer (MCU) made up of the H8S/2000 CPU employing Hitachi's original architecture as its core, and the peripheral functions required to configure a system.

The H8S/2000 CPU has an internal 32-bit configuration, sixteen 16-bit general registers, and a simple and optimized instruction set for high-speed operation. The H8S/2000 CPU can handle a 16-Mbyte linear address space.

This LSI is equipped with ROM and RAM memory, direct memory access controller (DMAC) bus master, an 8-bit timer (TMR), a watchdog timer (WDT), a universal serial bus 2 (USB2), a serial communication interface for boot mode (SCI), and I/O ports as on-chip peripheral modules required for system configuration.

A flash memory (F-ZTAT™*) version is available for this LSI's ROM. The F-ZTAT version provides flexibility as it can be reprogrammed in no time to cope with all situations from the early stages of mass production to full-scale mass production. This is particularly applicable to application devices with specifications that will most probably change.

This manual describes this LSI's hardware.

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

Target Users: This manual was written for users who will be using this LSI in the design of application systems. Target users are expected to understand the fundamentals of electrical circuits, logical circuits, and microcomputers.

Objective: This manual was written to explain the hardware functions and electrical characteristics of this LSI to the target users.
Refer to the H8S/2600 Series, H8S/2000 Series Programming Manual for a detailed description of the instruction set.

Notes on reading this manual:

In order to understand the overall functions of the chip

Read the manual according to the contents. This manual can be roughly categorized into parts on the CPU, system control functions, peripheral functions, and electrical characteristics.

In order to understand the details of the CPU's functions

Read the H8S/2600 Series, H8S/2000 Series Programming Manual.

In order to understand the details of a register when its name is known

Read the index that is the final part of the manual to find the page number of the entry on the register. The addresses, bits, and initial values of the registers are summarized in section 17, List of Registers.

Examples:	Register name:	The following notation is used for cases when the same or a similar function, e.g. DMAC or serial communication interface, is implemented on more than one channel: XXX_N (XXX is the register name and N is the channel number)
	Bit order:	The MSB is on the left and the LSB is on the right.
	Number notation:	Binary is B'xxxx, hexadecimal is H'xxxx.
	Signal notation:	An overbar is added to a low-active signal: $\overline{xxxx}$

Related Manuals: The latest versions of all related manuals are available from our web site. Please ensure you have the latest versions of all documents you require. http://www.hitachisemiconductor.com/

H8S/2170 F-ZTAT™ manuals:

Manual Title	ADE No.
H8S/2172 Series H8S/2170 F-ZTAT™ Hardware Manual	This manual
H8S/2600 Series, H8S/2000 Series Programming Manual	ADE-602-083

User's manuals for development tools:

Manual Title	ADE No.
H8S, H8/300 Series C/C++ Compiler, Assembler, Optimizing Linkage Editor User's Manual	ADE-702-247
H8S, H8/300 Series Simulator/Debugger User's Manual	ADE-702-282
H8S, H8/300 Series Hitachi Embedded Workshop, Hitachi Debugging Interface Tutorial	ADE-702-231
Hitachi Embedded Workshop User's Manual	ADE-702-201

Contents

Section 1 Overview 1
1.1 Features 1
1.2 Internal Block Diagram 2
1.3 Pin Description 3
1.3.1 Pin Arrangement 3
1.3.2 Pin Arrangements in Each Mode 4
1.3.3 Pin Functions 8

Section 2 CPU 13
2.1 Features 13
2.1.1 Differences between H8S/2600 CPU and H8S/2000 CPU 14
2.1.2 Differences from H8/300 CPU 15
2.1.3 Differences from H8/300H CPU 15
2.2 CPU Operating Modes 16
2.2.1 Normal Mode 16
2.2.2 Advanced Mode 18
2.3 Address Space 20
2.4 Register Configuration 21
2.4.1 General Registers 22
2.4.2 Program Counter (PC) 23
2.4.3 Extended Control Register (EXR) 23
2.4.4 Condition-Code Register (CCR) 24
2.4.5 Initial Register Values 25
2.5 Data Formats 26
2.5.1 General Register Data Formats 26
2.5.2 Memory Data Formats 28
2.6 Instruction Set 29
2.6.1 Table of Instructions Classified by Function 30
2.6.2 Basic Instruction Formats 39
2.7 Addressing Modes and Effective Address Calculation 40
2.7.1 Register Direct—Rn 41
2.7.2 Register Indirect—@ERn 41
2.7.3 Register Indirect with Displacement—@(d:16, ERn) or @(d:32, ERn) 41
2.7.4 Register Indirect with Post-Increment or Pre-Decrement—@ERn+ or @-ERn 41
2.7.5 Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32 41
2.7.6 Immediate—#xx:8, #xx:16, or #xx:32 42
2.7.7 Program-Counter Relative—@(d:8, PC) or @(d:16, PC) 42
2.7.8 Memory Indirect—@@aa:8 42

2.7.9 Effective Address Calculation 44
2.8 Processing States 46
2.9 Usage Note 47
2.9.1 Note on Bit Manipulation Instructions 47

Section 3 MCU Operating Modes 49
3.1 Operating Mode Selection 49
3.2 Register Descriptions 49
3.2.1 Mode Control Register (MDCR) 50
3.2.2 System Control Register (SYSCR) 51
3.3 Operating Modes 52
3.3.1 Mode 2 52
3.3.2 Pin Functions 52
3.4 Address Map 53

Section 4 Exception Handling 55
4.1 Exception Handling Types and Priority 55
4.2 Exception Sources and Exception Vector Table 56
4.3 Reset 57
4.3.1 Reset Exception Handling 57
4.3.2 Interrupts after Reset 58
4.3.3 On-Chip Peripheral Modules after Reset is Cancelled 58
4.4 Interrupt Exception Handling 59
4.5 Trap Instruction Exception Handling 59
4.6 Stack Status after Exception Handling 60
4.7 Usage Note 60

Section 5 Interrupt Controller 63
5.1 Features 63
5.2 Input/Output Pins 64
5.3 Register Descriptions 64
5.3.1 Interrupt Control Registers A to C (ICRA to ICRC) 64
5.3.2 Address Break Control Register (ABRKCR) 65
5.3.3 Break Address Registers A to C (PBARA to PBARC) 66
5.3.4 IRQ Sense Control Registers H, L (ISCRH, ISCRL) 67
5.3.5 IRQ Enable Register (IER) 68
5.3.6 IRQ Status Register (ISR) 68
5.4 Interrupt Sources 69
5.4.1 External Interrupts 69
5.4.2 Internal Interrupts 70
5.5 Interrupt Exception Handling Vector Table 70
5.6 Interrupt Control Modes and Interrupt Operation 72
5.6.1 Interrupt Control Mode 0 74

5.6.2 Interrupt Control Mode 1 76
5.6.3 Interrupt Exception Handling Sequence 78
5.6.4 Interrupt Response Times 80
5.7 Usage Notes 81
5.7.1 Conflict between Interrupt Generation and Disabling 81
5.7.2 Instructions that Disable Interrupts 82
5.7.3 Interrupts during Execution of EEPMOV Instruction 82

Section 6 Bus Controller (BSC) 83
6.1 Features 83
6.2 Input/Output Pins 85
6.3 Register Descriptions 86
6.3.1 Access Control Register (ACSCR) 87
6.3.2 $\overline{CS}$ Assertion Period Control Register (CSACR) 88
6.3.3 Wait Control Register (WTCR) 90
6.3.4 Bus Control Register (BCR) 92
6.3.5 Read Strobe Timing Control Register (RDNCR) 93
6.3.6 DRAM Control Register (DRAMCR) 94
6.3.7 DRAM Access Control Register (DRACCR) 97
6.3.8 Refresh Control Register (REFCR) 98
6.3.9 Refresh Timer Counter (RTCNT) 101
6.3.10 Refresh Time Constant Register (RTCOR) 101
6.4 Bus Control 101
6.4.1 Area Division 101
6.4.2 Address Map 102
6.4.3 Bus Specifications 104
6.4.4 Memory Interfaces 106
6.4.5 Chip Select Signals 107
6.5 Basic Bus Interface 108
6.5.1 Data Size and Data Alignment 108
6.5.2 Valid Strobes 109
6.5.3 Basic Timing 109
6.5.4 Wait Control 118
6.5.5 Read Strobe ($\overline{RD}$) Timing 119
6.5.6 Extension of Chip Select ($\overline{CS}$) Assertion Period 120
6.6 DRAM Interface 121
6.6.1 Setting DRAM Space 121
6.6.2 Address Multiplexing 121
6.6.3 Data Bus 122
6.6.4 Pins Used for DRAM Interface 122
6.6.5 Basic Timing 123
6.6.6 Column Address Output Cycle Control 124

6.6.7 Row Address Output State Control 125
6.6.8 Precharge State Control 127
6.6.9 Wait Control 128
6.6.10 Byte Access Control 130
6.6.11 Burst Operation 131
6.6.12 Refresh Control 136
6.6.13 DMAC Single Address Transfer Mode and DRAM Interface 139
6.7 Idle Cycle 141
6.7.1 Operation 141
6.7.2 Pin States in Idle Cycle 148
6.8 Write Data Buffer Function 149
6.9 Bus Arbitration 150
6.9.1 Operation 150
6.9.2 Bus Transfer Timing 150
6.10 Bus Controller Operation in Reset 151

Section 7 DMA Controller (DMAC) 153
7.1 Features 153
7.2 Input/Output Pins 155
7.3 Register Descriptions 157
7.3.1 DMA Source Address Register (DMSAR) 158
7.3.2 DMA Destination Address Register (DMDAR) 158
7.3.3 DMA Transfer Count Register (DMTCR) 158
7.3.4 DMA Mode Control Register (DMMDR) 160
7.3.5 DMA Address Control Register (DMACR) 165
7.3.6 USB Transfer Control Register (USTCR) 169
7.4 Operation 171
7.4.1 Transfer Modes 171
7.4.2 Address Modes (Dual Address Mode/Single Address Mode) 172
7.4.3 DMA Transfer Requests (Auto Request Mode/External Request Mode/ USB Transfer Request) 176
7.4.4 Bus Modes (Cycle Steal Mode/Burst Mode) 177
7.4.5 Transfer Modes (Normal Transfer Mode/Block Transfer Mode) 178
7.4.6 Repeat Area Function 179
7.4.7 Registers during DMA Transfer Operation 182
7.4.8 Channel Priority 186
7.4.9 DMAC Bus Cycles (Dual Address Mode) 189
7.4.10 DMAC Bus Cycles (Single Address Mode) 196
7.4.11 Examples of Operation Timing in Each Mode 201
7.4.12 Ending DMA Transfer 212
7.4.13 Relationship between DMAC and Other Bus Masters 213
7.5 Interrupt Sources 213
7.6 Usage Notes 216

Section 8 I/O Ports 219
8.1 Port 1 222
8.1.1 Port 1 Data Direction Register (P1DDR) 222
8.1.2 Port 1 Data Register (P1DR) 223
8.1.3 Port 1 Register (PORT1) 223
8.1.4 Pin Functions 224
8.2 Port 2 226
8.2.1 Port 2 Data Direction Register (P2DDR) 226
8.2.2 Port 2 Data Register (P2DR) 226
8.2.3 Port 2 Register (PORT2) 227
8.2.4 Pin Functions 227
8.3 Port 3 230
8.3.1 Port 3 Data Direction Register (P3DDR) 230
8.3.2 Port 3 Data Register (P3DR) 230
8.3.3 Port 3 Register (PORT3) 231
8.3.4 Pin Functions 231
8.4 Port 4 233
8.4.1 Port 4 Data Direction Register (P4DDR) 233
8.4.2 Port 4 Data Register (P4DR) 234
8.4.3 Port 4 Register (PORT4) 234
8.4.4 Pin Functions 235
8.5 Port 5 237
8.5.1 Port 5 Data Direction Register (P5DDR) 237
8.5.2 Port 5 Data Register (P5DR) 237
8.5.3 Port 5 Register (PORT5) 238
8.5.4 Pin Functions 238
8.6 Port 6 242
8.6.1 Port 6 Data Direction Register (P6DDR) 242
8.6.2 Port 6 Data Register (P6DR) 242
8.6.3 Port 6 Register (PORT6) 243
8.6.4 Pin Functions 243
8.7 Port 7 246
8.7.1 Port 7 Data Direction Register (P7DDR) 246
8.7.2 Port 7 Data Register (P7DR) 246
8.7.3 Port 7 Register (PORT7) 247
8.7.4 Pin Functions 247
8.8 Port 8 251
8.8.1 Port 8 Data Direction Register (P8DDR) 251
8.8.2 Port 8 Data Register (P8DR) 252
8.8.3 Port 8 Register (PORT8) 252
8.8.4 Pin Functions 253
8.9 Port 9 255

8.9.1 Port 9 Data Direction Register (P9DDR) 255
8.9.2 Port 9 Data Register (P9DR) 255
8.9.3 Port 9 Register (PORT9) 256
8.9.4 Pin Functions 256
8.10 Port A 258
8.10.1 Port A Data Direction Register (PADDR) 258
8.10.2 Port A Data Register (PADR) 259
8.10.3 Port A Register (PORTA) 259
8.10.4 Pin Functions 260
8.11 Pin Selection 261
8.11.1 Port Function Control Register 1 (PFCR1) 261
8.11.2 IRQ Sense Port Select Register (ISSR) 263

Section 9 8-Bit Timer (TMR) 265
9.1 Features 265
9.2 Input/Output Pins 267
9.3 Register Descriptions 267
9.3.1 Timer Counter (TCNT) 267
9.3.2 Time Constant Register A (TCORA) 268
9.3.3 Time Constant Register B (TCORB) 268
9.3.4 Timer Control Register (TCR) 268
9.3.5 Timer Control/Status Register (TCSR) 270
9.4 Operation 272
9.4.1 Pulse Output 272
9.5 Operation Timing 273
9.5.1 TCNT Incrementation Timing 273
9.5.2 Timing of CMFA and CMFB Setting when Compare-Match Occurs 274
9.5.3 Timing of Timer Output when Compare-Match Occurs 274
9.5.4 Timing of Compare Match Clear 275
9.5.5 Timing of TCNT External Reset 275
9.5.6 Timing of Overflow Flag (OVF) Setting 275
9.6 Operation with Cascaded Connection 276
9.6.1 16-Bit Counter Mode 276
9.6.2 Compare Match Count Mode 277
9.7 Interrupt Sources 277
9.7.1 Interrupt Sources 277
9.8 Usage Notes 278
9.8.1 Contention between TCNT Write and Clear 278
9.8.2 Contention between TCNT Write and Increment 279
9.8.3 Contention between TCOR Write and Compare Match 280
9.8.4 Contention between Compare Matches A and B 281
9.8.5 Switching of Internal Clocks and TCNT Operation 281
9.8.6 Mode Setting with Cascaded Connection 283

9.8.7 Interrupts in Module Stop Mode 283

Section 10 Watchdog Timer (WDT) 285
10.1 Features 285
10.2 Register Descriptions 286
10.2.1 Timer Counter (TCNT) 286
10.2.2 Timer Control/Status Register (TCSR) 286
10.3 Operation 288
10.3.1 Watchdog Timer Mode 288
10.3.2 Interval Timer Mode 289
10.3.3 Watchdog Timer Overflow Flag (OVF) Timing 289
10.4 Interrupt Sources 290
10.5 Usage Notes 290
10.5.1 Notes on Register Access 290
10.5.2 Conflict between Timer Counter (TCNT) Write and Increment 291
10.5.3 Changing Values of CKS2 to CKS0 Bits 292
10.5.4 Switching between Watchdog Timer Mode and Interval Timer Mode 292

Section 11 Serial Communication Interface for Boot Mode (SCI) 293
11.1 Features 293
11.2 Input/Output Pins 294
11.3 Register Descriptions 295
11.3.1 Receive Shift Register (RSR) 295
11.3.2 Receive Data Register (RDR) 295
11.3.3 Transmit Data Register (TDR) 295
11.3.4 Transmit Shift Register (TSR) 296
11.3.5 Serial Mode Register (SMR) 296
11.3.6 Serial Control Register (SCR) 297
11.3.7 Serial Status Register (SSR) 299
11.3.8 Bit Rate Register (BRR) 302
11.4 Operation in Asynchronous Mode 306
11.4.1 Data Transfer Format 307
11.4.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode 308
11.4.3 Clock 309
11.4.4 SCI Initialization (Asynchronous Mode) 309
11.4.5 Data Transmission (Asynchronous Mode) 310
11.4.6 Serial Data Reception (Asynchronous Mode) 312
11.5 Interrupt Sources 316
11.5.1 Interrupts in Normal Serial Communication Interface Mode 316
11.6 Usage Notes 317
11.6.1 Module Stop Mode Setting 317
11.6.2 Relation between Writes to TDR and the TDRE Flag 317

11.6.3 Operation in Case of Mode Transition 317

Section 12 Universal Serial Bus 2 (USB2) 321
12.1 Features 321
12.2 Input/Output Signals 323
12.3 Register Descriptions 324
12.3.1 Interrupt Flag Register 0 (IFR0) 325
12.3.2 Interrupt Select Register 0 (ISR0) 329
12.3.3 Interrupt Enable Register 0 (IER0) 330
12.3.4 EP0o Receive Data Size Register (EPSZ0o) 330
12.3.5 EP1 Receive Data Size Register (EPSZ1) 331
12.3.6 EP0i Data Register (EPDR0i) 331
12.3.7 EP0o Data Register (EPDR0o) 331
12.3.8 EP0s Data Register (EPDR0s) 332
12.3.9 EP1 Data Register (EPDR1) 332
12.3.10 EP2 Data Register (EPDR2) 332
12.3.11 EP3 Data Register (EPDR3) 333
12.3.12 Data Status Register 0 (DASTS0) 333
12.3.13 Packet Enable Register 0i (PKTE0i) 334
12.3.14 Packet Enable Register 2 (PKTE2) 334
12.3.15 Packet Enable Register 3 (PKTE3) 334
12.3.16 FIFO Clear Register 0 (FCLR0) 335
12.3.17 Endpoint Stall Register 0 (EPSTL0) 337
12.3.18 DMA Set Register 0 (DMA0) 337
12.3.19 Control Register (CTRL) 338
12.3.20 Port Function Control Register 3 (PFCR3) 339
12.3.21 USB Suspend Status Register (USBSUSP) 339
12.4 Interrupt Pins 341
12.4.1 USBI0 Interrupt 341
12.4.2 USBI1 Interrupt 341
12.5 Communication Operation 342
12.5.1 USB Cable Connection 342
12.5.2 USB Cable Disconnection 343
12.5.3 Control Transfer 344
12.5.4 EP1 Bulk-Out Transfer (Dual FIFO) 350
12.5.5 EP2 Bulk-In Transfer (Dual FIFO) 352
12.5.6 EP3 Interrupt-In Transfer 354
12.5.7 Processing of USB Standard Requests and Class/Vendor Requests 355
12.5.8 Stall Operations 356
12.5.9 Tree Configuration 359
12.5.10 Power Supply Specification 359
12.6 Notes on Using DMA 359
12.7 Transition to USB Suspend Mode 360

12.7.1 Suspend Signal Output 360
12.7.2 Software Standby in Suspend Mode 363
12.8 Usage Notes 365
12.8.1 Setup Data Reception 365
12.8.2 FIFO Clear 365
12.8.3 Operating Frequency 366
12.8.4 Interrupts 366
12.8.5 Register Access Size 366
12.8.6 Data Register Overread or Overwrite 366
12.8.7 EP0 Interrupt Sources Assignment 366
12.8.8 FIFO Size at Full Speed 367
12.8.9 Level Shifter for VBUS Pin 367
12.8.10 USB 2.0 Transceiver (Physical Layer) 367
12.8.11 EPDR0s Read 367
12.8.12 USB Bus Idle in High-Speed Mode 368
12.8.13 Note on USB Bus Disconnection 368
12.8.14 Example of External Circuit 368
12.8.15 External Physical Layer LSI 369
12.8.16 Operation at the Bus Reset Reception 369
12.8.17 Usage Notes in Control IN Transfer 370

Section 13 RAM 371

Section 14 Flash Memory (0.18-µm F-ZTAT Version) 373
14.1 Features 373
14.1.1 Operating Mode 375
14.1.2 Mode Comparison 376
14.1.3 Flash MAT Configuration 377
14.1.4 Block Division 378
14.1.5 Programming/Erasing Interface 379
14.2 Input/Output Pins 381
14.3 Register Descriptions 381
14.3.1 Programming/Erasing Interface Register 382
14.3.2 Programming/Erasing Interface Parameter 391
14.4 On-Board Programming Mode 401
14.4.1 Boot Mode 401
14.4.2 User Program Mode 405
14.4.3 User Boot Mode 415
14.4.4 Procedure Program and Storable Area for Programming Data 418
14.5 Protection 428
14.5.1 Hardware Protection 428
14.5.2 Software Protection 429

14.5.3 Error Protection 429
14.6 Switching between User MAT and User Boot MAT 430
14.7 Flash Memory Emulation in RAM 431
14.7.1 Emulation in RAM 431
14.7.2 RAM Overlap 433
14.8 Programmer Mode 434
14.9 Serial Communication Interface Specification for Boot Mode 435
14.10 Usage Notes 460

Section 15 Clock Pulse Generator 463
15.1 Oscillator 463
15.1.1 Connecting Crystal Resonator 463
15.1.2 External Clock Input 464
15.2 PLL Circuit 467
15.3 Usage Notes 467
15.3.1 Notes on Resonator 467
15.3.2 Notes on Board Design 467

Section 16 Power-Down Modes 469
16.1 Register Descriptions 472
16.1.1 Standby Control Register (SBYCR) 472
16.1.2 Module Stop Control Registers H and L (MSTPCRH, MSTPCRL) 473
16.2 Operation 474
16.2.1 Sleep Mode 474
16.2.2 Software Standby Mode 474
16.2.3 Hardware Standby Mode 476
16.2.4 Module Stop Mode 477
16.3 Usage Notes 478
16.3.1 I/O Port Status 478
16.3.2 Current Consumption during Oscillation Stabilization Standby Period 478
16.3.3 On-Chip Peripheral Module Interrupts 478
16.3.4 Writing to MSTPCR 478

Section 17 List of Registers 479
17.1 Register Addresses (Address Order) 480
17.2 Register Bits 485
17.3 Register States in Each Operating Mode 493

Section 18 Electrical Characteristics 497
18.1 Absolute Maximum Ratings 497
18.2 DC Characteristics 498
18.3 AC Characteristics 500
18.3.1 Clock Timing 501

18.3.2 Control Signal Timing 503
18.3.3 Bus Timing 504
18.3.4 DMAC Timing 515
18.3.5 Timing of On-Chip Peripheral Modules 518
18.4 Flash Memory Characteristics 521
18.4.1 Flash Memory Characteristics 521
18.5 Use Note (Internal Voltage Step Down) 521

Appendix 523
A. Port States in Each Processing State 523
B. Product Lineup 526
C. Package Dimensions 527

Index 529

Figures

Section 1 Overview
Figure 1.1 Internal Block Diagram 2
Figure 1.2 Pin Arrangement (TFP-100B) 3

Section 2 CPU
Figure 2.1 Exception Vector Table (Normal Mode) 17
Figure 2.2 Stack Structure in Normal Mode 17
Figure 2.3 Exception Vector Table (Advanced Mode) 18
Figure 2.4 Stack Structure in Advanced Mode 19
Figure 2.5 Memory Map 20
Figure 2.6 CPU Internal Registers 21
Figure 2.7 Usage of General Registers 22
Figure 2.8 Stack 23
Figure 2.9 General Register Data Formats (1) 26
Figure 2.9 General Register Data Formats (2) 27
Figure 2.10 Memory Data Formats 28
Figure 2.11 Instruction Formats (Examples) 40
Figure 2.12 Branch Address Specification in Memory Indirect Addressing Mode 43
Figure 2.13 State Transitions 47

Section 3 MCU Operating Modes
Figure 3.1 Address Map 53

Section 4 Exception Handling
Figure 4.1 Reset Sequence 58
Figure 4.2 Stack Status after Exception Handling 60
Figure 4.3 Operation when SP Value is Odd 61

Section 5 Interrupt Controller
Figure 5.1 Block Diagram of Interrupt Controller 63
Figure 5.2 Block Diagram of Interrupts IRQ7 to IRQ0 69
Figure 5.3 Block Diagram of Interrupt Control Operation 72
Figure 5.4 Flowchart of Procedure up to Interrupt Acceptance in Interrupt Control Mode 0 75
Figure 5.5 State Transition in Interrupt Control Mode 1 76
Figure 5.6 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 1 78
Figure 5.7 Interrupt Exception Handling 79
Figure 5.8 Conflict between Interrupt Generation and Disabling 81

Section 6 Bus Controller (BSC)

Figure 6.1 Block Diagram of Bus Controller 84
Figure 6.2 CS and Address Assertion Period Extension (Example of 3-State Access Space and RDNn = 0) 89
Figure 6.3 Read Strobe Negation Timing (Example of 3-State Access Space) 93
Figure 6.4 RAS Signal Assertion Timing (2-State Column Address Output Cycle, Full Access) 97
Figure 6.5 Area Divisions 102
Figure 6.6 Address Format 102
Figure 6.7 Address Map 104
Figure 6.8 CSn Signal Output Timing (n = 3 to 0) 107
Figure 6.9 Access Sizes and Data Alignment Control (8-Bit Access Space) 108
Figure 6.10 Access Sizes and Data Alignment Control (16-bit Access Space) 109
Figure 6.11 Bus Timing for 8-Bit, 2-State Access Space 110
Figure 6.12 Bus Timing for 8-Bit, 3-State Access Space 111
Figure 6.13 Bus Timing for 16-Bit, 2-State Access Space (Even Address Byte Access) 112
Figure 6.14 Bus Timing for 16-Bit, 2-State Access Space (Odd Address Byte Access) 113
Figure 6.15 Bus Timing for 16-Bit, 2-State Access Space (Word Access) 114
Figure 6.16 Bus Timing for 16-Bit, 3-State Access Space (Even Address Byte Access) 115
Figure 6.17 Bus Timing for 16-Bit, 3-State Access Space (Odd Address Byte Access) 116
Figure 6.18 Bus Timing for 16-Bit, 3-State Access Space (Word Access) 117
Figure 6.19 Example of Wait State Insertion Timing 118
Figure 6.20 Example of Read Strobe Timing 119
Figure 6.21 Example of Timing when Chip Select Assertion Period is Extended 120
Figure 6.22 DRAM Basic Access Timing (RAST = 0, CAST = 0) 123
Figure 6.23 Example of Access Timing with 3-State Column Address Output Cycle (RAST = 0) 124
Figure 6.24 Example of Access Timing when RAS Signal Goes Low from Beginning of T_r State (CAST = 0) 125
Figure 6.25 Example of Timing with One Row Address Output Hold State (RAST = 0, CAST = 0) 126
Figure 6.26 Example of Timing with Two-State Precharge Cycle (RAST = 0, CAST = 0) 127
Figure 6.27 Example of Wait State Insertion Timing (2-State Column Address Output) 128
Figure 6.28 Example of Wait State Insertion Timing (3-State Column Address Output) 129
Figure 6.29 2-CAS Control Timing (Write Access to Even Address: RAST = 0, CAST = 0) ... 130
Figure 6.30 Example of 2-CAS DRAM Connection 131
Figure 6.31 Operation Timing in Fast Page Mode (RAST = 0, CAST = 0) 132
Figure 6.32 Operation Timing in Fast Page Mode (RAST = 0, CAST = 1) 132
Figure 6.33 Example of Operation Timing in RAS Down Mode (RAST = 0, CAST = 0) 134
Figure 6.34 Example of Idle Cycle Insertion when RAS Down Mode cannot be Continued 134
Figure 6.35 Example of Operation Timing in RAS Up Mode (RAST = 0, CAST = 0) 135
Figure 6.36 RTCNT Operation 136
Figure 6.37 Compare Match Timing 137
Figure 6.38 CBR Refresh Timing 137

Figure 6.39 CBR Refresh Timing (RCW1 = 0, RCW0 = 1, RLW1 = 0, RLW0 = 0) 137
Figure 6.40 Self-Refresh Timing 138
Figure 6.41 Example of Timing when Precharge Time after Self-Refreshing is Extended by 2 States 139
Figure 6.42 Example of DACK Output Timing when DDS = 1 (RAST = 0, CAST = 0) 140
Figure 6.43 Example of DACK Output Timing when DDS = 0 (RAST = 0, CAST = 1) 141
Figure 6.44 Example of Idle Cycle Operation (Consecutive Reads in Different Areas) 142
Figure 6.45 Example of Idle Cycle Operation (Write after Read) 143
Figure 6.46 Example of Idle Cycle Operation (Read after Write) 144
Figure 6.47 Relationship between Chip Select (CS) and Read (RD) 144
Figure 6.48 Example of DRAM Full Access after External Read (CAST = 0) 145
Figure 6.49 Example of Idle Cycle Operation in RAS Down Mode (Consecutive Reads in Different Areas) (IDLE1 = 0, IDLE0 = 1, IDLC1 = 0, IDLC0 = 1, RAST = 0, and CAST = 0) 146
Figure 6.50 Example of Timing when Write Data Buffer Function is Used 149

Section 7 DMA Controller (DMAC)

Figure 7.1 Block Diagram of DMAC 155
Figure 7.2 Example of Timing in Dual Address Mode 173
Figure 7.3 Data Flow in Single Address Mode 174
Figure 7.4 Example of Timing in Single Address Mode 175
Figure 7.5 Example of Timing in Cycle Steal Mode 177
Figure 7.6 Examples of Timing in Burst Mode 178
Figure 7.7 Examples of Timing in Normal Transfer Mode 178
Figure 7.8 Example of Timing in Block Transfer Mode 179
Figure 7.9 Example of Repeat Area Function Operation 180
Figure 7.10 Example of Repeat Area Function Operation in Block Transfer Mode 181
Figure 7.11 DMTCR Update Operations in Normal Transfer Mode and Block Transfer Mode 184
Figure 7.12 Procedure for Changing Register Settings in Operating Channel 185
Figure 7.13 Example of Channel Priority Timing 187
Figure 7.14 Examples of Channel Priority Timing 188
Figure 7.15 Example of Normal Transfer Mode (Cycle Steal Mode) Transfer 189
Figure 7.16 Example of Normal Transfer Mode (Burst Mode) Transfer 190
Figure 7.17 Example of Normal Transfer Mode (Cycle Steal Mode) Transfer (Transfer Source: USB) 190
Figure 7.18 Example of Normal Transfer Mode (Cycle Steal Mode) Transfer (Transfer Destination: USB) 191
Figure 7.19 Example of Block Transfer Mode (Cycle Steal Mode) Transfer 191
Figure 7.20 Example of Normal Mode Transfer Activated by DREQ Pin Falling Edge 192
Figure 7.21 Example of Block Transfer Mode Transfer Activated by DREQ Pin Falling Edge 193

Figure 7.22 Example of Normal Mode Transfer Activated by DREQ Pin Low Level 194
Figure 7.23 Example of Block Transfer Mode Transfer Activated by DREQ Pin Low Level ... 195
Figure 7.24 Example of Single Address Mode (Byte Read) Transfer 196
Figure 7.25 Example of Single Address Mode (Word Read) Transfer 196
Figure 7.26 Example of Single Address Mode (Longword Read) Transfer 197
Figure 7.27 Example of Single Address Mode (Byte Write) Transfer 197
Figure 7.28 Example of Single Address Mode (Word Write) Transfer 198
Figure 7.29 Example of Single Address Mode (Longword Write) Transfer 198
Figure 7.30 Example of Single Address Mode Transfer Activated by DREQ Pin Falling Edge 199
Figure 7.31 Example of Single Address Mode Transfer Activated by DREQ Pin Low Level ... 200
Figure 7.32 Auto Request/Cycle Steal Mode/Normal Transfer Mode (No Contention/Dual Address Mode) 201
Figure 7.33 Auto Request/Cycle Steal Mode/Normal Transfer Mode (CPU Cycles/Single Address Mode) 202
Figure 7.34 Auto Request/Cycle Steal Mode/Normal Transfer Mode (Contention with Another Channel/Single Address Mode) 202
Figure 7.35 Auto Request/Burst Mode/Normal Transfer Mode (CPU Cycles/Dual Address Mode) 203
Figure 7.36 Auto Request/Burst Mode/Normal Transfer Mode (CPU Cycles/Single Address Mode) 203
Figure 7.37 Auto Request/Burst Mode/Normal Transfer Mode (Contention with Another Channel/Single Address Mode) 204
Figure 7.38 External Request/Cycle Steal Mode/Normal Transfer Mode (No Contention/Dual Address Mode/Low Level Sensing) 205
Figure 7.39 External Request/Cycle Steal Mode/Normal Transfer Mode (CPU Cycles/Single Address Mode/Low Level Sensing) 205
Figure 7.40 External Request/Cycle Steal Mode/Normal Transfer Mode (No Contention/Single Address Mode/Falling Edge Sensing) 206
Figure 7.41 External Request/Cycle Steal Mode/Normal Transfer Mode (Contention with Another Channel/Dual Address Mode/Low Level Sensing) 206
Figure 7.42 External Request/Cycle Steal Mode/Block Transfer Mode (No Contention/Dual Address Mode/Low Level Sensing) 208
Figure 7.43 External Request/Cycle Steal Mode/Block Transfer Mode (No Contention/Single Address Mode/Falling Edge Sensing) 209
Figure 7.44 External Request/Cycle Steal Mode/Block Transfer Mode (CPU Cycles/Single Address Mode/Low Level Sensing) 210
Figure 7.45 External Request/Cycle Steal Mode/Block Transfer Mode (Contention with Another Channel/Dual Address Mode/Low Level Sensing) 211
Figure 7.46 Transfer End Interrupt Logic 214
Figure 7.47 Example of Procedure for Restarting Transfer on Channel in which Transfer End Interrupt Occurred 215

Section 9 8-Bit Timer (TMR)

Figure 9.1 Block Diagram of 8-Bit Timer Module 266
Figure 9.2 Example of Pulse Output 272
Figure 9.3 Count Timing for Internal Clock Input 273
Figure 9.4 Count Timing for External Clock Input 273
Figure 9.5 Timing of CMF Setting 274
Figure 9.6 Timing of Timer Output 274
Figure 9.7 Timing of Compare Match Clear 275
Figure 9.8 Timing of Clearance by External Reset 275
Figure 9.9 Timing of OVF Setting 276
Figure 9.10 Contention between TCNT Write and Clear 278
Figure 9.11 Contention between TCNT Write and Increment 279
Figure 9.12 Contention between TCOR Write and Compare Match 280

Section 10 Watchdog Timer (WDT)

Figure 10.1 Block Diagram of WDT 285
Figure 10.2 Watchdog Timer Mode (RST/NMI = 1) Operation 288
Figure 10.3 Interval Timer Mode Operation 289
Figure 10.4 OVF Flag Set Timing 289
Figure 10.5 Output Timing of OVF 290
Figure 10.6 Writing to TCNT and TCSR 291
Figure 10.7 Conflict between TCNT Write and Increment 291

Section 11 Serial Communication Interface for Boot Mode (SCI)

Figure 11.1 Block Diagram of SCI 294
Figure 11.2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits) 306
Figure 11.3 Receive Data Sampling Timing in Asynchronous Mode 308
Figure 11.4 Sample SCI Initialization Flowchart 309
Figure 11.5 Example of Operation in Transmission in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit) 310
Figure 11.6 Sample Serial Transmission Flowchart 311
Figure 11.7 Example of SCI Operation in Reception (Example with 8-Bit Data, Parity, One Stop Bit) 312
Figure 11.8 Sample Serial Reception Data Flowchart (1) 314
Figure 11.8 Sample Serial Reception Data Flowchart (2) 315
Figure 11.9 Sample Flowchart for Mode Transition during Transmission 318
Figure 11.10 Sample Flowchart for Mode Transition during Reception 319

Section 12 Universal Serial Bus 2 (USB2)

Figure 12.1 Block Diagram of USB2 322
Figure 12.2 USB Cable Connection 342

Figure 12.3 USB Cable Disconnection 343
Figure 12.4 Control Transfer Stage Configuration 344
Figure 12.5 Setup Stage Operation 345
Figure 12.6 Data Stage Operation (Control-In) 346
Figure 12.7 Data Stage Operation (Control-Out) 347
Figure 12.8 Status Stage Operation (Control-In) 348
Figure 12.9 Status Stage Operation (Control-Out) 349
Figure 12.10 EP1 Bulk-Out Transfer Operation 351
Figure 12.11 EP2 Bulk-In Transfer Operation 353
Figure 12.12 EP3 Interrupt-In Transfer Operation 354
Figure 12.13 Forcible Stall by Firmware 357
Figure 12.14 Automatic Stall by USB Function Module 358
Figure 12.15 PKTE2 Operation for EP2 360
Figure 12.16 Enter/Recover Sequence of USB Suspend State 361
Figure 12.17 Enter/Recover Sequence of USB Suspend State and Software Standby Mode 363
Figure 12.18 Connection Example of External Circuit 369
Figure 12.19 Bus Reset Following Completion of First Bus Reset 369

Section 14 Flash Memory (0.18-µm F-ZTAT Version)

Figure 14.1 Block Diagram of Flash Memory 374
Figure 14.2 Mode Transition of Flash Memory 375
Figure 14.3 Flash Memory Configuration 377
Figure 14.4 Block Division of User MAT 378
Figure 14.5 Overview of User Procedure Program 379
Figure 14.6 System Configuration in Boot Mode 402
Figure 14.7 Automatic-Bit-Rate Adjustment Operation of SCI 402
Figure 14.8 Overview of Boot Mode State Transition Diagram 404
Figure 14.9 Programming/Erasing Overview Flow 405
Figure 14.10 RAM Map When Programming/Erasing is Executed 406
Figure 14.11 Programming Procedure 407
Figure 14.12 Erasing Procedure 412
Figure 14.13 Repeating Procedure of Erasing and Programming 414
Figure 14.14 Procedure for Programming User MAT in User Boot Mode 416
Figure 14.15 Procedure for Erasing User MAT in User Boot Mode 417
Figure 14.16 Transitions to Error-Protection State 430
Figure 14.17 Switching between the User MAT and User Boot MAT 431
Figure 14.18 Flowchart for Flash Memory Emulation in RAM 432
Figure 14.19 Example of RAM Overlap Operation (256-kbyte Flash Memory) 433
Figure 14.20 Memory Map in Programmer Mode 434
Figure 14.21 Boot Program States 436
Figure 14.22 Bit-Rate-Adjustment Sequence 437
Figure 14.23 Communication Protocol Format 438
Figure 14.24 New Bit-Rate Selection Sequence 448

Figure 14.25 Programming Sequence 451
Figure 14.26 Erasure Sequence 454

Section 15 Clock Pulse Generator

Figure 15.1 Block Diagram of Clock Pulse Generator 463
Figure 15.2 Connection of Crystal Resonator (Example) 464
Figure 15.3 Crystal Resonator Equivalent Circuit 464
Figure 15.4 External Clock Input (Examples) 465
Figure 15.5 External Clock Input Timing 466
Figure 15.6 Timing of External Clock Output Stabilization Delay Time 466
Figure 15.7 Note on Board Design for Oscillation Circuit 467

Section 16 Power-Down Modes

Figure 16.1 Mode Transitions 471
Figure 16.2 Software Standby Mode Application Example 476
Figure 16.3 Hardware Standby Mode Timing 477

Section 18 Electrical Characteristics

Figure 18.1 Sample of Dalington Transistor Drive Circuit 500
Figure 18.2 Output Load Circuit 500
Figure 18.3 System Clock Timing 501
Figure 18.4 Oscillation Stabilization Timing (1) 502
Figure 18.5 Oscillation Stabilization Timing (2) 502
Figure 18.6 Reset Input Timing 503
Figure 18.7 Interrupt Input Timing 503
Figure 18.8 Basic Bus Timing: Two-State Access 506
Figure 18.9 Basic Bus Timing: Three-State Access 507
Figure 18.10 Basic Bus Timing: Two-State Access (CS Assertion Period Extended) 508
Figure 18.11 Basic Bus Timing: Three-State Access (CS Assertion Period Extended) 509
Figure 18.12 DRAM Access Timing: Two-State Access 510
Figure 18.13 DRAM Access Timing: Two-State Burst Access 511
Figure 18.14 DRAM Access Timing: Three-State Access (RAST = 1) 512
Figure 18.15 DRAM Access Timing: Three-State Burst Access 513
Figure 18.16 CAS-Before-RAS Refresh Timing 514
Figure 18.17 CAS-Before-RAS Refresh Timing (with Wait Cycle Insertion) 514
Figure 18.18 Self-Refresh Timing (Return from Software Standby Mode: RAST = 0) 514
Figure 18.19 Self-Refresh Timing (Return from Software Standby Mode: RAST = 1) 515
Figure 18.20 DMAC Single Address Transfer Timing: Two-State Access 516
Figure 18.21 DMAC Single Address Transfer Timing: Three-State Access 517
Figure 18.22 DMAC TEND Output Timing 517
Figure 18.23 DMAC DREQ Input Timing 518
Figure 18.24 I/O Port Input/Output Timing 519

Figure 18.25 8-Bit Timer Output Timing 519
Figure 18.26 8-Bit Timer Clock Input Timing 519
Figure 18.27 8-Bit Timer Reset Input Timing 519
Figure 18.28 USB2 Input/output Timing 520
Figure 18.29 VCL Capacitor Connection Method 522

Appendix

Figure C.1 Package Dimensions (TFP-100B) 527

Tables

Section 1 Overview
Table 1.1 Pin Arrangements in Each Mode 4
Table 1.2 Pin Functions 8

Section 2 CPU
Table 2.1 Instruction Classification 29
Table 2.2 Operation Notation 30
Table 2.3 Data Transfer Instructions 31
Table 2.4 Arithmetic Operations Instructions (1) 32
Table 2.4 Arithmetic Operations Instructions (2) 33
Table 2.5 Logic Operations Instructions 34
Table 2.6 Shift Instructions 34
Table 2.7 Bit Manipulation Instructions (1) 35
Table 2.7 Bit Manipulation Instructions (2) 36
Table 2.8 Branch Instructions 37
Table 2.9 System Control Instructions 38
Table 2.10 Block Data Transfer Instructions 39
Table 2.11 Addressing Modes 40
Table 2.12 Absolute Address Access Ranges 42
Table 2.13 Effective Address Calculation (1) 44
Table 2.13 Effective Address Calculation (2) 45

Section 3 MCU Operating Modes
Table 3.1 MCU Operating Mode Selection 49
Table 3.2 Pin Functions in Operating Mode 2 52

Section 4 Exception Handling
Table 4.1 Exception Types and Priority 55
Table 4.2 Exception Handling Vector Table 56
Table 4.3 Status of CCR after Trap Instruction Exception Handling 59

Section 5 Interrupt Controller
Table 5.1 Pin Configuration 64
Table 5.2 Correspondence between Interrupt Source and ICR 65
Table 5.3 Interrupt Sources, Vector Addresses, and Interrupt Priorities 70
Table 5.4 Interrupt Control Modes 72
Table 5.5 Interrupts Acceptable in Each Interrupt Control Mode 73
Table 5.6 Operations and Control Signal Functions in Each Interrupt Control Mode 73
Table 5.7 Interrupt Response Times 80
Table 5.8 Number of States in Interrupt Handling Routine Execution Status 80

Section 6 Bus Controller (BSC)
Table 6.1 Pin Configuration 85
Table 6.2 Address Map 103
Table 6.3 Bus Specifications for Each Area (Basic Bus Interface) 105
Table 6.4 Data Buses Used and Valid Strobes 109
Table 6.5 Relation between Settings of Bits MXC2 to MXC0 and Address Multiplexing 121
Table 6.6 DRAM Interface Pins 122
Table 6.7 Idle Cycles in Mixed Accesses to Normal Space and DRAM Space 147
Table 6.8 Pin States in Idle Cycle 148

Section 7 DMA Controller (DMAC)
Table 7.1 Pin Configuration 156
Table 7.2 DMAC Transfer Modes 171
Table 7.3 DMAC Channel Priority 186
Table 7.4 Interrupt Sources and Priority Order 213

Section 8 I/O Ports
Table 8.1 Port Functions 220

Section 9 8-Bit Timer (TMR)
Table 9.1 Pin Configuration 267
Table 9.2 Clock Input to TCNT and Count Condition 270
Table 9.3 8-Bit Timer Interrupt Sources 277
Table 9.4 Timer Output Priorities 281
Table 9.5 Switching of Internal Clock and TCNT Operation 282

Section 10 Watchdog Timer (WDT)
Table 10.1 WDT Interrupt Source 290

Section 11 Serial Communication Interface for Boot Mode (SCI)
Table 11.1 Pin Configuration 294
Table 11.2 Relationships between N Setting in BRR and Bit Rate B 302
Table 11.3 BRR Settings for Various Bit Rates (Asynchronous Mode) (1) 303
Table 11.3 BRR Settings for Various Bit Rates (Asynchronous Mode) (2) 304
Table 11.4 Maximum Bit Rate for Each Frequency (Asynchronous Mode) 305
Table 11.5 Serial Transfer Formats (Asynchronous Mode) 307
Table 11.6 SSR Status Flags and Receive Data Handling 313
Table 11.7 SCI Interrupt Sources 316

Section 12 Universal Serial Bus 2 (USB2)
Table 12.1 Input/Output Signals 323
Table 12.2 Request Decoding by Firmware 355

Table 12.3 Tree Configuration 359
Table 12.4 FIFO Size in Each Transfer Mode 367

Section 14 Flash Memory (0.18-µm F-ZTAT Version)
Table 14.1 Comparison of Programming Modes 376
Table 14.2 Pin Configuration 381
Table 14.3 Register/Parameter and Target Mode 382
Table 14.4 Flash Memory Area Divisions 391
Table 14.5 Parameters and Target Modes 392
Table 14.6 Setting On-Board Programming Mode 401
Table 14.7 System Clock Frequency for Automatic-Bit-Rate Adjustment by This LSI 403
Table 14.8 Executable MAT 419
Table 14.9 (1) Useable Area for Programming in User Program Mode 420
Table 14.9 (2) Useable Area for Erasure in User Program Mode 422
Table 14.9 (3) Useable Area for Programming in User Boot Mode 424
Table 14.9 (4) Useable Area for Erasure in User Boot Mode 426
Table 14.10 Hardware Protection 428
Table 14.11 Software Protection 429
Table 14.12 Inquiry and Selection Commands 439
Table 14.13 Programming/Erasing Command 450
Table 14.14 Status Code 458
Table 14.15 Error Code 459

Section 15 Clock Pulse Generator
Table 15.1 Damping Resistance Value 464
Table 15.2 Crystal Resonator Characteristics 464
Table 15.3 External Clock Input Conditions 465
Table 15.4 External Clock Output Stabilization Delay Time 466

Section 16 Power-Down Modes
Table 16.1 Operating Modes and Internal States of LSI 470
Table 16.2 Operating Frequency and Standby Time 475

Section 18 Electrical Characteristics
Table 18.1 Absolute Maximum Ratings 497
Table 18.2 DC Characteristics (1) 498
Table 18.3 DC Characteristics (2) 499
Table 18.4 Permissible Output Currents 499
Table 18.5 Clock Timing 501
Table 18.6 Control Signal Timing 503
Table 18.7 Bus Timing (1) 504
Table 18.8 Bus Timing (2) 505

Table 18.9 DMAC Timing 515
Table 18.10 Timing of On-Chip Peripheral Modules 518
Table 18.11 Flash Memory Characteristics 521

Section 1 Overview

1.1 Features

- High-speed H8S/2000 CPU with an internal 16-bit architecture
  Upward-compatible with H8/300 and H8/300H CPUs on an object level
  Sixteen 16-bit general registers
  65 basic instructions
- Various peripheral functions
  DMA controller (DMAC)
  8-bit timer (TMR)
  Watchdog timer (WDT)
  Serial communication interface (SCI)
  Universal serial bus 2 (USB2)
  Clock pulse generator
- On-chip memory

ROM Type	Model	ROM	RAM	Remarks
Flash memory version	HD64F2170	256 kbytes	32 kbytes	

- General I/O ports
  I/O pins: 76
- Supports various power-down states
- Compact package

Package	(Code)	Body Size	Pin Pitch	Remarks
TQFP-100	TFP-100B	14.0 × 14.0 mm	0.5 mm	—

1.2 Internal Block Diagram

Figure 1.1 shows the internal block diagram.

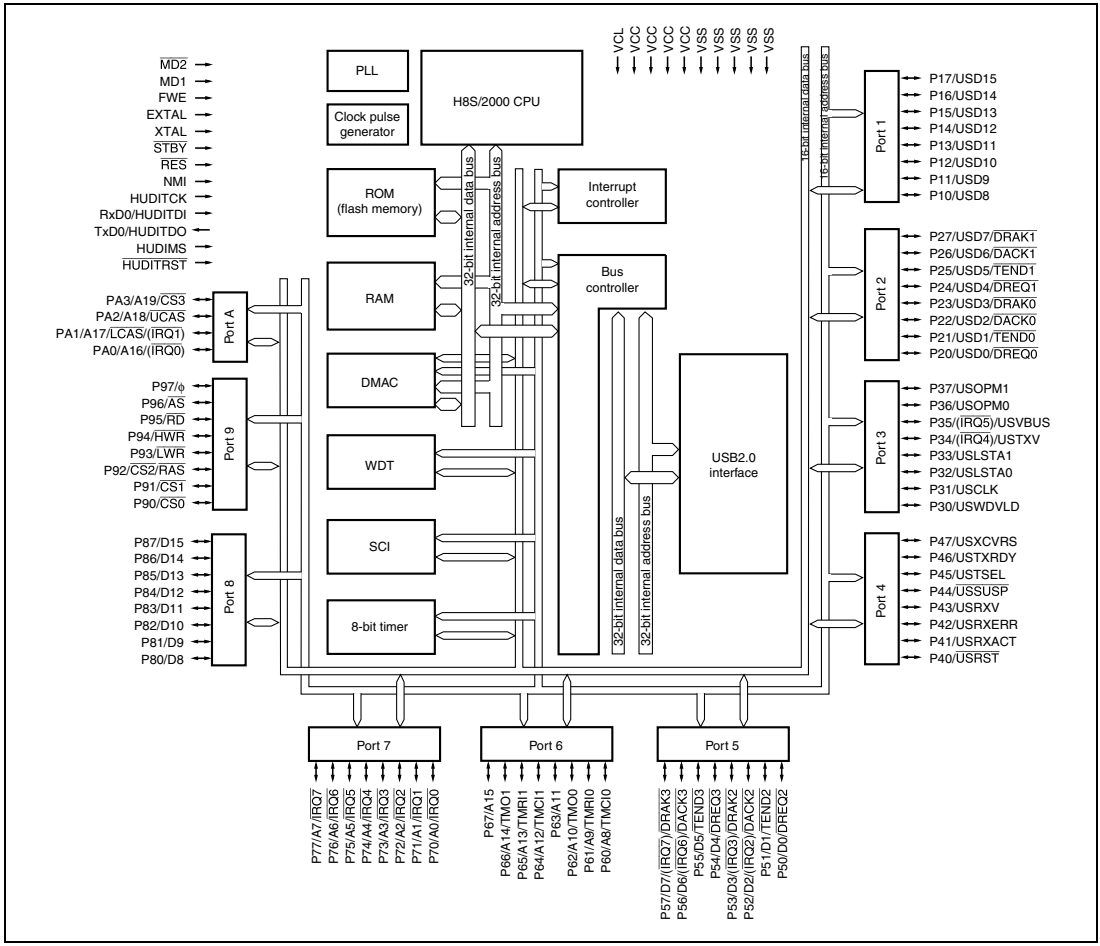


Figure 1.1 Internal Block Diagram

1.3 Pin Description

1.3.1 Pin Arrangement

Figure 1.2 shows the pin arrangement.

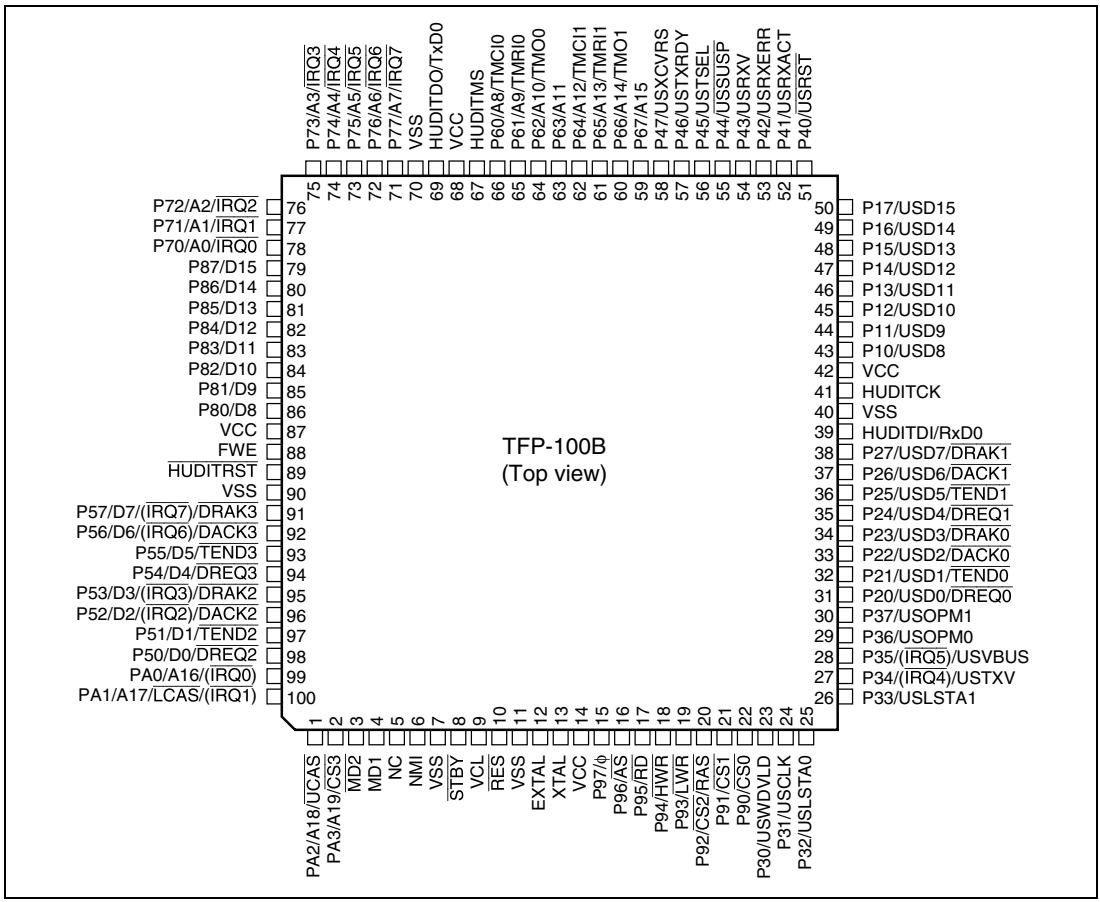


Figure 1.2 Pin Arrangement (TFP-100B)

1.3.2 Pin Arrangements in Each Mode

Table 1.1 lists the pin arrangements in each mode.

Table 1.1 Pin Arrangements in Each Mode

Pin No.	Pin Name		
TFP-100B	Extended Mode (EXPE = 1)	Single-Chip Mode (EXPE = 0)	Flash Memory Programmer Mode
1	PA2/A18/$\overline{\text{UCAS}}$	PA2	NC
2	PA3/A19/$\overline{\text{CS3}}$	PA3	NC
3	$\overline{\text{MD2}}$	$\overline{\text{MD2}}$	VSS
4	MD1	MD1	VSS
5	NC	NC	NC
6	NMI	NMI	NC
7	VSS	VSS	VSS
8	$\overline{\text{STBY}}$	$\overline{\text{STBY}}$	VCC
9	VCL	VCL	VCL
10	$\overline{\text{RES}}$	$\overline{\text{RES}}$	$\overline{\text{RES}}$
11	VSS	VSS	VSS
12	EXTAL	EXTAL	EXTAL
13	XTAL	XTAL	XTAL
14	VCC	VCC	VCC
15	P97/ϕ	P97/ϕ	NC
16	P96/$\overline{\text{AS}}$	P96	NC
17	P95/$\overline{\text{RD}}$	P95	NC
18	P94/$\overline{\text{HWR}}$	P94	NC
19	P93/$\overline{\text{LWR}}$	P93	NC
20	P92/$\overline{\text{CS2}}$/$\overline{\text{RAS}}$	P92	NC
21	P91/$\overline{\text{CS1}}$	P91	NC
22	P90/$\overline{\text{CS0}}$	P90	NC
23	P30/USWDVLD	P30/USWDVLD	NC
24	P31/USCLK	P31/USCLK	NC
25	P32/USLSTA0	P32/USLSTA0	NC
26	P33/USLSTA1	P33/USLSTA1	NC
27	P34/($\overline{\text{IRQ4}}$)/USTXV	P34/($\overline{\text{IRQ4}}$)/USTXV	NC

Pin No.	Pin Name		
TFP-100B	Extended Mode (EXPE = 1)	Single-Chip Mode (EXPE = 0)	Flash Memory Programmer Mode
28	P35/($\overline{IRQ5}$)/USVBUS	P35/($\overline{IRQ5}$)/USVBUS	NC
29	P36/USOPM0	P36/USOPM0	NC
30	P37/USOPM1	P37/USOPM1	NC
31	P20/USD0/$\overline{DREQ0}$	P20/USD0/$\overline{DREQ0}$	NC
32	P21/USD1/$\overline{TEND0}$	P21/USD1/$\overline{TEND0}$	NC
33	P22/USD2/$\overline{DACK0}$	P22/USD2/$\overline{DACK0}$	NC
34	P23/USD3/$\overline{DRAK0}$	P23/USD3/$\overline{DRAK0}$	NC
35	P24/USD4/$\overline{DREQ1}$	P24/USD4/$\overline{DREQ1}$	NC
36	P25/USD5/$\overline{TEND1}$	P25/USD5/$\overline{TEND1}$	NC
37	P26/USD6/$\overline{DACK1}$	P26/USD6/$\overline{DACK1}$	NC
38	P27/USD7/$\overline{DRAK1}$	P27/USD7/$\overline{DRAK1}$	NC
39	HUDITDI/RxD0	HUDITDI/RxD0	NC
40	VSS	VSS	VSS
41	HUDITCK	HUDITCK	NC
42	VCC	VCC	VCC
43	P10/USD8	P10/USD8	NC
44	P11/USD9	P11/USD9	NC
45	P12/USD10	P12/USD10	NC
46	P13/USD11	P13/USD11	NC
47	P14/USD12	P14/USD12	NC
48	P15/USD13	P15/USD13	NC
49	P16/USD14	P16/USD14	NC
50	P17/USD15	P17/USD15	NC
51	P40/$\overline{USRST}$	P40/$\overline{USRST}$	NC
52	P41/USRXACT	P41/USRXACT	NC
53	P42/USRXERR	P42/USRXERR	NC
54	P43/USRXV	P43/USRXV	NC
55	P44/$\overline{USSUSP}$	P44/$\overline{USSUSP}$	NC
56	P45/USTSEL	P45/USTSEL	NC
57	P46/USTXRDY	P46/USTXRDY	NC
58	P47/USXCVRS	P47/USXCVRS	NC
59	P67/A15	P67	NC

Pin No.	Pin Name		
TFP-100B	Extended Mode (EXPE = 1)	Single-Chip Mode (EXPE = 0)	Flash Memory Programmer Mode
60	P66/A14/TMO1	P66/TMO1	NC
61	P65/A13/TMRI1	P65/TMRI1	NC
62	P64/A12/TMCI1	P64/TMCI1	NC
63	P63/A11	P63	NC
64	P62/A10/TMO0	P62/TMO0	NC
65	P61/A9/TMRI0	P61/TMRI0	NC
66	P60/A8/TMCI0	P60/TMCI0	NC
67	HUDITMS	HUDITMS	NC
68	VCC	VCC	VCC
69	HUDITDO/TxD0	HUDITDO/TxD0	NC
70	VSS	VSS	VSS
71	P77/A7/$\overline{\text{IRQ7}}$	P77/$\overline{\text{IRQ7}}$	NC
72	P76/A6/$\overline{\text{IRQ6}}$	P76/$\overline{\text{IRQ6}}$	NC
73	P75/A5/$\overline{\text{IRQ5}}$	P75/$\overline{\text{IRQ5}}$	NC
74	P74/A4/$\overline{\text{IRQ4}}$	P74/$\overline{\text{IRQ4}}$	NC
75	P73/A3/$\overline{\text{IRQ3}}$	P73/$\overline{\text{IRQ3}}$	NC
76	P72/A2/$\overline{\text{IRQ2}}$	P72/$\overline{\text{IRQ2}}$	NC
77	P71/A1/$\overline{\text{IRQ1}}$	P71/$\overline{\text{IRQ1}}$	NC
78	P70/A0/$\overline{\text{IRQ0}}$	P70/$\overline{\text{IRQ0}}$	NC
79	P87/D15	P87	NC
80	P86/D14	P86	NC
81	P85/D13	P85	NC
82	P84/D12	P84	NC
83	P83/D11	P83	NC
84	P82/D10	P82	NC
85	P81/D9	P81	NC
86	P80/D8	P80	NC
87	VCC	VCC	VCC
88	FWE	FWE	FWE
89	$\overline{\text{HUDITRST}}$	$\overline{\text{HUDITRST}}$	NC
90	VSS	VSS	VSS
91	P57/D7/($\overline{\text{IRQ7}}$)/MSSRAC/$\overline{\text{DRAK3}}$	P57/($\overline{\text{IRQ7}}$)/MSSRAC/$\overline{\text{DRAK3}}$	NC

Pin No.	Pin Name		
TFP-100B	Extended Mode (EXPE = 1)	Single-Chip Mode (EXPE = 0)	Flash Memory Programmer Mode
92	P56/D6/(IRQ6)/DACK3	P56/(IRQ6)/DACK3	NC
93	P55/D5/MSBS/TEND3	P55/MSBS/TEND3	NC
94	P54/D4/MSDIO3/DREQ3	P54/MSDIO3/DREQ3	NC
95	P53/D3/(IRQ3)/MSDIO2/DRAK2	P53/(IRQ3)/MSDIO2/DRAK2	NC
96	P52/D2/(IRQ2)/MSDIO1/DACK2	P52/(IRQ2)/MSDIO1/DACK2	NC
97	P51/D1/MSDIO0/TEND2	P51/MSDIO0/TEND2	NC
98	P50/D0/MSCLK/DREQ2	P50/MSCLK/DREQ2	NC
99	PA0/A16/(IRQ0)	PA0/(IRQ0)	NC
100	PA1/A17/LCAS/(IRQ1)	PA1/LCAS/(IRQ1)	NC

1.3.3 Pin Functions

Table 1.2 lists the pin functions.

Table 1.2 Pin Functions

Type	Symbol	Pin No.	I/O	Function
Power	V_{CC}	14, 42, 68, 87	Input	For connection to the power supply. V_{CC} pins should be connected to the system power supply.
	V_{SS}	7, 11, 40, 70, 90	Input	For connection to ground. V_{SS} pins should be connected to the system power supply (0 V).
	V_{CL}	9	Output	The V_{CL} is an external capacity pin for internal step-down power. Connect this pin to V_{SS} through the external capacitor to stabilize the internal step-down power.
Clock	XTAL	13	Input	For connection to a crystal resonator. See section 15, Clock Pulse Generator for typical connection diagrams for a crystal resonator and external clock input.
	EXTAL	12	Input	For connection to a crystal resonator. The EXTAL pin can also input an external clock. See section 15, Clock Pulse Generator for typical connection diagrams for a crystal resonator and external clock input.
	ϕ	15	Output	Supplies the system clock to external devices.
Operating mode control	$\overline{MD2}$ MD1	3 4	Input	These pins set the operating mode. These pins should not be changed while the MCU is operating.
System control	$\overline{RES}$	10	Input	Reset pin. When this pin is driven low, the chip is reset.
	$\overline{STBY}$	8	Input	When this pin is driven low, a transition is made to hardware standby mode.
	FWE	88	Input	This pin is only for the flash memory. This pin is available only in the flash memory version.

Type	Symbol	Pin No.	I/O	Function
Interrupt signals	NMI	6	Input	Nonmaskable interrupt request pin. Fix high when not used.
	$\overline{IRQ7}$ to $\overline{IRQ0}$	71 to 78	Input	These pins request a maskable interrupt. The IRQ sense port select register (ISSR) selects whether the signal is input from the $\overline{IRQn}$ or ($\overline{IRQn}$). (n = 7 to 0)
	($\overline{IRQ7}$) to ($\overline{IRQ0}$)	91, 92, 28, 27, 95, 96, 99, 100	Input	
Address bus	A19 A18 A17 A16 A15 to A8 A7 to A0	2, 1, 100, 99, 59 to 66, 71 to 78	Output	These pins output an address.
Data bus	D15 to D8, D7 to D0	79 to 86, 91 to 98	Input/ output	These pins constitute a bidirectional data bus.
Bus control	$\overline{CS3}$	2	Output	Strobe signal indicating that area 3 is selected.
	$\overline{CS2}$/ $\overline{RAS}$	20	Output	Strobe signal indicating that area 2 is selected. Row address strobe signal for the DRAM.
	$\overline{CS1}$	21	Output	Strobe signal indicating that area 1 is selected.
	$\overline{CS0}$	22	Output	Strobe signal indicating that area 0 is selected.
	$\overline{AS}$	16	Output	When this pin is low, it indicates that address output on the address bus is valid.
	$\overline{RD}$	17	Output	When this pin is low, it indicates that the normal space is being read.
	$\overline{HWR}$	18	Output	Strobe signal indicating that normal space is to be written, and the upper half (D15 to D8) of the data bus is enabled.
	$\overline{LWR}$	19	Output	Strobe signal indicating that normal space is to be written, and the lower half (D7 to D0) of the data bus is enabled.
	$\overline{UCAS}$	1	Output	Upper column address strobe signal for accessing the 16-bit DRAM space or column address strobe signal for accessing the 8-bit DRAM space.
	$\overline{LCAS}$	100	Output	Lower column address strobe signal for accessing the 16-bit DRAM space.

Type	Symbol	Pin No.	I/O	Function
DMA controller (DMAC)	$\overline{\text{DREQ3}}$ $\overline{\text{DREQ2}}$ $\overline{\text{DREQ1}}$ $\overline{\text{DREQ0}}$	94, 98, 35, 31	Input	These signals request DMAC activation for channels 3 to 0.
	$\overline{\text{DACK3}}$ $\overline{\text{DACK2}}$ $\overline{\text{DACK1}}$ $\overline{\text{DACK0}}$	92 96 37 33	Output	DMAC single address transfer acknowledge signals for channels 3 to 0.
	$\overline{\text{TEND3}}$ $\overline{\text{TEND2}}$ $\overline{\text{TEND1}}$ $\overline{\text{TEND0}}$	93, 97, 36, 32	Output	These signals indicate the end of DMAC data transfer for channels 3 to 0.
	$\overline{\text{DRAK3}}$ $\overline{\text{DRAK2}}$ $\overline{\text{DRAK1}}$ $\overline{\text{DRAK0}}$	91, 95, 38, 34	Output	These signals notify DMAC external request acknowledge and execution start for channels 3 to 0 to external devices.
8-bit timer (TMR)	TMO1 TMO0	60, 64	Output	Compare match output pins.
	TMCI1 TMCI0	62, 66	Input	External clock input pins to counters.
	TMRI1 TMRI0	61, 65	Input	Counter reset input pins.
Serial communication interface for boot mode (SCI)	TxD0	69	Output	Data output pin.
	RxD0	39	Input	Data input pin.
Universal serial bus 2 (USB2)	USCLK	23	Input	USB clock.
	USVBUS	28	Input	Input pin for connection/disconnection detection of USB cable.
	USLSTA0	25	Input	Input signal pins from USB2.0 transceiver.
	USLSTA1	26	Input	
	USRXACT	52	Input	
	USRXERR	53	Input	
	USRXV	54	Input	
	USTXRDY	57	Input	
	USWDVLD	23	Input/ output	
	USOPM0	29	Output	Output signal pins for USB2.0 transceiver.
	USOPM1	30	Output	

Type	Symbol	Pin No.	I/O	Function
Universal serial bus 2 (USB2)	$\overline{\text{USRST}}$	51	Output	Output signal pins for USB2.0 transceiver.
	USTXV	27	Output	
	$\overline{\text{USSUSP}}$	55	Output	
	USTSEL	56	Output	
	USXCVRS	58	Output	
	USD15 to USD8 USD7 to USD0	50 to 43, 38 to 31	Input/ output	Data input/output.
I/O ports	P17 to P10	50 to 43	Input/ output	Eight-bit input/output pins.
	P27 to P20	38 to 31	Input/ output	Eight-bit input/output pins.
	P37 to P30	30 to 23	Input/ output	Eight-bit input/output pins.
	P47 to P40	58 to 51	Input/ output	Eight-bit input/output pins.
	P57 to P50	91 to 98	Input/ output	Eight-bit input/output pins.
	P67 to P60	59 to 66	Input/ output	Eight-bit input/output pins.
	P77 to P70	71 to 78	Input/ output	Eight-bit input/output pins.
	P87 to P80	79 to 86	Input/ output	Eight-bit input/output pins.
	P97 to P90	15 to 22	Input/ output	Eight-bit input/output pins.
	PA3 PA2 PA1 PA0	2, 1, 100, 99	Input/ output	Four-bit input/output pins.

Section 2 CPU

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

This section describes the H8S/2000 CPU. The usable modes and address spaces differ depending on the product. For details on each product, refer to section 3, MCU Operating Modes.

2.1 Features

- Upward-compatibility with H8/300 and H8/300H CPUs
  Can execute H8/300 and H8/300H CPU object programs
- General-register architecture
  Sixteen 16-bit general registers also usable as sixteen 8-bit registers or eight 32-bit registers
- Sixty-five basic instructions
  8/16/32-bit arithmetic and logic instructions
  Multiply and divide instructions
  Powerful bit-manipulation instructions
- Eight addressing modes
  Register direct [Rn]
  Register indirect [@ERn]
  Register indirect with displacement [@(d:16,ERn) or @(d:32,ERn)]
  Register indirect with post-increment or pre-decrement [@ERn+ or @–ERn]
  Absolute address [@aa:8, @aa:16, @aa:24, or @aa:32]
  Immediate [#xx:8, #xx:16, or #xx:32]
  Program-counter relative [@(d:8,PC) or @(d:16,PC)]
  Memory indirect [@@aa:8]
- 16-Mbyte address space
  Program: 16 Mbytes
  Data: 16 Mbytes
- High-speed operation
  All frequently-used instructions are executed in one or two states
  8/16/32-bit register-register add/subtract: 1 state
  8 × 8-bit register-register multiply: 12 states (MULXU.B), 13 states (MULXS.B)
  16 ÷ 8-bit register-register divide: 12 states (DIVXU.B)
  16 × 16-bit register-register multiply: 20 states (MULXU.W), 21 states (MULXS.W)
  32 ÷ 16-bit register-register divide: 20 states (DIVXU.W)

- Two CPU operating modes
  Normal mode*
  Advanced mode

Note: For this LSI, normal mode is not available.

- Power-down state
  Transition to power-down state by SLEEP instruction

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

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

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

		Execution States	
Instruction	**Mnemonic**	**H8S/2600**	**H8S/2000**
MULXU	MULXU.B Rs, Rd	3	12
	MULXU.W Rs, ERd	4	20
MULXS	MULXS.B Rs, Rd	4	13
	MULXS.W Rs, ERd	5	21

In addition, there are differences in address space, CCR and EXR register functions, power-down modes, etc., depending on the model.

2.1.2 Differences from H8/300 CPU

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

- More general registers and control registers
  Eight 16-bit extended registers, and one 8-bit and two 32-bit control registers, have been added.
- Expanded address space
  Normal mode supports the same 64-kbyte address space as the H8/300 CPU.
  Advanced mode supports a maximum 16-Mbyte address space.
- Enhanced addressing
  The addressing modes have been enhanced to make effective use of the 16-Mbyte address space.
- Enhanced instructions
  Addressing modes of bit-manipulation instructions have been enhanced.
  Signed multiply and divide instructions have been added.
  Two-bit shift and two-bit rotate instructions have been added.
  Instructions for saving and restoring multiple registers have been added.
  A test and set instruction has been added.
- Higher speed
  Basic instructions are executed twice as fast.

2.1.3 Differences from H8/300H CPU

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

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

2.2 CPU Operating Modes

The H8S/2000 CPU has two operating modes: normal and advanced. Normal mode supports a maximum 64-kbyte address space. Advanced mode supports a maximum 16-Mbyte address space. The mode is selected by the LSI's mode pins.

2.2.1 Normal Mode

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

- Address space

  Linear access to a maximum address space of 64 kbytes is possible.
- Extended registers (En)

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

  When extended register En is used as a 16-bit register it can contain any value, even when the corresponding general register (Rn) is used as an address register. (If general register Rn is referenced in the register indirect addressing mode with pre-decrement (@–Rn) or post-increment (@Rn+) and a carry or borrow occurs, the value in the corresponding extended register (En) will be affected.)
- Instruction set

  All instructions and addressing modes can be used. Only the lower 16 bits of effective addresses (EA) are valid.
- Exception vector table and memory indirect branch addresses

  In normal mode, the top area starting at H'0000 is allocated to the exception vector table. One branch address is stored per 16 bits. The exception vector table in normal mode is shown in figure 2.1. For details of the exception vector table, see section 4, Exception Handling.

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

  When the program counter (PC) is pushed onto the stack in a subroutine call, and the PC, condition-code register (CCR), and extended control register (EXR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.2. EXR is not pushed onto the stack in interrupt control mode 0. For details, see section 4, Exception Handling.

Note: For this LSI, normal mode is not available.

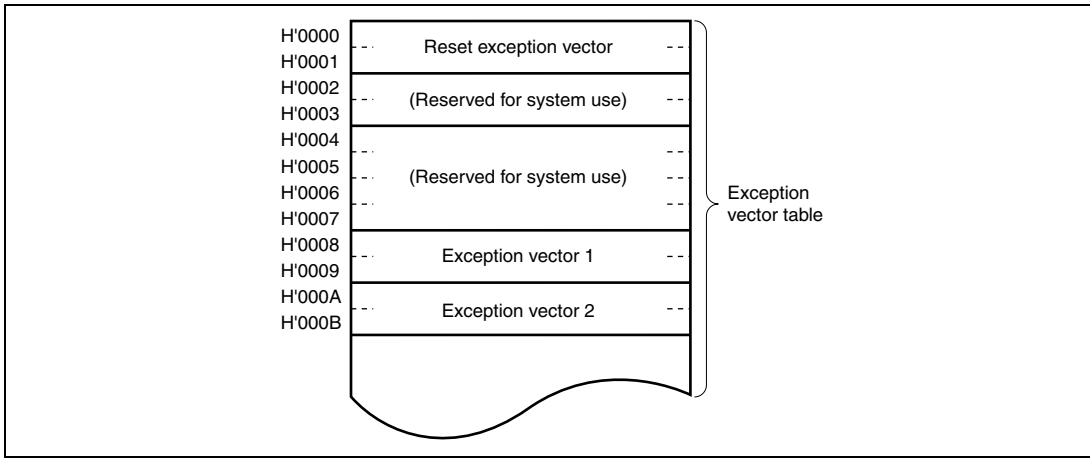
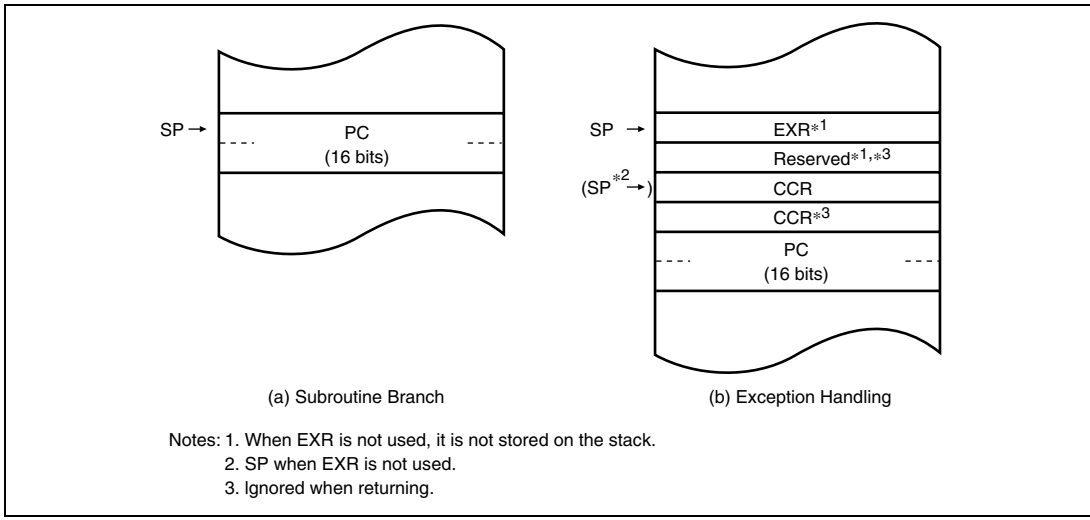


Figure 2.1 Exception Vector Table (Normal Mode)



Notes: 1. When EXR is not used, it is not stored on the stack.
2. SP when EXR is not used.
3. Ignored when returning.

Figure 2.2 Stack Structure in Normal Mode

2.2.2 Advanced Mode

- Address space

  Linear access to a maximum address space of 16 Mbytes is possible.
- Extended registers (En)

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

  All instructions and addressing modes can be used.
- Exception vector table and memory indirect branch addresses

  In advanced mode, the top area starting at H'00000000 is allocated to the exception vector table in 32-bit units. In each 32 bits, the upper 8 bits are ignored and a branch address is stored in the lower 24 bits (see figure 2.3). For details of the exception vector table, see section 4, Exception Handling.

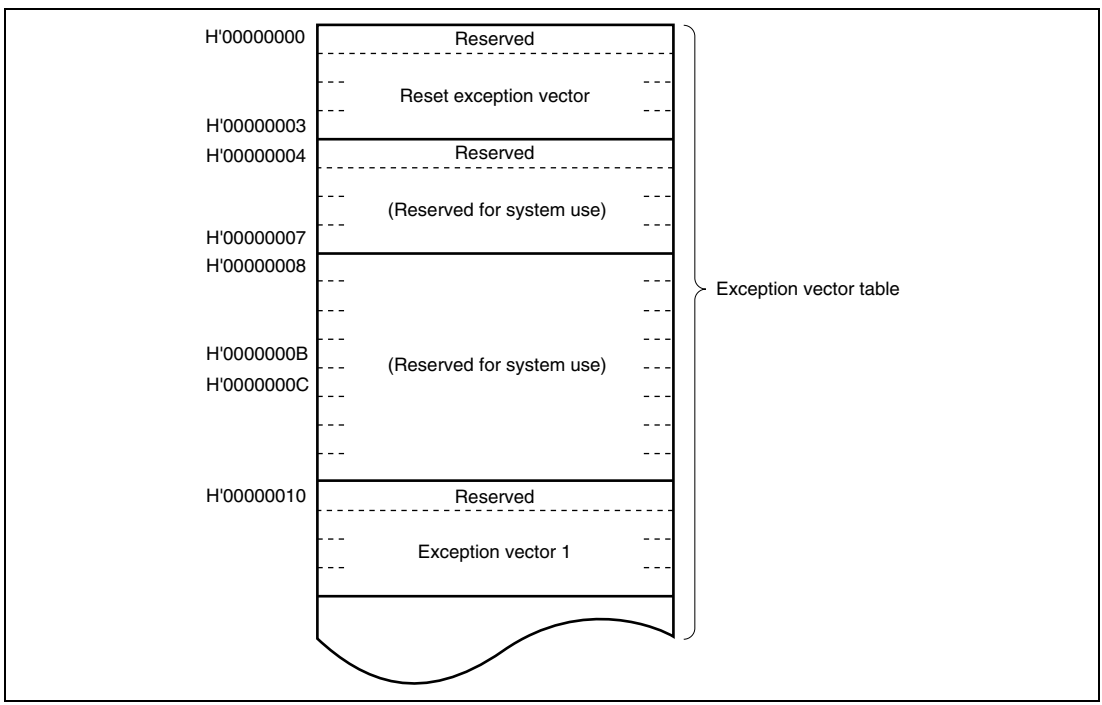


Figure 2.3 Exception Vector Table (Advanced Mode)

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

- Stack structure

  In advanced mode, when the program counter (PC) is pushed onto the stack in a subroutine call, and the PC, condition-code register (CCR), and extended control register (EXR) are pushed onto the stack in exception handling, they are stored as shown in figure 2.4. EXR is not pushed onto the stack in interrupt control mode 0. For details, see section 4, Exception Handling.

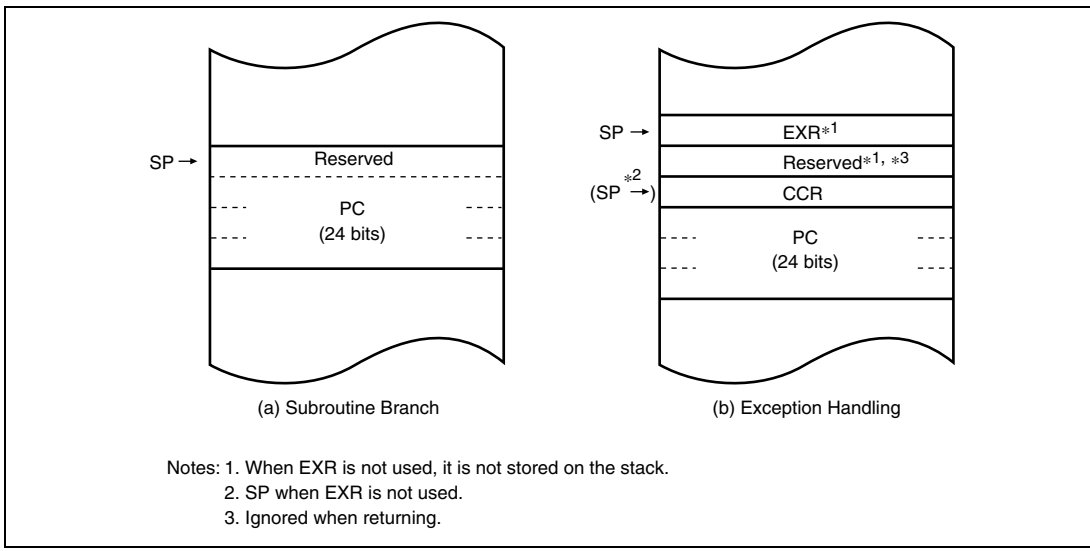


Figure 2.4 Stack Structure in Advanced Mode

2.3 Address Space

Figure 2.5 shows a memory map of the H8S/2000 CPU. The H8S/2000 CPU provides linear access to a maximum 64-kbyte address space in normal mode, and a maximum 16-Mbyte (architecturally 4-Gbyte) address space in advanced mode. The usable modes and address spaces differ depending on the product. For details on each product, refer to section 3, MCU Operating Modes.

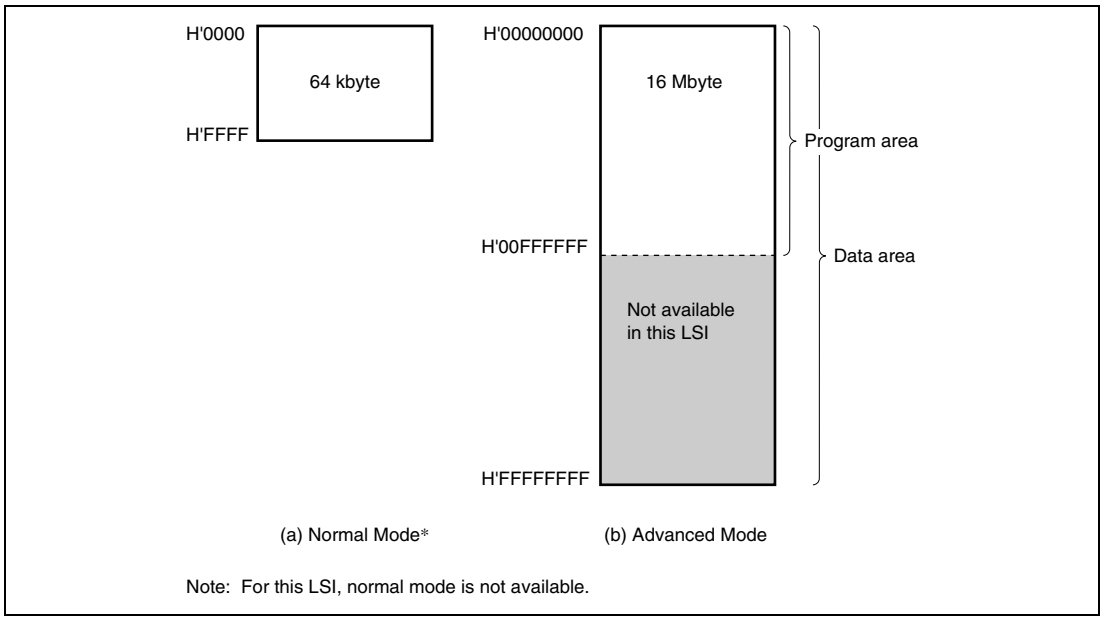


Figure 2.5 Memory Map

2.4 Register Configuration

The H8S/2000 CPU has the internal registers shown in figure 2.6. There are two types of registers: general registers and control registers. Control registers are a 24-bit program counter (PC), an 8-bit extended control register (EXR), and an 8-bit condition code register (CCR).

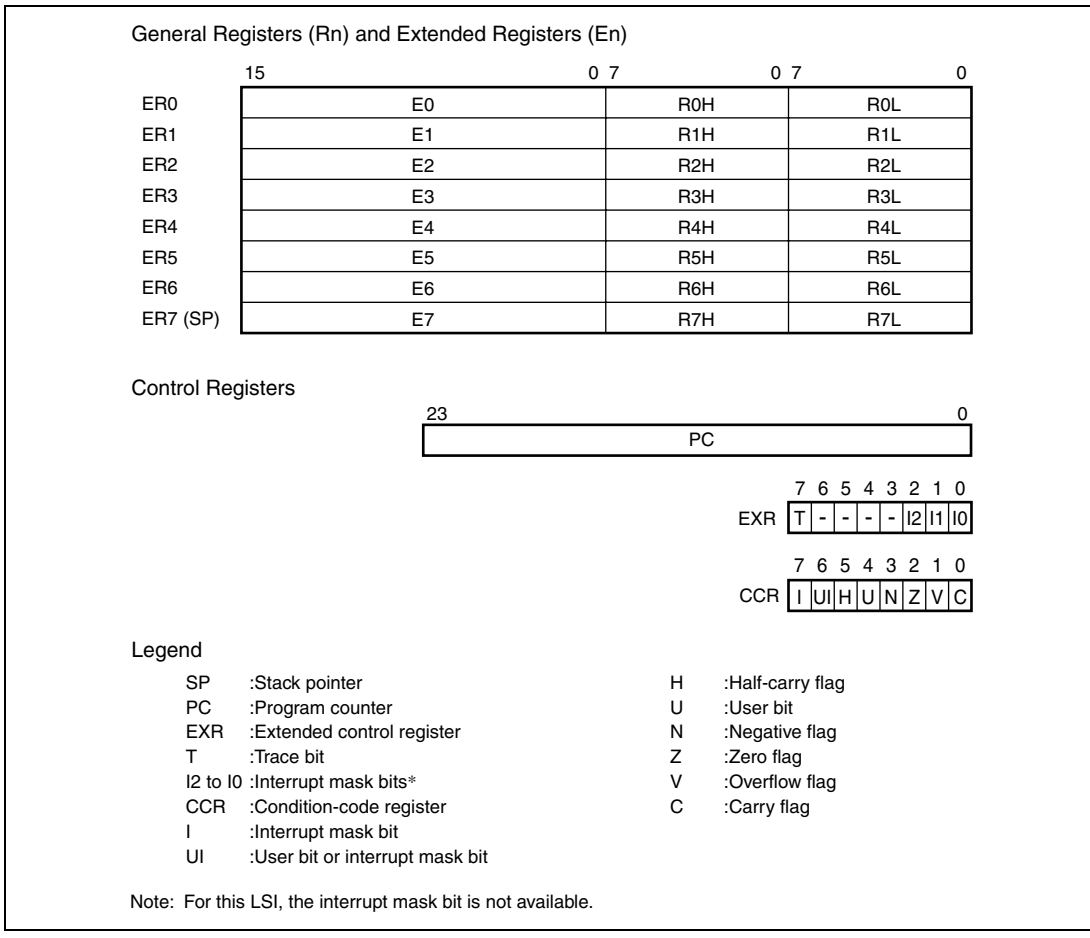


Figure 2.6 CPU Internal Registers

2.4.1 General Registers

The H8S/2000 CPU has eight 32-bit general registers. These general registers are all functionally alike and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. Figure 2.7 illustrates the usage of the general registers. When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

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

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

The usage of each register can be selected independently.

General register ER7 has the function of the stack pointer (SP) in addition to its general-register function, and is used implicitly in exception handling and subroutine calls. Figure 2.8 shows the stack.

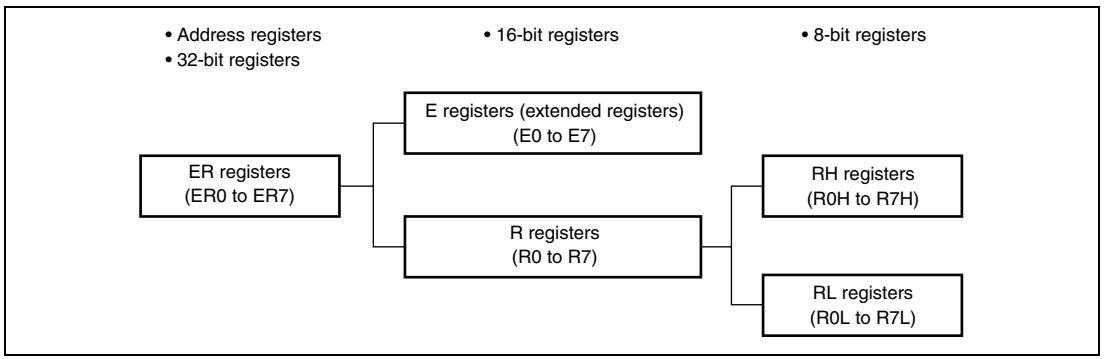


Figure 2.7 Usage of General Registers

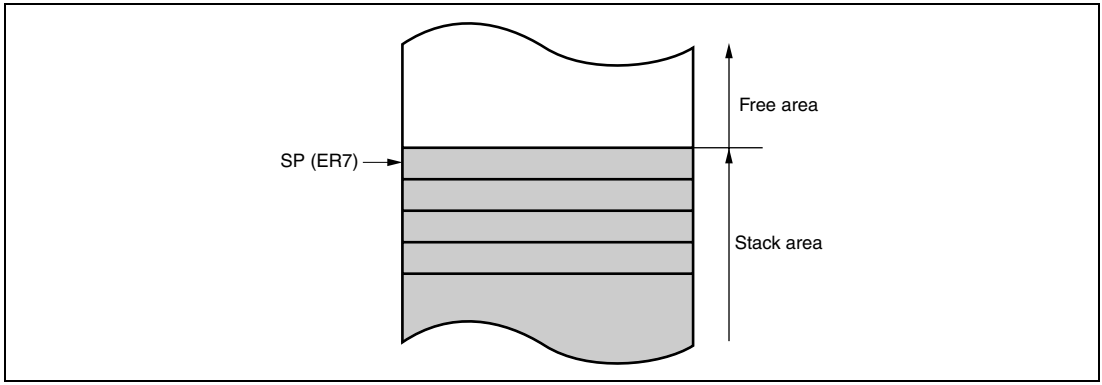


Figure 2.8 Stack

2.4.2 Program Counter (PC)

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

2.4.3 Extended Control Register (EXR)

EXR is an 8-bit register that can be operated by the LDC, STC, ANDC, ORC, and XORC instructions. When an instruction other than STC is executed, all interrupts including NMI are masked in three states after the instruction is completed.

Bit	Bit Name	Initial Value	R/W	Description
7	T	0	R/W	Trace Bit When this bit is set to 1, trace exception processing starts every when an instruction is executed. When this bit is cleared to 0, instructions are consecutively executed.
6 to3	—	All1	—	Reserved These bits are always read as 1.
2 to 0	I2	1	R/W	Interrupt Mask Bits 2 to 0
	I1	1	R/W	Specify interrupt request mask levels (0 to 7). In this LSI, these bits cannot be used as the interrupt mask level.
	I0	1	R/W	

2.4.4 Condition-Code Register (CCR)

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

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

Bit	Bit Name	Initial Value	R/W	Description
7	I	1	R/W	Interrupt Mask Bit Masks interrupts other than NMI when set to 1. NMI is accepted regardless of the I bit setting. The I bit is set to 1 at the start of an exception-handling sequence. For details, refer to section 5, Interrupt Controller.
6	UI	Undefined	R/W	User Bit or Interrupt Mask Bit Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.
5	H	Undefined	R/W	Half-Carry Flag When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.
4	U	Undefined	R/W	User Bit Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.
3	N	Undefined	R/W	Negative Flag Stores the value of the most significant bit of data as a sign bit.
2	Z	Undefined	R/W	Zero Flag Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.
1	V	Undefined	R/W	Overflow Flag Set to 1 when an arithmetic overflow occurs, and cleared to 0 otherwise.

Bit	Bit Name	Initial Value	R/W	Description
0	C	Undefined	R/W	Carry Flag Set to 1 when a carry occurs, and cleared to 0 otherwise. Used by: • Add instructions, to indicate a carry • Subtract instructions, to indicate a borrow • Shift and rotate instructions, to indicate a carry The carry flag is also used as a bit accumulator by bit manipulation instructions.

2.4.5 Initial Register Values

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

2.5 Data Formats

The H8S/2000 CPU can process 1-bit, 4-bit BCD, 8-bit (byte), 16-bit (word), and 32-bit (longword) data. Bit-manipulation instructions operate on 1-bit data by accessing bit n (n = 0, 1, 2, …, 7) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

2.5.1 General Register Data Formats

Figure 2.9 shows the data formats of general registers.

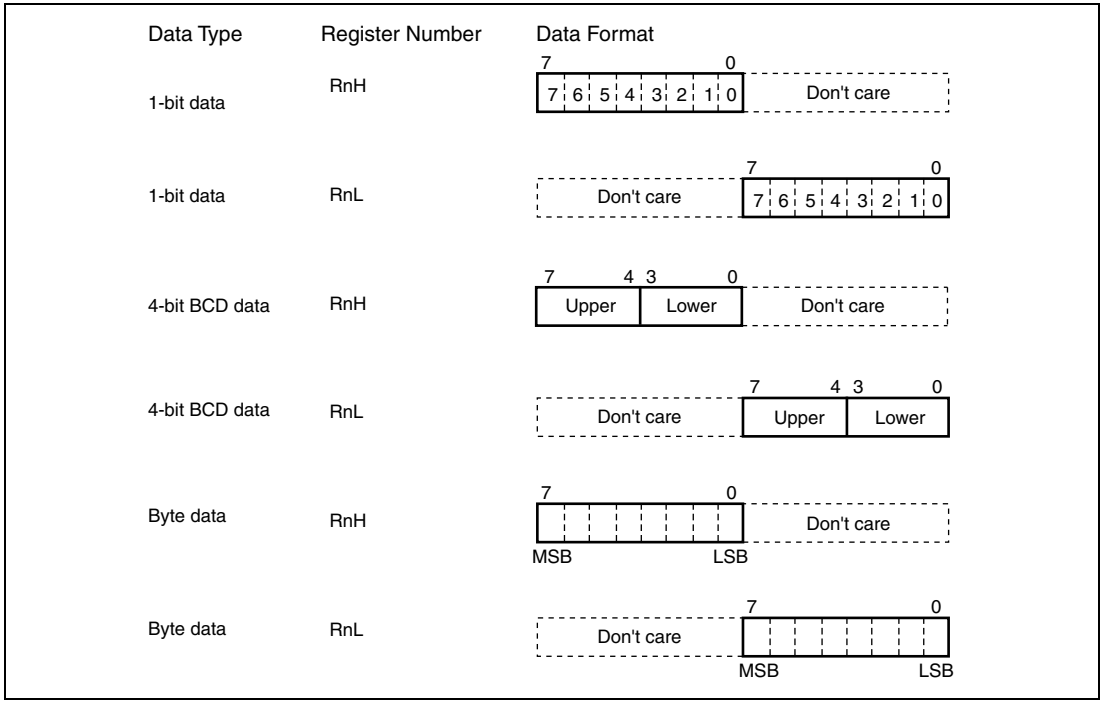


Figure 2.9 General Register Data Formats (1)

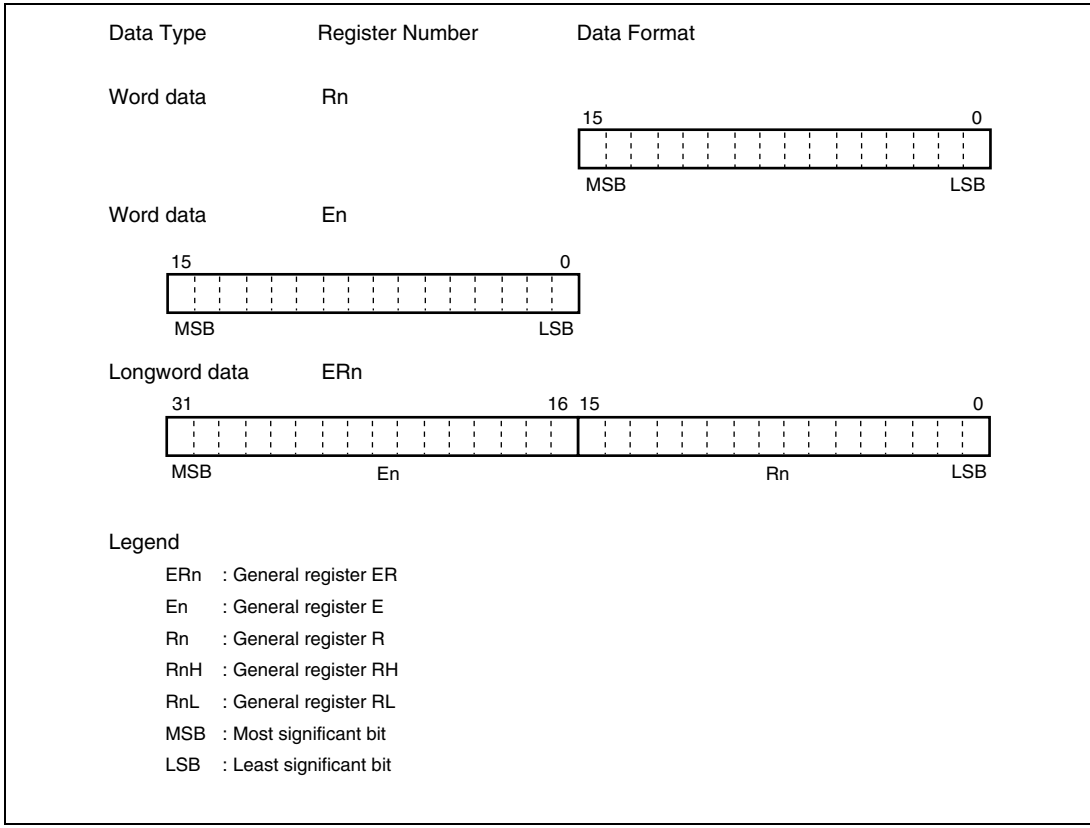


Figure 2.9 General Register Data Formats (2)

2.5.2 Memory Data Formats

Figure 2.10 shows the data formats in memory. The H8S/2000 CPU can access word data and longword data in memory, but word or longword data must begin at an even address. If an attempt is made to access word or longword data at an odd address, no address error occurs but the least significant bit of the address is regarded as 0, so the access starts at the preceding address. This also applies to instruction fetches.

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

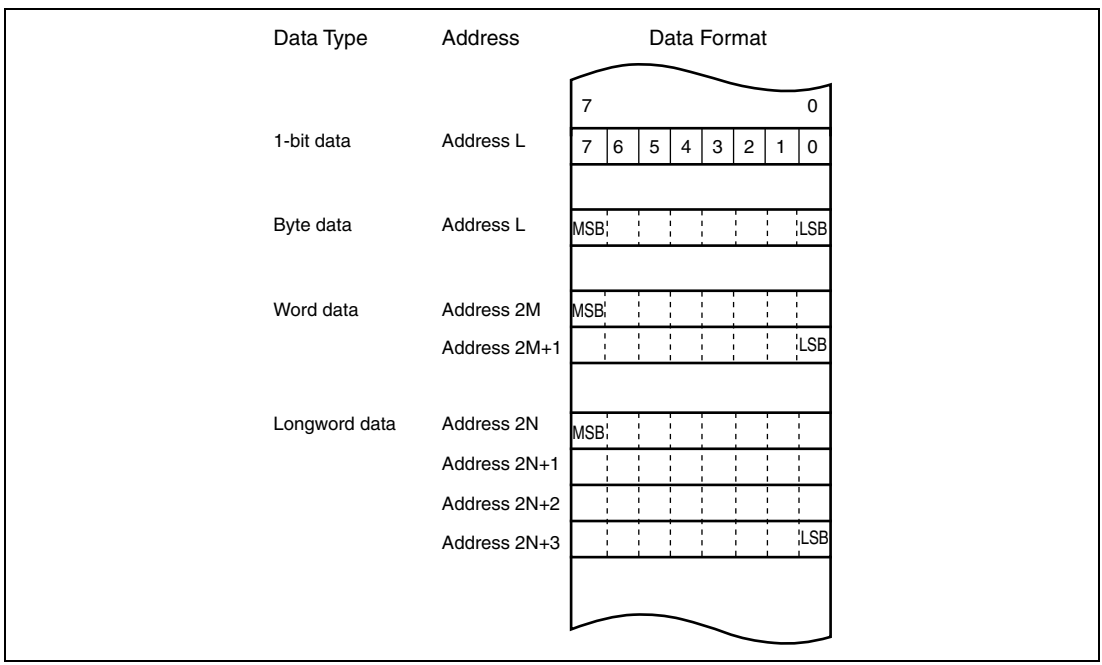


Figure 2.10 Memory Data Formats

2.6 Instruction Set

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

Table 2.1 Instruction Classification

Function	Instructions	Size	Types
Data transfer	MOV	B/W/L	5
	POP*[1], PUSH*[1]	W/L	
	LDM, STM	L	
	MOVFPE*[3], MOVTPE*[3]	B	
Arithmetic operations	ADD, SUB, CMP, NEG	B/W/L	19
	ADDX, SUBX, DAA, DAS	B	
	INC, DEC	B/W/L	
	ADDS, SUBS	L	
	MULXU, DIVXU, MULXS, DIVXS	B/W	
	EXTU, EXTS	W/L	
	TAS*[4]	B	
Logic operations	AND, OR, XOR, NOT	B/W/L	4
Shift	SHAL, SHAR, SHLL, SHLR, ROTL, ROTR, ROTXL, ROTXR	B/W/L	8
Bit manipulation	BSET, BCLR, BNOT, BTST, BLD, BILD, BST, BIST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR	B	14
Branch	B_{CC}*[2], JMP, BSR, JSR, RTS	–	5
System control	TRAPA, RTE, SLEEP, LDC, STC, ANDC, ORC, XORC, NOP	–	9
Block data transfer	EEPMOV	–	1
			Total: 65

Notes: B: Byte size; W: Word size; L: Longword size.

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

2.6.1 Table of Instructions Classified by Function

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

Table 2.2 Operation Notation

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

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

Table 2.3 Data Transfer Instructions

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

Note: Size refers to the operand size.
B: Byte
W: Word
L: Longword

Table 2.4 Arithmetic Operations Instructions (1)

Instruction	Size*	Function
ADD	B/W/L	Rd ± Rs → Rd, Rd ± #IMM → Rd
SUB		Performs addition or subtraction on data in two general registers, or on immediate data and data in a general register. (Subtraction on immediate data and data in a general register cannot be performed in bytes. Use the SUBX or ADD instruction.)
ADDX	B	Rd ± Rs ± C → Rd, Rd ± #IMM ± C → Rd
SUBX		Performs addition or subtraction with carry on data in two general registers, or on immediate data and data in a general register.
INC	B/W/L	Rd ± 1 → Rd, Rd ± 2 → Rd
DEC		Adds or subtracts the value 1 or 2 to or from data in a general register. (Only the value 1 can be added to or subtracted from byte operands.)
ADDS	L	Rd ± 1 → Rd, Rd ± 2 → Rd, Rd ± 4 → Rd
SUBS		Adds or subtracts the value 1, 2, or 4 to or from data in a 32-bit register.
DAA	B	Rd (decimal adjust) → Rd
DAS		Decimal-adjusts an addition or subtraction result in a general register by referring to CCR to produce 4-bit BCD data.
MULXU	B/W	Rd × Rs → Rd
		Performs unsigned multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits or 16 bits × 16 bits → 32 bits.
MULXS	B/W	Rd × Rs → Rd
		Performs signed multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits or 16 bits × 16 bits → 32 bits.
DIVXU	B/W	Rd ÷ Rs → Rd
		Performs unsigned division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder.

Note: Size refers to the operand size.
B: Byte
W: Word
L: Longword

Table 2.4 Arithmetic Operations Instructions (2)

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

Notes: 1. Size refers to the operand size.
B: Byte
W: Word
L: Longword
2. Only register ER0, ER1, ER4, or ER5 should be used when using the TAS instruction.

Table 2.5 Logic Operations Instructions

Instruction	Size*	Function
AND	B/W/L	Rd ∧ Rs → Rd, Rd ∧ #IMM → Rd Performs a logical AND operation on a general register and another general register or immediate data.
OR	B/W/L	Rd ∨ Rs → Rd, Rd ∨ #IMM → Rd Performs a logical OR operation on a general register and another general register or immediate data.
XOR	B/W/L	Rd ⊕ Rs → Rd, Rd ⊕ #IMM → Rd Performs a logical exclusive OR operation on a general register and another general register or immediate data.
NOT	B/W/L	~ Rd → Rd Takes the one's complement (logical complement) of data in a general register.

Note: Size refers to the operand size.
B: Byte
W: Word
L: Longword

Table 2.6 Shift Instructions

Instruction	Size*	Function
SHAL	B/W/L	Rd (shift) → Rd
SHAR		Performs an arithmetic shift on data in a general register. 1-bit or 2 bit shift is possible.
SHLL	B/W/L	Rd (shift) → Rd
SHLR		Performs a logical shift on data in a general register. 1-bit or 2 bit shift is possible.
ROTL	B/W/L	Rd (rotate) → Rd
ROTR		Rotates data in a general register. 1-bit or 2 bit rotation is possible.
ROTXL	B/W/L	Rd (rotate) → Rd
ROTXR		Rotates data including the carry flag in a general register. 1-bit or 2 bit rotation is possible.

Note: Size refers to the operand size.
B: Byte
W: Word
L: Longword

Table 2.7 Bit Manipulation Instructions (1)

Instruction	Size*	Function
BSET	B	1 → (<bit-No.> of <EAd>) Sets a specified bit in a general register or memory operand to 1. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BCLR	B	0 → (<bit-No.> of <EAd>) Clears a specified bit in a general register or memory operand to 0. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BNOT	B	~ (<bit-No.> of <EAd>) → (<bit-No.> of <EAd>) Inverts a specified bit in a general register or memory operand. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BTST	B	~ (<bit-No.> of <EAd>) → Z Tests a specified bit in a general register or memory operand and sets or clears the Z flag accordingly. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BAND	B	C ∧ (<bit-No.> of <EAd>) → C Logically ANDs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag.
BIAND	B	C ∧ (<bit-No.> of <EAd>) → C Logically ANDs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BOR	B	C ∨ (<bit-No.> of <EAd>) → C Logically ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag.
BIOR	B	C ∨ (~ <bit-No.> of <EAd>) → C Logically ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.

Note: Size refers to the operand size.
B: Byte

Table 2.7 Bit Manipulation Instructions (2)

Instruction	Size*	Function
BXOR	B	C ⊕ (<bit-No.> of <EAd>) → C Logically exclusive-ORs the carry flag with a specified bit in a general register or memory operand and stores the result in the carry flag.
BIXOR	B	C ⊕ ~ (<bit-No.> of <EAd>) → C Logically exclusive-ORs the carry flag with the inverse of a specified bit in a general register or memory operand and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BLD	B	(<bit-No.> of <EAd>) → C Transfers a specified bit in a general register or memory operand to the carry flag.
BILD	B	~ (<bit-No.> of <EAd>) → C Transfers the inverse of a specified bit in a general register or memory operand to the carry flag. The bit number is specified by 3-bit immediate data.
BST	B	C → (<bit-No.> of <EAd>) Transfers the carry flag value to a specified bit in a general register or memory operand.
BIST	B	~ C → (<bit-No.>. of <EAd>) Transfers the inverse of the carry flag value to a specified bit in a general register or memory operand. The bit number is specified by 3-bit immediate data.

Note: Size refers to the operand size.
B: Byte

Table 2.8 Branch Instructions

Instruction	Size	Function
Bcc	—	Branches to a specified address if a specified condition is true. The branching conditions are listed below.

Mnemonic	Description	Condition
BRA (BT)	Always (true)	Always
BRN (BF)	Never (false)	Never
BHI	High	$C \vee Z = 0$
BLS	Low or same	$C \vee Z = 1$
BCC (BHS)	Carry clear (high or same)	$C = 0$
BCS (BLO)	Carry set (low)	$C = 1$
BNE	Not equal	$Z = 0$
BEQ	Equal	$Z = 1$
BVC	Overflow clear	$V = 0$
BVS	Overflow set	$V = 1$
BPL	Plus	$N = 0$
BMI	Minus	$N = 1$
BGE	Greater or equal	$N \oplus V = 0$
BLT	Less than	$N \oplus V = 1$
BGT	Greater than	$Z \vee (N \oplus V) = 0$
BLE	Less or equal	$Z \vee (N \oplus V) = 1$

JMP	–	Branches unconditionally to a specified address.
BSR	–	Branches to a subroutine at a specified address
JSR	–	Branches to a subroutine at a specified address
RTS	–	Returns from a subroutine

Table 2.9 System Control Instructions

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

Note: Size refers to the operand size.
B: Byte
W: Word

Table 2.10 Block Data Transfer Instructions

Instruction	Size	Function

EEPMOV.B	—	if R4L ≠ 0 then
		Repeat @ER5+ → @ER6+
		R4L–1 → R4L
		Until R4L = 0
		else next:

EEPMOV.W	—	if R4 ≠ 0 then
		Repeat @ER5+ → @ER6+
		R4–1 → R4
		Until R4 = 0
		else next:
		Transfers a data block. Starting from the address set in ER5, transfers data for the number of bytes set in R4L or R4 to the address location set in ER6.
		Execution of the next instruction begins as soon as the transfer is completed.

2.6.2 Basic Instruction Formats

The H8S/2000 CPU instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op), a register field (r), an effective address extension (EA), and a condition field (cc).

Figure 2.11 shows examples of instruction formats.

- Operation field

  Indicates the function of the instruction, the addressing mode, and the operation to be carried out on the operand. The operation field always includes the first four bits of the instruction. Some instructions have two operation fields.
- Register field

  Specifies a general register. Address registers are specified by 3 bits, and data registers by 3 bits or 4 bits. Some instructions have two register fields, and some have no register field.
- Effective address extension

  8, 16, or 32 bits specifying immediate data, an absolute address, or a displacement.
- Condition field

  Specifies the branching condition of Bcc instructions.

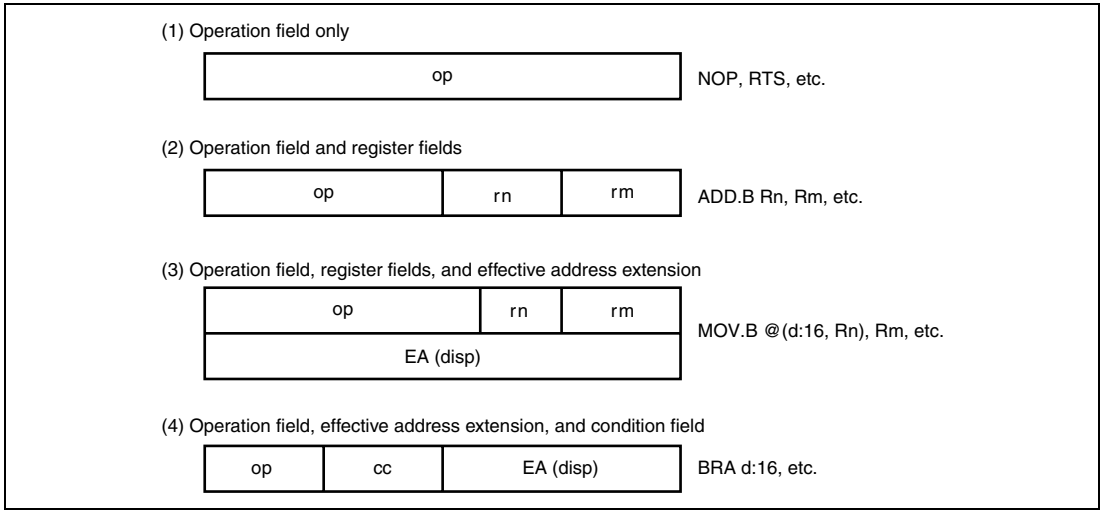


Figure 2.11 Instruction Formats (Examples)

2.7 Addressing Modes and Effective Address Calculation

The H8S/2000 CPU supports the eight addressing modes listed in table 2.11. Each instruction uses a subset of these addressing modes.

Arithmetic and logic operations instructions can use the register direct and immediate addressing modes. Data transfer instructions can use all addressing modes except program-counter relative and memory indirect. Bit manipulation instructions can use register direct, register indirect, or absolute addressing mode to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.

Table 2.11 Addressing Modes

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

2.7.1 Register Direct—Rn

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

2.7.2 Register Indirect—@ERn

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

2.7.3 Register Indirect with Displacement—@(d:16, ERn) or @(d:32, ERn)

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

2.7.4 Register Indirect with Post-Increment or Pre-Decrement—@ERn+ or @-ERn

Register Indirect with Post-Increment—@ERn+: The register field of the instruction code specifies an address register (ERn) which contains the address of a memory operand. After the operand is accessed, 1, 2, or 4 is added to the address register contents and the sum is stored in the address register. The value added is 1 for byte access, 2 for word access, and 4 for longword access. For word or longword transfer instructions, the register value should be even.

Register Indirect with Pre-Decrement—@-ERn: The value 1, 2, or 4 is subtracted from an address register (ERn) specified by the register field in the instruction code, and the result becomes the address of a memory operand. The result is also stored in the address register. The value subtracted is 1 for byte access, 2 for word access, and 4 for longword access. For word or longword transfer instructions, the register value should be even.

2.7.5 Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32

The instruction code contains the absolute address of a memory operand. The absolute address may be 8 bits long (@aa:8), 16 bits long (@aa:16), 24 bits long (@aa:24), or 32 bits long (@aa:32). Table 2.12 indicates the accessible absolute address ranges.

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

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

Table 2.12 Absolute Address Access Ranges

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

2.7.6 Immediate—#xx:8, #xx:16, or #xx:32

The 8-bit (#xx:8), 16-bit (#xx:16), or 32-bit (#xx:32) immediate data contained in a instruction code can be used directly as an operand.

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

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

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

2.7.8 Memory Indirect—@@aa:8

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

In normal mode, the memory operand is a word operand and the branch address is 16 bits long. In advanced mode, the memory operand is a longword operand, the first byte of which is assumed to be 0 (H'00).

Note that the top area of the address range in which the branch address is stored is also used for the exception vector area. For further details, refer to section 4, Exception Handling.

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

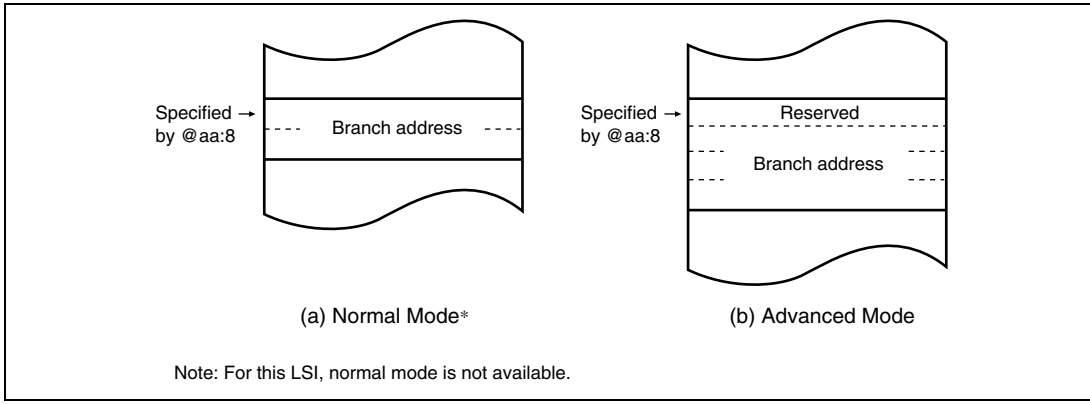


Figure 2.12 Branch Address Specification in Memory Indirect Addressing Mode

2.7.9 Effective Address Calculation

Table 2.13 indicates how effective addresses are calculated in each addressing mode. In normal mode, the upper 8 bits of the effective address are ignored in order to generate a 16-bit address.

Table 2.13 Effective Address Calculation (1)

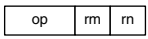
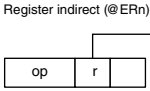
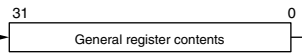
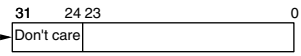
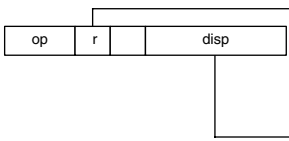
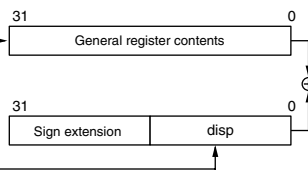
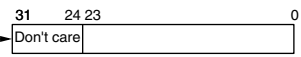
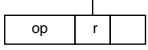
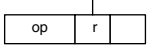
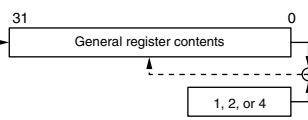
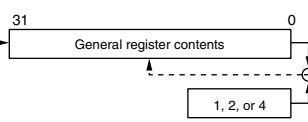
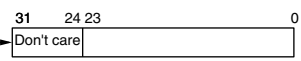
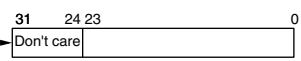
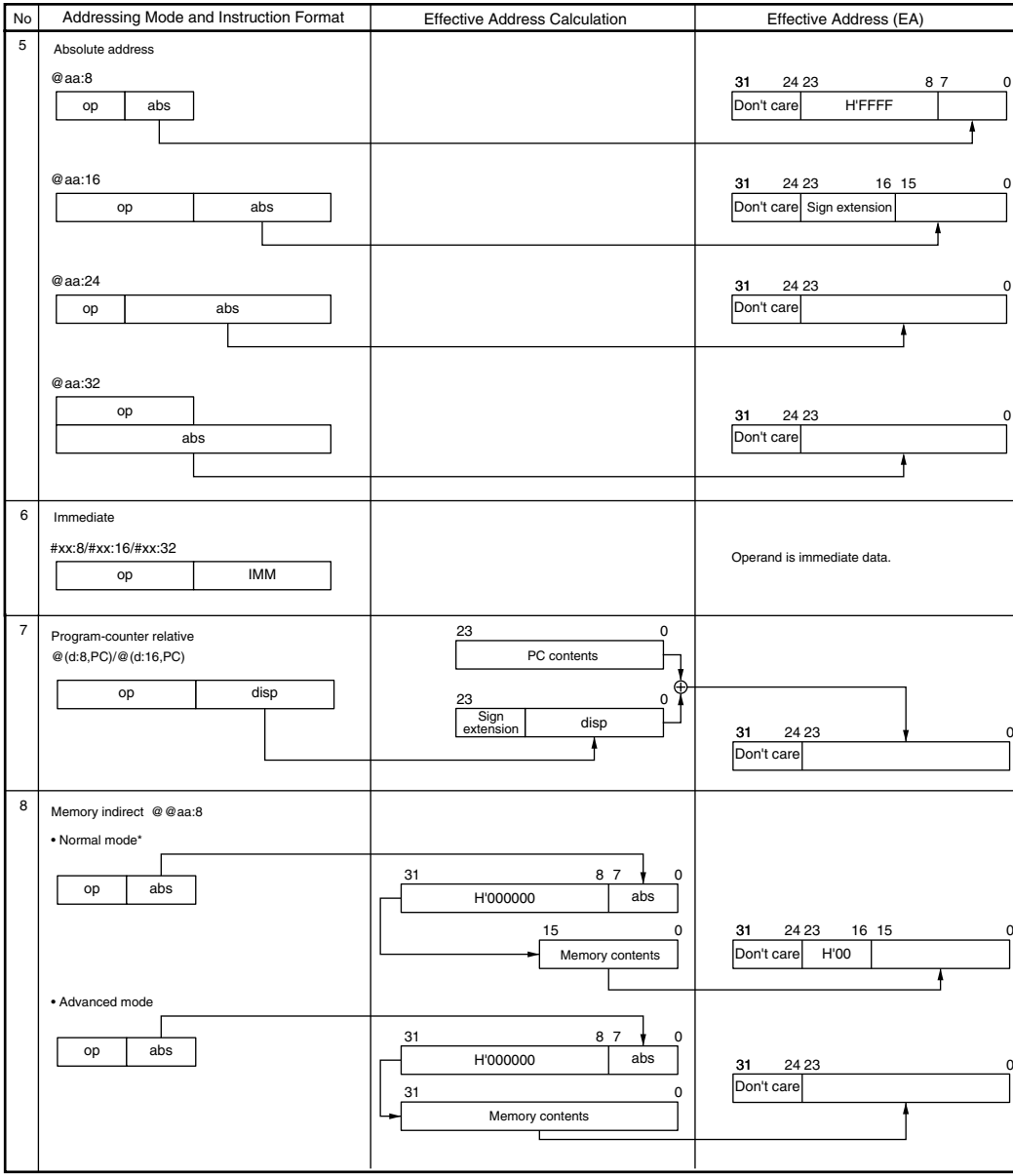
No	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
1	Register direct (Rn) 		Operand is general register contents.
2	Register indirect (@ERn) 		
3	Register indirect with displacement @(d:16,ERn) or @(d:32,ERn) 		
4	Register indirect with post-increment or pre-decrement •Register indirect with post-increment @ERn+  •Register indirect with pre-decrement @-ERn 	  Operand Size / Offset Byte 1 Word 2 Longword 4	 

Table 2.13 Effective Address Calculation (2)



Note: For this LSI, normal mode is not available.

2.8 Processing States

The H8S/2000 CPU has four main processing states: the reset state, exception handling state, program execution state, and program stop state. Figure 2.13 indicates the state transitions.

- Reset state

  In this state the CPU and internal peripheral modules are all initialized and stopped. When the $\overline{\text{RES}}$ input goes low, all current processing stops and the CPU enters the reset state. All interrupts are masked in the reset state. Reset exception handling starts when the $\overline{\text{RES}}$ signal changes from low to high. For details, refer to section 4, Exception Handling.

  The reset state can also be entered by a watchdog timer overflow.

- Exception-handling state

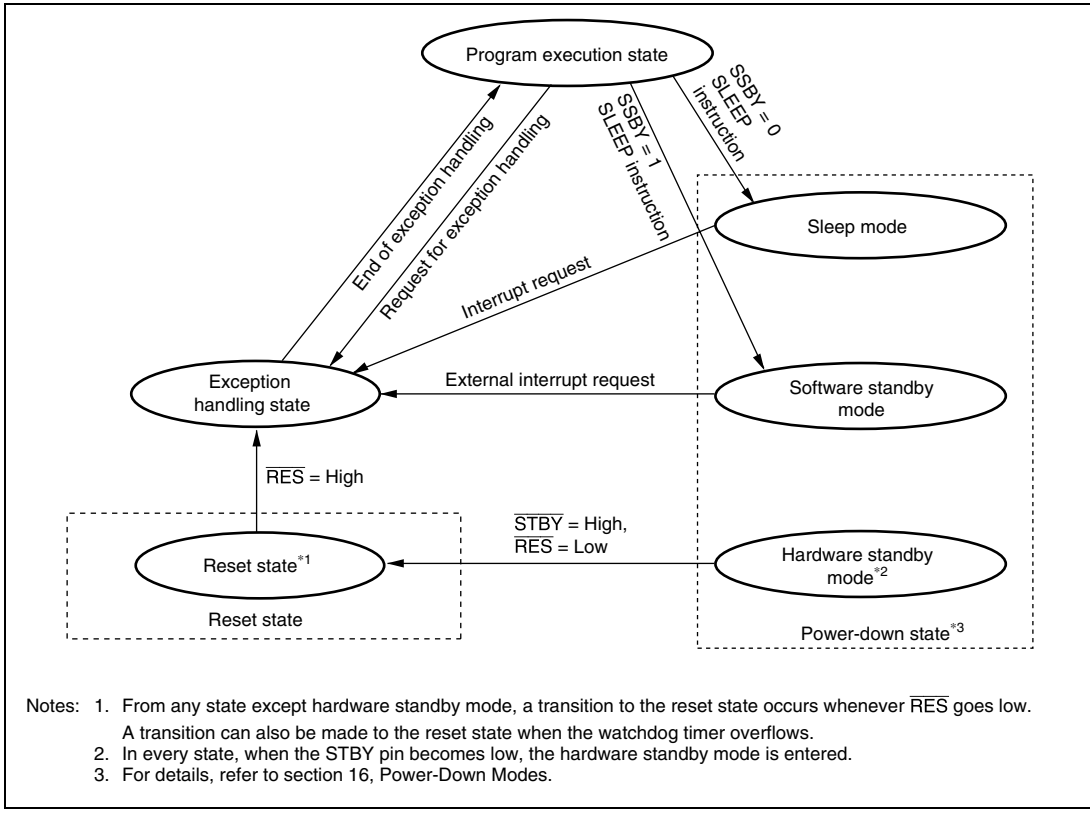
  The exception-handling state is a transient state that occurs when the CPU alters the normal processing flow due to an exception source, such as, a reset, trace, interrupt, or trap instruction. The CPU fetches a start address (vector) from the exception vector table and branches to that address. For further details, refer to section 4, Exception Handling.

- Program execution state

  In this state the CPU executes program instructions in sequence.

- Program stop state

  This is a power-down state in which the CPU stops operating. The program stop state occurs when a SLEEP instruction is executed or the CPU enters hardware standby mode. For details, refer to section 16, Power-Down Modes.



Notes: 1. From any state except hardware standby mode, a transition to the reset state occurs whenever $\overline{RES}$ goes low.
A transition can also be made to the reset state when the watchdog timer overflows.
2. In every state, when the STBY pin becomes low, the hardware standby mode is entered.
3. For details, refer to section 16, Power-Down Modes.

Figure 2.13 State Transitions

2.9 Usage Note

2.9.1 Note on Bit Manipulation Instructions

Bit manipulation instructions such as BSET, BCLR, BNOT, BST, and BIST read data in byte units, perform bit manipulation, and write data in byte units. Thus, care must be taken when these bit manipulation instructions are executed for a register or port including write-only bits.

In addition, the BCLR instruction can be used to clear the flag of an internal I/O register. In this case, if the flag to be cleared has been set by an interrupt processing routine, the flag need not be read before executing the BCLR instruction.

Section 3 MCU Operating Modes

3.1 Operating Mode Selection

This LSI supports single operating mode (mode 2). The operating mode is determined by the setting of the mode pins ($\overline{MD2}$ and MD1). Table 3.1 shows the MCU operating mode selection.

Table 3.1 MCU Operating Mode Selection

MCU Operating Mode	$\overline{MD2}$	MD1	CPU Operating Mode	Description	On-Chip ROM
2	1	1	Advanced mode	Extended mode with on-chip ROM	Enabled
				Single-chip mode	

Mode 2 is single-chip mode after a reset. The CPU can switch to extended mode by setting bit EXPE in MDCR to 1.

Modes 0, 1, 3, 5, and 7 cannot be used in this LSI. Modes 4 and 6 are specific modes. Thus, mode pins should be set to enable mode 2 in normal program execution state. Mode pins should not be changed during operation.

Mode 4 is a boot mode to program/erase the flash memory.

Mode 6 is on-chip emulation mode. This mode is controlled by the on-chip emulator (E10A) via the JTAG interface, and on-chip emulation can be performed.

3.2 Register Descriptions

The following registers are related to the operating mode.

- Mode control register (MDCR)
- System control register (SYSCR)

3.2.1 Mode Control Register (MDCR)

MDCR is used to set an operating mode and to monitor the current operating mode.

Bit	Bit Name	Initial Value	R/W	Description
7	EXPE	0	R/W	Extended Mode Enable Specifies extended mode. 0: Single-chip mode 1: Extended mode
6 to 3	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
2	MDS2	—*	R	Mode Select 2, 1
1	MDS1	—*	R	These bits indicate the input levels at mode pins ($\overline{MD2}$ and MD1) (the current operating mode). Bits MDS2 and MDS1 correspond to $\overline{MD2}$ and MD1, respectively. MDS2 and MDS1 are read-only bits and they cannot be written to. The mode pin ($\overline{MD2}$ and MD1) input levels are latched into these bits when MDCR is read. These latches are canceled by a reset.
0	—	0	R	Reserved This bit is always read as 0 and cannot be modified.

Note: * The initial values are determined by the settings of the $\overline{MD2}$ and MD1 pins.

3.2.2 System Control Register (SYSCR)

SYSCR monitors a reset source, selects the interrupt control mode and the detection edge for NMI, and controls on-chip RAM address space.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R/W	Reserved The initial value should not be changed.
5	INTM1	0	R	These bits select the control mode of the interrupt controller. For details on the interrupt control modes, see section 5.6, Interrupt Control Modes and Interrupt Operation. 00: Interrupt control mode 0 01: Interrupt control mode 1 10: Setting prohibited 11: Setting prohibited
4	INTM0	0	R/W	
3	XRST	1	R	External Reset This bit indicates the reset source. A reset is caused by an external reset input, or when the watchdog timer overflows. 0: A reset is caused when the watchdog timer overflows. 1: A reset is caused by an external reset.
2	NMIEG	0	R/W	NMI Edge Select Selects the valid edge of the NMI interrupt input. 0: An interrupt is requested at the falling edge of NMI input 1: An interrupt is requested at the rising edge of NMI input
1	—	0	R/W	Reserved The initial value should not be changed.
0	RAME	1	R/W	RAM Enable Enables or disables on-chip RAM. The RAME bit is initialized when the reset state is released. 0: On-chip RAM is disabled 1: On-chip RAM is enabled

3.3 Operating Modes

3.3.1 Mode 2

The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is enabled.

After a reset, the LSI is set to single-chip mode. To access an external address space, bit EXPE in MDCR should be set to 1.

In extended mode, ports 6 and 7 function as input ports after a reset. Ports 6 and 7 function as an address bus by setting the AHOE, AMOE, ALOE bits to 1 in the port function control register 1 (PFCR1). Ports 5 and 8 function as a data bus, and parts of ports 9 and A function bus control signals.

3.3.2 Pin Functions

Table 3.2 shows pin functions in operating mode 2.

Table 3.2 Pin Functions in Operating Mode 2

Port		Mode 2
Port 5		I/O port*/Data bus I/O
Port 6		I/O port*/Address bus output
Port 7		I/O port*/Address bus output
Port 8		I/O port*/Data bus I/O
Port 9	P97	I/O port*/Clock I/O
	P96 to P90	Input port*/Control signal output
Port A	PA3 to PA1	I/O port*/Address bus output/Control signal output
	PA0	I/O port*/Address bus output

Legend

*: After a reset

3.4 Address Map

Figure 3.1 shows the address map in each operating mode.

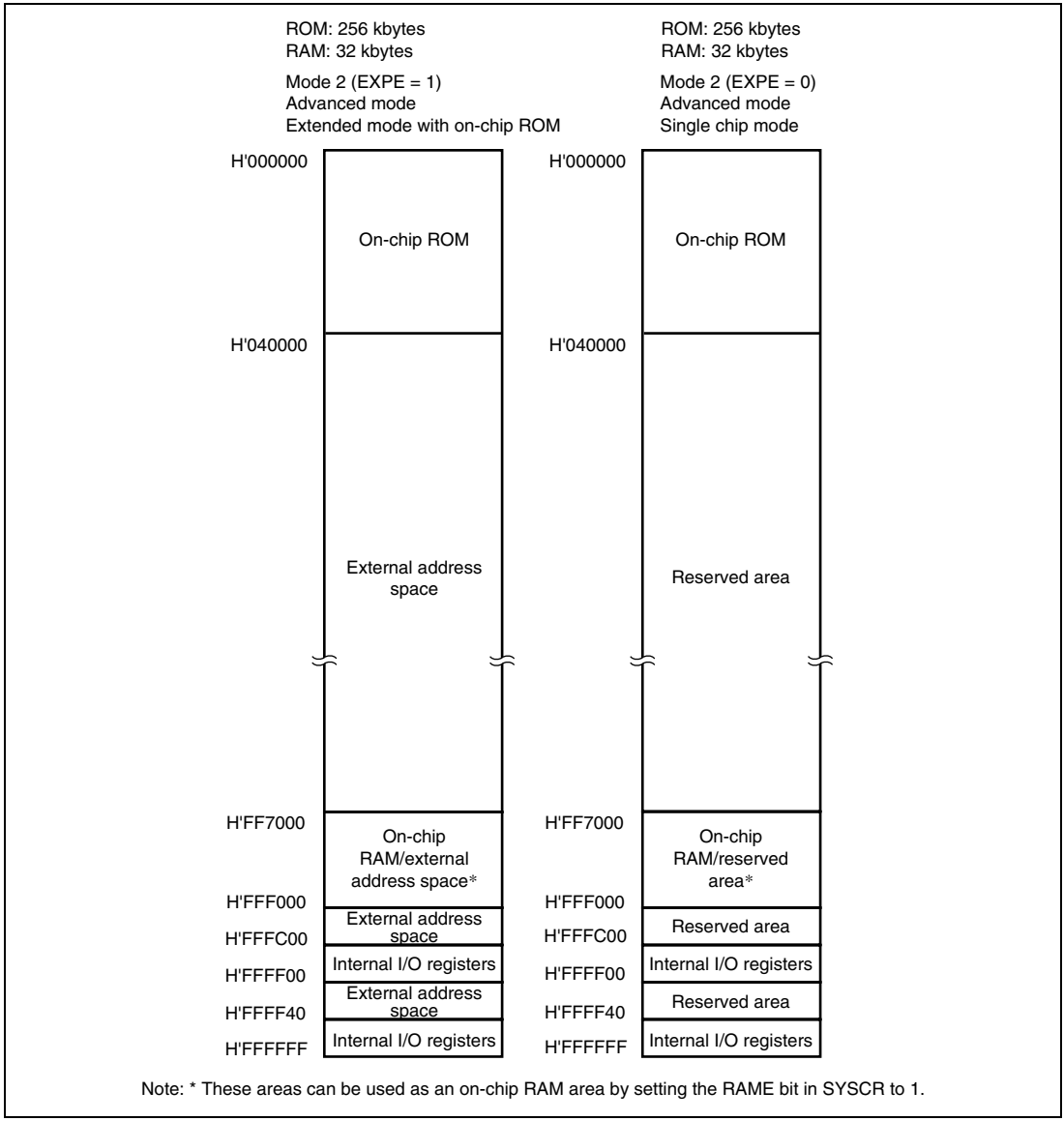


Figure 3.1 Address Map

Section 4 Exception Handling

4.1 Exception Handling Types and Priority

As table 4.1 indicates, exception handling may be caused by a reset, interrupt, or trap instruction. Exception handling is prioritized as shown in table 4.1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority.

Table 4.1 Exception Types and Priority

Priority	Exception Type	Start of Exception Handling
High ▲	Reset	Starts immediately after a low-to-high transition of the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows.
	Interrupt	Starts when execution of the current instruction or exception handling ends, if an interrupt request has been issued. Interrupt detection is not performed on completion of ANDC, ORC, XORC, or LDC instruction execution, or on completion of reset exception handling.
	Trap instruction	Started by execution of a trap (TRAPA) instruction. Trap instruction exception handling requests are accepted at all times in program execution state.
Low		

4.2 Exception Sources and Exception Vector Table

Different vector addresses are assigned to different exception sources. Table 4.2 lists the exception sources and their vector addresses.

Table 4.2 Exception Handling Vector Table

Exception Source	Vector Number	Vector Address Advanced Mode
Reset	0	H'000000 to H'000003
Reserved for system use	1 ⋮ 6	H'000004 to H'000007 ⋮ H'000018 to H'00001B
External interrupt (NMI)	7	H'00001C to H'00001F
Trap instruction (four sources)	8	H'000020 to H'000023
	9	H'000024 to H'000027
	10	H'000028 to H'00002B
	11	H'00002C to H'00002F
Reserved for system use	12 ⋮ 15	H'000030 to H'000033 ⋮ H'00003C to H'00003F
External interrupt IRQ0	16	H'000040 to H'000043
External interrupt IRQ1	17	H'000044 to H'000047
External interrupt IRQ2	18	H'000048 to H'00004B
External interrupt IRQ3	19	H'00004C to H'00004F
External interrupt IRQ4	20	H'000050 to H'000053
External interrupt IRQ5	21	H'000054 to H'000057
External interrupt IRQ6	22	H'000058 to H'00005B
External interrupt IRQ7	23	H'00005C to H'00005F
Internal interrupt*	24 ⋮ 110	H'000060 to H'000063 ⋮ H'0001B8 to H'0001BB

Note: * For details on the internal interrupt vector table, see section 5.5, Interrupt Exception Handling Vector Table.

4.3 Reset

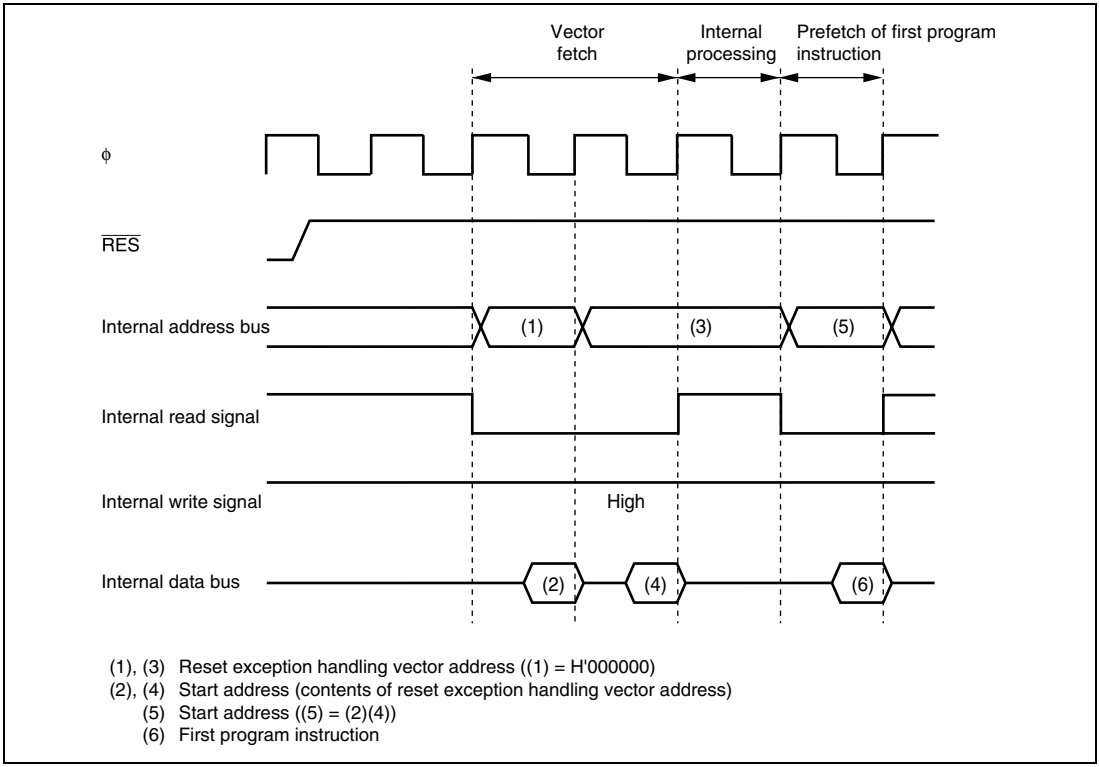
A reset has the highest exception priority. When the $\overline{\text{RES}}$ pin goes low, all processing halts and this LSI enters the reset. To ensure that this LSI is reset, hold the $\overline{\text{RES}}$ pin low for at least 20 ms at power-on. To reset the chip during operation, hold the $\overline{\text{RES}}$ pin low for at least 20 states. A reset initializes the internal state of the CPU and the registers of on-chip peripheral modules. The chip can also be reset by overflow of the watchdog timer. For details, see section 10, Watchdog Timer (WDT).

4.3.1 Reset Exception Handling

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

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

Figure 4.1 shows an example of the reset sequence.



(1), (3) Reset exception handling vector address ((1) = H'000000)
(2), (4) Start address (contents of reset exception handling vector address)
(5) Start address ((5) = (2)(4))
(6) First program instruction

Figure 4.1 Reset Sequence

4.3.2 Interrupts after Reset

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

4.3.3 On-Chip Peripheral Modules after Reset is Cancelled

After a reset is cancelled, the module stop control register (MSTPCR) is initialized, and all modules except the DMAC operate in module stop mode. Therefore, the registers of on-chip peripheral modules cannot be read from or written to. To read from and write to these registers, clear module stop mode.

4.4 Interrupt Exception Handling

Interrupts are controlled by the interrupt controller. The sources to start interrupt exception handling are external interrupt sources (NMI and IRQ7 to IRQ0) and internal interrupt sources from the on-chip peripheral modules. NMI is an interrupt with the highest priority. For details, see section 5, Interrupt Controller.

Interrupt exception handling is conducted as follows:

1. The values in the program counter (PC) and condition code register (CCR) are saved to the stack.
2. A vector address corresponding to the interrupt source is generated, the start address is loaded from the vector table to the PC, and program execution begins from that address.

4.5 Trap Instruction Exception Handling

Trap instruction exception handling starts when a TRAPA instruction is executed. Trap instruction exception handling can be executed at all times in the program execution state.

Trap instruction exception handling is conducted as follows:

1. The values in the program counter (PC) and condition code register (CCR) are saved to the stack.
2. A vector address corresponding to the interrupt source is generated, the start address is loaded from the vector table to the PC, and program execution starts from that address.

The TRAPA instruction fetches a start address from a vector table entry corresponding to a vector number from 0 to 3, as specified in the instruction code.

Table 4.3 shows the status of CCR after execution of trap instruction exception handling.

Table 4.3 Status of CCR after Trap Instruction Exception Handling

	CCR	
Interrupt Control Mode	**I**	**UI**
0	Set to 1	Retains value prior to execution
1	Set to 1	Set to 1

4.6 Stack Status after Exception Handling

Figure 4.2 shows the stack after completion of trap instruction exception handling and interrupt exception handling.

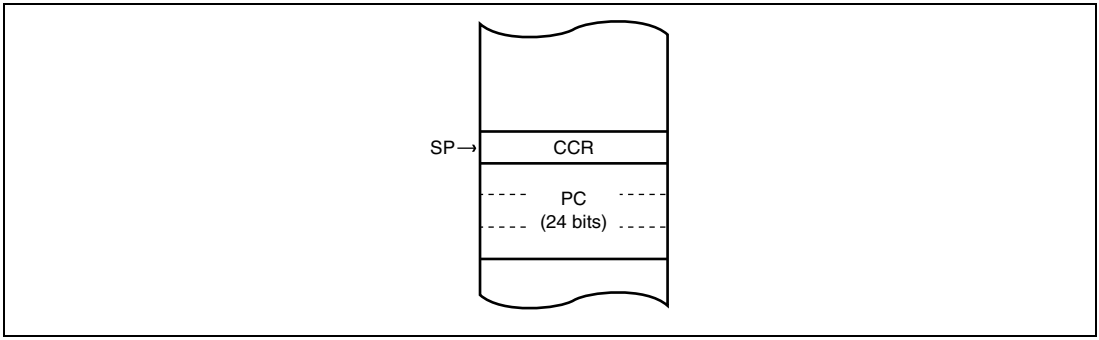


Figure 4.2 Stack Status after Exception Handling

4.7 Usage Note

When accessing word data or longword data, this LSI assumes that the lowest address bit is 0. The stack should always be accessed in words or longwords, and the value of the stack pointer (SP: ER7) should always be kept even.

Use the following instructions to save registers:

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

Use the following instructions to restore registers:

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

Setting SP to an odd value may lead to a malfunction. Figure 4.3 shows an operation example when the SP value is odd.

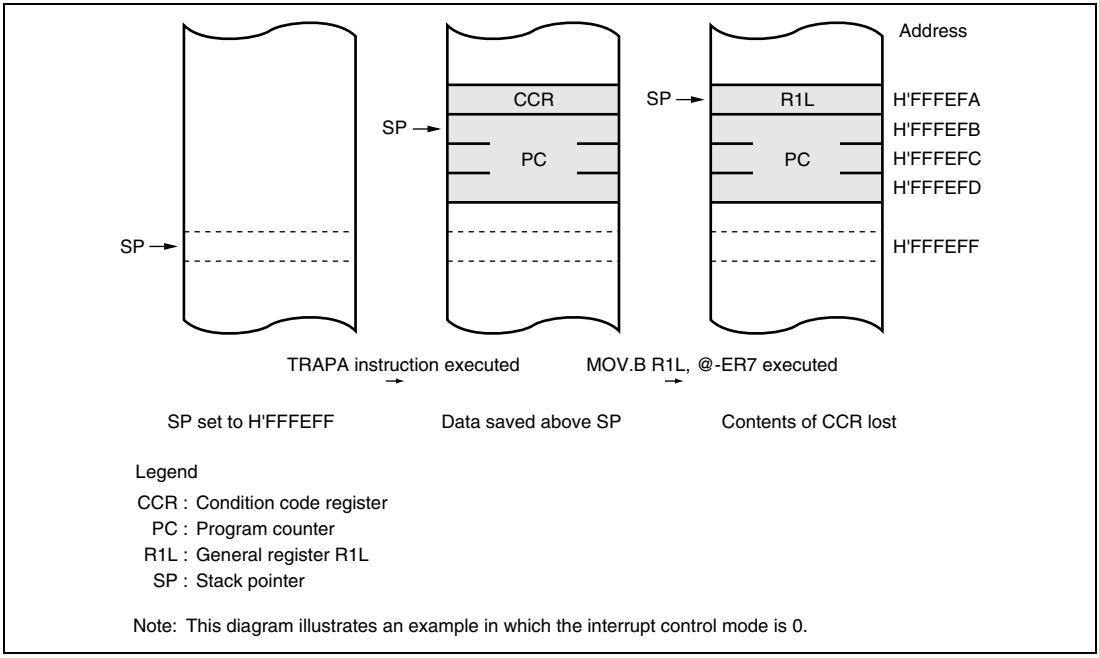


Figure 4.3 Operation when SP Value is Odd

Section 5 Interrupt Controller

5.1 Features

- Two interrupt control modes

  Any of two interrupt control modes can be set by means of the INTM1 and INTM0 bits in the system control register (SYSCR).
- Priorities settable with ICR

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

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

  NMI is the highest-priority interrupt, and is accepted at all times. Rising edge or falling edge detection can be selected for NMI. Falling-edge, rising-edge, or both-edge detection, or level sensing, can be selected for $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$.

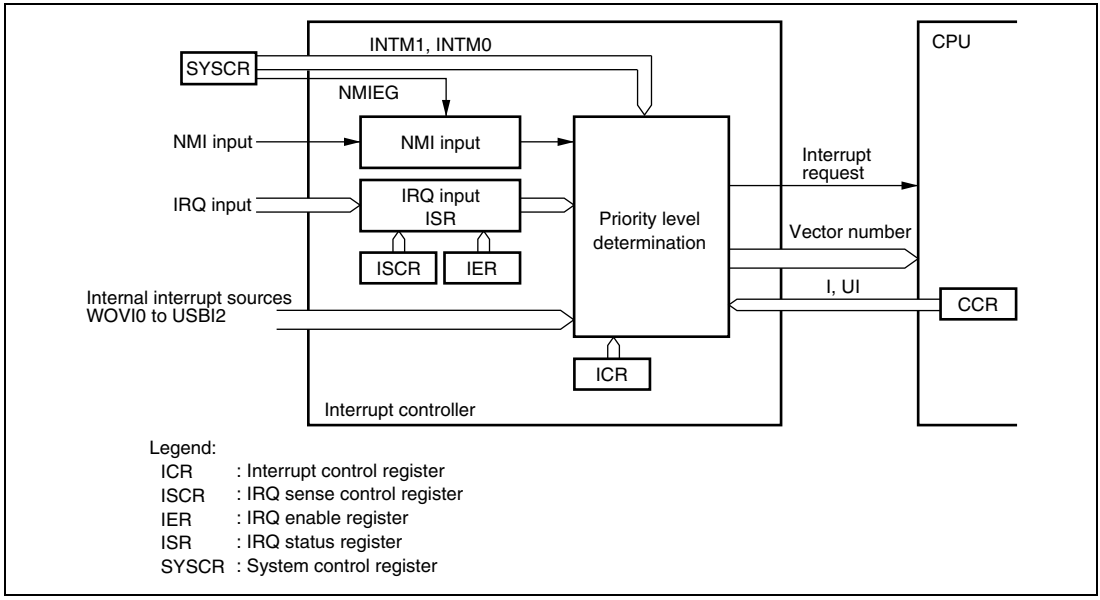


Figure 5.1 Block Diagram of Interrupt Controller

5.2 Input/Output Pins

Table 5.1 summarizes the pins of the interrupt controller.

Table 5.1 Pin Configuration

Symbol	I/O	Function
NMI	Input	Nonmaskable external interrupt Rising edge or falling edge can be selected
$\overline{IRQ7}$ to $\overline{IRQ0}$	Input	Maskable external interrupts Rising edge, falling edge, or both edges, or level sensing, can be selected individually for each pin. Whether the IRQn interrupt is input from the $\overline{IRQn}$ or ($\overline{IRQn}$) is selectable. (n = 7 to 0)

5.3 Register Descriptions

The interrupt controller has the following registers. For details on the system control register (SYSCR), see section 3.2.2, System Control Register (SYSCR), and for details on the IRQ sense port select register (ISSR), see section 8.11.2, IRQ Sense Port Select Register (ISSR).

- Interrupt control registers A to C (ICRA to ICRC)
- Address break control register (ABRKCR)
- Break address registers A to C (PBARA to PBARC)
- IRQ sense control registers H, L (ISCRH, ISCRL)
- IRQ enable register (IER)
- IRQ status register (ISR)

5.3.1 Interrupt Control Registers A to C (ICRA to ICRC)

The ICR registers set interrupt control levels for interrupts other than NMI.

The correspondence between interrupt sources and ICRA to ICRC settings is shown in table 5.2.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	ICRn7 to IRCn0	All 0	R/W	Interrupt Control Level 0: Corresponding interrupt source is interrupt control level 0 (no priority) 1: Corresponding interrupt source is interrupt control level 1 (priority)

Note: n: A to C

Table 5.2 Correspondence between Interrupt Source and ICR

		Register		
Bit	**Bit Name**	**ICRA**	**ICRB**	**ICRC**
7	ICRn7	IRQ0	—	SCI
6	ICRn6	IRQ1	Suspend recover interrupt	—
5	ICRn5	IRQ2, IRQ3	DMAC	—
4	ICRn4	IRQ4, IRQ5	—	—
3	ICRn3	IRQ6, IRQ7	TMR_0	—
2	ICRn2	—	TMR_1	—
1	ICRn1	WDT	—	—
0	ICRn0	Refresh timer	—	USB2

Note: n: A to C

—: Reserved. The write value should always be 0.

5.3.2 Address Break Control Register (ABRKCR)

ABRKCR controls the address breaks. When both the CMF flag and BIE flag are set to 1, an address break is requested.

Bit	Bit Name	Initial Value	R/W	Description
7	CMIF	Undefined	R/W	Condition Match Flag Address break source flag. Indicates that an address specified by BARA to BARC is prefetched. [Clearing condition] When an exception handling is executed for an address break interrupt. [Setting condition] When an address specified by BARA to BARC is prefetched while the BIE flag is set to 1.
6 to 1	—	All 0	R	Reserved These bits are always read as 0 and cannot be modified.
0	BIE	0	R/W	Break Interrupt Enable Enables or disables address break. 0: Disabled 1: Enabled

5.3.3 Break Address Registers A to C (PBARA to PBARC)

The PBAR registers specify an address that is to be a break address. An address in which the first byte of an instruction exists should be set as a break address.

- PBARA

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	A23 to A16	All 0	R/W	Addresses 23 to 16 The A23 to A16 bits are compared with A23 to A16 in the internal address bus.

- PBARB

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	A15 to A8	All 0	R/W	Addresses 15 to 8 The A15 to A8 bits are compared with A15 to A8 in the internal address bus.

- **PBARC**

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	A7 to A1	All 0	R/W	Addresses 7 to 1 The A7 to A1 bits are compared with A7 to A1 in the internal address bus.
0	—	0	R	Reserved This bit is always read as 0 and cannot be modified.

5.3.4 IRQ Sense Control Registers H, L (ISCRH, ISCRL)

The ISCR registers select the source that generates an interrupt request at pins $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$.

The $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$ pins can be switched to input pins by setting the IRQ sense port select register (ISSR).

- ISCRH

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ7SCB	0	R/W	IRQn Sense Control B IRQn Sense Control A 00: Interrupt request generated at low level of $\overline{\text{IRQn}}$ input 01: Interrupt request generated at falling edge of $\overline{\text{IRQn}}$ input 10: Interrupt request generated at rising edge of $\overline{\text{IRQn}}$ input 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQn}}$ input (n = 7 to 4)
6	IRQ7SCA	0	R/W	
5	IRQ6SCB	0	R/W	
4	IRQ6SCA	0	R/W	
3	IRQ5SCB	0	R/W	
2	IRQ5SCA	0	R/W	
1	IRQ4SCB	0	R/W	
0	IRQ4SCA	0	R/W	

- ISCRL

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ3SCB	0	R/W	IRQn Sense Control B IRQn Sense Control A 00: Interrupt request generated at low level of $\overline{\text{IRQn}}$ input 01: Interrupt request generated at falling edge of $\overline{\text{IRQn}}$ input 10: Interrupt request generated at rising edge of $\overline{\text{IRQn}}$ input 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQn}}$ input (n = 3 to 0)
6	IRQ3SCA	0	R/W	
5	IRQ2SCB	0	R/W	
4	IRQ2SCA	0	R/W	
3	IRQ1SCB	0	R/W	
2	IRQ1SCA	0	R/W	
1	IRQ0SCB	0	R/W	
0	IRQ0SCA	0	R/W	

5.3.5 IRQ Enable Register (IER)

IER controls the enabling and disabling of interrupt requests IRQ7 to IRQ0.

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ7E	0	R/W	IRQn Enable (n = 7 to 0)
6	IRQ6E	0	R/W	The IRQn interrupt request is enabled when this bit is 1.
5	IRQ5E	0	R/W	
4	IRQ4E	0	R/W	
3	IRQ3E	0	R/W	
2	IRQ2E	0	R/W	
1	IRQ1E	0	R/W	
0	IRQ0E	0	R/W	

5.3.6 IRQ Status Register (ISR)

ISR is a flag register that indicates the status of IRQ7 to IRQ0 interrupt requests.

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ7F	0	R/W	[Setting condition]
6	IRQ6F	0	R/W	When the interrupt source selected by the ISCR registers occurs
5	IRQ5F	0	R/W	
4	IRQ4F	0	R/W	[Clearing conditions]
3	IRQ3F	0	R/W	• When reading IRQnF flag when IRQnF = 1, then writing 0 to IRQnF flag
2	IRQ2F	0	R/W	• When interrupt exception handling is executed when low-level detection is set and $\overline{\text{IRQn}}$ input is high
1	IRQ1F	0	R/W	• When IRQn interrupt exception handling is executed when falling-edge, rising-edge, or both-edge detection is set
0	IRQ0F	0	R/W	(n = 7 to 0)

5.4 Interrupt Sources

5.4.1 External Interrupts

There are two external interrupts: NMI and IRQ7 to IRQ0. These interrupts can be used to restore this LSI from software standby mode.

NMI Interrupt: NMI is the highest-priority interrupt, and is always accepted by the CPU regardless of the interrupt control mode or the status of the CPU interrupt mask bits. The NMIEG bit in SYSCR can be used to select whether an interrupt is requested at a rising edge or a falling edge on the NMI pin.

IRQ7 to IRQ0 Interrupts: Interrupts IRQ7 to IRQ0 are requested by an input signal at pins $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$. Interrupts IRQ7 to IRQ0 have the following features:

- The interrupt exception handling for interrupt requests IRQ7 to IRQ0 can be started at an independent vector address.
- Using ISCR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, at pins $\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$.
- Enabling or disabling of interrupt requests IRQ7 to IRQ0 can be selected with IER.
- The status of interrupt requests IRQ7 to IRQ0 is indicated in ISR. ISR flags can be cleared to 0 by software.

The detection of IRQ7 to IRQ0 interrupts does not depend on whether the relevant pin has been set for input or output. However, when a pin is used as an external interrupt input pin, do not clear the corresponding port DDR to 0 to use the pin as an I/O pin for another function.

A block diagram of interrupts IRQ7 to IRQ0 is shown in figure 5.2.

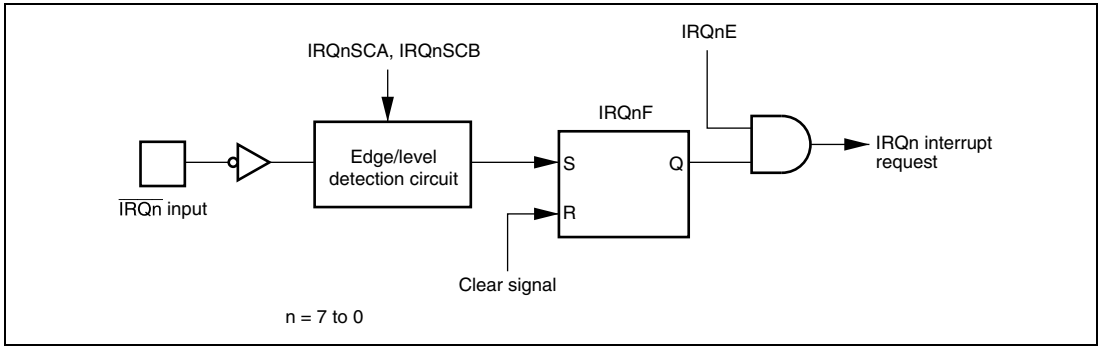


Figure 5.2 Block Diagram of Interrupts IRQ7 to IRQ0

5.4.2 Internal Interrupts

Internal interrupts issued from the on-chip peripheral modules have the following features:

- For each on-chip peripheral module there are flags that indicate the interrupt request status, and enable bits that individually select enabling or disabling of these interrupts. When the enable bit for a particular interrupt source is set to 1, an interrupt request is sent to the interrupt controller.
- The control level for each interrupt can be set by ICR.

5.5 Interrupt Exception Handling Vector Table

Table 5.3 lists interrupt exception handling sources, vector addresses, and interrupt priorities. For default priorities, the lower the vector number, the higher the priority. Modules set at the same priority will conform to their default priorities. Priorities within a module are fixed.

An interrupt control level can be specified for a module to which an ICR bit is assigned. Interrupt requests from modules that are set to interrupt control level 1 (priority) by the ICR bit setting and the I and UI bits in CCR are given priority and processed before interrupt requests from modules that are set to interrupt control level 0 (no priority).

Table 5.3 Interrupt Sources, Vector Addresses, and Interrupt Priorities

Origin of Interrupt Source	Name	Vector Number	Vector Address	ICR	Priority
External pin	NMI	7	H'00001C	—	High
	IRQ0	16	H'000040	ICRA7	↑
	IRQ1	17	H'000044	ICRA6	
	IRQ2 IRQ3	18 19	H'000048 H'00004C	ICRA5	
	IRQ4 IRQ5	20 21	H'000050 H'000054	ICRA4	
	IRQ6 IRQ7	22 23	H'000058 H'00005C	ICRA3	
—	Reserved for system use	24	H'000060	—	
WDT	WOVI0 (Interval timer)	25	H'000064	ICRA1	
Refresh timer	CMI (Compare match)	26	H'000068	ICRA0	
—	Address break	27	H'00006C	—	
—	Reserved for system use	28 29	H'000070 H'000074	—	Low

Origin of Interrupt Source	Name	Vector Number	Vector Address	ICR	Priority
External pin	SUSRI (Suspend recover interrupt)	30	H'000078	ICRB6	High
—	Reserved for system use	31 to 33	H'00007C to H'000084	—	
DMAC	DEND0 DEND1 DEND2 DEND3	34 35 36 37	H'000088 H'00008C H'000090 H'000094	ICRB5	
—	Reserved for system use	38 to 63	H'000098 to H'0000FC	—	
TMR_0	CMIA0 (Compare match A) CMIB0 (Compare match B) OVI0 (Overflow) Reserved for system use	64 65 66 67	H'000100 H'000104 H'000108 H'00010C	ICRB3	
TMR_1	CMIA1 (Compare match A) CMIB1 (Compare match B) OVI1 (Overflow) Reserved for system use	68 69 70 71	H'000110 H'000114 H'000118 H'00011C	ICRB2	
—	Reserved for system use	72 to 79	H'000120 to H'00013C	—	
SCI	ERI0 (Reception error 0) RXI0 (Reception completion 0) TXI0 (Transmission data empty 0) TEI0 (Transmission end 0)	80 81 82 83	H'000140 H'000144 H'000148 H'00014C	ICRC7	
—	Reserved for system use	84 to 107	H'000150 to H'0001AC	—	
USB2	USBI0 USBI1 USBI2 Reserved for system use	108 109 110 111	H'0001B0 H'0001B4 H'0001B8 H'0001BC	ICRC0	Low

5.6 Interrupt Control Modes and Interrupt Operation

The interrupt controller has two modes: Interrupt control mode 0 and interrupt control mode 1. Interrupt operations differ depending on the interrupt control mode. NMI interrupts and address break interrupts are always accepted except for in reset state or in hardware standby mode. The interrupt control mode is selected by SYSCR. Table 5.4 shows the interrupt control modes.

Table 5.4 Interrupt Control Modes

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

Figure 5.3 shows a block diagram of the priority decision circuit.

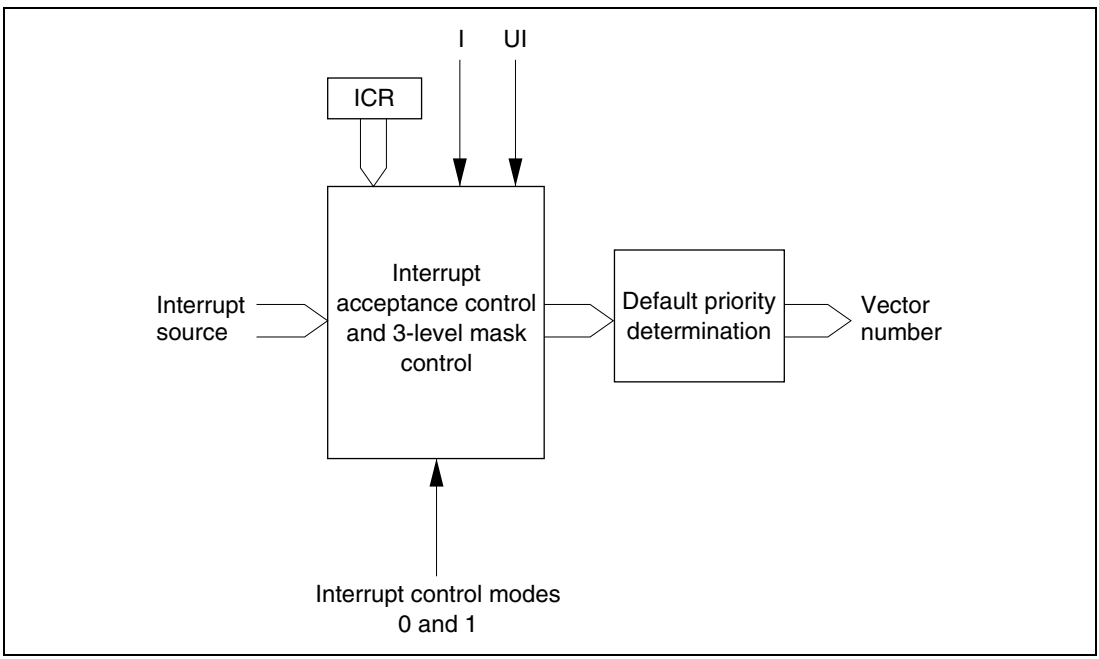


Figure 5.3 Block Diagram of Interrupt Control Operation

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

Table 5.5 shows the interrupts that can be accepted in each interrupt control mode.

Table 5.5 Interrupts Acceptable in Each Interrupt Control Mode

Interrupt Control Mode	I Bit	UI Bit	NMI, Address Break	Peripheral Module Interrupt
0	0	—	O	O (All interrupts)*
	1	—	O	X
1	0	—	O	O (All interrupts)*
	1	0	O	O (Interrupts with ICR = 1)
		1	O	X

Legend

—: Don't care

Note: * Interrupt control level 1 has priority.

Default Priority Determination: The priority is determined for the selected interrupt, and a vector number is generated.

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

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

Table 5.6 shows operations and control signal functions in each interrupt control mode.

Table 5.6 Operations and Control Signal Functions in Each Interrupt Control Mode

Interrupt	Setting		Interrupt Acceptance Control 3-Level Control				Default Priority	
Control Mode	INTM1	INTM0		I	UI	ICR	Determination	T (Trace)
0	0	0	O	IM	—	PR	O	—
1		1	O	IM	IM	PR	O	—

Legend

O : Interrupt operation control performed

IM : Used as an interrupt mask bit

PR : Sets priority

— : Not used

5.6.1 Interrupt Control Mode 0

In interrupt control mode 0, interrupt requests other than NMI and address break are masked by ICR and the I bit of the CCR in the CPU. The interrupt requests are held pending when the I bit is set to 1. Figure 5.4 shows a flowchart of the interrupt acceptance operation.

1. If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
2. According to the interrupt control level specified in ICR, the interrupt controller only accepts an interrupt request with interrupt control level 1 (priority), and holds pending an interrupt request with interrupt control level 0 (no priority). If several interrupt requests are issued, an interrupt request with the highest priority is accepted according to the priority order, an interrupt handling is requested to the CPU, and other interrupt requests are held pending.
3. If the I bit in CCR is set to 1, only NMI and address break interrupts are accepted by the interrupt controller, and other interrupt requests are held pending. If the I bit is cleared to 0, any interrupt request is accepted.
4. When the CPU accepts an interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC and CCR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.
6. Next, the I bit in CCR is set to 1. This masks all interrupts except for NMI and address break interrupts.
7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.

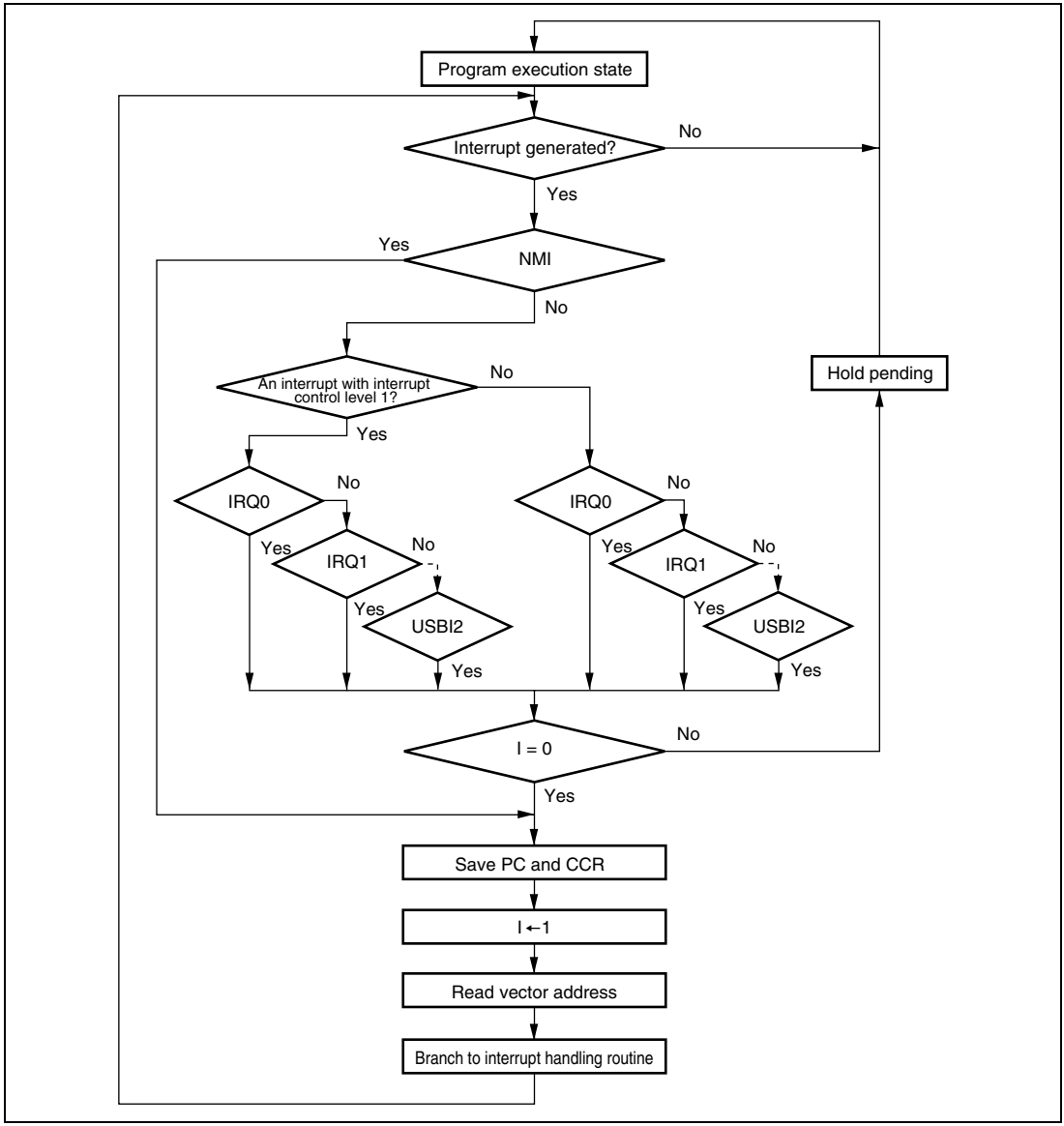


Figure 5.4 Flowchart of Procedure up to Interrupt Acceptance in Interrupt Control Mode 0

5.6.2 Interrupt Control Mode 1

In interrupt control mode 1, mask control is applied to three levels for IRQ and on-chip peripheral module interrupt requests by comparing the I and UI bits in CCR in the CPU, and the ICR setting. The interrupt requests are held pending when the I bit is set to 1.

1. An interrupt request with interrupt control level 0 is accepted when the I bit in CCR is cleared to 0. When the I bit is set to 1, the interrupt request is held pending.
2. An interrupt request with interrupt control level 1 is accepted when the I bit or UI bit in CCR is cleared to 0. When both I and UI bits are set to 1, the interrupt request is held pending.

For instance, the state transition when the interrupt enable bit corresponding to each interrupt is set to 1, and ICRA to ICRC are set to H'20, H'00, and H'00, respectively (IRQ2 and IRQ3 interrupts are set to interrupt control level 1, and other interrupts are set to interrupt control level 0) is shown below. Figure 5.5 shows a state transition diagram.

1. All interrupt requests are accepted when I = 0. (Priority order: NMI > IRQ2 > IRQ3 > IRQ0 > IRQ1 > address break …)
2. Only NMI, IRQ2, IRQ3, and address break interrupt requests are accepted when I = 1 and UI = 0.
3. Only NMI and address break interrupt requests are accepted when I = 1 and UI = 1.

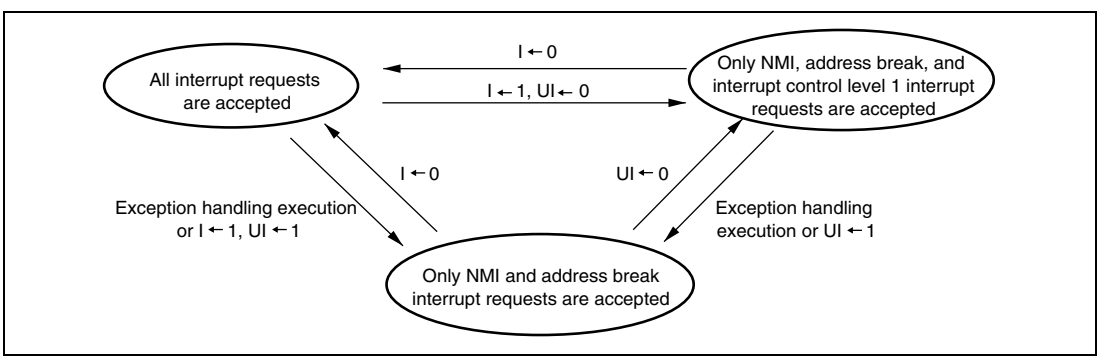


Figure 5.5 State Transition in Interrupt Control Mode 1

Figure 5.6 shows a flowchart of the interrupt acceptance operation.

1. If an interrupt source occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
2. According to the interrupt control level specified in ICR, the interrupt controller only accepts an interrupt request with interrupt control level 1 (priority), and holds pending an interrupt request with interrupt control level 0 (no priority). If several interrupt requests are issued, an interrupt request with the highest priority is accepted according to the priority order, an interrupt handling is requested to the CPU, and other interrupt requests are held pending.
3. An interrupt request with interrupt control level 1 is accepted when the I bit is cleared to 0, or when the I bit is set to 1 while the UI bit is cleared to 0.

   An interrupt request with interrupt control level 0 is accepted when the I bit is cleared to 0. When the I bit is set to 1, only an NMI or address break interrupt request is accepted, and other interrupts are held pending.

   When both the I and UI bits are set to 1, only an NMI or address break interrupt request is accepted, and other interrupts are held pending.

   When the I bit is cleared to 0, the UI bit is not affected.
4. When the CPU accepts an interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC and CCR are saved to the stack area by interrupt exception handling. The PC saved on the stack shows the address of the first instruction to be executed after returning from the interrupt handling routine.
6. The I and UI bits in CCR are set to 1. This masks all interrupts except for an NMI or address break interrupt.
7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.

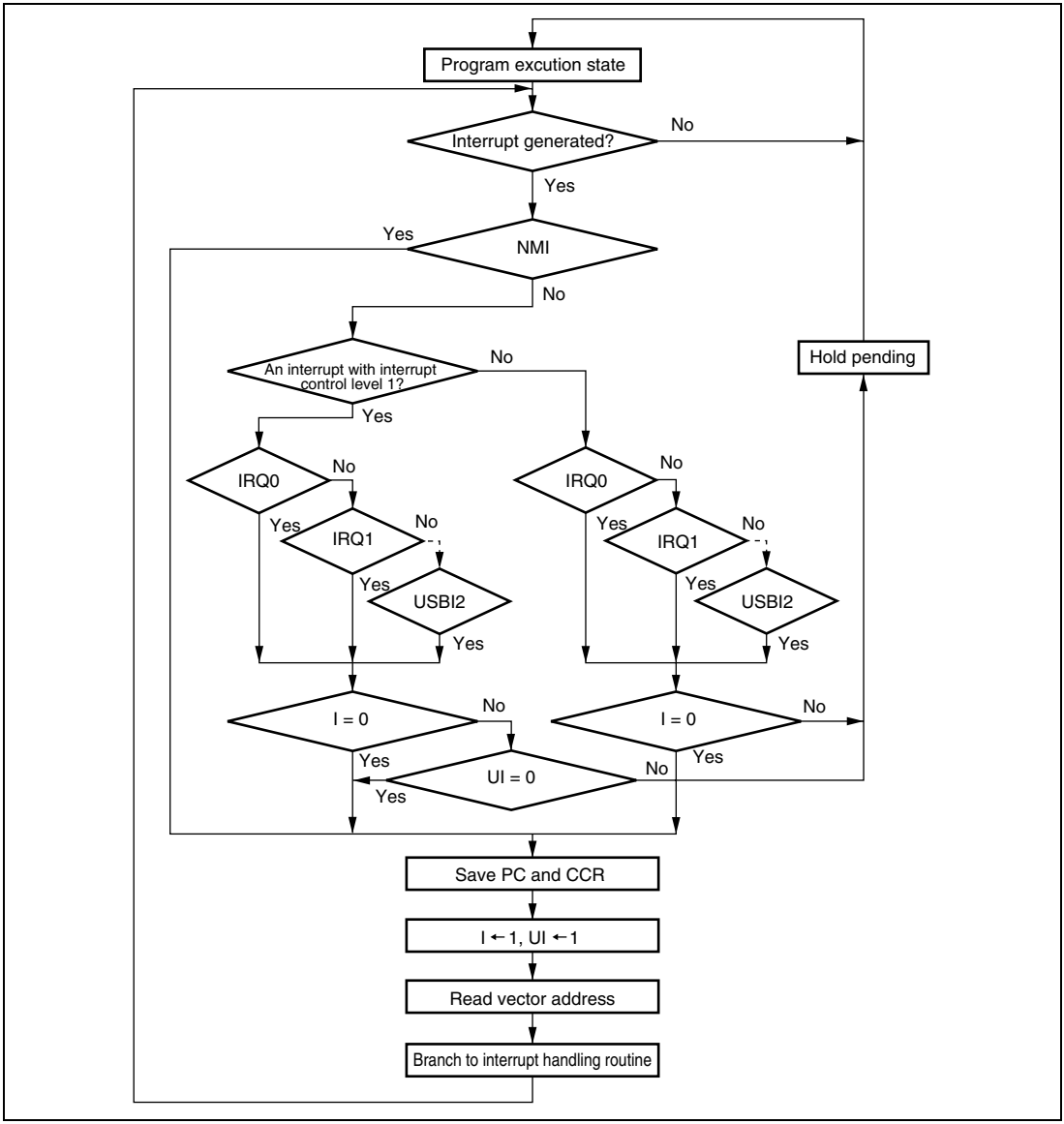


Figure 5.6 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 1

5.6.3 Interrupt Exception Handling Sequence

Figure 5.7 shows the interrupt exception handling sequence. The example shown is for the case where interrupt control mode 0 is set in advanced mode, and the program area and stack area are in on-chip memory.

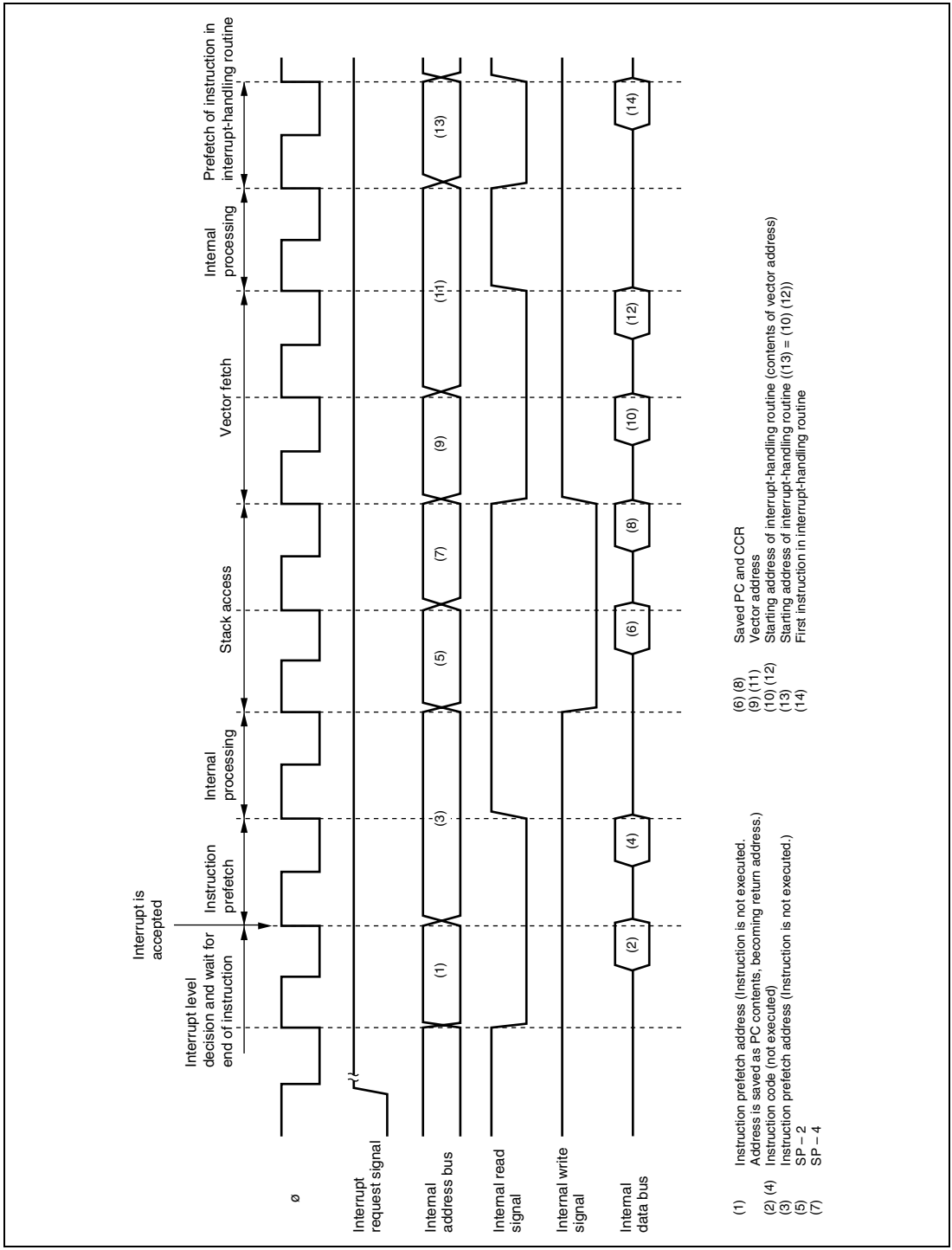


Figure 5.7 Interrupt Exception Handling

5.6.4 Interrupt Response Times

Table 5.7 shows interrupt response times – the intervals between generation of an interrupt request and execution of the first instruction in the interrupt handling routine. The execution status symbols used in table 5.7 are explained in table 5.8.

Table 5.7 Interrupt Response Times

No.	Execution Status	Advanced Mode
1	Interrupt priority determination*[1]	3
2	Number of wait states until executing instruction ends*[2]	1 to (19 + 2·S_I)
3	PC, CCR stack save	2·S_K
4	Vector fetch	2·S_I
5	Instruction fetch*[3]	2·S_I
6	Internal processing*[4]	2
	Total (using on-chip memory)	12 to 32

Notes: 1. Two states in case of internal interrupt.
2. Refers to MULXS and DIVXS instructions.
3. Prefetch after interrupt acceptance and prefetch of interrupt handling routine.
4. Internal processing after interrupt acceptance and internal processing after vector fetch.

Table 5.8 Number of States in Interrupt Handling Routine Execution Status

	Object of Access				
		External Device			
		8-Bit Bus		16-Bit Bus	
Symbol	Internal Memory	2-State Access	3-State Access	2-State Access	3-State Access
Instruction fetch S_I	1	4	6 + 2m	2	3 + m
Branch address read S_J					
Stack manipulation S_K					

Legend
m: Number of wait states in external device access.

5.7 Usage Notes

5.7.1 Conflict between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to disable interrupt requests, the disabling becomes effective after execution of the instruction. When an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, and if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request of higher priority than that interrupt, interrupt exception handling will be executed for the higher-priority interrupt, and the lower-priority interrupt will be ignored. The same rule is also applied when an interrupt source flag is cleared to 0. Figure 5.8 shows an example in which the CMIEA bit in the TMR's TCR register is cleared to 0.

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

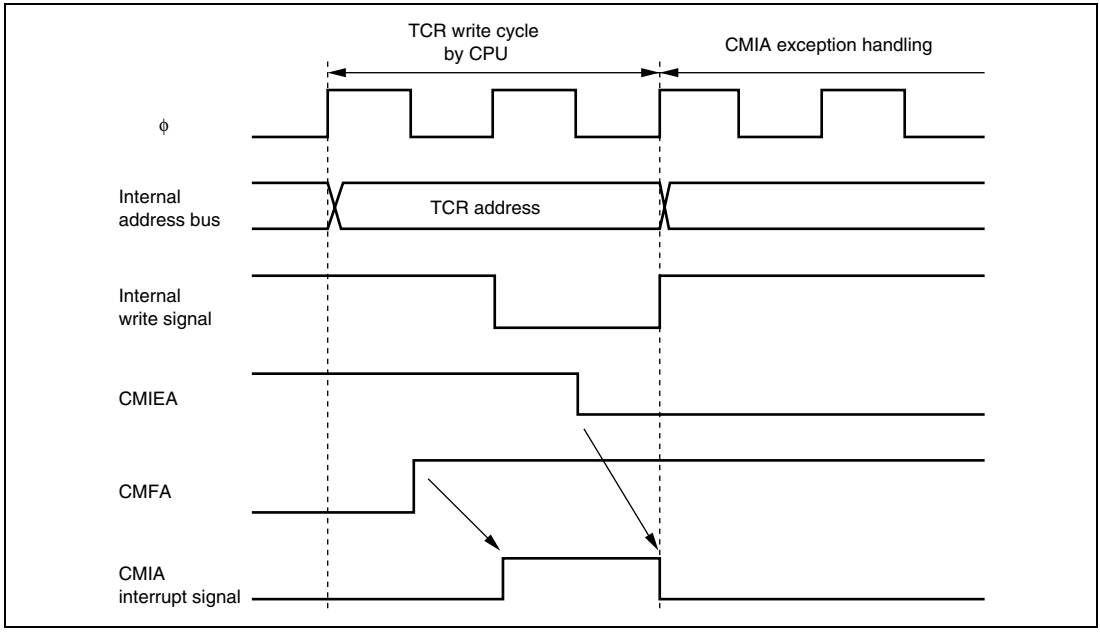


Figure 5.8 Conflict between Interrupt Generation and Disabling

5.7.2 Instructions that Disable Interrupts

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

5.7.3 Interrupts during Execution of EEPMOV Instruction

Interrupt operation differs between the EEPMOV.B instruction and the EEPMOV.W instruction.

With the EEPMOV.B instruction, an interrupt request (including NMI) issued during the transfer is not accepted until the move is completed.

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

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

Section 6 Bus Controller (BSC)

This LSI has an on-chip bus controller (BSC) that manages the external address space divided into four areas. The bus specifications such as the bus width and number of access states can be set independently for each area. Therefore multiple memories and external I/O devices can be connected easily to each area.

The bus controller also has a bus arbitration function, and controls the operation of the bus masters—the CPU and DMA controller (DMAC). A block diagram of the bus controller is shown in figure 6.1.

6.1 Features

- Manages external address space in area units
  Manages the external address space divided into four areas of 2/10 Mbytes
  Bus specifications can be set independently for each area
  DRAM interface can be set
- Basic bus interface
  Chip select signals ($\overline{CS0}$ to $\overline{CS3}$) can be output for areas 0 to 3
  8-bit access or 16-bit access can be selected for each area
  2-state access or 3-state access can be selected for each area
  Program wait states can be inserted for each area
  CS assertion period extend states can be inserted for each area
- DRAM interface
  DRAM interface can be set for area 2
  Multiplex output of row/column address (8/9/10/11 bits)
  Byte and word control by CAS2 method
  Burst operation can be performed in high-speed page mode
  Tp cycle insertion to ensure RAS precharge time
  CAS before RAS refresh (CBR refresh) or self refresh can be selected
- Idle cycle insertion
  Idle cycles can be inserted when external read cycles between different areas are continued
  Idle cycles can be inserted when write cycles are continued after a read cycle
  Idle cycles can be inserted when accesses between different areas are continued
- Write buffer function
  An external write cycle and internal access can be executed in parallel
  DMAC single address mode and internal access can be executed in parallel
- Bus arbitration function

Includes a bus arbiter that arbitrates bus mastership between the CPU and DMAC

- Others

  A refresh counter (refresh timer) can be used as an interval timer

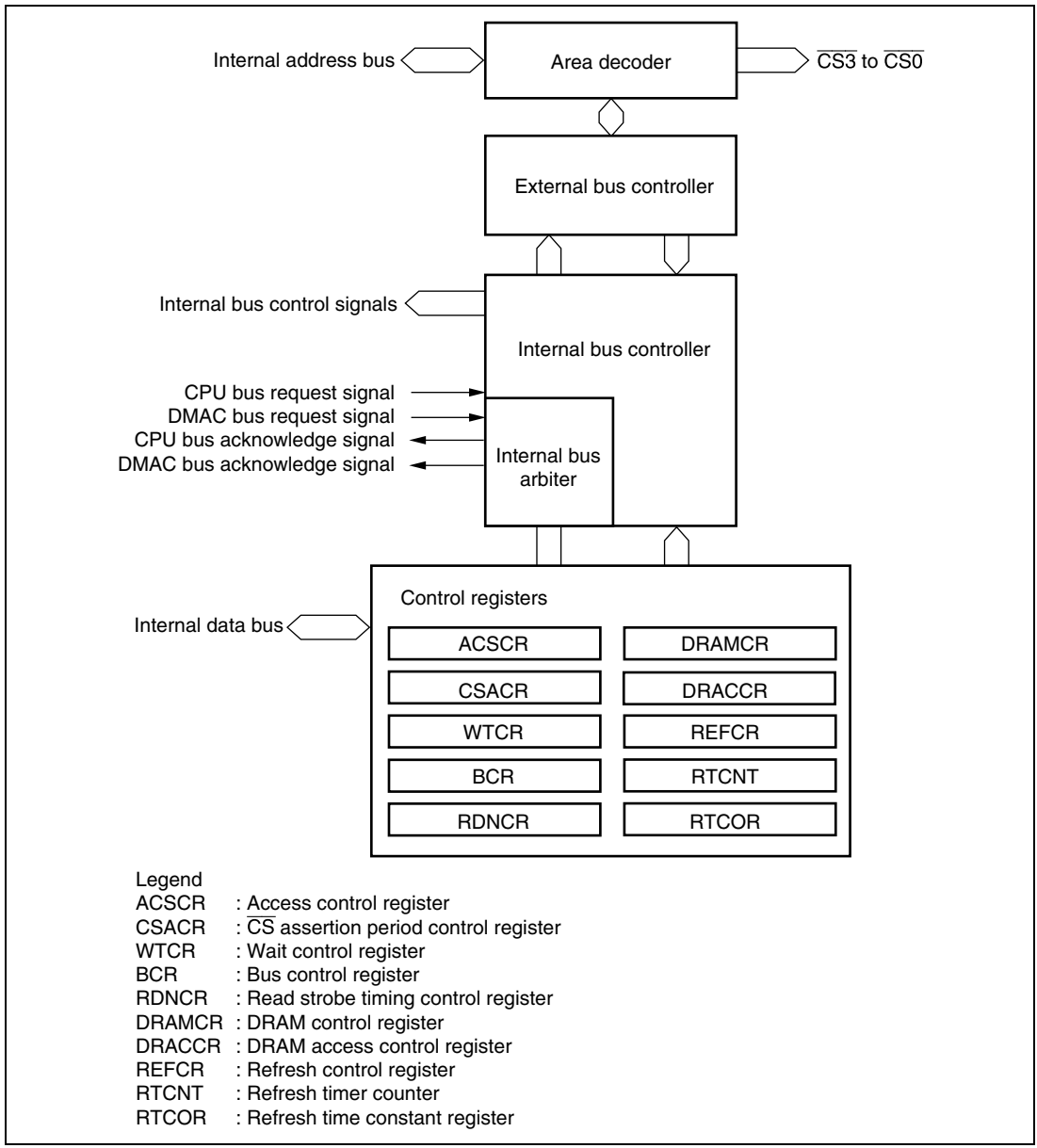


Figure 6.1 Block Diagram of Bus Controller

6.2 Input/Output Pins

Table 6.1 shows the pin configuration of the bus controller.

Table 6.1 Pin Configuration

Name	Symbol	I/O	Function
Address strobe	$\overline{AS}$	Output	Strobe signal indicating that normal space is accessed and address output on address bus is enabled.
Read	$\overline{RD}$	Output	Strobe signal indicating that normal space is being read.
High write	$\overline{HWR}$	Output	Strobe signal indicating that normal space is written to, and upper half (D15 to D8) of data bus is enabled.
Low write	$\overline{LWR}$	Output	Strobe signal indicating that normal space is written to, and lower half (D7 to D0) of data bus is enabled.
Chip select 0	$\overline{CS0}$	Output	Strobe signal indicating that area 0 is selected.
Chip select 1	$\overline{CS1}$	Output	Strobe signal indicating that area 1 is selected
Chip select 2/ row address strobe	$\overline{CS2}$/ $\overline{RAS}$	Output	Strobe signal indicating that area 2 is selected/DRAM row address strobe signal
Chip select 3	$\overline{CS3}$	Output	Strobe signal indicating that area 3 is selected.
Upper column address strobe	$\overline{UCAS}$	Output	16-bit DRAM space upper column address strobe signal or 8-bit DRAM space column address strobe signal
Lower column address strobe	$\overline{LCAS}$	Output	16-bit DRAM space lower column address strobe signal
Data transfer acknowledge 3 (DMAC)	$\overline{DACK3}$	Output	Data transfer acknowledge signal for single address transfer by DMAC channel 3.
Data transfer acknowledge 2 (DMAC)	$\overline{DACK2}$	Output	Data transfer acknowledge signal for single address transfer by DMAC channel 2.
Data transfer acknowledge 1 (DMAC)	$\overline{DACK1}$	Output	Data transfer acknowledge signal for single address transfer by DMAC channel 1.
Data transfer acknowledge 0 (DMAC)	$\overline{DACK0}$	Output	Data transfer acknowledge signal for single address transfer by DMAC channel 0.

6.3 Register Descriptions

The bus controller has the following registers.

- Access control register (ACSCR)
- $\overline{CS}$ assertion period control register (CSACR)
- Wait control register (WTCR)
- Bus control register (BCR)
- Read strobe timing control register (RDNCR)
- DRAM control register (DRAMCR)
- DRAM access control register (DRACCR)
- Refresh control register (REFCR)
- Refresh timer counter (RTCNT)
- Refresh time constant register (RTCOR)

6.3.1 Access Control Register (ACSCR)

ACSCR designates each area in the external address space as either 8-bit access space or 16-bit access space. ACSCR designates each area in the external address space as either 2-state access space or 3-state access space.

ACSCR is initialized to H′FF at a reset or in hardware standby mode but not initialized in software standby mode.

Bit	Bit Name	Initial Value	R/W	Description
7	ABW3	1	R/W	Area 3 to 0 Bus Width Control
6	ABW2	1	R/W	These bits select whether the corresponding area is to be designated as 8-bit access space or 16-bit access space.
5	ABW1	1	R/W	
4	ABW0	1	R/W	0: Area n is designated as 16-bit access space
				1: Area n is designated as 8-bit access space
3	AST3	1	R/W	Area 3 to 0 Access State Control
2	AST2	1	R/W	These bits select whether the corresponding area is to be designated as 2-state access space or 3-state access space. Wait state insertion is enabled or disabled at the same time.
1	AST1	1	R/W	
0	AST0	1	R/W	0: Area n is designated as 2-state access space Wait state insertion in area n access is disabled
				1: Area n is designated as 3-state access space Wait state insertion in area n access is enabled
				(n = 3 to 0)

6.3.2 $\overline{CS}$ Assertion Period Control Register (CSACR)

CSACR selects whether or not the assertion period of the basic bus interface chip select signals ($\overline{CSn}$) and address signals is to be extended. Extending the assertion period of the $\overline{CSn}$ and address signals allows flexible interfacing to external I/O devices.

Bit	Bit Name	Initial Value	R/W	Description
7	CSXH3	0	R/W	$\overline{CS}$ and Address Signal Assertion Period Control 1
6	CSXH2	0	R/W	These bits specify whether or not the T_h cycle is to be inserted (see figure 6.2). When an area for which the CSXHn bit is set to 1 is accessed, a one-state T_h cycle, in which only the $\overline{CSn}$ and address signals are asserted, is inserted before the normal access cycle. 0: In area n basic bus interface access, the $\overline{CSn}$ and address assertion period (T_h) is not extended 1: In area n basic bus interface access, the $\overline{CSn}$ and address assertion period (T_h) is extended
5	CSXH1	0	R/W	
4	CSXH0	0	R/W	
3	CSXT3	0	R/W	$\overline{CS}$ and Address Signal Assertion Period Control 2
2	CSXT2	0	R/W	These bits specify whether or not the T_t cycle is to be inserted (see figure 6.2). When an area for which the CSXTn bit is set to 1 is accessed, a one-state T_t cycle, in which only the $\overline{CSn}$ and address signals are asserted, is inserted before the normal access cycle. 0: In area n basic bus interface access, the $\overline{CSn}$ and address assertion period (T_t) is not extended 1: In area n basic bus interface access, the $\overline{CSn}$ and address assertion period (T_t) is extended (n = 3 to 0)
1	CSXT1	0	R/W	
0	CSXT0	0	R/W	

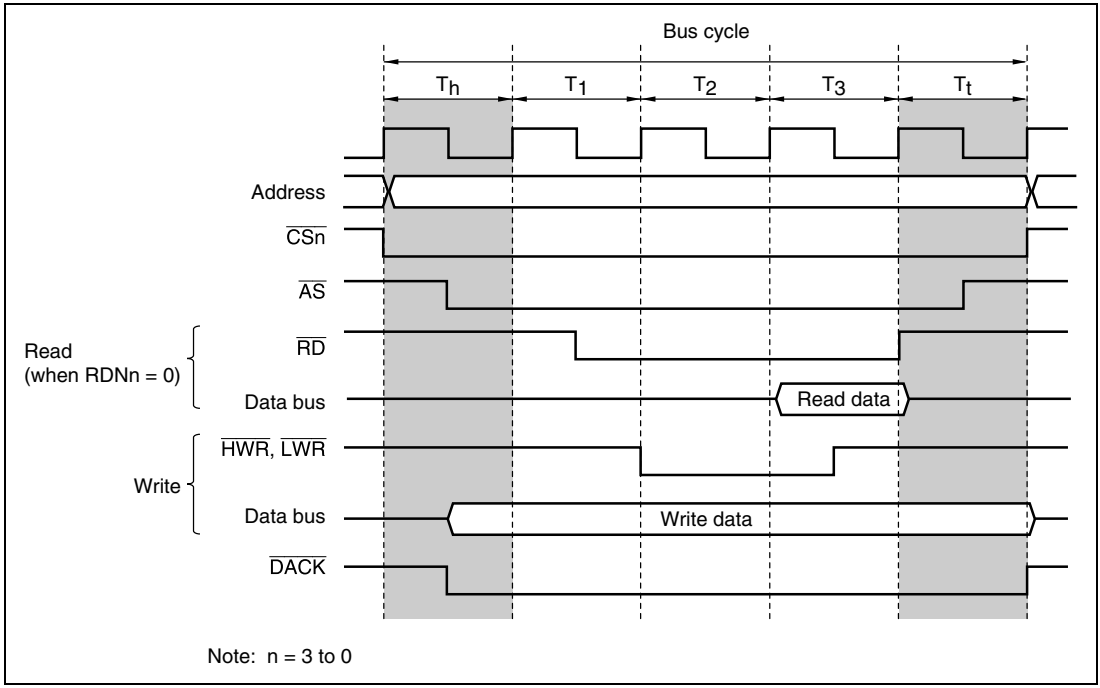


Figure 6.2 $\overline{\text{CS}}$ and Address Assertion Period Extension (Example of 3-State Access Space and RDNn = 0)

6.3.3 Wait Control Register (WTCR)

WTCR selects the number of program wait states for each area in the external address space.

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
14	W32	1	R/W	Area 3 Wait Control 2 to 0
13	W31	1	R/W	These bits select the number of program wait states when accessing area 3 while AST3 bit in ACSCR = 1.
12	W30	1	R/W	000: Program wait not inserted 001: 1 program wait state inserted 010: 2 program wait states inserted 011: 3 program wait states inserted 100: 4 program wait states inserted 101: 5 program wait states inserted 110: 6 program wait states inserted 111: 7 program wait states inserted
11	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
10	W22	1	R/W	Area 2 Wait Control 2 to 0
9	W21	1	R/W	These bits select the number of program wait states when accessing area 2 while AST2 bit in ACSCR = 1.
8	W20	1	R/W	000: Program wait not inserted 001: 1 program wait state inserted 010: 2 program wait states inserted 011: 3 program wait states inserted 100: 4 program wait states inserted 101: 5 program wait states inserted 110: 6 program wait states inserted 111: 7 program wait states inserted

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
6	W12	1	R/W	Area 1 Wait Control 2 to 0 These bits select the number of program wait states when accessing area 1 while AST1 bit in ACSCR = 1. 000: Program wait not inserted 001: 1 program wait state inserted 010: 2 program wait states inserted 011: 3 program wait states inserted 100: 4 program wait states inserted 101: 5 program wait states inserted 110: 6 program wait states inserted 111: 7 program wait states inserted
5	W11	1	R/W	
4	W10	1	R/W	
3	—	0	R	Reserved This bit is always read as 0 and cannot be modified.
2	W02	1	R/W	Area 0 Wait Control 2 to 0 These bits select the number of program wait states when accessing area 0 while AST0 bit in ACSCR = 1. 000: Program wait not inserted 001: 1 program wait state inserted 010: 2 program wait states inserted 011: 3 program wait states inserted 100: 4 program wait states inserted 101: 5 program wait states inserted 110: 6 program wait states inserted 111: 7 program wait states inserted
1	W01	1	R/W	
0	W00	1	R/W	

6.3.4 Bus Control Register (BCR)

BCR is used for idle cycle settings and enabling or disabling of the write data buffer function.

Bit	Bit Name	Initial Value	R/W	Description
15 to 9	—	All 0	R/W	Reserved These bits can be read from or written to. However, the write value should always be 0.
8	WDBE	0	R/W	Write Data Buffer Enable The write data buffer function can be used for an external write cycle or DMAC single address transfer cycle. 0: Write data buffer function not used 1: Write data buffer function used
7 to 4	—	All 0	R/W	Reserved These bits can be read from or written to. However, the write value should always be 0.
3	IDLE1	0	R/W	Idle Cycle Enable These bits enable the idle cycle insertion. 00: Idle cycle insertion is disabled. 01: When read accesses to different areas are continued or external accesses are continued after a single address transfer, idle cycle insertion is enabled. 10: When read accesses to different areas are continued, external accesses are continued after a single address transfer, or write accesses are continued after a read, idle cycle insertion is enabled. 11: When read accesses to different areas are continued, external accesses are continued after a single address transfer, write accesses are continued after a read, or read accesses are continued after a write, idle cycle insertion is enabled.
2	IDLE0	0	R/W	
1	IDLC1	0	R/W	Idle Cycle State Number Select These bits specify the number of idle cycle states to be inserted. 00: 1 state 01: 2 states 10: 3 states 11: 4 states
0	IDLC0	0	R/W	

6.3.5 Read Strobe Timing Control Register (RDNCR)

RDNCR selects the read strobe signal ($\overline{RD}$) negation timing in a read access to normal space.

Bit	Bit Name	Initial Value	R/W	Description
7	RDN3	0	R/W	Read Strobe Timing Control 3 to 0
6	RDN2	0	R/W	These bits set the negation timing of the read
5	RDN1	0	R/W	strobe in a corresponding area read access.
4	RDN0	0	R/W	As shown in figure 6.3, the read strobe for an area for which the RDNn bit is set to 1 is negated one half-state earlier than that for an area for which the RDNn bit is cleared to 0. The read data setup and hold time specifications are also one half-state earlier. 0: In an area n read access, the $\overline{RD}$ is negated at the end of the read cycle 1: In an area n read access, the $\overline{RD}$ is negated one half-state before the end of the read cycle (n = 3 to 0)
3 to 0	—	All 0	R/W	Reserved These bits can be read from or written to. However, the write value should always be 0.

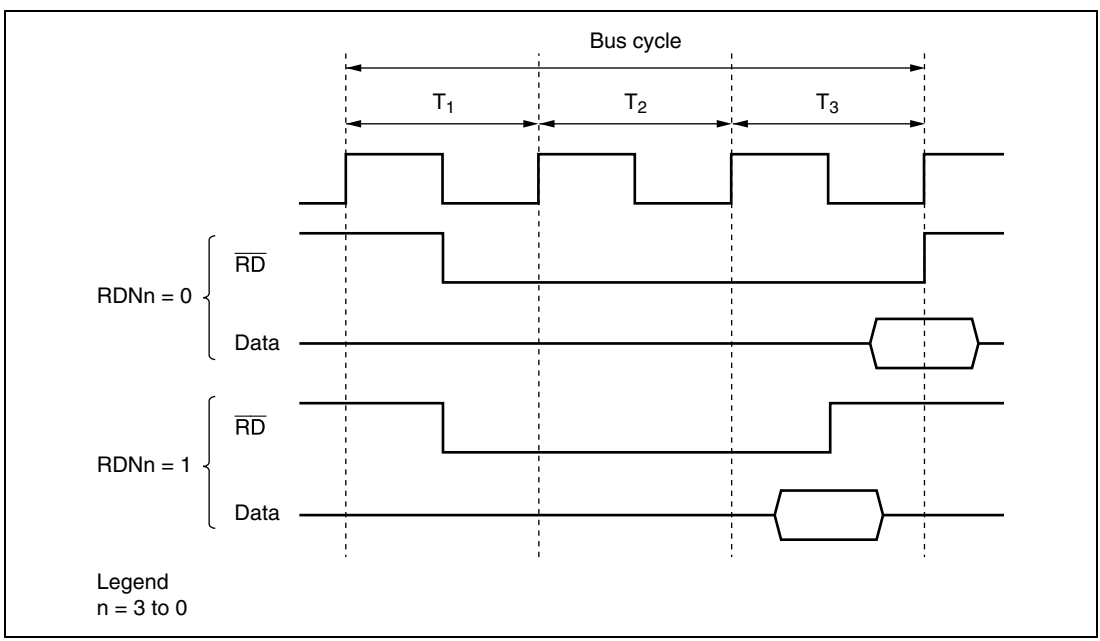


Figure 6.3 Read Strobe Negation Timing (Example of 3-State Access Space)

6.3.6 DRAM Control Register (DRAMCR)

DRAMCR is used to make DRAM interface settings.

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R/W	Reserved This bit can be read from or written to. However, the write value should always be 0.
14	RAST	0	R/W	$\overline{RAS}$ Assertion Timing Select Selects whether, in DRAM access, the $\overline{RAS}$ signal is asserted from the start of the T_r cycle (rising edge of ϕ) or from the falling edge of ϕ. Figure 6.4 shows the relationship between the RAST bit setting and the $\overline{RAS}$ assertion timing. 0: $\overline{RAS}$ is asserted from ϕ falling edge in T_r cycle 1: $\overline{RAS}$ is asserted from start of T_r cycle
13	—	0	R/W	Reserved This bit can be read from or written to. However, the write value should always be 0.
12	CAST	0	R/W	Column Address Output Cycle Number Select Selects whether the column address output cycle in DRAM access comprises 3 states or 2 states. 0: Column address output cycle comprises 2 states 1: Column address output cycle comprises 3 states
11 to 9	—	All 0	R/W	Reserved These bits can be read from or written to. However, the write value should always be 0.
8	DSET	0	R/W	DRAM Space Setting Specifies area 2 as DRAM space. 0: Area 2 is specified as normal space 1: Area 2 is specifies as DRAM space
7	BE	0	R/W	Burst Access Enable Selects enabling or disabling of burst access to areas designated as DRAM space. DRAM space burst access is performed in fast page mode. When using EDO page mode DRAM, the $\overline{RD}$ signal must be connected as the $\overline{OE}$ signal. 0: Full access 1: Access in fast page mode

Bit	Bit Name	Initial Value	R/W	Description
6	RCDM	0	R/W	$\overline{\text{RAS}}$ Down Mode When access to DRAM space is interrupted by an access to normal space, an access to an internal I/O register, etc., this bit selects whether the $\overline{\text{RAS}}$ signal is held low while waiting for the next DRAM access ($\overline{\text{RAS}}$ down mode), or is driven high again ($\overline{\text{RAS}}$ up mode). The setting of this bit is valid only when the BE bit is set to 1. If this bit is cleared to 0 when set to 1 in the $\overline{\text{RAS}}$ down state, the $\overline{\text{RAS}}$ down state is cleared at that point, and $\overline{\text{RAS}}$ goes high. When using DRAM interface in $\overline{\text{RAS}}$ down mode and $\overline{\text{RAS}}$ down state is not continued, a 1-state idle cycle is inserted to drive $\overline{\text{RAS}}$ signal high. 0: $\overline{\text{RAS}}$ up mode selected for DRAM space access 1: $\overline{\text{RAS}}$ down mode selected for DRAM space access
5	DDS	0	R/W	DMAC Single Address Transfer Option Selects whether full access is always performed or burst access is enabled when DMAC single address transfer is performed on the DRAM interface. When the BE bit is cleared to 0 in DRAMCR, disabling DRAM burst access, DMAC single address transfer is performed in full access mode regardless of the setting of this bit. This bit has no effect on other bus master external accesses or DMAC dual address transfers. If this bit is set to 1, the DACK output timing is changed. 0: Full access is always executed 1: Burst access is enabled
4, 3	—	All 0	R/W	Reserved These bits can be read from or written to. However, the write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
2	MXC2	0	R/W	Address Multiplex Select
1	MXC1	0	R/W	These bits select the size of the shift toward the lower half of the row address in row address/column address multiplexing. In burst operation on the DRAM interface, these bits also select the row address bits to be used for comparison.
0	MXC0	0	R/W	000: 8-bit shift • When 8-bit access space is designated: Row address bits A23 to A8 used for comparison • When 16-bit access space is designated: Row address bits A23 to A9 used for comparison 001: 9-bit shift • When 8-bit access space is designated: Row address bits A23 to A9 used for comparison • When 16-bit access space is designated: Row address bits A23 to A10 used for comparison 010: 10-bit shift • When 8-bit access space is designated: Row address bits A23 to A10 used for comparison • When 16-bit access space is designated: Row address bits A23 to A11 used for comparison 011: 11-bit shift • When 8-bit access space is designated: Row address bits A23 to A11 used for comparison When 16-bit access space is designated: Row address bits A23 to A12 used for comparison

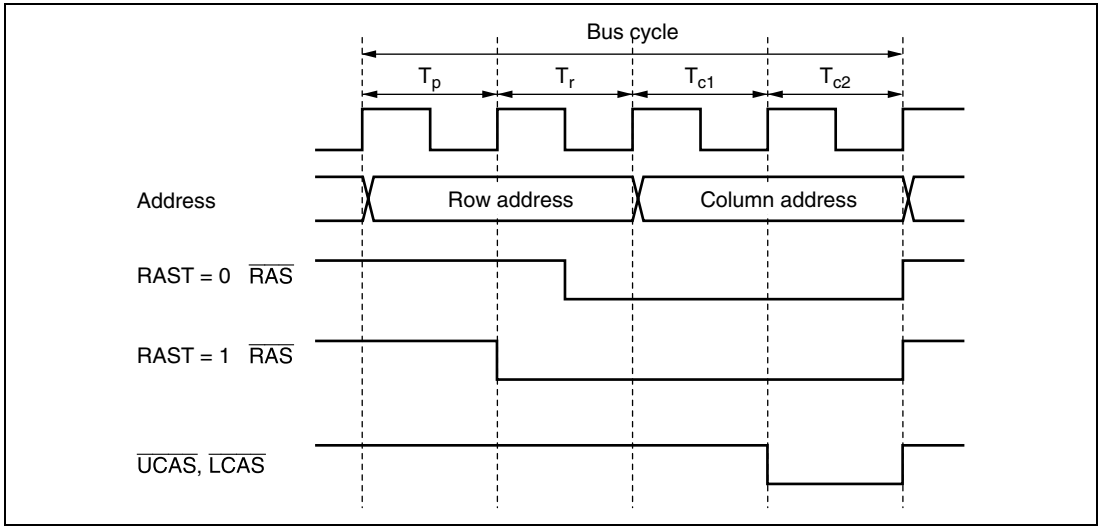


Figure 6.4 $\overline{RAS}$ Signal Assertion Timing (2-State Column Address Output Cycle, Full Access)

6.3.7 DRAM Access Control Register (DRACCR)

DRACCR is used to set the DRAM interface bus specifications.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R/W	Reserved These bits can be read from or written to. However, the write value should always be 0.
5	TPC1	0	R/W	Precharge State Control
4	TPC0	0	R/W	These bits select the number of states in the RAS precharge cycle in normal access and refreshing. 00: 1 state 01: 2 states 10: 3 states 11: 4 states
3, 2	—	All 0	R/W	Reserved These bits can be read from or written to. However, the write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
1	RCD1	0	R/W	RAS-CAS Wait Control
0	RCD0	0	R/W	These bits select a wait cycle to be inserted between the $\overline{RAS}$ assert cycle and $\overline{CAS}$ assert cycle. 00: Wait cycle not inserted 01: 1-state wait cycle inserted 10: 2-state wait cycle inserted 11: 3-state wait cycle inserted

6.3.8 Refresh Control Register (REFCR)

REFCR specifies DRAM interface refresh control.

Bit	Bit Name	Initial Value	R/W	Description
15	CMF	0	R/(W)*	Compare Match Flag Status flag that indicates a match between the values of RTCNT and RTCOR. [Clearing conditions] • When 0 is written to CMF after reading CMF = 1 while the RFSHE bit is cleared to 0 • When CBR refreshing is executed while the RFSHE bit is set to 1 [Setting condition] When RTCOR = RTCNT
14	CMIE	0	R/W	Compare Match Interrupt Enable Enables or disables interrupt requests (CMI) by the CMF flag when the CMF flag is set to 1. This bit is valid when refresh control is not performed (RFSHE = 0). When the refresh control is performed (RFSHE = 1), this bit is always cleared to 0 and cannot be modified. 0: Interrupt request by CMF flag disabled 1: Interrupt request by CMF flag enabled

Bit	Bit Name	Initial Value	R/W	Description
13	RCW1	0	R/W	$\overline{CAS}$-$\overline{RAS}$ Wait Control
12	RCW0	0	R/W	These bits select the number of wait cycles to be inserted between the $\overline{CAS}$ assert cycle and $\overline{RAS}$ assert cycle in a DRAM refresh cycle. 00: Wait state not inserted 01: 1 wait state inserted 10: 2 wait states inserted 11: 3 wait states inserted
11	—	0	R/W	Reserved This bit can be read from or written to. However, the write value should always be 0.
10	RTCK2	0	R/W	Refresh Counter Clock Select
9	RTCK1	0	R/W	These bits select the clock to be used to increment the refresh counter. When the input clock is selected with bits RTCK2 to RTCK0, the refresh counter begins counting up. 000: Count operation halted 001: Count on ϕ/2 010: Count on ϕ/8 011: Count on ϕ/32 100: Count on ϕ/128 101: Count on ϕ/512 110: Count on ϕ/2048 111: Count on ϕ/4096
8	RTCK0	0	R/W	
7	RFSHE	0	R/W	Refresh Control Refresh control can be performed. When refresh control is not performed, the refresh timer can be used as an interval timer. 0: Refresh control is not performed 1: Refresh control is performed
6	—	0	R/W	Reserved This bit can be read from or written to. However, the write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
5	RLW1	0	R/W	Refresh Cycle Wait Control
4	RLW0	0	R/W	These bits select the number of wait states to be inserted in a DRAM interface CAS-before-RAS refresh cycle. 00: No wait state inserted 01: 1 wait state inserted 10: 2 wait states inserted 11: 3 wait states inserted
3	SLFRF	0	R/W	Self-Refresh Enable If this bit is set to 1, DRAM self-refresh mode is selected when a transition is made to the software standby state. This bit is valid when the RFSHE bit is set to 1, enabling refresh operations. 0: Self-refreshing is disabled 1: Self-refreshing is enabled
2	TPCS2	0	R/W	Self-Refresh Precharge Cycle Control
1	TPCS1	0	R/W	These bits select the number of states in the precharge cycle immediately after self-refreshing. The number of states in the precharge cycle immediately after self-refreshing are added to the number of states set by bits TPC1 and TPC0 in DRACCR. 000: [TPC set value] states 001: [TPC set value + 1] states 010: [TPC set value + 2] states 011: [TPC set value + 3] states 100: [TPC set value + 4] states 101: [TPC set value + 5] states 110: [TPC set value + 6] states 111: [TPC set value + 7] states
0	TPCS0	0	R/W	

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

6.3.9 Refresh Timer Counter (RTCNT)

RTCNT is an 8-bit readable/writable up-counter. RTCNT counts up using the internal clock selected by bits RTCK2 to RTCK0 in REFCR.

When RTCNT matches RTCOR (compare match), the CMF flag in REFCR is set to 1 and RTCNT is cleared to H'00. If the RFSHE bit in REFCR is set to 1 at this time, a refresh cycle is started. If the RFSHE bit is cleared to 0 and the CMIE bit in REFCR is set to 1, a compare match interrupt (CMI) is generated.

RTCNT is initialized to H'00 by a reset and in hardware standby mode. It is not initialized in software standby mode.

6.3.10 Refresh Time Constant Register (RTCOR)

RTCOR is an 8-bit readable/writable register that sets the period for compare match operations with RTCNT.

The values of RTCOR and RTCNT are constantly compared, and if they match, the CMF flag in REFCR is set to 1 and RTCNT is cleared to H'00.

RTCOR is initialized to H'FF by a reset and in hardware standby mode. It is not initialized in software standby mode.

6.4 Bus Control

6.4.1 Area Division

The bus controller divides the 16-Mbyte address space into areas shown in figure 6.5, and performs bus control for external address space in area units. Chip select signals ($\overline{CS0}$ to $\overline{CS3}$) can be output for each area.

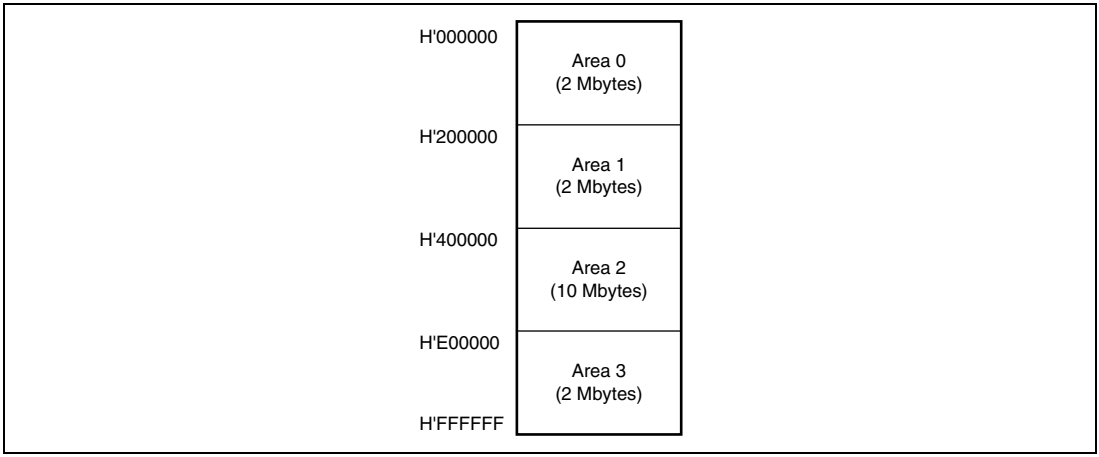
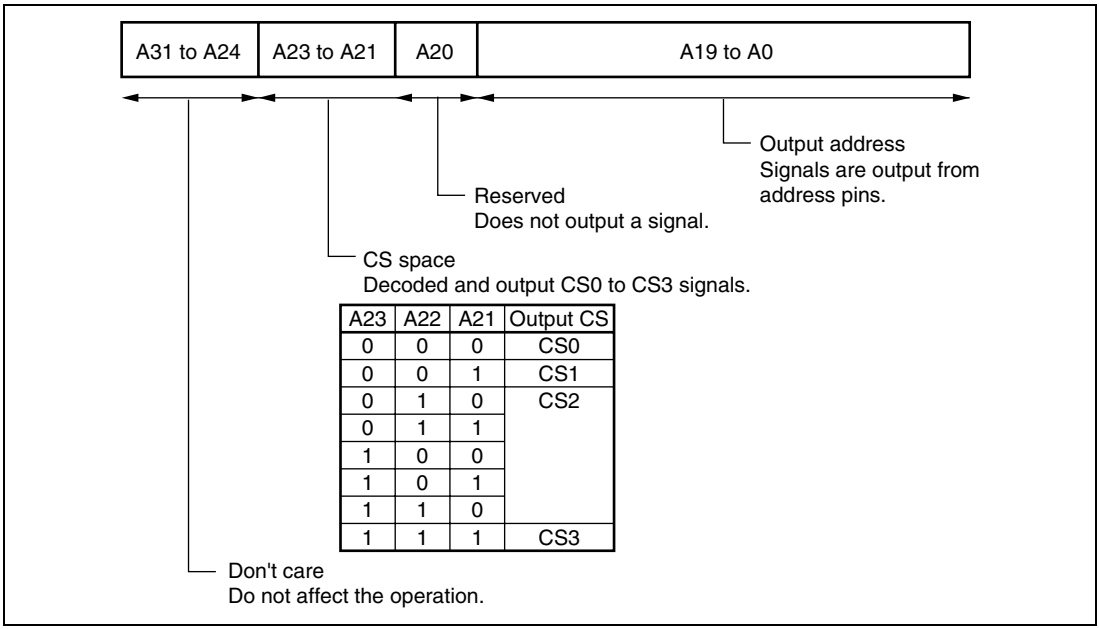


Figure 6.5 Area Divisions

6.4.2 Address Map

Figure 6.6 shows the address format.



A23	A22	A21	Output CS
0	0	0	CS0
0	0	1	CS1
0	1	0	CS2
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	CS3

Figure 6.6 Address Format

Bits A31 to A24 do not affect the operation.

Bits A23 to A21 are decoded by the chip select signals (CS3 to CS0) for each area and output.

Bit A20 is not output externally.

Bits A19 to A0 are output externally.

Enabling or disabling external output of bits A19 to A0 can be selected by the setting of PFCR1. For details, refer to section 8.11.1, Port Function Control Register 1 (PFCR1).

Table 6.2 and figure 6.7 show the address map.

Table 6.2 Address Map

Address	Space Type	Memory Type	Size	Bus Width
H'000000 to H'1FFFFF	CS0 space/ on-chip ROM space	External space/ on-chip ROM	2 Mbytes	8/16
H'200000 to H'3FFFFF	CS1 space	External space	2 Mbytes	8/16
H'400000 to H'DFFFFF	CS2 space/ DRAM space	External space/ DRAM	10 Mbytes	8/16
H'E00000 to H'FFFFFF	CS3 space/ on-chip RAM space*/ I/O space	External space/ on-chip RAM*/I/O space	2 Mbytes	8/16

Note: * On-chip RAM space when the RAME bit in SYSCR is 1.

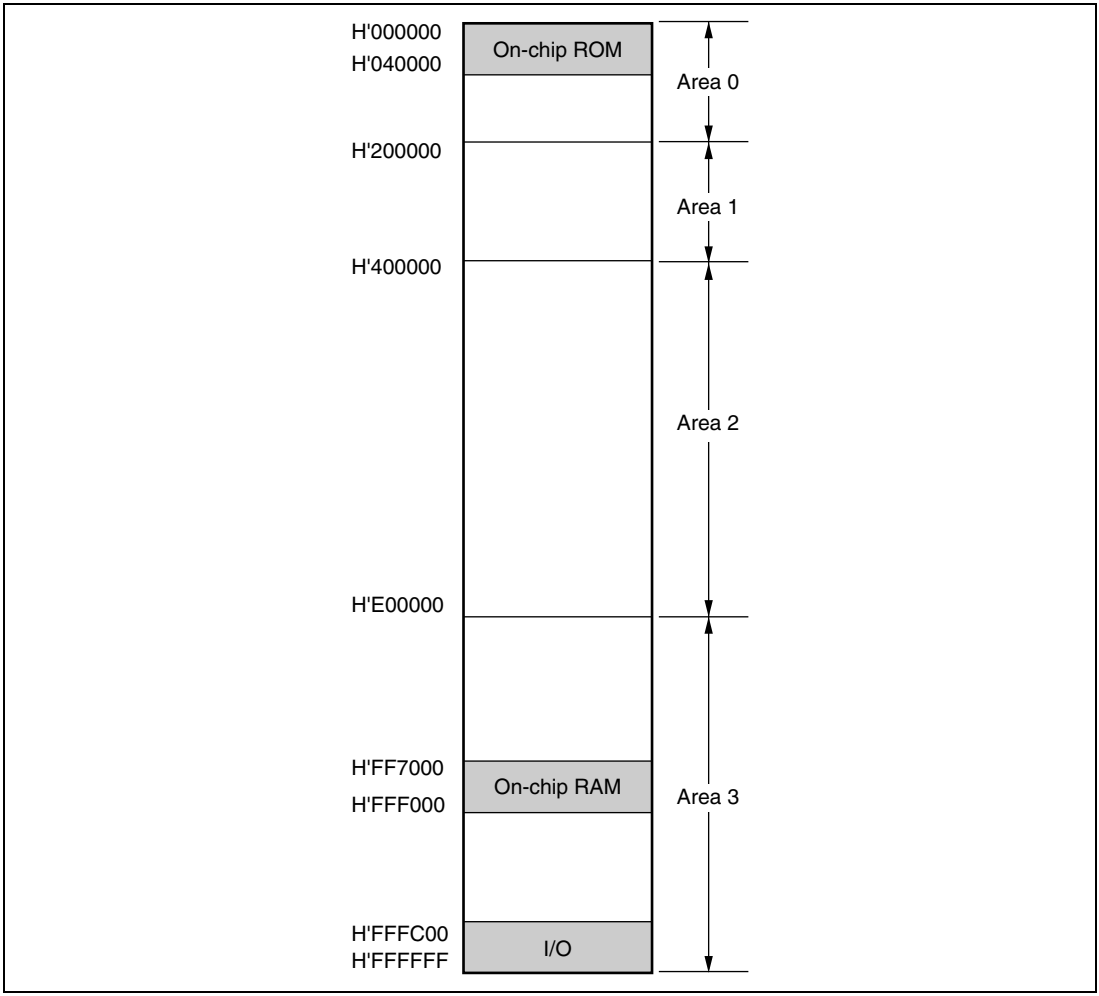


Figure 6.7 Address Map

6.4.3 Bus Specifications

The external address space bus specifications consist of five elements: bus width, number of access states, number of program wait states, read strobe timing, and chip select ($\overline{CS}$) assertion period extension states. The bus width and number of access states for on-chip memory and internal I/O registers are fixed, and are not affected by the bus controller.

Bus Width: A bus width of 8 or 16 bits can be selected with ACSCR. An area for which an 8-bit bus is selected functions as an 8-bit access space, and an area for which a 16-bit bus is selected functions as a 16-bit access space.

Number of Access States: Two or three access states can be selected with ACSCR. An area for which 2-state access is selected functions as a 2-state access space, and an area for which 3-state access is selected functions as a 3-state access space. With the DRAM interface, the number of access states may be determined without regard to the setting of ACSCR.

When 2-state access space is designated, wait insertion is disabled. When 3-state access space is designated, it is possible to insert program waits by means of WTCR.

Number of Program Wait States: When 3-state access space is designated by ACSCR, the number of program wait states to be inserted automatically is selected with WTCR. From 0 to 7 program wait states can be selected. Table 6.3 shows the bus specifications (bus width, and number of access states and program wait states) for each basic bus interface area.

Table 6.3 Bus Specifications for Each Area (Basic Bus Interface)

ACSCR		WTCR			Bus Specifications (Basic Bus Interface)		
ABWn	ASTn	Wn2	Wn1	Wn0	Bus Width	Access States	Program Wait States
0	0	—	—	—	16	2	0
	1	0	0	0		3	0
				1			1
			1	0			2
				1			3
		1	0	0			4
				1			5
			1	0			6
				1			7
1	0	—	—	—	8	2	0
	1	0	0	0		3	0
				1			1
			1	0			2
				1			3
		1	0	0			4
				1			5
			1	0			6
				1			7

Legend n = 3 to 0

Read Strobe Timing: RDNCR can be used to select either of two negation timings (at the end of the read cycle or one half-state before the end of the read cycle) for the read strobe ($\overline{RD}$) used in the basic bus interface space.

Chip Select ($\overline{CS}$) Assertion Period Extension States: Some external I/O devices require a setup time and hold time between address and $\overline{CS}$ signals and strobe signals such as $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$. CSACR can be used to insert states in which only the $\overline{CS}$, $\overline{AS}$, and address signals are asserted before and after a basic bus space access cycle.

6.4.4 Memory Interfaces

The memory interfaces in this LSI comprise a basic bus interface that allows direct connection of ROM, SRAM, and so on; and a DRAM interface that allows direct connection of DRAM. The interface can be selected independently for each area.

An area for which the basic bus interface is designated functions as normal space and an area for which the DRAM interface is designated functions as DRAM space

The initial state of each area is basic bus interface, 3-state access space. The initial bus width is 8 bits.

Area 0: Area 0 includes on-chip ROM and the space excluding on-chip ROM is external address space by setting the EXPE bit in MDCR to 1.

When area 0 external space is accessed, the $\overline{CS0}$ signal can be output.

Only basic bus interface can be used for area 0.

Area 1: All of area 1 is external address space by setting the EXPE bit in MDCR to 1.

When area 1 external address space is accessed, the $\overline{CS1}$ signal can be output.

Only basic bus interface can be used for area 1.

Area 2: All of area 2 is external address space by setting the EXPE bit in MDCR to 1.

When area 2 external space is accessed, signal $\overline{CS2}$ can be output.

Basic bus interface or DRAM interface can be selected for area 2. With the DRAM interface, the $\overline{CS2}$ signal is used as the $\overline{RAS}$ signal.

If area 2 is designated as DRAM space, large-capacity (e.g. 64-Mbit) DRAM can be connected. In this case, the $\overline{CS2}$ signal is used as the $\overline{RAS}$ signal for DRAM space.

Area 3: Area 3 includes the on-chip RAM and internal I/O registers. The space excluding the on-chip RAM and internal I/O registers is external address space by setting the EXPE bit in MDCR to 1. The on-chip RAM is enabled when the RAME bit is set to 1 in the system control register (SYSCR); when the RAME bit is cleared to 0, the on-chip RAM is disabled and the corresponding addresses are in external address space.

When area 3 external address space is accessed, the $\overline{CS3}$ signal can be output.

Only the basic bus interface can be used for the area 3 memory interface.

6.4.5 Chip Select Signals

This LSI can output chip select signals ($\overline{CS3}$ to $\overline{CS0}$) for areas 3 to 0. The signal outputs low when the corresponding external space area is accessed. Figure 6.8 shows an example of $\overline{CS3}$ to $\overline{CS0}$ signals output timing.

The $\overline{CS0}$ pin is placed in the output state by setting the EXPE bit in MDCR to 1. Pins $\overline{CS3}$ to $\overline{CS1}$ are placed in the input state after a reset and so the corresponding CS output should be enabled by setting the PFCR1 register when outputting signals $\overline{CS3}$ to $\overline{CS1}$.

For details, refer to section 8.11.1, Port Function Control Register 1 (PFCR1).

When area 2 is designated as DRAM space, output $\overline{CS2}$ is used as the $\overline{RAS}$ signal.

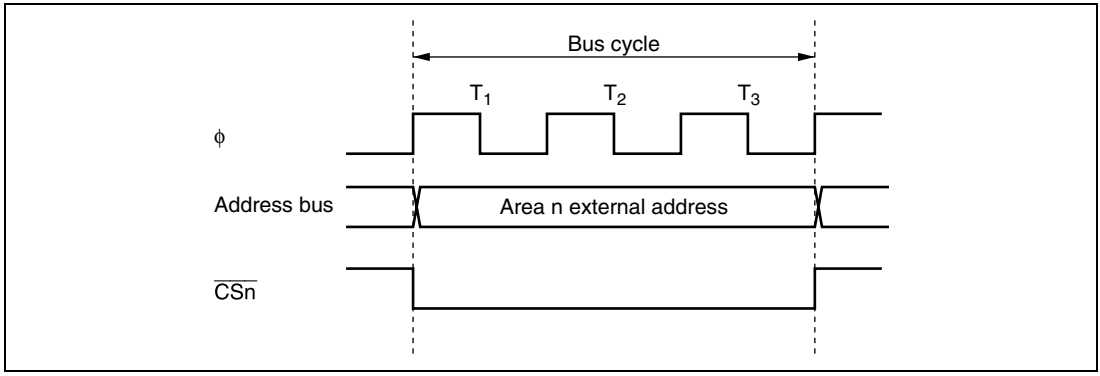


Figure 6.8 $\overline{CSn}$ Signal Output Timing (n = 3 to 0)

6.5 Basic Bus Interface

The basic bus interface enables direct connection of ROM, SRAM, and so on.

6.5.1 Data Size and Data Alignment

Data sizes for the CPU and other internal bus masters are byte, word, and longword. The bus controller has a data alignment function, and when accessing external address space, controls whether the upper data bus (D15 to D8) or lower data bus (D7 to D0) is used according to the bus specifications for the area being accessed (8-bit access space or 16-bit access space) and the data size.

8-Bit Access Space: Figure 6.9 illustrates data alignment control for the 8-bit access space. With the 8-bit access space, the upper data bus (D15 to D8) is always used for accesses. The amount of data that can be accessed at one time is one byte: a word access is performed as two byte accesses, and a longword access, as four byte accesses.

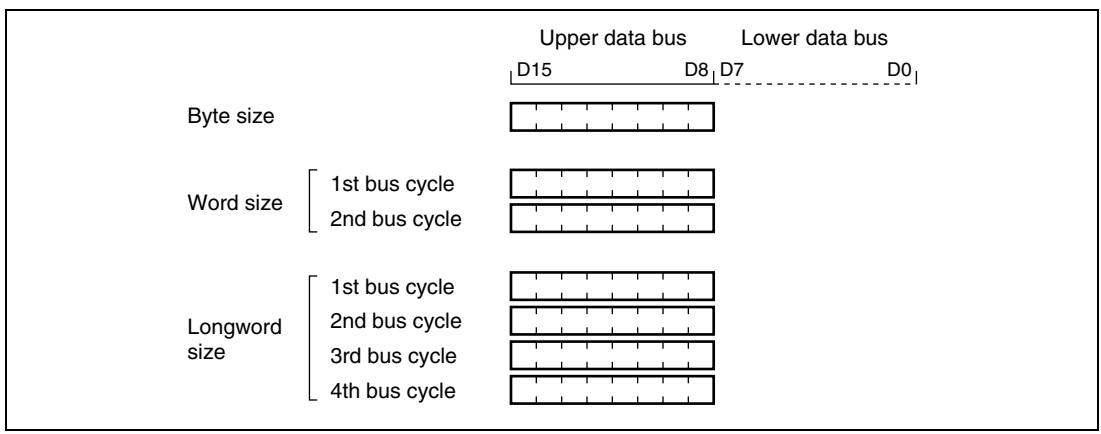


Figure 6.9 Access Sizes and Data Alignment Control (8-Bit Access Space)

16-Bit Access Space: Figure 6.10 illustrates data alignment control for the 16-bit access space. With the 16-bit access space, the upper data bus (D15 to D8) and lower data bus (D7 to D0) are used for accesses. The amount of data that can be accessed at one time is one byte or one word, and a longword access is executed as two word accesses.

In byte access, whether the upper or lower data bus is used is determined by whether the address is even or odd. The upper data bus is used for an even address, and the lower data bus for an odd address.

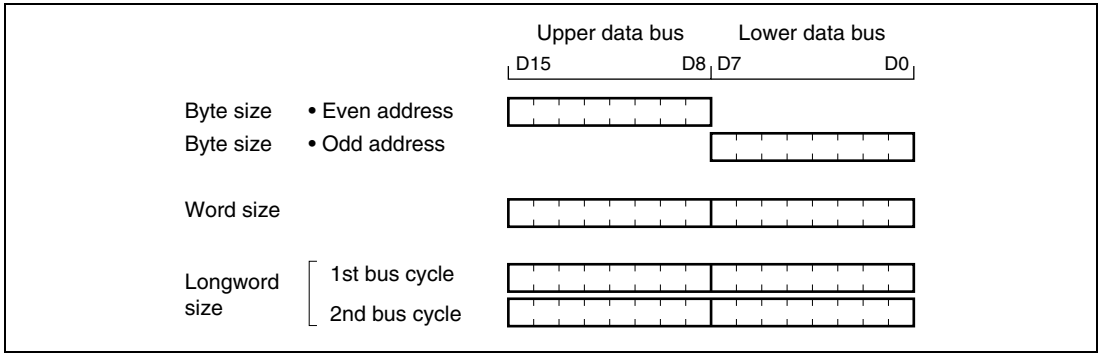


Figure 6.10 Access Sizes and Data Alignment Control (16-bit Access Space)

6.5.2 Valid Strobes

Table 6.4 shows the data buses used and valid strobes for the access spaces.

In a read, the $\overline{RD}$ signal is valid for both the upper and the lower half of the data bus. In a write, the $\overline{HWR}$ signal is valid for the upper half of the data bus, and the $\overline{LWR}$ signal for the lower half.

Table 6.4 Data Buses Used and Valid Strobes

Area	Access Size	Read/ Write	Address	Valid Strobe	Upper Data Bus (D15 to D8)	Lower Data Bus (D7 to D0)
8-bit access space	Byte	Read	—	$\overline{RD}$	Valid	Invalid
		Write	—	$\overline{HWR}$		Hi-Z
16-bit access space	Byte	Read	Even	$\overline{RD}$	Valid	Invalid
			Odd		Invalid	Valid
		Write	Even	$\overline{HWR}$	Valid	Hi-Z
			Odd	$\overline{LWR}$	Hi-Z	Valid
	Word	Read	—	$\overline{RD}$	Valid	Valid
		Write	—	$\overline{HWR}$, $\overline{LWR}$	Valid	Valid

Note: Hi-Z: High-impedance state
Invalid: Input state; input value is ignored.

6.5.3 Basic Timing

8-Bit, 2-State Access Space: Figure 6.11 shows the bus timing for an 8-bit, 2-state access space. When an 8-bit access space is accessed, the upper half (D15 to D8) of the data bus is used.

When all areas are designated as 8-bit space, the $\overline{\text{LWR}}$ pin can be used as the I/O port. However, when all areas are designated as 16-bit space, the $\overline{\text{LWR}}$ pin is always fixed high. Wait states cannot be inserted.

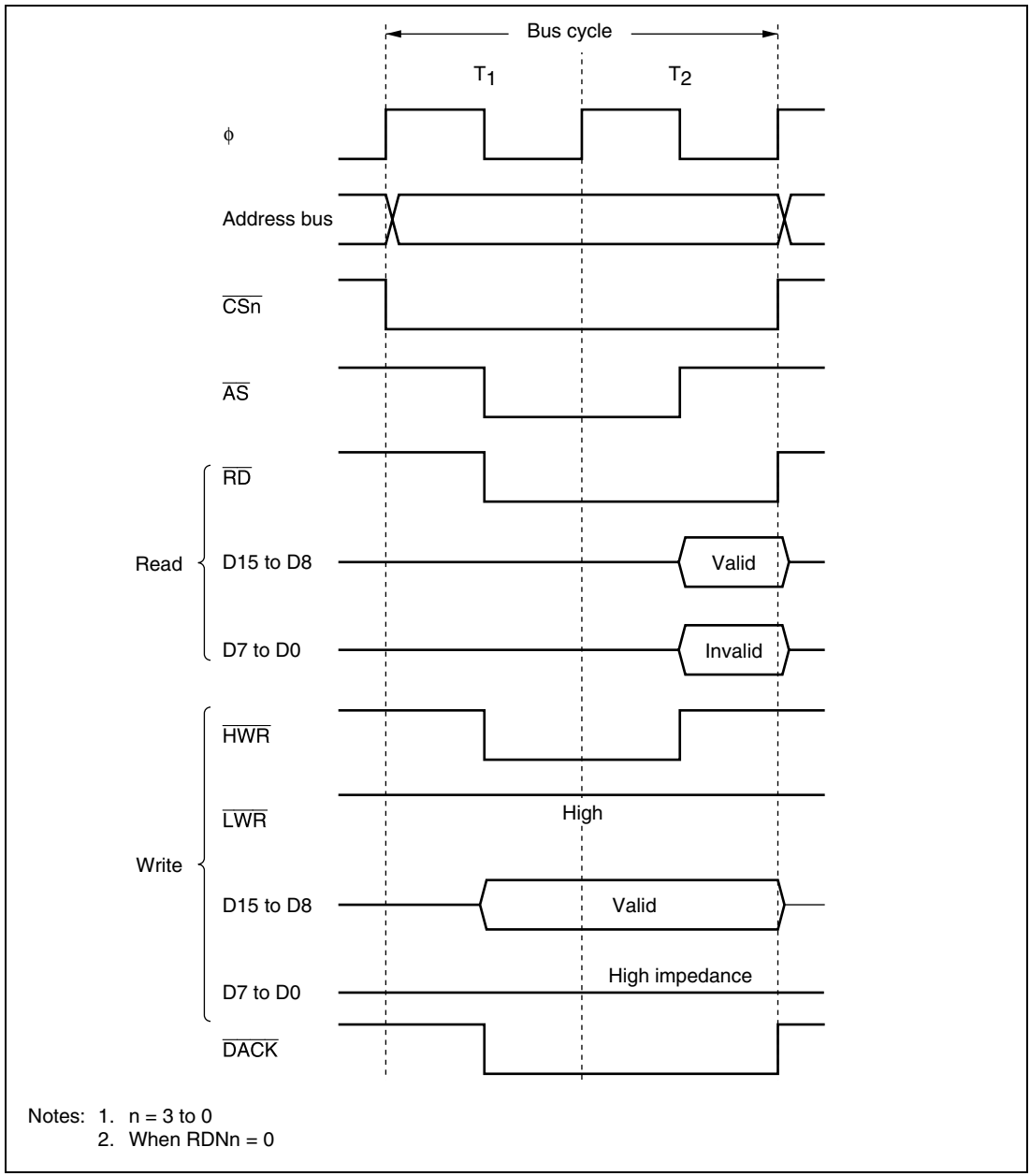


Figure 6.11 Bus Timing for 8-Bit, 2-State Access Space

8-Bit, 3-State Access Space: Figure 6.12 shows the bus timing for an 8-bit, 3-state access space. When an 8-bit access space is accessed, the upper half (D15 to D8) of the data bus is used.

When all areas are designated as 8-bit space, the $\overline{\text{LWR}}$ pin can be used as the I/O port. However, when all areas are designated as 16-bit space, the $\overline{\text{LWR}}$ pin is always fixed high. Wait states can be inserted.

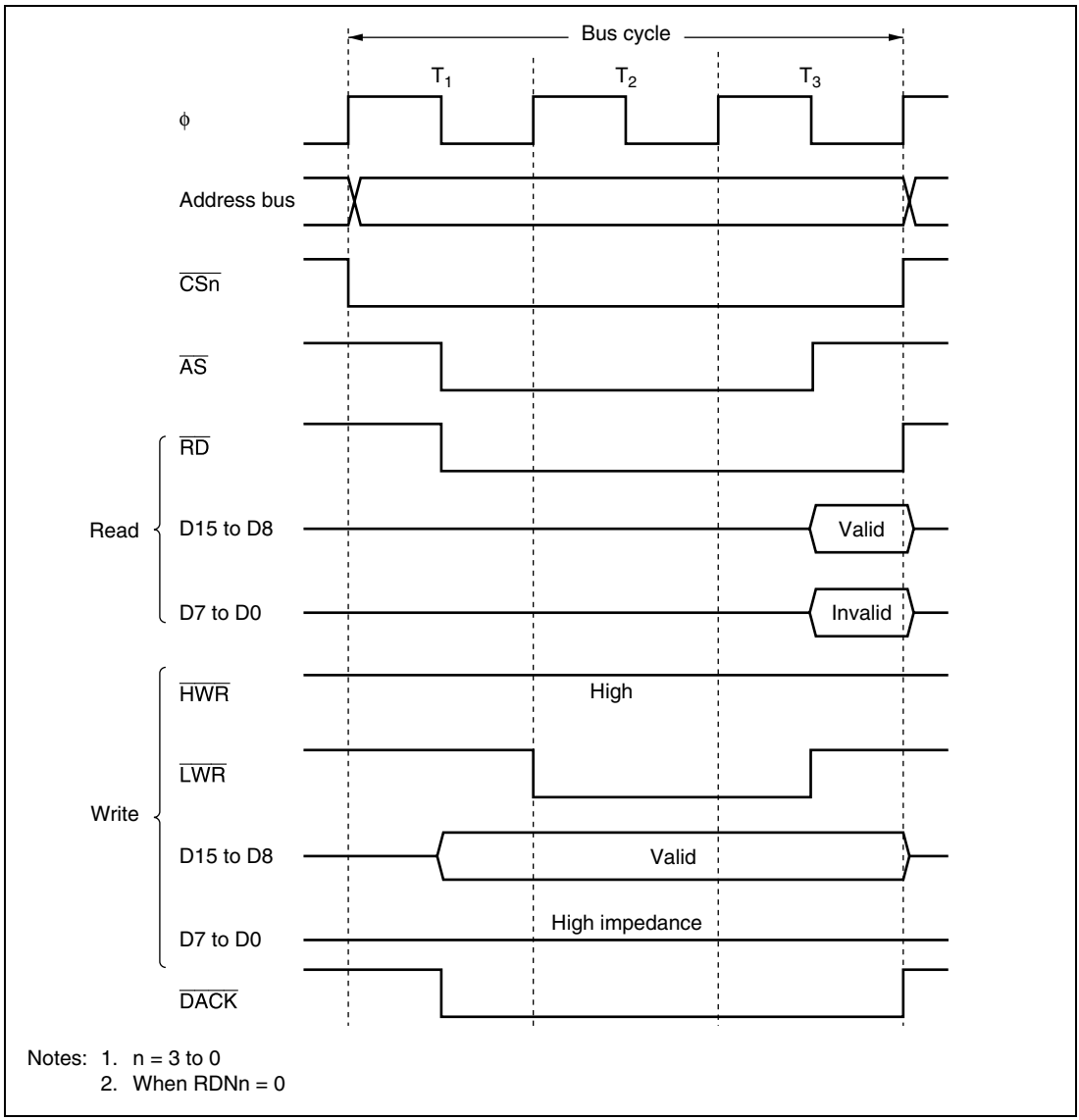


Figure 6.12 Bus Timing for 8-Bit, 3-State Access Space

16-Bit, 2-State Access Space: Figures 6.13 to 6.15 show bus timings for a 16-bit, 2-state access space. When a 16-bit access space is accessed, the upper half (D15 to D8) of the data bus is used for even addresses, and the lower half (D7 to D0) for odd addresses. Wait states cannot be inserted.

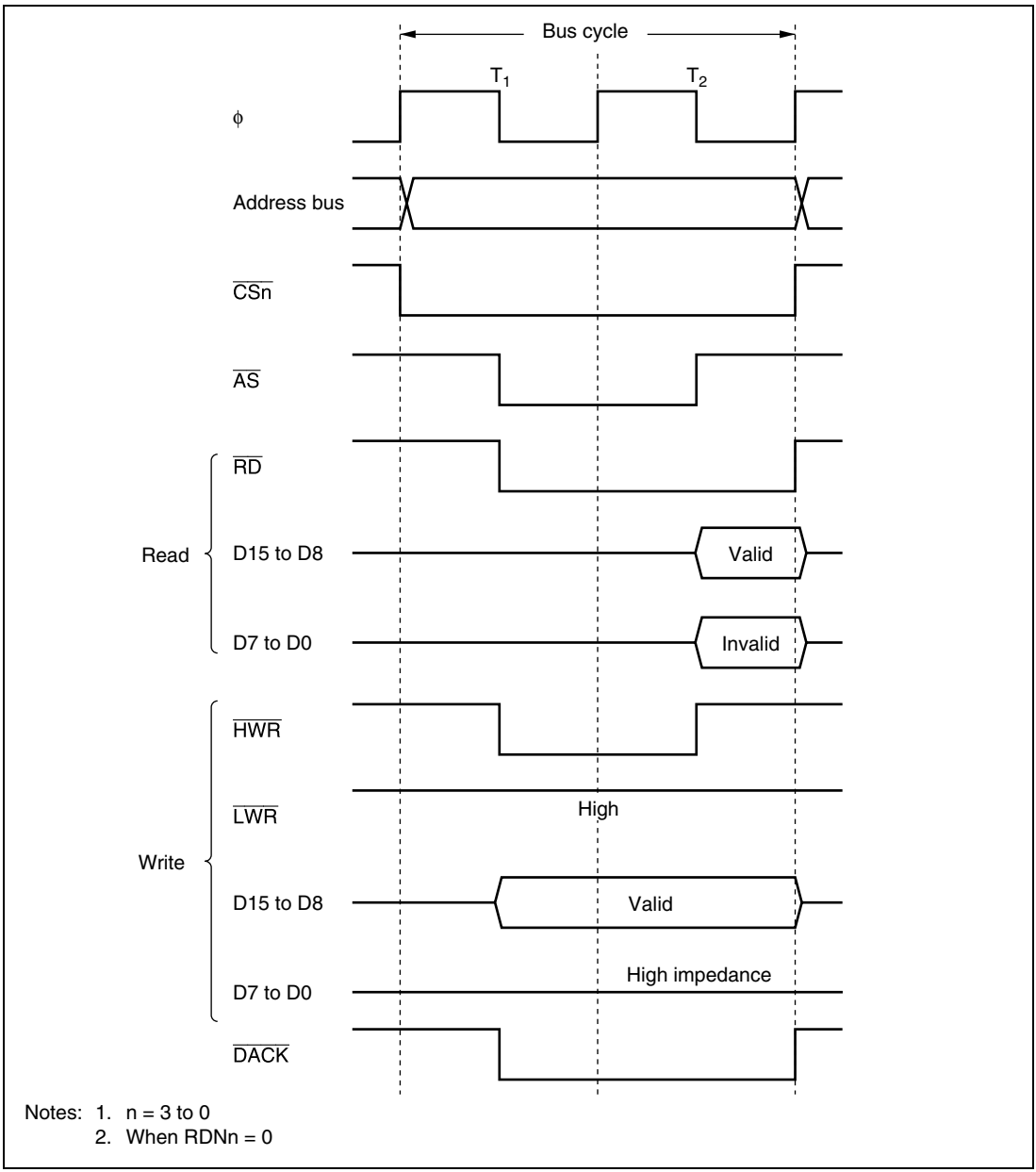
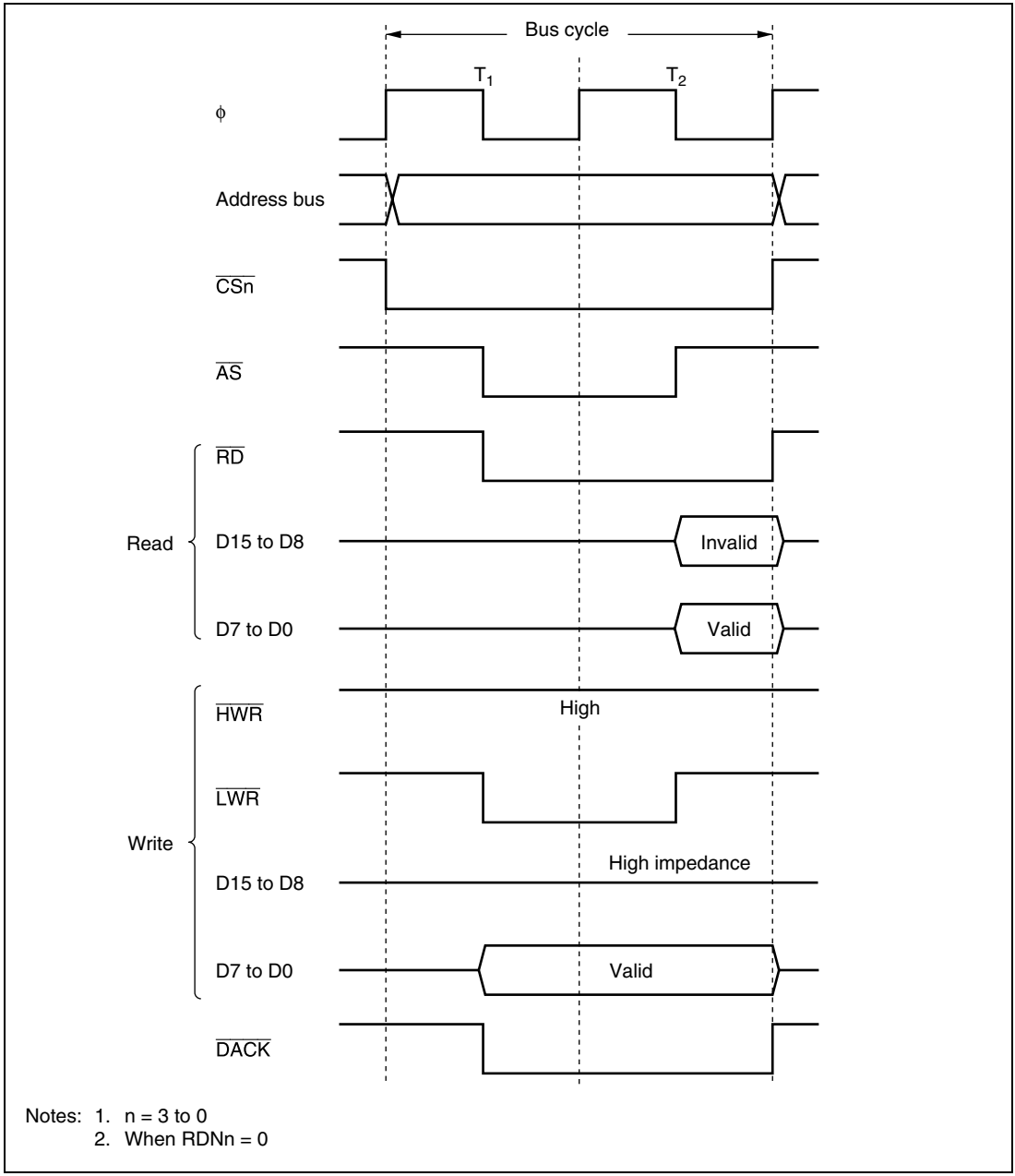
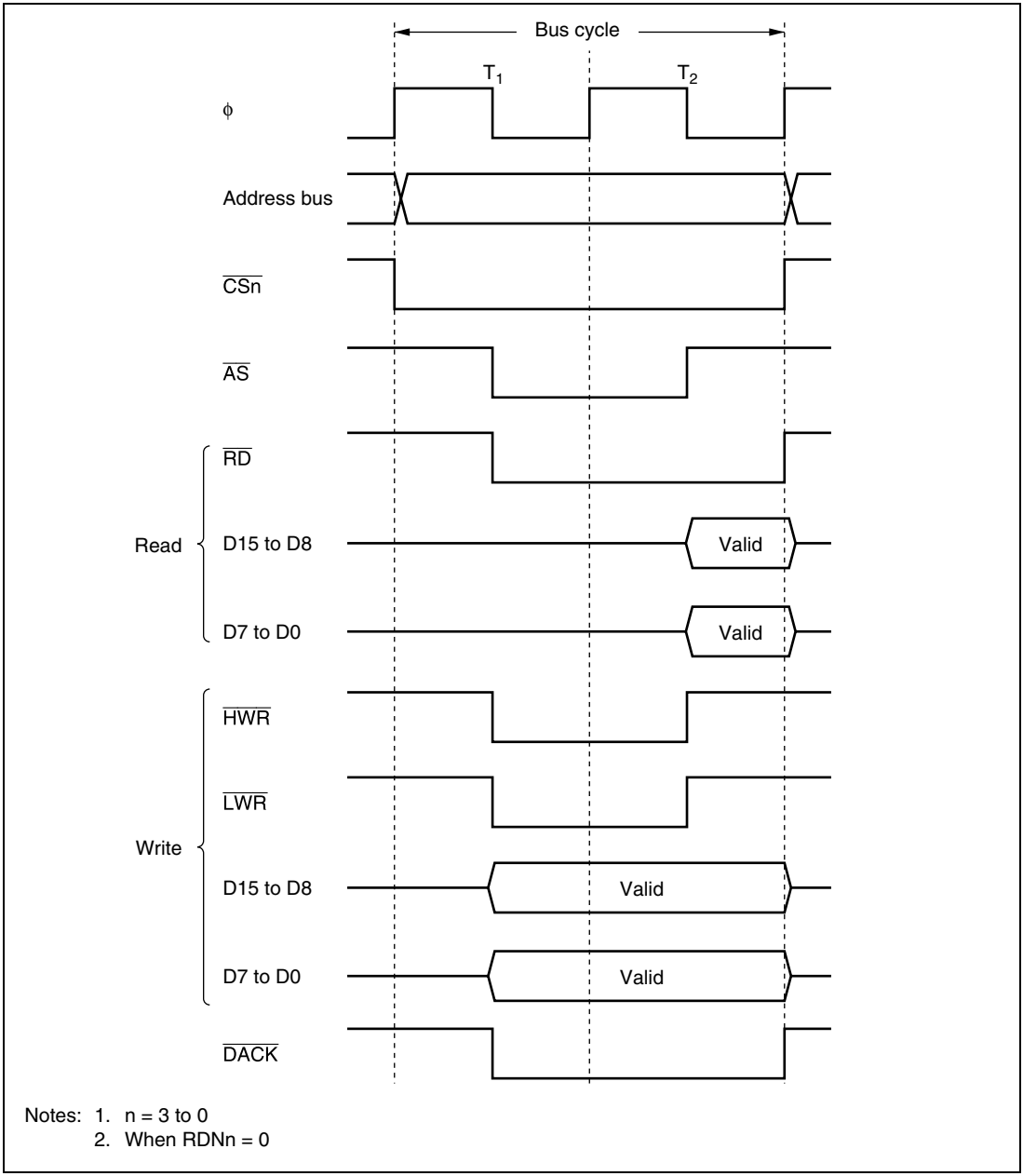


Figure 6.13 Bus Timing for 16-Bit, 2-State Access Space (Even Address Byte Access)



Notes: 1. n = 3 to 0
2. When RDNn = 0

Figure 6.14 Bus Timing for 16-Bit, 2-State Access Space (Odd Address Byte Access)



Notes: 1. n = 3 to 0
2. When RDNn = 0

Figure 6.15 Bus Timing for 16-Bit, 2-State Access Space (Word Access)

16-Bit, 3-State Access Space: Figures 6.16 to 6.18 show bus timings for a 16-bit, 3-state access space. When a 16-bit access space is accessed, the upper half (D15 to D8) of the data bus is used for the even address, and the lower half (D7 to D0) for the odd address. Wait states can be inserted.

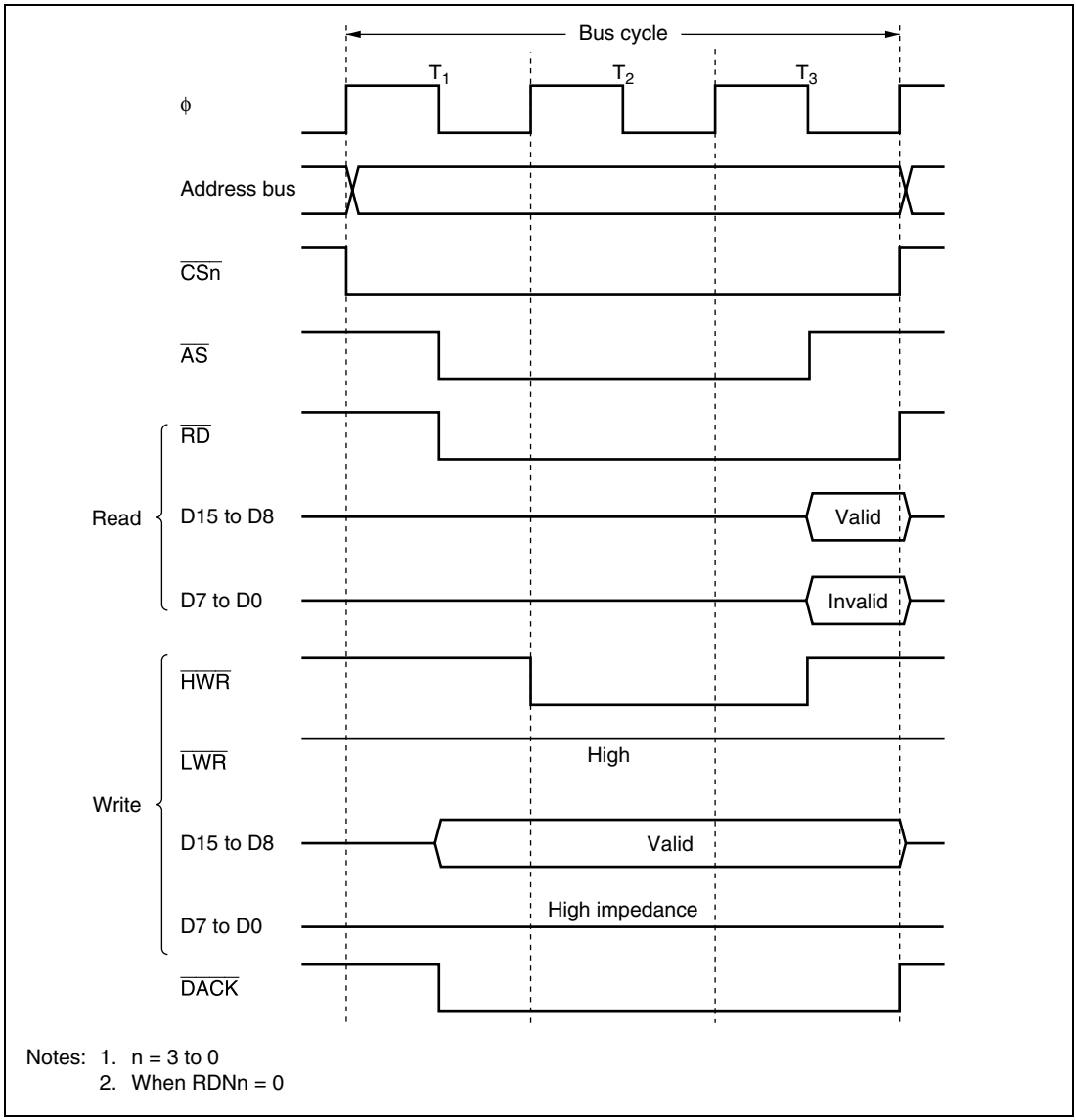


Figure 6.16 Bus Timing for 16-Bit, 3-State Access Space (Even Address Byte Access)

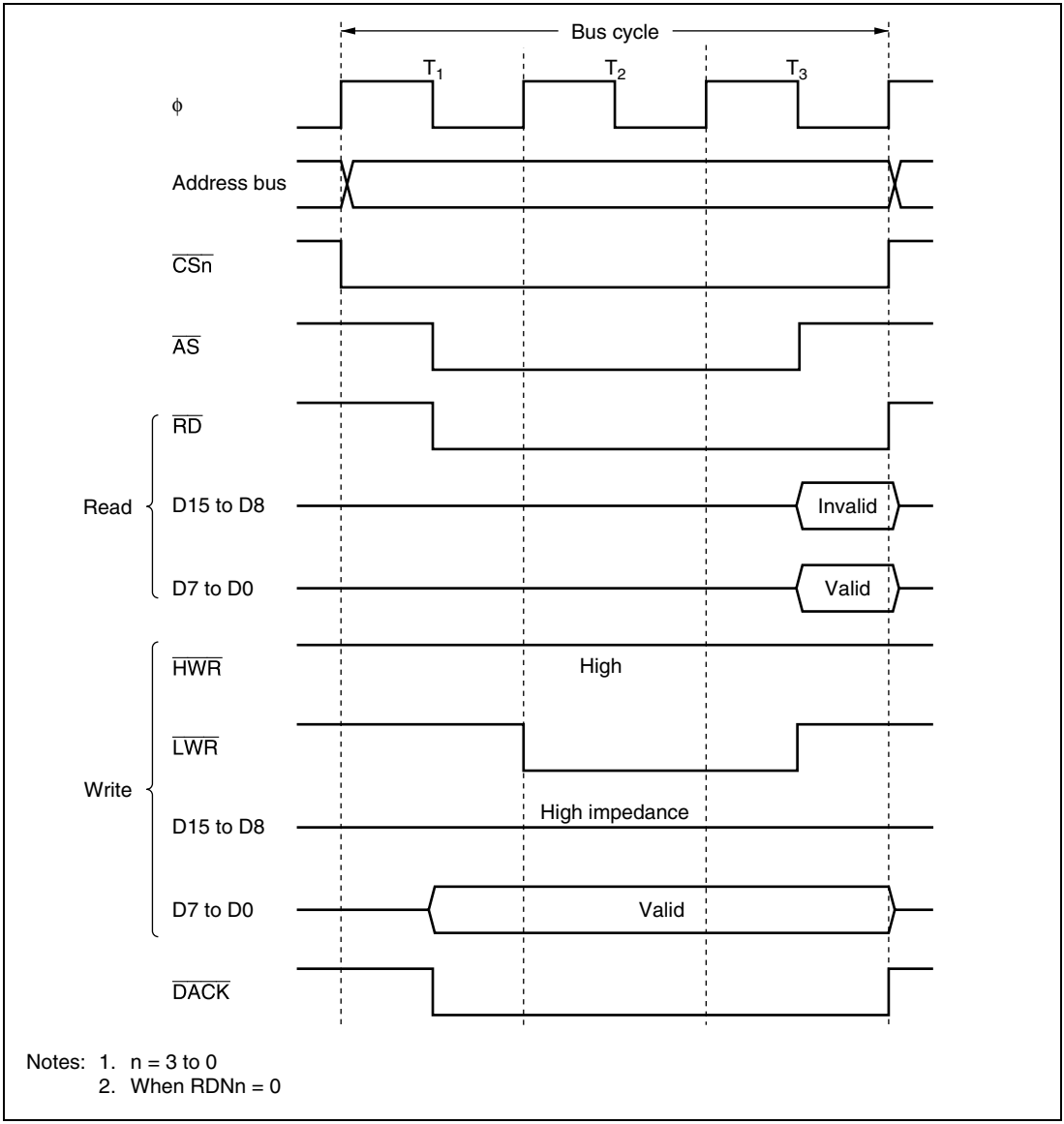
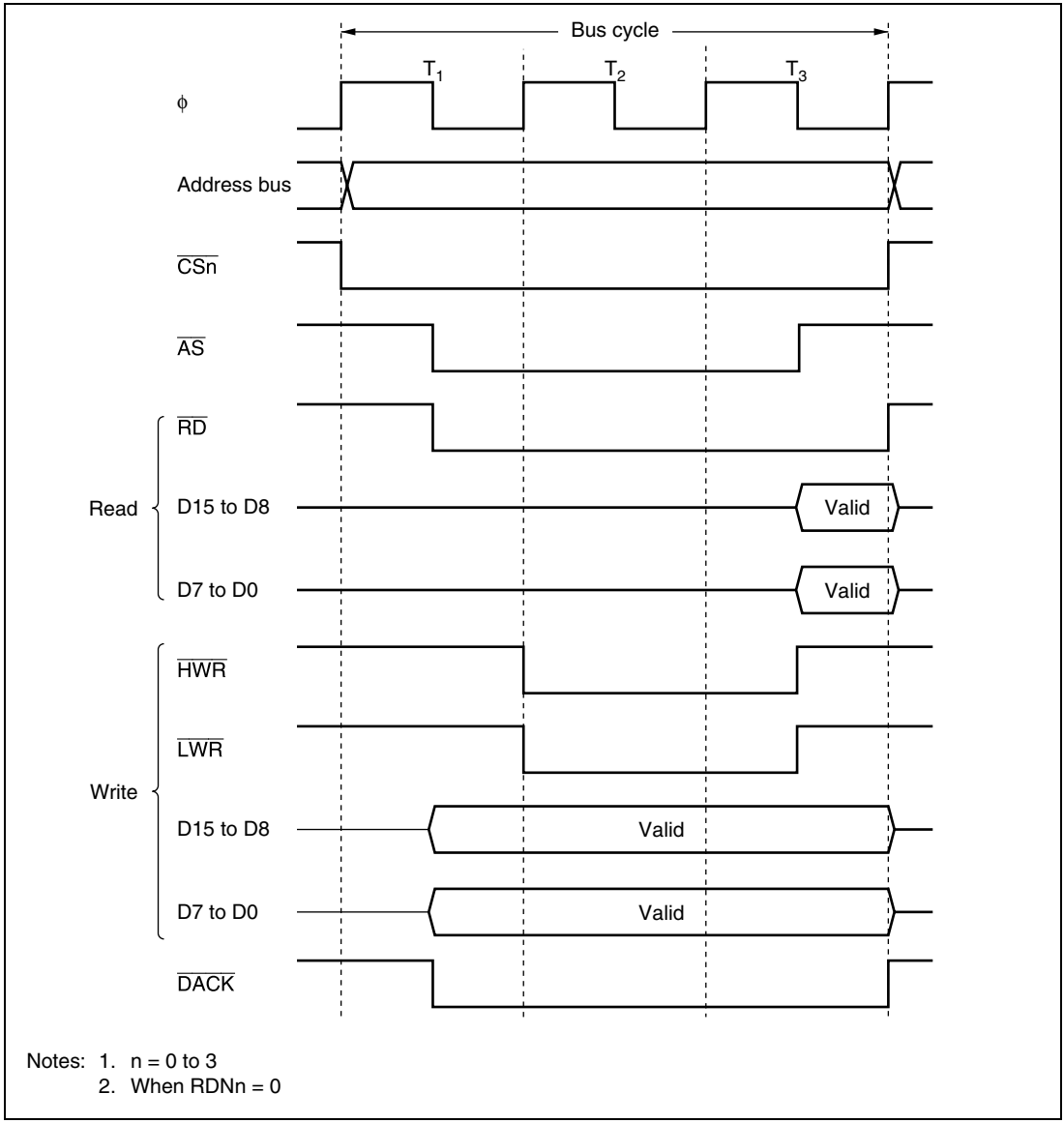


Figure 6.17 Bus Timing for 16-Bit, 3-State Access Space (Odd Address Byte Access)



Notes: 1. n = 0 to 3
2. When RDNn = 0

Figure 6.18 Bus Timing for 16-Bit, 3-State Access Space (Word Access)

6.5.4 Wait Control

When accessing external space, this LSI can extend the bus cycle by inserting one or more wait states (T_w).

From 0 to 7 wait states can be inserted automatically between the T_2 state and T_3 state on an individual area basis in 3-state access space, according to the setting of WTCR.

Figure 6.19 shows an example of wait state insertion timing.

The settings after a reset are: 3-state access and insertion of 7 program wait states.

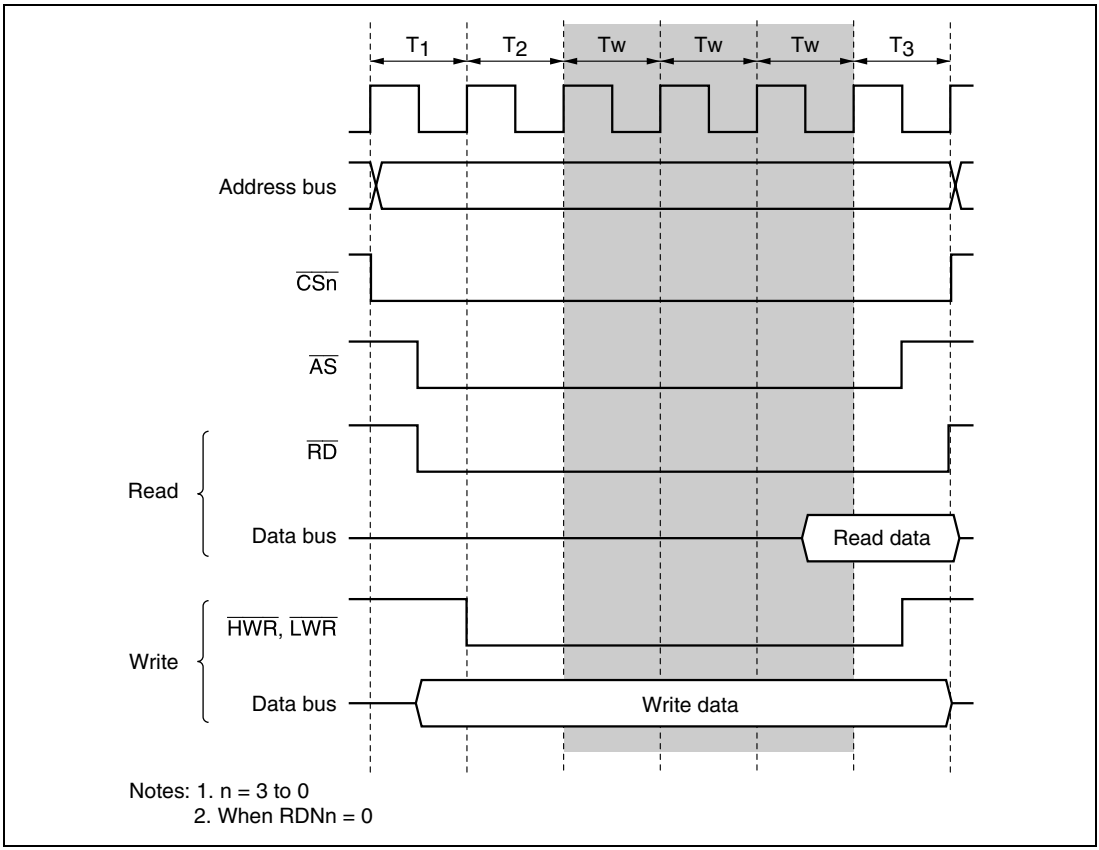


Figure 6.19 Example of Wait State Insertion Timing

6.5.5 Read Strobe ($\overline{RD}$) Timing

The read strobe ($\overline{RD}$) timing can be changed for individual areas by setting bits RDN3 to RDN0 to 1 in RDNCR.

When the DMAC is used in single address mode, note that if the $\overline{RD}$ timing is changed by setting RDNn to 1, the $\overline{RD}$ timing will change relative to the rise of $\overline{DACK}$.

Figure 6.20 shows an example of the timing when the read strobe timing is changed in basic bus 3-state access space.

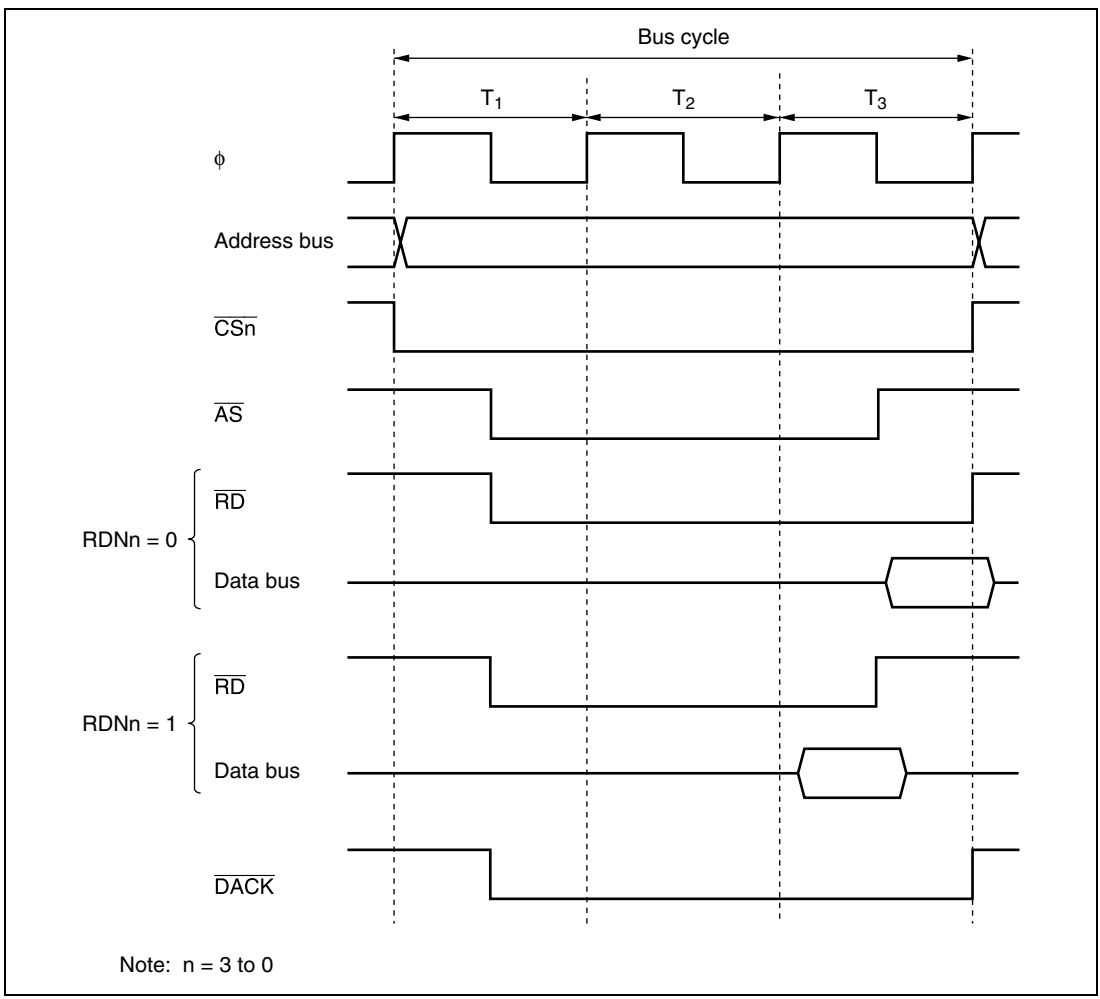


Figure 6.20 Example of Read Strobe Timing

6.5.6 Extension of Chip Select ($\overline{CS}$) Assertion Period

Some external I/O devices require a setup time and hold time between address and $\overline{CS}$ signals and strobe signals such as $\overline{RD}$, $\overline{HWR}$, and $\overline{LWR}$. Settings can be made in the CSACR register to insert states in which only the $\overline{CS}$, $\overline{AS}$, and address signals are asserted before and after a basic bus space access cycle. Extension of the $\overline{CS}$ assertion period can be set for individual areas. With the $\overline{CS}$ assertion extension period in write access, the data setup and hold times are less stringent since the write data is output to the data bus.

Figure 6.21 shows an example of the timing when the $\overline{CS}$ assertion period is extended in basic bus 3-state access space.

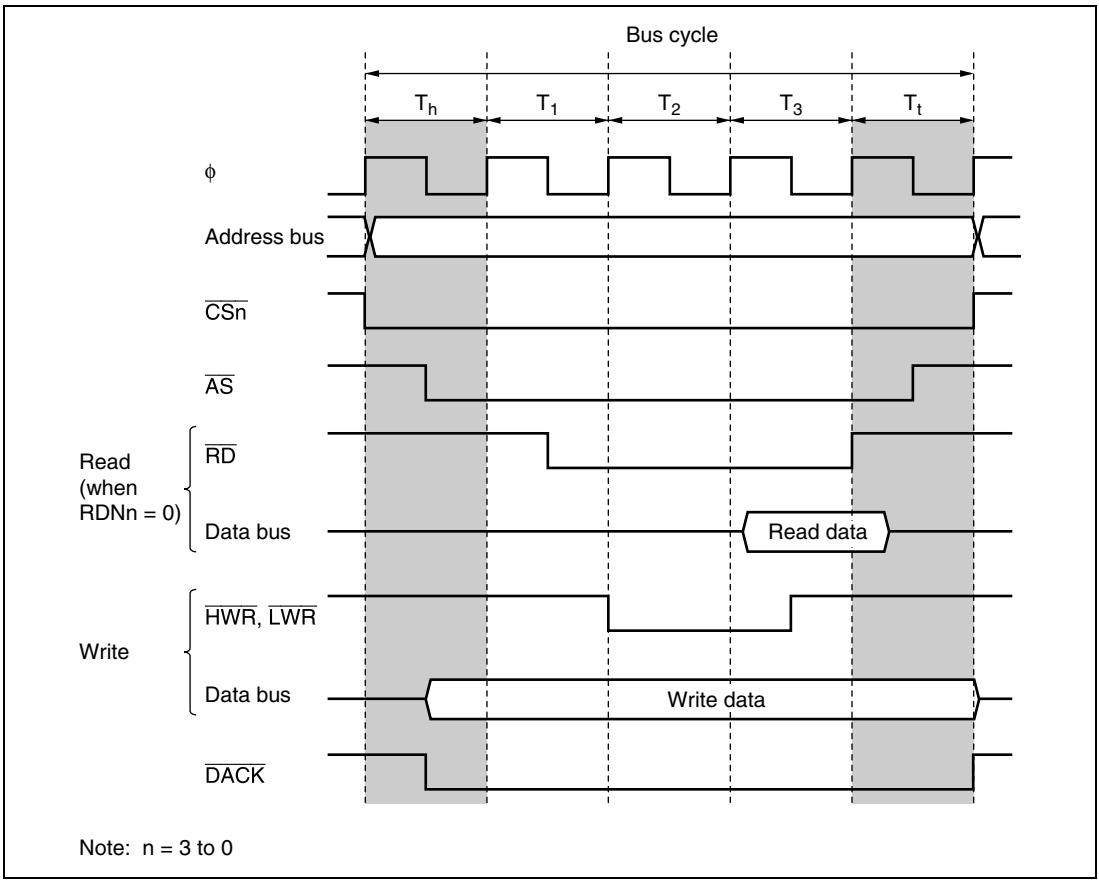


Figure 6.21 Example of Timing when Chip Select Assertion Period is Extended

Both extension state T_h inserted before the basic bus cycle and extension state T_t inserted after the basic bus cycle, or only one of these, can be specified for individual areas. Insertion or non-insertion can be specified for the T_h state with the upper 4 bits (CSXH3 to CSXH0) in the CSACR register, and for the T_t state with the lower 4 bits (CSXT3 to CSXT0).

6.6 DRAM Interface

In this LSI, external space area 2 can be designated as DRAM space, and DRAM interfacing performed. The DRAM interface allows DRAM to be directly connected to this LSI. A DRAM space of 10 Mbytes can be set by means of bit DSET in DRAMCR. Burst operation is also possible, using fast page mode.

6.6.1 Setting DRAM Space

Area 2 is designated as DRAM space by setting bit DSET in DRAMCR to 1.

In DRAM space, the $\overline{RAS}$ signal is valid. The bus specifications for DRAM space such as the bus width, number of wait states, and so on are determined according to the settings for area 2.

6.6.2 Address Multiplexing

With DRAM space, the row address and column address are multiplexed. In address multiplexing, the size of the shift of the row address is selected with bits MXC2 to MXC0 in DRAMCR. Table 6.5 shows the relation between the settings of bits MXC2 to MXC0 and the shift size.

Table 6.5 Relation between Settings of Bits MXC2 to MXC0 and Address Multiplexing

	DRAMCR				Address Pins																	
	MXC2	MXC1	MXC0	Shift Size	A19 to A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Row address	0	0	0	8 bits	A19 to A16	A23	A22	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8
			1	9 bits	A19 to A16	A15	A23	A22	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9
		1	0	10 bits	A19 to A16	A15	A14	A23	A22	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10
			1	11 bits	A19 to A16	A15	A14	A13	A23	A22	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11
	1	—	—	Reserved (setting prohibited)	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
Column address	—	—	—	—	A19 to A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0

6.6.3 Data Bus

If the ABW2 bit in ACSCR is set to 1, that area is designated as 8-bit DRAM space; if the bit is cleared to 0, the area is designated as 16-bit DRAM space. In 16-bit DRAM space, ×16-bit configuration DRAM can be connected directly.

In 8-bit DRAM space the upper half of the data bus, D15 to D8, is enabled, while in 16-bit DRAM space both the upper and lower halves of the data bus, D15 to D0, are enabled.

Access sizes and data alignment are the same as for the basic bus interface: see section 6.5.1, Data Size and Data Alignment.

6.6.4 Pins Used for DRAM Interface

Table 6.6 shows the pins used for DRAM interfacing and their functions. Although the $\overline{CS2}$ pin is in the input state after a reset, the $\overline{RAS}$ signal is output after the DSET bit in DRAMCR is set and DRAM space is designated.

For details, refer to section 8, I/O Ports.

Table 6.6 DRAM Interface Pins

Pin	With DRAM Setting	Name	I/O	Function
$\overline{HWR}$	$\overline{WE}$	Write enable	Output	Write enable for DRAM space access
$\overline{CS2}$	$\overline{RAS}$	Row address strobe	Output	Row address strobe when area 2 is designated as DRAM space
$\overline{UCAS}$	$\overline{UCAS}$	Upper column address strobe	Output	Upper column address strobe for 16-bit DRAM space access or column address strobe for 8-bit DRAM space access
$\overline{LCAS}$	$\overline{LCAS}$	Lower column address strobe	Output	Lower column address strobe signal for 16-bit DRAM space access
$\overline{RD}$	$\overline{OE}$	Output enable	Output	Output enable signal for DRAM space access
A15 to A0	A15 to A0	Address pins	Output	Row address/column address multiplexed output
D15 to D0	D15 to D0	Data pins	I/O	Data input/output pins

6.6.5 Basic Timing

Figure 6.22 shows the basic access timing for DRAM space.

The four states of the basic timing consist of one T_p (precharge cycle) state, one T_r (row address output cycle) state, and two T_{c1} and T_{c2} (column address output cycle) states.

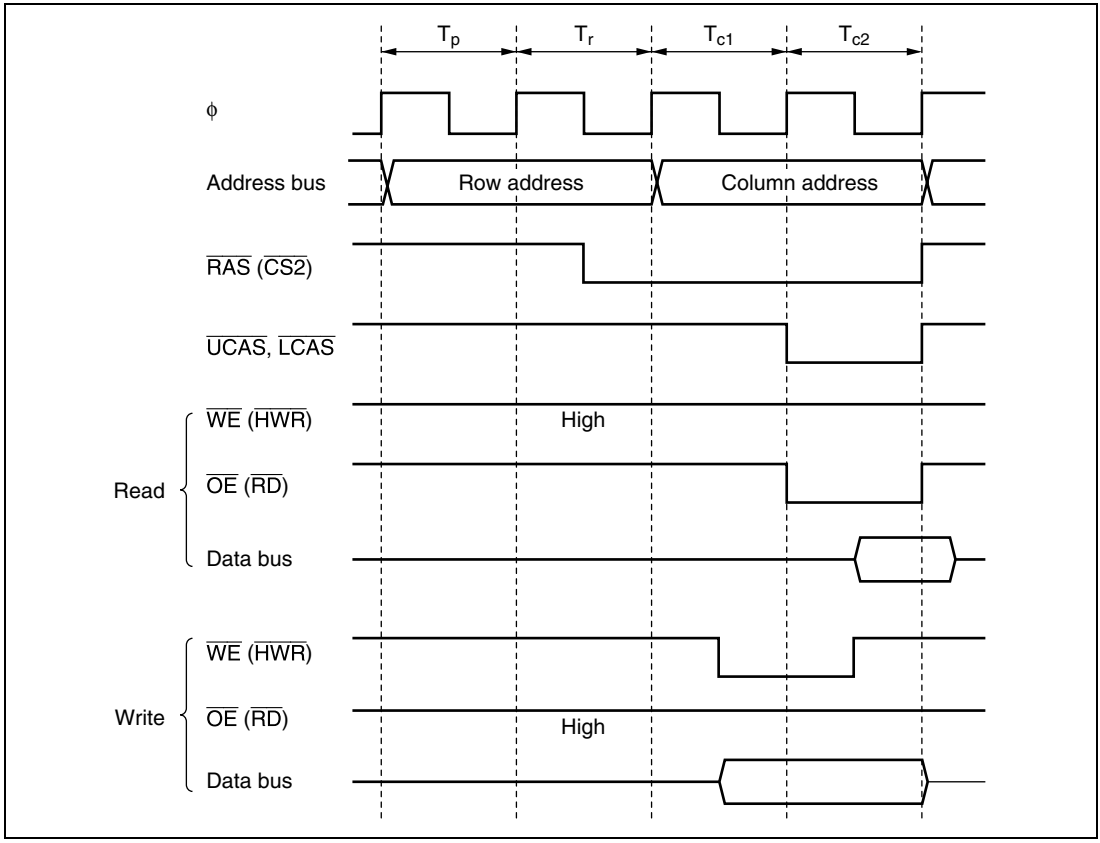


Figure 6.22 DRAM Basic Access Timing (RAST = 0, CAST = 0)

When DRAM space is accessed, the $\overline{RD}$ signal is output as the $\overline{OE}$ signal for DRAM. When connecting DRAM provided with an EDO page mode, the $\overline{OE}$ signal should be connected to the ($\overline{OE}$) pin of the DRAM.

6.6.6 Column Address Output Cycle Control

The column address output cycle can be changed from 2 states to 3 states by setting the CAST bit to 1 in DRAMCR. Use the setting that gives the optimum specification values ($\overline{CAS}$ pulse width, etc.) according to the DRAM connected and the operating frequency of this LSI. Figure 6.23 shows an example of the timing when a 3-state column address output cycle is selected.

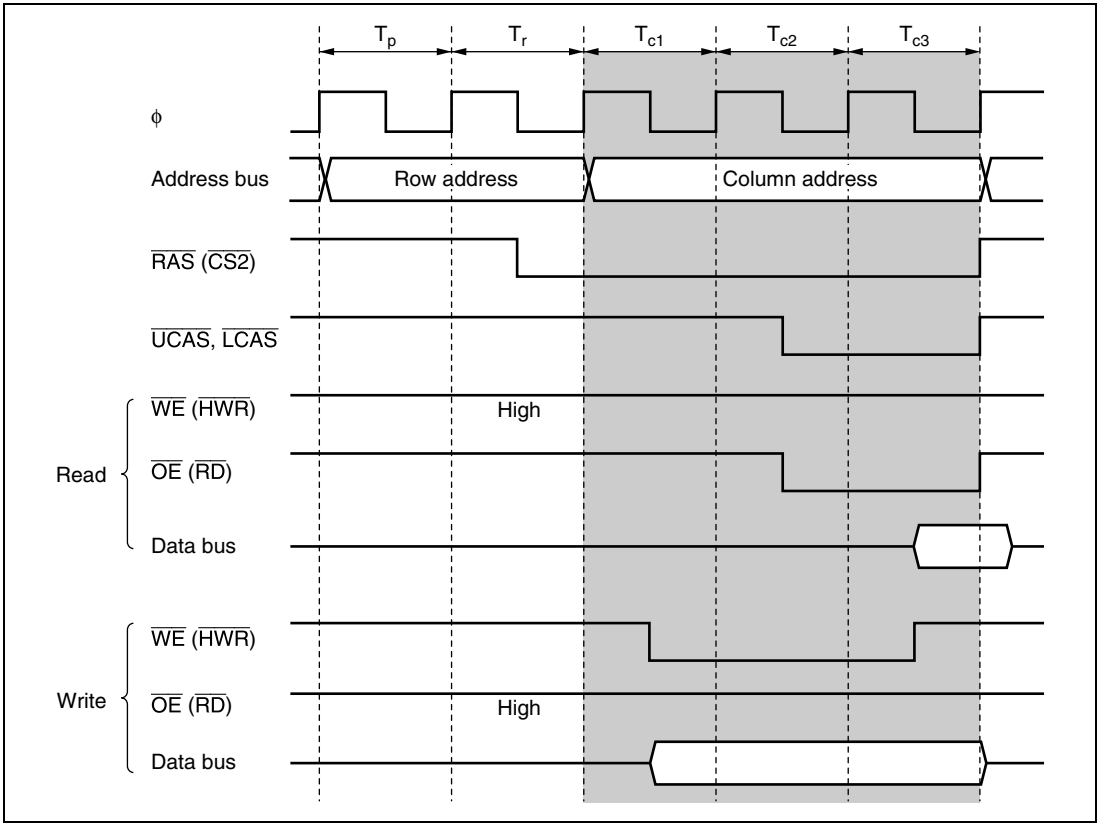


Figure 6.23 Example of Access Timing with 3-State Column Address Output Cycle (RAST = 0)

6.6.7 Row Address Output State Control

If the RAST bit is set to 1 in DRAMCR, the $\overline{RAS}$ signal goes low from the beginning of the T_r state, and the row address hold time and DRAM read access time are changed relative to the fall of the $\overline{RAS}$ signal. Use the optimum setting according to the DRAM connected and the operating frequency of this LSI. Figure 6.24 shows an example of the timing when the $\overline{RAS}$ signal goes low from the beginning of the T_r state.

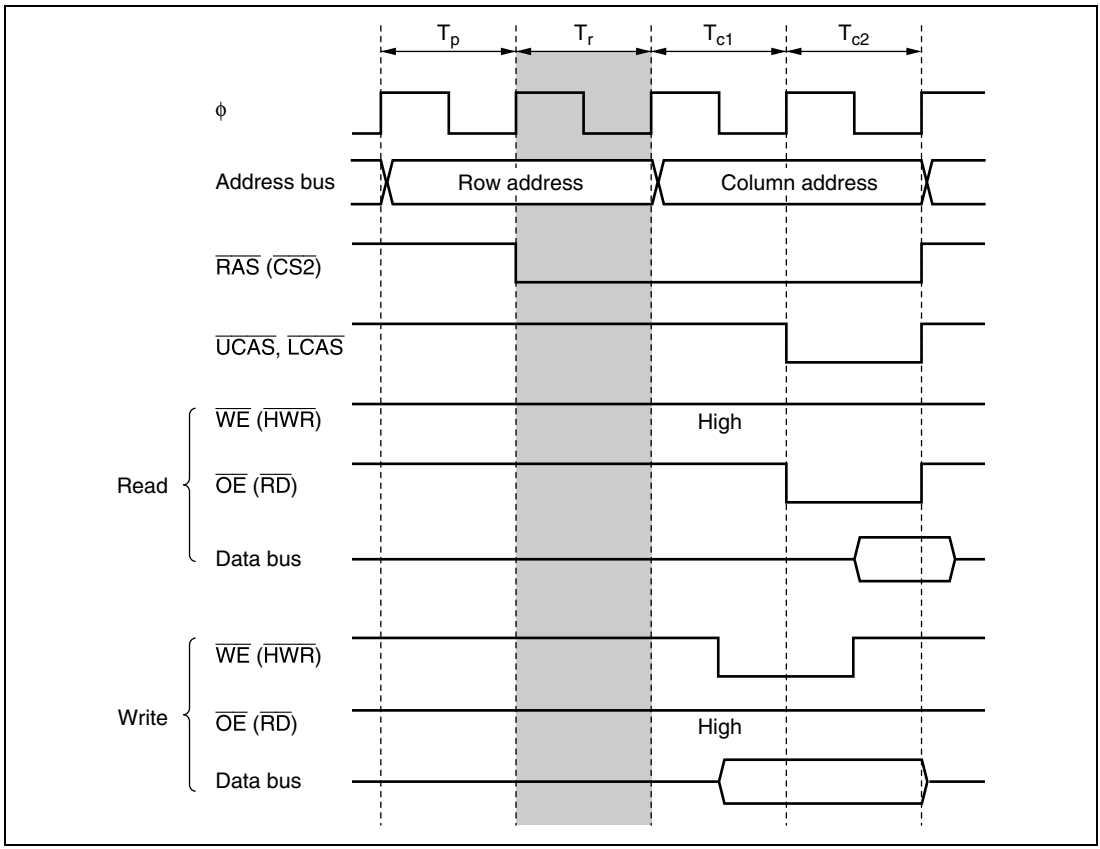


Figure 6.24 Example of Access Timing when $\overline{RAS}$ Signal Goes Low from Beginning of T_r State (CAST = 0)

If a row address hold time or read access time is necessary, making a setting in bits RCD1 and RCD0 in DRACCR allows from one to three T_{rw} states, in which row address output is maintained, to be inserted between the T_r cycle, in which the $\overline{RAS}$ signal goes low, and the T_{c1} cycle, in which the column address is output. Use the setting that gives the optimum row address signal hold time relative to the falling edge of the $\overline{RAS}$ signal according to the DRAM connected and the operating frequency of this LSI. Figure 6.25 shows an example of the timing when one T_{rw} state is set.

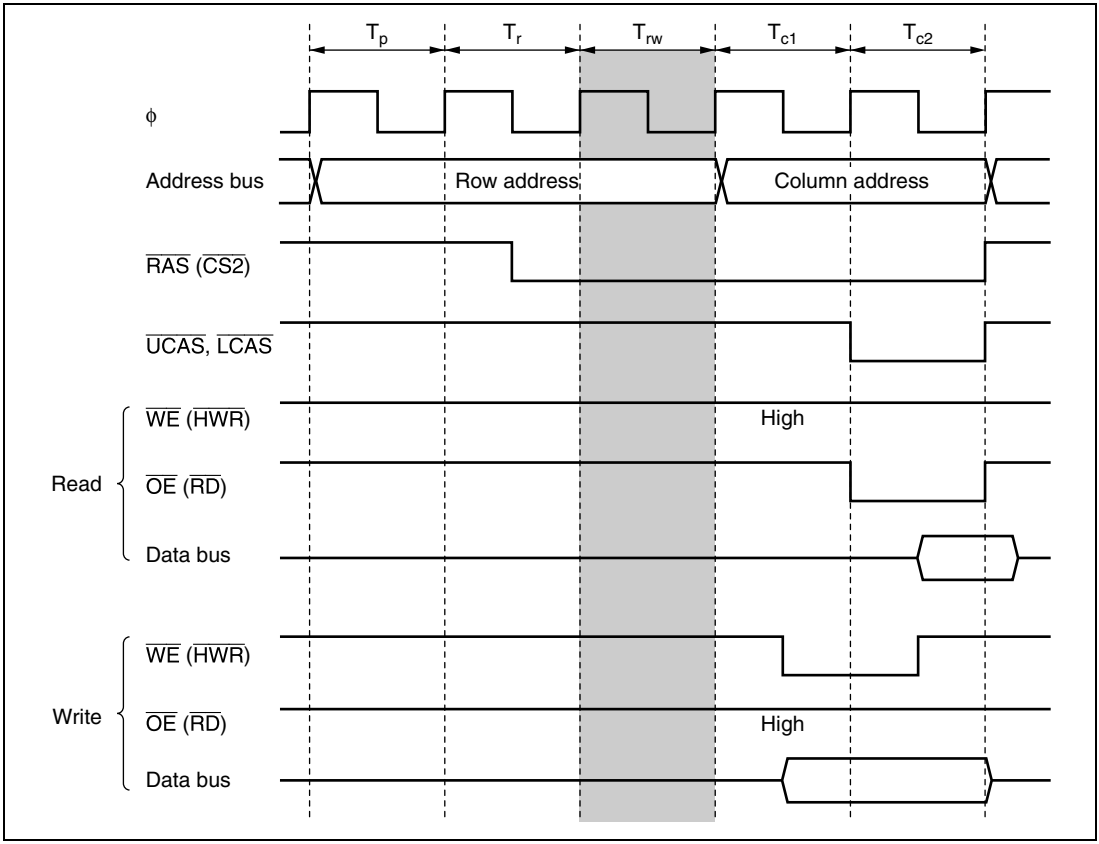


Figure 6.25 Example of Timing with One Row Address Output Hold State (RAST = 0, CAST = 0)

6.6.8 Precharge State Control

When DRAM is accessed, a $\overline{RAS}$ precharge time must be secured. With this LSI, one T_p state is always inserted when DRAM space is accessed. From one to four T_p states can be selected by setting bits TPC1 and TPC0 in DRACCR. Set the optimum number of T_p cycles according to the DRAM connected and the operating frequency of this LSI. Figure 6.26 shows the timing when two T_p states are inserted. The setting of bits TPC1 and TPC0 is also valid for T_p states in refresh cycles.

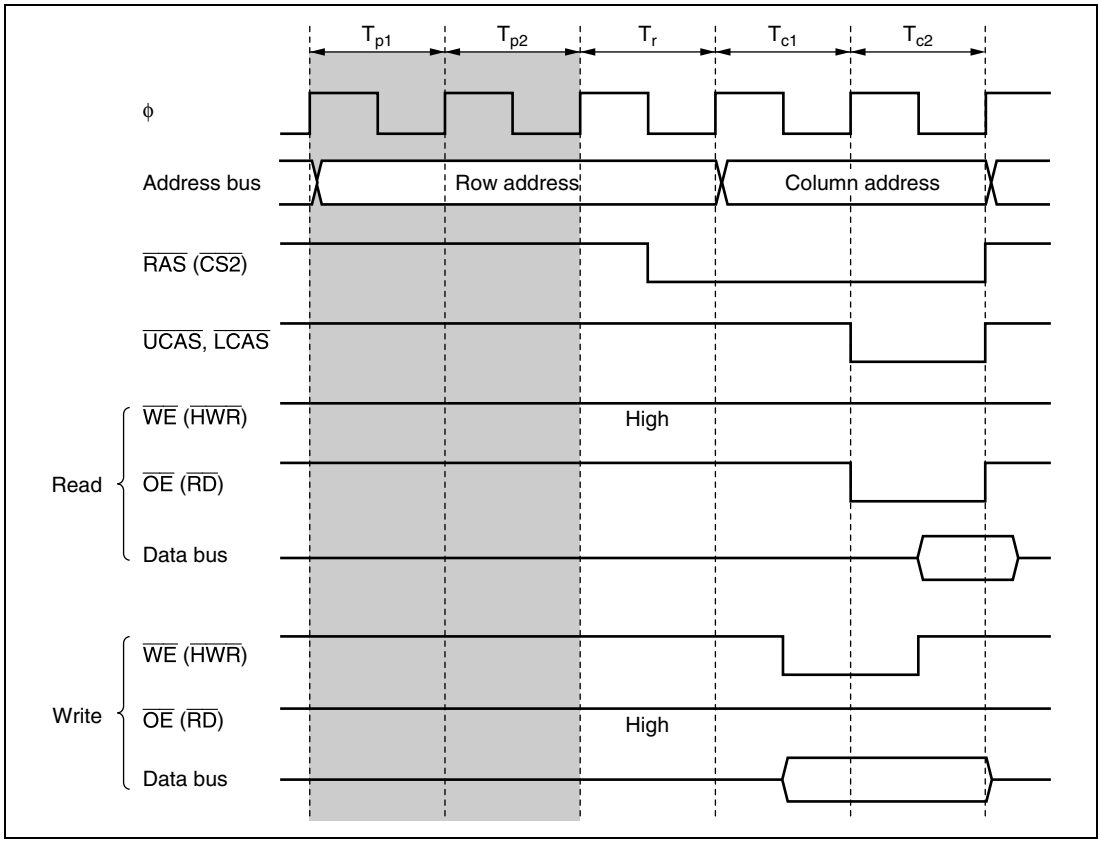


Figure 6.26 Example of Timing with Two-State Precharge Cycle (RAST = 0, CAST = 0)

6.6.9 Wait Control

When inserting wait states in a DRAM access cycle, program wait insertion is specified.

Wait states are inserted to extend the $\overline{CAS}$ assertion period in a read access to DRAM space, and to extend the write data setup time relative to the falling edge of $\overline{CAS}$ in a write access.

When the AST2 bit in ACSCR is set to 1, from 0 to 7 wait states can be inserted automatically between the T_{c1} state and T_{c2} state, according to the settings of WTCR.

Figures 6.27 and 6.28 show examples of wait cycle insertion timing in the case of 2-state and 3-state column address output cycles.

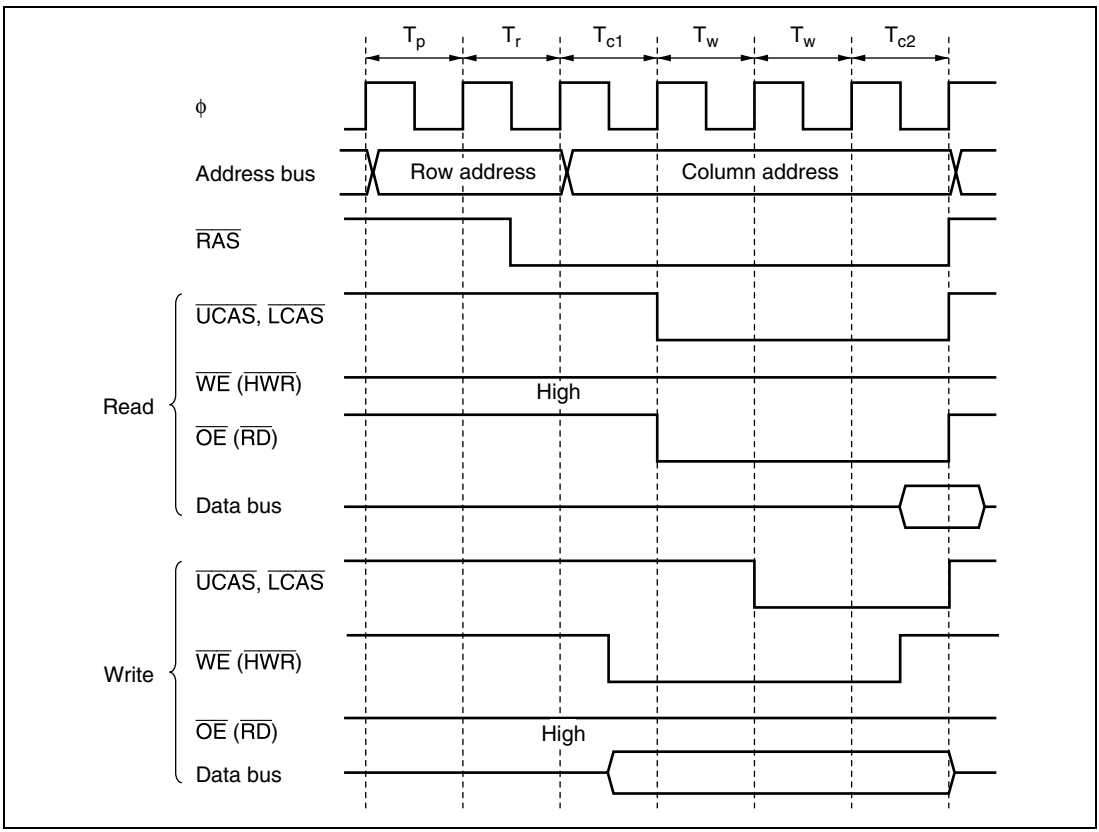


Figure 6.27 Example of Wait State Insertion Timing (2-State Column Address Output)

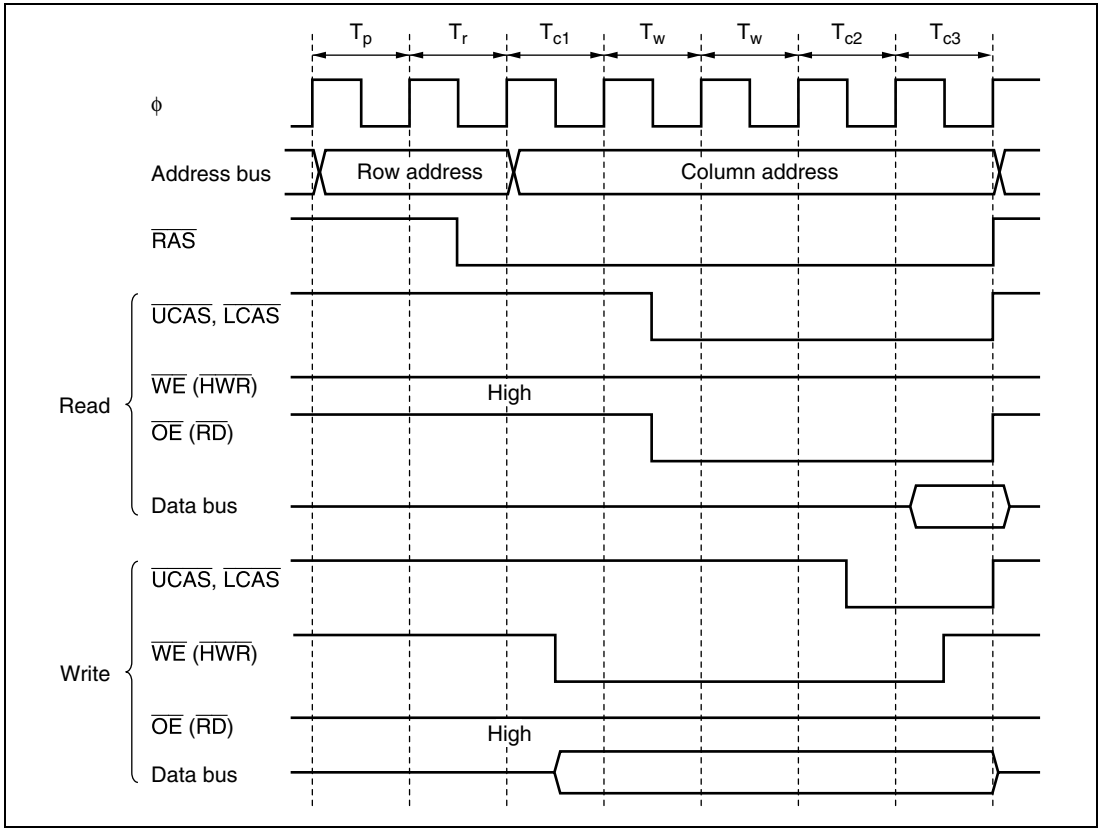


Figure 6.28 Example of Wait State Insertion Timing (3-State Column Address Output)

6.6.10 Byte Access Control

When DRAM with a ×16-bit configuration is connected, the 2-CAS access method is used for the control signals needed for byte access. Figure 6.29 shows the control timing for 2-CAS access, and figure 6.30 shows an example of 2-CAS DRAM connection.

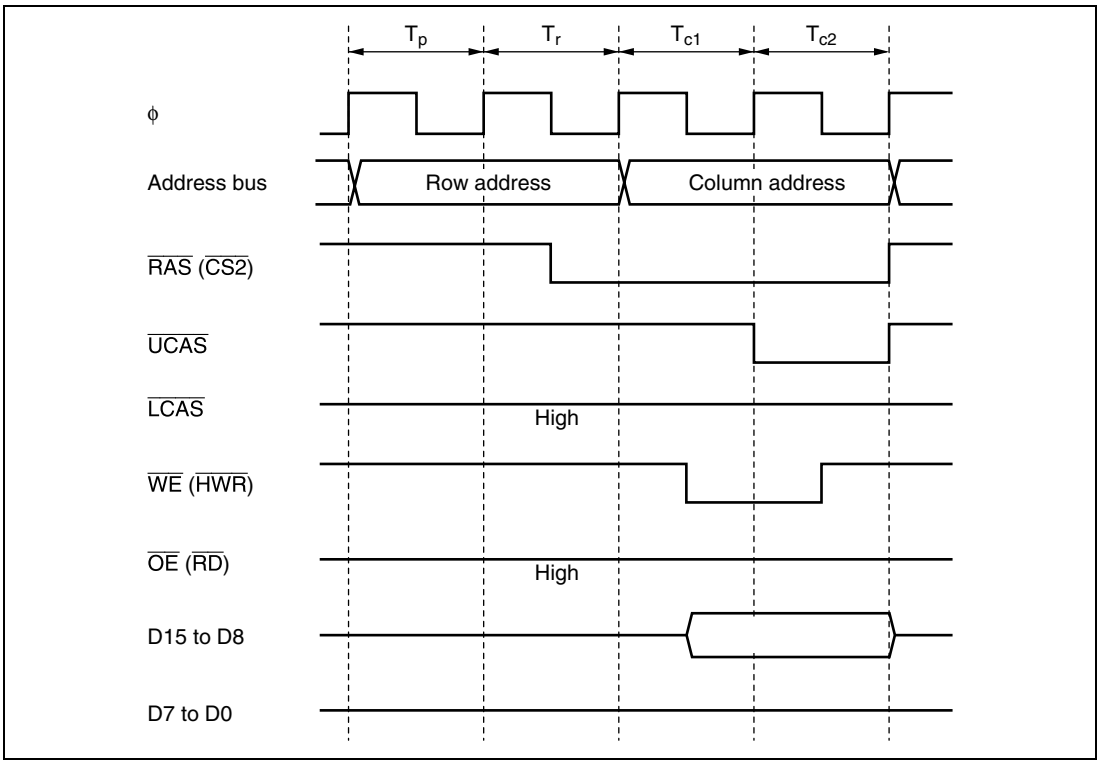


Figure 6.29 2-CAS Control Timing (Write Access to Even Address: RAST = 0, CAST = 0)

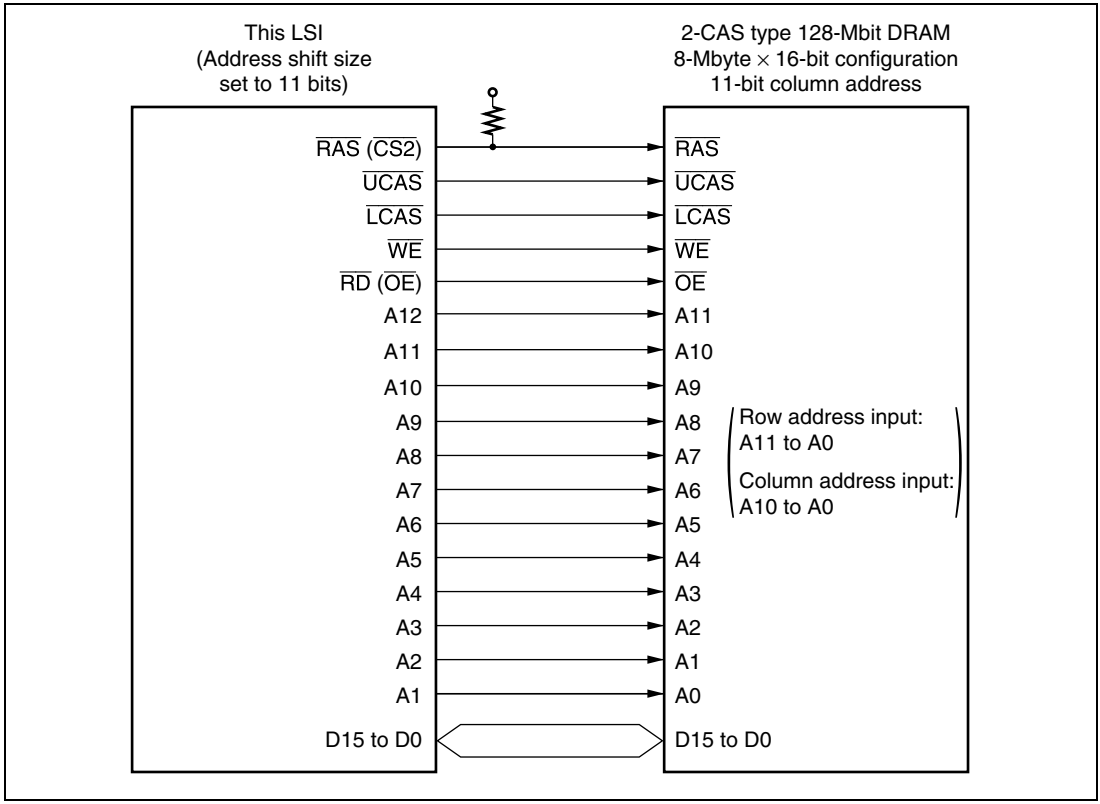


Figure 6.30 Example of 2-CAS DRAM Connection

6.6.11 Burst Operation

With DRAM, in addition to full access (normal access) in which data is accessed by outputting a row address for each access, a fast page mode is also provided which can be used when making consecutive accesses to the same row address. This mode enables fast (burst) access of data by simply changing the column address after the row address has been output. Burst access can be selected by setting the BE bit to 1 in DRAMCR.

Burst Access (Fast Page Mode): Figures 6.31 and 6.32 show the operation timing for burst access. When there are consecutive access cycles for DRAM space, the $\overline{\text{CAS}}$ signal and column address output cycles (two states) continue as long as the row address is the same for consecutive access cycles. The row address used for the comparison is set with bits MXC2 to MXC0 in DRAMCR.

The bus cycle can also be extended in burst access by inserting wait states. The wait state insertion method and timing are the same as for full access. For details, see section 6.6.9, Wait Control.

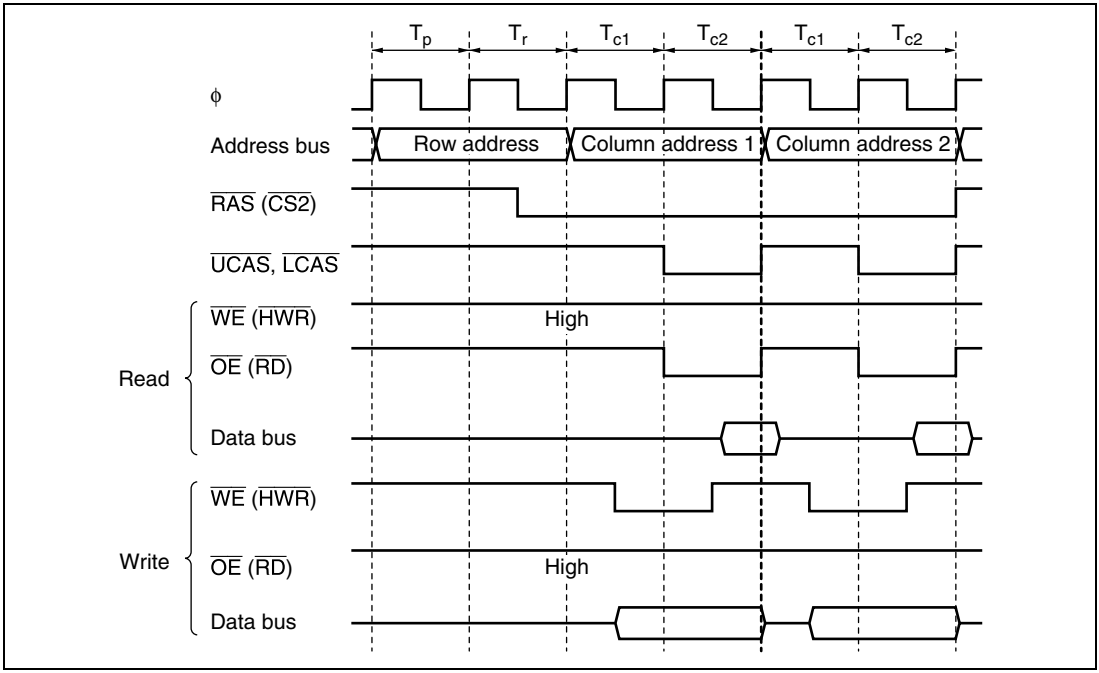


Figure 6.31 Operation Timing in Fast Page Mode (RAST = 0, CAST = 0)

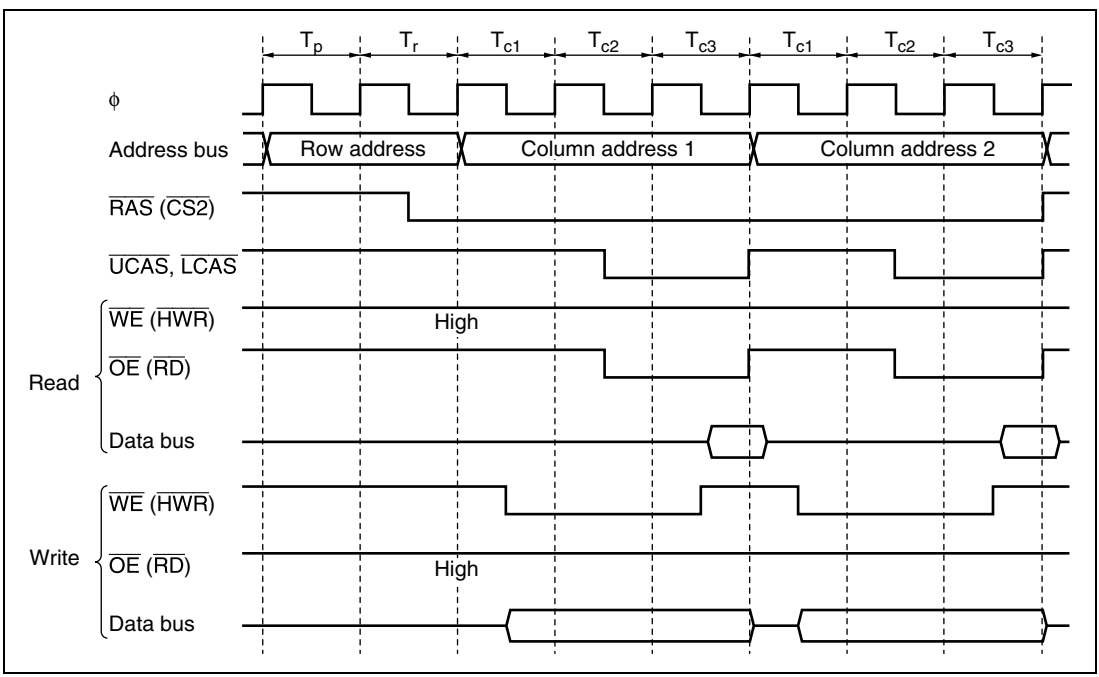


Figure 6.32 Operation Timing in Fast Page Mode (RAST = 0, CAST = 1)

RAS Down Mode and RAS Up Mode: Even when burst operation is selected, it may happen that access to DRAM space is not continuous, but is interrupted by access to another space. In this case, if the $\overline{RAS}$ signal is held low during the access to the other space, burst operation can be resumed when the same row address in DRAM space is accessed again.

- RAS Down Mode

  To select RAS down mode, set both the RCDM bit and the BE bit to 1 in DRAMCR. If access to DRAM space is interrupted and another space is accessed, the $\overline{RAS}$ signal is held low during the access to the other space, and burst access is performed when the row address of the next DRAM space access is the same as the row address of the previous DRAM space access. Figure 6.33 shows an example of the timing in RAS down mode.

  When the row address for the next DRAM space access does not match the row address for the previous DRAM space access and the RAS down state cannot be continued, one-state RAS up cycle (T_{RU}) is inserted immediately before the DRAM access. Figure 6.34 shows an example of the idle cycle insertion when RAS down mode is not continued.

  Note, however, that the $\overline{RAS}$ signal will go high if:

  — a refresh operation is initiated in the RAS down state

  — self-refreshing is performed

  — the chip enters software standby mode

  — the RCDM bit or BE bit is cleared to 0

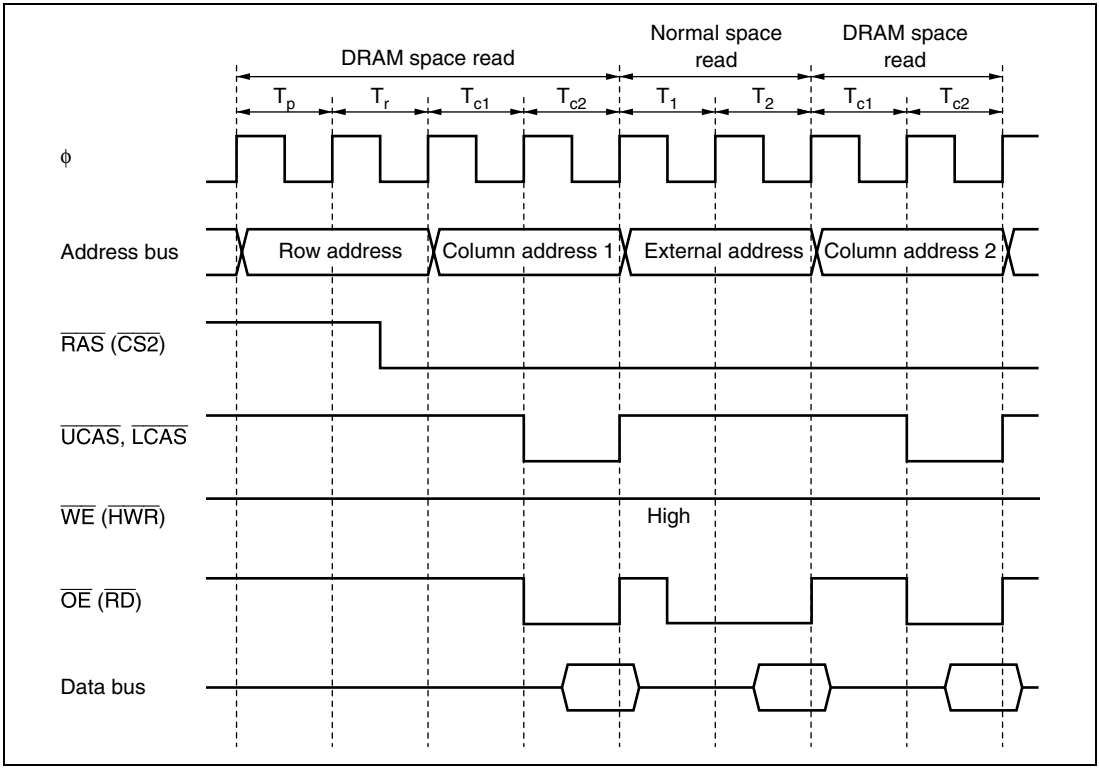


Figure 6.33 Example of Operation Timing in RAS Down Mode (RAST = 0, CAST = 0)

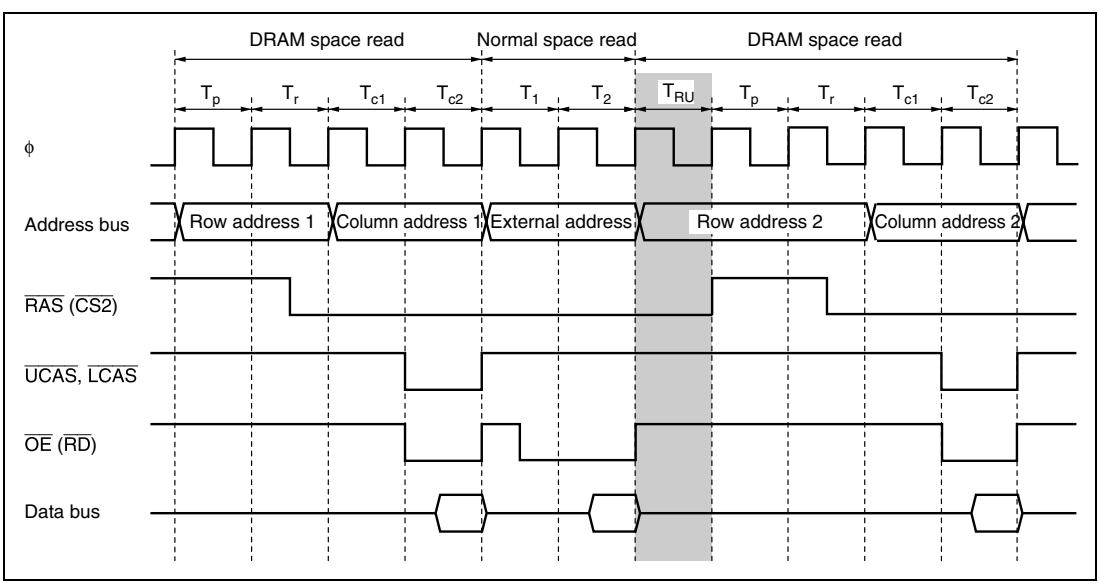


Figure 6.34 Example of Idle Cycle Insertion when RAS Down Mode cannot be Continued

- RAS Up Mode

  To select RAS up mode, clear the RCDM bit to 0 in DRAMCR. Each time access to DRAM space is interrupted and another space is accessed, the $\overline{RAS}$ signal goes high again. Burst operation is only performed if DRAM space is continuous. Figure 6.35 shows an example of the timing in RAS up mode.

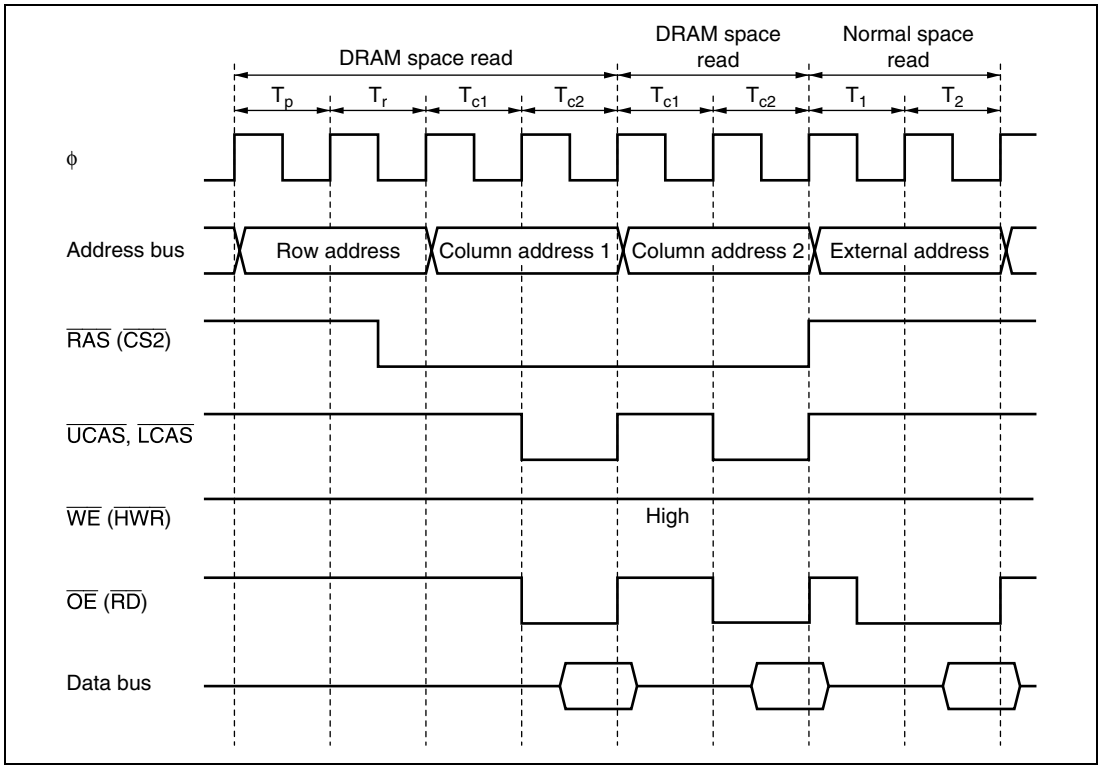


Figure 6.35 Example of Operation Timing in RAS Up Mode (RAST = 0, CAST = 0)

6.6.12 Refresh Control

This LSI is provided with a DRAM refresh control function. CAS-before-RAS (CBR) refreshing is used. In addition, self-refreshing can be executed when the chip enters the software standby state.

Refresh control is enabled when area 2 is designated as DRAM space in accordance with the setting of bit DSET in DRAMCR.

CAS-before-RAS (CBR) Refreshing: To select CBR refreshing, set the RFSHE bit to 1 in REFCR.

With CBR refreshing, RTCNT counts up using the input clock selected by bits RTCK2 to RTCK0 in REFCR, and when the count matches the value set in RTCOR (compare match), refresh control is performed. At the same time, RTCNT is reset and starts counting up again from H'00. Refreshing is thus repeated at fixed intervals determined by RTCOR and bits RTCK2 to RTCK0. Set a value in RTCOR and bits RTCK2 to RTCK0 that will meet the refreshing interval specification for the DRAM used.

When bits RTCK2 to RTCK0 in REFCR are set, RTCNT starts counting up. RTCNT and RTCOR settings should therefore be completed before setting bits RTCK2 to RTCK0. RTCNT operation is shown in figure 6.36, compare match timing in figure 6.37, and CBR refresh timing in figure 6.38.

Access to external space other than DRAM space is not possible in parallel during the CBR refresh period.

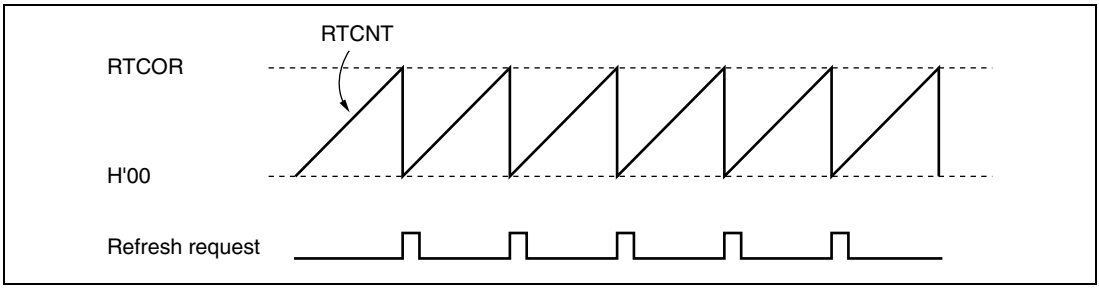


Figure 6.36 RTCNT Operation

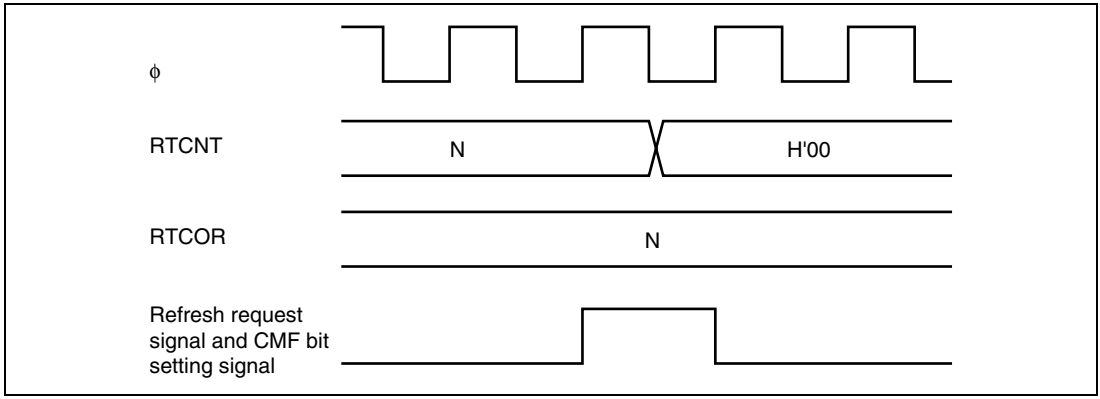


Figure 6.37 Compare Match Timing

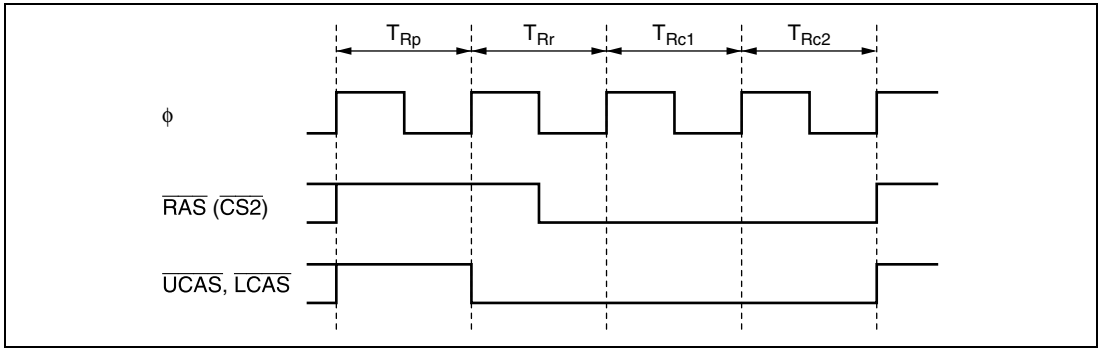


Figure 6.38 CBR Refresh Timing

A setting can be made in bits RCW1 and RCW0 in REFCR to delay $\overline{RAS}$ signal output by one to three cycles. Use bits RLW1 and RLW0 in REFCR to adjust the width of the $\overline{RAS}$ signal. The settings of bits RCW1, RCW0, RLW1, and RLW0 are valid only in refresh operations.

Figure 6.39 shows the timing when bits RCW1 and RCW0 are set.

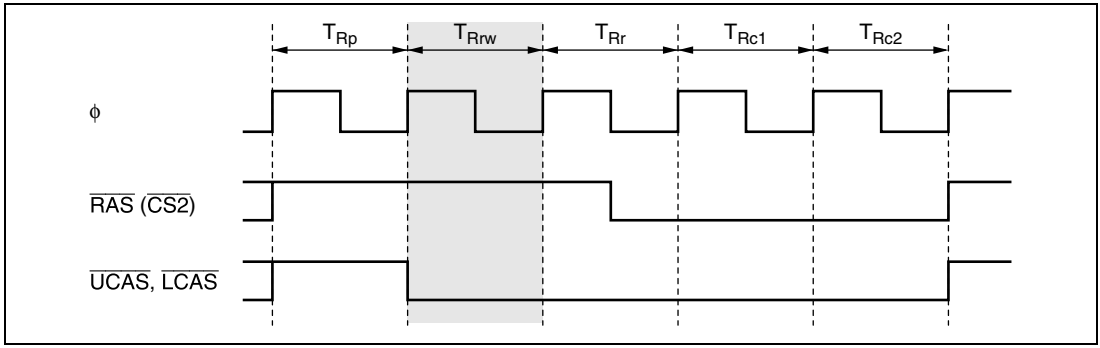


Figure 6.39 CBR Refresh Timing (RCW1 = 0, RCW0 = 1, RLW1 = 0, RLW0 = 0)

Self-Refreshing: A self-refresh mode (battery backup mode) is provided for DRAM as a kind of standby mode. In this mode, refresh timing and refresh addresses are generated within the DRAM.

To select self-refreshing, set the RFSHE bit and SLFRF bit to 1 in REFCR. When a SLEEP instruction is executed to enter software standby mode, the $\overline{CAS}$ and $\overline{RAS}$ signals are output and DRAM enters self-refresh mode, as shown in figure 6.40.

If a CBR refresh request occurs when making a transition to software standby mode, CBR refreshing is executed, then self-refresh mode is entered.

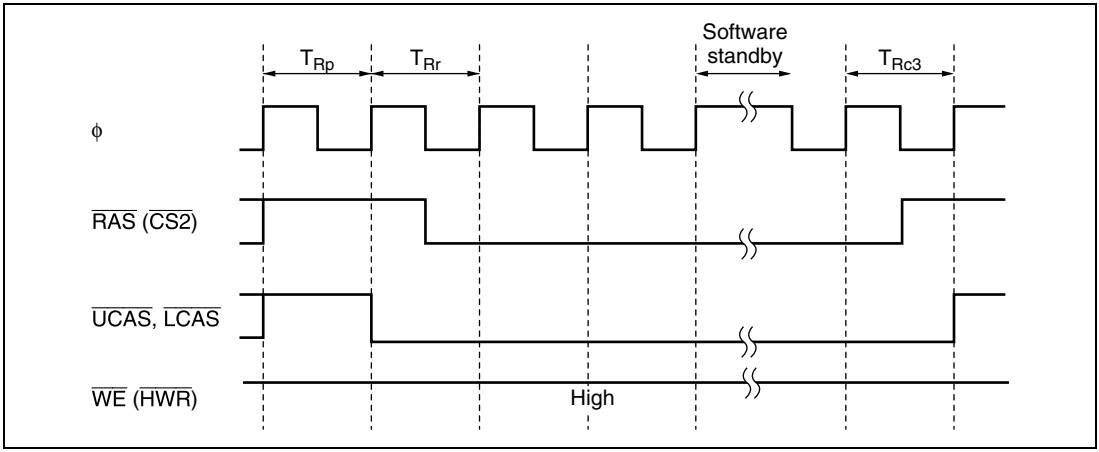


Figure 6.40 Self-Refresh Timing

In some DRAMs provided with a self-refresh mode, the $\overline{RAS}$ signal precharge time immediately after self-refreshing is longer than the normal precharge time. A setting can be made in bits TPCS2 to TPCS0 in REFCR to make the precharge time immediately after self-refreshing from 1 to 7 states longer than the normal precharge time. In this case, too, normal precharging is performed according to the setting of bits TPC1 and TPC0 in DRACCR, and therefore a setting should be made to give the optimum post-self-refresh precharge time, including this time. Figure 6.41 shows an example of the timing when the precharge time immediately after self-refreshing is extended by 2 states.

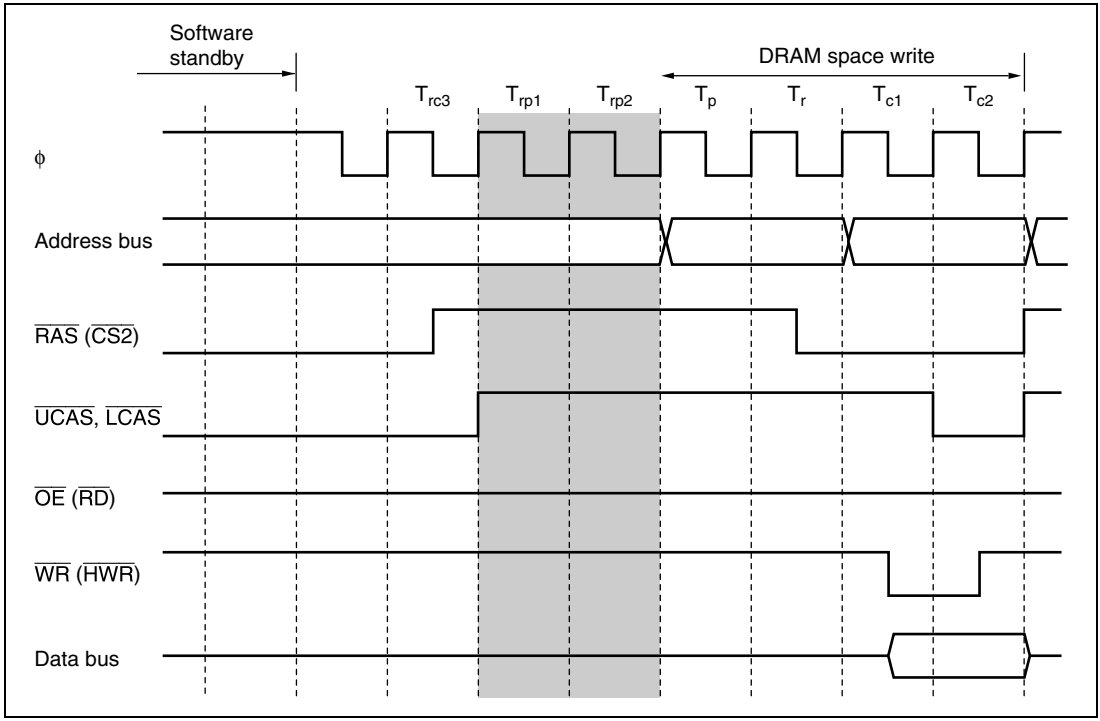


Figure 6.41 Example of Timing when Precharge Time after Self-Refreshing is Extended by 2 States

6.6.13 DMAC Single Address Transfer Mode and DRAM Interface

When burst mode is selected on the DRAM interface, the $\overline{\text{DACK}}$ output timing can be selected with the DDS bit in DRAMCR. When DRAM space is accessed in DMAC single address mode at the same time, this bit selects whether or not burst access is to be performed.

When DDS = 1: Burst access is performed by determining the address only, irrespective of the bus master. With the DRAM interface, the $\overline{\text{DACK}}$ output goes low from the T_{c1} state.

Figure 6.42 shows the $\overline{\text{DACK}}$ output timing for the DRAM interface when DDS = 1.

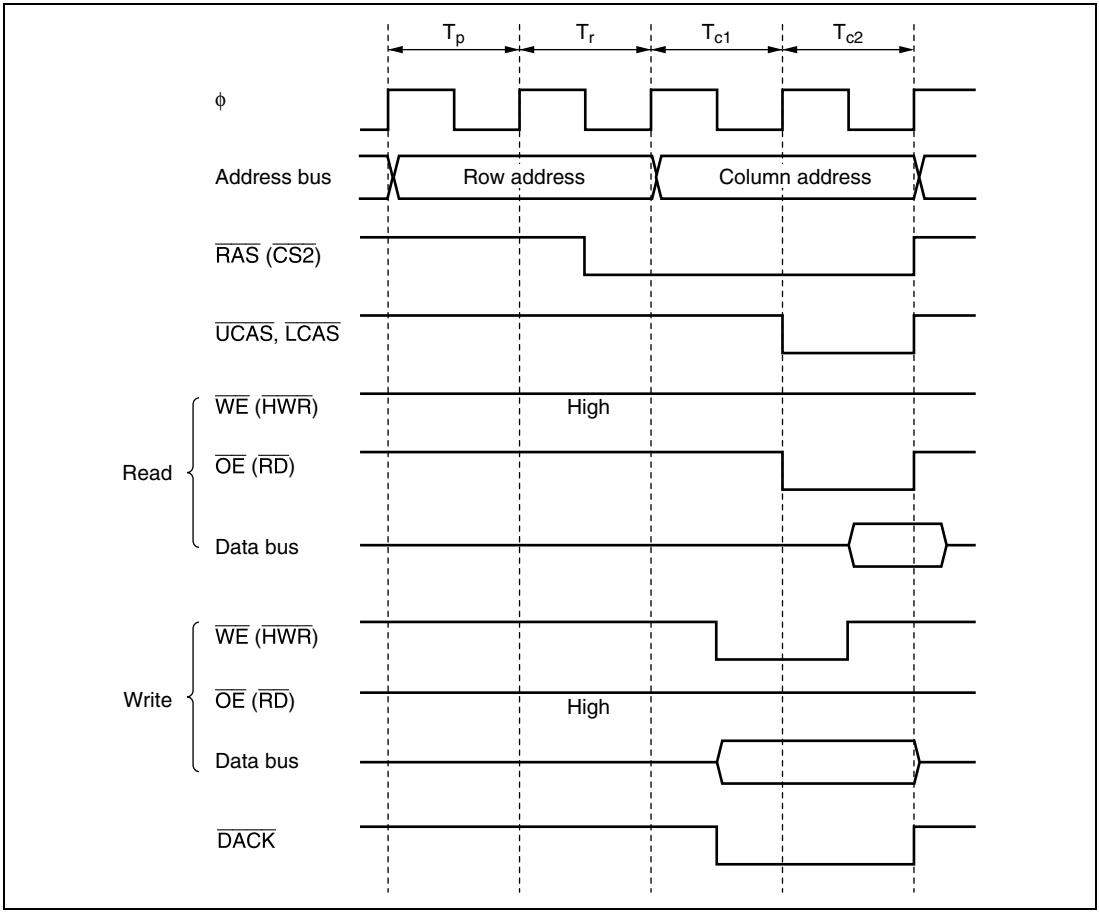


Figure 6.42 Example of $\overline{\text{DACK}}$ Output Timing when DDS = 1 (RAST = 0, CAST = 0)

When DDS = 0: When DRAM space is accessed in DMAC single address transfer mode, full access (normal access) is always performed. With the DRAM interface, the $\overline{\text{DACK}}$ output goes low from the T_r state.

In modes other than DMAC single address transfer mode, burst access can be used when accessing DRAM space.

Figure 6.43 shows the $\overline{\text{DACK}}$ output timing for the DRAM interface when DDS = 0.

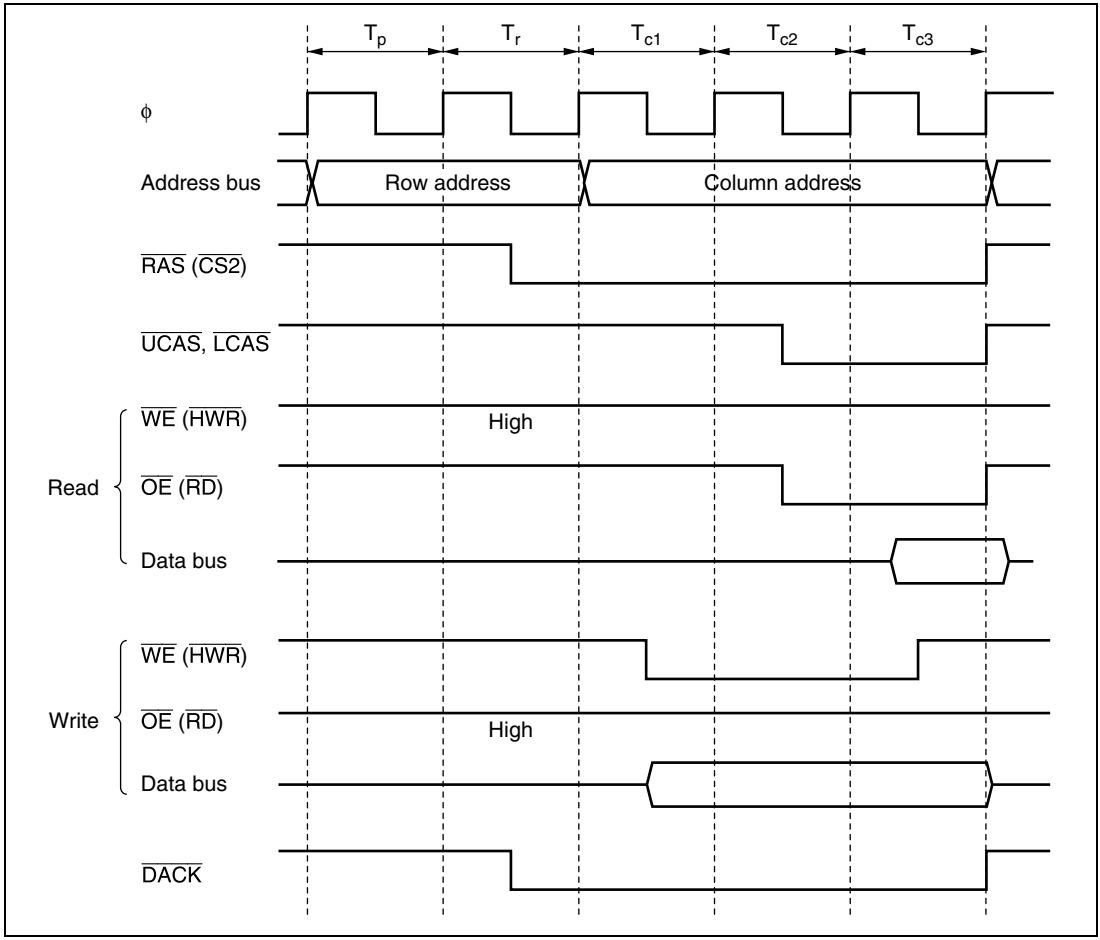


Figure 6.43 Example of $\overline{\text{DACK}}$ Output Timing when DDS = 0 (RAST = 0, CAST = 1)

6.7 Idle Cycle

6.7.1 Operation

When this LSI accesses external address space, it can insert an idle cycle (T_i) between bus cycles in the following three cases: (1) when read accesses in different areas occur consecutively or when an external access cycle occurs after a single address transfer, (2) when (1) occurs and a write cycle occurs immediately after a read cycle, and (3) when (1) and (2) occur and a read cycle occurs immediately after a write cycle. A condition for idle cycle insertion can be selected with the IDLE1 and IDLE0 bits in BCR. The number of idle cycles to be inserted can be set from one to four states by setting the IDLC1 and IDLC0 bits in BCR. By inserting an idle cycle it is possible, for example, to avoid data collisions between ROM, etc., with a long output floating time, and high-speed memory, I/O interfaces, and so on.

Consecutive Reads in Different Areas: If consecutive reads in different areas occur while the IDLE1 and IDLE0 bits in BCR are set to either B′01, B′10, or B′11, an idle cycle which is set by the IDLC1 and IDLC0 bits in BCR is inserted at the start of the second read cycle.

Figure 6.44 shows an example of the operation in this case. In this example, bus cycle A is a read cycle for ROM with a long output floating time, and bus cycle B is a read cycle for SRAM, each being located in a different area. In (a), an idle cycle is not inserted, and a collision occurs in bus cycle B between the read data from ROM and that from SRAM. In (b), an idle cycle is inserted, and a data collision is prevented.

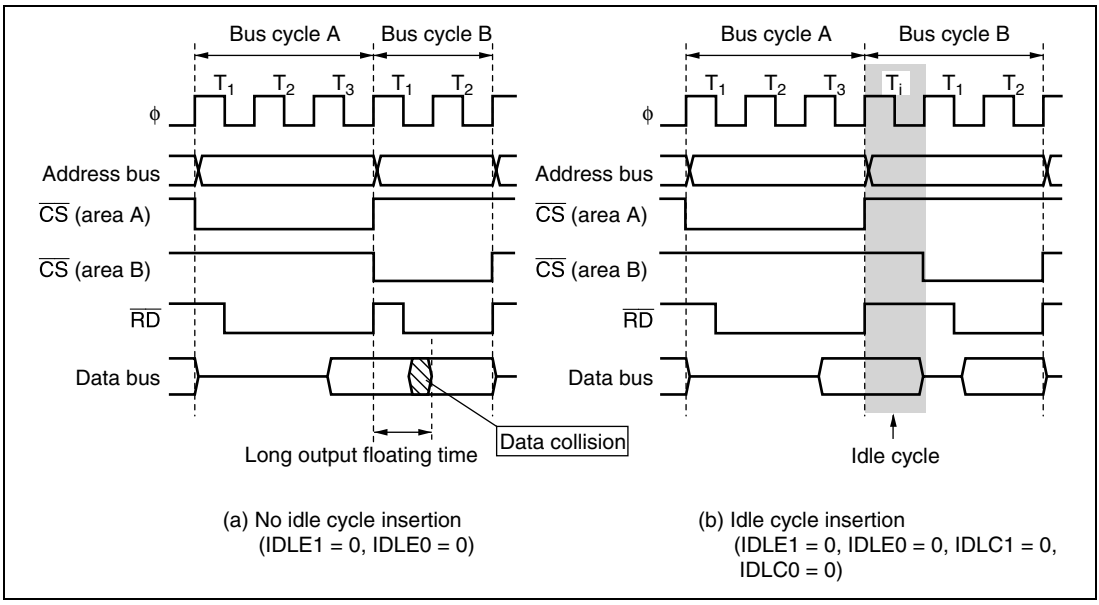


Figure 6.44 Example of Idle Cycle Operation (Consecutive Reads in Different Areas)

Write after Read: If an external write occurs after an external read while the IDLE1 and IDLE0 bits in BCR are set to either B′10 or B′11, an idle cycle which is set by the IDLC1 and IDLC0 bits in BCR is inserted at the start of the write cycle.

Figure 6.45 shows an example of the operation in this case. In this example, bus cycle A is a read cycle for ROM with a long output floating time, and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a collision occurs in bus cycle B between the read data from ROM and the CPU write data. In (b), an idle cycle is inserted, and a data collision is prevented.

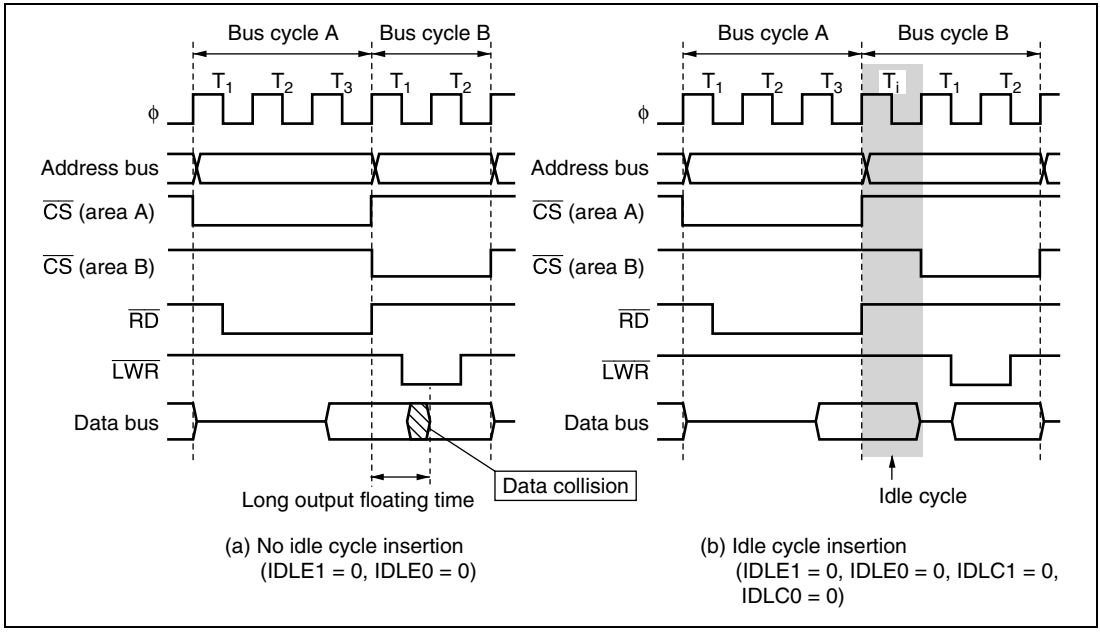


Figure 6.45 Example of Idle Cycle Operation (Write after Read)

Read after Write: If an external read occurs after an external write while the IDLE1 and IDLE0 bits in BCR are set to B′11, an idle cycle which is set by the IDLC1 and IDLC0 bits in BCR is inserted at the start of the read cycle.

Figure 6.46 shows an example of the operation in this case. In this example, bus cycle A is a CPU write cycle and bus cycle B is a read cycle from the SRAM. In (a), an idle cycle is not inserted, and a collision occurs in bus cycle B between the CPU write data and read data from the SRAM. In (b), an idle cycle is inserted, and a data collision is prevented.

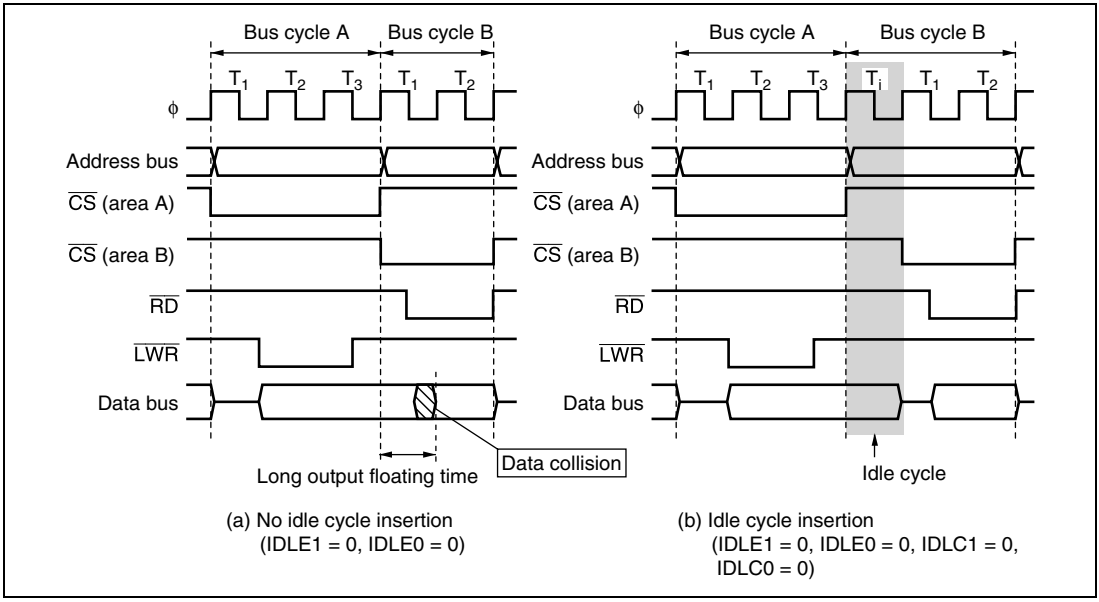


Figure 6.46 Example of Idle Cycle Operation (Read after Write)

Relationship between Chip Select ($\overline{CS}$) Signal and Read ($\overline{RD}$) Signal: Depending on the system's load conditions, the $\overline{RD}$ signal may lag behind the $\overline{CS}$ signal. An example is shown in figure 6.47. In this case, with the setting for no idle cycle insertion (a), there may be a period of overlap between the bus cycle A $\overline{RD}$ signal and the bus cycle B $\overline{CS}$ signal. Setting idle cycle insertion, as in (b), however, will prevent any overlap between the $\overline{RD}$ and $\overline{CS}$ signals.

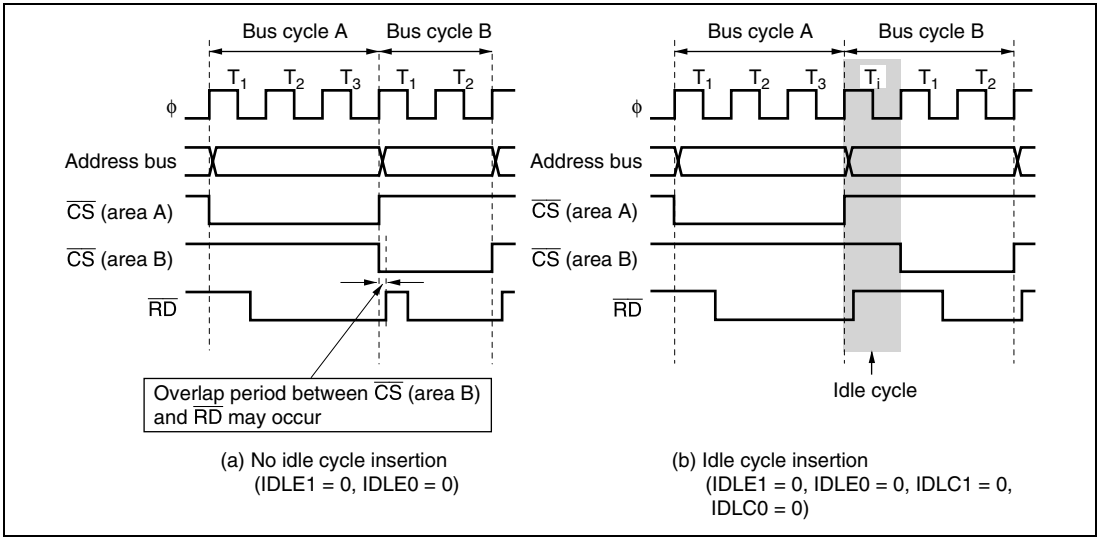


Figure 6.47 Relationship between Chip Select ($\overline{CS}$) and Read ($\overline{RD}$)

Idle Cycle in Case of DRAM Space Access after Normal Space Access: In a DRAM space access following a normal space access, the settings of bits IDLE1, IDLE0, IDLC1, and IDLC0 in BCR are valid. However, in the case of consecutive reads in different areas, for example, if the second read is a full access to DRAM space, idle cycles include T_p and T_i cycles. The timing when a four-state idle cycle is inserted in a full access to DRAM space is shown in figure 6.48.

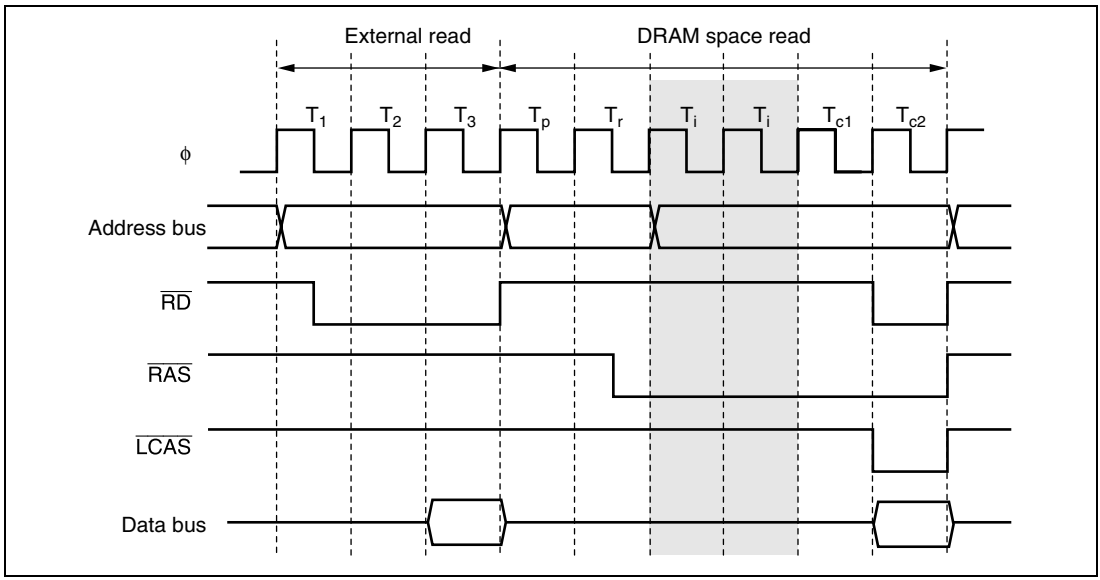


Figure 6.48 Example of DRAM Full Access after External Read (CAST = 0)

In burst access in RAS down mode, the settings of bits IDLE1, IDLE0, IDLC1, and IDLC0 are valid and an idle cycle is inserted. The timing in this case is illustrated in figure 6.49.

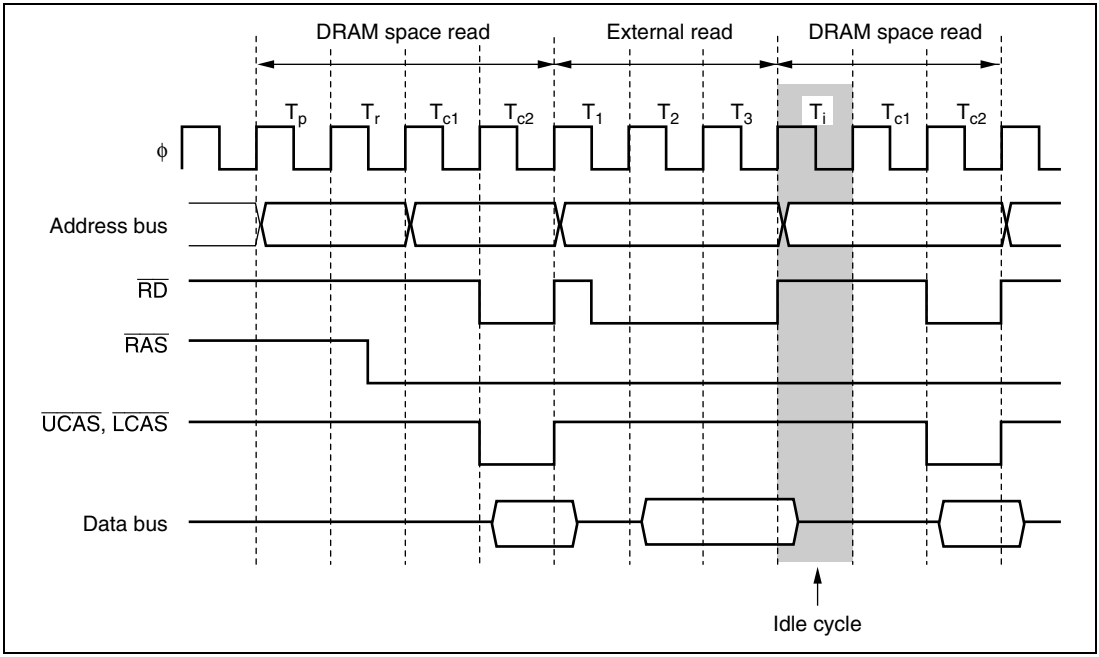


Figure 6.49 Example of Idle Cycle Operation in RAS Down Mode (Consecutive Reads in Different Areas) (IDLE1 = 0, IDLE0 = 1, IDLC1 = 0, IDLC0 = 1, RAST = 0, and CAST = 0)

Table 6.7 shows whether there is an idle cycle insertion or not in the case of mixed accesses to normal space and DRAM space.

Table 6.7 Idle Cycles in Mixed Accesses to Normal Space and DRAM Space

Previous Access	Next Access	IDLC1	IDLC0	IDLE1	IDLE0	Idle cycle
Normal/DRAM space read	Normal/DRAM space read (different area)	—	—	0	0	Disabled
		0	0	0	1	1 state inserted
				1	0	
					1	
			1	0	1	2 states inserted
				1	0	
					1	
		1	0	0	1	3 states inserted
				1	0	
					1	
			1	0	1	4 states inserted
				1	0	
					1	
Single address transfer	External space access	—	—	0	0	Disabled
		0	0	0	1	1 state inserted
				1	0	
					1	
			1	0	1	2 states inserted
				1	0	
					1	
		1	0	0	1	3 states inserted
				1	0	
					1	
			1	0	1	4 states inserted
				1	0	
					1	

Previous Access	Next Access	IDLC1	IDLC0	IDLE1	IDLE0	Idle cycle
Normal/DRAM space read	Normal/DRAM space write	—	—	0	0	Disabled
					1	
		0	0	1	0	1 state inserted
					1	
			1	1	0	2 states inserted
					1	
		1	0	1	0	3 states inserted
					1	
			1	1	0	4 states inserted
					1	
Normal/DRAM space write	Normal/DRAM space read	—	—	0	0	Disabled
					1	
				1	0	
		0	0	1	1	1 state inserted
			1	1	1	2 states inserted
		1	0	1	1	3 states inserted
			1	1	1	4 states inserted

6.7.2 Pin States in Idle Cycle

Table 6.8 shows the pin states in an idle cycle.

Table 6.8 Pin States in Idle Cycle

Pins	Pin State
A19 to A0	Contents of following bus cycle
D15 to D0	High impedance
$\overline{\text{CSn}}$ (n = 3 to 0)	High*
$\overline{\text{UCAS}}$, $\overline{\text{LCAS}}$	High
$\overline{\text{AS}}$	High
$\overline{\text{RD}}$	High
$\overline{\text{HWR}}$, $\overline{\text{LWR}}$	High
$\overline{\text{RAS}}$	High*
$\overline{\text{WE}}$	High
$\overline{\text{DACKn}}$ (n = 3 to 0)	High

Note: * Remains low in DRAM space RAS down mode.

6.8 Write Data Buffer Function

This LSI has a write data buffer function for the external data bus. Using the write data buffer function enables external writes and DMA single address mode transfers to be executed in parallel with internal accesses. The write data buffer function is made available by setting the WDBE bit to 1 in BCR.

Figure 6.50 shows an example of the timing when the write data buffer function is used. When this function is used, if an external address space write or DMA single address mode transfer continues for two states or longer, and there is an internal access next, an external write only is executed in the first state, but from the next state onward an internal access (on-chip memory or internal I/O register read/write) is executed in parallel with the external address space write rather than waiting until it ends.

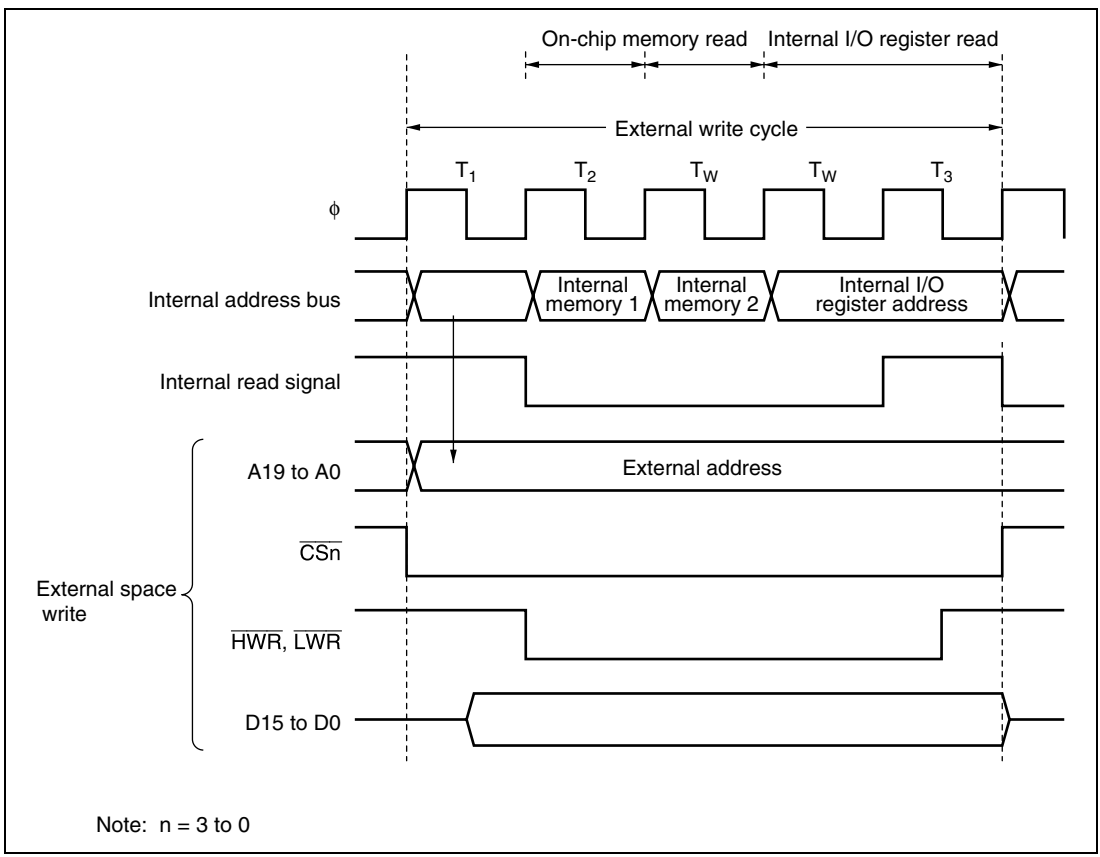


Figure 6.50 Example of Timing when Write Data Buffer Function is Used

6.9 Bus Arbitration

This LSI has a bus arbiter that arbitrates bus master operations (bus arbitration).

There are two bus masters—the CPU and DMAC—that perform read/write operations when they have possession of the bus. Each bus master requests the bus by means of a bus request signal. The bus arbiter determines priorities at the prescribed timing, and permits use of the bus by means of a bus request acknowledge signal. The selected bus master then takes possession of the bus and begins its operation.

6.9.1 Operation

The bus arbiter detects the bus masters' bus request signals, and if the bus is requested, sends a bus request acknowledge signal to the bus master. If there are bus requests from more than one bus master, the bus request acknowledge signal is sent to the one with the highest priority. When a bus master receives the bus request acknowledge signal, it takes possession of the bus until that signal is canceled.

The order of priority of the bus master is as follows:

(High) DMAC > CPU (Low)

6.9.2 Bus Transfer Timing

Even if a bus request is received from a bus master with a higher priority than that of the bus master that has acquired the bus and is currently operating, the bus is not necessarily transferred immediately. There are specific timings at which each bus master can relinquish the bus.

CPU: The CPU is the lowest-priority bus master, and if a bus request is received from the DMAC, the bus arbiter transfers the bus to the bus master that issued the request. The timing for transfer of the bus is as follows:

- The bus is transferred at a break between bus cycles. However, if a bus cycle is executed in discrete operations, as in the case of a longword-size access, the bus is not transferred between the component operations.
- With bit manipulation instructions such as BSET and BCLR, the sequence of operations is: data read (read), relevant bit manipulation operation (modify), write-back (write). The bus is not transferred during this read-modify-write cycle, which is executed as a series of bus cycles.
- If the CPU is in sleep mode, the bus is transferred immediately.

DMAC: The DMAC sends the bus arbiter a request for the bus when an activation request is generated.

In normal transfer mode or in cycle steal transfer mode, the DMAC releases the bus after a single transfer. In block transfer mode, it releases the bus after transfer of one block, and in burst mode, after completion of the transfer.

6.10 Bus Controller Operation in Reset

In a reset, this LSI, including the bus controller, enters the reset state immediately, and any executing bus cycle is aborted.

Section 7 DMA Controller (DMAC)

This LSI has an on-chip DMA controller (DMAC) which can carry out data transfer on up to 4 channels.

7.1 Features

- Number of channels: Four channels
- Address space: Physical address space (16-Mbyte external space)
- Transfer data length: Byte, word, or longword can be selected.
- Maximum number of transfers: 16,777,215/infinite (free-running)
- Address mode: Dual address mode or single address mode can be selected.

  Dual address mode

  Addresses of transfer source and transfer destination are accessed.

  Values set in the internal DMAC register are addresses to be accessed for transfer source and transfer destination.

  Single data transfer requires two bus cycles.

  Single address mode

  The peripheral device of transfer source or transfer destination is accessed by the $\overline{\text{DACK}}$ signal and another one is accessed by the address. Single data transfer requires one bus cycle.
- Transfer request: The DMAC transfer activation requests are as follows.

  External request

  Four $\overline{\text{DREQ}}$ pins. Low-level detection or falling-edge detection can be selected.

  External requests can be accepted on all channels.

  Auto request

  A transfer request is automatically generated from the internal DMAC.

  On-chip USB

  A transfer request can be accepted from the on-chip USB on all channels.
- Bus mode: Cycle steal mode or burst mode can be selected.
- Transfer mode: Normal mode or block transfer mode can be selected.

  Normal mode

  Single data transfer is performed for single transfer request.

  The number of transfers is specified as 24 bits (max. 16 Mbytes)

  Block transfer mode (only for external request)

  Single block (specified number) data transfer is performed for single transfer request.
- Interrupt request: An interrupt request can be sent to the CPU at the end of the specified number of transfers.
- Repeat area set function

This function enables data transfer of ring buffer, etc. efficiently because values in the upper bits of the transfer address register are fixed and address values in the specific range are repeated.

Repeat area can be set from one bit (two bytes) to 23 bits (8 Mbytes).

Repeat area can be set for both transfer source and transfer destination.

Interrupt request generation can be set by overflow determination of repeat area.

- Acceptance of a transfer request and the start of transfer processing can be notified to an external device via the $\overline{\text{DRAK}}$ pin.

Figure 7.1 shows a block diagram of the DMAC.

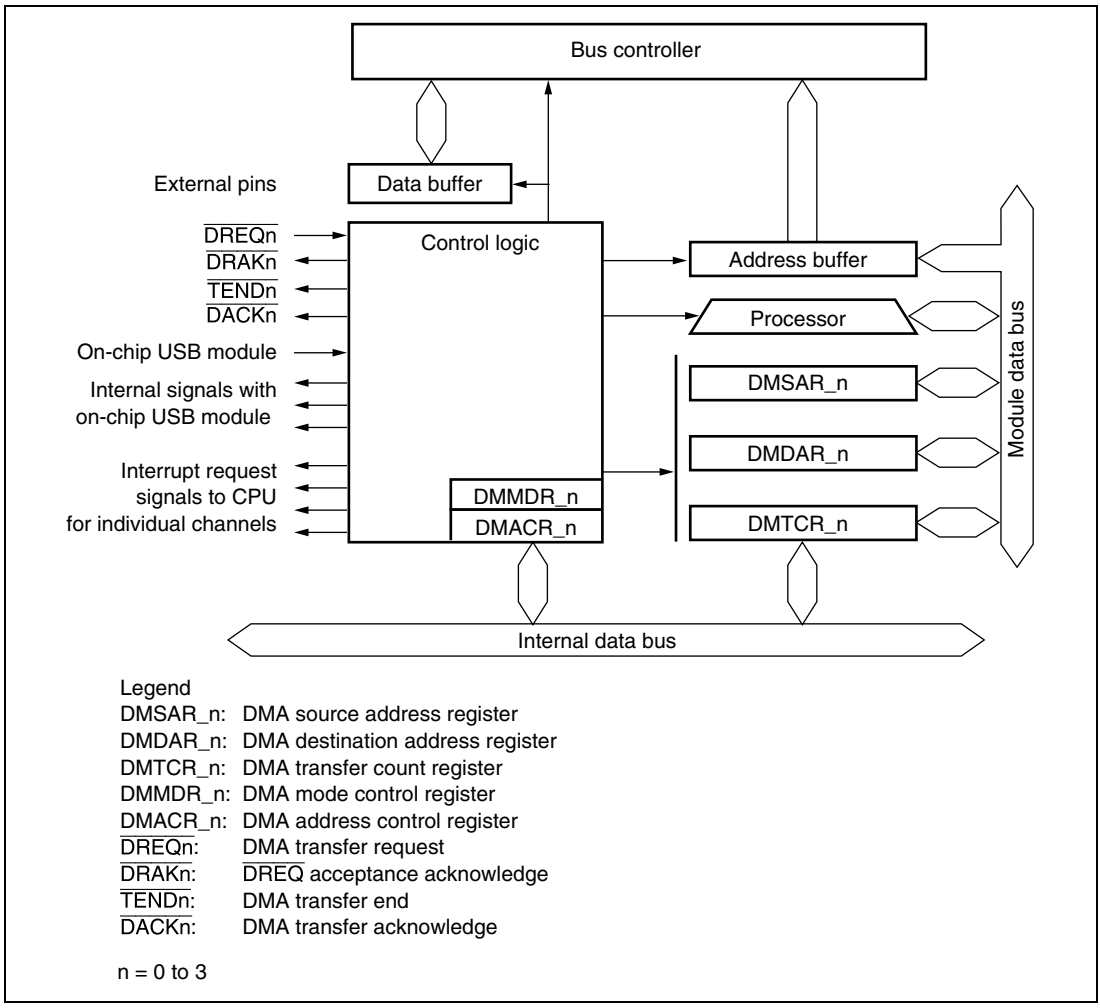


Figure 7.1 Block Diagram of DMAC

7.2 Input/Output Pins

Table 7.1 shows the pin configuration of the DMAC.

The corresponding port to the $\overline{\text{DACK}}$ pin automatically enters the output state by the setting of the single address transfer mode. When the $\overline{\text{DREQ}}$ pin is used, the corresponding port must not enter the output state. Whether the corresponding port to the $\overline{\text{TEND}}$/$\overline{\text{DRAK}}$ pin is used as $\overline{\text{TEND}}$/$\overline{\text{DRAK}}$ pin can be set by the register.

Table 7.1 Pin Configuration

Channel	Name	Abbreviation	I/O	Function
0	DMA request 0	$\overline{\text{DREQ0}}$	Input	Channel 0 external request
	DMA transfer acknowledge 0	$\overline{\text{DACK0}}$	Output	Channel 0 single address transfer acknowledge
	DMA transfer end 0	$\overline{\text{TEND0}}$	Output	Channel 0 transfer end
	$\overline{\text{DREQ0}}$ acceptance acknowledge	$\overline{\text{DRAK0}}$	Output	Notification to external device of channel 0 external request acceptance and start of transfer processing
1	DMA request 1	$\overline{\text{DREQ1}}$	Input	Channel 1 external request
	DMA transfer acknowledge 1	$\overline{\text{DACK1}}$	Output	Channel 1 single address transfer acknowledge
	DMA transfer end 1	$\overline{\text{TEND1}}$	Output	Channel 1 transfer end
	$\overline{\text{DREQ1}}$ acceptance acknowledge	$\overline{\text{DRAK1}}$	Output	Notification to external device of channel 1 external request acceptance and start of transfer processing
2	DMA request 2	$\overline{\text{DREQ2}}$	Input	Channel 2 external request
	DMA transfer acknowledge 2	$\overline{\text{DACK2}}$	Output	Channel 2 single address transfer acknowledge
	DMA transfer end 2	$\overline{\text{TEND2}}$	Output	Channel 2 transfer end
	$\overline{\text{DREQ2}}$ acceptance acknowledge	$\overline{\text{DRAK2}}$	Output	Notification to external device of channel 2 external request acceptance and start of transfer processing
3	DMA request 3	$\overline{\text{DREQ3}}$	Input	Channel 3 external request
	DMA transfer acknowledge 3	$\overline{\text{DACK3}}$	Output	Channel 3 single address transfer acknowledge
	DMA transfer end 3	$\overline{\text{TEND3}}$	Output	Channel 3 transfer end
	$\overline{\text{DREQ3}}$ acceptance acknowledge	$\overline{\text{DRAK3}}$	Output	Notification to external device of channel 3 external request acceptance and start of transfer processing

7.3 Register Descriptions

The DMAC has the following registers.

- DMA source address register_0 (DMSAR_0)
- DMA destination address register_0 (DMDAR_0)
- DMA transfer count register_0 (DMTCR_0)
- DMA mode control register_0 (DMMDR_0)
- DMA address control register_0 (DMACR_0)
- DMA source address register_1 (DMSAR_1)
- DMA destination address register_1 (DMDAR_1)
- DMA transfer count register_1 (DMTCR_1)
- DMA mode control register_1 (DMMDR_1)
- DMA address control register_1 (DMACR_1)
- DMA source address register_2 (DMSAR_2)
- DMA destination address register_2 (DMDAR_2)
- DMA transfer count register_2 (DMTCR_2)
- DMA mode control register_2 (DMMDR_2)
- DMA address control register_2 (DMACR_2)
- DMA source address register_3 (DMSAR_3)
- DMA destination address register_3 (DMDAR_3)
- DMA transfer count register_3 (DMTCR_3)
- DMA mode control register_3 (DMMDR_3)
- DMA address control register_3 (DMACR_3)
- USB transfer control register (USTCR)

7.3.1 DMA Source Address Register (DMSAR)

DMSAR is a 32-bit readable/writable register that specifies the transfer source address. An address update function is provided that updates the register contents to the next transfer source address each time transfer processing is performed. In single address mode, the DMSAR value is ignored when a device with $\overline{\text{DACK}}$ is specified as the transfer source.

The upper 8 bits of DMSAR are reserved; they are always read as 0 and cannot be modified. Only 0 should be written to these bits.

The DMSAR value is undefined at a reset or in hardware standby mode.

Do not write to DMSAR for a channel on which DMA transfer is in progress.

DMSAR can be read at all times by the CPU. When reading DMSAR for a channel on which DMA transfer processing is in progress, a longword-size read must be executed.

7.3.2 DMA Destination Address Register (DMDAR)

DMDAR is a 32-bit readable/writable register that specifies the transfer destination address. An address update function is provided that updates the register contents to the next transfer destination address each time transfer processing is performed. In single address mode, the DMDAR value is ignored when a device with $\overline{\text{DACK}}$ is specified as the transfer destination.

The upper 8 bits of DMDAR are reserved; they are always read as 0 and cannot be modified. Only 0 should be written to these bits.

The DMDAR value is undefined at a reset or in hardware standby mode.

Do not write to DMDAR for a channel on which DMA transfer is in progress.

DMDAR can be read at all times by the CPU. When reading DMDAR for a channel on which DMA transfer processing is in progress, a longword-size read must be executed.

7.3.3 DMA Transfer Count Register (DMTCR)

DMTCR specifies the number of transfers. The function differs according to the transfer mode (normal/block).

The DMTCR value is undefined at a reset or in hardware standby mode.

Do not write to DMTCR for a channel on which DMA transfer is in progress.

Normal Transfer Mode:

Bit	Bit Name	Initial Value	R/W	Description
31 to 24	—	All 0	—	Reserved These bits are always read as 0. The write value should always be 0.
23 to 0		Undefined	R/W	24-Bit Transfer Counter These bits specify the number of transfers (number of bytes, word, or longwords). Setting H'000001 specifies one transfer. Setting H'000000 means no specification for the number of transfers, and the transfer counter function is halted. In this case, there is no transfer end interrupt by the transfer counter. Setting H'FFFFFF specifies the maximum number of transfers, that is 16,777,215. During DMA transfer, this counter shows the remaining number of transfers. This counter can be read at all times. When reading DMTCR for a channel on which DMA transfer processing is in progress, a longword-size read must be executed.

Block Transfer Mode:

Bit	Bit Name	Initial Value	R/W	Description
31 to24	—	All 0	—	Reserved These bits are always read as 0. The write value should always be 0.
23 to 16		Undefined	R/W	Block Size These bits specify the block size (number of bytes, words, or longwords) for block transfer. Setting H'01 specifies one as the block, while setting H'00 specifies the maximum block size, that is 256. The register value always indicates the specified block size.
15 to 0		Undefined	R/W	16-Bit Transfer Counter These bits specify the number of block transfers (number of bytes, word, or longwords). Setting H'0001 specifies one block transfer. Setting H'0000 means no specification for the number of transfers, and the transfer counter function is halted. In this case, there is no transfer end interrupt by the transfer counter. Setting H'FFFF specifies the maximum number of block transfers, that is 65,535. During DMA transfer, this counter shows the remaining number of block transfers.

7.3.4 DMA Mode Control Register (DMMDR)

DMMDR specifies the operating mode and transfer type.

Bit	Bit Name	Initial Value	R/W	Description
15	DA	0	R/(W)*[1]	DMA Active Controls the DMA operation. When this bit is set to 1, this indicates that an DMA operation is in progress. When auto request mode is specified (by bits MDS1 and MDS0), transfer processing begins when this bit is set to 1. With external requests, transfer processing begins when a transfer request is issued after this bit has been set to 1. When this bit is cleared to 0 during an DMA operation, transfer is halted. If this bit is cleared to 0 during an DMA operation in block transfer mode, transfer processing is continued for the currently executing one-block transfer, and the bit is cleared on completion of the currently executing one-block transfer. If an external source that ends (aborts) transfer occurs, this bit is automatically cleared to 0 and transfer is terminated. Do not change the operating mode, transfer method, or other parameters while this bit is set to 1. 0: Data transfer disabled on corresponding channel [Clearing conditions] • When the specified number of transfers end • When operation is halted by a repeat area overflow interrupt • When 0 is written to DA while DA = 1 (In block transfer mode, write is effective after end of one-block transfer) • Reset, NMI interrupt, hardware standby mode 1: Data transfer enabled on corresponding channel and during an DMA operation.

Bit	Bit Name	Initial Value	R/W	Description
14	BEF	0	R/(W)*2	Block Transfer Error Flag Flag that indicates the occurrence of an error during block transfer. If an NMI interrupt is generated during block transfer, the DMAC immediately terminates the DMA operation and sets this bit to 1. The address registers indicate the next transfer addresses during block transfer, but the data for which transfer has been performed within the block size is lost. To clear this bit, 0 should be written after reading 1 from this bit. 0: No block transfer error [Clearing condition] Writing 0 to BEF after reading BEF = 1 1: Block transfer error and block transfer is abnormal. [Setting condition] NMI interrupt during block transfer
13	DRAKE	0	R/W	$\overline{\text{DRAK}}$ Pin Output Enable Enables output from the $\overline{\text{DREQ}}$ acknowledge/transfer processing start ($\overline{\text{DRAK}}$) pin. 0: $\overline{\text{DRAK}}$ pin output disabled 1: $\overline{\text{DRAK}}$ pin output enabled
12	TENDE	0	R/W	$\overline{\text{TEND}}$ Pin Output Enable Enables output from the DMA transfer end ($\overline{\text{TEND}}$) pin. 0: $\overline{\text{TEND}}$ pin output disabled 1: $\overline{\text{TEND}}$ pin output enabled
11	DREQS	0	R/W	$\overline{\text{DREQ}}$ Select Specifies low level sensing or falling edge sensing as the sampling method for the $\overline{\text{DREQ}}$ pin used in external request mode. 0: Low level sensing (Low level sensing is used for the first transfer after transfer is enabled.) 1: Falling edge sensing
10	AMS	0	R/W	Address Mode Select Selects single address mode or dual address mode. When single address mode is selected, the $\overline{\text{DACK}}$ pin is valid. 0: Dual address mode 1: Single address mode

Bit	Bit Name	Initial Value	R/W	Description
9	MDS1	0	R/W	Mode Select 1 and 0
8	MDS0	0	R/W	These bits specify the activation source, bus mode, and transfer mode.

	Activation Source	Bus Mode	Transfer Mode
00	Auto request	Cycle steal mode	Normal transfer mode
01	Auto request	Burst mode	Normal transfer mode
10	External request, on-chip USB	Cycle steal mode	Normal transfer mode
11	External request, on-chip USB	Cycle steal mode	Block transfer mode

Note: The transfer from the on-chip USB can be the requested by setting USTCR. See section 7.3.6, USB Transfer Control Register (USTCR).

7	DIE	0	R/W	DMA Interrupt Enable Enables or disables interrupt requests. When this bit is set to 1, an interrupt request is generated when the IRF bit in DMMDR is set to 1. The interrupt request is cleared by clearing this bit or the IRF bit in DMMDR to 0. 0: Interrupt request is not generated 1: Interrupt request is generated

Bit	Bit Name	Initial Value	R/W	Description
6	IRF	0	R/(W)*2	Interrupt Request Flag Flag indicating that an interrupt request has occurred and transfer has ended. To clear this bit, the DA bit in DMMDR is set to 1 or 0 is written after reading 1 from this bit. 0: No interrupt request [Clearing conditions] • Writing 1 to the DA bit in DMMDR • Writing 0 to IRF after reading IRF = 1 1: Interrupt request occurrence [Setting conditions] • Transfer end interrupt request generated by transfer counter • Source address repeat area overflow interrupt request • Destination address repeat area overflow interrupt request
5	TCEIE	0	R/W	Transfer Counter End Interrupt Enable Enables or disables transfer end interrupt requests by the transfer counter. When transfer ends according to the transfer counter while this bit is set to 1, the IRF bit in DMMDR is set to 1, indicating that an interrupt request has occurred. 0: Transfer end interrupt requests by transfer counter are disabled 1: Transfer end interrupt requests by transfer counter are enabled
4	SDIR	0	R/W	Single Address Direction Specifies the data transfer direction in single address mode. In dual address mode (AMS = 0), the specification by this bit is ignored. 0: Transfer direction: DMSAR → external device with DACK 1: Transfer direction: External device with DACK → DMDAR

Bit	Bit Name	Initial Value	R/W	Description
3	DTSIZE	0	R/W	Data Transmit Size Specifies the size of data to be transferred by combination of the LWSIZE bit. LWSIZE DTSIZE 0 0: Byte-size (8-bit) specification 0 1: Word-size (16-bit) specification 0 0: Longword-size (32-bit) specification 1 1: Reserved (setting prohibited)
2	—	0	R/W	Reserved This bit can be read from or written to. However, the write value should always be 0.
1	LWSIZE	0	R/W	Longword Data Transmit Size Specifies the size of data to be transferred by combination of the DTSIZE bit.
0	—	0	R/W	Reserved This bit can be read from or written to. However, the write value should always be 0.

Notes: 1. There is a period when the written value is not reflected immediately.
2. Only 0 can be written after reading 1, to clear the flag.

7.3.5 DMA Address Control Register (DMACR)

DMACR specifies address register incrementing/decrementing and use of the repeat area function.

Bit	Bit Name	Initial Value	R/W	Description
15	SAT1	0	R/W	Source Address Update Mode
14	SAT0	0	R/W	These bits specify incrementing/decrementing of the transfer source address (DMSAR). When an external device with DACK is designated as the transfer source in single address mode, the specification by these bits is ignored. 0X: Source address (DMSAR) is fixed 10: Source address is incremented (+1 in byte transfer, +2 in word transfer, or +4 in longword transfer) 11: Source address is decremented (–1 in byte transfer, –2 in word transfer, or –4 in longword transfer)
13	SARIE	0	R/W	Source Address Repeat Interrupt Enable When this bit is set to 1, in the event of source address repeat area overflow, the IRF bit in DMMDR is set to 1 and the DA bit in DMMDR cleared to 0, and transfer is terminated. If the DIE bit in DMMDR is 1 when the IRF bit in DMMDR is set to 1, an interrupt request is sent to the CPU. When used together with block transfer mode, a source address repeat interrupt is requested at the end of a block-size transfer. If the DA bit is set to 1 in DMMDR for the channel on which transfer is terminated by a source address repeat interrupt, transfer can be resumed from the state in which it ended. If a source address repeat area has not been designated, this bit is ignored. 0: Source address repeat interrupt is not requested 1: When source address repeat area overflow occurs, the IRF bit in DMMDR is set to 1 and an interrupt is requested

Bit	Bit Name	Initial Value	R/W	Description
12	SARA4	0	R/W	Source Address Repeat Area
11	SARA3	0	R/W	These bits specify the source address (DMSAR) repeat area. The repeat area function updates the specified lower address bits, leaving the remaining upper address bits always the same. A repeat area size of 2 bytes to 8 Mbytes can be specified. The setting interval is a power-of-two number of bytes. When repeat area overflow results from incrementing or decrementing an address, the lower address is the start address of the repeat area in the case of address incrementing, or the last address of the repeat area in the case of address decrementing. If the SARIE bit is set to 1, an interrupt can be requested when repeat area overflow occurs. 00000: Not designated source address (DMSAR) as repeat area 00001: Lower 1 bit (2-byte area) in DMSAR designated as repeat area 00010: Lower 2 bits (4-byte area) in DMSAR designated as repeat area 00011: Lower 3 bits (8-byte area) in DMSAR designated as repeat area 00100: Lower 4 bits (16-byte area) in DMSAR designated as repeat area : : 10011: Lower 19 bits (512-kbyte area) in DMSAR designated as repeat area 10100: Lower 20 bits (1-Mbyte area) in DMSAR designated as repeat area 10101: Lower 21 bits (2-Mbyte area) in DMSAR designated as repeat area 10110: Lower 22 bits (4-Mbyte area) in DMSAR designated as repeat area 10111: Lower 23 bits (8-Mbyte area) in DMSAR designated as repeat area
10	SARA2	0	R/W	
9	SARA1	0	R/W	
8	SARA0	0	R/W	

Bit	Bit Name	Initial Value	R/W	Description
7	DAT1	0	R/W	Destination Address Update Mode
6	DAT0	0	R/W	These bits specify incrementing/decrementing of the transfer destination address (DMDAR). When an external device with DACK is designated as the transfer destination in single address mode, the specification by these bits is ignored. 0X: Destination address (DMDAR) is fixed 10: Destination address is incremented (+1 in byte transfer, +2 in word transfer, or +4 in longword transfer) 11: Destination address is decremented (–1 in byte transfer, –2 in word transfer, or –4 in longword transfer)
5	DARIE	0	R/W	Destination Address Repeat Interrupt Enable When this bit is set to 1, in the event of destination address repeat area overflow the IRF bit is set to 1 and the DA bit cleared to 0 in DMMDR, and transfer is terminated. If the DIE bit in DMMDR is 1 when the IRF bit in DMMDR is set to 1, an interrupt request is sent to the CPU. When used together with block transfer mode, a destination address repeat interrupt is requested at the end of a block-size transfer. If the DA bit is set to 1 in DMMDR for the channel on which transfer is terminated by a destination address repeat interrupt, transfer can be resumed from the state in which it ended. If a destination address repeat area has not been designated, this bit is ignored. 0: Destination address repeat interrupt is not requested 1: When destination address repeat area overflow occurs, the IRF bit in DMMDR is set to 1 and an interrupt is requested

Bit	Bit Name	Initial Value	R/W	Description
4	DARA4	0	R/W	Destination Address Repeat Area
3	DARA3	0	R/W	These bits specify the destination address (DMDAR) repeat area. The repeat area function updates the specified lower address bits, leaving the remaining upper address bits always the same.
2	DARA2	0	R/W	
1	DARA1	0	R/W	
0	DARA0	0	R/W	
				A repeat area size of 2 bytes to 8 Mbytes can be specified. The setting interval is a power-of-two number of bytes.
				When repeat area overflow results from incrementing or decrementing an address, the lower address is the start address of the repeat area in the case of address incrementing, or the last address of the repeat area in the case of address decrementing.
				If the DARIE bit is set to 1, an interrupt can be requested when repeat area overflow occurs.
				00000: Not designated destination address (DMDAR) as repeat area
				00001: Lower 1 bit (2-byte area) in DMDAR designated as repeat area
				00010: Lower 2 bits (4-byte area) in DMDAR designated as repeat area
				00011: Lower 3 bits (8-byte area) designated as repeat area
				00100: Lower 4 bits (16-byte area) in DMDAR designated as repeat area
				: :
				10011: Lower 19 bits (512-kbyte area) in DMDAR designated as repeat area
				10100: Lower 20 bits (1-Mbyte area) in DMDAR designated as repeat area
				10101: Lower 21 bits (2-Mbyte area) in DMDAR designated as repeat area
				10110: Lower 22 bits (4-Mbyte area) in DMDAR designated as repeat area
				10111: Lower 23 bits (8-Mbyte area) in DMDAR designated as repeat area

7.3.6 USB Transfer Control Register (USTCR)

USTCR specifies the transfer source from the on-chip USB, etc.

Bit	Bit Name	Initial Value	R/W	Description
15	EP1DMAE	0	R/W	Endpoint 1 (EP1) DMA Enable Enables a transfer source from the on-chip USB (transfer direction: reading from the on-chip USB (EP1)). When this bit is set to 1, a transfer request from the USB is selected as a transfer source. In block transfer mode, the on-chip USB request must not be set as an activation source. While the DA bit in DMMDR is set to 1, the EP1DMAE value must not be changed. 0: Transfer request from on-chip USB (EP1) not accepted. 1: Transfer request from on-chip USB (EP1) accepted. The $\overline{\text{DREQ}}$ pin on the corresponding channel is not available.
14	URCHS1	0	R/W	USB Read Channel Select
13	URCHS0	0	R/W	When the DMA transfer is performed by a transfer request from the USB (EP1), these bits select the DMAC channel to be used. When the channel which accepts a request is selected and the EP1DMAE bit is set to 1, the corresponding channel accepts a USB request rather than an external request. In this case, the transfer direction is reading from the USB (EP1). Therefore, the source address must be specified as the FIFO in the USB (EP1). While the DA bit in DMMDR is set to 1, these bits must not be changed. A transfer request from the endpoint 1 (EP1) (reading from the on-chip USB) or a transfer request from the endpoint 2 (EP2) (writing to the on-chip USB) must not be set to the same channel. 00: Channel 0 can accept the EP1 transfer request. 01: Channel 1 can accept the EP1 transfer request. 10: Channel 2 can accept the EP1 transfer request. 11: Channel 3 can accept the EP1 transfer request.

Bit	Bit Name	Initial Value	R/W	Description
12	—	0	R/W	Reserved This bit can be read from or written to. However, the write value should always be 0.
11	EP2DMAE	0	R/W	Endpoint 2 (EP2) DMA Enable Enables a transfer source from the on-chip USB (transfer direction: writing to the on-chip USB). When this bit is set to 1, a transfer request from the USB is selected as a transfer source. In block transfer mode, the on-chip USB request must not be set as an activation source. While the DA bit in DMMDR is set to 1, the EP2DMAE value must not be changed. 0: Transfer request from on-chip USB (EP2) not accepted. 1: Transfer request from on-chip USB (EP2) accepted. The $\overline{\text{DREQ}}$ pin on the corresponding channel is not available.
10	UWCHS1	0	R/W	USB Write Channel Select
9	UWCHS0	0	R/W	When the DMA transfer is performed by a transfer request from the USB (EP2), these bits select the DMAC channel to be used. When the channel which accepts a request is selected and the EP2DMAE bit is set to 1, the corresponding channel accepts a USB request rather than an external request. In this case, the transfer direction is writing to the on-chip USB (EP2). Therefore, the destination address must be specified as the FIFO in the on-chip USB (EP2). While the DA bit in DMMDR is set to 1, these bits must not be changed. A transfer request from the endpoint 1 (EP1) (reading from the on-chip USB) or a transfer request from the endpoint 2 (EP2) (writing to the on-chip USB) must not be set to the same channel. 00: Channel 0 can accept the EP2 transfer request. 01: Channel 1 can accept the EP2 transfer request. 10: Channel 2 can accept the EP2 transfer request. 11: Channel 3 can accept the EP2 transfer request.

Bit	Bit Name	Initial Value	R/W	Description
8 to 0	—	All 0	R/W	Reserved These bits can be read from or written to. However, the write value should always be 0.

7.4 Operation

All DMAC functions on four channels are common. Each mode can be set independently for each channel. The DMAC functions can be used by combining each function.

7.4.1 Transfer Modes

The transfer modes of the DMAC are summarized in table 7.2.

Table 7.2 DMAC Transfer Modes

Address Mode	Transfer Mode	Bus Mode	Transfer Origin	Number of Transfers	Address Registers	
					Source	Destination
Dual address mode	Normal transfer mode	Burst/cycle steal mode	Auto request	1 to 16,777,215 or no specification	DMSAR	DMDAR
		Cycle steal mode	External request			
			On-chip USB			
	Block transfer mode	Burst transfer of specified block size for a single transfer request Block size: 1 to 256 bytes, words, or longwords	External request	1 to 65,535 or no specification		
Single address mode	• Direct data transfer to/from external device using $\overline{\text{DACK}}$ pin instead of source or destination address register • Above transfer mode can be specified in addition to address register setting • One transfer possible in one bus cycle In single address mode, a transfer request from the on-chip USB is not available. (Transfer mode variations are the same as in dual address mode except for a transfer request from the on-chip USB.)				DMSAR/ $\overline{\text{DACK}}$	$\overline{\text{DACK}}$/ DMDAR

The transfer mode can be set independently for each channel. In normal transfer mode, a one-byte, one-word, or one-longword transfer is executed in response to one transfer request. With auto requests, burst or cycle steal transfer mode can be set. In normal or burst transfer mode, continuous, high-speed transfer can be performed until the specified number of transfers have been executed or the transfer enable bit is cleared to 0. In block transfer mode, a transfer of the specified block size is executed in response to one transfer request. The block size can be from 1 to 256 bytes, words, or longwords. Within a block, transfer can be performed at the same high speed as in burst transfer mode. When the “no specification” setting (DMTCR = H'000000) is made for the number of transfers, the transfer counter is halted and there is no limit on the number of transfers, allowing transfer to be performed endlessly.

Incrementing or decrementing the memory address by 1, 2, or 4, or leaving the address unchanged, can be specified independently for each address register. In all transfer modes, it is possible to set a repeat area comprising a power-of-two number of bytes.

7.4.2 Address Modes (Dual Address Mode/Single Address Mode)

Dual Address Mode: In dual address mode, both the transfer source and transfer destination are specified by registers in the DMAC, and one transfer is executed in two bus cycles.

The transfer source address is set in the source address register (DMSAR), and the transfer destination address is set in the destination address register (DMDAR).

In a transfer operation, the value in external memory specified by the transfer source address is read in the first bus cycle, and is written to the external memory specified by the transfer destination address in the next bus cycle.

These consecutive read and write cycles are indivisible: another bus cycle (external access by an internal bus master or refresh cycle) does not occur between these two cycles.

$\overline{\text{TEND}}$ pin output can be enabled or disabled by means of the TENDE bit in DMMDR. $\overline{\text{TEND}}$ is output for two consecutive bus cycles. The $\overline{\text{DACK}}$ signal is not output.

Figure 7.2 shows an example of the timing in dual address mode.

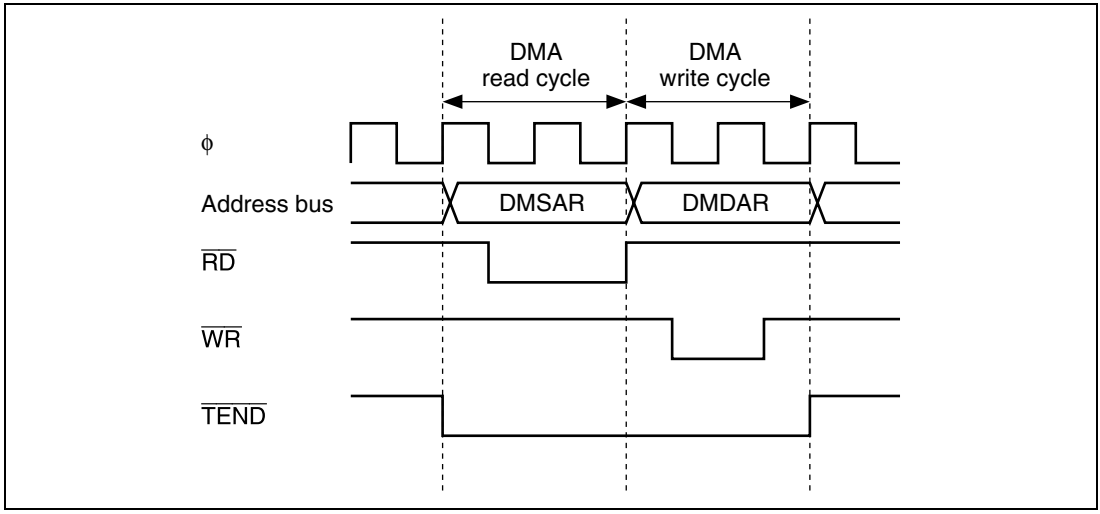


Figure 7.2 Example of Timing in Dual Address Mode

Single Address Mode: In single address mode, the $\overline{\text{DACK}}$ signal is used instead of the source or destination address register to transfer data directly between an external device and external memory. In this mode, the DMAC accesses the transfer source or transfer destination external device by outputting the external I/O strobe signal ($\overline{\text{DACK}}$), and at the same time accesses the other external device in the transfer by outputting an address. In this way, DMA transfer can be executed in one bus cycle. In the example of transfer between external memory and an external device with DACK shown in figure 7.3, data is output to the data bus by the external device and written to external memory in the same bus cycle.

The transfer direction, that is whether the external device with DACK is the transfer source or transfer destination, can be specified with the SDIR bit in DMMDR. Transfer is performed from the external memory (DMSAR) to the external device with DACK when SDIR = 0, and from the external device with DACK to the external memory (DMDAR) when SDIR = 1.

The setting in the source or destination address register not used in the transfer is ignored.

The $\overline{\text{DACK}}$ pin becomes valid automatically when single address mode is selected.

The $\overline{\text{DACK}}$ pin is active-low. $\overline{\text{TEND}}$ pin output can be enabled or disabled by means of the TENDE bit in DMMDR. $\overline{\text{TEND}}$ is output for one bus cycle.

Figure 7.3 shows the data flow in single address mode, and figure 7.4 shows an example of the timing.

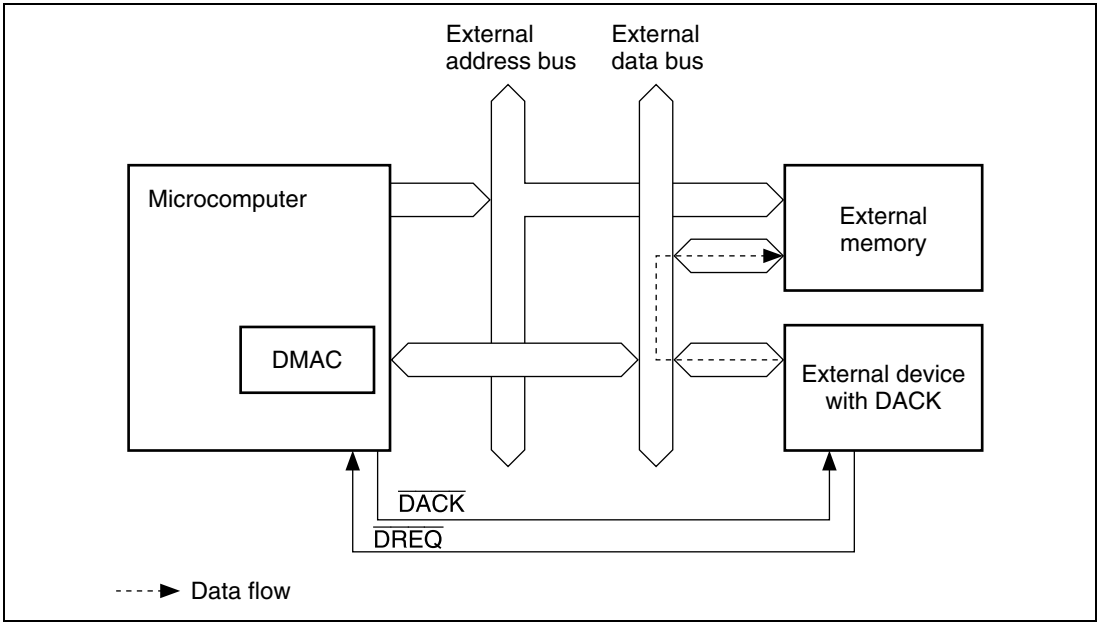


Figure 7.3 Data Flow in Single Address Mode

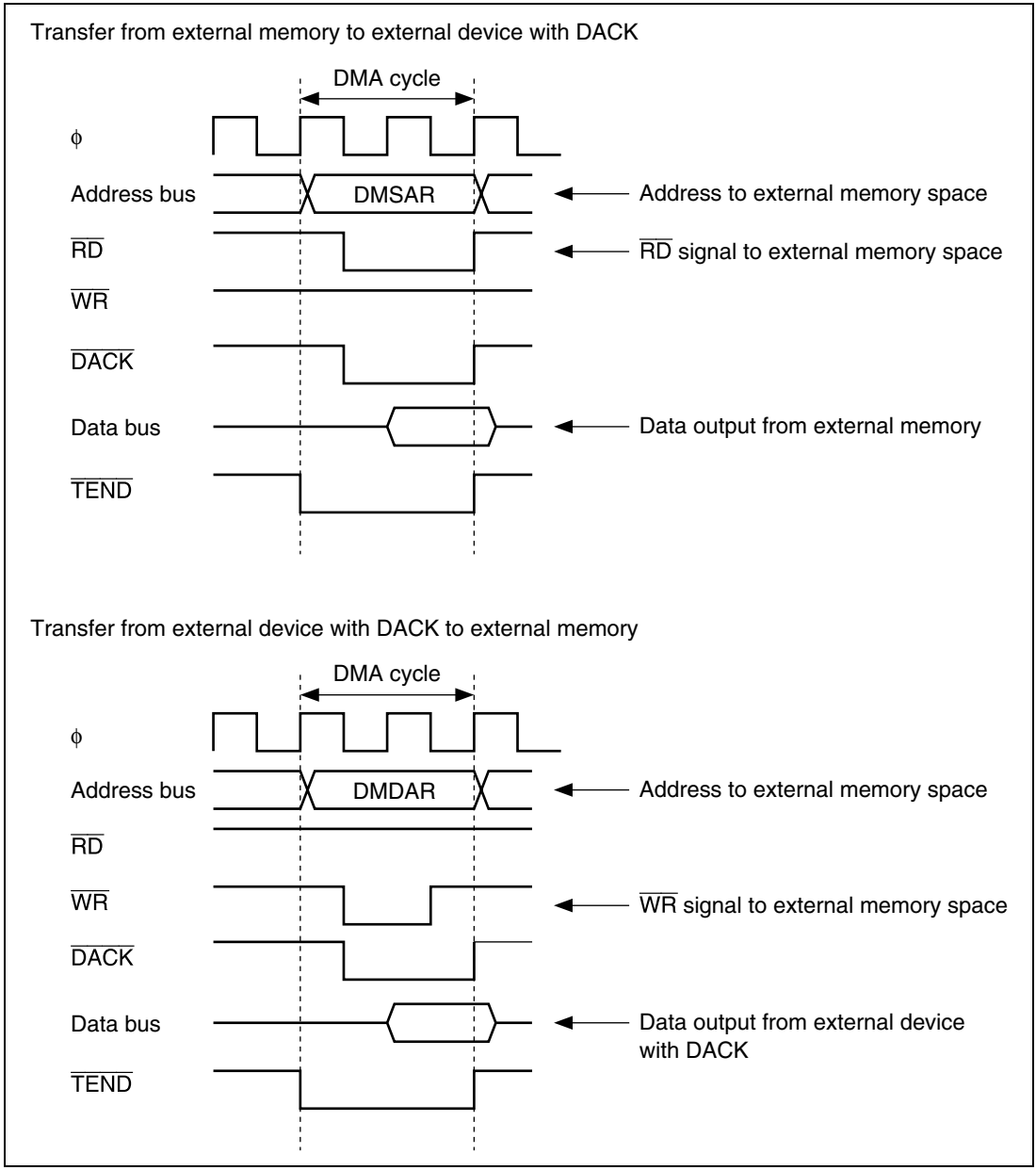


Figure 7.4 Example of Timing in Single Address Mode

7.4.3 DMA Transfer Requests (Auto Request Mode/External Request Mode/USB Transfer Request)

Auto Request Mode: In auto request mode, transfer request signals are automatically generated within the DMAC in cases where a transfer request signal is not issued from outside, such as in transfer between two memories, or between a peripheral module that is not capable of generating transfer requests and memory. In auto request mode, transfer is started when the DA bit is set to 1 in DMMDR.

In auto request mode, either cycle steal mode or burst mode can be selected as the bus mode. Block transfer mode cannot be used.

External Request Mode: In external request mode, transfer is started by a transfer request signal ($\overline{\text{DREQ}}$) from a device external to this LSI. DMA transfer is started when $\overline{\text{DREQ}}$ is input while DMA transfer is enabled (DA = 1).

The transfer request source need not be the data transfer source or data transfer destination.

The transfer request signal is accepted via the $\overline{\text{DREQ}}$ pin. Either falling edge sensing or low level sensing can be selected for the $\overline{\text{DREQ}}$ pin by means of the DREQS bit in DMMDR (low level sensing when DREQS = 0, falling edge sensing when DREQS = 1).

Setting the DRAKE bit to 1 in DMMDR enables a signal confirming transfer request acceptance to be output from the $\overline{\text{DRAK}}$ pin. The $\overline{\text{DRAK}}$ signal is output when acceptance and transfer processing has been started in response to a single external request. The $\overline{\text{DRAK}}$ signal enables the external device to determine the timing of $\overline{\text{DREQ}}$ signal negation, and makes it possible to provide handshaking between the transfer request source and the DMAC.

In external request mode, block transfer mode can be used instead of burst mode. Block transfer mode allows continuous execution (burst operation) of the specified number of transfers (the block size) in response to a single transfer request. In block transfer mode, the $\overline{\text{DRAK}}$ signal is output only once for a one-block transfer, since the transfer request via the $\overline{\text{DREQ}}$ pin is for a block unit.

USB Request Mode: In USB request mode, DMA transfer can be executed by a transfer request from the on-chip USB. When a transfer request from the USB can be accepted and DMA transfer is enabled (DA = 1), DMA transfer is started after a transfer request from the USB is input.

When a transfer request for the endpoint 1 is accepted, the DMAC transfers the endpoint 1 data. When a transfer request for the endpoint 2 is accepted, the DMAC transfers data to the endpoint 2.

When a transfer request from the USB is used as a transfer source, single address mode, block transfer mode, and normal/burst transfer mode cannot be used.

7.4.4 Bus Modes (Cycle Steal Mode/Burst Mode)

There are two bus modes: cycle steal mode and burst mode. When the activation source is an auto request, either cycle steal mode or burst mode can be selected. When the activation source is an external request, cycle steal mode is used.

Cycle Steal Mode: In cycle steal mode, the DMAC releases the bus at the end of each transfer of a transfer unit (byte, word, or block). If there is a subsequent transfer request, the DMAC takes back the bus, performs another transfer-unit transfer, and then releases the bus again. This procedure is repeated until the transfer end condition is satisfied.

If a transfer request occurs in another channel during DMA transfer, the bus is temporarily released, then transfer is performed on the channel for which the transfer request was issued. If there is no external space bus request from another bus master, a one-cycle bus release interval is inserted. For details on the operation when there are requests for a number of channels, see section 7.4.8, Channel Priority.

Figure 7.5 shows an example of the timing in cycle steal mode.

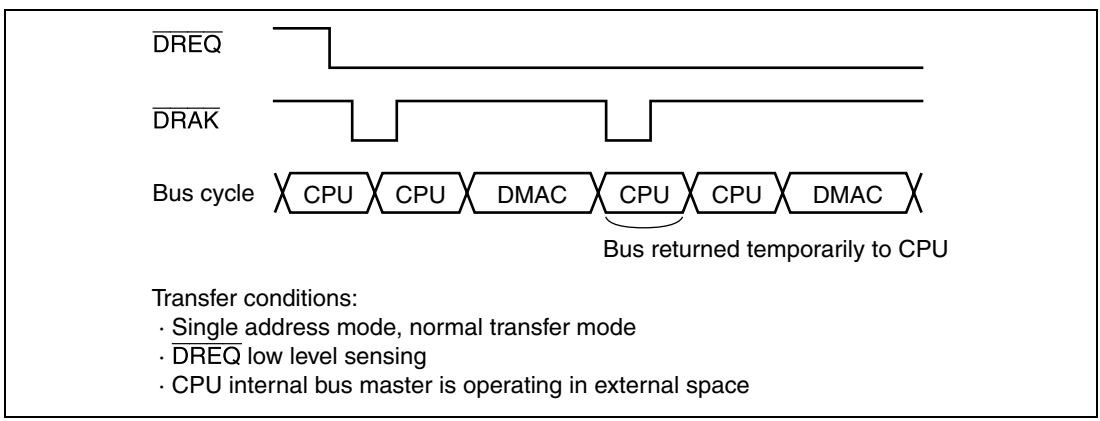


Figure 7.5 Example of Timing in Cycle Steal Mode

Burst Mode: In burst mode, once the DMAC acquires the bus it continues transferring data, without releasing the bus, until the transfer end condition is satisfied. There is no burst mode in external request mode. In burst mode, once transfer is started it is not interrupted even if there is a transfer request from another channel with higher priority. When the burst mode channel finishes its transfer, it releases the bus in the next cycle in the same way as in cycle steal mode.

When the DA bit is cleared to 0 in DMMDR, DMA transfer is halted. However, DMA transfer is executed for all transfer requests generated within the DMAC up until the DA bit was cleared to 0. If a repeat area overflow interrupt is generated, the DA bit is cleared to 0 and transfer is terminated.

Figure 7.6 shows an example of the timing in burst mode.

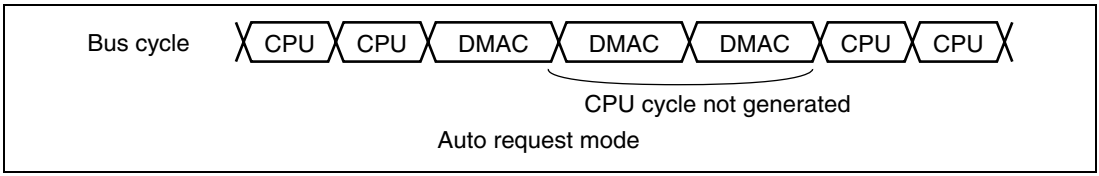


Figure 7.6 Examples of Timing in Burst Mode

7.4.5 Transfer Modes (Normal Transfer Mode/Block Transfer Mode)

There are two transfer modes: normal transfer mode and block transfer mode. When the activation source is an external request, either normal transfer mode or block transfer mode can be selected. When the activation source is an auto request, normal transfer mode is used.

Normal Transfer Mode: In normal transfer mode, transfer of one transfer unit is processed in response to one transfer request. DMTCR functions as a 24-bit transfer counter.

The $\overline{\text{TEND}}$ signal is output only for the last DMA transfer. The $\overline{\text{DRAK}}$ signal is output each time a transfer request is accepted and transfer processing is started. Figure 7.7 shows examples of DMA transfer timing in normal transfer mode.

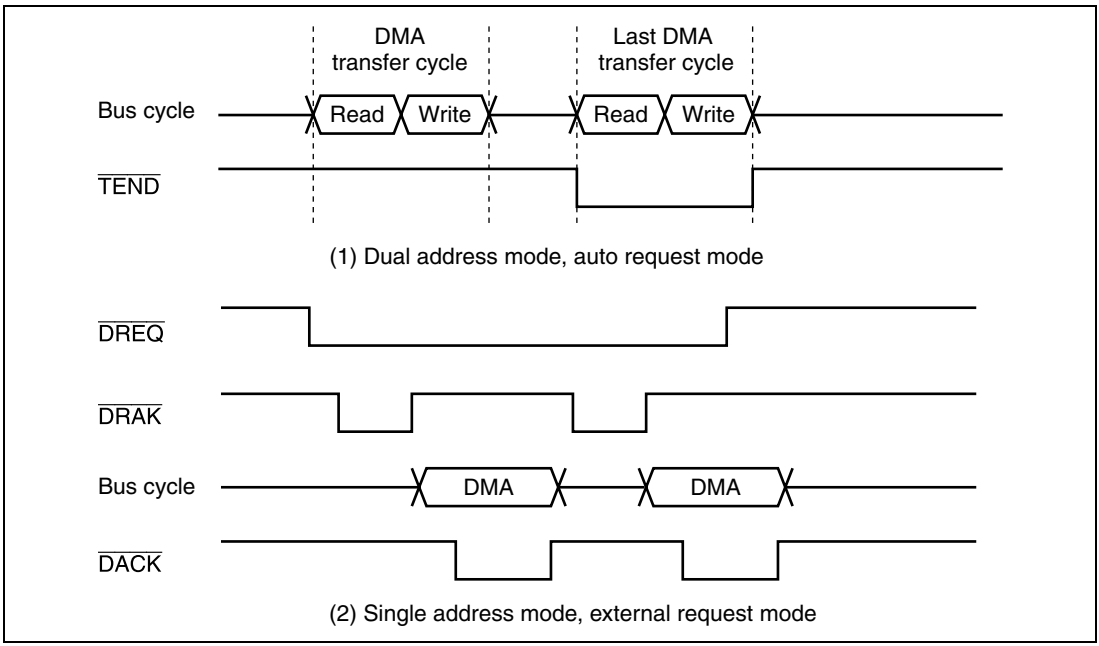


Figure 7.7 Examples of Timing in Normal Transfer Mode

Block Transfer Mode: In block transfer mode, the number of bytes, words, or longwords specified by the block size is transferred in response to one transfer request. The upper 8 bits of DMTCR specify the block size, and the lower 16 bits function as a 16-bit transfer counter. A block size of 1 to 256 can be specified.

During transfer of a block, transfer requests for other higher-priority channels are held pending. When transfer of one block is completed, the bus is released in the next cycle.

Address register values are updated in the same way as in normal mode. There is no function for restoring the initial address register values after each block transfer.

The $\overline{\text{TEND}}$ signal is output for each block transfer in the DMA transfer cycle in which the block ends. The $\overline{\text{DRAK}}$ signal is output once for one transfer request (for transfer of one block).

Caution is required when setting the repeat area overflow interrupt of the repeat area function in block transfer mode. See section 7.4.6, Repeat Area Function, for details. Block transfer is aborted if an NMI interrupt is generated. See section 7.4.12, Ending DMA Transfer, for details.

Figure 7.8 shows an example of DMA transfer timing in block transfer mode.

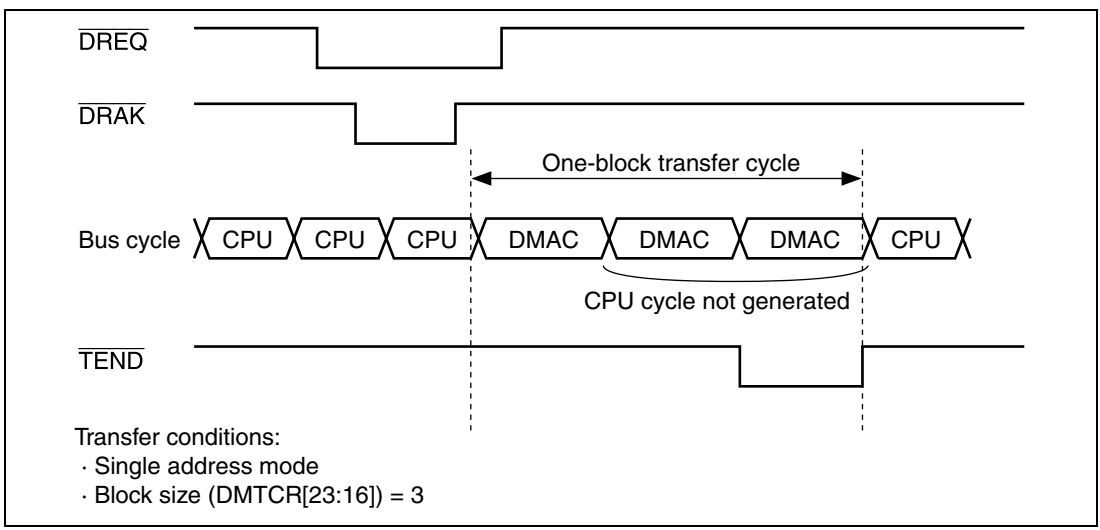


Figure 7.8 Example of Timing in Block Transfer Mode

7.4.6 Repeat Area Function

The DMAC has a function for designating a repeat area for source addresses and/or destination addresses. When a repeat area is designated, the address register values repeat within the range specified as the repeat area. Normally, when a ring buffer is involved in a transfer, an operation is required to restore the address register value to the buffer start address each time the address register value is the last address in the buffer (i.e. when ring buffer address overflow occurs), but

if the repeat area function is used, the operation that restores the address register value to the buffer start address is performed automatically within the DMAC.

The repeat area function can be set independently for the source address register and the destination address register. The source address repeat area is specified by bits SARA4 to SARA0 in DMACR, and the destination address repeat area by bits DARA4 to DARA0 in DMACR. The size of each repeat area can be specified independently.

When the address register value is the last address in the repeat area and repeat area overflow occurs, DMA transfer can be temporarily halted and an interrupt request sent to the CPU. If the SARIE bit in DMACR is set to 1, when the source address register overflows the repeat area, the IRF bit is set to 1 and the DA bit cleared to 0 in DMMDR, and transfer is terminated. If DIE = 1 in DMMDR, an interrupt is requested. If the DARIE bit in DMACR is set to 1, the above applies to the destination address register.

If the DA bit in DMMDR is set to 1 during interrupt generation, transfer is resumed. Figure 7.9 illustrates the operation of the repeat area function.

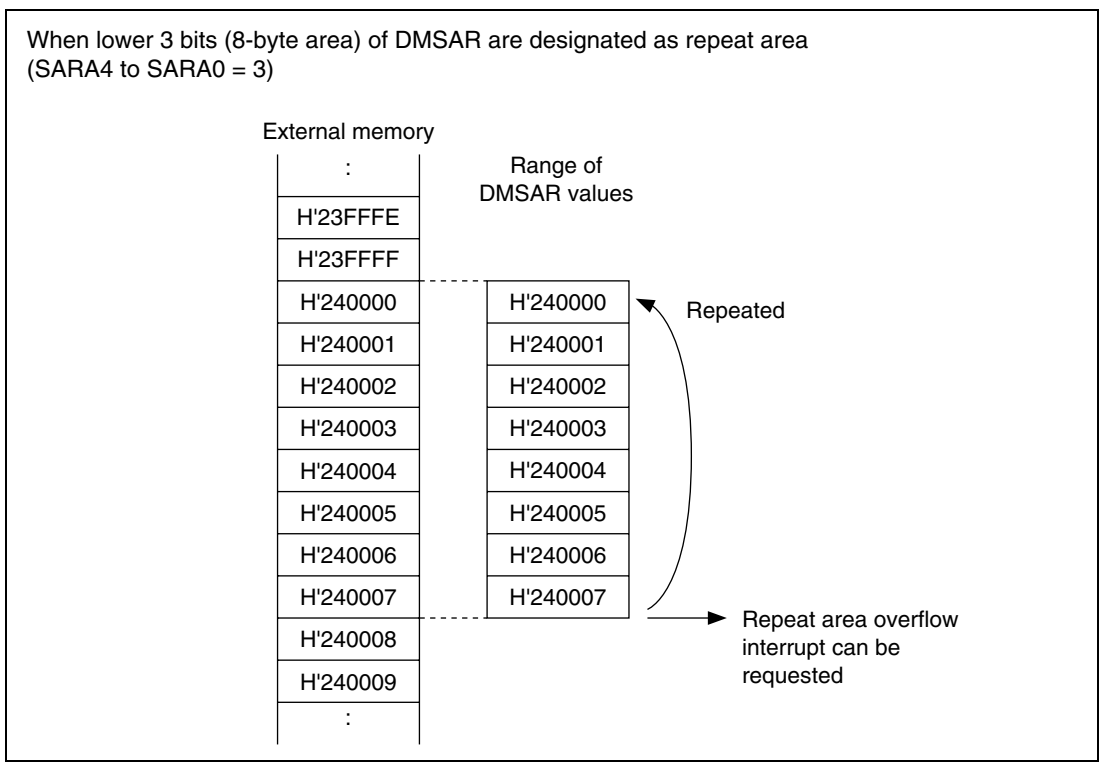


Figure 7.9 Example of Repeat Area Function Operation

Caution is required when the repeat area overflow interrupt function is used together with block transfer mode. If transfer is always terminated when repeat area overflow occurs in block transfer mode, the block size must be a power of two, or alternatively, the address register value must be set so that the end of a block coincides with the end of the repeat area range.

If repeat area overflow occurs while a block is being transferred in block transfer mode, the repeat interrupt request is held pending until the end of the block, and transfer overrun will occur. Figure 7.10 shows an example in which block transfer mode is used together with the repeat area function.

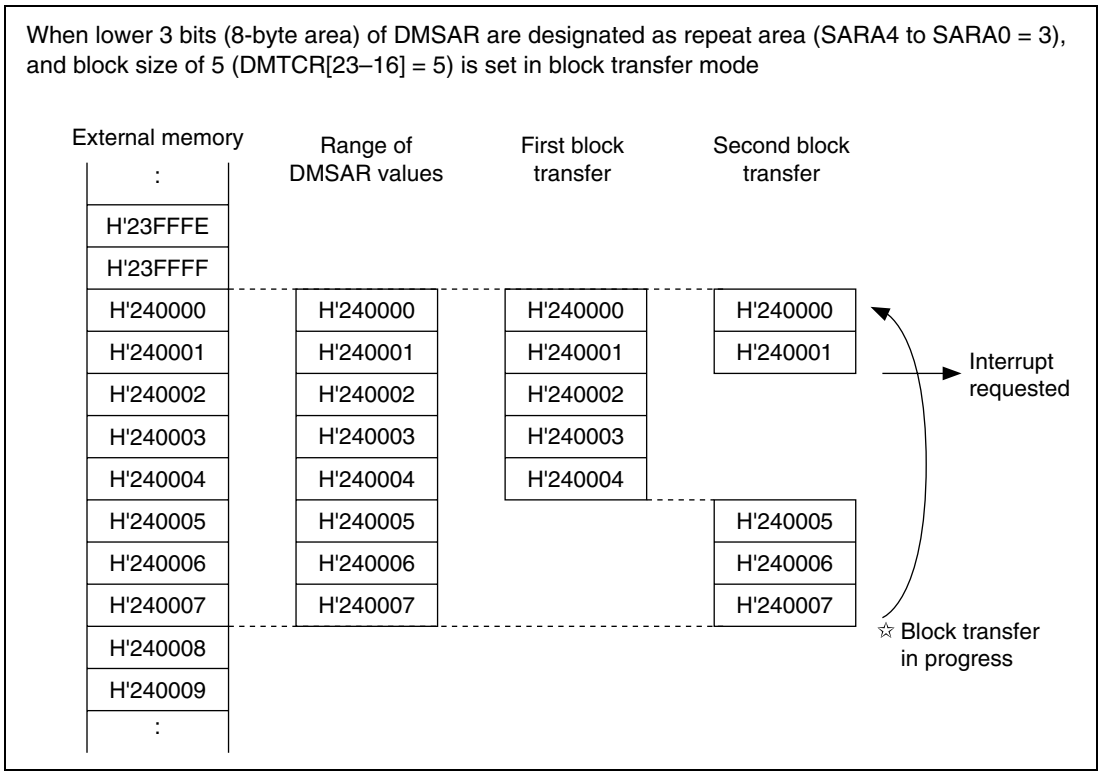


Figure 7.10 Example of Repeat Area Function Operation in Block Transfer Mode

7.4.7 Registers during DMA Transfer Operation

DMAC register values are updated as DMA transfer processing is performed. The updated values depend on various settings and the transfer status. The following registers and bits are updated: DMSAR, DMDAR, DMTCR, and bits DA, BEF, and IRF in DMMDR.

DMA Source Address Register (DMSAR): When the DMSAR address is accessed as the transfer source, after the DMSAR value is output, DMSAR is updated with the address to be accessed next. Bits SAT1 and SAT0 in DMACR specify incrementing or decrementing. The address is fixed when SAT1 = 0, incremented when SAT1 = 1 and SAT0 = 0, and decremented when SAT0 = 1.

The size of the increment or decrement is determined by the size of the data transferred. When the LWSIZE and DTSIZE bits in DMMDR = 0, the data is byte-size and the address is incremented or decremented by 1; when LWSIZE = 0 and DTSIZE = 1, the data is word-size and the address is incremented or decremented by 2; when LWSIZE = 1 and DTSIZE = 0, the data is longword-size and the address is incremented or decremented by 4.

When a repeat area setting is made, the operation conforms to that setting. The upper part of the address set for the repeat area function is fixed, and is not affected by address updating.

When DMSAR is read during a transfer operation, a longword access must be used. During a transfer operation, DMSAR may be updated without regard to accesses from the CPU, and the correct values may not be read if the upper and lower words are read separately. In a longword access, the DMAC buffers the DMSAR value to ensure that the correct value is output.

Do not write to DMSAR for a channel on which a transfer operation is in progress.

DMA Destination Address Register (DMDAR): When the DMDAR address is accessed as the transfer destination, after the DMDAR value is output, DMDAR is updated with the address to be accessed next. Bits DAT1 and DAT0 in DMACR specify incrementing or decrementing. The address is fixed when DAT1 = 0, incremented when DAT1 = 1 and DAT0 = 0, and decremented when DAT0 = 1.

The size of the increment or decrement is determined by the size of the data transferred. When the LWSIZE and DTSIZE bits in DMMDR = 0, the data is byte-size and the address is incremented or decremented by 1; when LWSIZE = 0 and DTSIZE = 1, the data is word-size and the address is incremented or decremented by 2; when LWSIZE = 1 and DTSIZE = 0, the data is longword-size and the address is incremented or decremented by 4.

When a repeat area setting is made, the operation conforms to that setting. The upper part of the address set for the repeat area function is fixed, and is not affected by address updating.

When DMDAR is read during a transfer operation, a longword access must be used. During a transfer operation, DMDAR may be updated without regard to accesses from the CPU, and the

correct values may not be read if the upper and lower words are read separately. In a longword access, the DMAC buffers the DMDAR value to ensure that the correct value is output.

Do not write to DMDAR for a channel on which a transfer operation is in progress.

DMA Transfer Count Register (DMTCR): When a DMA transfer is performed, the value in DMTCR is decremented by 1. However, when the DMTCR value is 0, transfers are not counted and the DMTCR value does not change.

DMTCR functions differently in block transfer mode. The upper 8 bits, DMTCR[23:16], are used to specify the block size, and their value does not change. The lower 16 bits, DMTCR[15:0], function as a transfer counter, the value of which is decremented by 1 when a DMA transfer is performed. However, when the DMTCR[15:0] value is 0, transfers are not counted and the DMTCR[15:0] value does not change.

In normal transfer mode, all of the lower 24 bits of DMTCR may change, so when DMTCR is read by the CPU during DMA transfer, a longword access must be used. During a transfer operation, DMTCR may be updated without regard to accesses from the CPU, and the correct values may not be read if the upper and lower words are read separately. In a longword access, the DMAC buffers the DMTCR value to ensure that the correct value is output.

In block transfer mode, the upper 8 bits are never updated, so there is no problem with using word access. Do not write to DMTCR for a channel on which a transfer operation is in progress. If there is contention between an address update associated with DMA transfer and a write by the CPU, the CPU write has priority.

In the event of contention between an DMTCR update from 1 to 0 and a write (of a nonzero value) by the CPU, the CPU write value has priority as the DMTCR value, but transfer is terminated. Transfer does not end if the CPU writes 0 to DMTCR.

Figure 7.11 shows DMTCR update operations in normal transfer mode and block transfer mode.

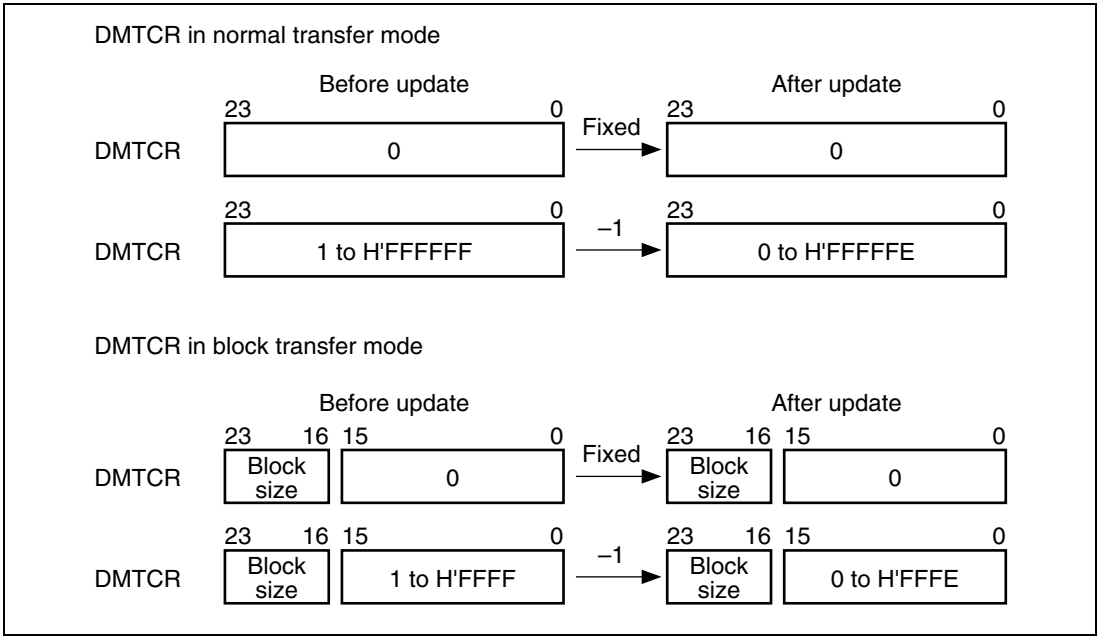


Figure 7.11 DMTCR Update Operations in Normal Transfer Mode and Block Transfer Mode

DA Bit in DMMDR: The DA bit in DMMDR is written to by the CPU to control enabling and disabling of data transfer, but may be cleared automatically by the DMAC due to the DMA transfer status. There are also periods during transfer when a 0-write to the DA bit by the CPU is not immediately effective.

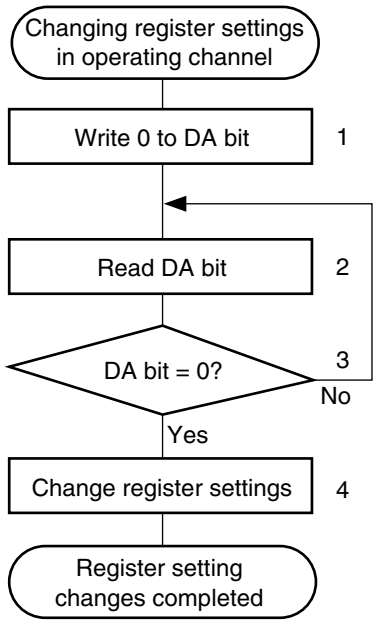
Conditions for DA bit clearing by the DMAC include the following:

- When the DMTCR value changes from 1 to 0, and transfer ends
- When a repeat area overflow interrupt is requested, and transfer ends
- When an NMI interrupt is generated, and transfer halts
- A reset
- Hardware standby mode
- When 0 is written to the DA bit, and transfer halts

When transfer is halted by writing 0 to the DA bit, the DA bit remains at 1 during the DMA transfer period. In block transfer mode, since a block-size transfer is carried out without interruption, the DA bit remains at 1 from the time 0 is written to it until the end of the current block-size transfer.

In burst mode, transfer is halted for up to three DMA transfers following the bus cycle in which 0 is written to the DA bit. The DA bit remains set to 1 from the time of the 0-write until the end of the last DMA cycle. Writes (except to the DA bit) are prohibited to registers of a channel for

which the DA bit is set to 1. When changing register settings after a 0-write to the DA bit, it is necessary to confirm that the DA bit has been cleared to 0. Figure 7.12 shows the procedure for changing register settings in an operating channel.



1. Write 0 to the DA bit in DMMDR.
2. Read the DA bit.
3. Confirm that DA = 0. If DA = 1, this indicates that DMA transfer is in progress.
4. Write the required set values to the registers.

Figure 7.12 Procedure for Changing Register Settings in Operating Channel

BEF Bit in DMMDR: In block transfer mode, the specified number of transfers (equivalent to the block size) is performed in response to a single transfer request. To ensure that the correct number of transfers is carried out, a block-size transfer is always executed, except in the event of a reset, transition to standby mode, or generation of an NMI interrupt.

If an NMI interrupt is generated during block transfer, operation is halted midway through a block-size transfer and the DA bit is cleared to 0, terminating the transfer operation. In this case the BEF bit, which indicates the occurrence of an error during block transfer, is set to 1.

IRF Bit in DMMDR: The IRF bit in DMMDR is set to 1 when an interrupt request source occurs. If the DIE bit in DMMDR is 1 at this time, an interrupt is requested. The timing for setting the IRF bit to 1 is when the DA bit in DMMDR is cleared to 0 and transfer ends following the end of the DMA transfer bus cycle in which the source generating the interrupt occurred.

If the DA bit is set to 1 and transfer is resumed during interrupt handling, the IRF bit is automatically cleared to 0 and the interrupt request is cleared. For details on interrupts, see section 7.5, Interrupt Sources.

7.4.8 Channel Priority

The priority of the DMAC channels is: channel 0 > channel 1 > channel 2> channel 3. Table 7.3 shows the DMAC channel priority.

Table 7.3 DMAC Channel Priority

Channel	Priority
Channel 0	High
Channel 1	
Channel 2	
Channel 3	Low

If transfer requests occur simultaneously for a number of channels, the highest-priority channel according to the priority in table 7.3 is selected for transfer.

Transfer Requests from Multiple Channels (Except Auto Request Cycle Steal Mode): If transfer requests for different channels are issued during a transfer operation, the highest-priority channel (excluding the currently transferring channel) is selected. The selected channel begins transfer after the currently transferring channel releases the bus. If there is a bus request from a bus master other than the DMAC at this time, a cycle for the other bus master is initiated. If there is no other bus request, the bus is released for one cycle.

Channels are not switched during burst transfer or transfer of a block in block transfer mode.

Figure 7.13 shows an example of the transfer timing when transfer requests occur simultaneously for channels 0, 1, and 2. The example in the figure is for external request cycle steal mode.

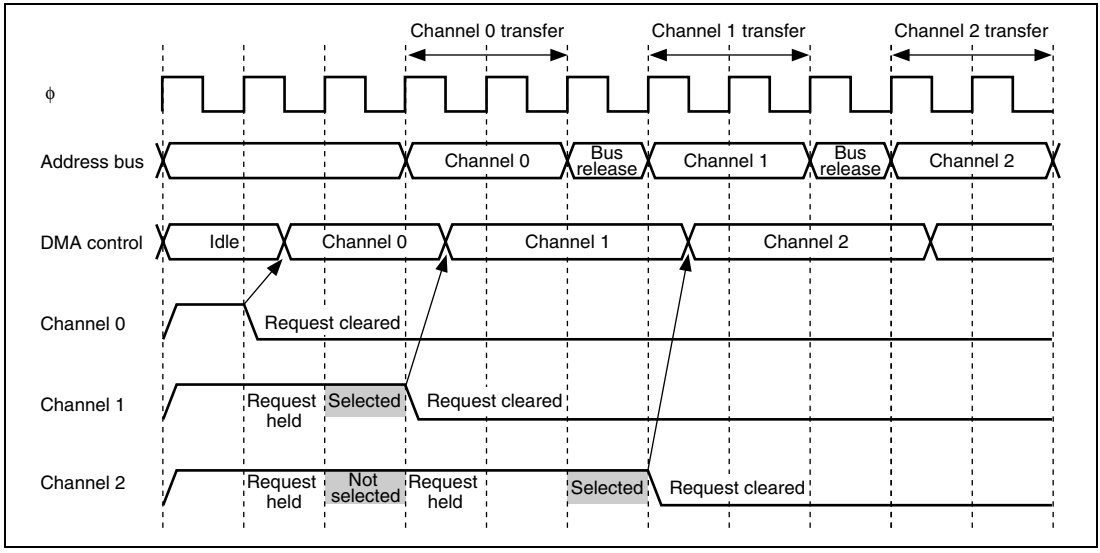


Figure 7.13 Example of Channel Priority Timing

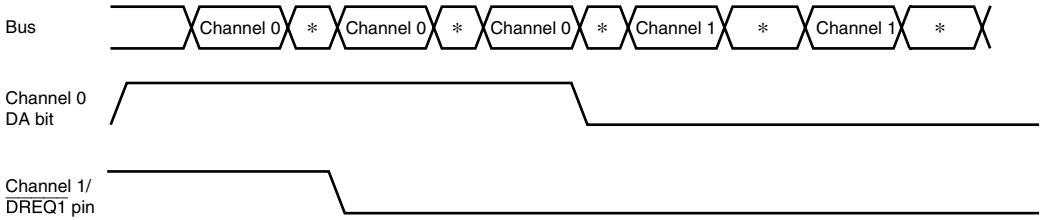
Transfer Requests from Multiple Channels in Auto Request Cycle Steal Mode: If transfer requests for different channels are issued during a transfer in auto request cycle steal mode, the operation depends on the channel priority. If the channel that made the transfer request is of higher priority than the channel currently performing transfer, the channel that made the transfer request is selected. If the channel that made the transfer request is of lower priority than the channel currently performing transfer, that channel's transfer request is held pending, and the currently transferring channel remains selected.

The selected channel begins transfer after the currently transferring channel releases the bus. If there is a bus request from a bus master other than the DMAC at this time, a cycle for the other bus master is initiated. If there is no other bus request, the bus is released for one cycle.

Figure 7.14 shows examples of transfer timing in cases that include auto request cycle steal mode.

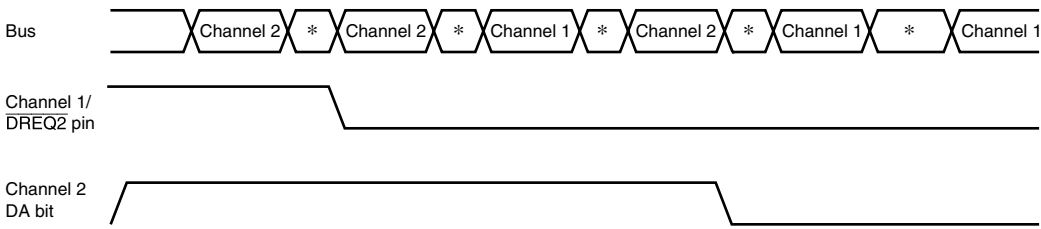
Conditions (1)

Channel 0: Auto request, cycle steal mode
Channel 1: External request, cycle steal mode, low level activation



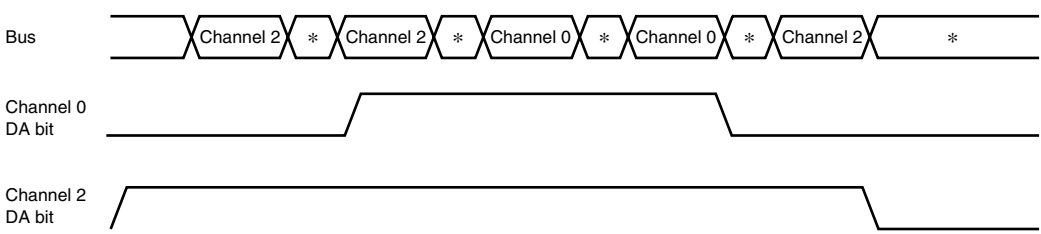
Conditions (2)

Channel 1: External request, cycle steal mode, low level activation
Channel 2: Auto request, cycle steal mode



Conditions (3)

Channel 0: Auto request, cycle steal mode
Channel 2: Auto request, cycle steal mode



*: Bus release

Figure 7.14 Examples of Channel Priority Timing

7.4.9 DMAC Bus Cycles (Dual Address Mode)

Normal Transfer Mode (Cycle Steal Mode): Figure 7.15 shows an example of transfer when $\overline{\text{TEND}}$ output is enabled, and word-size, normal transfer mode (cycle steal mode) is performed from external 16-bit, 2-state access space to external 16-bit, 2-state access space.

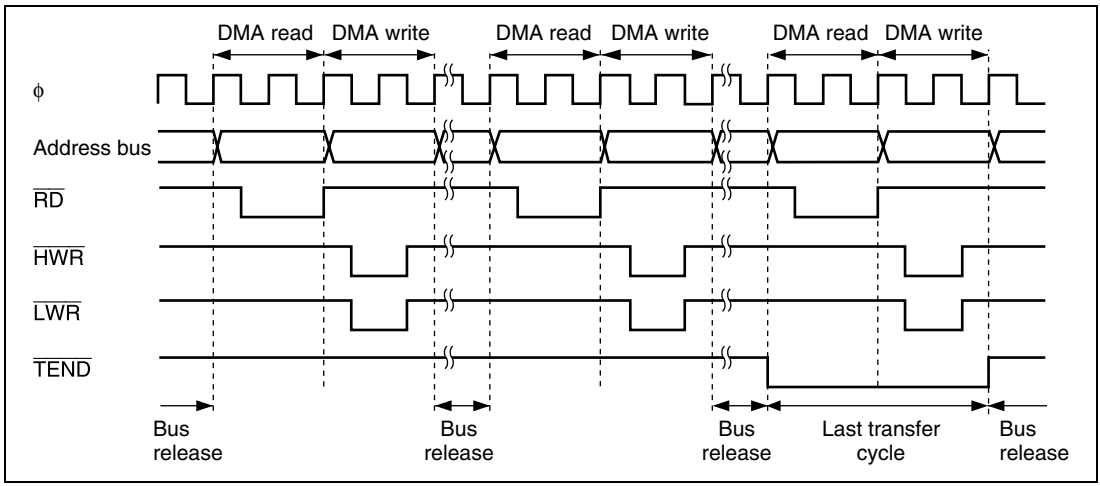


Figure 7.15 Example of Normal Transfer Mode (Cycle Steal Mode) Transfer

After one byte, word, or longword has been transferred, the bus is released. While the bus is released, at least one CPU bus cycle is initiated.

Normal Transfer Mode (Burst Mode): Figure 7.16 shows an example of transfer when $\overline{\text{TEND}}$ output is enabled, and word-size, normal transfer mode (burst mode) is performed from external 16-bit, 2-state access space to external 16-bit, 2-state access space.

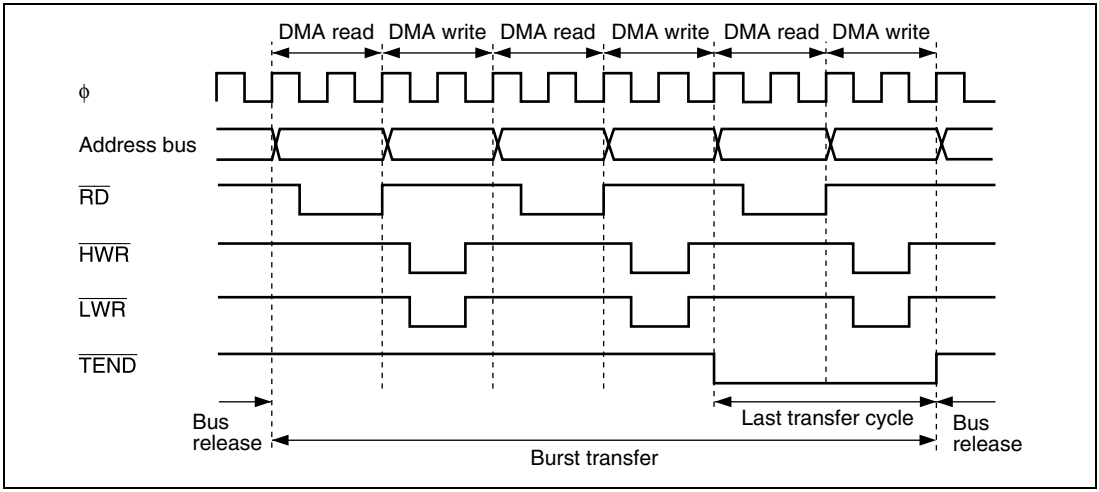


Figure 7.16 Example of Normal Transfer Mode (Burst Mode) Transfer

In burst mode, one-byte, one-word, or one-longword transfers are executed continuously until transfer ends. Once burst transfer starts, requests from other channels, even of higher priority, are held pending until transfer ends.

If an NMI interrupt is generated while a channel designated for burst transfer is enabled for transfer, the DA bit is cleared and transfer is disabled. If a burst transfer has already been initiated within the DMAC, the bus is released on completion of the currently executing byte or word transfer, and burst transfer is aborted. If the last transfer cycle in burst transfer has been initiated within the DMAC, transfer is executed to the end even if the DA bit is cleared.

Normal Transfer Mode (Cycle Steal Mode: Transfer Source is USB): Figure 7.17 shows an example of transfer when USB transfer is enabled, and word-size, normal transfer mode (cycle steal mode) is performed from the FIFO in the USB to external 16-bit, 2-state access space.

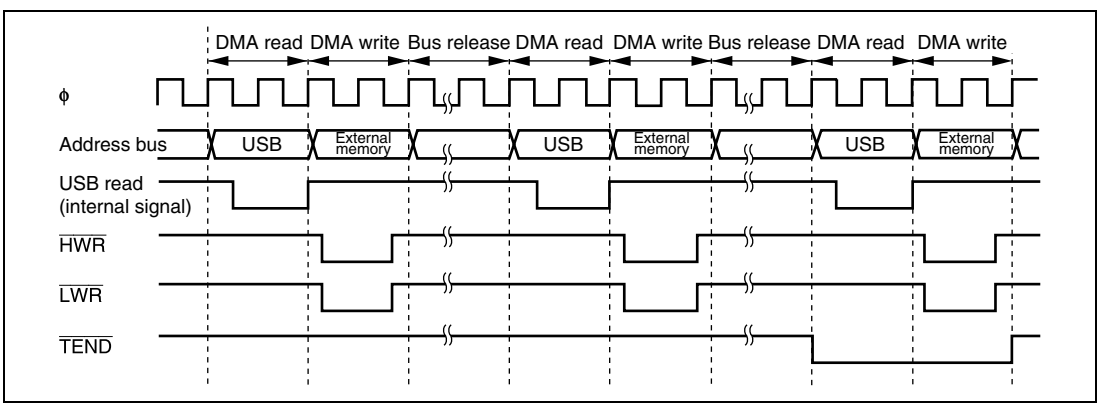


Figure 7.17 Example of Normal Transfer Mode (Cycle Steal Mode) Transfer (Transfer Source: USB)

After one byte, word, or longword has been transferred, the bus is released. While the bus is released, at least one CPU bus cycle is initiated.

Normal Transfer Mode (Cycle Steal Mode: Transfer Destination is USB): Figure 7.18 shows an example of transfer when USB transfer is enabled, and word-size, normal transfer mode (cycle steal mode) is performed from external 16-bit, 2-state access space to the FIFO in the USB.

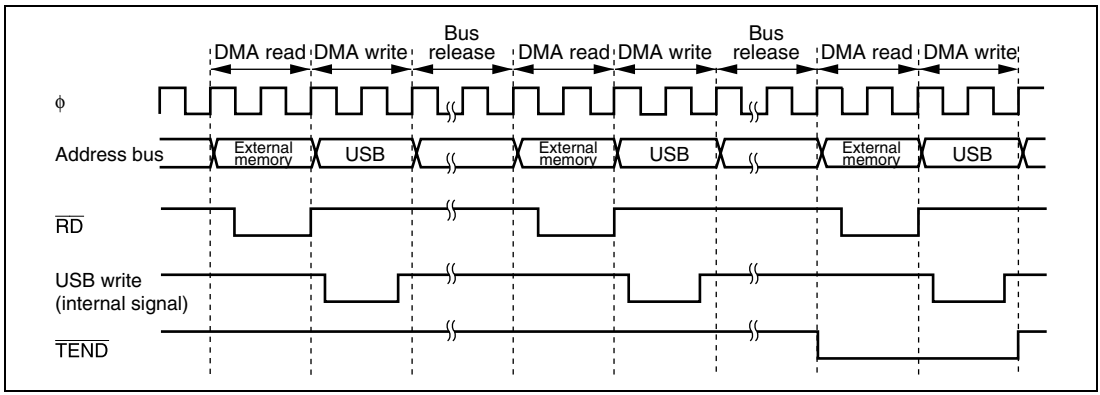


Figure 7.18 Example of Normal Transfer Mode (Cycle Steal Mode) Transfer (Transfer Destination: USB)

After one byte, word, or longword has been transferred, the bus is released. While the bus is released, at least one CPU bus cycle is initiated.

Block Transfer Mode (Cycle Steal Mode): Figure 7.19 shows an example of transfer when $\overline{\text{TEND}}$ output is enabled, and word-size, block transfer mode (cycle steal mode) is performed from external 16-bit, 2-state access space to external 16-bit, 2-state access space.

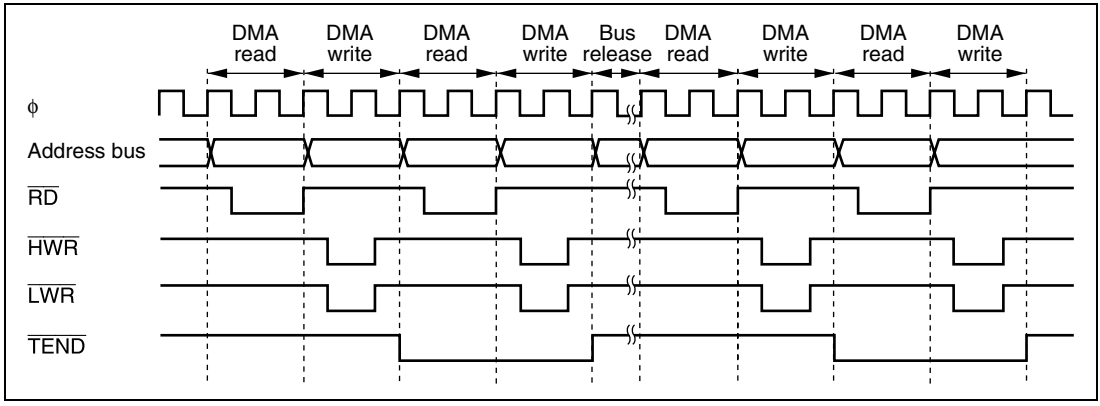
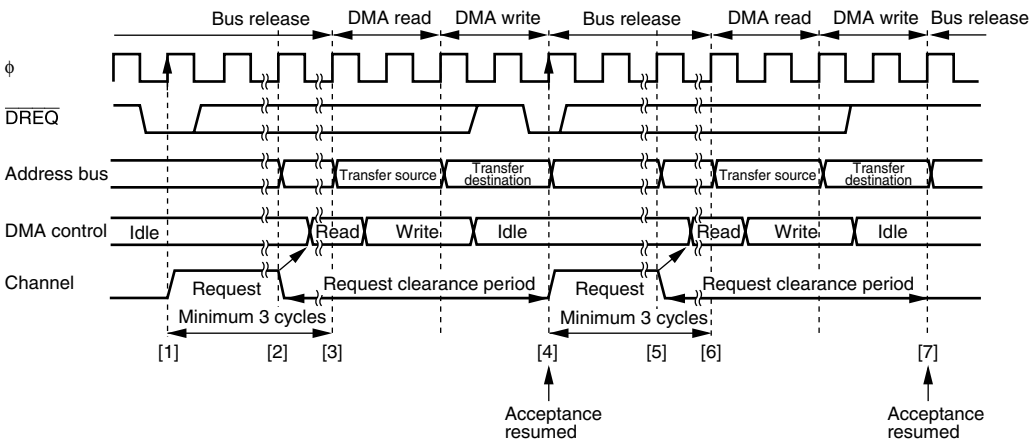


Figure 7.19 Example of Block Transfer Mode (Cycle Steal Mode) Transfer

One block is transferred in response to one transfer request, and after the transfer, the bus is released. While the bus is released, one or more CPU bus cycles are initiated.

$\overline{\text{DREQ}}$ Pin Falling Edge Activation Timing: Figure 7.20 shows an example of normal mode transfer activated by the $\overline{\text{DREQ}}$ pin falling edge.



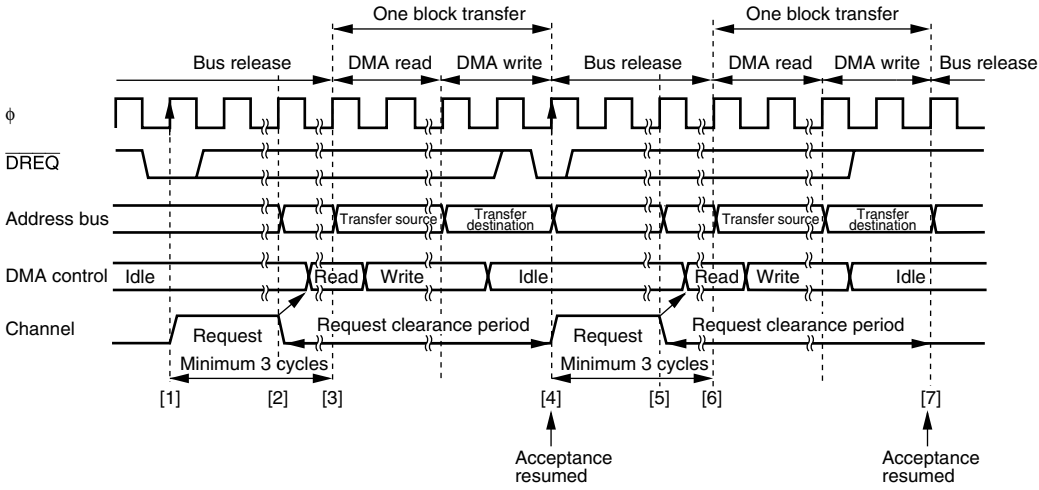
[1] Acceptance after transfer enabling; $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.
[2], [5] Request is cleared at end of next bus cycle, and activation is started in DMAC.
[3], [6] DMA cycle start; $\overline{\text{DREQ}}$ pin high level sampling is started at rise of φ.
[4], [7] When $\overline{\text{DREQ}}$ pin high level has been sampled, acceptance is resumed after completion of write cycle. (As in [1], $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.)

Figure 7.20 Example of Normal Mode Transfer Activated by $\overline{\text{DREQ}}$ Pin Falling Edge

$\overline{\text{DREQ}}$ pin sampling is performed in each cycle starting at the next rise of φ after the end of the DMMDR write cycle for setting the transfer-enabled state.

When a low level is sampled at the $\overline{\text{DREQ}}$ pin while acceptance via the $\overline{\text{DREQ}}$ pin is possible, the request is held within the DMAC. Then when activation is initiated within the DMAC, the request is cleared, and $\overline{\text{DREQ}}$ pin high level sampling for edge sensing is started. If $\overline{\text{DREQ}}$ pin high level sampling is completed by the end of the DMA write cycle, acceptance resumes after the end of the write cycle, and $\overline{\text{DREQ}}$ pin low level sampling is performed again; this sequence of operations is repeated until the end of the transfer.

Figure 7.21 shows an example of block transfer mode transfer activated by the $\overline{\text{DREQ}}$ pin falling edge.



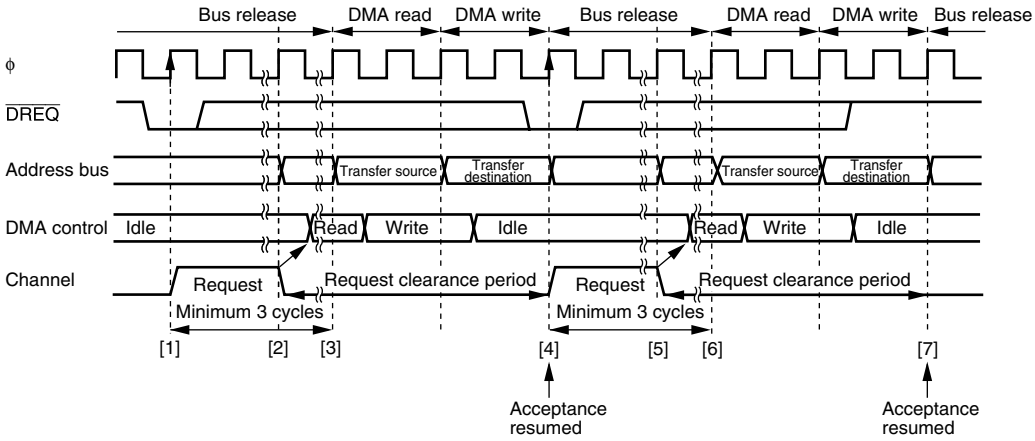
[1] Acceptance after transfer enabling; $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.
[2], [5] Request is cleared at end of next bus cycle, and activation is started in DMAC.
[3], [6] DMA cycle start; $\overline{\text{DREQ}}$ pin high level sampling is started at rise of φ.
[4], [7] When $\overline{\text{DREQ}}$ pin high level has been sampled, acceptance is resumed after completion of dead cycle. (As in [1], $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.)

Figure 7.21 Example of Block Transfer Mode Transfer Activated by $\overline{\text{DREQ}}$ Pin Falling Edge

$\overline{\text{DREQ}}$ pin sampling is performed in each cycle starting at the next rise of φ after the end of the DMMDR write cycle for setting the transfer-enabled state.

When a low level is sampled at the $\overline{\text{DREQ}}$ pin while acceptance via the $\overline{\text{DREQ}}$ pin is possible, the request is held within the DMAC. Then when activation is initiated within the DMAC, the request is cleared, and $\overline{\text{DREQ}}$ pin high level sampling for edge sensing is started. If $\overline{\text{DREQ}}$ pin high level sampling is completed by the end of the DMA write cycle, acceptance resumes after the end of the write cycle, and $\overline{\text{DREQ}}$ pin low level sampling is performed again; this sequence of operations is repeated until the end of the transfer.

$\overline{\text{DREQ}}$ Pin Low Level Activation Timing: Figure 7.22 shows an example of normal mode transfer activated by the $\overline{\text{DREQ}}$ pin low level.



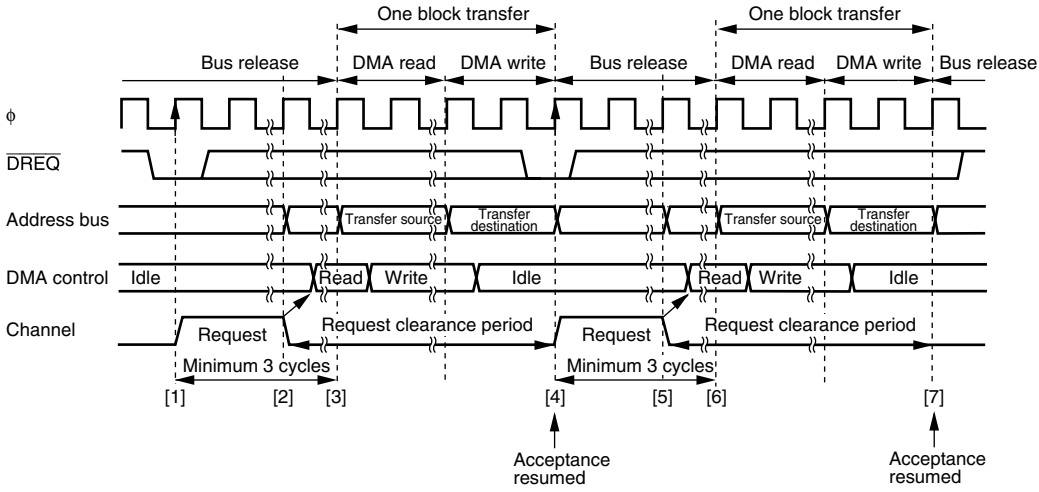
[1] Acceptance after transfer enabling; $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.
[2], [5] Request is cleared at end of next bus cycle, and activation is started in DMAC.
[3], [6] DMA cycle is started.
[4], [7] Acceptance is resumed after completion of write cycle.
(As in [1], $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.)

Figure 7.22 Example of Normal Mode Transfer Activated by $\overline{\text{DREQ}}$ Pin Low Level

$\overline{\text{DREQ}}$ pin sampling is performed in each cycle starting at the next rise of φ after the end of the DMMDR write cycle for setting the transfer-enabled state.

When a low level is sampled at the $\overline{\text{DREQ}}$ pin while acceptance via the $\overline{\text{DREQ}}$ pin is possible, the request is held within the DMAC. Then when activation is initiated within the DMAC, the request is cleared. At the end of the write cycle, acceptance resumes and $\overline{\text{DREQ}}$ pin low level sampling is performed again; this sequence of operations is repeated until the end of the transfer.

Figure 7.23 shows an example of block transfer mode transfer activated by the $\overline{\text{DREQ}}$ pin low level.



[1] Acceptance after transfer enabling; $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.
[2], [5] Request is cleared at end of next bus cycle, and activation is started in DMAC.
[3], [6] DMA cycle is started.
[4], [7] Acceptance is resumed after completion of dead cycle.
(As in [1], $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.)

Figure 7.23 Example of Block Transfer Mode Transfer Activated by $\overline{\text{DREQ}}$ Pin Low Level

$\overline{\text{DREQ}}$ pin sampling is performed in each cycle starting at the next rise of φ after the end of the DMMDR write cycle for setting the transfer-enabled state.

When a low level is sampled at the $\overline{\text{DREQ}}$ pin while acceptance via the $\overline{\text{DREQ}}$ pin is possible, the request is held within the DMAC. Then when activation is initiated within the DMAC, the request is cleared. At the end of the write cycle, acceptance resumes and $\overline{\text{DREQ}}$ pin low level sampling is performed again; this sequence of operations is repeated until the end of the transfer.

7.4.10 DMAC Bus Cycles (Single Address Mode)

Single Address Mode (Read): Figure 7.24 shows an example of transfer when $\overline{\text{TEND}}$ output is enabled, and byte-size, single address mode transfer (read) is performed from external 8-bit, 2-state access space to an external device.

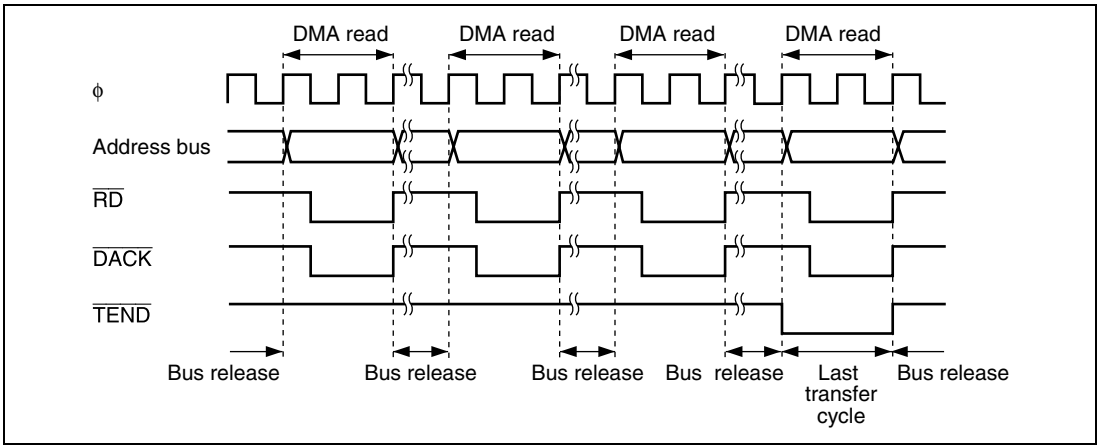


Figure 7.24 Example of Single Address Mode (Byte Read) Transfer

Figure 7.25 shows an example of transfer when $\overline{\text{TEND}}$ output is enabled, and word-size, single address mode transfer (read) is performed from external 8-bit, 2-state access space to an external device.

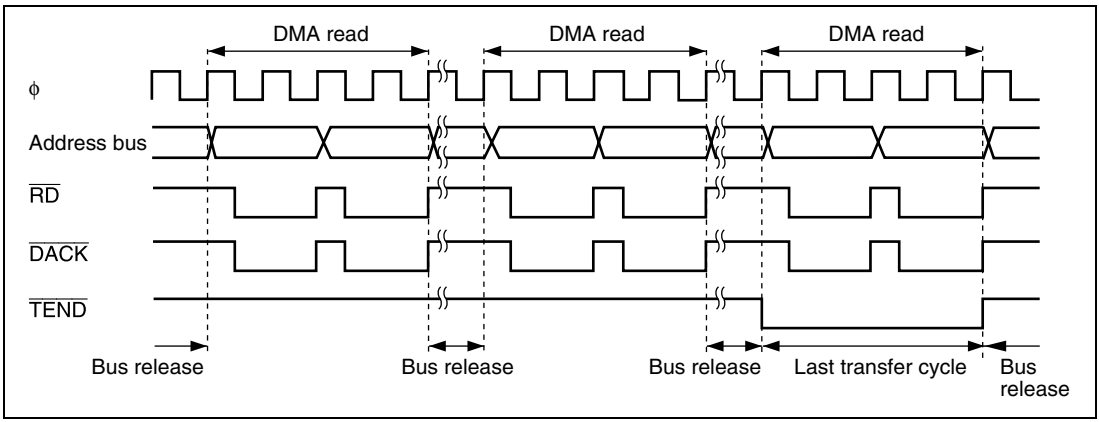


Figure 7.25 Example of Single Address Mode (Word Read) Transfer

After one byte or word has been transferred in response to one transfer request, the bus is released. While the bus is released, one or more CPU bus cycles are initiated.

Figure 7.26 shows an example of transfer when $\overline{\text{TEND}}$ output is enabled, and longword-size, single address mode transfer (read) is performed from external 16-bit, 2-state access space to an external device.

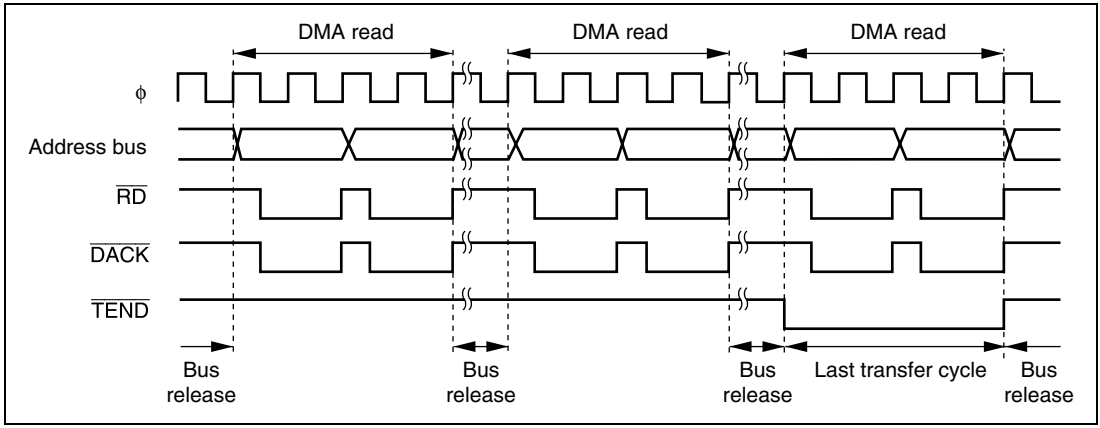


Figure 7.26 Example of Single Address Mode (Longword Read) Transfer

Single Address Mode (Write): Figure 7.27 shows an example of transfer when $\overline{\text{TEND}}$ output is enabled, and byte-size, single address mode transfer (write) is performed from an external device to external 8-bit, 2-state access space.

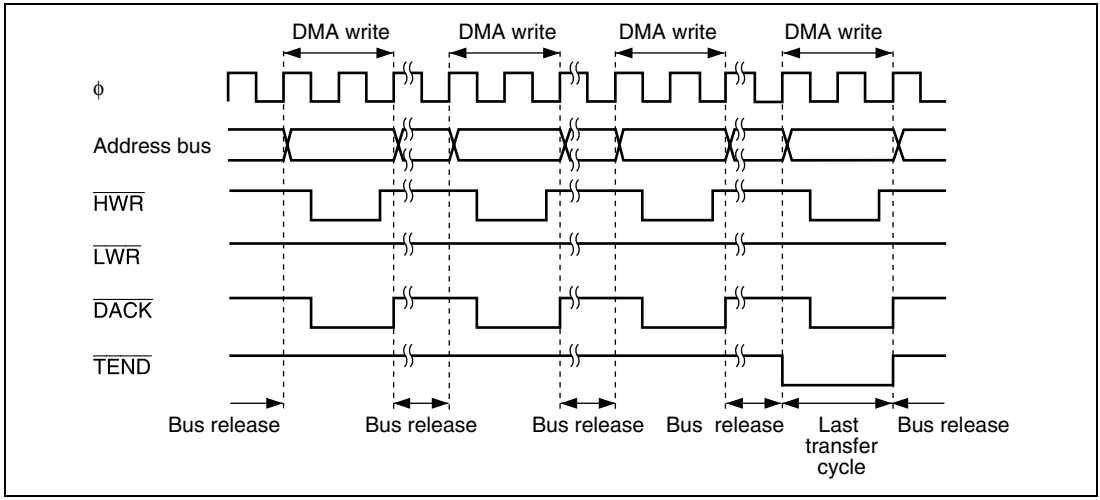


Figure 7.27 Example of Single Address Mode (Byte Write) Transfer

Figure 7.28 shows an example of transfer when $\overline{\text{TEND}}$ output is enabled, and word-size, single address mode transfer (write) is performed from an external device to external 8-bit, 2-state access space.

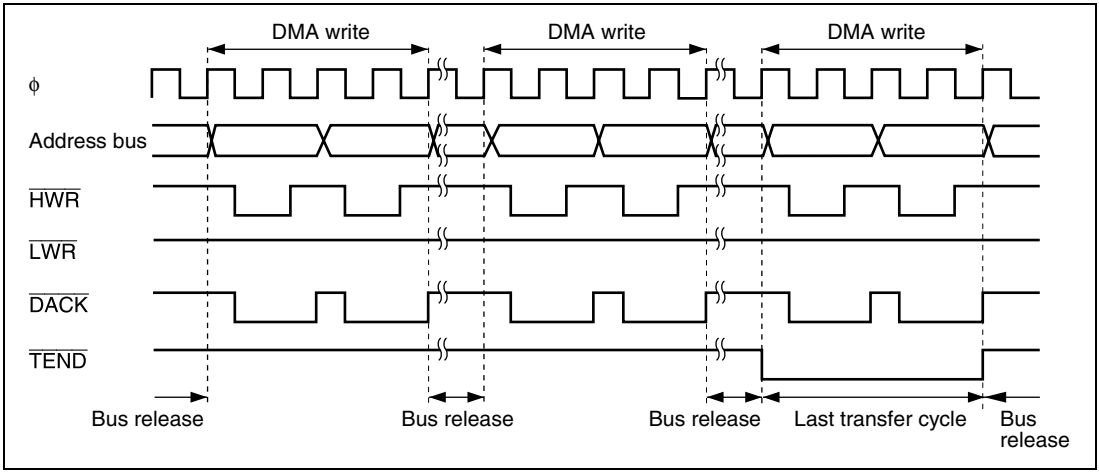


Figure 7.28 Example of Single Address Mode (Word Write) Transfer

After one byte or word has been transferred in response to one transfer request, the bus is released. While the bus is released, one or more CPU bus cycles are initiated.

Figure 7.29 shows an example of transfer when $\overline{\text{TEND}}$ output is enabled, and longword-size, single address mode transfer (write) is performed from an external device to external 16-bit, 2-state access space.

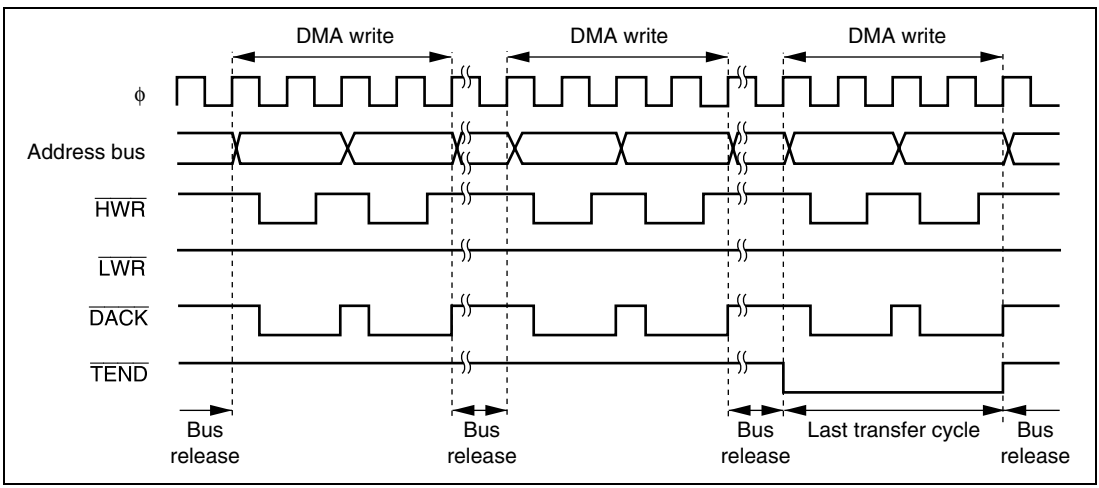
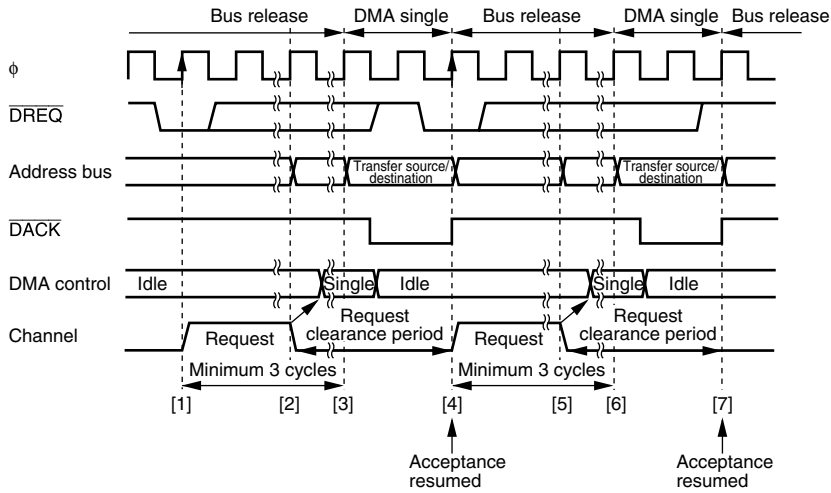


Figure 7.29 Example of Single Address Mode (Longword Write) Transfer

$\overline{\text{DREQ}}$ Pin Falling Edge Activation Timing: Figure 7.30 shows an example of single address mode transfer activated by the $\overline{\text{DREQ}}$ pin falling edge.



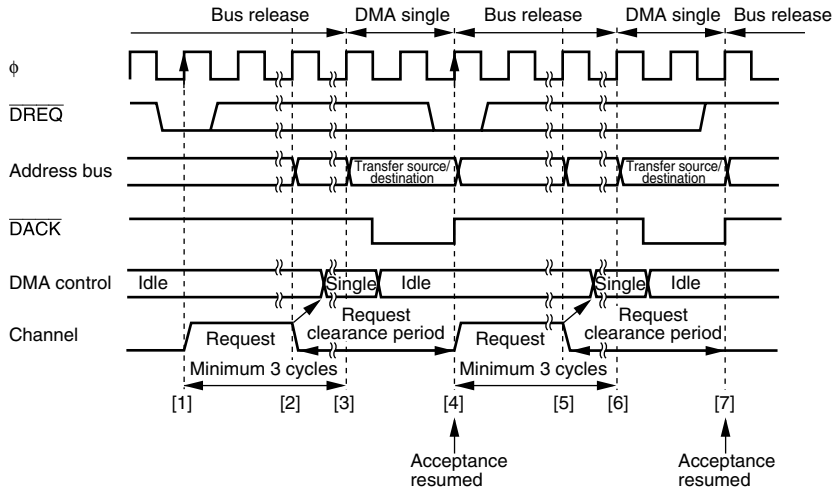
[1] Acceptance after transfer enabling; $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.
[2], [5] Request is cleared at end of next bus cycle, and activation is started in DMAC.
[3], [6] DMA cycle start; $\overline{\text{DREQ}}$ pin high level sampling is started at rise of φ.
[4], [7] When $\overline{\text{DREQ}}$ pin high level has been sampled, acceptance is resumed after completion of single cycle. (As in [1], $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.)

Figure 7.30 Example of Single Address Mode Transfer Activated by $\overline{\text{DREQ}}$ Pin Falling Edge

$\overline{\text{DREQ}}$ pin sampling is performed in each cycle starting at the next rise of φ after the end of the DMMDR write cycle for setting the transfer-enabled state.

When a low level is sampled at the $\overline{\text{DREQ}}$ pin while acceptance via the $\overline{\text{DREQ}}$ pin is possible, the request is held within the DMAC. Then when activation is initiated within the DMAC, the request is cleared, and $\overline{\text{DREQ}}$ pin high level sampling for edge sensing is started. If $\overline{\text{DREQ}}$ pin high level sampling is completed by the end of the DMA single cycle, acceptance resumes after the end of the single cycle, and $\overline{\text{DREQ}}$ pin low level sampling is performed again; this sequence of operations is repeated until the end of the transfer.

$\overline{\text{DREQ}}$ Pin Low Level Activation Timing: Figure 7.31 shows an example of single address mode transfer activated by the $\overline{\text{DREQ}}$ pin low level.



[1] Acceptance after transfer enabling; $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.
[2], [5] Request is cleared at end of next bus cycle, and activation is started in DMAC.
[3], [6] DMA cycle is started.
[4], [7] Acceptance is resumed after completion of single cycle.
(As in [1], $\overline{\text{DREQ}}$ pin low level is sampled at rise of φ, and request is held.)

Figure 7.31 Example of Single Address Mode Transfer Activated by $\overline{\text{DREQ}}$ Pin Low Level

$\overline{\text{DREQ}}$ pin sampling is performed in each cycle starting at the next rise of φ after the end of the DMMDR write cycle for setting the transfer-enabled state.

When a low level is sampled at the $\overline{\text{DREQ}}$ pin while acceptance via the $\overline{\text{DREQ}}$ pin is possible, the request is held within the DMAC. Then when activation is initiated within the DMAC, the request is cleared. At the end of the single cycle, acceptance resumes and $\overline{\text{DREQ}}$ pin low level sampling is performed again; this sequence of operations is repeated until the end of the transfer.

7.4.11 Examples of Operation Timing in Each Mode

Examples of operation timings for various conditions in each mode are shown. The contention with other bus master is described by using the CPU external bus cycle as an example.

Auto Request/Cycle Steal Mode/Normal Transfer Mode: When the DA bit is set to 1 in DMMDR, an DMA transfer cycle is started a minimum of three cycles later. There is a one-cycle bus release interval between the end of a one-transfer-unit DMA cycle and the start of the next transfer.

If there is a transfer request for another channel of higher priority, the transfer request by the original channel is held pending, and transfer is performed on the higher-priority channel from the next transfer. Transfer on the original channel is resumed on completion of the higher-priority channel transfer.

Figures 7.32 to 7.34 show operation timing examples for various conditions.

- No contention/dual address mode (see figure 7.32)
- CPU cycles/single address mode (see figure 7.33)
- Contention with another channel/single address mode (see figure 7.34)

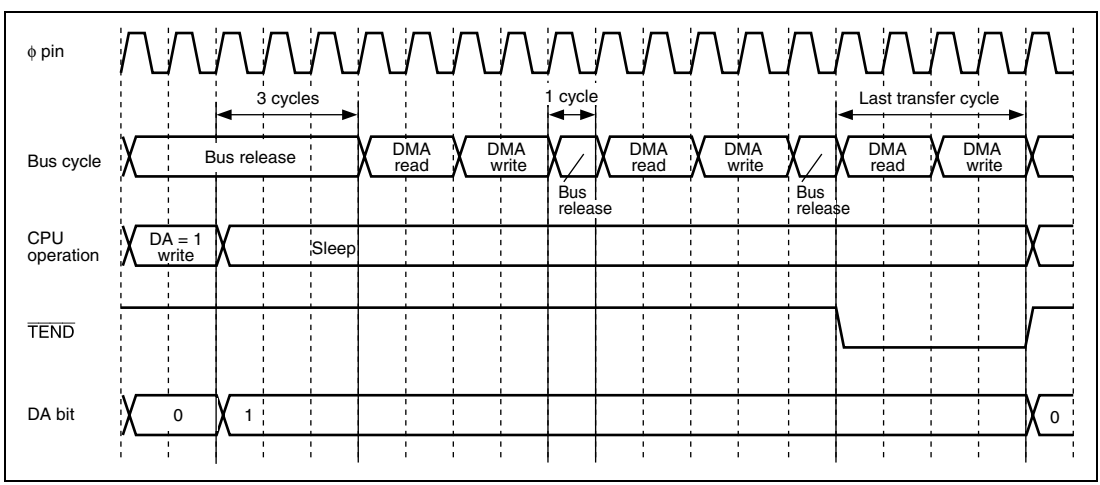


Figure 7.32 Auto Request/Cycle Steal Mode/Normal Transfer Mode (No Contention/Dual Address Mode)

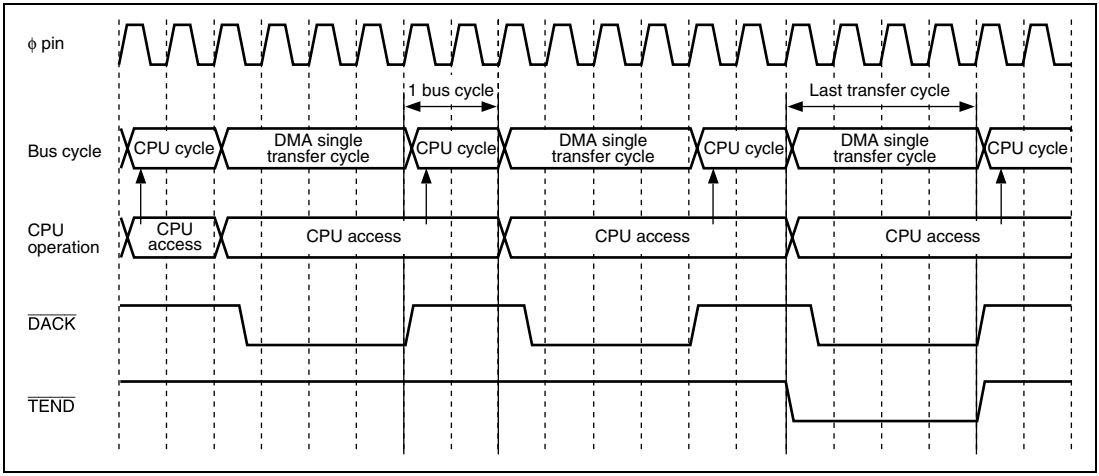


Figure 7.33 Auto Request/Cycle Steal Mode/Normal Transfer Mode (CPU Cycles/Single Address Mode)

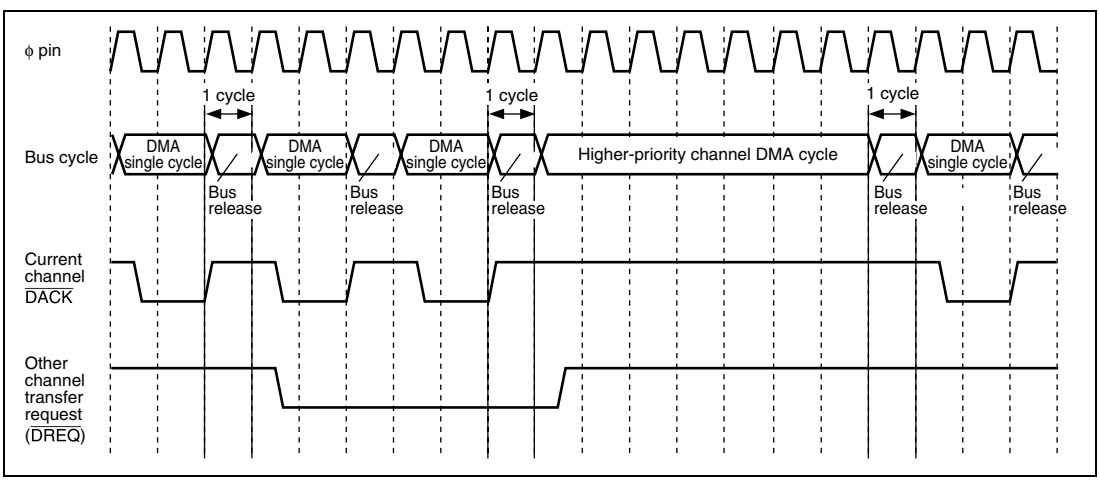


Figure 7.34 Auto Request/Cycle Steal Mode/Normal Transfer Mode (Contention with Another Channel/Single Address Mode)

Auto Request/Burst Mode/Normal Transfer Mode: When the DA bit is set to 1 in DMMDR, an DMA transfer cycle is started a minimum of three cycles later. Once transfer is started, it continues (as a burst) until the transfer end condition is satisfied.

Transfer requests for other channels are held pending until the end of transfer on the current channel.

Figures 7.35 to 7.37 show operation timing examples for various conditions.

- CPU cycles/dual address mode (see figure 7.35)
- CPU cycles/single address mode (see figure 7.36)
- Contention with another channel/dual address mode (see figure 7.37)

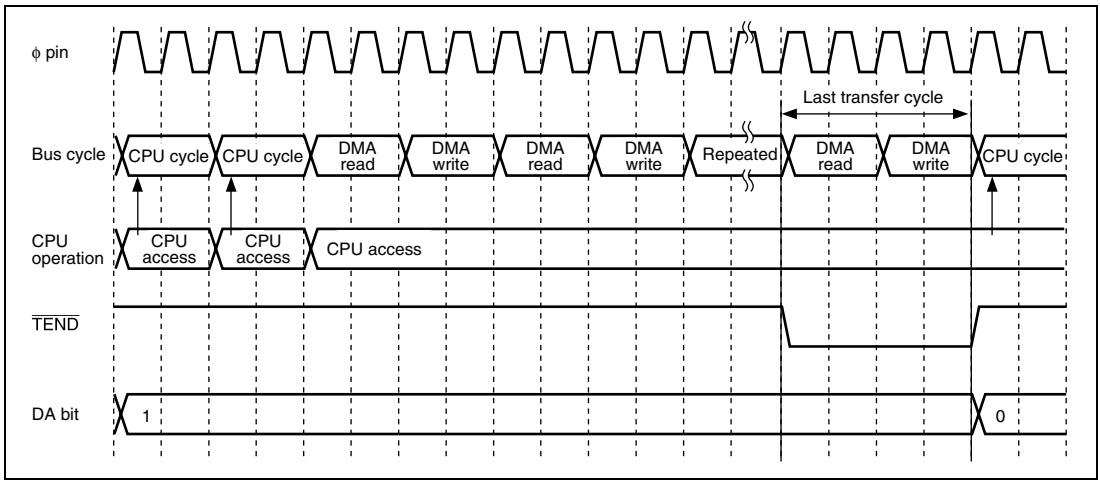


Figure 7.35 Auto Request/Burst Mode/Normal Transfer Mode (CPU Cycles/Dual Address Mode)

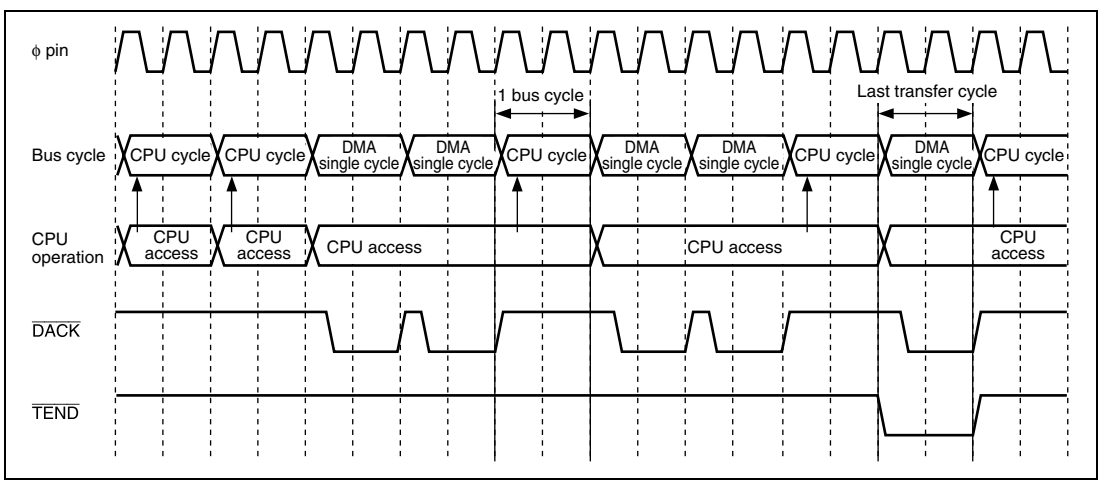


Figure 7.36 Auto Request/Burst Mode/Normal Transfer Mode (CPU Cycles/Single Address Mode)

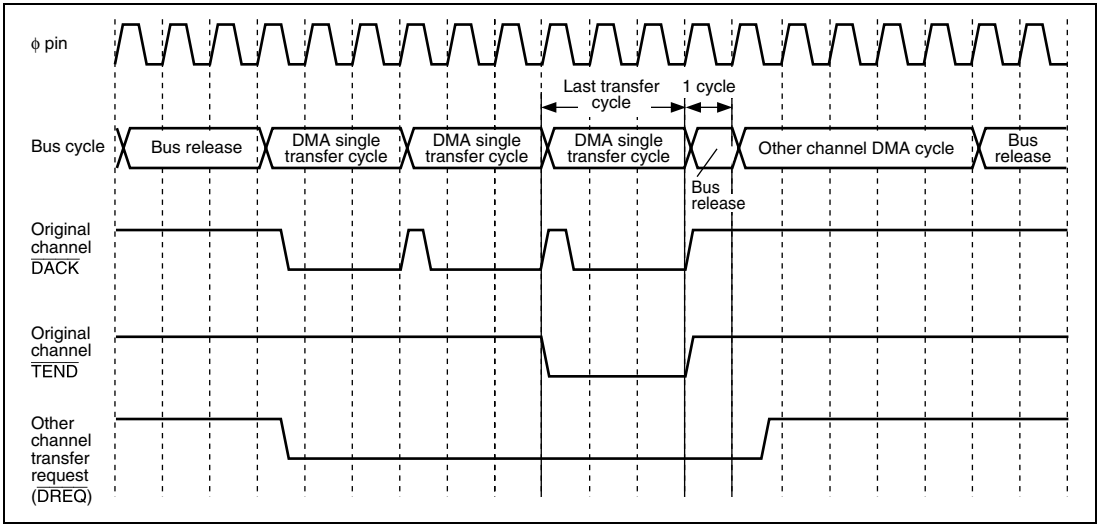


Figure 7.37 Auto Request/Burst Mode/Normal Transfer Mode (Contention with Another Channel/Single Address Mode)

External Request/Cycle Steal Mode/Normal Transfer Mode: In external request mode, an DMA transfer cycle is started a minimum of three cycles after a transfer request is accepted. The next transfer request is accepted after the end of a one-transfer-unit DMA cycle. For external bus space CPU cycles, at least two bus cycles are generated before the next DMA cycle.

If a transfer request is generated for another channel, an DMA cycle for the other channel is generated before the next DMA cycle.

The $\overline{\text{DREQ}}$ pin sensing timing is different for low level sensing and falling edge sensing. The same applies to transfer request acceptance and transfer start timing.

Figures 7.38 to 7.41 show operation timing examples for various conditions.

- No contention/dual address mode/low level sensing (see figure 7.38)
- CPU cycles/single address mode/low level sensing (see figure 7.39)
- No contention/single address mode/falling edge sensing (see figure 7.40)
- Contention with another channel/dual address mode/low level sensing (see figure 7.41)

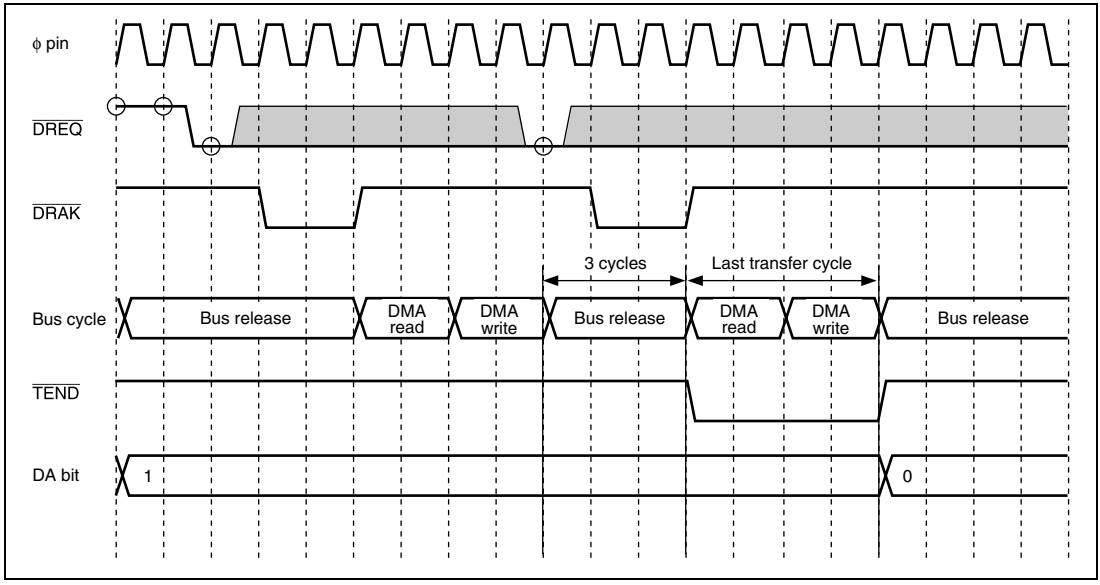


Figure 7.38 External Request/Cycle Steal Mode/Normal Transfer Mode (No Contention/Dual Address Mode/Low Level Sensing)

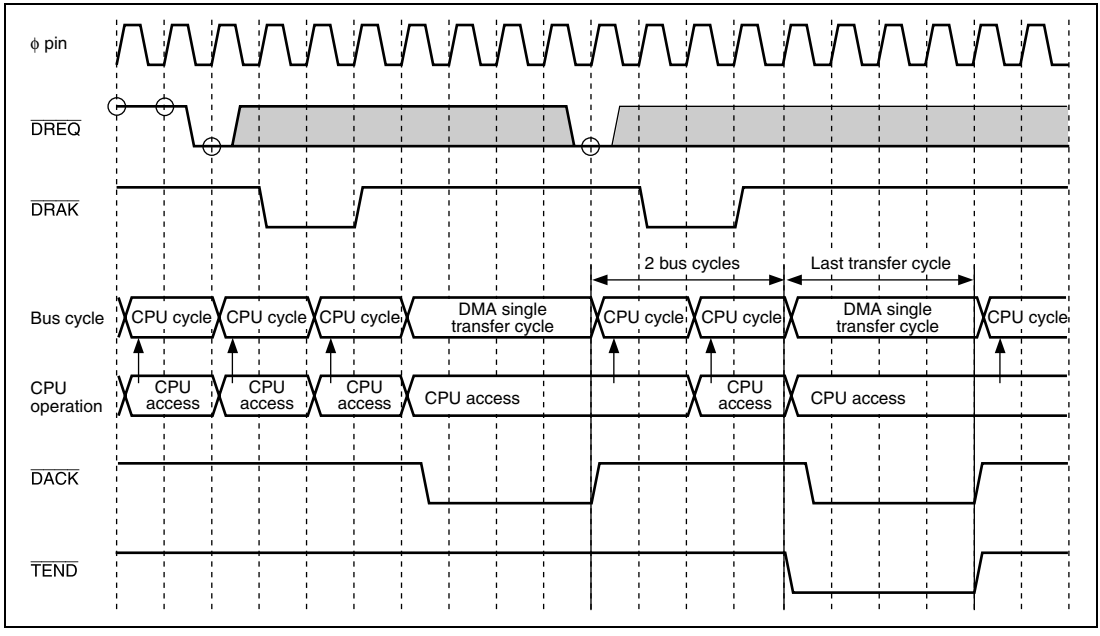


Figure 7.39 External Request/Cycle Steal Mode/Normal Transfer Mode (CPU Cycles/Single Address Mode/Low Level Sensing)

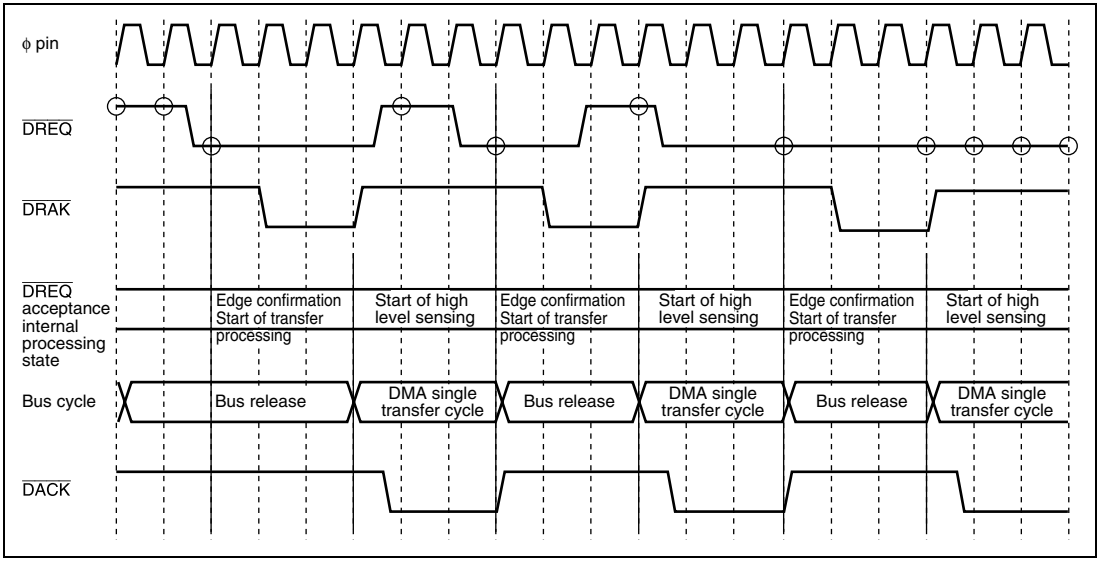


Figure 7.40 External Request/Cycle Steal Mode/Normal Transfer Mode (No Contention/Single Address Mode/Falling Edge Sensing)

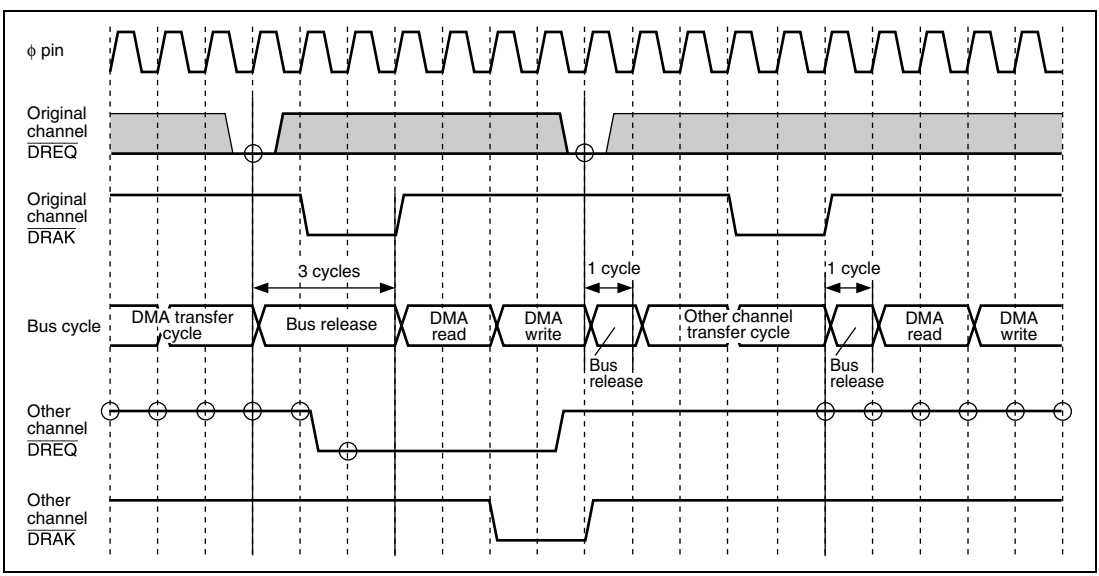


Figure 7.41 External Request/Cycle Steal Mode/Normal Transfer Mode (Contention with Another Channel/Dual Address Mode/Low Level Sensing)

External Request/Cycle Steal Mode/Block Transfer Mode: In block transfer mode, transfer of one block is performed continuously in the same way as in burst mode. The timing of the start of the next block transfer is the same as in normal transfer mode. If a transfer request is generated for another channel, an DMA cycle for the other channel is generated before the next block transfer. The $\overline{\text{DREQ}}$ pin sensing timing is different for low level sensing and falling edge sensing. The same applies to transfer request acceptance and transfer start timing.

Figures 7.42 to 7.45 show operation timing examples for various conditions.

- No contention/dual address mode/low level sensing (see figure 7.42)
- No contention/single address mode/falling edge sensing (see figure 7.43)
- CPU cycles/single address mode/low level sensing (see figure 7.44)
- Contention with another channel/dual address mode/low level sensing (see figure 7.45)

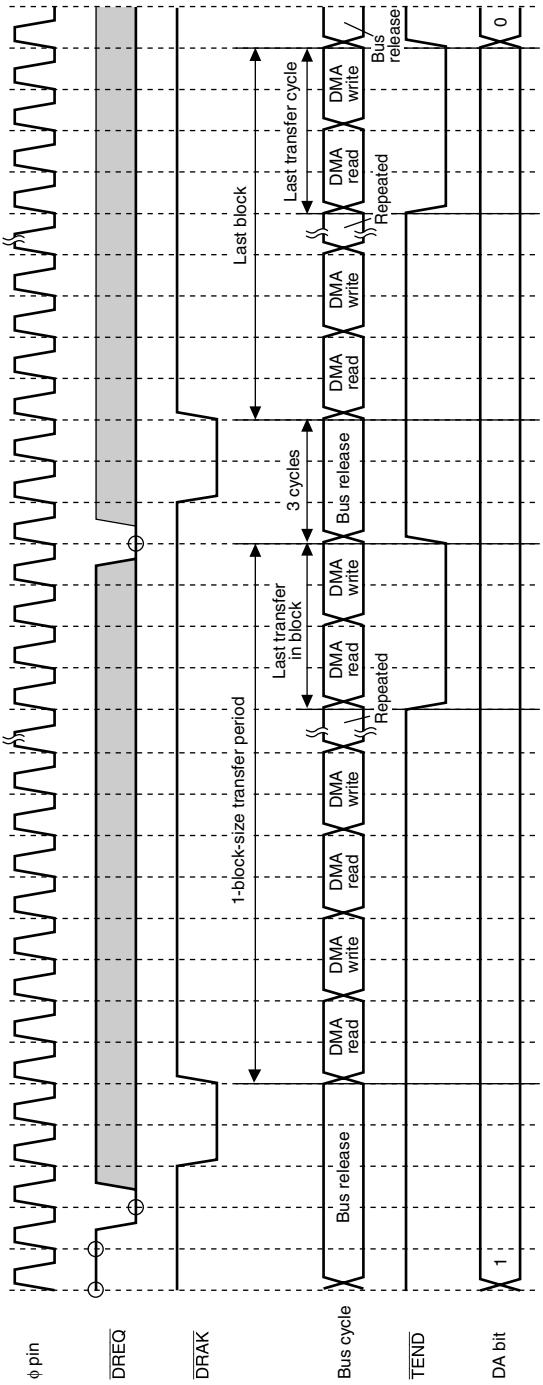


Figure 7.42 External Request/Cycle Steal Mode/Block Transfer Mode (No Contention/Dual Address Mode/Low Level Sensing)

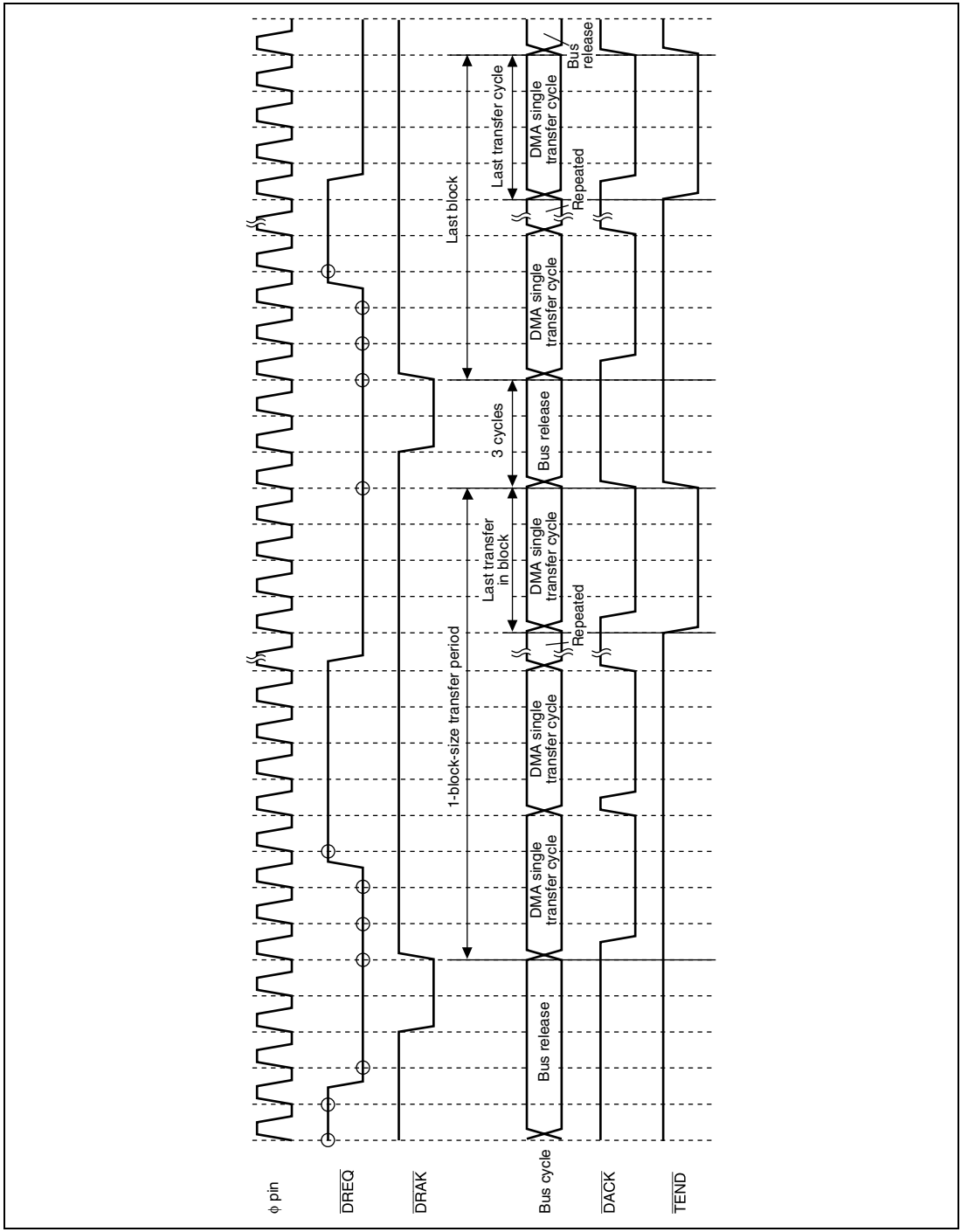


Figure 7.43 External Request/Cycle Steal Mode/Block Transfer Mode (No Contention/Single Address Mode/Falling Edge Sensing)

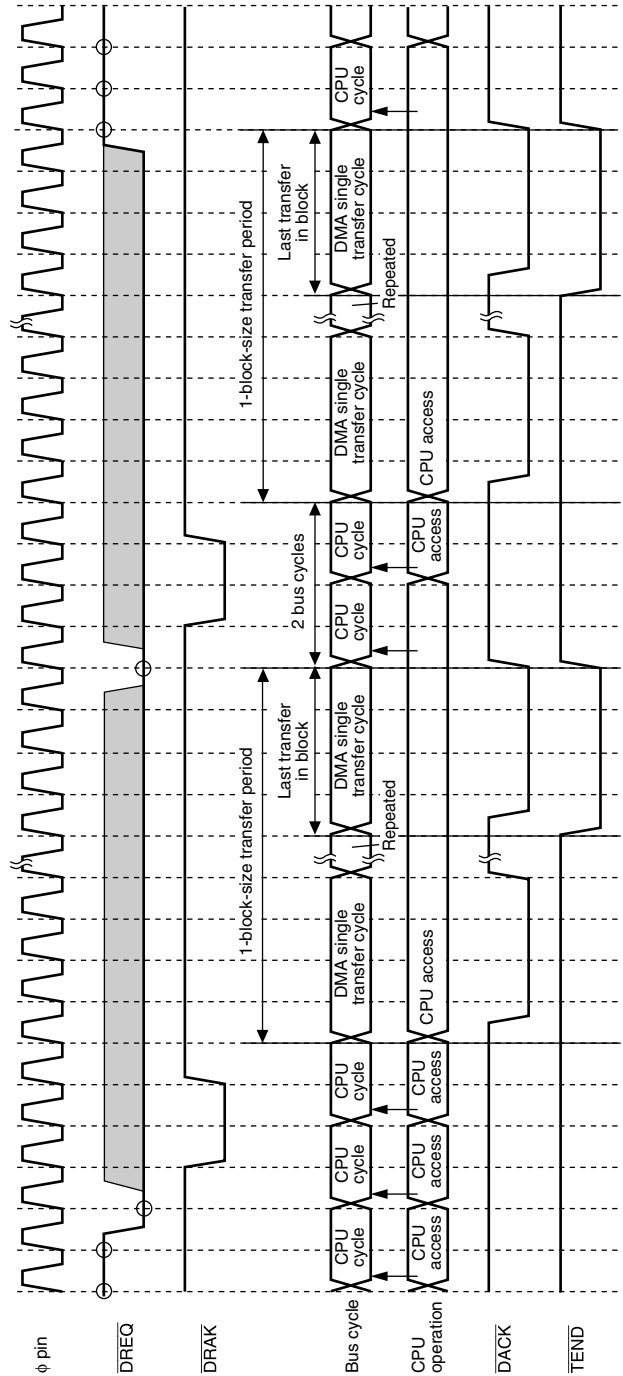


Figure 7.44 External Request/Cycle Steal Mode/Block Transfer Mode (CPU Cycles/Single Address Mode/Low Level Sensing)

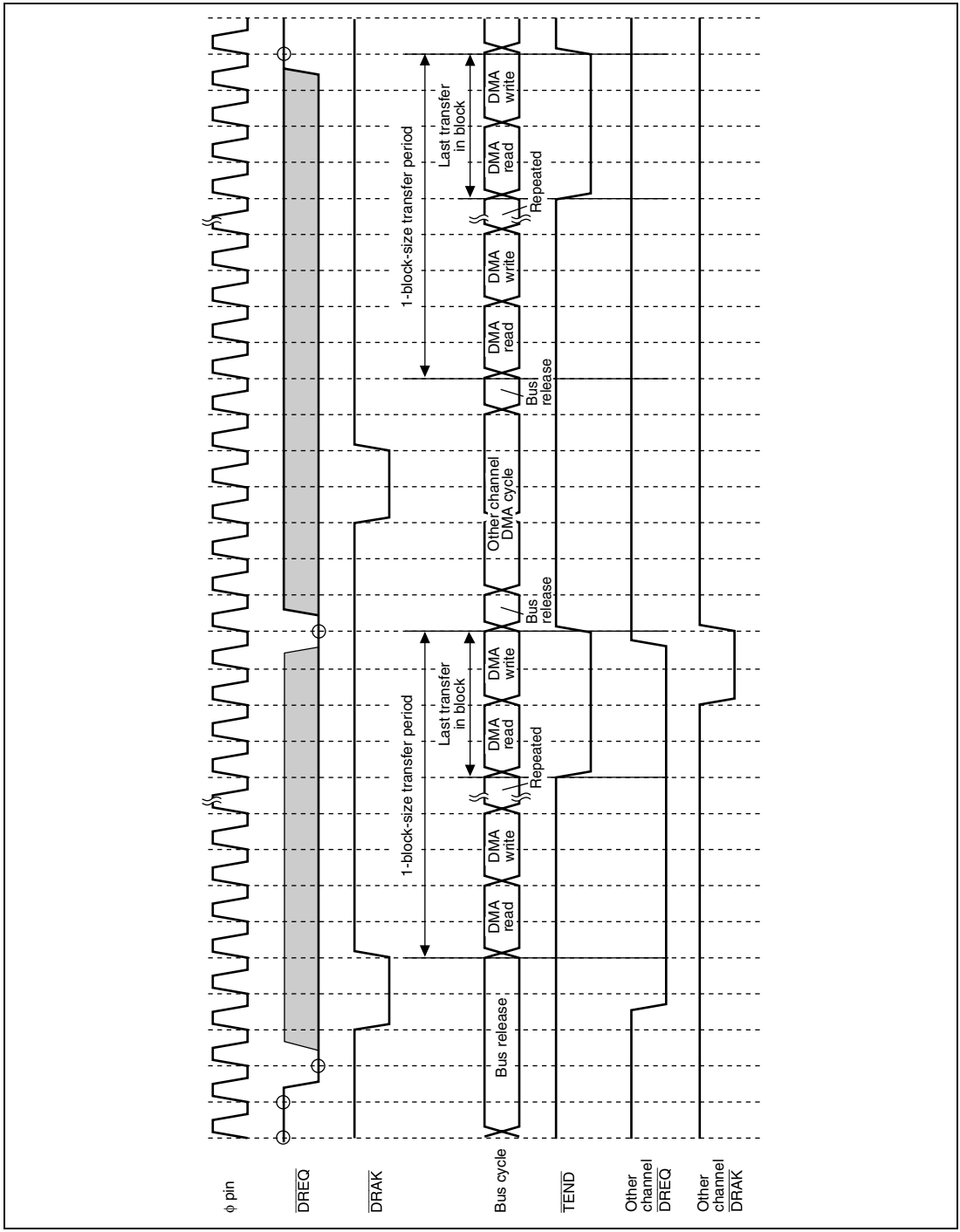


Figure 7.45 External Request/Cycle Steal Mode/Block Transfer Mode (Contention with Another Channel/Dual Address Mode/Low Level Sensing)

7.4.12 Ending DMA Transfer

The operation for ending DMA transfer depends on the transfer end conditions. When DMA transfer ends, the DA bit in DMMDR changes from 1 to 0, indicating that DMA transfer has ended.

Transfer End by 1 → 0 Transition of DMTCR: When the value of DMTCR changes from 1 to 0, DMA transfer ends on the corresponding channel and the DA bit in DMMDR is cleared to 0. If the TCEIE bit in DMMDR is set at this time, a transfer end interrupt request is generated by the transfer counter and the IRF bit in DMMDR is set to 1.

In block transfer mode, DMA transfer ends when the value of bits 15 to 0 in DMTCR changes from 1 to 0.

DMA transfer does not end if the DMTCR value has been 0 since before the start of transfer.

Transfer End by Repeat Area Overflow Interrupt: If an address overflows the repeat area when a repeat area specification has been made and repeat interrupts have been enabled (with the SARIE or DARIE bit in DMACR), a repeat area overflow interrupt is requested. DMA transfer ends, the DA bit in DMMDR is cleared to 0, and the IRF bit in DMMDR is set to 1.

In dual address mode, if a repeat area overflow interrupt is requested during a read cycle, the following write cycle processing is still executed.

In block transfer mode, if a repeat area overflow interrupt is requested during transfer of a block, transfer continues to the end of the block. Transfer end by means of a repeat area overflow interrupt occurs between block-size transfers.

Transfer End by 0-Write to DA Bit in DMMDR: When 0 is written to the DA bit in DMMDR by the CPU, etc., transfer ends after completion of the DMA cycle in which transfer is in progress or a transfer request was accepted.

In block transfer mode, DMA transfer halts after completion of one-block-size transfer.

The DA bit in DMMDR is not cleared to 0 until all transfer processing has ended. Up to that point, the value of the DA bit will be read as 1.

Transfer Abort by NMI Interrupt: DMA transfer is aborted when an NMI interrupt is generated. The DA bit is cleared to 0 in all channels. In external request mode, DMA transfer is performed for all transfer requests for which $\overline{\text{DRAK}}$ has been output. In dual address mode, processing is executed for the write cycle following the read cycle.

In block transfer mode, operation is aborted even in the middle of a block-size transfer. As the transfer is halted midway through a block, the BEF bit in DMMDR is set to 1 to indicate that the block transfer was not carried out normally.

When transfer is aborted, register values are retained, and as the address registers indicate the next transfer addresses, transfer can be resumed by setting the DA bit to 1 in DMMDR. If the BEF bit is 1 in DMMDR, transfer can be resumed from midway through a block.

Hardware Standby Mode and Reset Input: The DMAC is initialized in hardware standby mode and by a reset. DMA transfer is not guaranteed in these cases.

7.4.13 Relationship between DMAC and Other Bus Masters

The read and write operations in a DMA transfer cycle are indivisible, and a refresh cycle or internal bus master (CPU) access cycle never occurs between the two.

When read and write cycles occur consecutively, as in burst transfer or block transfer, a refresh may be inserted after the write cycle. As the CPU has lower priority than the DMAC, the CPU access is not executed until the DMAC releases the bus.

The DMAC releases the bus in the following cases:

1. When DMA transfer is performed in cycle steal mode
2. When switching to a different channel
3. When transfer ends in burst transfer mode
4. When transfer of one block ends in block transfer mode

7.5 Interrupt Sources

DMAC interrupt sources are a transfer end indicated by the transfer counter, and repeat area overflow interrupts. Table 7.4 shows the interrupt sources and their priority.

Table 7.4 Interrupt Sources and Priority Order

Interrupt	Interrupt source	Interrupt Priority
DMTEND0	Transfer end indicated by channel 0 transfer counter Channel 0 source address repeat area overflow Channel 0 destination address repeat area overflow	High
DMTEND1	Transfer end indicated by channel 1 transfer counter Channel 1 source address repeat area overflow Channel 1 destination address repeat area overflow	
DMTEND2	Transfer end indicated by channel 2 transfer counter Channel 2 source address repeat area overflow Channel 2 destination address repeat area overflow	
DMTEND3	Transfer end indicated by channel 3 transfer counter Channel 3 source address repeat area overflow Channel 3 destination address repeat area overflow	Low

Interrupt sources can be enabled or disabled by means of the DIE bit in DMMDR for the relevant channel, and can be sent to the interrupt controller independently.

The relative priority of the channels is determined by the interrupt controller (see table 7.4). Figure 7.46 shows the transfer end interrupt logic. A transfer end interrupt is generated whenever the DIE bit is set to 1 while the IRF bit is set to 1 in DMMDR.

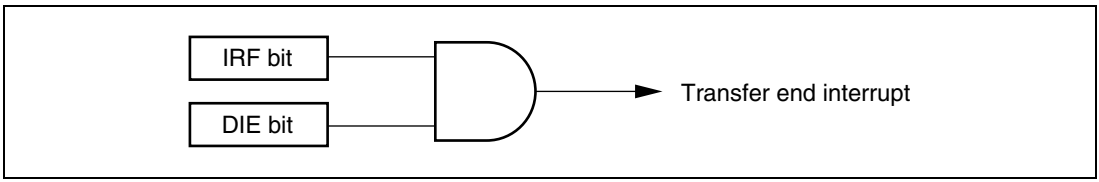
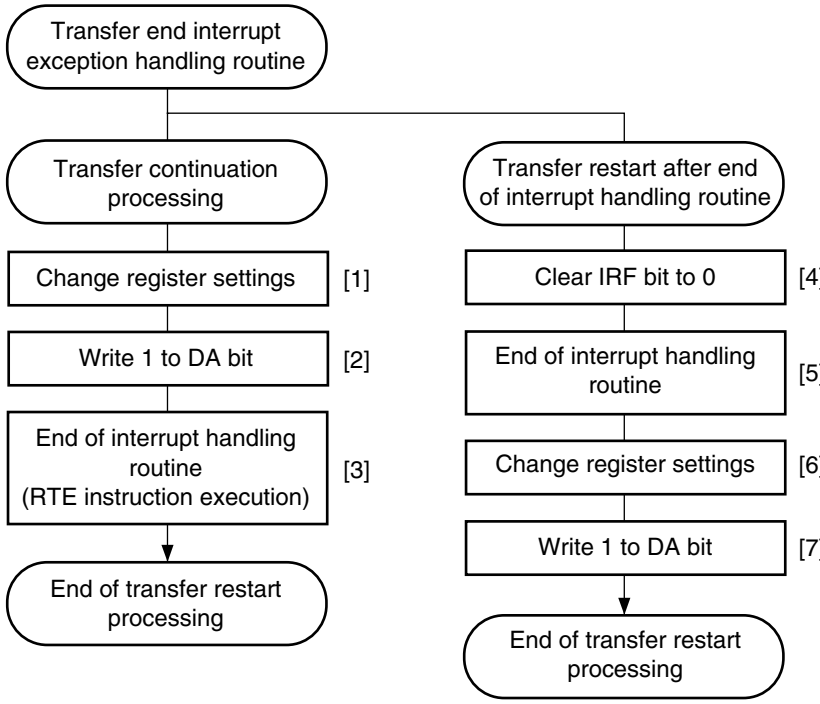


Figure 7.46 Transfer End Interrupt Logic

Interrupt source settings are made individually with the interrupt enable bits in the registers for the relevant channels. The transfer counter's transfer end interrupt is enabled or disabled by means of the TCEIE bit in DMMDR, the source address register repeat area overflow interrupt by means of the SARIE bit in DMACR, and the destination address register repeat area overflow interrupt by means of the DARIE bit in DMACR. When an interrupt source occurs while the corresponding interrupt enable bit is set to 1, the IRF bit in DMMDR is set to 1. The IRF bit is set by all interrupt sources indiscriminately.

The transfer end interrupt can be cleared either by clearing the IRF bit to 0 in DMMDR within the interrupt handling routine, or by re-setting the transfer counter and address registers and then setting the DA bit to 1 in DMMDR to perform transfer continuation processing.

An example of the procedure for clearing the transfer end interrupt and restarting transfer is shown in figure 7.47.



[1] Write set values to the registers (transfer counter, address registers, etc.).

[2] Write 1 to the DA bit in DMMDR to restart DMA operation. When 1 is written to the DA bit, the IRF bit in DMMDR is automatically cleared to 0 and the interrupt source is cleared.

[3] The interrupt handling routine is ended with an RTE instruction, etc.

[4] Clear the IRF bit to 0 in DMMDR by first reading 1 from it, then writing 0.

[5] After the interrupt handling routine is ended with an RTE instruction, etc., interrupt masking is cleared.

[6] Write set values to the registers (transfer counter, address registers, etc.).

[7] Write 1 to the DA bit in DMMDR to restart DMA operation.

Figure 7.47 Example of Procedure for Restarting Transfer on Channel in which Transfer End Interrupt Occurred

7.6 Usage Notes

DMAC Register Access during Operation: Except for clearing the DA bit to 0 in DMMDR, settings should not be changed for a channel in operation (including the transfer standby state). Transfer must be disabled before changing a setting for an operational channel.

Module Stop State: When the DMACKSTP bit is set to 1 in MSTPCR, the DMAC clock stops and the DMAC enters the module stop state. However, 1 cannot be written to the DMACKSTP bit when any of the DMAC's channels is enabled for transfer, or when an interrupt is being requested. Before setting the DMACKSTP bit, first clear the DA bit in DMMDR to 0, then clear the IRF or DIE bit in DMMDR to 0.

When the DMAC clock stops, DMAC registers can no longer be accessed. The following DMAC register settings remain valid in the module stop state, and so should be changed, if necessary, before making the module stop transition.

- TENDE = 1 in DMMDR ($\overline{\text{TEND}}$ pin enable)
- DRAKE = 1 in DMMDR ($\overline{\text{DRAK}}$ pin enable)
- AMS = 1 in DMMDR ($\overline{\text{DACK}}$ pin enable)

$\overline{\text{DREQ}}$ Pin Falling Edge Activation: Falling edge sensing on the $\overline{\text{DREQ}}$ pin is performed in synchronization with DMAC internal operations, as indicated below.

[1] Activation request standby state: Waits for low level sensing on $\overline{\text{DREQ}}$ pin, then goes to [2].

[2] Transfer standby state: Waits for DMAC data transfer to become possible, then goes to [3].

[3] Activation request disabled state: Waits for high level sensing on $\overline{\text{DREQ}}$ pin, then goes to [1].

After DMAC transfer is enabled, the DMAC goes to state [1], so low level sensing is used for the initial activation after transfer is enabled.

Activation Source Acceptance: At the start of activation source acceptance, low level sensing is used for both falling edge sensing and low level sensing on the $\overline{\text{DREQ}}$ pin. Therefore, a request is accepted in the case of a low level at the $\overline{\text{DREQ}}$ pin that occurs before execution of the DMMDR write for setting the transfer-enabled state.

When the DMAC is activated, make sure, if necessary, that a low level does not remain at the $\overline{\text{DREQ}}$ pin from the previous end of transfer, etc.

Enabling Interrupt Requests when IRF = 1 in DMMDR: When transfer is started while the IRF bit is set to 1 in DMMDR, if the DIE bit is set to 1 in DMMDR together with the DA bit in DMMDR, enabling interrupt requests, an interrupt will be requested since DIE = 1 and IRF = 1. To prevent the occurrence of an erroneous interrupt request when transfer starts, ensure that the IRF bit is cleared to 0 before the DIE bit is set to 1.

Source Address/Destination Address: When transfer data size is specified as word or longword, an even (word) value or value multiplied by four (longword) must be set in DMSAR and DMDAR.

When transfer data size is specified as word and an odd value is set in DMSAR or DMDAR, the lowest bit of the address is regarded as 0 and accessed.

When transfer data size is specified as longword and a value not multiplied by four is set in DMSAR or DMDAR, the lowest two bits of the address are regarded as 0 and accessed.

Section 8 I/O Ports

Table 8.1 summarizes the port functions. The pins of each port also have other functions such as input/output or interrupt input pins of on-chip peripheral modules. Each I/O port includes a data direction register (DDR) that controls input/output, a data register (DR) that stores output data, and a port register (PORT) used to read the pin states.

Ports 1 to 4 can drive a single TTL load and 30 pF capacity load. Ports 5 to 9, and A can drive a single TTL load and 50 pF capacity load.

All of the I/O ports can drive a Darlington transistor when outputting data.

Ports 3 (P34, P35), 5 (P52, P53, P56, P57), 7, and A (PA0, PA1) are Schmitt-triggered inputs when used as the IRQ input.

Port 3 (P32, P33) is a Schmitt-triggered input when the software standby state is entered by using the USB.

Table 8.1 Port Functions

Port	Description	Function
Port 1	General I/O port also functioning as USB2 I/O	P17/USD15
		P16/USD14
		P15/USD13
		P14/USD12
		P13/USD11
		P12/USD10
		P11/USD9
		P10/USD8
Port 2	General I/O port also functioning as USB2 I/O and DMAC I/O	P27/USD7/$\overline{\text{DRAK1}}$
		P26/USD6/$\overline{\text{DACK1}}$
		P25/USD5/$\overline{\text{TEND1}}$
		P24/USD4/$\overline{\text{DREQ1}}$
		P23/USD3/$\overline{\text{DRAK0}}$
		P22/USD2/$\overline{\text{DACK0}}$
		P21/USD1/$\overline{\text{TEND0}}$
		P20/USD0/$\overline{\text{DREQ0}}$
Port 3	General I/O port also functioning as USB2 I/O and interrupt input	P37/USOPM1
		P36/USOPM0
		P35/($\overline{\text{IRQ5}}$)/VBUS
		P34/($\overline{\text{IRQ4}}$)/USTXV
		P33/USLSTA1
		P32/USLSTA0
		P31/USCLK
		P30/USWDVLD
Port 4	General I/O port also functioning as USB2 I/O	P47/USXCVRS
		P46/USTXRDY
		P45/USTSEL
		P44/$\overline{\text{USSUSP}}$
		P43/USRXV
		P42/USRXERR
		P41/USRXACT
		P40/$\overline{\text{USRST}}$

Port	Description	Function
Port 5	General I/O port also functioning as external data bus I/O, DMAC I/O, and interrupt input	P57/D7/($\overline{\text{IRQ7}}$)/$\overline{\text{DRAK3}}$
		P56/D6/($\overline{\text{IRQ6}}$)/$\overline{\text{DACK3}}$
		P55/D5/$\overline{\text{TEND3}}$
		P54/D4/$\overline{\text{DREQ3}}$
		P53/D3/($\overline{\text{IRQ3}}$)/$\overline{\text{DRAK2}}$
		P52/D2/($\overline{\text{IRQ2}}$)/$\overline{\text{DACK2}}$
		P51/D1/$\overline{\text{TEND2}}$
		P50/D0/$\overline{\text{DREQ2}}$
Port 6	General I/O port also functioning as external address bus output and timer I/O	P67/A15
		P66/A14/TMO1
		P65/A13/TMRI1
		P64/A12/TMCI1
		P63/A11
		P62/A10/TMO0
		P61/A9/TMRI0
		P60/A8/TMCI0
Port 7	General I/O port also functioning as external address bus output and interrupt input	P77/A7/$\overline{\text{IRQ7}}$
		P76/A6/$\overline{\text{IRQ6}}$
		P75/A5/$\overline{\text{IRQ5}}$
		P74/A4/$\overline{\text{IRQ4}}$
		P73/A3/$\overline{\text{IRQ3}}$
		P72/A2/$\overline{\text{IRQ2}}$
		P71/A1/$\overline{\text{IRQ1}}$
		P70/A0/$\overline{\text{IRQ0}}$
Port 8	General I/O port also functioning as external data bus I/O	P87/D15
		P86/D14
		P85/D13
		P84/D12
		P83/D11
		P82/D10
		P81/D9
		P80/D8

Port	Description	Function
Port 9	General I/O port also functioning as external bus control output	P97/φ
		P96/$\overline{AS}$
		P95/$\overline{RD}$
		P94/$\overline{HWR}$
		P93/$\overline{LWR}$
		P92/$\overline{CS2}$/$\overline{RAS}$
		P91/$\overline{CS1}$
		P90/$\overline{CS0}$
Port A	General I/O port also functioning as external address bus output, external bus control output, and interrupt input	PA3/A19/$\overline{CS3}$
		PA2/A18/$\overline{UCAS}$
		PA1/A17/$\overline{LCAS}$/($\overline{IRQ1}$)
		PA0/A16/($\overline{IRQ0}$)

8.1 Port 1

Port 1 is an 8-bit I/O port that also has other functions. The port 1 has the following registers.

- Port 1 data direction register (P1DDR)
- Port 1 data register (P1DR)
- Port 1 register (PORT1)

8.1.1 Port 1 Data Direction Register (P1DDR)

The individual bits of P1DDR specify input or output for the pins of port 1. P1DDR cannot be read; if it is, an undefined value will be read.

Bit	Bit Name	Initial Value	R/W	Description
7	P17DDR	0	W	When a pin function is specified to a general purpose I/O, setting this bit to 1 makes the corresponding port 1 pin an output pin, while clearing this bit to 0 makes the pin an input pin.
6	P16DDR	0	W	
5	P15DDR	0	W	
4	P14DDR	0	W	
3	P13DDR	0	W	
2	P12DDR	0	W	
1	P11DDR	0	W	
0	P10DDR	0	W	

8.1.2 Port 1 Data Register (P1DR)

P1DR stores output data for the port 1 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P17DR	0	R/W	Output data for a pin is stored when the pin function is specified to a general purpose output.
6	P16DR	0	R/W	
5	P15DR	0	R/W	
4	P14DR	0	R/W	
3	P13DR	0	R/W	
2	P12DR	0	R/W	
1	P11DR	0	R/W	
0	P10DR	0	R/W	

8.1.3 Port 1 Register (PORT1)

PORT1 shows the pin states of the port 1. PORT1 cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	P17	—*	R	If a PORT1 read is performed while P1DDR bits are set to 1, the P1DR values are read. If a PORT1 read is performed while P1DDR bits are cleared to 0, the pin states are read.
6	P16	—*	R	
5	P15	—*	R	
4	P14	—*	R	
3	P13	—*	R	
2	P12	—*	R	
1	P11	—*	R	
0	P10	—*	R	

Note: * Determined by the states of pins P17 to P10.

8.1.4 Pin Functions

Port 1 pins also function as the pins for USB I/O. The correspondence between the register specification and the pin functions is shown below.

- P17/USD15

  The pin function is switched as shown below according to the combination of the P17DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P17DDR	0	1	—
Pin function	P17 input	P17 output	USD15 I/O

- P16/USD14

  The pin function is switched as shown below according to the combination of the P16DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P16DDR	0	1	—
Pin function	P16 input	P16 output	USD14 I/O

- P15/USD13

  The pin function is switched as shown below according to the combination of the P15DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P15DDR	0	1	—
Pin function	P15 input	P15 output	USD13 I/O

- P14/USD12

  The pin function is switched as shown below according to the combination of the P14DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P14DDR	0	1	—
Pin function	P14 input	P14 output	USD12 I/O

- P13/USD11

  The pin function is switched as shown below according to the combination of the P13DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P13DDR	0	1	—
Pin function	P13 input	P13 output	USD11 I/O

- P12/USD10

  The pin function is switched as shown below according to the combination of the P12DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P12DDR	0	1	—
Pin function	P12 input	P12 output	USD10 I/O

- P11/USD9

  The pin function is switched as shown below according to the combination of the P11DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P11DDR	0	1	—
Pin function	P11 input	P11 output	USD9 I/O

- P10/USD8

  The pin function is switched as shown below according to the combination of the P10DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P10DDR	0	1	—
Pin function	P10 input	P10 output	USD8 I/O

8.2 Port 2

Port 2 is an 8-bit I/O port that also has other functions. The port 2 has the following registers.

- Port 2 data direction register (P2DDR)
- Port 2 data register (P2DR)
- Port 2 register (PORT2)

8.2.1 Port 2 Data Direction Register (P2DDR)

The individual bits of P2DDR specify input or output for the pins of port 2. P2DDR cannot be read; if it is, an undefined value will be read.

Bit	Bit Name	Initial Value	R/W	Description
7	P27DDR	0	W	When a pin function is specified to a general purpose I/O, setting this bit to 1 makes the corresponding port 2 pin an output pin, while clearing this bit to 0 makes the pin an input pin.
6	P26DDR	0	W	
5	P25DDR	0	W	
4	P24DDR	0	W	
3	P23DDR	0	W	
2	P22DDR	0	W	
1	P21DDR	0	W	
0	P20DDR	0	W	

8.2.2 Port 2 Data Register (P2DR)

P2DR stores output data for the port 2 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P27DR	0	R/W	Output data for a pin is stored when the pin function is specified to a general purpose output.
6	P26DR	0	R/W	
5	P25DR	0	R/W	
4	P24DR	0	R/W	
3	P23DR	0	R/W	
2	P22DR	0	R/W	
1	P21DR	0	R/W	
0	P20DR	0	R/W	

8.2.3 Port 2 Register (PORT2)

PORT2 shows the pin states of the port 2. PORT2 cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	P27	—*	R	If a PORT2 read is performed while P2DDR bits are set to 1, the P2DR values are read. If a PORT2 read is performed while P2DDR bits are cleared to 0, the pin states are read.
6	P26	—*	R	
5	P25	—*	R	
4	P24	—*	R	
3	P23	—*	R	
2	P22	—*	R	
1	P21	—*	R	
0	P20	—*	R	

Note: * Determined by the states of pins P27 to P20.

8.2.4 Pin Functions

Port 2 pins also function as the pins for DMAC I/O and USB I/O. The correspondence between the register specification and the pin functions is shown below.

- P27/USD7/$\overline{\text{DRAK1}}$

  The pin function is switched as shown below according to the combination of the P27DDR bit, the DRAKE bit in DMMDR_1, and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1			0
DRAKE	0		1	—
P27DDR	0	1	—	—
Pin function	P27 input	P27 output	$\overline{\text{DRAK1}}$ output	USD7 I/O

- P26/USD6/$\overline{\text{DACK1}}$

  The pin function is switched as shown below according to the combination of the P26DDR bit, the AMS bit in DMMDR_1, the EXPE bit in MDCR, and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1					0
EXPE	0		1			—
AMS	—		0		1	—
P26DDR	0	1	0	1	—	—
Pin function	P26 input	P26 output	P26 input	P26 output	$\overline{\text{DACK1}}$ output	USD6 I/O

- P25/USD5/$\overline{\text{TEND1}}$

  The pin function is switched as shown below according to the combination of the P25DDR bit, the TENDE bit in DMMDR_1, and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1			0
TENDE	0		1	—
P25DDR	0	1	—	—
Pin function	P25 input	P25 output	$\overline{\text{TEND1}}$ output	USD5 I/O

- P24/USD4/$\overline{\text{DREQ1}}$

  The pin function is switched as shown below according to the combination of the P24DDR bit, the DREQS bit in DMMDR_1, and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P24DDR	0	1	—
Pin function	P24 input	P24 output	USD4 I/O
	$\overline{\text{DREQ1}}$ input*		

Note: * When DREQS = 1, this pin functions as $\overline{\text{DREQ1}}$ input.

- P23/USD3/$\overline{\text{DRAK0}}$

  The pin function is switched as shown below according to the combination of the P23DDR bit, the DRAKE bit in DMMDR_0, and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1			0
DRAKE	0		1	—
P23DDR	0	1	—	—
Pin function	P23 input	P23 output	$\overline{\text{DRAK0}}$ output	USD3 I/O

- P22/USD2/$\overline{\text{DACK0}}$

  The pin function is switched as shown below according to the combination of the P22DDR bit, the AMS bit in DMMDR_0, the EXPE bit in MDCR, and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1					0
EXPE	0		1			—
AMS	—		0		1	—
P22DDR	0	1	0	1	—	—
Pin function	P22 input	P22 output	P22 input	P22 output	$\overline{\text{DACK0}}$ output	USD2 I/O

- P21/USD1/$\overline{\text{TEND0}}$

  The pin function is switched as shown below according to the combination of the P21DDR bit, the TENDE bit in DMMDR_0, and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1			0
TENDE	0		1	—
P21DDR	0	1	—	—
Pin function	P21 input	P21 output	$\overline{\text{TEND0}}$ output	USD1 I/O

- P20/USD0/$\overline{\text{DREQ0}}$

  The pin function is switched as shown below according to the combination of the P20DDR bit, the DREQS bit in DMMDR_0, and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P20DDR	0	1	—
Pin function	P20 input	P20 output	USD0 I/O
	$\overline{\text{DREQ0}}$ input*		

Note: * When DREQS = 1, this pin functions as $\overline{\text{DREQ0}}$ input.

8.3 Port 3

Port 3 is an 8-bit I/O port that also has other functions. The port 3 has the following registers.

- Port 3 data direction register (P3DDR)
- Port 3 data register (P3DR)
- Port 3 register (PORT3)

8.3.1 Port 3 Data Direction Register (P3DDR)

The individual bits of P3DDR specify input or output for the pins of port 3. P3DDR cannot be read; if it is, an undefined value will be read.

Bit	Bit Name	Initial Value	R/W	Description
7	P37DDR	0	W	When a pin function is specified to a general purpose I/O, setting this bit to 1 makes the corresponding port 3 pin an output pin, while clearing this bit to 0 makes the pin an input pin.
6	P36DDR	0	W	
5	P35DDR	0	W	
4	P34DDR	0	W	
3	P33DDR	0	W	
2	P32DDR	0	W	
1	P31DDR	0	W	
0	P30DDR	0	W	

8.3.2 Port 3 Data Register (P3DR)

P3DR stores output data for the port 3 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P37DR	0	R/W	Output data for a pin is stored when the pin function is specified to a general purpose output.
6	P36DR	0	R/W	
5	P35DR	0	R/W	
4	P34DR	0	R/W	
3	P33DR	0	R/W	
2	P32DR	0	R/W	
1	P31DR	0	R/W	
0	P30DR	0	R/W	

8.3.3 Port 3 Register (PORT3)

PORT3 shows the pin states of the port 3. PORT3 cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	P37	—*	R	If a PORT3 read is performed while P3DDR bits are set to 1, the P3DR values are read. If a PORT3 read is performed while P3DDR bits are cleared to 0, the pin states are read.
6	P36	—*	R	
5	P35	—*	R	
4	P34	—*	R	
3	P33	—*	R	
2	P32	—*	R	
1	P31	—*	R	
0	P30	—*	R	

Note: * Determined by the states of pins P37 to P30.

8.3.4 Pin Functions

Port 3 pins also function as the pins for USB I/O and interrupt input. The correspondence between the register specification and the pin functions is shown below.

- P37/USOPM1

  The pin function is switched as shown below according to the combination of the P37DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P37DDR	0	1	—
Pin function	P37 input	P37 output	USOPM1 output

- P36/USOPM0

  The pin function is switched as shown below according to the combination of the P36DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P36DDR	0	1	—
Pin function	P36 input	P36 output	USOPM0 output

- P35/($\overline{\text{IRQ5}}$)/USVBUS

  The pin function is switched as shown below according to the combination of the P35DDR bit, the USBCKSTP bit in MSTPCRL, and the ISS5 bit in ISSR.

USBCKSTP	1		0
P35DDR	0	1	—
Pin function	P35 input	P35 output	USVBUS input
	$\overline{\text{IRQ5}}$ interrupt input*		

Note: * When ISS5 = 1, this pin functions as $\overline{\text{IRQ5}}$ interrupt input.

- P34/($\overline{\text{IRQ4}}$)/USTXV

  The pin function is switched as shown below according to the combination of the P34DDR bit, the USBCKSTP bit in MSTPCRL, and the ISS4 bit in ISSR.

USBCKSTP	1		0
P34DDR	0	1	—
Pin function	P34 input	P34 output	USTXV output
	$\overline{\text{IRQ4}}$ interrupt input*		

Note: * When ISS4 = 1, this pin functions as $\overline{\text{IRQ4}}$ interrupt input.

- P33/USLSTA1

  The pin function is switched as shown below according to the combination of the P33DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P33DDR	0	1	—
Pin function	P33 input	P33 output	USLSTA1 input

- P32/USLSTA0

  The pin function is switched as shown below according to the combination of the P32DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P32DDR	0	1	—
Pin function	P32 input	P32 output	USLSTA0 input

- P31/USCLK

  The pin function is switched as shown below according to the combination of the P31DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P31DDR	0	1	—
Pin function	P31 input	P31 output	USCLK input

- P30/USWDVLD

  The pin function is switched as shown below according to the combination of the P30DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P30DDR	0	1	—
Pin function	P30 input	P30 output	USWDVLD I/O

8.4 Port 4

Port 4 is an 8-bit I/O port that also has other functions. The port 4 has the following registers.

- Port 4 data direction register (P4DDR)
- Port 4 data register (P4DR)
- Port 4 register (PORT4)

8.4.1 Port 4 Data Direction Register (P4DDR)

The individual bits of P4DDR specify input or output for the pins of port 4. P4DDR cannot be read; if it is, an undefined value will be read.

Bit	Bit Name	Initial Value	R/W	Description
7	P47DDR	0	W	When a pin function is specified to a general purpose I/O, setting this bit to 1 makes the corresponding port 4 pin an output pin, while clearing this bit to 0 makes the pin an input pin.
6	P46DDR	0	W	
5	P45DDR	0	W	
4	P44DDR	0	W	
3	P43DDR	0	W	
2	P42DDR	0	W	
1	P41DDR	0	W	
0	P40DDR	0	W	

8.4.2 Port 4 Data Register (P4DR)

P4DR stores output data for the port 4 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P47DR	0	R/W	Output data for a pin is stored when the pin function is specified to a general purpose output.
6	P46DR	0	R/W	
5	P45DR	0	R/W	
4	P44DR	0	R/W	
3	P43DR	0	R/W	
2	P42DR	0	R/W	
1	P41DR	0	R/W	
0	P40DR	0	R/W	

8.4.3 Port 4 Register (PORT4)

PORT4 shows the pin states of the port 4. PORT4 cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	P47	—*	R	If a PORT4 read is performed while P4DDR bits are set to 1, the P4DR values are read. If a PORT4 read is performed while P4DDR bits are cleared to 0, the pin states are read.
6	P46	—*	R	
5	P45	—*	R	
4	P44	—*	R	
3	P43	—*	R	
2	P42	—*	R	
1	P41	—*	R	
0	P40	—*	R	

Note: * Determined by the states of pins P47 to P40.

8.4.4 Pin Functions

Port 4 pins also function as the pins for USB I/O. The correspondence between the register specification and the pin functions is shown below.

- P47/USXCVRS

  The pin function is switched as shown below according to the combination of the P47DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P47DDR	0	1	—
Pin function	P47 input	P47 output	USXCVRS output

- P46/USTXRDY

  The pin function is switched as shown below according to the combination of the P46DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P46DDR	0	1	—
Pin function	P46 input	P46 output	USTXRDY input

- P45/USTSEL

  The pin function is switched as shown below according to the combination of the P45DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P45DDR	0	1	—
Pin function	P45 input	P45 output	USTSEL output

- P44/$\overline{\text{USSUSP}}$

  The pin function is switched as shown below according to the combination of the P44DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P44DDR	0	1	—
Pin function	P44 input	P44 output	$\overline{\text{USSUSP}}$ output

- P43/USRXV

  The pin function is switched as shown below according to the combination of the P43DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P43DDR	0	1	—
Pin function	P43 input	P43 output	USRXV input

- P42/USRXERR

  The pin function is switched as shown below according to the combination of the P42DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P42DDR	0	1	—
Pin function	P42 input	P42 output	USRXERR input

- P41/USRXACT

  The pin function is switched as shown below according to the combination of the P41DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P41DDR	0	1	—
Pin function	P41 input	P41 output	USRXACT input

- P40/$\overline{\text{USRST}}$

  The pin function is switched as shown below according to the combination of the P40DDR bit and the USBCKSTP bit in MSTPCRL.

USBCKSTP	1		0
P40DDR	0	1	—
Pin function	P40 input	P40 output	$\overline{\text{USRST}}$ output

8.5 Port 5

Port 5 is an 8-bit I/O port that also has other functions. The port 5 has the following registers.

- Port 5 data direction register (P5DDR)
- Port 5 data register (P5DR)
- Port 5 register (PORT5)

8.5.1 Port 5 Data Direction Register (P5DDR)

The individual bits of P5DDR specify input or output for the pins of port 5. P5DDR cannot be read; if it is, an undefined value will be read.

Bit	Bit Name	Initial Value	R/W	Description
7	P57DDR	0	W	When a pin function is specified to a general purpose I/O, setting this bit to 1 makes the corresponding port 5 pin an output pin, while clearing this bit to 0 makes the pin an input pin.
6	P56DDR	0	W	
5	P55DDR	0	W	
4	P54DDR	0	W	
3	P53DDR	0	W	
2	P52DDR	0	W	
1	P51DDR	0	W	
0	P50DDR	0	W	

8.5.2 Port 5 Data Register (P5DR)

P5DR stores output data for the port 5 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P57DR	0	R/W	Output data for a pin is stored when the pin function is specified to a general purpose output.
6	P56DR	0	R/W	
5	P55DR	0	R/W	
4	P54DR	0	R/W	
3	P53DR	0	R/W	
2	P52DR	0	R/W	
1	P51DR	0	R/W	
0	P50DR	0	R/W	

8.5.3 Port 5 Register (PORT5)

PORT5 shows the pin states of the port 5. PORT5 cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	P57	—*	R	If a PORT5 read is performed while P5DDR bits are set to 1, the P5DR values are read. If a PORT5 read is performed while P5DDR bits are cleared to 0, the pin states are read.
6	P56	—*	R	
5	P55	—*	R	
4	P54	—*	R	
3	P53	—*	R	
2	P52	—*	R	
1	P51	—*	R	
0	P50	—*	R	

Note: * Determined by the states of pins P57 to P50.

8.5.4 Pin Functions

Port 5 pins also function as the pins for external data bus I/O, DMAC I/O, and interrupt input. The correspondence between the register specification and the pin functions is shown below.

- P57/D7/($\overline{\text{IRQ7}}$)/$\overline{\text{DRAK3}}$

  The pin function is switched as shown below according to the combination of the P57DDR bit, the DRAKE bit in DMMDR_3, the EXPE bit in MDCR, and the ISS7 bit in ISSR.

EXPE	0		1			
Bus mode	—		All areas are 8-bit space			At least one area is 16-bit space
DRAKE	0		0		1	—
P57DDR	0	1	0	1	—	—
Pin function	P57 input	P57 output	P57 input	P57 output	$\overline{\text{DRAK3}}$ output	D7 I/O
	$\overline{\text{IRQ7}}$ interrupt input*					

Note: * When ISS7 = 1, this pin functions as $\overline{\text{IRQ7}}$ interrupt input.

- P56/D6/($\overline{\text{IRQ6}}$)/$\overline{\text{DACK3}}$

  The pin function is switched as shown below according to the combination of the P56DDR bit, the AMS bit in DMMDR_3, the EXPE bit in MDCR, and the ISS6 bit in ISSR.

EXPE	0		1			
Bus mode	—		All areas are 8-bit space			At least one area is 16-bit space
AMS	0		0		1	—
P56DDR	0	1	0	1	—	—
Pin function	P56 input	P56 output	P56 input	P56 output	$\overline{\text{DACK3}}$ output	D6 I/O
	$\overline{\text{IRQ6}}$ interrupt input*					

Note: * When ISS6 = 1, this pin functions as $\overline{\text{IRQ6}}$ interrupt input.

- P55/D5/$\overline{\text{TEND3}}$

  The pin function is switched as shown below according to the combination of the P55DDR bit, the TENDE bit in DMMDR_3, and the EXPE bit in MDCR.

EXPE	0		1			
Bus mode	—		All areas are 8-bit space			At least one area is 16-bit space
TENDE	0		0		1	—
P55DDR	0	1	0	1	—	—
Pin function	P55 input	P55 output	P55 input	P55 output	$\overline{\text{TEND3}}$ output	D5 I/O

- P54/D4/$\overline{\text{DREQ3}}$

  The pin function is switched as shown below according to the combination of the P54DDR bit, the DREQS bit in DMMDR_3, and the EXPE bit in MDCR.

EXPE	0		1		
Bus mode	—		All areas are 8-bit space		At least one area is 16-bit space
P54DDR	0	1	0	1	—
Pin function	P54 input	P54 output	P54 input	P54 output	D4 I/O
	$\overline{\text{DREQ3}}$ input*				

Note: * When DREQS = 1, this pin functions as $\overline{\text{DREQ3}}$ input.

- P53/D3/($\overline{\text{IRQ3}}$)/$\overline{\text{DRAK2}}$

  The pin function is switched as shown below according to the combination of the P53DDR bit, the DRAKE bit in DMMDR_2, the EXPE bit in MDCR, and the ISS3 bit in ISSR.

EXPE	0		1			
Bus mode	—		All areas are 8-bit space			At least one area is 16-bit space
DRAKE	0		0		1	—
P53DDR	0	1	0	1	—	—
Pin function	P53 input	P53 output	P53 input	P53 output	$\overline{\text{DRAK2}}$ output	D3 I/O
	$\overline{\text{IRQ3}}$ interrupt input*					

Note: * When ISS3 = 1, this pin functions as $\overline{\text{IRQ3}}$ interrupt input.

- P52/D2/($\overline{\text{IRQ2}}$)/$\overline{\text{DACK2}}$

  The pin function is switched as shown below according to the combination of the P52DDR bit, the AMS bit in DMMDR_2, the EXPE bit in MDCR, and the ISS2 bit in ISSR.

EXPE	0		1			
Bus mode	—		All areas are 8-bit space			At least one area is 16-bit space
AMS	0		0		1	—
P52DDR	0	1	0	1	—	—
Pin function	P52 input	P52 output	P52 input	P52 output	$\overline{\text{DACK2}}$ output	D2 I/O
	$\overline{\text{IRQ2}}$ interrupt input*					

Note: * When ISS2 = 1, this pin functions as $\overline{\text{IRQ2}}$ interrupt input.

- P51/D1/$\overline{\text{TEND2}}$

  The pin function is switched as shown below according to the combination of the P51DDR bit, the TENDE bit in DMMDR_2, and the EXPE bit in MDCR.

EXPE	0		1			
Bus mode	—		All areas are 8-bit space			At least one area is 16-bit space
TENDE	0		0		1	—
P51DDR	0	1	0	1	—	—
Pin function	P51 input	P51 output	P51 input	P51 output	$\overline{\text{TEND2}}$ output	D1 I/O

- P50/D0/$\overline{\text{DREQ2}}$

  The pin function is switched as shown below according to the combination of the P50DDR bit, the DREQS bit in DMMDR_2, and the EXPE bit in MDCR.

EXPE	0		1		
Bus mode	—		All areas are 8-bit space		At least one area is 16-bit space
P50DDR	0	1	0	1	—
Pin function	P50 input	P50 output	P50 input	P50 output	D0 I/O
	$\overline{\text{DREQ2}}$ input*				

Note: * When DREQS = 1, this pin functions as $\overline{\text{DREQ2}}$ input.

8.6 Port 6

Port 6 is an 8-bit I/O port that also has other functions. The port 6 has the following registers.

- Port 6 data direction register (P6DDR)
- Port 6 data register (P6DR)
- Port 6 register (PORT6)

8.6.1 Port 6 Data Direction Register (P6DDR)

The individual bits of P6DDR specify input or output for the pins of port 6. P6DDR cannot be read; if it is, an undefined value will be read.

Bit	Bit Name	Initial Value	R/W	Description
7	P67DDR	0	W	When a pin function is specified to a general purpose I/O, setting this bit to 1 makes the corresponding port 6 pin an output pin, while clearing this bit to 0 makes the pin an input pin.
6	P66DDR	0	W	
5	P65DDR	0	W	
4	P64DDR	0	W	
3	P63DDR	0	W	
2	P62DDR	0	W	
1	P61DDR	0	W	
0	P60DDR	0	W	

8.6.2 Port 6 Data Register (P6DR)

P6DR stores output data for the port 6 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P67DR	0	R/W	Output data for a pin is stored when the pin function is specified to a general purpose output.
6	P66DR	0	R/W	
5	P65DR	0	R/W	
4	P64DR	0	R/W	
3	P63DR	0	R/W	
2	P62DR	0	R/W	
1	P61DR	0	R/W	
0	P60DR	0	R/W	

8.6.3 Port 6 Register (PORT6)

PORT6 shows the pin states of the port 6. PORT6 cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	P67	—*	R	If a PORT6 read is performed while P6DDR bits are set to 1, the P6DR values are read. If a PORT6 read is performed while P6DDR bits are cleared to 0, the pin states are read.
6	P66	—*	R	
5	P65	—*	R	
4	P64	—*	R	
3	P63	—*	R	
2	P62	—*	R	
1	P61	—*	R	
0	P60	—*	R	

Note: * Determined by the states of pins P67 to P60.

8.6.4 Pin Functions

Port 6 pins also function as the pins for external address bus output and timer I/O. The correspondence between the register specification and the pin functions is shown below.

- P67/A15

  The pin function is switched as shown below according to the combination of the P67DDR bit, the EXPE bit in MDCR, and the AMOE bit in PFCR1.

EXPE	0		1		
AMOE	—		0		1
P67DDR	0	1	0	1	—
Pin function	P67 input	P67 output	P67 input	P67 output	A15 output

- P66/A14/TMO1

  The pin function is switched as shown below according to the combination of the P66DDR bit, the EXPE bit in MDCR, the AMOE bit in PFCR1, and the OS3 to OS0 bits in TCSR_1.

EXPE	0			1			
AMOE	—			0			1
OS3 to OS0	All 0		Not all 0	All 0		Not all 0	—
P66DDR	0	1	—	0	1	—	—
Pin function	P66 input	P66 output	TMO1 output	P66 input	P66 output	TMO1 output	A14 output

- P65/A13/TMRI1

  The pin function is switched as shown below according to the combination of the P65DDR bit, the EXPE bit in MDCR, and the AMOE bit in PFCR1.

EXPE	0		1		
AMOE	—		0		1
P65DDR	0	1	0	1	—
Pin function	P65 input	P65 output	P65 input	P65 output	A13 output
	TMRI1 input*				

Note:* When used as a TMR counter reset, set the CCLR1 and CCLR0 bits in TCR_1 to 11.

- P64/A12/TMCI1

  The pin function is switched as shown below according to the combination of the P64DDR bit, the EXPE bit in MDCR, and the AMOE bit in PFCR1.

EXPE	0		1		
AMOE	—		0		1
P64DDR	0	1	0	1	—
Pin function	P64 input	P64 output	P64 input	P64 output	A12 output
	TMCI1 input*				

Note:* When used as an external clock input pin of TMR, select the external clock by bits CKS2 to CKS0 in TCR_1.

- P63/A11

  The pin function is switched as shown below according to the combination of the P63DDR bit, the EXPE bit in MDCR, and the AMOE bit in PFCR1.

EXPE	0		1		
AMOE	—		0		1
P63DDR	0	1	0	1	—
Pin function	P63 input	P63 output	P63 input	P63 output	A11 output

- P62/A10/TMO0

  The pin function is switched as shown below according to the combination of the P62DDR bit, the EXPE bit in MDCR, the AMOE bit in PFCR1, and the OS3 to OS0 bits in TCSR_0.

EXPE	0			1			
AMOE	—			0			1
OS3 to OS0	All 0		Not all 0	All 0		Not all 0	—
P62DDR	0	1	—	0	1	—	—
Pin function	P62 input	P62 output	TMO0 output	P62 input	P62 output	TMO0 output	A10 output

- P61/A9/TMRI0

  The pin function is switched as shown below according to the combination of the P61DDR bit, the EXPE bit in MDCR, and the AMOE bit in PFCR1.

EXPE	0		1		
AMOE	—		0		1
P61DDR	0	1	0	1	—
Pin function	P61 input	P61 output	P61 input	P61 output	A9 output
	TMRI0 input*				

Note:* When used as a TMR counter reset, set the CCLR1 and CCLR0 bits in TCR_0 to 11.

- P60/A8/TMCI0

  The pin function is switched as shown below according to the combination of the P60DDR bit, the EXPE bit in MDCR, and the AMOE bit in PFCR1.

EXPE	0		1		
AMOE	—		0		1
P60DDR	0	1	0	1	—
Pin function	P60 input	P60 output	P60 input	P60 output	A8 output
	TMCI0 input*				

Note:* When used as an external clock input pin of TMR, select the external clock by bits CKS2 to CKS0 in TCR_0.

8.7 Port 7

Port 7 is an 8-bit I/O port that also has other functions. The port 7 has the following registers.

- Port 7 data direction register (P7DDR)
- Port 7 data register (P7DR)
- Port 7 register (PORT7)

8.7.1 Port 7 Data Direction Register (P7DDR)

The individual bits of P7DDR specify input or output for the pins of port 7. P7DDR cannot be read; if it is, an undefined value will be read.

Bit	Bit Name	Initial Value	R/W	Description
7	P77DDR	0	W	When a pin function is specified to a general purpose I/O, setting this bit to 1 makes the corresponding port 7 pin an output pin, while clearing this bit to 0 makes the pin an input pin.
6	P76DDR	0	W	
5	P75DDR	0	W	
4	P74DDR	0	W	
3	P73DDR	0	W	
2	P72DDR	0	W	
1	P71DDR	0	W	
0	P70DDR	0	W	

8.7.2 Port 7 Data Register (P7DR)

P7DR stores output data for the port 7 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P77DR	0	R/W	Output data for a pin is stored when the pin function is specified to a general purpose output.
6	P76DR	0	R/W	
5	P75DR	0	R/W	
4	P74DR	0	R/W	
3	P73DR	0	R/W	
2	P72DR	0	R/W	
1	P71DR	0	R/W	
0	P70DR	0	R/W	

8.7.3 Port 7 Register (PORT7)

PORT7 shows the pin states of the port 7. PORT7 cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	P77	—*	R	If a PORT7 read is performed while P7DDR bits are set to 1, the P7DR values are read. If a PORT7 read is performed while P7DDR bits are cleared to 0, the pin states are read.
6	P76	—*	R	
5	P75	—*	R	
4	P74	—*	R	
3	P73	—*	R	
2	P72	—*	R	
1	P71	—*	R	
0	P70	—*	R	

Note: * Determined by the states of pins P77 to P70.

8.7.4 Pin Functions

Port 7 pins also function as the pins for external address bus output and interrupt input. The correspondence between the register specification and the pin functions is shown below.

- P77/A7/$\overline{\text{IRQ7}}$

  The pin function is switched as shown below according to the combination of the P77DDR bit, the EXPE bit in MDCR, the ALOE bit in PFCR1, and the ISS7 bit in ISSR.

EXPE	0		1		
ALOE	—		0		1
P77DDR	0	1	0	1	—
Pin function	P77 input	P77 output	P77 input	P77 output	A7 output
	$\overline{\text{IRQ7}}$ interrupt input*				

Note: * When ISS7 = 0, this pin functions as $\overline{\text{IRQ7}}$ interrupt input.

- P76/A6/$\overline{\text{IRQ6}}$

  The pin function is switched as shown below according to the combination of the P76DDR bit, the EXPE bit in MDCR, the ALOE bit in PFCR1, and the ISS6 bit in ISSR.

EXPE	0		1		
ALOE	—		0		1
P76DDR	0	1	0	1	—
Pin function	P76 input	P76 output	P76 input	P76 output	A6 output
	$\overline{\text{IRQ6}}$ interrupt input*				

Note: * When ISS6 = 0, this pin functions as $\overline{\text{IRQ6}}$ interrupt input.

- P75/A5/$\overline{\text{IRQ5}}$

  The pin function is switched as shown below according to the combination of the P75DDR bit, the EXPE bit in MDCR, the ALOE bit in PFCR1, and the ISS5 bit in ISSR.

EXPE	0		1		
ALOE	—		0		1
P75DDR	0	1	0	1	—
Pin function	P75 input	P75 output	P75 input	P75 output	A5 output
	$\overline{\text{IRQ5}}$ interrupt input*				

Note: * When ISS5 = 0, this pin functions as $\overline{\text{IRQ5}}$ interrupt input.

- P74/A4/$\overline{\text{IRQ4}}$

  The pin function is switched as shown below according to the combination of the P74DDR bit, the EXPE bit in MDCR, the ALOE bit in PFCR1, and the ISS4 bit in ISSR.

EXPE	0		1		
ALOE	—		0		1
P74DDR	0	1	0	1	—
Pin function	P74 input	P74 output	P74 input	P74 output	A4 output
	$\overline{\text{IRQ4}}$ interrupt input*				

Note: * When ISS4 = 0, this pin functions as $\overline{\text{IRQ4}}$ interrupt input.

- P73/A3/$\overline{\text{IRQ3}}$

  The pin function is switched as shown below according to the combination of the P73DDR bit, the EXPE bit in MDCR, the ALOE bit in PFCR1, and the ISS3 bit in ISSR.

EXPE	0		1		
ALOE	—		0		1
P73DDR	0	1	0	1	—
Pin function	P73 input	P73 output	P73 input	P73 output	A3 output
	$\overline{\text{IRQ3}}$ interrupt input*				

Note: * When ISS3 = 0, this pin functions as $\overline{\text{IRQ3}}$ interrupt input.

- P72/A2/$\overline{\text{IRQ2}}$

  The pin function is switched as shown below according to the combination of the P72DDR bit, the EXPE bit in MDCR, the ALOE bit in PFCR1, and the ISS2 bit in ISSR.

EXPE	0		1		
ALOE	—		0		1
P72DDR	0	1	0	1	—
Pin function	P72 input	P72 output	P72 input	P72 output	A2 output
	$\overline{\text{IRQ2}}$ interrupt input*				

Note: * When ISS2 = 0, this pin functions as $\overline{\text{IRQ2}}$ interrupt input.

- P71/A1/$\overline{\text{IRQ1}}$

  The pin function is switched as shown below according to the combination of the P71DDR bit, the EXPE bit in MDCR, the ALOE bit in PFCR1, and the ISS1 bit in ISSR.

EXPE	0		1		
ALOE	—		0		1
P71DDR	0	1	0	1	—
Pin function	P71 input	P71 output	P71 input	P71 output	A1 output
	$\overline{\text{IRQ1}}$ interrupt input*				

Note: * When ISS1 = 0, this pin functions as $\overline{\text{IRQ1}}$ interrupt input.

- P70/A0/$\overline{\text{IRQ0}}$

  The pin function is switched as shown below according to the combination of the P70DDR bit, the EXPE bit in MDCR, the ALOE bit in PFCR1, and the ISS0 bit in ISSR.

EXPE	0		1		
ALOE	—		0		1
P70DDR	0	1	0	1	—
Pin function	P70 input	P70 output	P70 input	P70 output	A0 output
	$\overline{\text{IRQ0}}$ interrupt input*				

Note: * When ISS0 = 0, this pin functions as $\overline{\text{IRQ0}}$ interrupt input.

8.8 Port 8

Port 8 is an 8-bit I/O port that also has other functions. The port 8 has the following registers.

- Port 8 data direction register (P8DDR)
- Port 8 data register (P8DR)
- Port 8 register (PORT8)

8.8.1 Port 8 Data Direction Register (P8DDR)

The individual bits of P8DDR specify input or output for the pins of port 8. P8DDR cannot be read; if it is, an undefined value will be read.

Bit	Bit Name	Initial Value	R/W	Description
7	P87DDR	0	W	When a pin function is specified to a general purpose I/O, setting this bit to 1 makes the corresponding port 8 pin an output pin, while clearing this bit to 0 makes the pin an input pin.
6	P86DDR	0	W	
5	P85DDR	0	W	
4	P84DDR	0	W	
3	P83DDR	0	W	
2	P82DDR	0	W	
1	P81DDR	0	W	
0	P80DDR	0	W	

8.8.2 Port 8 Data Register (P8DR)

P8DR stores output data for the port 8 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P87DR	0	R/W	Output data for a pin is stored when the pin function is specified to a general purpose output.
6	P86DR	0	R/W	
5	P85DR	0	R/W	
4	P84DR	0	R/W	
3	P83DR	0	R/W	
2	P82DR	0	R/W	
1	P81DR	0	R/W	
0	P80DR	0	R/W	

8.8.3 Port 8 Register (PORT8)

PORT8 shows the pin states of the port 8. PORT8 cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	P87	—*	R	If a PORT8 read is performed while P8DDR bits are set to 1, the P8DR values are read. If a PORT8 read is performed while P8DDR bits are cleared to 0, the pin states are read.
6	P86	—*	R	
5	P85	—*	R	
4	P84	—*	R	
3	P83	—*	R	
2	P82	—*	R	
1	P81	—*	R	
0	P80	—*	R	

Note: * Determined by the states of pins P87 to P80.

8.8.4 Pin Functions

Port 8 pins also function as the pins for external data bus I/O. The correspondence between the register specification and the pin functions is shown below.

- P87/D15

  The pin function is switched as shown below according to the combination of the P87DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P87DDR	0	1	—
Pin function	P87 input	P87 output	D15 I/O

- P86/D14

  The pin function is switched as shown below according to the combination of the P86DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P86DDR	0	1	—
Pin function	P86 input	P86 output	D14 I/O

- P85/D13

  The pin function is switched as shown below according to the combination of the P85DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P85DDR	0	1	—
Pin function	P85 input	P85 output	D13 I/O

- P84/D12

  The pin function is switched as shown below according to the combination of the P84DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P84DDR	0	1	—
Pin function	P84 input	P84 output	D12 I/O

- P83/D11

  The pin function is switched as shown below according to the combination of the P83DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P83DDR	0	1	—
Pin function	P83 input	P83 output	D11 I/O

- P82/D10

  The pin function is switched as shown below according to the combination of the P82DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P82DDR	0	1	—
Pin function	P82 input	P82 output	D10 I/O

- P81/D9

  The pin function is switched as shown below according to the combination of the P81DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P81DDR	0	1	—
Pin function	P81 input	P81 output	D9 I/O

- P80/D8

  The pin function is switched as shown below according to the combination of the P80DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P80DDR	0	1	—
Pin function	P80 input	P80 output	D8 I/O

8.9 Port 9

Port 9 is an 8-bit I/O port that also has other functions. The port 9 has the following registers.

- Port 9 data direction register (P9DDR)
- Port 9 data register (P9DR)
- Port 9 register (PORT9)

8.9.1 Port 9 Data Direction Register (P9DDR)

The individual bits of P9DDR specify input or output for the pins of port 9. P9DDR cannot be read; if it is, an undefined value will be read.

Bit	Bit Name	Initial Value	R/W	Description
7	P97DDR	0	W	When a pin function is specified to a general purpose I/O, setting this bit to 1 makes the corresponding port 9 pin an output pin, while clearing this bit to 0 makes the pin an input pin.
6	P96DDR	0	W	
5	P95DDR	0	W	
4	P94DDR	0	W	
3	P93DDR	0	W	
2	P92DDR	0	W	
1	P91DDR	0	W	
0	P90DDR	0	W	

8.9.2 Port 9 Data Register (P9DR)

P9DR stores output data for the port 9 pins.

Bit	Bit Name	Initial Value	R/W	Description
7	P97DR	0	R/W	Output data for a pin is stored when the pin function is specified to a general purpose output.
6	P96DR	0	R/W	
5	P95DR	0	R/W	
4	P94DR	0	R/W	
3	P93DR	0	R/W	
2	P92DR	0	R/W	
1	P91DR	0	R/W	
0	P90DR	0	R/W	

8.9.3 Port 9 Register (PORT9)

PORT9 shows the pin states of the port 9. PORT9 cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	P97	—*	R	If a PORT9 read is performed while P9DDR bits are set to 1, the P9DR values are read. If a PORT9 read is performed while P9DDR bits are cleared to 0, the pin states are read.
6	P96	—*	R	
5	P95	—*	R	
4	P94	—*	R	
3	P93	—*	R	
2	P92	—*	R	
1	P91	—*	R	
0	P90	—*	R	

Note: * Determined by the states of pins P97 to P90.

8.9.4 Pin Functions

Port 9 pins also function as the pins for external bus control output. The correspondence between the register specification and the pin functions is shown below.

- P97/ϕ

  The pin function is switched as shown below according to the combination of the P97DDR bit and the CKOE bit in PFCR1.

CKOE	0		1
P97DDR	0	1	—
Pin function	P97 input	P97 output	ϕ output

- P96/$\overline{AS}$

  The pin function is switched as shown below according to the combination of the P96DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P96DDR	0	1	—
Pin function	P96 input	P96 output	$\overline{AS}$ output

- P95/$\overline{RD}$

  The pin function is switched as shown below according to the combination of the P95DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P95DDR	0	1	—
Pin function	P95 input	P95 output	$\overline{RD}$ output

- P94/$\overline{HWR}$

  The pin function is switched as shown below according to the combination of the P94DDR bit and the EXPE bit in MDCR.

EXPE	0		1
P94DDR	0	1	—
Pin function	P94 input	P94 output	$\overline{HWR}$ output

- P93/$\overline{LWR}$

  The pin function is switched as shown below according to the combination of the P93DDR bit and the EXPE bit in MDCR.

EXPE	0		1		
Bus mode	—		All areas are 8-bit space		At least one area is 16-bit space
P93DDR	0	1	0	1	—
Pin function	P93 input	P93 output	P93 input	P93 output	$\overline{LWR}$ output

- P92/$\overline{CS2}$/$\overline{RAS}$

  The pin function is switched as shown below according to the combination of the P92DDR bit, the EXPE bit in MDCR, and the CS2E bit in PFCR1.

EXPE	0		1			
Area 2	—		Normal space			DRAM space
CS2E	—		0		1	—
P92DDR	0	1	0	1	—	—
Pin function	P92 input	P92 output	P92 input	P92 output	$\overline{CS2}$ output	$\overline{RAS}$ output

- P91/$\overline{CS1}$

  The pin function is switched as shown below according to the combination of the P91DDR bit, the EXPE bit in MDCR, and the CS1E bit in PFCR1.

EXPE	0		1		
CS1E	—		0		1
P91DDR	0	1	0	1	—
Pin function	P91 input	P91 output	P91 input	P91 output	$\overline{CS1}$ output

- P90/$\overline{CS0}$

  The pin function is switched as shown below according to the combination of the P90DDR bit, the EXPE bit in MDCR, and the CS0E bit in PFCR1.

EXPE	0		1		
CS0E	—		0		1
P90DDR	0	1	0	1	—
Pin function	P90 input	P90 output	P90 input	P90 output	$\overline{CS0}$ output

8.10 Port A

Port A is a 4-bit I/O port that also has other functions. The port A has the following registers.

- Port A data direction register (PADDR)
- Port A data register (PADR)
- Port A register (PORTA)

8.10.1 Port A Data Direction Register (PADDR)

The individual bits of PADDR specify input or output for the pins of port A. PADDR cannot be read; if it is, an undefined value will be read.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	Undefined	—	Reserved If these bits are read, an undefined value will be read. These bits cannot be modified.
3	PA3DDR	0	W	When a pin function is specified to a general purpose I/O, setting this bit to 1 makes the corresponding port A pin an output pin, while clearing this bit to 0 makes the pin an input pin.
2	PA2DDR	0	W	
1	PA1DDR	0	W	
0	PA0DDR	0	W	

8.10.2 Port A Data Register (PADR)

PADR stores output data for the port A pins.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	Undefined	—	Reserved If these bits are read, an undefined value will be read. These bits cannot be modified.
3	PA3DR	0	R/W	Output data for a pin is stored when the pin function is specified to a general purpose output.
2	PA2DR	0	R/W	
1	PA1DR	0	R/W	
0	PA0DR	0	R/W	

8.10.3 Port A Register (PORTA)

PORTA shows the pin states of the port A. PORTA cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	RxDMON	—*[1]	R	This bit is read as the reversed value of the HUDIDI/RxD0 pin. 1 is read when the HUDIDI/RxD0 pin is low, and 0 is read when HUDIDI/RxD0 pin is high.
6 to 4	—	Undefined	—	Reserved If these bits are read, an undefined value will be read.
3	PA3	—*[2]	R	If a PORTA read is performed while PADDR bits are set to 1, the PADR values are read. If a PORTA read is performed while PADDR bits are cleared to 0, the pin states are read.
2	PA2	—*[2]	R	
1	PA1	—*[2]	R	
0	PA0	—*[2]	R	

Notes: 1. Determined by the state of the HUDIDI/RxD0 pin.
2. Determined by the states of pins PA3 to PA0.

8.10.4 Pin Functions

Port A pins also function as the pins for external address bus output, external bus control signal output, and interrupt input. The correspondence between the register specification and the pin functions is shown below.

- PA3/A19/$\overline{CS3}$

  The pin function is switched as shown below according to the combination of the PA3DDR bit, the EXPE bit in MDCR, and the CS3E and AHOE bits in PFCR1.

EXPE	0		1			
CS3E	—		0			1
AHOE	—		0		1	—
PA3DDR	0	1	0	1	—	—
Pin function	PA3 input	PA3 output	PA3 input	PA3 output	A19 output	$\overline{CS3}$ output

- PA2/A18/$\overline{UCAS}$

  The pin function is switched as shown below according to the combination of the PA2DDR bit, the EXPE bit in MDCR, and the AHOE bit in PFCR1.

EXPE	0		1			
Area 2	—		Normal space			DRAM space
AHOE	—		0		1	—
PA2DDR	0	1	0	1	—	—
Pin function	PA2 input	PA2 output	PA2 input	PA2 output	A18 output	$\overline{UCAS}$ output

- PA1/A18/$\overline{\text{UCAS}}$

  The pin function is switched as shown below according to the combination of the PA2DDR bit, the EXPE bit in MDCR, and the AHOE bit in PFCR1.

EXPE	0		1			
Area 2	—		Normal space			DRAM space
AHOE	—		0		1	—
PA1DDR	0	1	0	1	—	—
Pin function	PA1 input	PA1 output	PA1 input	PA1 output	A17 output	$\overline{\text{LCAS}}$ output
	$\overline{\text{IRQ1}}$ interrupt input*					

Note: * When ISS1 = 1, this pin functions as $\overline{\text{IRQ1}}$ interrupt input.

- PA0/A16/($\overline{\text{IRQ0}}$)

  The pin function is switched as shown below according to the combination of the PA0DDR bit, the EXPE bit in MDCR, the AHOE bit in PFCR1, and the ISS0 bit in ISSR.

EXPE	0		1		
AHOE	—		0		1
PA0DDR	0	1	0	1	—
Pin function	PA0 input	PA0 output	PA0 input	PA0 output	A16 output
	$\overline{\text{IRQ0}}$ interrupt input*				

Note: * When ISS0 = 1, this pin functions as $\overline{\text{IRQ0}}$ interrupt input.

8.11 Pin Selection

8.11.1 Port Function Control Register 1 (PFCR1)

PFCR1 performs I/O port control for the external interface pin and ϕ output pin.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved This bit can be read from or written to. However, the write value should always be 0.
6	CS1E	0	R/W	$\overline{CS1}$ Enable Enables or disables output for $\overline{CS1}$. 0: Set as I/O port 1: Set as $\overline{CS1}$ output pin
5	CS2E	0	R/W	$\overline{CS2}$ Enable Enables or disables output for $\overline{CS2}$. 0: Set as I/O port 1: Set as $\overline{CS2}$ output pin
4	CS3E	0	R/W	$\overline{CS3}$ Enable Enables or disables output for $\overline{CS3}$. 0: Set as I/O port 1: Set as $\overline{CS3}$ output pin
3	CKOE	1	R/W	ϕ Output Enable Enables or disables output for ϕ. 0: Set as I/O port 1: Set as ϕ output pin
2	ALOE	1	R/W	Address Output Enable Enables or disables address output. 0: Set as I/O port 1: Set A7 to A0 pins as address output pins
1	AMOE	1	R/W	Address Output Enable Enables or disables address output. 0: Set as I/O port 1: Set A15 to A8 pins as address output pins
0	AHOE	0	R/W	Address Output Enable Enables or disables address output. 0: Set as I/O port 1: Set A19 to A16 pins as address output pins

8.11.2 IRQ Sense Port Select Register (ISSR)

ISSR selects the input pins for IRQ7 to IRQ0.

Bit	Bit Name	Initial Value	R/W	Description
7	ISS7	0	R/W	Selects an input pin for $\overline{\text{IRQ7}}$. 0: P77 1: P57
6	ISS6	0	R/W	Selects an input pin for $\overline{\text{IRQ6}}$. 0: P76 1: P56
5	ISS5	0	R/W	Selects an input pin for $\overline{\text{IRQ5}}$. 0: P75 1: P35
4	ISS4	0	R/W	Selects an input pin for $\overline{\text{IRQ4}}$. 0: P74 1: P34
3	ISS3	0	R/W	Selects an input pin for $\overline{\text{IRQ3}}$. 0: P73 1: P53
2	ISS2	0	R/W	Selects an input pin for $\overline{\text{IRQ2}}$. 0: P72 1: P52
1	ISS1	0	R/W	Selects an input pin for $\overline{\text{IRQ1}}$. 0: P71 1: PA1
0	ISS0	0	R/W	Selects an input pin for $\overline{\text{IRQ0}}$. 0: P70 1: PA0

Section 9 8-Bit Timer (TMR)

This LSI has an on-chip 8-bit timer module with two channels operating on the basis of an 8-bit counter. The 8-bit timer module can be used to count external events and be used as a multifunction timer in a variety of applications, such as generation of counter reset, interrupt requests, and pulse output with an arbitrary duty cycle using a compare-match signal with two registers.

9.1 Features

- Selection of four clock sources

  The counters can be driven by one of three internal clock signals (ϕ/8, ϕ/64, or ϕ/8192) or an external clock input
- Selection of three ways to clear the counters

  The counters can be cleared on compare match A or B, or by an external reset signal
- Timer output control by a combination of two compare match signals

  The timer output signal in each channel is controlled by a combination of two independent compare match signals, enabling the timer to generate output waveforms with an arbitrary duty cycle or PWM output
- Provision for cascading of two channels (TMR_0 and TMR_1)

  Operation as a 16-bit timer is possible, using TMR_0 for the upper 8 bits and TMR_1 for the lower 8 bits (16-bit count mode)

  TMR_1 can be used to count TMR_0 compare matches (compare match count mode)
- Three independent interrupts

  Compare match A and B and overflow interrupts can be requested independently

Figure 9.1 shows a block diagram of the 8-bit timer module (TMR_0 and TMR_1).

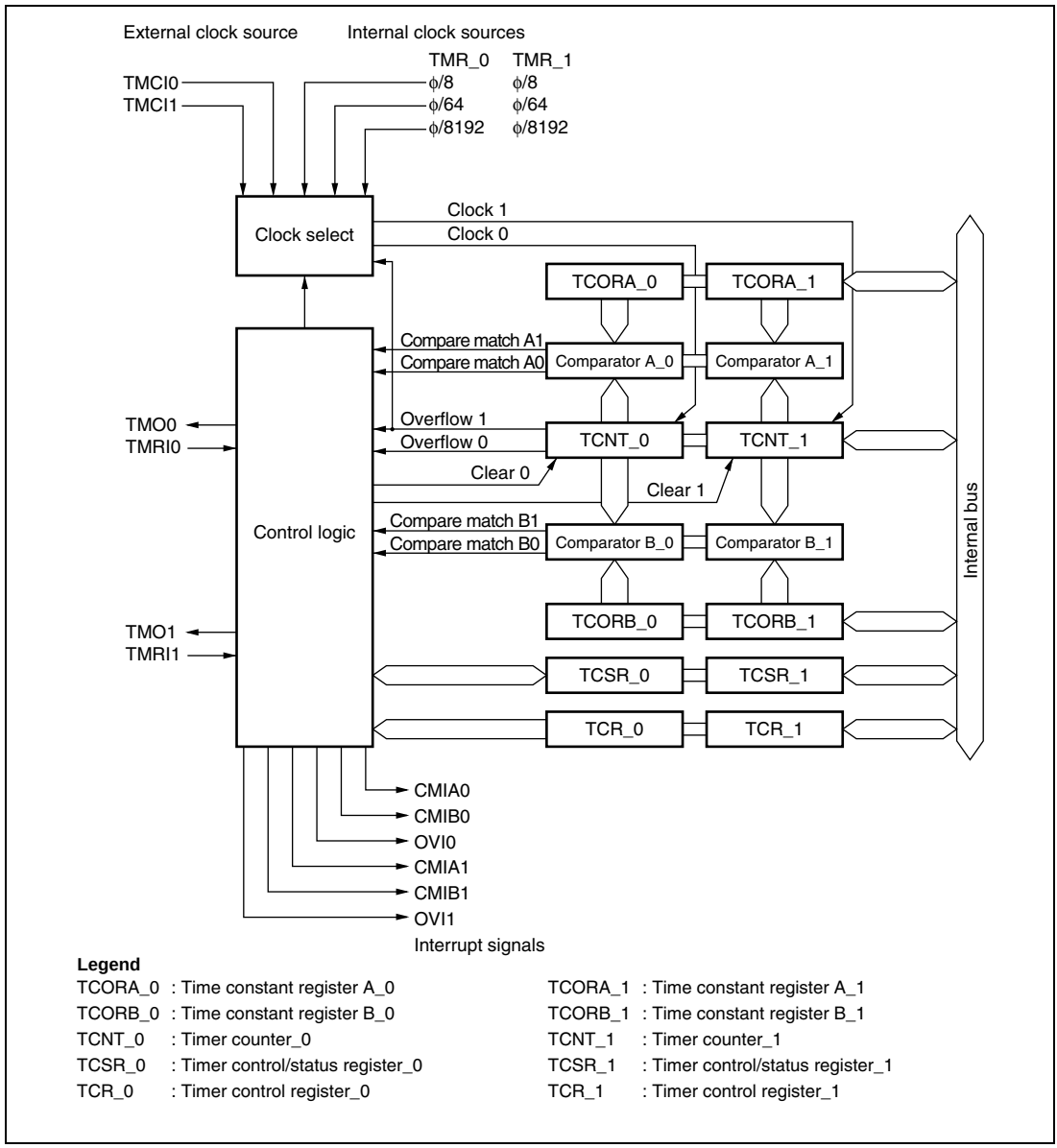


Figure 9.1 Block Diagram of 8-Bit Timer Module

9.2 Input/Output Pins

Table 9.1 shows the pin configuration of the 8-bit timer module.

Table 9.1 Pin Configuration

Channel	Name	Symbol	I/O	Function
0	Timer output pin	TMO0	Output	Outputs at compare match
	Timer clock input pin	TMCI0	Input	Inputs external clock for counter
	Timer reset input pin	TMRI0	Input	Inputs external reset to counter
1	Timer output pin	TMO1	Output	Outputs at compare match
	Timer clock input pin	TMCI1	Input	Inputs external clock for counter
	Timer reset input pin	TMRI1	Input	Inputs external reset to counter

9.3 Register Descriptions

The 8-bit timer module has the following registers. For details on the module stop control register, refer to section 16.1.2 Module Stop Control Registers H, L (MSTPCRH, MSTPCRL).

- Timer counter_0 (TCNT_0)
- Time constant register A_0 (TCORA_0)
- Time constant register B_0 (TCORB_0)
- Timer control register_0 (TCR_0)
- Timer control/status register_0 (TCSR_0)
- Timer counter_1 (TCNT_1)
- Time constant register A_1 (TCORA_1)
- Time constant register B_1 (TCORB_1)
- Timer control register_1 (TCR_1)
- Timer control/status register_1 (TCSR_1)

9.3.1 Timer Counter (TCNT)

TCNT is 8-bit up-counter. TCNT_0 and TCNT_1 comprise a single 16-bit register so they can be accessed together by a word transfer instruction. Bits CKS2 to CKS0 in TCR are used to select a clock. TCNT can be cleared by an external reset input or by a compare match signal A or B. Which signal is to be used for clearing is selected by bits CCLR1 and CCLR0 in TCR. When TCNT overflows from H'FF to H'00, OVF in TCSR is set to 1. TCNT is initialized to H'00.

9.3.2 Time Constant Register A (TCORA)

TCORA is 8-bit readable/writable register. TCORA_0 and TCORA_1 comprise a single 16-bit register so they can be accessed together by a word transfer instruction.

The value in TCORA is continually compared with the value in TCNT. When a match is detected, the corresponding CMFA flag in TCSR is set to 1. Note, however, that comparison is disabled during the T2 state of a TCORA write cycle.

The timer output from the TMO pin can be freely controlled by this compare match signal (compare match A) and the settings of bits OS1 and OS0 in TCSR.

TCORA is initialized to H'FF.

9.3.3 Time Constant Register B (TCORB)

TCORB is 8-bit readable/writable register. TCORB_0 and TCORB_1 comprise a single 16-bit register so they can be accessed together by a word transfer instruction.

TCORB is continually compared with the value in TCNT. When a match is detected, the corresponding CMFB flag in TCSR is set to 1. Note, however, that comparison is disabled during the T2 state of a TCOBR write cycle.

The timer output from the TMO pin can be freely controlled by this compare match signal (compare match B) and the settings of bits OS3 and OS2 in TCSR.

TCORB is initialized to H'FF.

9.3.4 Timer Control Register (TCR)

TCR selects the clock source and the time at which TCNT is cleared, and controls interrupts.

Bit	Bit Name	Initial Value	R/W	Description
7	CMIEB	0	R/W	Compare Match Interrupt Enable B Selects whether CMFB interrupt requests (CMIB) are enabled or disabled when the CMFB flag in TCSR is set to 1. 0: CMFB interrupt requests (CMIB) are disabled 1: CMFB interrupt requests (CMIB) are enabled

Bit	Bit Name	Initial Value	R/W	Description
6	CMIEA	0	R/W	Compare Match Interrupt Enable A Selects whether CMFA interrupt requests (CMIA) are enabled or disabled when the CMFA flag in TCSR is set to 1. 0: CMFA interrupt requests (CMIA) are disabled 1: CMFA interrupt requests (CMIA) are enabled
5	OVIE	0	R/W	Timer Overflow Interrupt Enable Selects whether OVF interrupt requests (OVI) are enabled or disabled when the OVF flag in TCSR is set to 1. 0: OVF interrupt requests (OVI) are disabled 1: OVF interrupt requests (OVI) are enabled
4	CCLR1	0	R/W	Counter Clear 1 and 0
3	CCLR0	0	R/W	These bits select the method by which TCNT is cleared 00: Clearing is disabled 01: Clear by compare match A 10: Clear by compare match B 11: Clear by rising edge of external reset input
2	CKS2	0	R/W	Clock Select 2 to 0
1	CKS1	0	R/W	These bits select the clock input to TCNT and count condition. See table 9.2.
0	CKS0	0	R/W	

Table 9.2 Clock Input to TCNT and Count Condition

Channel	TCR Bit 2 CKS2	TCR Bit 1 CKS1	TCR Bit 0 CKS0	Description
TMR_0	0	0	0	Clock input disabled
			1	Internal clock, counted at falling edge of ϕ/8
		1	0	Internal clock, counted at falling edge of ϕ/64
			1	Internal clock, counted at falling edge of ϕ/8192
	1	0	0	Count at TCNT_1 overflow signal*
TMR_1	0	0	0	Clock input disabled
			1	Internal clock, counted at falling edge of ϕ/8
		1	0	Internal clock, counted at falling edge of ϕ/64
			1	Internal clock, counted at falling edge of ϕ/8192
	1	0	0	Count at TCNT_0 compare match A*
All	1	0	1	External clock, counted at rising edge
		1	0	External clock, counted at falling edge
		1	1	External clock, counted at both rising and falling edges

Note: If the count input of TMR_0 is the TCNT_1 overflow signal and that of TMR_1 is the TCNT_0 compare match signal, no incrementing clock is generated. Do not use this setting.

9.3.5 Timer Control/Status Register (TCSR)

TCSR displays status flags, and controls compare match output.

Bit	Bit Name	Initial Value	R/W	Description
7	CMFB	0	R/(W)*	Compare Match Flag B [Setting condition] • Set when TCNT matches TCORB [Clearing condition] • Cleared by reading CMFB when CMFB = 1, then writing 0 to CMFB

Bit	Bit Name	Initial Value	R/W	Description
6	CMFA	0	R/(W)*	Compare Match Flag A [Setting condition] • Set when TCNT matches TCORA [Clearing condition] • Cleared by reading CMFA when CMFA = 1, then writing 0 to CMFA
5	OVF	0	R/(W)*	Timer Overflow Flag [Setting condition] • Set when TCNT overflows from H'FF to H'00 [Clearing condition] • Cleared by reading OVF when OVF = 1, then writing 0 to OVF
4	—	Undefined	—	Reserved This bit is always read as an undefined value, and cannot be modified.
3	OS3	0	R/W	Output Select 3 and 2 These bits select a method of TMO pin output when compare match B of TCORB and TCNT occurs. 00: No change when compare match B occurs 01: 0 is output when compare match B occurs 10: 1 is output when compare match B occurs 11: Output is inverted when compare match B occurs (toggle output)
2	OS2	0	R/W	
1	OS1	0	R/W	Output Select 1 and 0 These bits select a method of TMO pin output when compare match A of TCORA and TCNT occurs. 00: No change when compare match A occurs 01: 0 is output when compare match A occurs 10: 1 is output when compare match A occurs 11: Output is inverted when compare match A occurs (toggle output)
0	OS0	0	R/W	

Note: Only 0 can be written to clear these flags.

9.4 Operation

9.4.1 Pulse Output

Figure 9.2 shows an example that the 8-bit timer is used to generate a pulse output with a selected duty cycle. The control bits are set as follows:

[1] In TCR, bit CCLR1 is cleared to 0 and bit CCLR0 is set to 1 so that the timer counter is cleared at a TCORA compare match.

[2] In TCSR, bits OS3 to OS0 are set to B'0110, causing the output to change to 1 at a TCORA compare match and to 0 at a TCORB compare match.

With these settings, the 8-bit timer provides output of pulses at a rate determined by TCORA with a pulse width determined by TCORB. No software intervention is required.

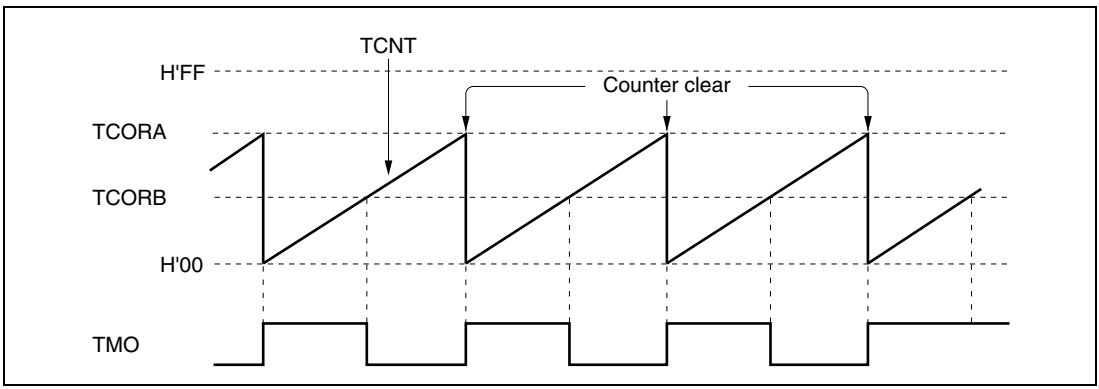


Figure 9.2 Example of Pulse Output

9.5 Operation Timing

9.5.1 TCNT Incrementation Timing

Figure 9.3 shows the count timing for internal clock input. Figure 9.4 shows the count timing for external clock signal. Note that the external clock pulse width must be at least 1.5 states for incrementation at a single edge, and at least 2.5 states for incrementation at both edges. The counter will not increment correctly if the pulse width is less than these values.

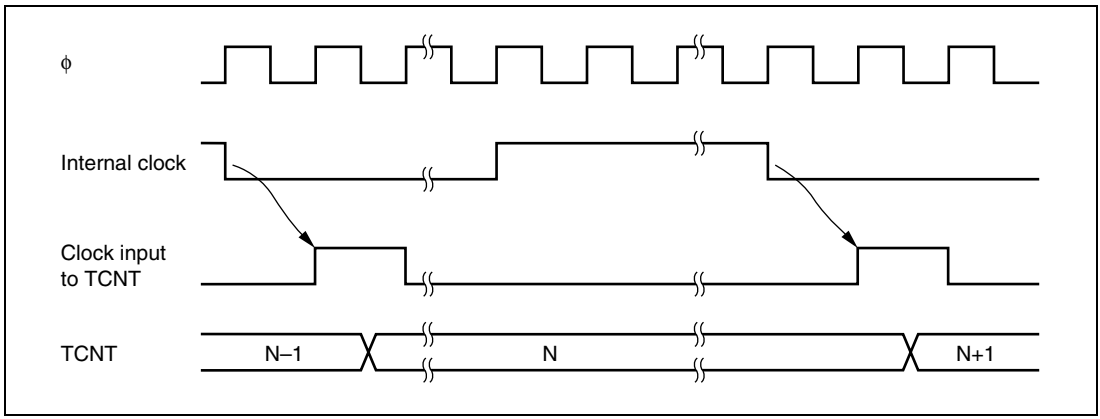


Figure 9.3 Count Timing for Internal Clock Input

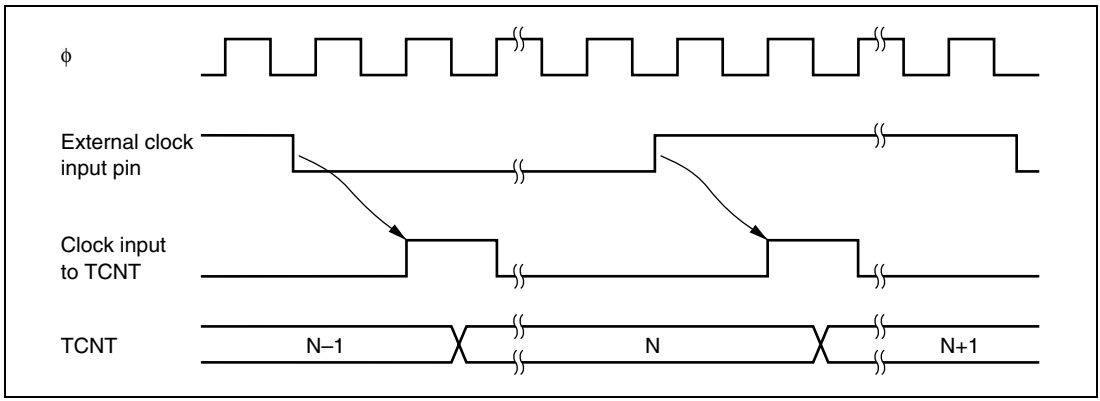


Figure 9.4 Count Timing for External Clock Input

9.5.2 Timing of CMFA and CMFB Setting when Compare-Match Occurs

The CMFA and CMFB flags in TCSR are set to 1 by a compare match signal generated when the TCOR and TCNT values match. The compare match signal is generated at the last state in which the match is true, just before the timer counter is updated. Therefore, when TCOR and TCNT match, the compare match signal is not generated until the next incrementation clock input. Figure 9.5 shows this timing.

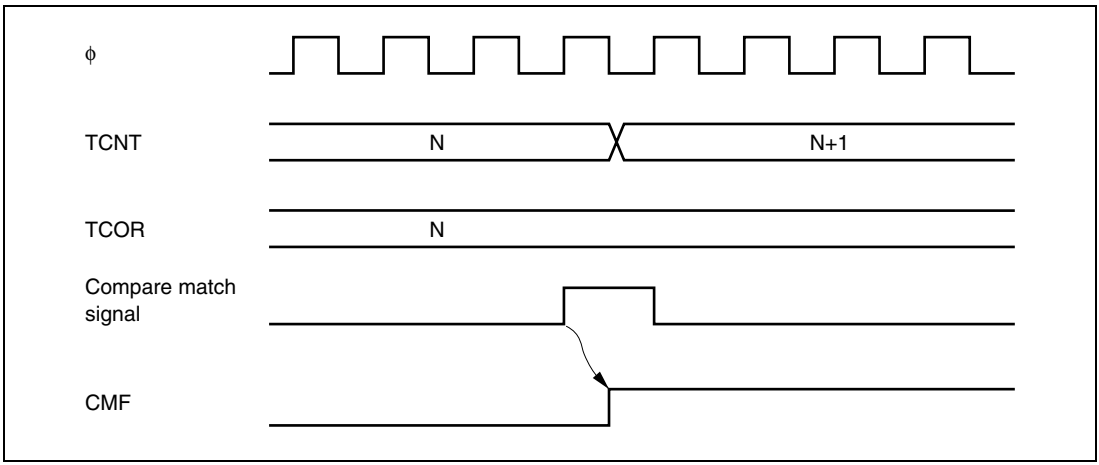


Figure 9.5 Timing of CMF Setting

9.5.3 Timing of Timer Output when Compare-Match Occurs

When compare match A or B occurs, the timer output changes as specified by bits OS3 to OS0 in TCSR.

Figure 9.6 shows the timing when the output is set to toggle at compare match A.

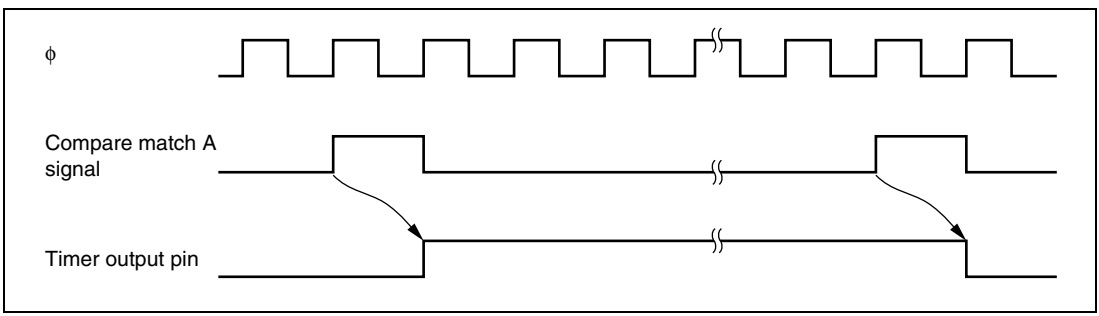


Figure 9.6 Timing of Timer Output

9.5.4 Timing of Compare Match Clear

TCNT is cleared when compare match A or B occurs, depending on the setting of the CCLR1 and CCLR0 bits in TCR. Figure 9.7 shows the timing of this operation.

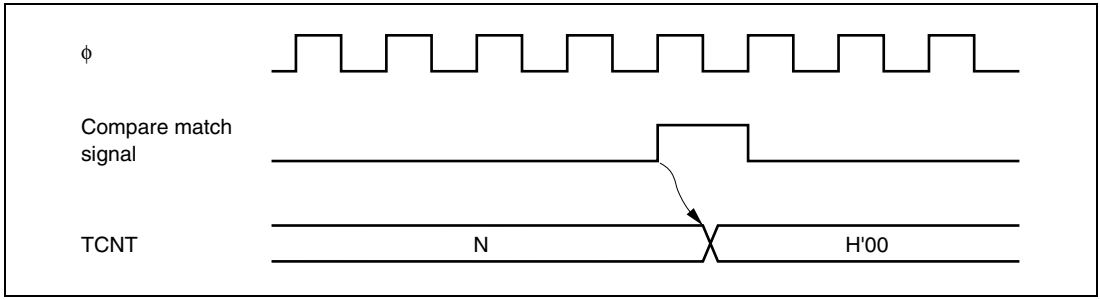


Figure 9.7 Timing of Compare Match Clear

9.5.5 Timing of TCNT External Reset

TCNT is cleared at the rising edge of an external reset input, depending on the settings of the CCLR1 and CCLR0 bits in TCR. The clear pulse width must be at least 1.5 states. Figure 9.8 shows the timing of this operation.

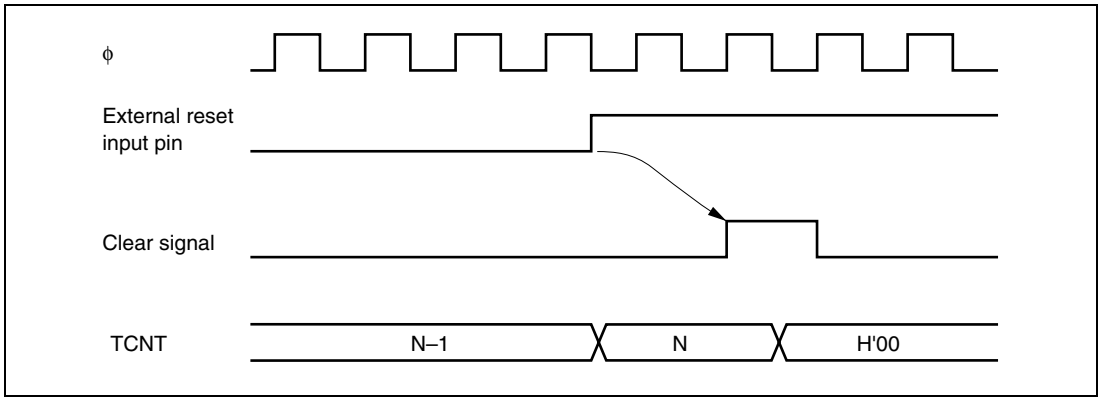


Figure 9.8 Timing of Clearance by External Reset

9.5.6 Timing of Overflow Flag (OVF) Setting

The OVF in TCSR is set to 1 when TCNT overflows (changes from H'FF to H'00). Figure 9.9 shows the timing of this operation.

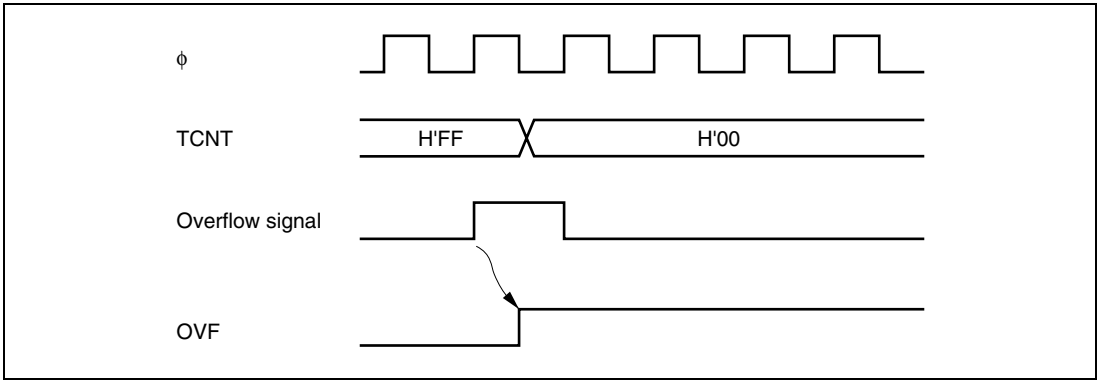


Figure 9.9 Timing of OVF Setting

9.6 Operation with Cascaded Connection

If bits CKS2 to CKS0 in either TCR_0 or TCR_1 are set to B'100, the 8-bit timers of the two channels are cascaded. With this configuration, a single 16-bit timer could be used (16-bit counter mode) or compare matches of the 8-bit channel 0 could be counted by the timer of channel 1 (compare match count mode). In this case, the timer operates as below.

9.6.1 16-Bit Counter Mode

When bits CKS2 to CKS0 in TCR_0 are set to B'100, the timer functions as a single 16-bit timer with channel 0 occupying the upper 8 bits and channel 1 occupying the lower 8 bits.

[1] Setting of compare match flags

- The CMF flag in TCSR_0 is set to 1 when a 16-bit compare match event occurs.
- The CMF flag in TCSR_1 is set to 1 when a lower 8-bit compare match event occurs.

[2] Counter clear specification

- If the CCLR1 and CCLR0 bits in TCR_0 have been set for counter clear at compare match, the 16-bit counters (TCNT_0 and TCNT_1 together) are cleared when a 16-bit compare match event occurs. The 16-bit counters (TCNT0 and TCNT1 together) are cleared even if counter clear by the TMRI0 pin has also been set.
- The settings of the CCLR1 and CCLR0 bits in TCR_1 are ignored. The lower 8 bits cannot be cleared independently.

[3] Pin output

- Control of output from the TMO0 pin by bits OS3 to OS0 in TCSR_0 is in accordance with the 16-bit compare match conditions.
- Control of output from the TMO1 pin by bits OS3 to OS0 in TCSR_1 is in accordance with the lower 8-bit compare match conditions.

9.6.2 Compare Match Count Mode

When bits CKS2 to CKS0 in TCR_1 are B'100, TCNT_1 counts compare match A's for channel 0.

Channels 0 and 1 are controlled independently. Conditions such as setting of the CMF flag, generation of interrupts, output from the TMO pin, and counter clear are in accordance with the settings for each channel.

9.7 Interrupt Sources

9.7.1 Interrupt Sources

There are three 8-bit timer interrupt sources: CMIA, CMIB, and OVI. Their relative priorities are shown in table 9.3. Each interrupt source is set as enabled or disabled by the corresponding interrupt enable bit in TCR or TCSR, and independent interrupt requests are sent for each to the interrupt controller.

Table 9.3 8-Bit Timer Interrupt Sources

Name	Interrupt Source	Interrupt Flag	Priority
CMIA0	TCORA_0 compare match	CMFA	High
CMIB0	TCORB_0 compare match	CMFB	↑
OVI0	TCNT_0 overflow	OVF	Low
CMIA1	TCORA_1 compare match	CMFA	High
CMIB1	TCORB_1 compare match	CMFB	↑
OVI1	TCNT_1 overflow	OVF	Low

9.8 Usage Notes

9.8.1 Contention between TCNT Write and Clear

If a timer counter clock pulse is generated during the T_2 state of a TCNT write cycle, the clear takes priority, so that the counter is cleared and the write is not performed.

Figure 9.10 shows this operation.

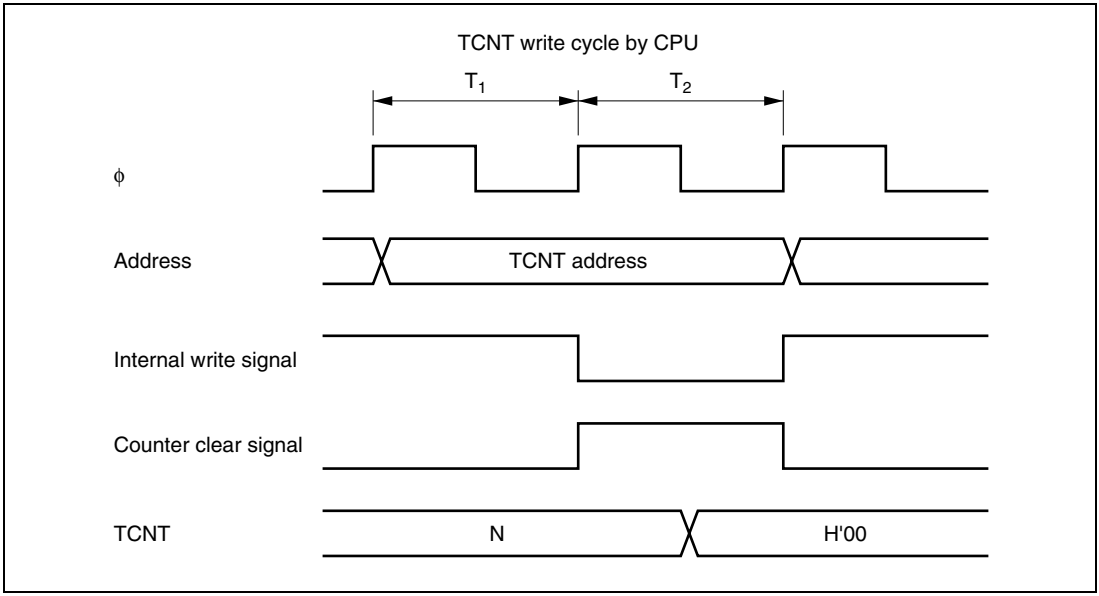


Figure 9.10 Contention between TCNT Write and Clear

9.8.2 Contention between TCNT Write and Increment

If a timer counter clock pulse is generated during the T_2 state of a TCNT write cycle, the write takes priority and the counter is not incremented.

Figure 9.11 shows this operation.

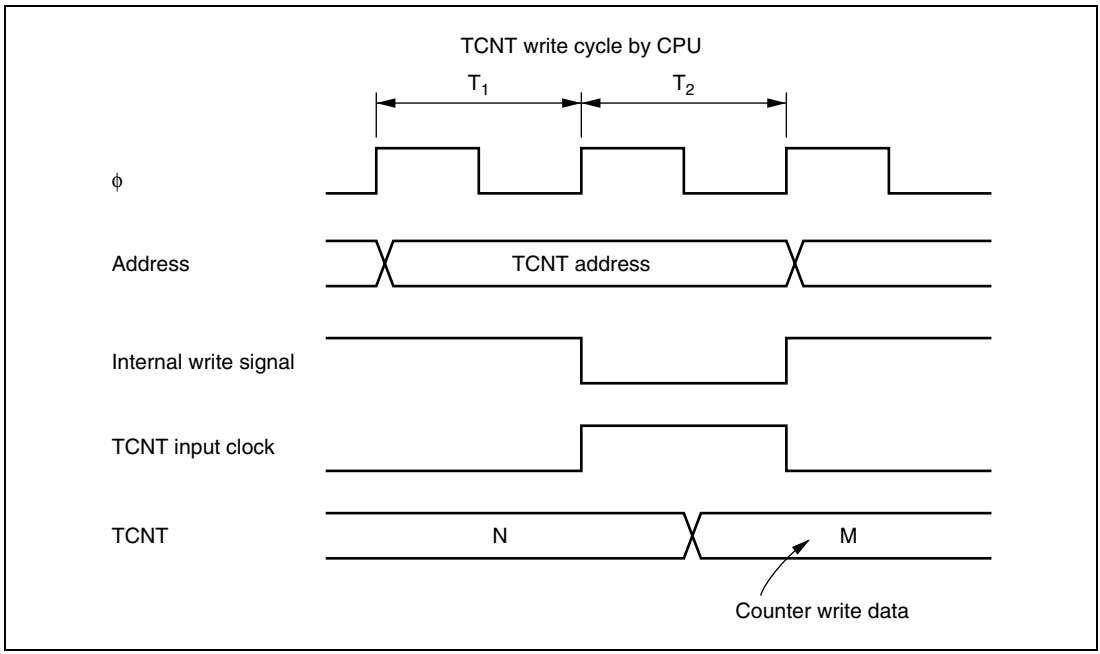


Figure 9.11 Contention between TCNT Write and Increment

9.8.3 Contention between TCOR Write and Compare Match

During the T_2 state of a TCOR write cycle, the TCOR write has priority and the compare match signal is inhibited even if a compare match event occurs as shown in figure 9.12.

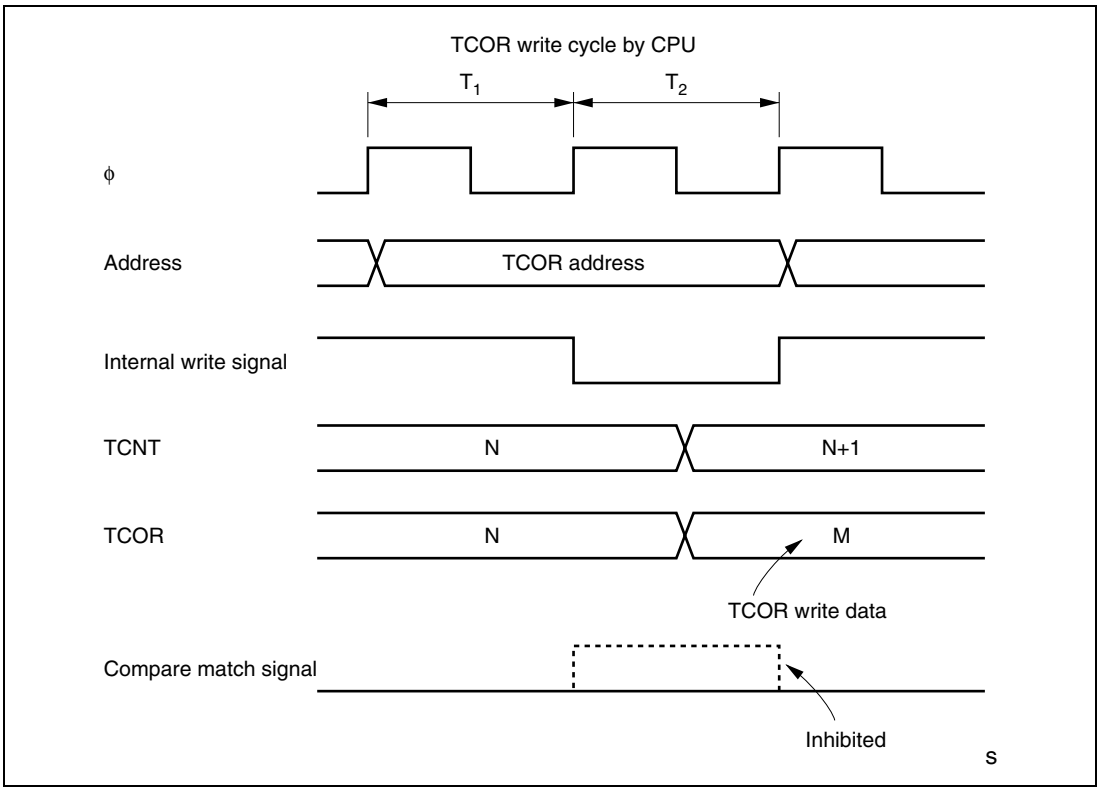


Figure 9.12 Contention between TCOR Write and Compare Match

9.8.4 Contention between Compare Matches A and B

If compare match events A and B occur at the same time, the 8-bit timer operates in accordance with the priorities for the output statuses set for compare match A and compare match B, as shown in table 9.4.

Table 9.4 Timer Output Priorities

Output Setting	Priority
Toggle output	High
1 output	↑
0 output	
No change	Low

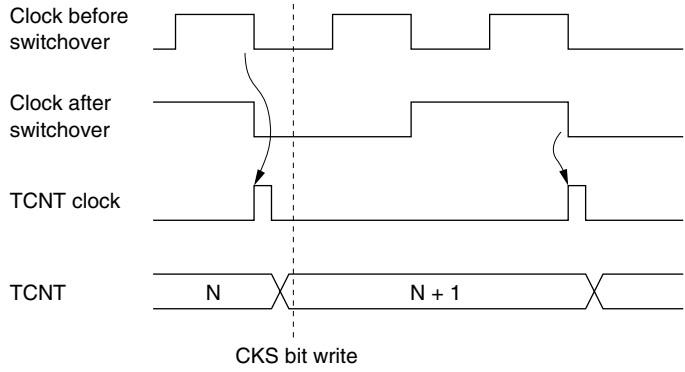
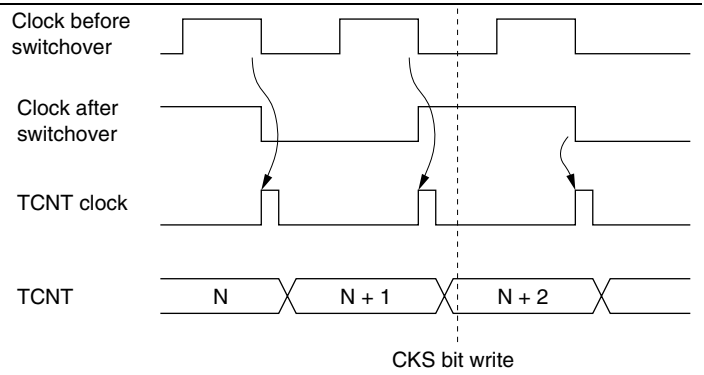
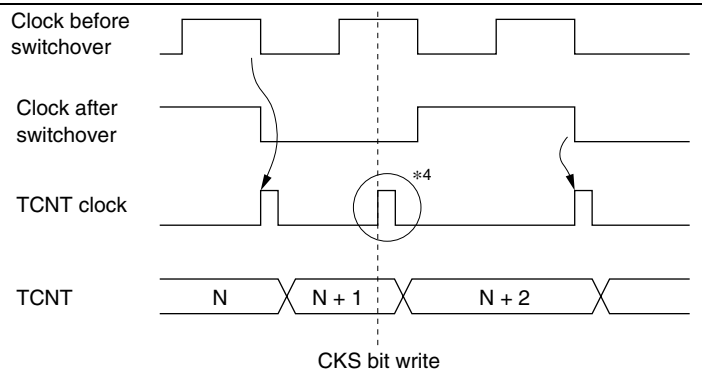
9.8.5 Switching of Internal Clocks and TCNT Operation

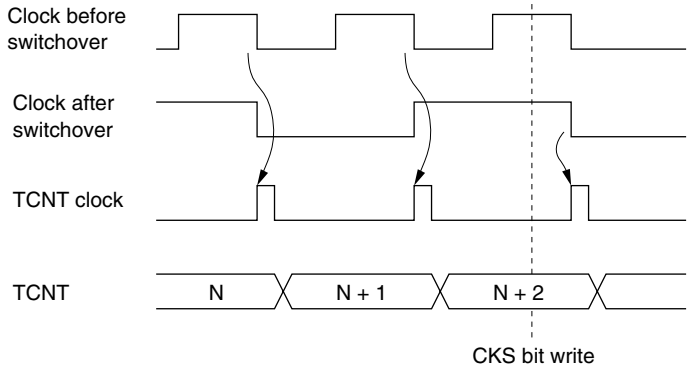
TCNT may increment erroneously when the internal clock is switched over. Table 9.5 shows the relationship between the timing at which the internal clock is switched (by writing to the CKS1 and CKS0 bits) and the TCNT operation.

When the TCNT clock is generated from an internal clock, the falling edge of the internal clock pulse is detected. If clock switching causes a change from high to low level, as shown in case 3 in table 9.5, a TCNT clock pulse is generated on the assumption that the switchover is a falling edge. This increments TCNT.

The erroneous incrementation can also happen when switching between internal and external clocks.

Table 9.5 Switching of Internal Clock and TCNT Operation

No.	Timing of Switchover by Means of CKS1 and CKS0 Bits	TCNT Clock Operation
1	Switching from low to low*1	
2	Switching from low to high*2	
3	Switching from high to low*3	

No.	Timing of Switchover by Means of CKS1 and CKS0 Bits	TCNT Clock Operation
4	Switching from high to high	

Notes: 1. Includes switching from low to stop, and from stop to low.
2. Includes switching from stop to high.
3. Includes switching from high to stop.
4. Generated on the assumption that the switchover is a falling edge; TCNT is incremented.

9.8.6 Mode Setting with Cascaded Connection

If 16-bit counter mode and compare match count mode are specified at the same time, input clocks for TCNT_0 and TCNT_1 are not generated, and the counter stops. Do not specify 16-bit counter and compare match count modes simultaneously.

9.8.7 Interrupts in Module Stop Mode

If module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source. Interrupts should therefore be disabled before entering module stop mode.

Section 10 Watchdog Timer (WDT)

This LSI incorporates an 8-bit watchdog timer with one channel (WDT). If a system crash prevents the CPU from writing to the timer counter, thus allowing it to overflow, the WDT can generate an internal reset signal or an internal NMI interrupt signal.

When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer operation, an interval timer interrupt is generated each time the counter overflows. A block diagram of the WDT is shown in figure 10.1.

10.1 Features

- Selectable from eight counter input clocks.
- Switchable between watchdog timer mode and interval timer mode

Watchdog Timer Mode:

- If the counter overflows, an internal reset or an internal NMI interrupt is generated.

Internal Timer Mode:

- If the counter overflows, an internal timer interrupt (WOVI) is generated.

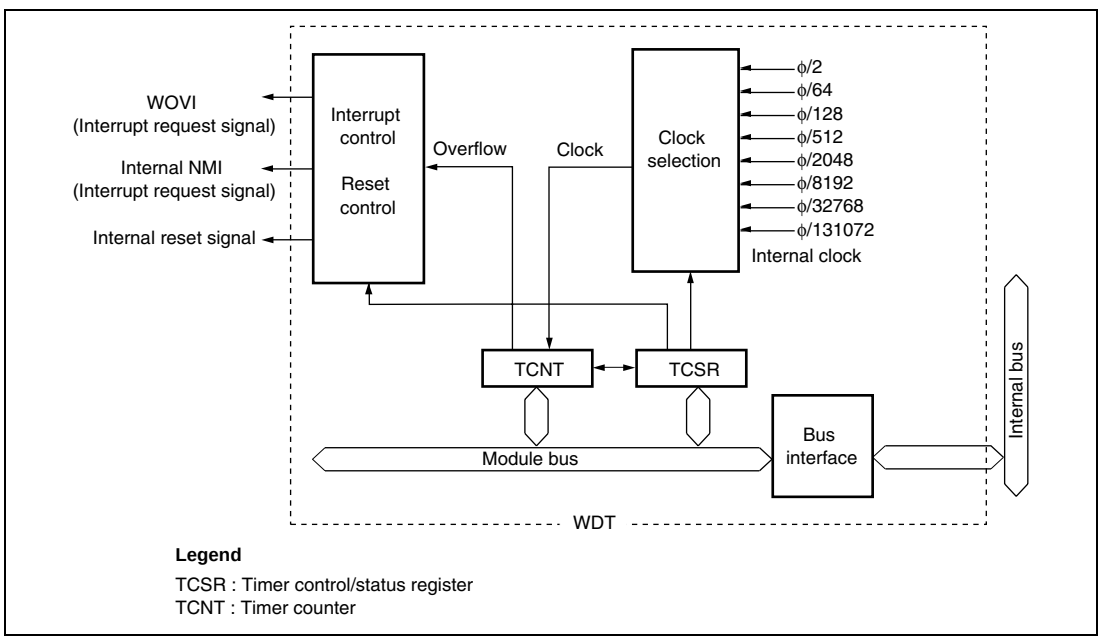


Figure 10.1 Block Diagram of WDT

10.2 Register Descriptions

The WDT has the following registers. To prevent accidental overwriting, TCSR and TCNT have to be written to in a method different from normal registers. For details, see section 10.5.1, Notes on Register Access.

- Timer counter (TCNT)
- Timer control/status register (TCSR)

10.2.1 Timer Counter (TCNT)

TCNT is an 8-bit readable/writable up-counter. TCNT is initialized to H'00 when the TME bit in timer control/status register (TCSR) is cleared to 0.

10.2.2 Timer Control/Status Register (TCSR)

TCSR selects the clock source to be input to TCNT, and the timer mode.

Bit	Bit Name	Initial Value	R/W	Description
7	OVF	0	R/(W)*[1]	Overflow Flag Indicates that TCNT has overflowed (changes from H'FF to H'00). [Setting condition] When TCNT overflows (changes from H'FF to H'00) When internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset. [Clearing conditions] • When TCSR is read when OVF = 1*[2], then 0 is written to OVF • When 0 is written to TME
6	WT/$\overline{IT}$	0	R/W	Timer Mode Select Selects whether the WDT is used as a watchdog timer or interval timer. 0: Interval timer mode 1: Watchdog timer mode
5	TME	0	R/W	Timer Enable When this bit is set to 1, TCNT starts counting. When this bit is cleared, TCNT stops counting and is initialized to H'00.

Bit	Bit Name	Initial Value	R/W	Description
4	—	0	R/W	Reserved The initial value should not be changed.
3	RST/$\overline{\text{NMI}}$	0	R/W	Reset or NMI Selects to request an internal reset or an NMI interrupt when TCNT has overflowed. 0: An NMI interrupt is requested 1: An internal reset is requested
2	CKS2	0	R/W	Clock Select 2 to 0 Selects the clock source to be input to. The overflow frequency for ø = 33 MHz is enclosed in parentheses. 000: φ/2 (frequency: 15.5 μs) 001: φ/64 (frequency: 496.4 μs) 010: φ/128 (frequency: 992.9 μs) 011: φ/512 (frequency: 3.9 ms) 100: φ/2048 (frequency: 15.8 ms) 101: φ/8192 (frequency: 65.5 ms) 110: φ/32768 (frequency: 254.2 ms) 111: φ/131072 (frequency: 1.01 s)
1	CKS1	0	R/W	
0	CKS0	0	R/W	

Notes: 1. Only 0 can be written to clear the flag.
2. When the OVF flag is polled with the interval timer interrupt disabled, OVF = 1 must be read at least twice.

10.3 Operation

10.3.1 Watchdog Timer Mode

To use the WDT as a watchdog timer, set the WT/$\overline{IT}$ bit and the TME bit in TCSR to 1. While the WDT is used as a watchdog timer, if TCNT overflows without being rewritten because of a system malfunction or another error, an internal reset or NMI interrupt request is generated. TCNT does not overflow while the system is operating normally. Software must prevent TCNT overflows by rewriting the TCNT value (normally be writing H'00) before overflows occurs.

If the RST/$\overline{NMI}$ bit of TCSR is set to 1, when the TCNT overflows, an internal reset signal for this LSI is issued for 518 system clocks as shown in figure 10.2. If the RST/$\overline{NMI}$ bit is cleared to 0, when the TCNT overflows, an NMI interrupt request is generated.

An internal reset request from the watchdog timer and a reset input from the $\overline{RES}$ pin are processed in the same vector. Reset source can be identified by the XRST bit status in SYSCR. If a reset caused by a signal input to the $\overline{RES}$ pin occurs at the same time as a reset caused by a WDT overflow, the $\overline{RES}$ pin reset has priority and the XRST bit in SYSCR is set to 1.

An NMI interrupt request from the watchdog timer and an interrupt request from the NMI pin are processed in the same vector. Do not handle an NMI interrupt request from the watchdog timer and an interrupt request from the NMI pin at the same time.

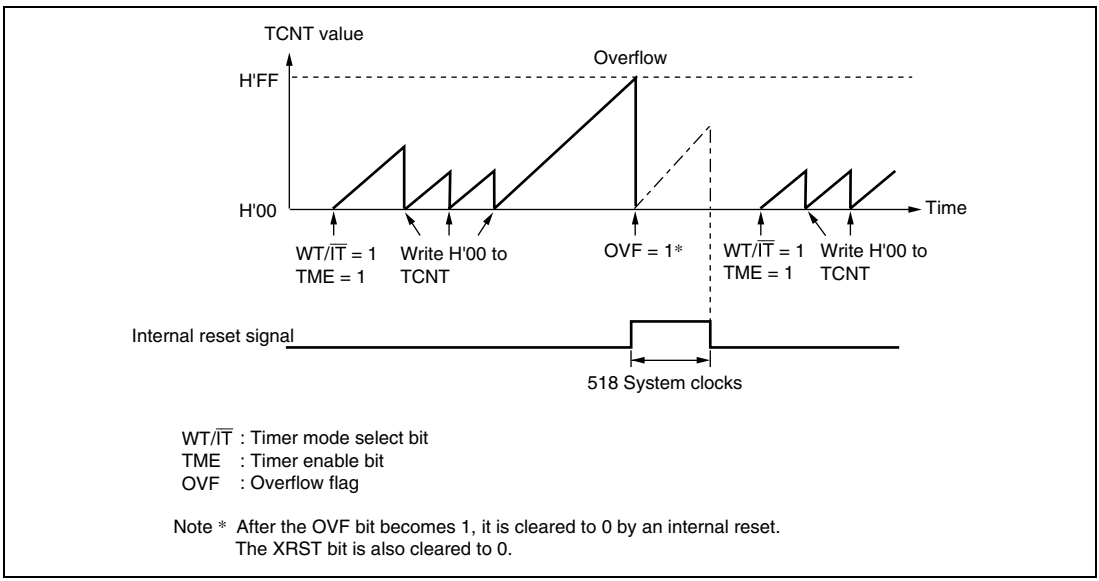


Figure 10.2 Watchdog Timer Mode (RST/$\overline{NMI}$ = 1) Operation

10.3.2 Interval Timer Mode

When the WDT is used as an interval timer, an interval timer interrupt (WOVI) is generated each time the TCNT overflows, as shown in figure 10.3. Therefore, an interrupt can be generated at intervals. When the TCNT overflows in interval timer mode, an interval timer interrupt (WOVI) is requested at the same time the OVF bit of TCSR is set to 1. The timing is shown figure 10.4.

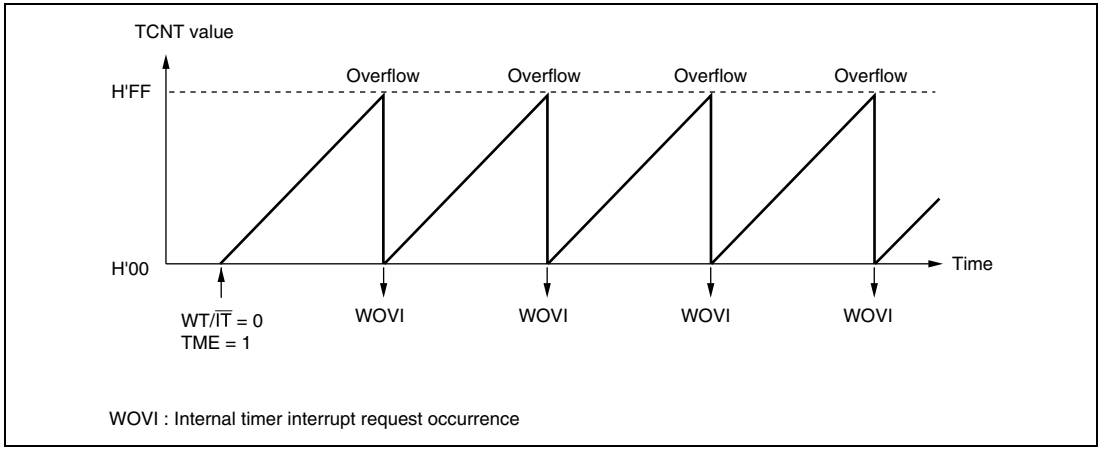


Figure 10.3 Interval Timer Mode Operation

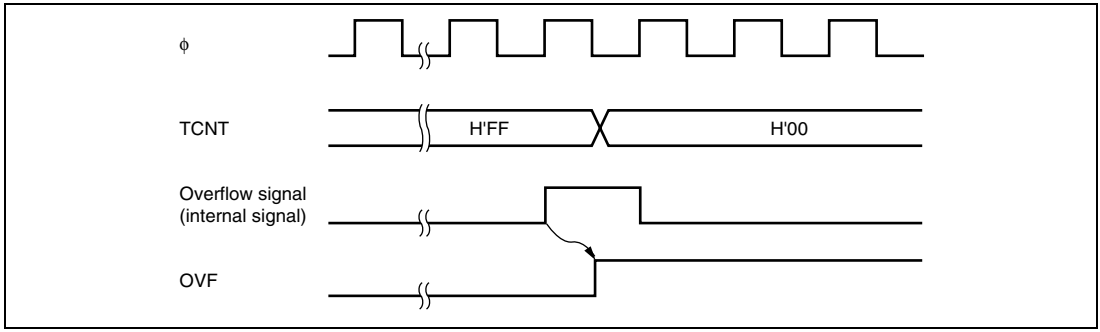


Figure 10.4 OVF Flag Set Timing

10.3.3 Watchdog Timer Overflow Flag (OVF) Timing

When TCNT overflows in watchdog timer mode, the OVF bit in TCSR is set to 1. When the RST/$\overline{\text{NMI}}$ bit is 1 here, the internal reset signal is generated for the entire LSI. The timing is shown in figure 10.5.

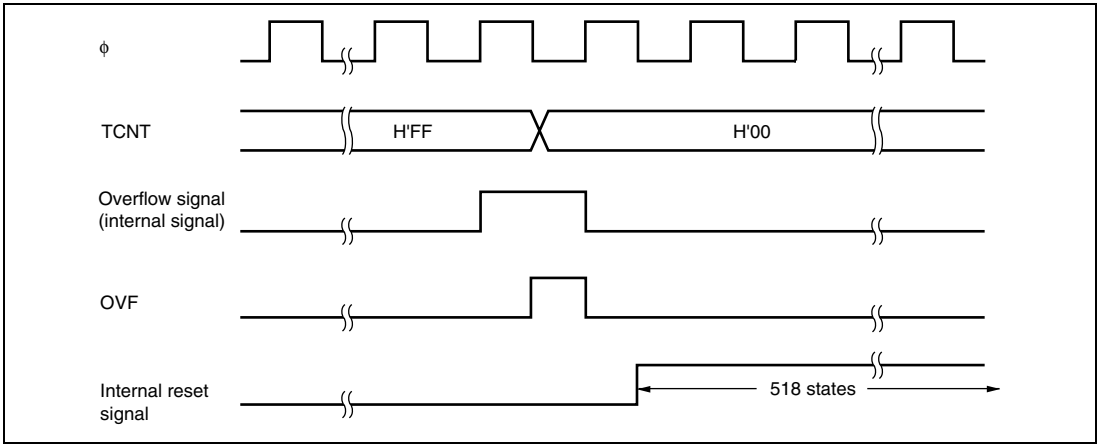


Figure 10.5 Output Timing of OVF

10.4 Interrupt Sources

During interval timer mode operation, an overflow generates an interval timer interrupt (WOVI). The interval timer interrupt is requested whenever the OVF flag is set to 1 in TCSR. OVF must be cleared to 0 in the interrupt handling routine.

When the NMI interrupt request is selected in watchdog timer mode, an NMI interrupt request is generated by an overflow.

Table 10.1 WDT Interrupt Source

Name	Interrupt Source	Interrupt Flag
WOVI	TCNT overflow	OVF

10.5 Usage Notes

10.5.1 Notes on Register Access

The watchdog timer's registers, TCNT and TCSR differ from other registers in being more difficult to write to. The procedures for writing to and reading from these registers are given below.

Writing to TCNT and TCSR:

These registers must be written to by a word transfer instruction. They cannot be written to by a byte transfer instruction.

TCNT and TCSR both have the same write address. Therefore, satisfy the relative condition shown in figure 10.6 to write to TCNT or TCSR. To write to TCNT, the higher bytes must contain the value H'5A and the lower bytes must contain the write data before the transfer instruction execution. To write to TCSR, the higher bytes must contain the value H'A5 and the lower bytes must contain the write data.

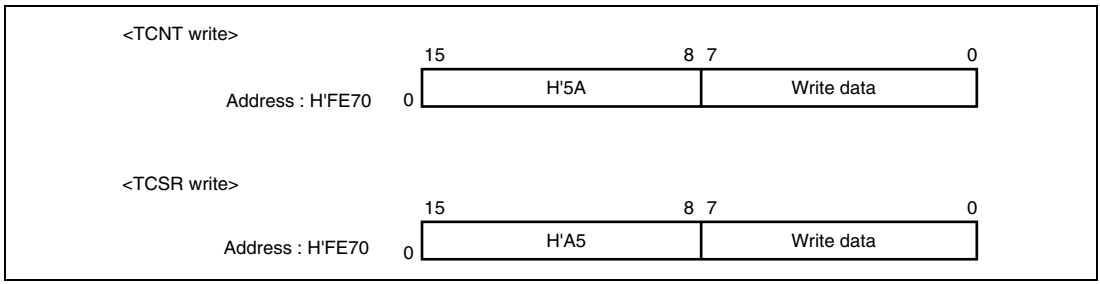


Figure 10.6 Writing to TCNT and TCSR

Reading from TCNT and TCSR:

These registers are read in the same way as other registers. The read address is H'FE70 for TCSR and H'FE71 for TCNT.

10.5.2 Conflict between Timer Counter (TCNT) Write and Increment

If a timer counter clock pulse is generated during the T2 state of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 10.7 shows this operation.

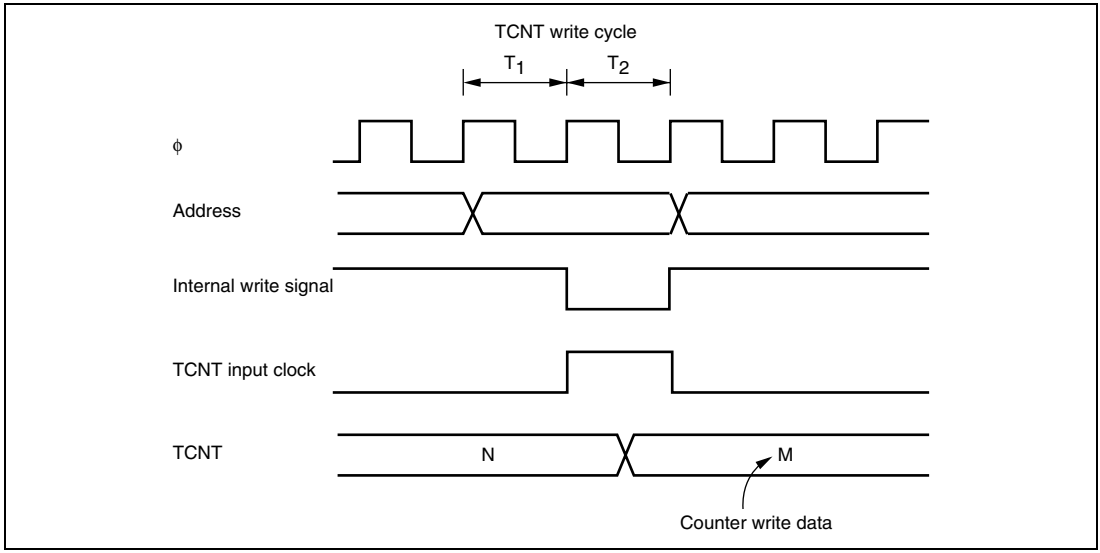


Figure 10.7 Conflict between TCNT Write and Increment

10.5.3 Changing Values of CKS2 to CKS0 Bits

If bits CKS2 to CKS0 in TCSR are written to while the WDT is operating, errors could occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before changing the values of bits CKS2 to CKS0.

10.5.4 Switching between Watchdog Timer Mode and Interval Timer Mode

If the mode is switched from watchdog timer to interval timer, while the WDT is operating, errors could occur in the incrementation. Software must stop the watchdog timer (by clearing the TME bit to 0) before switching the mode.

Section 11 Serial Communication Interface for Boot Mode (SCI)

This LSI has a serial communication interface for boot mode (SCI) which is on-board programming mode of flash memory. The SCI can handle asynchronous serial communication. Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA). Figure 11.1 shows a block diagram of the SCI.

11.1 Features

- Choice of asynchronous or clocked synchronous serial communication mode
- Full-duplex communication capability

  The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously. Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data.
- On-chip baud rate generator allows any bit rate to be selected
- Choice of LSB-first
- Four interrupt sources

  Four interrupt sources — transmit-end, transmit-data-empty, receive-data-full, and receive error — that can issue requests.
- Module stop mode can be set

Asynchronous mode

- Data length: 7 or 8 bits
- Stop bit length: 1 or 2 bits
- Parity: Even, odd, or none
- Receive error detection: Parity, overrun, and framing errors
- Break detection: Break can be detected by reading the RxD pin level directly in case of a framing error

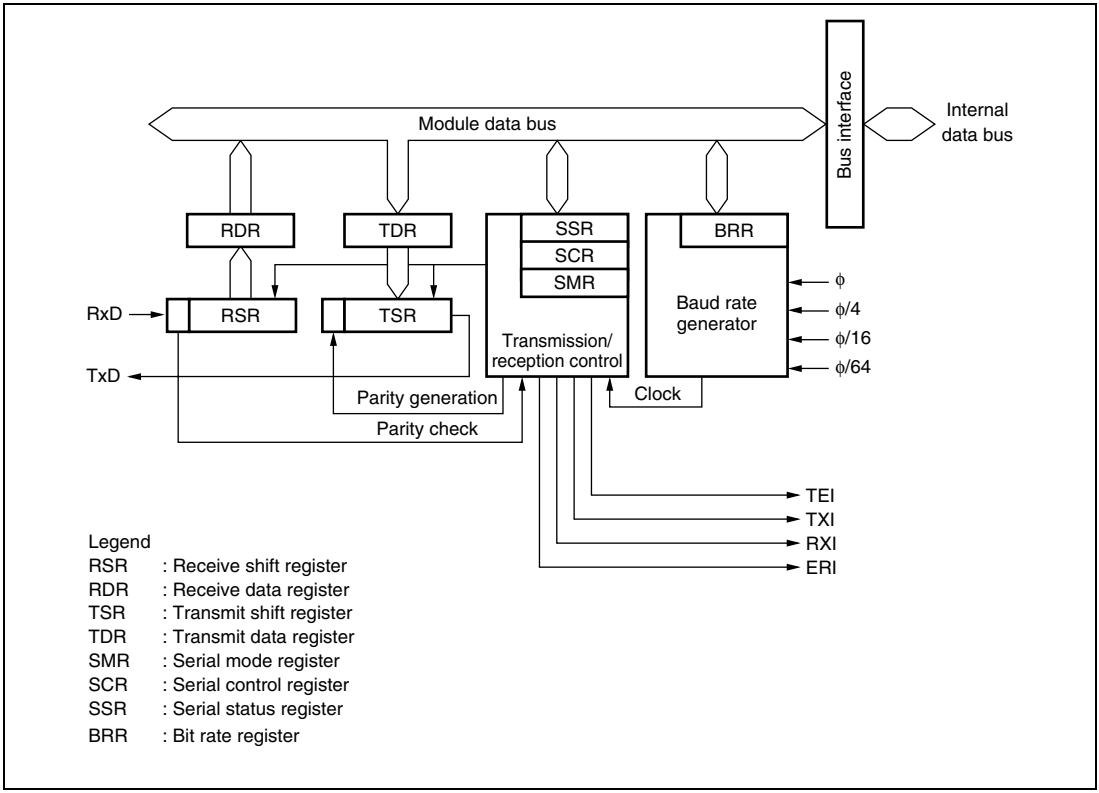


Figure 11.1 Block Diagram of SCI

11.2 Input/Output Pins

Table 11.1 shows the pin configuration of the SCI.

Table 11.1 Pin Configuration

Channel	Pin Name*	I/O	Function
0	RxD0	Input	Channel 0 receive data input
	TxD0	Output	Channel 0 transmit data output

Note: * Pin names RxD and TxD are used in the text, omitting the channel designation.

11.3 Register Descriptions

The SCI has the following registers. For details on the module stop control register, see section 16.1.2, Module Stop Control Registers H and L (MSTPCRH, MSTPCRL). To read the state of the RxD pin, see section 8.10.3, Port A Register (PORTA).

- Receive shift register (RSR)
- Transmit shift register (TSR)
- Receive data register (RDR)
- Transmit data register (TDR)
- Serial mode register (SMR)
- Serial control register (SCR)
- Serial status register (SSR)
- Bit rate register (BRR)

11.3.1 Receive Shift Register (RSR)

RSR is a shift register used to receive serial data that is input to the RxD pin and convert it into parallel data. When one byte of data has been received, it is transferred to RDR automatically. RSR cannot be directly accessed by the CPU.

11.3.2 Receive Data Register (RDR)

RDR is an 8-bit register that stores receive data. When the SCI has received one byte of serial data, it transfers the received serial data from RSR to RDR where it is stored. After this, RSR is receive-enabled. Since RSR and RDR function as a double buffer in this way, enables continuous receive operations to be performed. After confirming that the RDRF bit in SSR is set to 1, read RDR for only once. RDR cannot be written to by the CPU.

11.3.3 Transmit Data Register (TDR)

TDR is an 8-bit register that stores transmit data. When the SCI detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts transmission. The double-buffered structures of TDR and TSR enable continuous serial transmission. If the next transmit data has already been written to TDR during serial transmission, the SCI transfers the written data to TSR to continue transmission. Although TDR can be read or written to by the CPU at all times, to achieve reliable serial transmission, write transmit data to TDR for only once after confirming that the TDRE bit in SSR is set to 1.

11.3.4 Transmit Shift Register (TSR)

TSR is a shift register that transmits serial data. To perform serial data transmission, the SCI first transfers transmit data from TDR to TSR, then sends the data to the TxD pin starting. TSR cannot be directly accessed by the CPU.

11.3.5 Serial Mode Register (SMR)

SMR is used to set the SCI's serial transfer format and select the on-chip baud rate generator clock source.

Bit	Bit Name	Initial Value	R/W	Description
7	C/$\overline{A}$	0	R/W	Communication Mode 0: Asynchronous mode 1: Setting prohibited
6	CHR	0	R/W	Character Length (enabled only in asynchronous mode) 0: Selects 8 bits as the data length. 1: Selects 7 bits as the data length. The MSB (bit 7) of TDR is not transmitted in transmission.
5	PE	0	R/W	Parity Enable (enabled only in asynchronous mode) When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception.
4	O/$\overline{E}$	0	R/W	Parity Mode (enabled only when the PE bit is 1 in asynchronous mode) 0: Selects even parity. 1: Selects odd parity.
3	STOP	0	R/W	Stop Bit Length (enabled only in asynchronous mode) Selects the stop bit length in transmission. 0: 1 stop bit 1: 2 stop bits In reception, only the first stop bit is checked regardless of the STOP bit setting. If the second stop bit is 0, it is treated as the start bit of the next transmit character.
2	—	0	R/W	Reserved The initial value should not be changed.

Bit	Bit Name	Initial Value	R/W	Description
1	CKS1	0	R/W	Clock Select 1 and 0
0	CKS0	0	R/W	These bits select the clock source for the on-chip baud rate generator. 00: ϕ clock (n = 0) 01: ϕ/4 clock (n = 1) 10: ϕ/16 clock (n = 2) 11: ϕ/64 clock (n = 3) For the relation between the bit rate register setting and the baud rate, see section 11.3.8, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 11.3.8, Bit Rate Register (BRR)).

11.3.6 Serial Control Register (SCR)

SCR performs enabling or disabling of SCI transfer operations and interrupt requests, and selection of the transfer/receive clock source. For details on interrupt requests, refer to section 11.5, Interrupt Sources.

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1, TXI interrupt request is enabled. TXI interrupt request cancellation can be performed by reading 1 from the TDRE flag, then clearing it to 0, or clearing the TIE bit to 0.
6	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, RXI and ERI interrupt requests are enabled. RXI and ERI interrupt request cancellation can be performed by reading 1 from the RDRF flag, or the FER, PER, or ORER flag, then clearing the flag to 0, or by clearing the RIE bit to 0.

Bit	Bit Name	Initial Value	R/W	Description
5	TE	0	R/W	Transmit Enable When this bit s set to 1, transmission is enabled. In this state, serial transmission is started when transmit data is written to TDR and the TDRE flag in SSR is cleared to 0. SMR setting must be performed to decide the transfer format before setting the TE bit to 1. The TDRE flag in SSR is fixed at 1 if transmission is disabled by clearing this bit to 0.
4	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled. Serial reception is started in this state when a start bit is detected in asynchronous mode. SMR setting must be performed to decide the transfer format before setting the RE bit to 1. Clearing the RE bit to 0 does not affect the RDRF, FER, PER, and ORER flags, which retain their states.
3	—	0	R/W	Reserved The initial value should not be changed.
2	TEIE	0	R/W	Transmit End Interrupt Enable When this bit is set to 1, TEI interrupt request is enabled. TEI cancellation can be performed by reading 1 from the TDRE flag in SSR, then clearing it to 0 and clearing the TEND flag to 0, or by clearing the TEIE bit to 0.
1	CKE1	0	R/W	Clock Enable 1 and 0
0	CKE0	0	R/W	Selects the clock source. Asynchronous mode 0X: On-chip baud rate generator 1X: Setting prohibited

Legend: X: Don’t care

11.3.7 Serial Status Register (SSR)

SSR is a register containing status flags of the SCI. Flags TDRE, RDRF, ORER, PER, and FER can only be cleared.

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	Transmit Data Register Empty Indicates whether TDR contains transmit data. [Setting conditions] • When the TE bit in SCR is 0 • When data is transferred from TDR to TSR, and data writing to TDR is enabled. [Clearing condition] • When 0 is written to TDRE after reading TDRE = 1
6	RDRF	0	R/(W)*	Receive Data Register Full Indicates that the received data is stored in RDR. [Setting condition] • When serial reception ends normally and receive data is transferred from RSR to RDR [Clearing condition] • When 0 is written to RDRF after reading RDRF = 1 The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0. Exercise care because if reception of the next data is completed while the RDRF flag is set to 1, an overrun error occurs and receive data will be lost.

Bit	Bit Name	Initial Value	R/W	Description
5	ORER	0	R/(W)*	Overrun Error Indicates that an overrun error occurred while receiving and the reception has ended abnormally. [Setting condition] • When the next serial reception is completed while RDRF = 1 The receive data prior to the overrun error is retained in RDR, and the data received subsequently is lost. Also, subsequent serial reception cannot be continued while the ORER flag is set to 1. [Clearing condition] • When 0 is written to ORER after reading ORER = 1 The ORER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.
4	FER	0	R/(W)*	Framing Error Indicates that a framing error occurred while receiving in asynchronous mode and the reception has ended abnormally. [Setting condition] • When the stop bit is 0 In 2-stop-bit mode, only the first stop bit is checked for a value of 0; the second stop bit is not checked. If a framing error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the FER flag is set to 1. [Clearing condition] • When 0 is written to FER after reading FER = 1 The FER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.

Bit	Bit Name	Initial Value	R/W	Description
3	PER	0	R/(W)*	Parity Error Indicates that a parity error occurred while receiving in asynchronous mode and the reception has ended abnormally. [Setting condition] • When a parity error is detected during reception If a parity error occurs, the receive data is transferred to RDR but the RDRF flag is not set. Also, subsequent serial reception cannot be continued while the PER flag is set to 1. [Clearing condition] • When 0 is written to PER after reading PER = 1 The PER flag is not affected and retains its previous state when the RE bit in SCR is cleared to 0.
2	TEND	1	R	Transmit End [Setting conditions] • When the TE bit in SCR is 0 • When TDRE = 1 at transmission of the last bit of a 1-byte serial transmit character [Clearing condition] • When 0 is written to TDRE after reading TDRE = 1
1, 0	—	All 0	R/W	Reserved The initial value should not be changed.

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

11.3.8 Bit Rate Register (BRR)

BRR is an 8-bit register that adjusts the bit rate. As the SCI performs baud rate generator control independently for each channel, different bit rates can be set for each channel. Table 11.2 shows the relationships between the N setting in BRR and bit rate B for asynchronous mode. The initial value of BRR is H'FF, and it can be read or written to by the CPU at all times.

Table 11.2 Relationships between N Setting in BRR and Bit Rate B

Mode	Bit Rate	Error
Asynchronous Mode	$B = \frac{\phi \times 10^6}{64 \times 2^{2n-1} \times (N+1)}$	$\text{Error (\%)} = \{ \frac{\phi \times 10^6}{B \times 64 \times 2^{2n-1} \times (N+1)} - 1 \} \times 100$

Note: B: Bit rate (bit/s)
N: BRR setting for baud rate generator ($0 \leq N \leq 255$)
ϕ: Operating frequency (MHz)
n: Determined by the SMR settings shown in the following tables.

SMR Setting

CKS1	CKS0	n
0	0	0
0	1	1
1	0	2
1	1	3

Table 11.3 shows sample N settings in BRR in asynchronous mode. Table 11.4 shows the maximum bit rate for each frequency in asynchronous mode.

Table 11.3 BRR Settings for Various Bit Rates (Asynchronous Mode) (1)

	Operating Frequency ϕ (MHz)								
	10			12			12.288		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	177	–0.25	2	212	0.03	2	217	0.08
150	2	129	0.16	2	155	0.16	2	159	0.00
300	2	64	0.16	2	77	0.16	2	79	0.00
600	1	129	0.16	1	155	0.16	1	159	0.00
1200	1	64	0.16	1	77	0.16	1	79	0.00
2400	0	129	0.16	0	155	0.16	0	159	0.00
4800	0	64	0.16	0	77	0.16	0	79	0.00
9600	0	32	–1.36	0	38	0.16	0	39	0.00
19200	0	15	1.73	0	19	–2.34	0	19	0.00
31250	0	9	0.00	0	11	0.00	0	11	2.40
38400	0	7	1.73	0	9	–2.34	0	9	0.00

	Operating Frequency ϕ (MHz)											
	14			14.7456			16			17.2032		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	248	–0.17	3	64	0.70	3	70	0.03	3	75	0.48
150	2	181	0.16	2	191	0.00	2	207	0.16	2	223	0.00
300	2	90	0.16	2	95	0.00	2	103	0.16	2	111	0.00
600	1	181	0.16	1	191	0.00	1	207	0.16	1	223	0.00
1200	1	90	0.16	1	95	0.00	1	103	0.16	1	111	0.00
2400	0	181	0.16	0	191	0.00	0	207	0.16	0	223	0.00
4800	0	90	0.16	0	95	0.00	0	103	0.16	0	111	0.00
9600	0	45	–0.93	0	47	0.00	0	51	0.16	0	55	0.00
19200	0	22	–0.93	0	23	0.00	0	25	0.16	0	27	0.00
31250	0	13	0.00	0	14	–1.70	0	15	0.00	0	16	1.20
38400	—	—	—	0	11	0.00	0	12	0.16	0	13	0.00

Table 11.3 BRR Settings for Various Bit Rates (Asynchronous Mode) (2)

	Operating Frequency ϕ (MHz)											
	18			19.6608			20			25		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	79	–0.12	3	86	0.31	3	88	–0.25	3	110	–0.02
150	2	233	0.16	2	255	0.00	3	64	0.16	3	80	–0.47
300	2	116	0.16	2	127	0.00	2	129	0.16	2	162	0.15
600	1	233	0.16	1	255	0.00	2	64	0.16	2	80	–0.47
1200	1	116	0.16	1	127	0.00	1	129	0.16	1	162	0.15
2400	0	233	0.16	0	255	0.00	1	64	0.16	1	80	–0.47
4800	0	116	0.16	0	127	0.00	0	129	0.16	0	162	0.15
9600	0	58	–0.69	0	63	0.00	0	64	0.16	0	80	–0.47
19200	0	28	1.02	0	31	0.00	0	32	–1.36	0	40	–0.76
31250	0	17	0.00	0	19	–1.70	0	19	0.00	0	24	0.00
38400	0	14	–2.34	0	15	0.00	0	15	1.73	0	19	1.73

	Operating Frequency ϕ (MHz)					
	30			33		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)
110	3	132	0.13	3	145	0.33
150	3	97	–0.35	3	106	0.39
300	2	194	0.16	2	214	–0.07
600	2	97	–0.35	2	106	0.39
1200	1	194	0.16	1	214	–0.07
2400	1	97	–0.35	1	106	0.39
4800	0	194	0.16	0	214	–0.07
9600	0	97	–0.35	0	106	0.39
19200	0	48	–0.35	0	53	–0.54
31250	0	29	0	0	32	0
38400	0	23	1.73	0	26	–0.54

Table 11.4 Maximum Bit Rate for Each Frequency (Asynchronous Mode)

ϕ (MHz)	Maximum Bit Rate (bit/s)	n	N
10	312500	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
19.6608	614400	0	0
20	625000	0	0
25	781250	0	0
30	937500	0	0
33	1031250	0	0

11.4 Operation in Asynchronous Mode

Figure 11.2 shows the general format for asynchronous serial communication. One frame consists of a start bit (low level), followed by transfer data, a parity bit, and finally stop bits (high level). In asynchronous serial communication, the transmission line is usually held in the mark state (high level). The SCI monitors the transmission line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication. In asynchronous serial communication, the communication line is usually held in the mark state (high level). The SCI monitors the communication line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

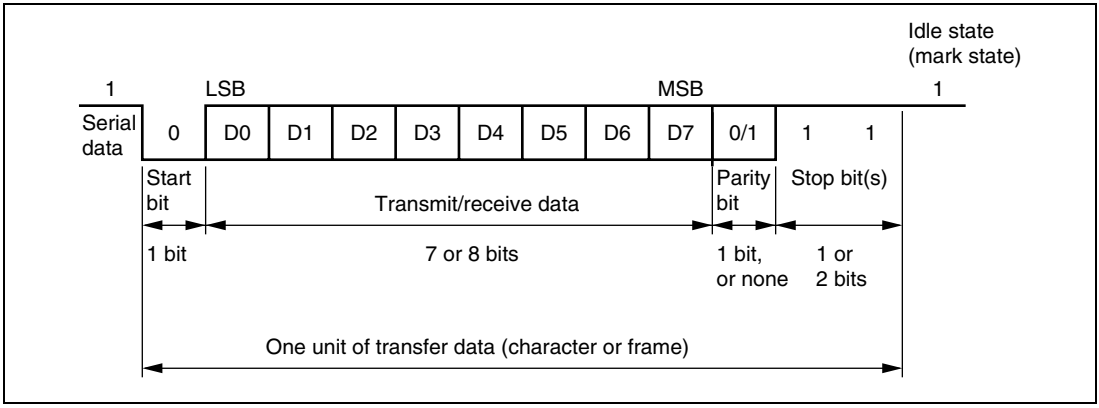


Figure 11.2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)

11.4.1 Data Transfer Format

Table 11.5 shows the data transfer formats that can be used in asynchronous mode. Any of eight transfer formats can be selected according to the SMR setting.

Table 11.5 Serial Transfer Formats (Asynchronous Mode)

SMR Settings			Serial Transfer Format and Frame Length
CHR	PE	STOP	1 2 3 4 5 6 7 8 9 10 11 12
0	0	0	S 8-bit data STOP
0	0	1	S 8-bit data STOP STOP
0	1	0	S 8-bit data P STOP
0	1	1	S 8-bit data P STOP STOP
1	0	0	S 7-bit data STOP
1	0	1	S 7-bit data STOP STOP
1	1	0	S 7-bit data P STOP
1	1	1	S 7-bit data P STOP STOP

Legend

S : Start bit
STOP : Stop bit
P : Parity bit

11.4.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI operates on a basic clock with a frequency of 16 times the bit rate. In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Receive data is latched at the middle of each bit by sampling the data at the rising edge of the 8th pulse of the basic clock as shown in figure 11.3. Thus the reception margin in asynchronous mode is given by formula (1) below.

$$M = \{ (0.5 - \frac{1}{2N}) - (L - 0.5) F - \frac{|D - 0.5|}{N} (1 + F) \} \times 100 \ [\%] \quad \text{... Formula (1)}$$

Where M: Reception Margin
N: Ratio of bit rate to clock (N = 16)
D: Clock duty cycle (D = 0.5 to 1.0)
L: Frame length (L = 9 to 12)
F: Absolute value of clock rate deviation

Assuming values of F = 0 and D = 0.5 in formula (1), a reception margin is given by formula below.

$M = \{0.5 - 1/(2 \times 16)\} \times 100 \ [\%] = 46.875\%$

However, this is only the computed value, and a margin of 20% to 30% should be allowed in system design.

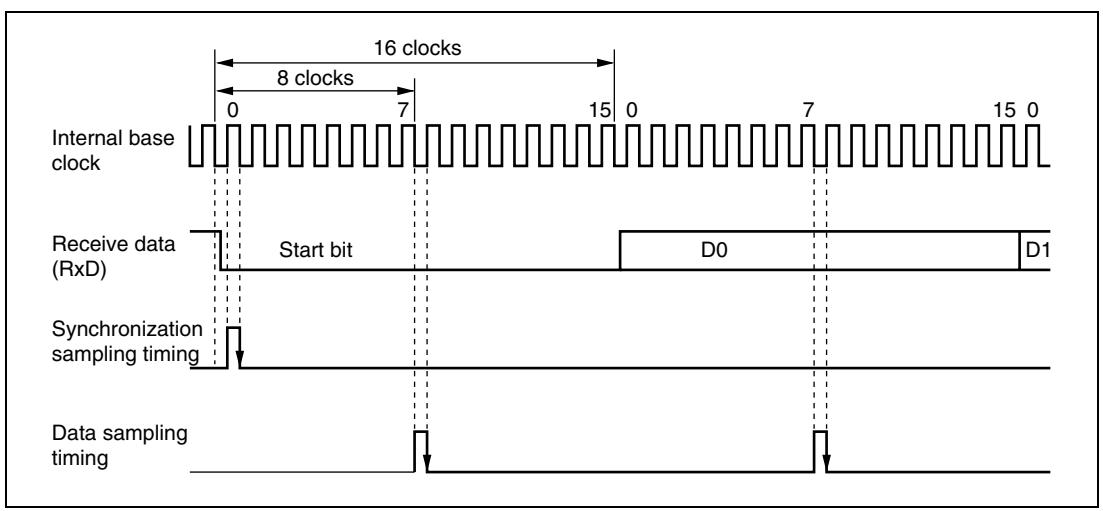


Figure 11.3 Receive Data Sampling Timing in Asynchronous Mode

11.4.3 Clock

An internal clock generated by the on-chip baud rate generator can be selected as the SCI's serial clock, according to the setting of the C/$\overline{A}$ bit in SMR and the CKE1 and CKE0 bits in SCR.

11.4.4 SCI Initialization (Asynchronous Mode)

Before transmitting and receiving data, you should first clear the TE and RE bits in SCR to 0, then initialize the SCI as shown in figure 11.4. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change. When the TE bit is cleared to 0, the TDRE flag is set to 1. Note that clearing the RE bit to 0 does not initialize the contents of the RDRF, PER, FER, and ORER flags, or the contents of RDR.

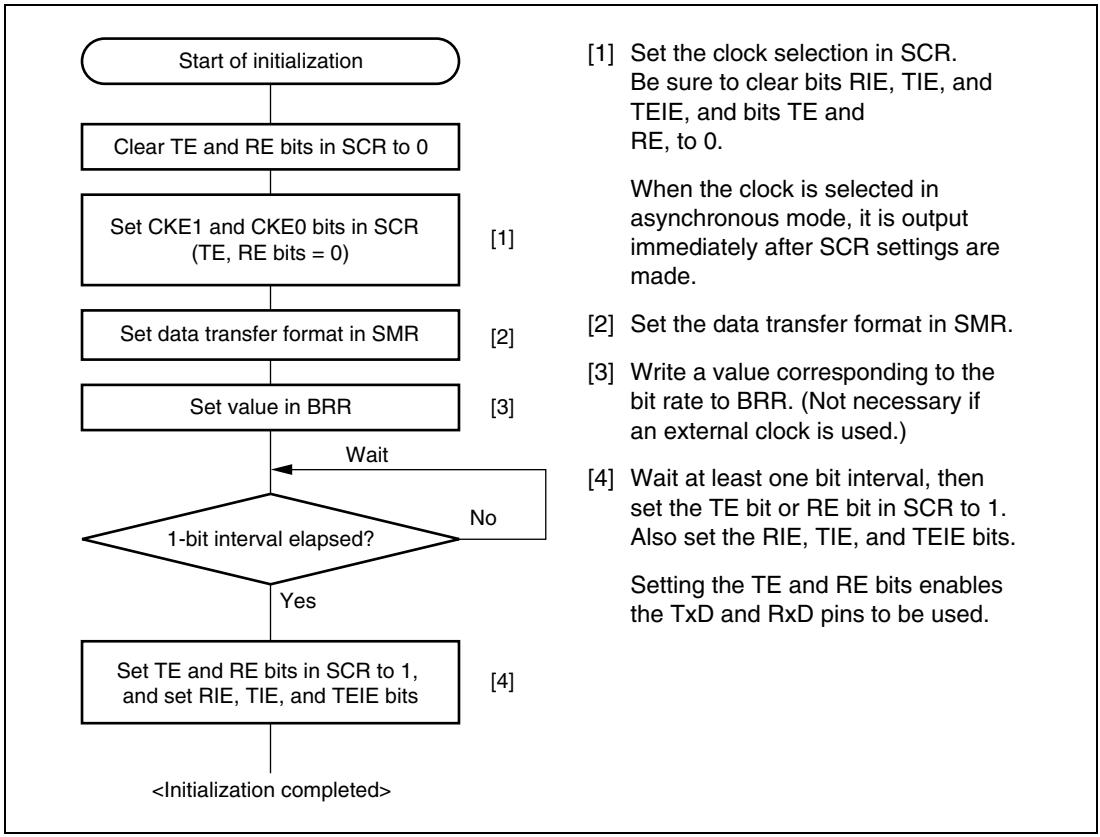


Figure 11.4 Sample SCI Initialization Flowchart

11.4.5 Data Transmission (Asynchronous Mode)

Figure 11.5 shows an example of the operation for transmission in asynchronous mode. In transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR, and if is cleared to 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a transmit data empty interrupt request (TXI) is generated. Because the TXI interrupt routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. Data is sent from the TxD pin in the following order: start bit, transmit data, parity bit (may be omitted depending on the format), and stop bit.
4. The SCI checks the TDRE flag at the timing for sending the stop bit.
5. If the TDRE flag is 0, the data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.
6. If the TDRE flag is 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the "mark state" is entered in which 1 is output. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

Figure 11.6 shows a sample flowchart for transmission in asynchronous mode.

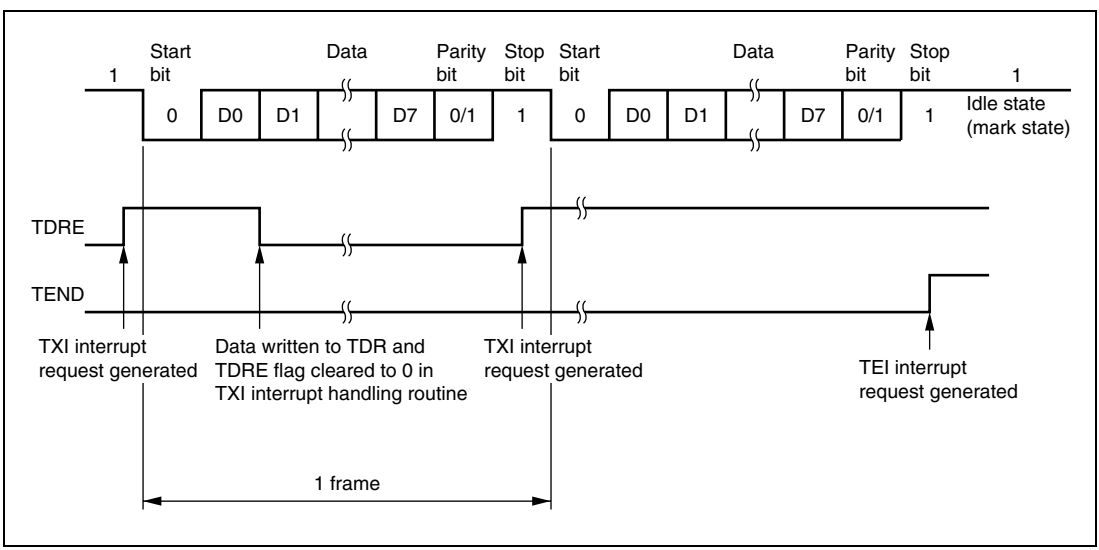
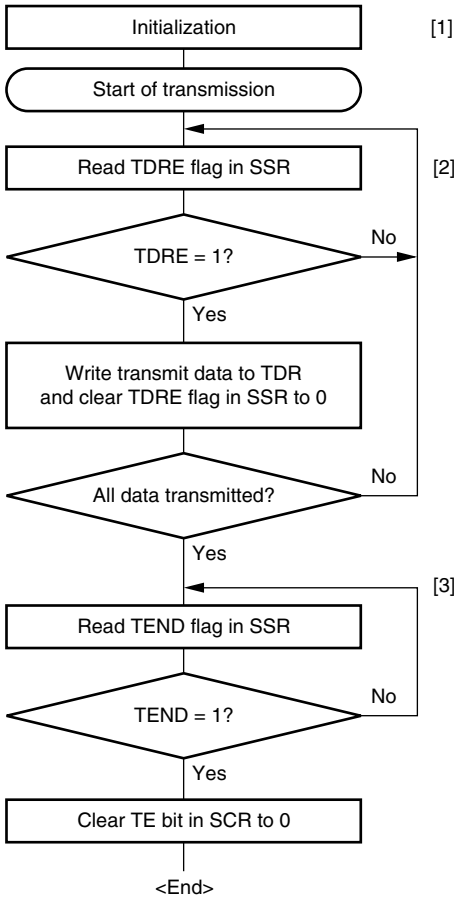


Figure 11.5 Example of Operation in Transmission in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit)



[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin.
After the TE bit is set to 1, a frame of 1s is output, and transmission is enabled.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0.

Figure 11.6 Sample Serial Transmission Flowchart

11.4.6 Serial Data Reception (Asynchronous Mode)

Figure 11.7 shows an example of the operation for reception in asynchronous mode. In serial reception, the SCI operates as described below.

1. The SCI monitors the communication line, and if a start bit is detected, performs internal synchronization, receives receive data in RSR, and checks the parity bit and stop bit.
2. If an overrun error (when reception of the next data is completed while the RDRF flag is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If a parity error is detected, the PER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
4. If a framing error (when the stop bit is 0) is detected, the FER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
5. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.

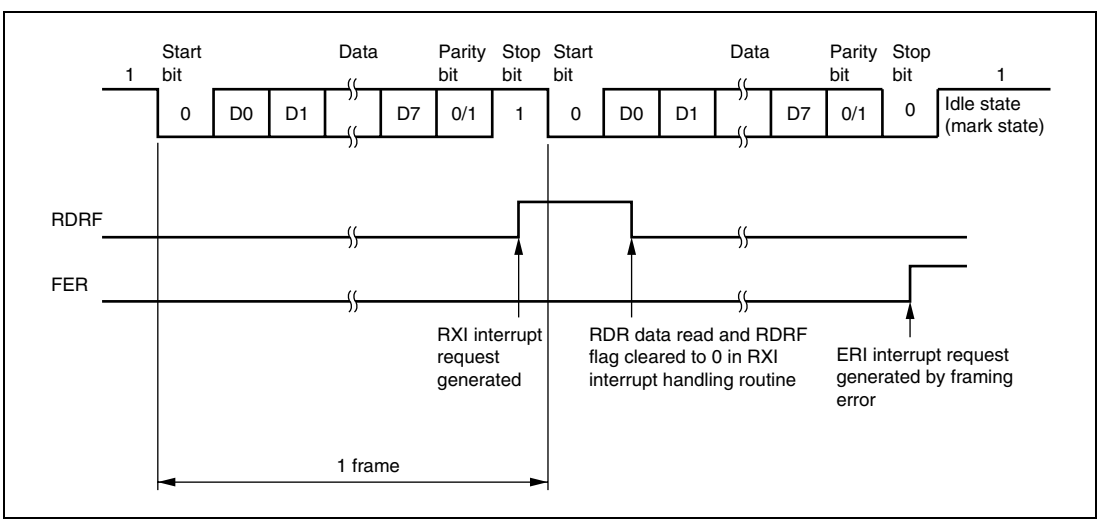


Figure 11.7 Example of SCI Operation in Reception (Example with 8-Bit Data, Parity, One Stop Bit)

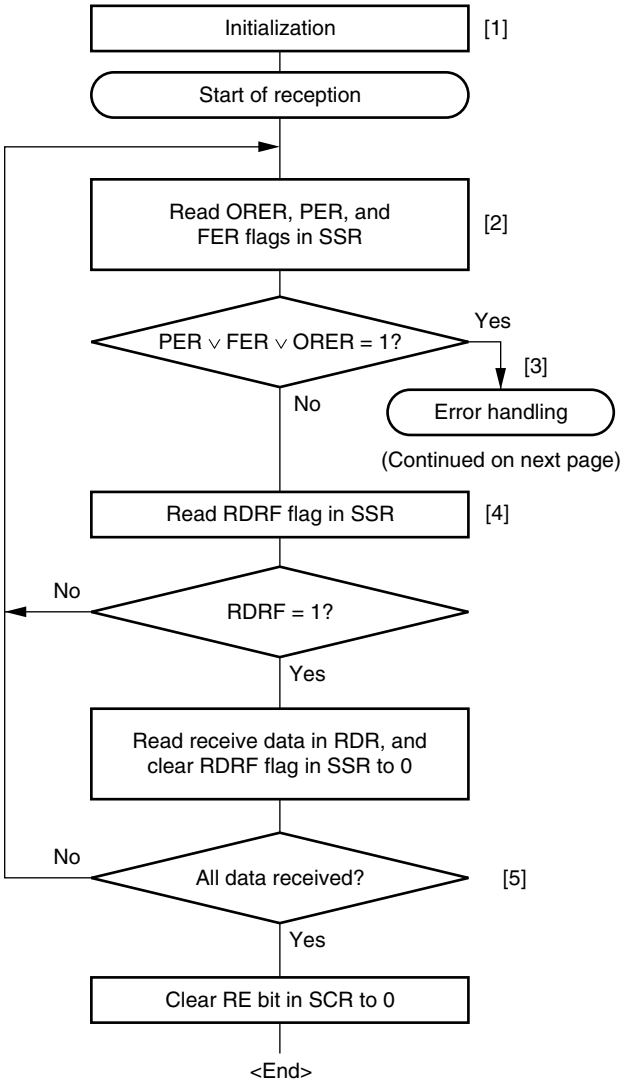
Table 11.6 shows the states of the SSR status flags and receive data handling when a receive error is detected. If a receive error is detected, the RDRF flag retains its state before receiving data. Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER,

FER, PER, and RDRF bits to 0 before resuming reception. Figure 11.8 shows a sample flowchart for serial data reception.

Table 11.6 SSR Status Flags and Receive Data Handling

SSR Status Flag					
RDRF*	**ORER**	**FER**	**PER**	**Receive Data**	**Receive Error Type**
1	1	0	0	Lost	Overrun error
0	0	1	0	Transferred to RDR	Framing error
0	0	0	1	Transferred to RDR	Parity error
1	1	1	0	Lost	Overrun error + framing error
1	1	0	1	Lost	Overrun error + parity error
0	0	1	1	Transferred to RDR	Framing error + parity error
1	1	1	1	Lost	Overrun error + framing error + parity error

Note: The RDRF flag retains its state before data reception.



[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error handling:
If a receive error occurs, read the ORER, PER, and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER, PER, and FER flags are all cleared to 0. Reception cannot be resumed if any of these flags are set to 1.

[4] SCI status check and receive data read :
Read SSR and check that RDRF = 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the stop bit for the current frame is received, read the RDRF flag, read RDR, and clear the RDRF flag to 0.

Figure 11.8 Sample Serial Reception Data Flowchart (1)

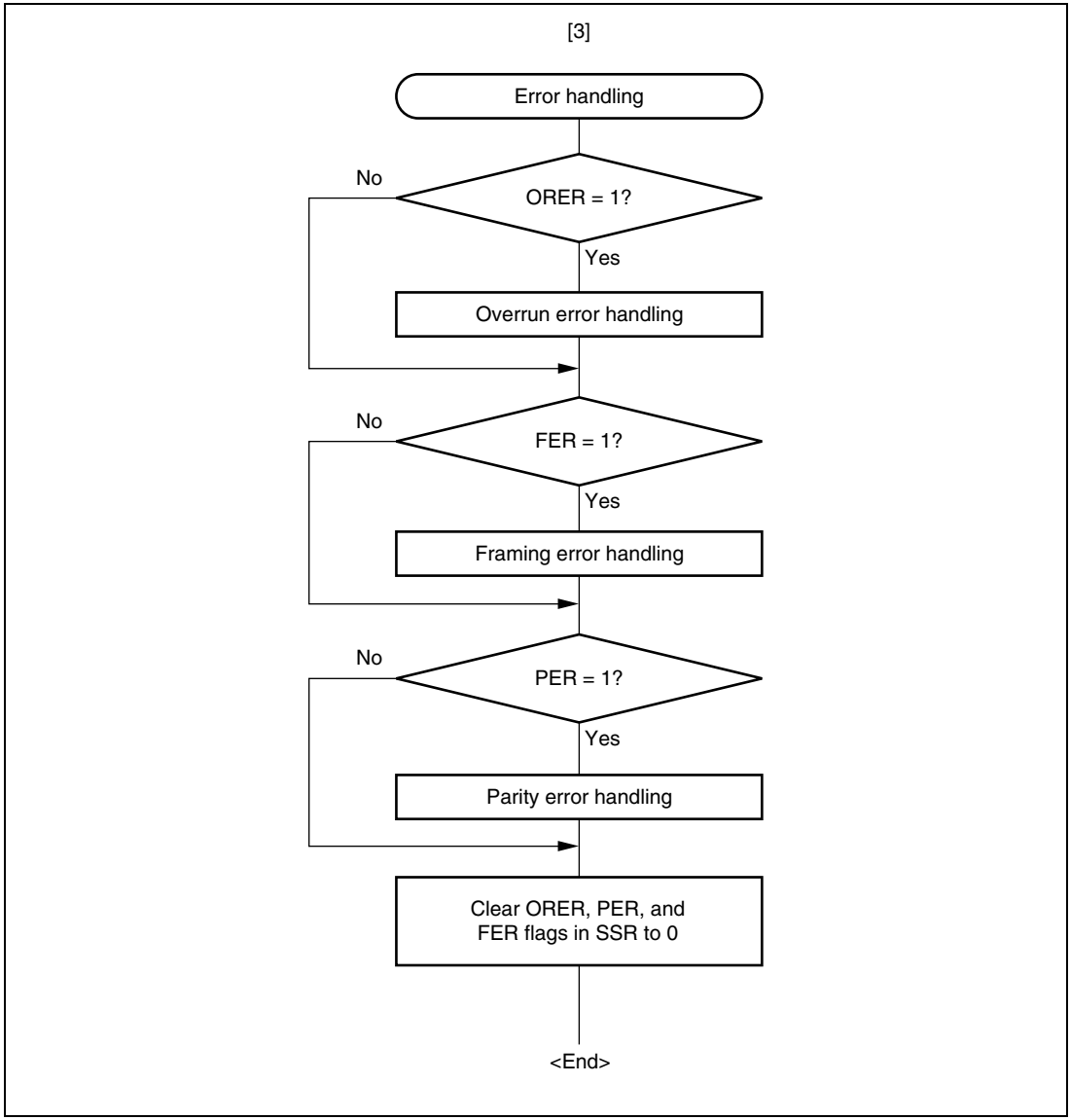


Figure 11.8 Sample Serial Reception Data Flowchart (2)

11.5 Interrupt Sources

11.5.1 Interrupts in Normal Serial Communication Interface Mode

Table 11.7 shows the interrupt sources in normal serial communication interface mode. A different interrupt vector is assigned to each interrupt source, and individual interrupt sources can be enabled or disabled using the enable bits in SCR.

When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR is set to 1, an ERI interrupt request is generated. A TEI interrupt is generated when the TEND flag is set to 1 while the TEIE bit is set to 1. If a TEI interrupt and a TXI interrupt are generated simultaneously, the TXI interrupt has priority for acceptance. However, note that if the TDRE and TEND flags are cleared simultaneously by the TXI interrupt routine, the SCI cannot branch to the TEI interrupt routine later.

Table 11.7 SCI Interrupt Sources

Channel	Name	Interrupt Source	Interrupt Flag	Priority
0	ERI0	Receive Error	ORER, FER, PER	High
	RXI0	Receive Data Full	RDRF	
	TXI0	Transmit Data Empty	TDRE	
	TEI0	Transmission End	TEND	Low

11.6 Usage Notes

11.6.1 Module Stop Mode Setting

SCI operation can be disabled or enabled using the module stop control register. The initial setting is for SCI operation to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 16, Power-Down Modes.

11.6.2 Relation between Writes to TDR and the TDRE Flag

The TDRE flag in SSR is a status flag that indicates that transmit data has been transferred from TDR to TSR. When the SCI transfers data from TDR to TSR, the TDRE flag is set to 1.

Data can be written to TDR regardless of the state of the TDRE flag. However, if new data is written to TDR when the TDRE flag is cleared to 0, the data stored in TDR will be lost since it has not yet been transferred to TSR. It is therefore essential to check that the TDRE flag is set to 1 before writing transmit data to TDR.

11.6.3 Operation in Case of Mode Transition

- Transmission

  Operation should be stopped (by clearing TE, TIE, and TEIE to 0) before making a module stop mode or software standby mode transition. TSR, TDR, and SSR are reset. The output pin states in module stop mode or software standby mode depend on the port settings, and become high-level output after the relevant mode is cleared. If a transition is made during transmission, the data being transmitted will be undefined.

  When transmitting without changing the transmit mode after the relevant mode is cleared, transmission can be started by setting TE to 1 again, and performing the following sequence: SSR read → TDR write → TDRE clearance. To transmit with a different transmit mode after clearing the relevant mode, the procedure must be started again from initialization.

  Figure 11.9 shows a sample flowchart for mode transition during transmission.

- Reception

  Receive operation should be stopped (by clearing RE to 0) before making a module stop mode or software standby mode transition. RSR, RDR, and SSR are reset. If a transition is made during reception, the data being received will be invalid.

  To continue receiving without changing the reception mode after the relevant mode is cleared, set RE to 1 before starting reception. To receive with a different receive mode, the procedure must be started again from initialization.

  Figure 11.10 shows a sample flowchart for mode transition during reception.

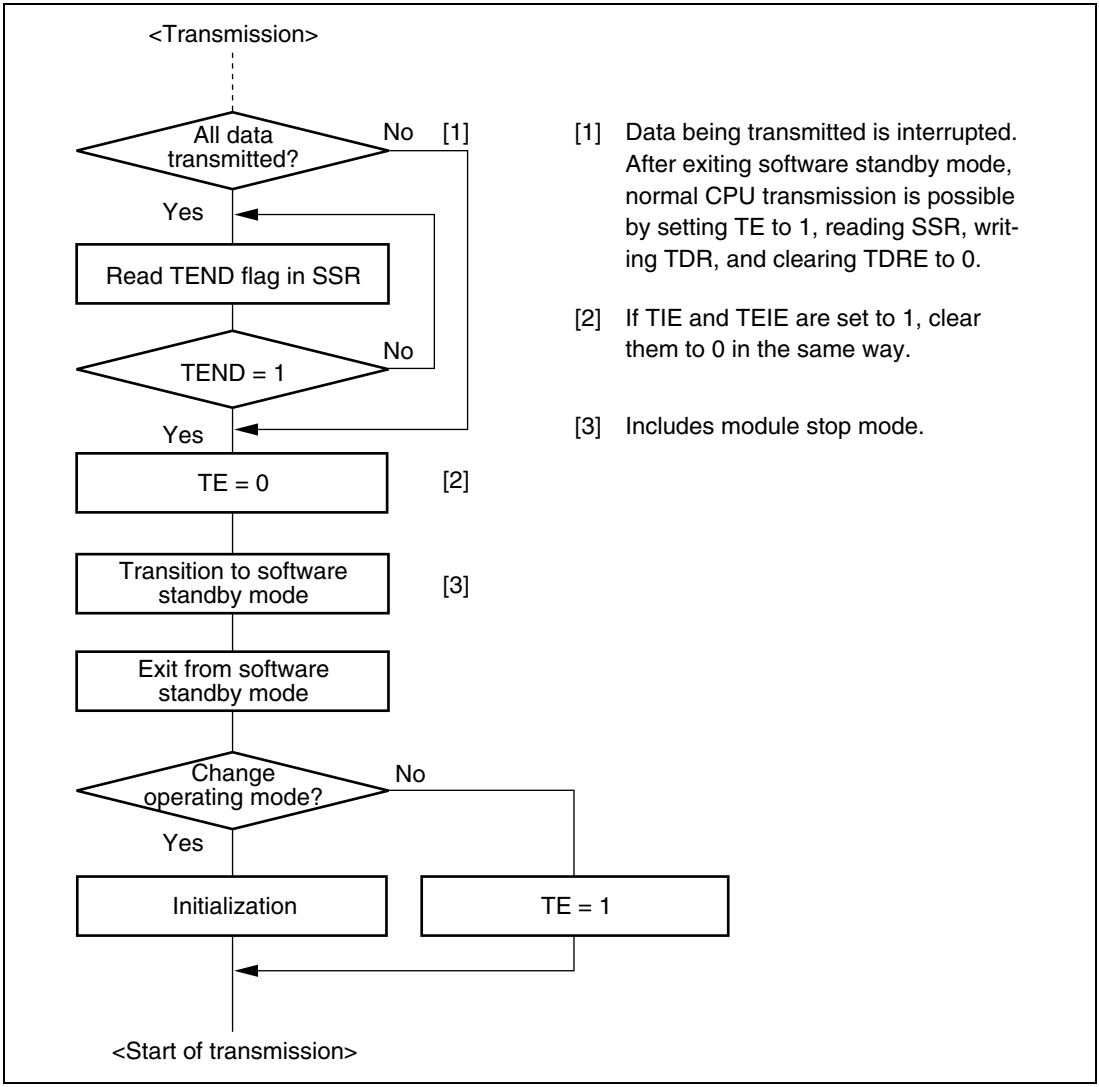


Figure 11.9 Sample Flowchart for Mode Transition during Transmission

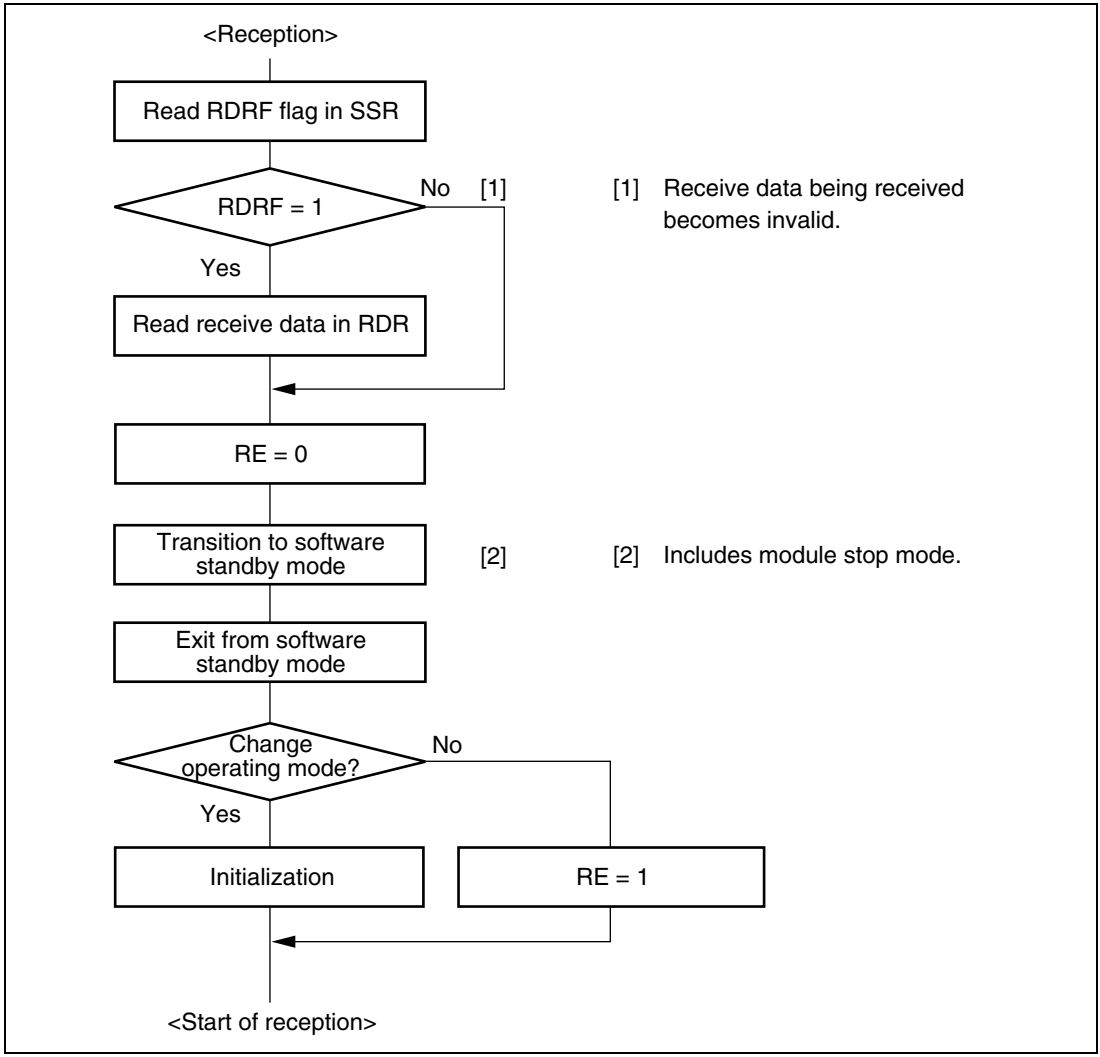


Figure 11.10 Sample Flowchart for Mode Transition during Reception

Section 12 Universal Serial Bus 2 (USB2)

This LSI incorporates a USB2 function module supporting the USB standard. Figure 12.1 shows the block diagram of the USB2.

12.1 Features

- Supports the USB standard version 2.0
  USB standard requests are processed automatically (except for some requests)
  Get Descriptor, Class, and Vendor requests are processed by firmware
- High-speed mode and full-speed mode are supported
- On-chip four endpoints; EP0, EP1, EP2, and EP3

	Max. Packet Size					
Endpoint	Full Speed	High Speed	FIFO Configuration	Transfer Method	Transfer Direction	DMA Transfer
EP0s	8 bytes	8 bytes	Single	Setup	Out	—
EP0i	64 bytes	64 bytes	Single	Control	In	—
EP0o	64 bytes	64 bytes	Single	Control	Out	—
EP1	64 bytes	512 bytes	Dual	Bulk	Out	Possible
EP2	64 bytes	512 bytes	Dual	Bulk	In	Possible
EP3	64 bytes	64 bytes	Single	Interrupt	In	—

- Control, Bulk, and Interrupt transfers are supported
- The maximum packet size in high-speed mode and full-speed mode is switched automatically
- DMA transfer interface
  DMA transfer is enabled for endpoints 1 and 2
- Interrupt interface
  Two interrupt requests (USBI0 and USBI1) are supported as the interrupt request output pins.
  Each interrupt source can be assigned via the internal registers

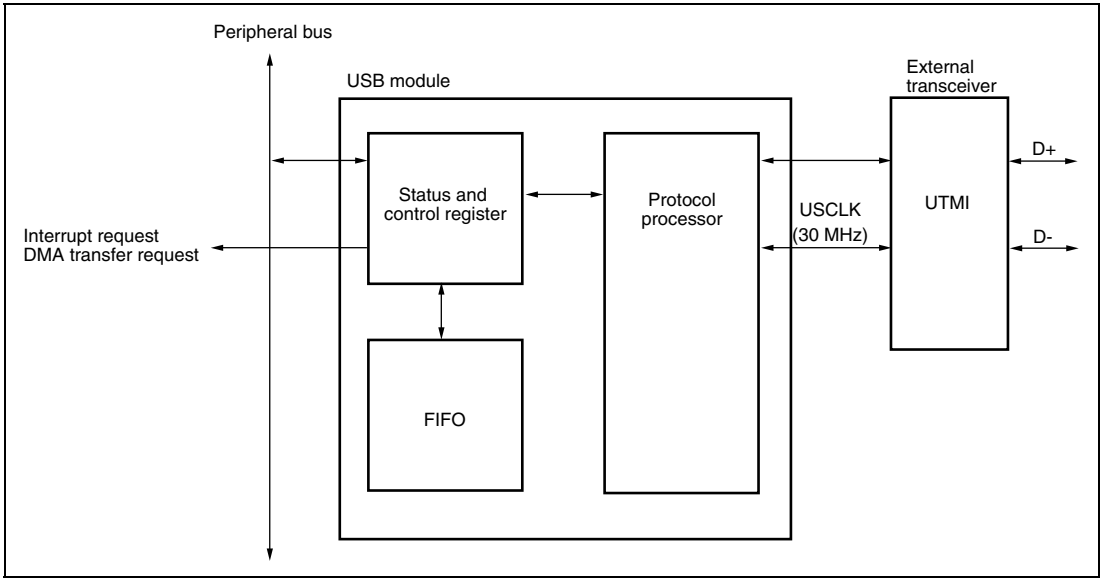


Figure 12.1 Block Diagram of USB2

12.2 Input/Output Signals

Table 12.1 lists the I/O signals in the system bus or transceiver side and their function descriptions when the on-chip bus interface is used in this module.

Table 12.1 Input/Output Signals

Classification	Symbol	I/O	Function
USB bus power supply	USVBUS	Input	USB Bus Power Supply Signal This is a connection or disconnection detection pin for the USB cable. This pin is connected to the VBUS pin in the USB connector. High: VBUS pin = low means disconnection Low: VBUS pin = high means connection Note: This signal should be input after the signal of the VBUS pin of the USB connector is inverted.
Transceiver signal	USCLK	Input	USB Clock This is a USB clock (30 MHz) which is output by the transceiver.
DrVCC	USRXV USRXERR USRXACT USTXRDY USLSTA[1:0]	Input	USB 2.0 Transceiver Input Signal These signals are connected to the USB 2.0 transceiver. They are complied with the UTMI specification. For details, refer to the UTMI Specifications.
	USD[15:0] USWDVLD	I/O	USB 2.0 Transceiver I/O Signal These signals are connected to the USB 2.0 transceiver. They are complied with the UTMI specification. For details, refer to the UTMI Specifications.
	USTSEL USOPM[1:0] USXCVRS USTXV $\overline{\text{USSUSP}}$	Output	USB 2.0 Transceiver Output Signal These signals are connected to the USB 2.0 transceiver. They are complied with the UTMI specification. For details, refer to the UTMI specification.
Monitor pin	$\overline{\text{USRST}}$	Output	This is a monitor pin.

12.3 Register Descriptions

The USB2 has the following registers.

- Interrupt flag register 0 (IFR0)
- Interrupt select register 0 (ISR0)
- Interrupt enable register 0 (IER0)
- EP0o receive data size register (EPSZ0o)
- EP1 receive data size register (EPSZ1)
- EP0i data register (DPDR0i)
- EP0o data register (DPDR0o)
- EP0s data register (DPDR0s)
- EP1 data register (EPDR1)
- EP2 data register (DPDR2)
- EP3 data register (DPDR3)
- Data status register 0 (DASTS0)
- Packet enable register 0i (PKTE0i)
- Packet enable register 2 (PKTE2)
- Packet enable register 3 (PKTE3)
- FIFO clear register 0 (FCLR0)
- Endpoint stall register 0 (EPSTL0)
- DMA set register 0 (DMA0)
- Control register (CTRL)
- Port function control register 3 (PFCR3)
- USB suspend status register (USBSUSP)

12.3.1 Interrupt Flag Register 0 (IFR0)

IFR0 indicates the setup request reception, EP0i, EP0o, EP1, EP2, and EP3 transmission/reception, and bus reset state and monitors a VBUS interrupt flag and USB mode interrupt flag. If the corresponding flag is set to 1, the corresponding interrupt request is output. A flag in this register can be cleared by writing 0 to it. Writing 1 to a flag is invalid and causes no operation. Note that the EP1FULL and EP2EMPTY bits are status bits indicating the FIFO states of the EP1 and EP2. Therefore these bits cannot be cleared. The VBUS MN and MODE MN bits are also status bits so they cannot be cleared.

Bit	Bit Name	Initial Value	R/W	Description
31 to 27	—	All 0	R	Reserved The write value should always be 0.
26	MODE MN1	0	R	USB Mode Status 1 This bit is a status bit which indicates the USB transfer mode. This bit is used as two bits with the MODE MN0 bit. 0: At a reset or when the cable is disconnected 1: Full-speed mode (12 Mbps) 2: High-speed mode (480 Mbps) 3: Chirp mode Refer to section 12.8.12, USB Bus Idle in High-Speed Mode. This bit cannot be cleared because this bit is a status bit.
25	MODE MN0	0	R	USB Mode Status 0 This bit is a status bit which indicates the USB transfer mode. This bit is used as two bits with the MODE MN1 bit. This bit cannot be cleared because this bit is a status bit.

Bit	Bit Name	Initial Value	R/W	Description
24	MODE F	0	R/W	USB Transfer Mode Change Detection [Setting condition] This bit is set to 1 when the USB mode status bits are changed. [Clearing conditions] • At a reset • When 0 is written to this bit
23 to 18	—	0	R	Reserved The write value should always be 0.
17	VBUS MN	0	R/W	USB Connection Status This bit is a status bit which monitors the state of the USVBUS pin. This bit reflects the state of the USVBUS pin. 1: The USVBUS pin is connected. 0: The USVBUS pin is not connected. This bit cannot be cleared because this bit is a status bit.
16	VBUS F	0	R/W	USB Disconnection Detection [Setting condition] This bit is set to 1 when the function is connected/disconnected to/from the USB bus. To detect connection/disconnection, the VBUS pin is used. [Clearing conditions] • At a reset • When 0 is written to this bit
15 to 10	—	0	R	Reserved The write value should always be 0.
9	EP3TR	0	R/W	EP3 Transfer Request [Setting condition] This bit is set to 1 if there is no valid data in the FIFO when an IN token is sent from the host to EP3. [Clearing conditions] • At a reset • When 0 is written to this bit

Bit	Bit Name	Initial Value	R/W	Description
8	EP3TS	0	R/W	EP3 Transmit Complete [Setting condition] This bit is set to 1 if the data written in EP3 is transmitted to the host normally and the ACK handshake is returned. [Clearing conditions] At a reset When 0 is written to this bit
7	BRST	0	R/W	Bus Reset [Setting condition] This bit is set to 1 when the bus reset signal is detected on the USB bus. [Clearing conditions] At a reset When 0 is written to this bit
6	EP2TR	0	R/W	EP2 Transfer Request [Setting condition] This bit is set to 1 if there is no valid data in both FIFOs when an IN token is sent from the host to EP2. [Clearing conditions] At a reset When 0 is written to this bit
5	EP2EMPTY	0	R	EP2 FIFO Empty*[2] [Setting conditions] • At a reset • EP2 has a dual FIFO configuration. This bit is set to 1 if there is no valid data at least in the single FIFO. If data is full in both FIFOs, this bit is set to 0. [Clearing condition] This bit cannot be cleared because this bit is a status bit.
4	EP1FULL	0	R	EP1 FIFO Full*[1] [Setting condition] EP1 has a dual FIFO configuration. This bit is set to 1 if data is full at least in the single FIFO. If there is no valid data in both FIFOs, this bit is set to 0. [Clearing condition] This bit cannot be cleared because this bit is a status bit.

Bit	Bit Name	Initial Value	R/W	Description
3	SETUPTS	0	R/W	Setup Request Receive Complete [Setting condition] This bit is set to 1 if EP0s normally receives 8-byte data to be decoded by the function from the host and returns the ACK handshake to the host. [Clearing conditions] At a reset When 0 is written to this bit
2	EP0oTS	0	R/W	EP0o Receive Complete [Setting condition] This bit is set to 1 if EP0o receives data from the host normally and returns the NYET or ACK handshake to the host. [Clearing conditions] At a reset When 0 is written to this bit
1	EP0iTR	0	R/W	EP0i Transfer Request [Setting condition] This bit is set to 1 if there is no valid data in the FIFO when an IN token is sent from the host to EP0i [Clearing conditions] At a reset When 0 is written to this bit
0	EP0iTS	0	R/W	EP0i Transmit Complete [Setting condition] This bit is set to 1 if the data written in EP0i is transmitted to the host normally and the ACK handshake is returned. [Clearing conditions] At a reset When 0 is written to this bit

Notes: 1. FIFO Full
In case of IN FIFO: The data which can be transmitted is in the FIFO.
In case of OUT FIFO: The data which is valid is in the FIFO.

2. FIFO Empty
In case of IN FIFO: The data which can be transmitted is not in the FIFO.
In case of OUT FIFO: The data which is valid is not in the FIFO.

12.3.2 Interrupt Select Register 0 (ISR0)

ISR0 sets interrupt requests indicated in the interrupt flag register. When the corresponding bit is cleared to 0, the USBI0 interrupt request is output. When the corresponding bit is set to 1, the USBI1 interrupt request is output. In the initial value, each interrupt source in the interrupt flag register is requested from USBI0.

Bit	Bit Name	Initial Value	R/W	Description
31 to 25	—	All 0	R	Reserved The write value should always be 0.
24	MODE F	0	R/W	Selects the MODE F interrupt.
23 to 17	—	All 0	R	Reserved The write value should always be 0.
16	VBUS F	0	R/W	Selects the VBUS F interrupt.
15 to 10	—	All 0	R	Reserved The write value should always be 0.
9	EP3TR	0	R/W	Selects the EP3TR interrupt.
8	EP3TS	0	R/W	Selects the EP3TS interrupt.
7	BRST	0	R/W	Selects the BRST interrupt.
6	EP2TR	0	R/W	Selects the EP2TR interrupt.
5	EP2EMPTY	0	R/W	Selects the EP2EMPTY interrupt.
4	EP1FULL	0	R/W	Selects the EP1FULL interrupt
3	SETUPTS	0	R/W	Selects the SETUPTS interrupt.
2	EP0oTS	0	R/W	Selects the EP0oTS interrupt.
1	EP0iTR	0	R/W	Selects the EP0iTR interrupt.
0	EP0iTS	0	R/W	Selects the EP0iTS interrupt.

12.3.3 Interrupt Enable Register 0 (IER0)

IER0 enables the interrupt request indicated in the interrupt flag register. When an interrupt flag is set while the corresponding bit in IER0 is set to 1, an interrupt request selected by the interrupt select register is asserted.

Bit	Bit Name	Initial Value	R/W	Description
31 to 25	—	All 0	R	Reserved The write value should always be 0.
24	MODE F	0	R/W	Enables the MODE F interrupt.
23 to 17	—	All 0	R	Reserved The write value should always be 0.
16	VBUS F	0	R/W	Enables the VBUS F interrupt.
15 to 10	—	All 0	R	Reserved The write value should always be 0.
9	EP3TR	0	R/W	Enables the EP3TR interrupt.
8	EP3TS	0	R/W	Enables the EP3TS interrupt.
7	BRST	0	R/W	Enables the BRST interrupt.
6	EP2TR	0	R/W	Enables the EP2TR interrupt.
5	EP2EMPTY	0	R/W	Enables the EP2EMPTY interrupt.
4	EP1FULL	0	R/W	Enables the EP1FULL interrupt
3	SETUPTS	0	R/W	Enables the SETUPTS interrupt.
2	EP0oTS	0	R/W	Enables the EP0oTS interrupt.
1	EP0iTR	0	R/W	Enables the EP0iTR interrupt.
0	EP0iTS	0	R/W	Enables the EP0iTS interrupt.

12.3.4 EP0o Receive Data Size Register (EPSZ0o)

EPSZ0o is a receive data size register for endpoint 0o. EPSZ0o indicates the number of bytes of data to be received from the host.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	—	R	EP0o Receive Data Size

12.3.5 EP1 Receive Data Size Register (EPSZ1)

EPSZ1 is a receive data size register for endpoint 1. EPSZ1 indicates the number of bytes of data to be received from the host. The FIFO for endpoint 1 has a dual-FIFO configuration. The data size indicated by this register refers to the currently selected FIFO.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	—	R	EP1 Receive Data Size

12.3.6 EP0i Data Register (EPDR0i)

EPDR0i is a 64-byte transmit FIFO buffer for endpoint 0. EPDR0i stores number of packets of transmit data for control-in. If one packet of data is written and number of transmit data is written in the packet enable register 0i (PKTE0i), transmit data is valid. If data is transmitted and then the ACK handshake is returned from the host, the EP0iTS bit in IFR0 is set. EPDR0i can be initialized by setting the EP0iCLR bit in the FIFO clear register 0. When the setup is received, EPDR0i is cleared. After the setup data is received, transmission is impossible until the SETUP TS bit is cleared.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	All 0	W	64-Byte Transmit FIFO Buffer for EP0

12.3.7 EP0o Data Register (EPDR0o)

EPDR0o is a 64-byte receive FIFO buffer for endpoint 0 and has a single FIFO buffer. When reception is completed, the USB returns the NYET handshake (high-speed mode) or ACK handshake (full-speed mode) to the host. EPDR0o stores receive data for endpoint 0 except for the setup request. When data is received normally, the EP0oTS bit in IFR0 is set and the number of receive bytes is indicated in the EP0o receive data size register. After the setup data is received, reception is impossible until the SETUP TS bit is cleared.

Though the 0-length packet can be received, the ACK handshake (both high-speed and full-speed modes) is returned to the host and data is ignored. However, the EP0oTS flag in IFR0 is set.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	All 0	R	64-Byte Receive FIFO Buffer for EP0

12.3.8 EP0s Data Register (EPDR0s)

EPDR0s is a data register only for the setup request for endpoint 0. EPDR0s stores 8-byte request data sent from the host in setup stage. Note that only request data to be processed by the microcomputer is received. When a request processed by the USB automatically is received, data is not stored.

When data reception is started in the next setup stage during reading, data is overwritten unconditionally.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	All 0	R	Data Register only for EP0 Setup Request

12.3.9 EP1 Data Register (EPDR1)

EPDR1 consists of two 512-byte receive FIFO buffers for endpoint 1. The size of EPDR1 is as twice as the maximum packet size in high-speed mode and has a dual-FIFO configuration. When there is no data in the single FIFO buffer, the USB returns the ACK handshake (both high-speed and full-speed modes) to the host. When reception is completed and data is full in the both FIFO buffers, the USB returns the NYET handshake (high-speed mode) or ACK handshake (full-speed mode) to the host. The number of receive bytes is indicated in EPSZ1. DMA transfer can be performed for receive data in EPDR1. EPDR1 can be initialized by setting the EP1CLR bit in the FIFO clear register 0.

Though the 0-length packet can be received, the FIFO is not full, the ACK handshake (both high-speed and full-speed modes) is returned to the host, and data is ignored. Therefore the EP1 FULL status flag in IFR0 is not set.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	All 0	W	Two 512-byte Receive FIFO Buffers for EP1

12.3.10 EP2 Data Register (EPDR2)

EPDR2 is a 1024-byte transmit FIFO buffer for endpoint 2. The size of EPDR2 is as twice as the maximum packet size in high-speed mode and has a dual-FIFO configuration. When transmit data is written in EPDR2 and number of transmit data is written in the packet enable register 2 (PKTE2), one packet of transmit data is valid and the buffer is switched. DMA transfer can be performed for transmit data to EPDR2. EPDR2 can be initialized by setting the EP2CLR bit in the FIFO clear register 0.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	All 0	R	1024-Byte Transmit FIFO Buffer for EP2

12.3.11 EP3 Data Register (EPDR3)

EPDR3 is a 64-byte transmit FIFO buffer for endpoint 3. EPDR3 stores one packet of transmit data in the interrupt transfer for endpoint 3. If one packet of data is written and number of transmit data is written in the packet enable register 3 (PKTE3), transmit data is valid. If one packet of data is transmitted normally and the ACK handshake is returned from the host, the EP3TS bit in IFR0 is set. EPDR3 can be initialized by setting the EP3CLR bit in the FIFO clear register 0.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	All 0	R	64-Byte Transmit FIFO Buffer for EP3

12.3.12 Data Status Register 0 (DASTS0)

DASTS0 indicates whether the IN FIFO data registers contain valid data or not. A bit in DASTS0 is set to 1 when data written to the corresponding IN FIFO becomes valid after the number of transfer bytes is written in the packet enable register. A bit in DASTS0 is cleared to 0 when all valid data is sent to the host. For endpoint 2, having a dual-FIFO configuration, the corresponding bit in DASTS0 is cleared to 0 when both FIFOs become empty.

Bit	Bit Name	Initial Value	R/W	Description
31 to 6	—	All 0	R	Reserved The write value should always be 0.
5	EP3DE	0	R	EP3 Data Enable Set to 1 when EP3 contains valid data and cleared to 0 when EP3 contains no valid data.
4	EP2DE	0	R	EP2 Data Enable Set to 1 when EP2 contains valid data and cleared to 0 when EP2 contains no valid data.
3 to 1	—	All 0	R	Reserved The write value should always be 0.
0	EP0iDE	0	R	EP0i Data Enable Set to 1 when EP0i contains valid data and cleared to 0 when EP0i contains no valid data.

12.3.13 Packet Enable Register 0i (PKTE0i)

The number of transmit data is written in PKTE0i after writing transmit data in EPDR0i. Then transmit data becomes valid and data is transmitted by the next IN token. Data is not transmitted only by writing data in EPDR0i.

The number of data bytes written in EPDR0i must match the number of transmit data bytes to be written in PKTE0i.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	All 0	W	Number of Transmit Data

12.3.14 Packet Enable Register 2 (PKTE2)

The number of transmit data is written in PKTE2 after writing transmit data in EPDR2. Then transmit data becomes valid and data is transmitted by the next IN token. Data is not transmitted only by writing data in EPDR2.

The number of data bytes written in EPDR2 must match the number of transmit data bytes to be written in PKTE2.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	All 0	W	Number of Transmit Data

12.3.15 Packet Enable Register 3 (PKTE3)

The number of transmit data is written in PKTE3 after writing transmit data in EPDR3. Then transmit data becomes valid and data is transmitted by the next IN token. Data is not transmitted only by writing data in EPDR3.

The number of data bytes written in EPDR3 must match the number of transmit data bytes to be written in PKTE3.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	D31 to D0	All 0	W	Number of Transmit Data

12.3.16 FIFO Clear Register 0 (FCLR0)

FCLR0 is a one-shot register used to clear the FIFO for each endpoint. Writing 1 to a bit clears the data in the corresponding FIFO.

For IN FIFO, writing 1 to a bit in FCLR0 clears the data for which the corresponding bit in the packet enable register is not set to 1 after data write, or data that is validated by setting the corresponding bit in the packet enable register.

For OUT FIFO, writing 1 to a bit in FCLR0 clears data that has been received. EP1 and EP2, having a dual-FIFO configuration, are cleared by entire FIFOs. Note that this trigger does not clear the corresponding interrupt flag. Accordingly, care must be taken not to clear data that is currently being received or transmitted.

Bits 6, 5, and 0 are also used as the status bits. The function of the status bit is described in the lower column of the bit description.

Bit	Bit Name	Initial Value	R/W	Description
31 to 7	—	All 0	W	Reserved The write value should always be 0.
6	EP3CLR	0	W	EP3 Clear 1 is written when clearing EP3 IN FIFO. Writing 0 is invalid and no operation is performed.
			R	EP3 FIFO Clear Status [Setting condition] This bit is set to 1 when the EP3 FIFO is forcibly cleared by the FCLR register. When this bit is set to 1, access to the EP3 FIFO is prohibited. This bit is cleared to 0 automatically after the FIFO is internally cleared. Confirm that this bit is cleared to 0 and then wait for at least four cycles, before accessing to the EP3. [Clearing condition] This bit cannot be cleared because this bit is a status bit.

Bit	Bit Name	Initial Value	R/W	Description
5	EP2CLR	0	W	EP2 Clear 1 is written when clearing EP2 IN FIFO. Writing 0 is invalid and no operation is performed.
			R	EP2 FIFO Clear Status [Setting condition] This bit is set to 1 when the EP2 FIFO is forcibly cleared by the FCLR register. When this bit is set to 1, access to the EP2 FIFO is prohibited. This bit is cleared to 0 automatically after the FIFO is internally cleared. Confirm that this bit is cleared to 0 and then wait for at least four cycles, before accessing to the EP2. [Clearing condition] This bit cannot be cleared because this bit is a status bit.
4	EP1CLR	0	W	EP1 Clear 1 is written when clearing EP1 OUT FIFO. Writing 0 is invalid and no operation is performed.
3, 2	—	All 0	W	Reserved The write value should always be 0.
1	EP0oCLR	0	W	EP0o Clear 1 is written when clearing EP0o OUT FIFO. Writing 0 is invalid and no operation is performed.
0	EP0iCLR	0	W	EP0i Clear 1 is written when clearing EP0i IN FIFO. Writing 0 is invalid and no operation is performed.
			R	EP0i FIFO Clear Status [Setting condition] This bit is set to 1 when the EP0i FIFO is forcibly cleared by the FCLR register. When this bit is set to 1, access to the EP0i FIFO is prohibited. This bit is cleared to 0 automatically after the FIFO is internally cleared. Confirm that this bit is cleared to 0 and then wait for at least four cycles, before accessing to the EP0i. [Clearing condition] This bit cannot be cleared because this bit is a status bit.

12.3.17 Endpoint Stall Register 0 (EPSTL0)

EPSTL0 is used to stall each endpoint. When 1 is written in a bit, the corresponding endpoint returns a stall handshake to the host, following from the next transfer. The stall bit for endpoint 0 is cleared automatically on reception of 8-byte request data for which decoding is performed by the function, and thus the EP0 STL bit is cleared to 0. When the SETUP TS flag in IFR0 is set to 1, a write of 1 to the EP0 STL bit is ignored. For details, refer to section 12.5.8, Stall Operations.

When the ASCE bit in CTRL is set to 1, the EPxSTL (x = 0, 1, 2, 3) bit is automatically cleared. For details, refer to section 12.3.19, Control Register (CTRL).

Bit	Bit Name	Initial Value	R/W	Description
31 to 4	—	All 0	R	Reserved The write value should always be 0.
3	EP3STL	0	R/W	EP3 Stall Sets the EP3 stall state.
2	EP2STL	0	R/W	EP2 Stall Sets the EP2 stall state.
1	EP1STL	0	R/W	EP1 Stall Sets the EP1 stall state.
0	EP0STL	0	R/W	EP0 Stall Sets the EP0 stall state.

12.3.18 DMA Set Register 0 (DMA0)

DMA0 is set when the DMAC dual address transfer is used for data registers for endpoints 1 and 2.

For endpoint 1, if 1 is written in the EP1 DMAE bit, the transfer is requested to the DMAC when the EP1 FIFO is full at least in the single FIFO. That is, when there is valid receive data in the FIFO, the transfer is requested to the DMAC. When all receive data is read and both FIFOs are empty, the transfer is not requested to the DMCA any more.

For endpoint 2, if 1 is written in the EP2 DMAE bit, the transfer is requested to the DMAC when the EP2 FIFO is empty at least in the single FIFO. That is, when there is no valid data in the FIFO, the transfer is requested to the DMAC. When data is written by the microcomputer and both FIFOs are full, the transfer is not requested to the DMCA any more.

Since an interrupt request is not masked automatically, the EP1 FULL and EP2 EMPTY bits in IER0 are cleared to 0 and an interrupt should not be requested by an interrupt pin.

Bit	Bit Name	Initial Value	R/W	Description
31 to 2	—	All 0	R	Reserved The write value should always be 0.
1	EP2DMAE	0	R/W	EP2 DMA Enable Enables the DMA transfer for EP2.
0	EP1DMAE	0	R/W	EP1 DMA Enable Enables the DMA transfer for EP1.

12.3.19 Control Register (CTRL)

CTRL controls the USB module.

Bit	Bit Name	Initial Value	R/W	Description
31 to 2	—	All 0	R	Reserved The write value should always be 0.
1	ASCE	0	R/W	Automatic Stall Clear Enable When this bit is set to 1, the stall handshake is returned to the host and then the stall set bit (EPxSTL (x = 0, 1, 2, 3) bit in EPSTL0) for the returned endpoint is automatically cleared. The automatic stall clear enabling is common to all endpoints. This function cannot be controlled individually for each endpoint. When this bit is cleared to 0, the EPxSTL (x = 0, 1, 2, 3) bit is not automatically cleared. The user needs to clear the EPxSTL (x = 0, 1, 2, 3) bit. To enable this bit, this bit should be set to 1 before the EPxSTL (x = 0, 1, 2, 3) bit in EPSTL0 is set to 1.
0	PULLUPE	0	R/W	Pull-Up Enable Controls pull-up of the PHY complying with the UTMI. If 1 is written in this bit, pull-up is enabled for the PHY.

12.3.20 Port Function Control Register 3 (PFCR3)

PFCR3 controls the USB software reset and suspend reset.

Bit	Bit Name	Initial Value	R/W	Description
7	SUSRIF	0	R/W	Suspend Reset Interrupt Flag For details on operation, refer to section 12.7.2, Software Standby in Suspend Mode.
6	SUSRIE	0	R/W	Suspend Reset Interrupt Enable For details on operation, refer to section 12.7.2, Software Standby in Suspend Mode.
5 to 1	—	All 0	R/W	Reserved These bits can be read from or written to. However, the write value should always be 0.
0	USBSWRST	0	R/W	USB Module Software Reset While this bit is set to 1, the USB module is in the reset state.

12.3.21 USB Suspend Status Register (USBSUSP)

USBSUSP specifies the USB state and enables or disables a USB suspend interrupt request. USBSUSP is initialized to H′00 at a reset or in hardware standby mode.

Bit	Bit Name	Initial Value	R/W	Description
7	USUSMONI	0	R*[1]	USB Suspend Monitor This bit is a monitor bit that indicates whether the USB is in the normal state or in the suspend state. This bit can be read from but cannot be modified. 0: Indicates that the USB is in the normal state. 1: Indicates that the USB is in the suspend state.

Bit	Bit Name	Initial Value	R/W	Description
6	USUSFG	0	R/(W)*[2]	USB Suspend Interrupt Flag This bit is a status flag that indicates the transition from the normal state to the suspend state is made. [Clearing condition] While USUSFG = 1, 0 is written to USUSFG after reading USUSFG [Setting condition] When transition from the normal state to the suspend state is made
5	USUSFGE	0	R/W	USB Suspend Interrupt Enable Enables or disables the USB suspend interrupt request to the CPU. 0: The USB suspend interrupt request is disabled. 1: The USB suspend interrupt request is enabled.
4	USUSOUT	0	R/W	USB Suspend Output Enable Enables or disables the suspend state notification to the external transceiver when the USB enters the suspend state. 0: Not notified that the USB enters the suspend state to the external transceiver. The external pin, $\overline{\text{USSUSP}}$, is always set to 1. 1: Notified that the USB enters the suspend state to the external transceiver. When the USB is in the suspend state, the external pin, $\overline{\text{USSUSP}}$, is cleared to 0.
3 to 0	—	All 0	R/W	Reserved The initial value should not be changed.

Notes: 1. Cannot be modified.
2. Only 0 can be written after reading 1 to clear the flag.

12.4 Interrupt Pins

This module has two interrupt sources. The interrupt select register 0 is used to set the correspondence between interrupt flags and interrupt sources (USBI1 and USBI0). Basically, all processing is possible by using only one interrupt source. If there is an interrupt source whose response needs to be fast, priority can be set by using the second interrupt source.

Among interrupt sources of this module, an interrupt source related to endpoint 0 (bits 3 to 0 in IFR0) must be assigned to the same interrupt source.

12.4.1 USBI0 Interrupt

The USBI0 is an interrupt request for an interrupt source in which 0 is set by the interrupt select register 0. Among interrupt sources assigned to the USBI0, when only one of a corresponding bit in the interrupt flag register 0 is set to 1, an interrupt request occurs.

12.4.2 USBI1 Interrupt

The USBI1 is an interrupt request for an interrupt source in which 1 is set by the interrupt select register 0. Among interrupt sources assigned to the USBI1, when only one of a corresponding bit in the interrupt flag register 0 is set to 1, an interrupt request occurs.

12.5 Communication Operation

12.5.1 USB Cable Connection

If the USB cable enters the connection state from the disconnection state, perform the operation as shown in figure 12.2.

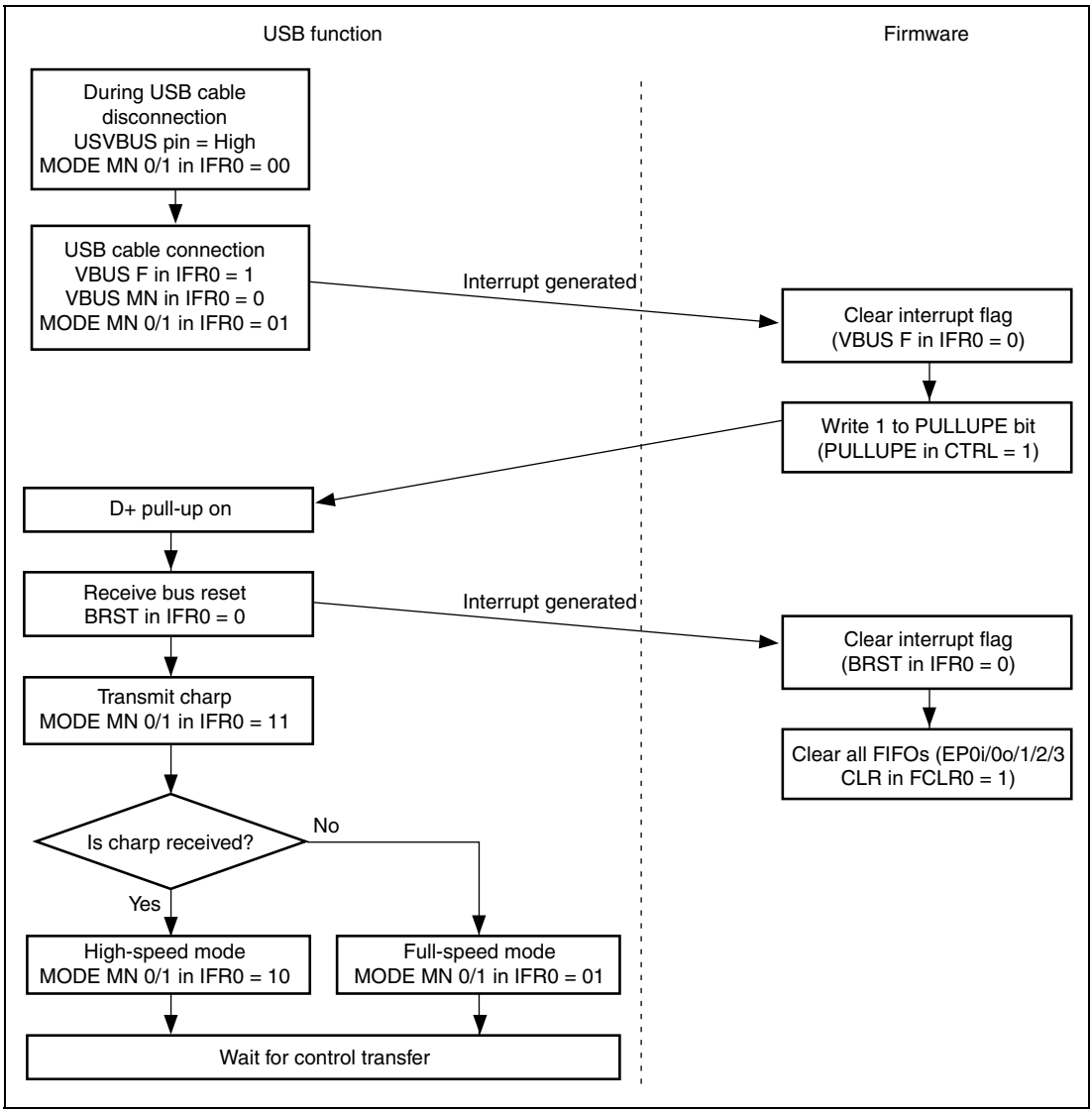


Figure 12.2 USB Cable Connection

As not shown in figure 12.2, when the MODE F bit in IER0 is set to 1 and the MODE MN0 and MODE MN1 bits in IFR0 are changed, an interrupt occurs. This interrupt can be used to manage the bulk maximum packet size and descriptor information.

12.5.2 USB Cable Disconnection

If the USB cable enters the disconnection state from the connection state, perform the operation as shown in figure 12.3.

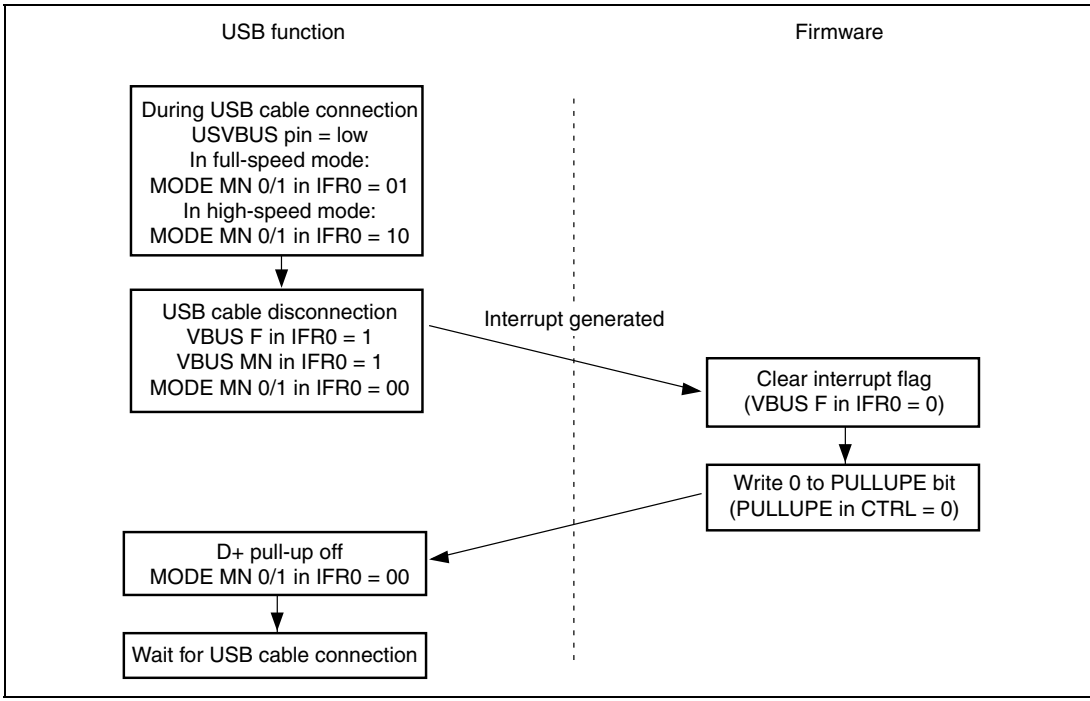


Figure 12.3 USB Cable Disconnection

As not shown in figure 12.3, when the MODE F bit in IER0 is set to 1 and the MODE MN0 and MODE MN1 bits in IFR0 are changed, an interrupt occurs.

12.5.3 Control Transfer

The control transfer consists of three stages; setup, data (sometimes omitted), and status, as shown in figure 12.4. The data stage consists of multiple bus transactions. Figures 12.5 to 12.9 show operation flows in each stage.

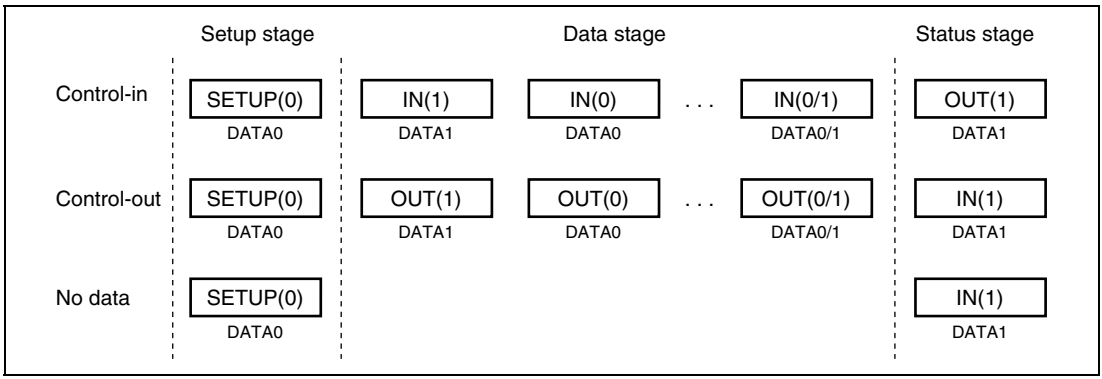
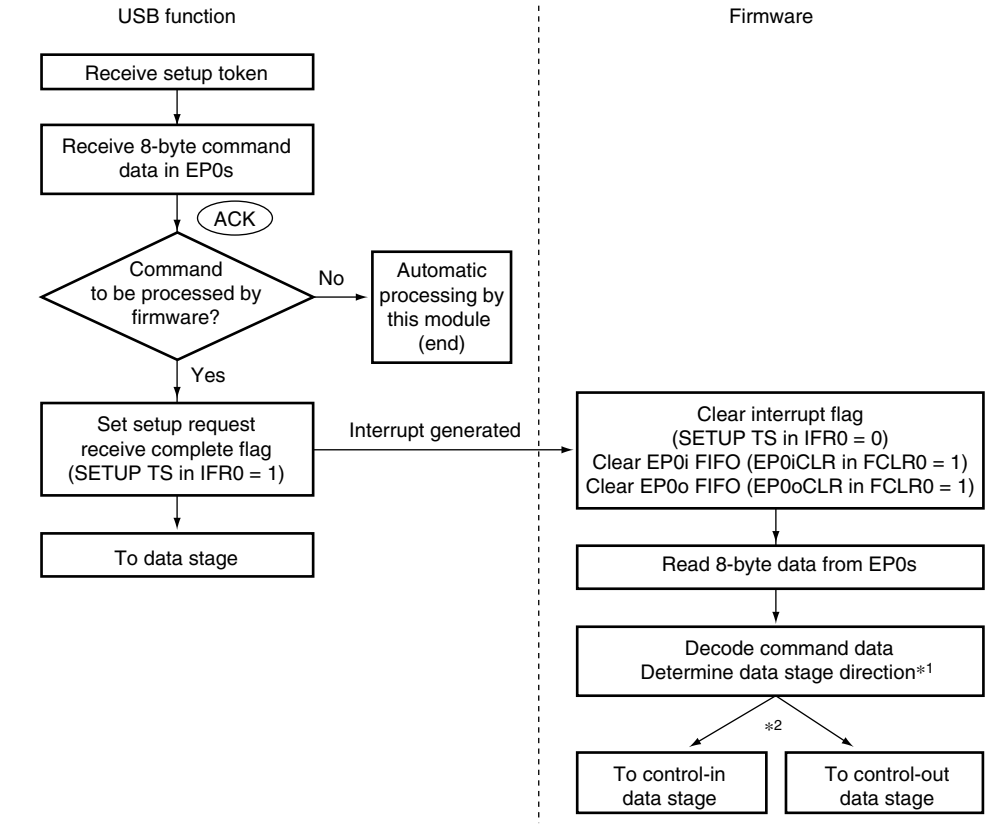


Figure 12.4 Control Transfer Stage Configuration

1. Setup Stage



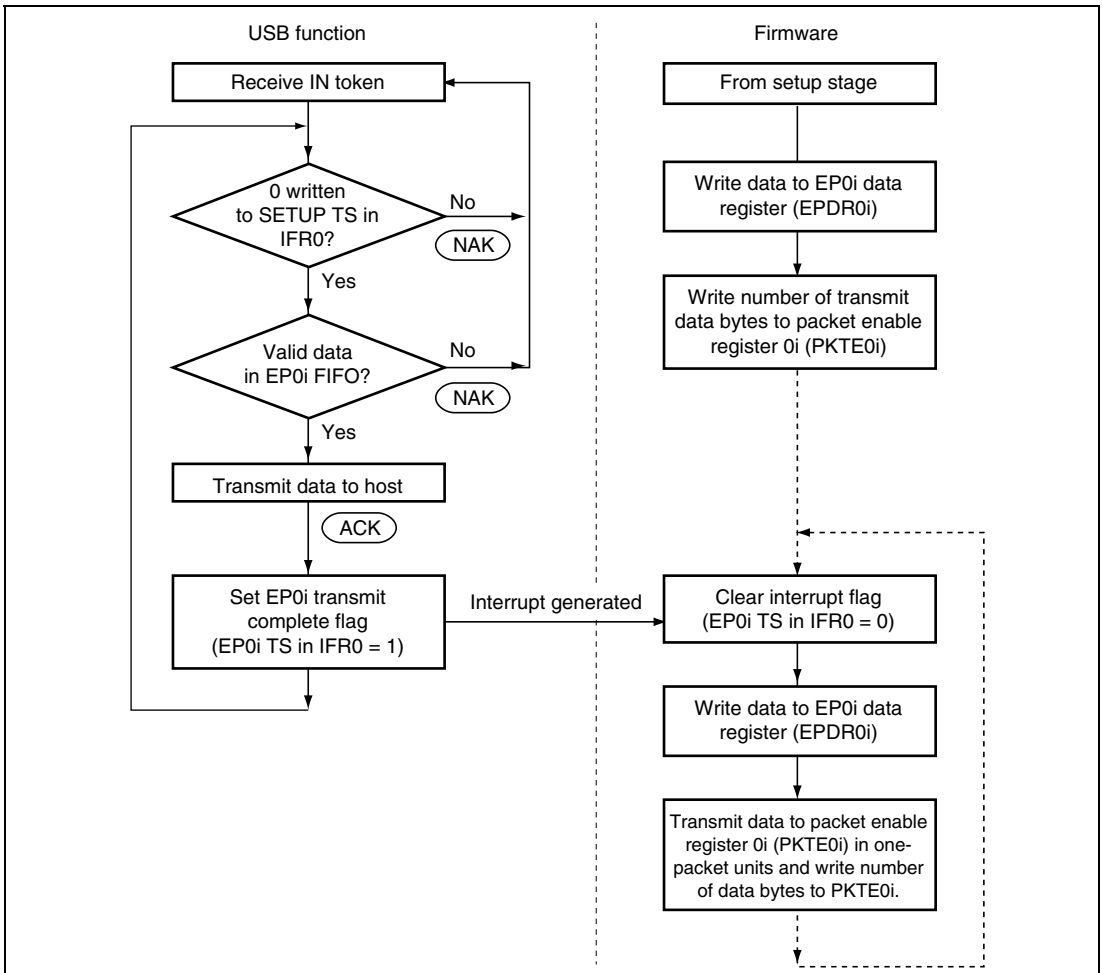
Notes: 1. In the setup stage, the firmware first analyzes the request data that is sent from the host required to be processed by the firmware, and determines subsequent processing. (For example, the data stage direction.)
2. When the transfer direction is control-out, the EP0i transfer request interrupt that is required in the status stage should be enabled. When the transfer direction is control-in, this interrupt is not required and must be disabled.

Figure 12.5 Setup Stage Operation

2. Data Stage (Control-In)

The firmware first analyzes the request data that is sent from the host in the setup stage, and determines the subsequent data stage direction. If the result of request data analysis is that the data stage is in-transfer, one packet of data to be sent to the host is written to the FIFO. If there is more data to be sent, this data is written to the FIFO after the data written first has been sent to the host (EP0i TS bit in IFR0 is set to 1).

The end of the data stage is identified when the host transmits an OUT token and the status stage is entered.



Note: If the size of the data transmitted by the function is smaller than the data size requested by the host, the function indicates the end of the data stage by returning to the host a packet shorter than the maximum packet size. If the size of the data transmitted by the function is an integral multiple of the maximum packet size, the function indicates the end of the data stage by transmitting a zero-length packet.

Figure 12.6 Data Stage Operation (Control-In)

3. Data Stage (Control-Out)

The firmware first analyzes the request data that is sent from the host in the setup stage, and determines the subsequent data stage direction. If the result of request data analysis is that the data stage is out-transfer, data from the host is waited for, and after data is received (EP0o TS bit in IFR0 is set to 1), data is read from the FIFO.

The end of the data stage is identified when the host transmits an IN token and the status stage is entered.

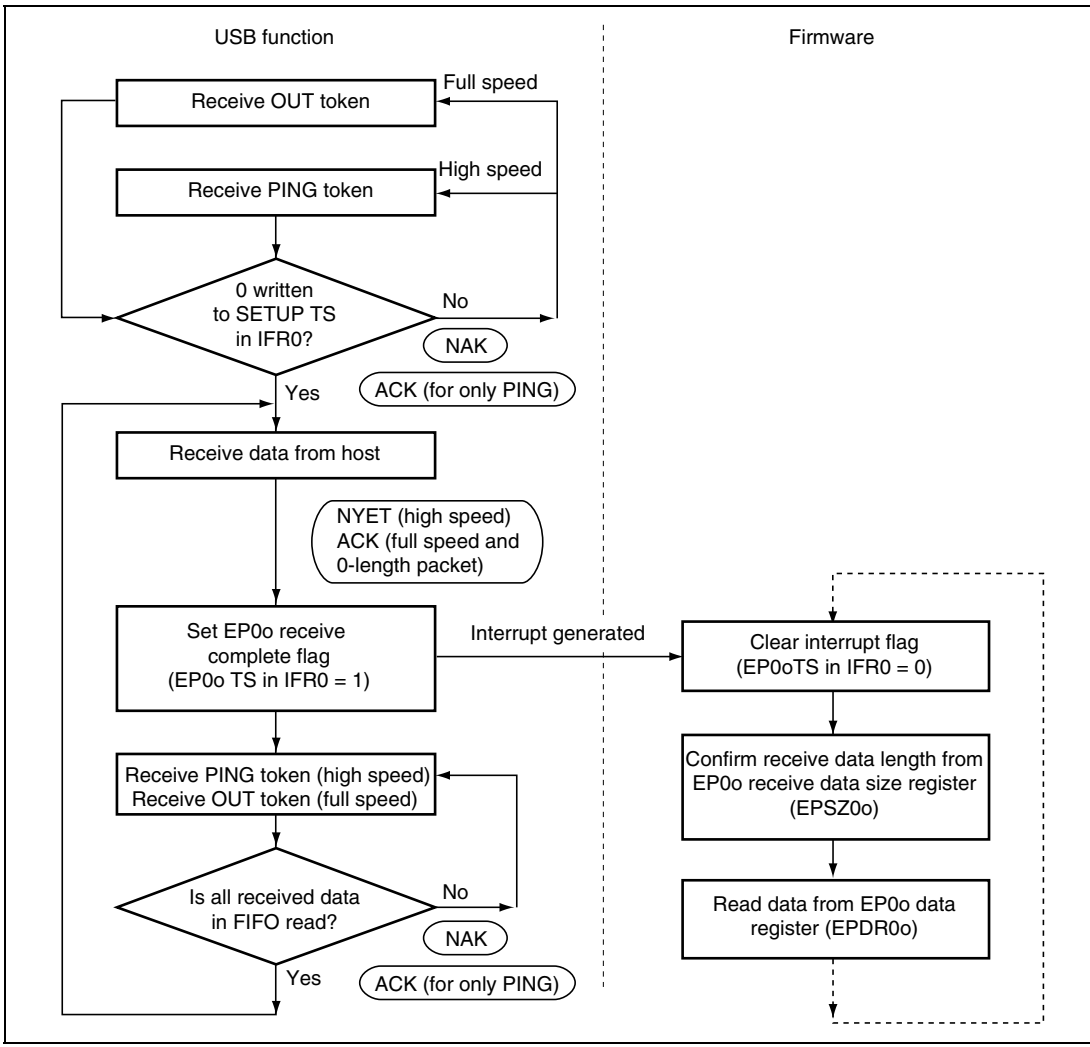


Figure 12.7 Data Stage Operation (Control-Out)

4. Status Stage (Control-In)

The control-in status stage starts with an OUT token from the host. The firmware receives 0-length packet from the host, and ends control transfer.

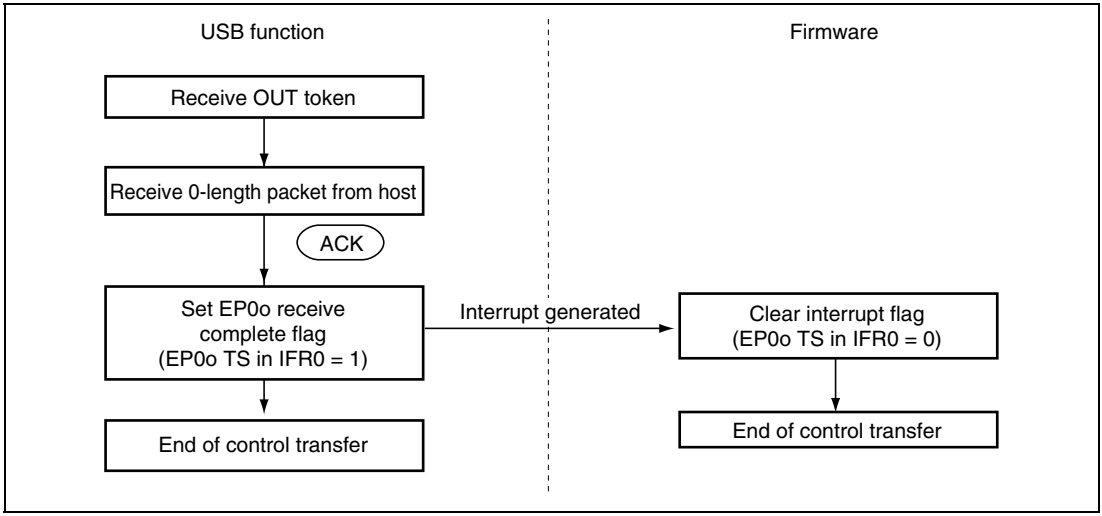


Figure 12.8 Status Stage Operation (Control-In)

5. Status Stage (Control-Out)

The control-out status stage starts with an IN token from the host. When an IN-token is received at the start of the status stage, there is not yet any data in the EP0i FIFO, and so an EP0i transfer request interrupt is generated. The firmware recognizes from this interrupt that the status stage has started. Next, in order to transmit 0-length packet to the host, 0 is written in the packet enable register 0i but no data is written to the EP0i FIFO. As a result, the next IN token causes 0-length packet to be transmitted to the host, and control transfer ends.

After the firmware has finished all processing relating to the data stage, 0 should be written in the packet enable register 0i.

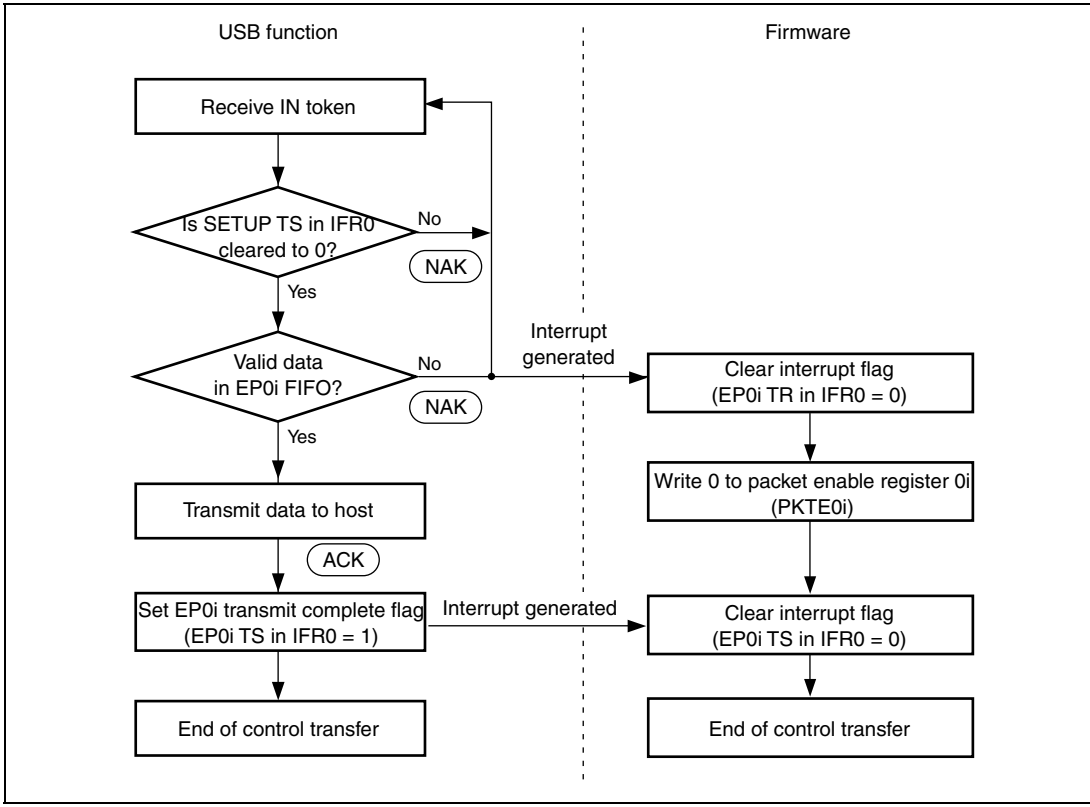


Figure 12.9 Status Stage Operation (Control-Out)

12.5.4 EP1 Bulk-Out Transfer (Dual FIFO)

EP1 has two 64-byte FIFOs in full-speed mode and two 512-byte FIFOs in high-speed mode, but the user can receive data and read receive data without being aware of this dual-FIFO configuration. Make sure to confirm that the EP1 FULL bit in IFR0 is set to 1 before reading the single FIFO.

When one FIFO is full after reception is completed, the EP1 FULL bit in IFR0 is set. After the first receive operation into one of the FIFOs when both FIFOs are empty, the other FIFO is empty, and so the next packet can be received immediately. When both FIFOs are full, NAK is returned to the host automatically. When reading of the receive data is completed following data reception, this operation empties the FIFO that has just been read, and makes it ready to receive the next packet. If 0-length packet is received from the host, the ACK handshake is returned to the host regardless of mode (full-speed or high-speed mode) and the EP1 FULL bit in IFR0 is not set.

Note: The dual-configured FIFOs are handled in packet units. Therefore, even if either of FIFOs is empty because of a short packet, when the packet is received successfully, both FIFOs become full.

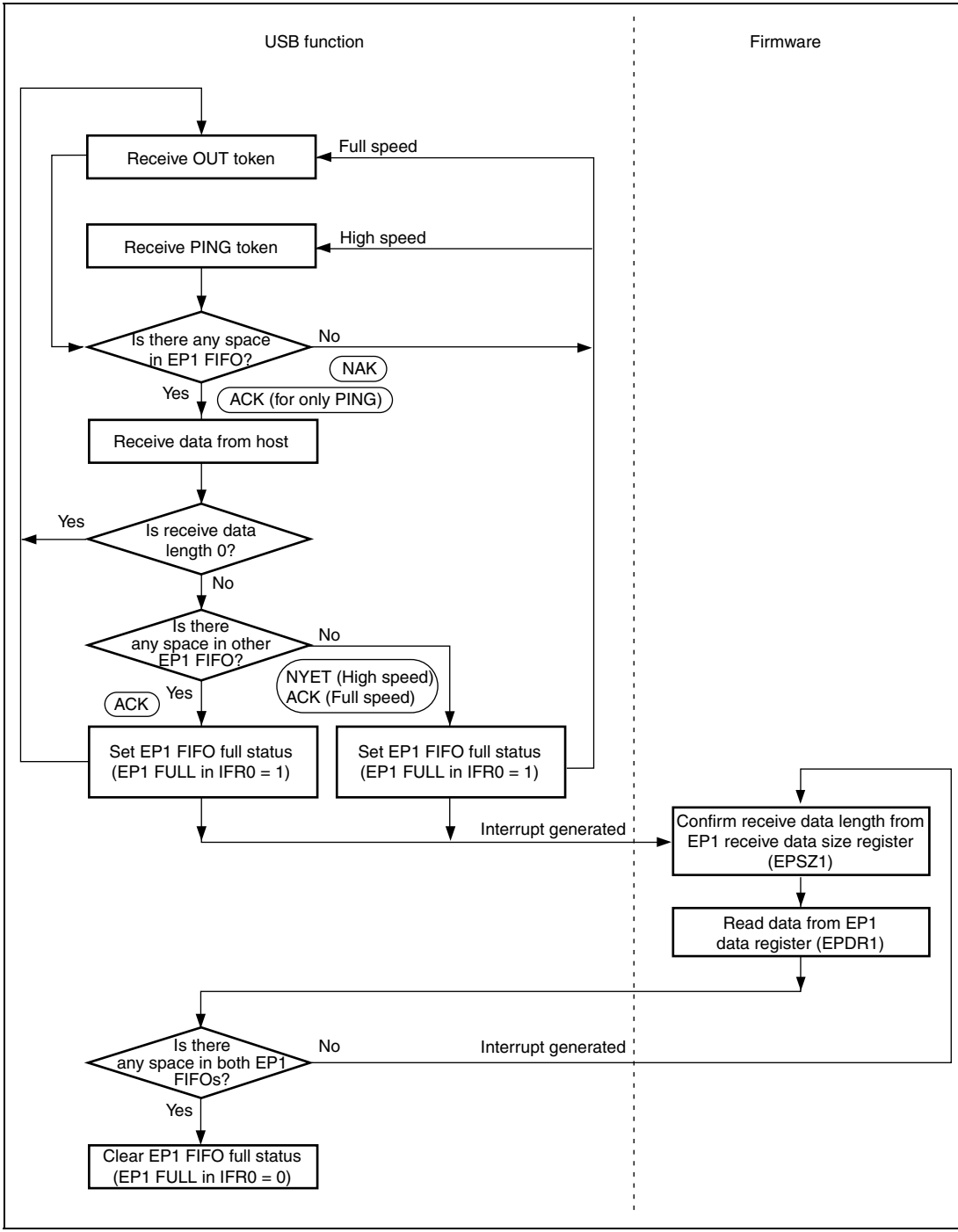


Figure 12.10 EP1 Bulk-Out Transfer Operation

12.5.5 EP2 Bulk-In Transfer (Dual FIFO)

EP2 has two 64-byte FIFOs in full-speed mode and two 512-byte FIFOs in high-speed mode, but the user can transmit data and write transmit data without being aware of this dual-FIFO configuration. However, one data write should be performed for one FIFO. For example, even if both FIFOs are empty, it is not possible to write the number of transmit data in PKTE2 at one time after consecutively writing 128 bytes of data in full-speed mode or 1024 bytes of data in high-speed mode. The number of transmit data must be written in PKTE2 for each 64-byte write in full-speed mode or 512-byte write in high-speed mode. Make sure to confirm that the EP2EMPTY bit in IFR0 is set to 1 before writing data.

When performing bulk-in transfer, as there is no valid data in the FIFOs on reception of the first IN token, the EP2 TR interrupt in IFR0 is requested. With this interrupt, 1 is written to the EP2 EMPTY bit in IER0, and the EP2 FIFO empty interrupt is enabled. At first, both EP2 FIFOs are empty, and so an EP2 FIFO empty interrupt is generated immediately.

The data to be transmitted is written to the data register using this interrupt. After the first transmit data write for one FIFO, the other FIFO is empty, and so the next transmit data can be written to the other FIFO immediately. When both FIFOs are full, the EP2 EMPTY bit is cleared to 0. If at least one FIFO is empty, the EP2 EMPTY bit in IFR0 is set to 1. When ACK is returned from the host after data transmission is completed, the FIFO used in the data transmission becomes empty. If the other FIFO contains valid transmit data at this time, transmission can be continued.

When transmission of all data has been completed, write 0 to the EP2 EMPTY bit in IER0 and disable interrupt requests.

Note: The dual-configured FIFOs are handled in packet units. Therefore, even if either of FIFOs is empty because of a short packet, when the number of transmit data is written in the packet enable register 2 (PKTE2), both FIFOs become full.

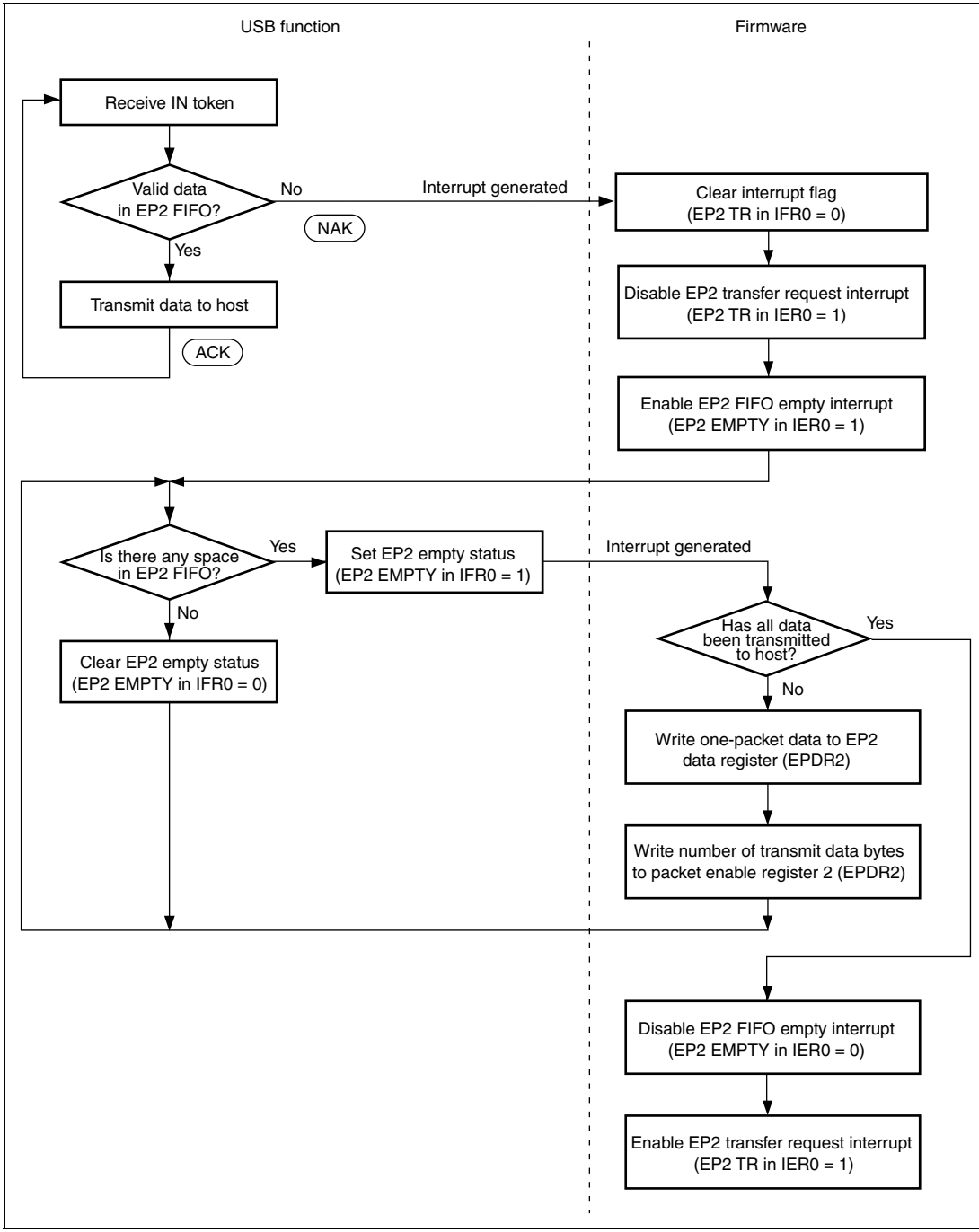
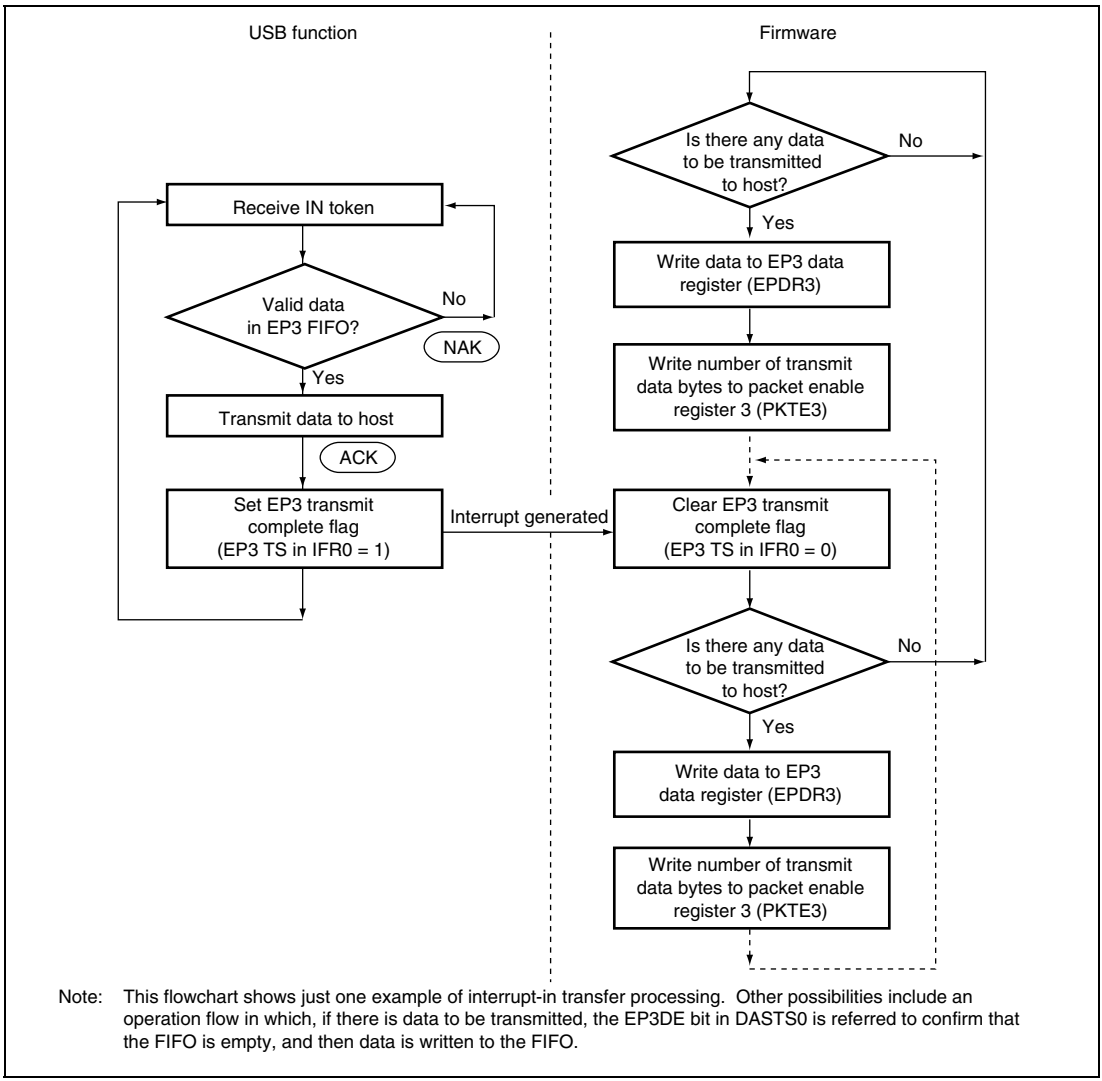


Figure 12.11 EP2 Bulk-In Transfer Operation

12.5.6 EP3 Interrupt-In Transfer



Note: This flowchart shows just one example of interrupt-in transfer processing. Other possibilities include an operation flow in which, if there is data to be transmitted, the EP3DE bit in DASTS0 is referred to confirm that the FIFO is empty, and then data is written to the FIFO.

Figure 12.12 EP3 Interrupt-In Transfer Operation

12.5.7 Processing of USB Standard Requests and Class/Vendor Requests

1. Processing of Requests Transmitted by Control Transfer

A request transmitted from the host by control transfer may require decoding and execution of request processing by the firmware. Whether or not request decoding is required by the firmware is indicated in table 12.2 below.

Table 12.2 Request Decoding by Firmware

Decoding not Necessary by Firmware	Decoding Necessary by Firmware
Clear Feature	Get Descriptor
Get Configuration	Synch Frame
Get Interface	Set Descriptor
Get Status	Class/Vendor request
Set Address	
Set Configuration	
Set Feature	
Set Interface	

If decoding is not necessary by the firmware, request decoding and data stage and status stage processing are performed automatically. No processing is necessary by the user. An interrupt is not generated in this case.

If decoding is necessary by the firmware, this module stores the request in the EP0s FIFO. After normal reception is completed, the SETUPTS flag in IER0 is set and an interrupt request is generated. In the interrupt routine, eight bytes of data must be read from the EP0s data register (EPDR0s) and decoded by the firmware. The necessary data stage and status stage processing should then be carried out according to the result of the decoding operation.

12.5.8 Stall Operations

This section describes stall operations in the USB module. There are two cases in which the USB module stall function is used:

- When the firmware forcibly stalls an endpoint for some reason
- When a stall is performed automatically within the USB module due to a USB specification violation

The USB module has internal status bits that hold the status (stall or non-stall) of each endpoint. When a transaction is sent from the host, the module refers these internal status bits and determines whether to return a stall to the host. These bits cannot be cleared by the firmware; they must be cleared with a Clear Feature request from the host. However, the internal status bit for EP0 is cleared automatically at the reception of the setup request.

1. Forcible Stall by Firmware

The firmware uses EPSTL0 to issue a stall request for the USB module. When the firmware wishes to stall a specific endpoint, it sets the corresponding bit in EPSTL0 (1-1 in figure 12.13). The internal status bits are not changed at this time.

When a transaction is sent from the host for the endpoint for which the corresponding bit in EPSTL0 was set, the USB module refers the internal status bit, and if this is not set, refers the corresponding bit in EPSTL0 (1-2 in figure 12.13). If the corresponding bit in EPSTL0 is set, the USB module sets the internal status bit and returns a stall handshake to the host (1-3 in figure 12.13). If the corresponding bit in EPSTL0 is not set, the internal status bit is not changed and the transaction is accepted.

Once an internal status bit is set, it remains set until cleared by a Clear Feature request from the host, without regarding to EPSTL0. Even after a corresponding bit is cleared by the Clear Feature request (3-1 in figure 12.13), the USB module continues to return a stall handshake while the bit in EPSTL0 is set, since the internal status bit is set each time a transaction is executed for the corresponding endpoint (1-2 in figure 12.13). To clear a stall, therefore, it is necessary for the corresponding bit in EPSTL0 to be cleared by the firmware and also for the internal status bit to be cleared with a Clear Feature request (2-1, 2-2, and 2-3 in figure 12.13).

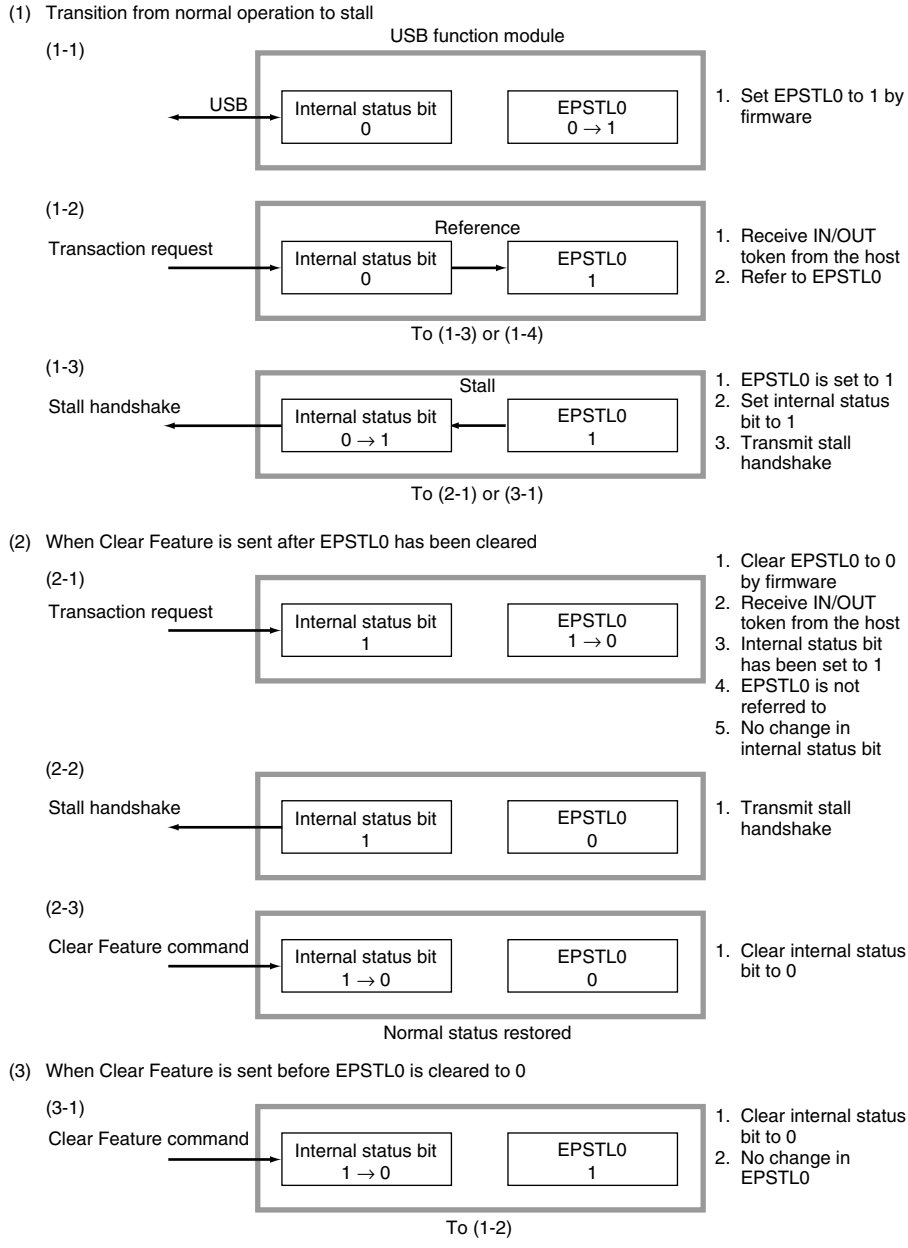


Figure 12.13 Forcible Stall by Firmware

2. Automatic Stall by USB Function Module

When a stall setting is made with the Set Feature request or in the event of a USB specification violation, the USB module automatically sets the internal status bit for the corresponding endpoint without regarding to EPSTL0, and returns a stall handshake (1-1 in figure 12.14).

Once an internal status bit is set, it remains set until cleared by a Clear Feature request from the host, without regarding to EPSTL0. After a corresponding bit is cleared by the Clear Feature request, EPSTL0 is referred (3-1 in figure 12.14). The USB module continues to return a stall handshake while the internal status bit is set, since the internal status bit is set even if a transaction is executed for the corresponding endpoint (2-1 and 2-2 in figure 12.14). To clear a stall, therefore, the internal status bit must be cleared with a Clear Feature request (3-1 in figure 12.14). If set by the firmware, EPSTL0 should also be cleared (2-1 in figure 12.14).

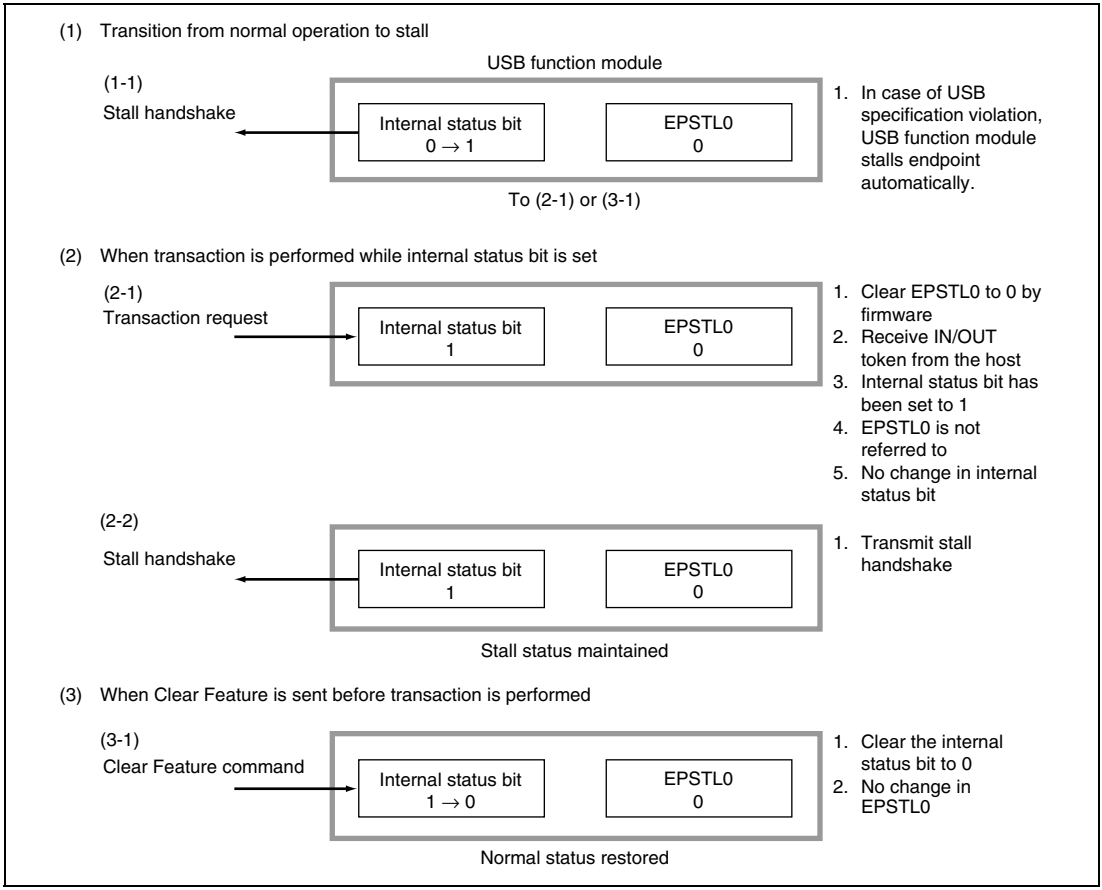


Figure 12.14 Automatic Stall by USB Function Module

12.5.9 Tree Configuration

This section describes the tree configuration of this module. The USB determines the tree configuration in the function using three parameters such as Configuration, Interface, and Alternate. The tree configuration of this module is shown in table 12.3.

Table 12.3 Tree Configuration

EP Number	Conf.	Int.	Alt.	Transfer Method	Transfer Direction
0	—	—	—	Control	In/Out
1	1	0	0	Bulk	Out
2	1	0	0	Bulk	In
3	1	0	0	Interrupt	In

12.5.10 Power Supply Specification

This module functions by self power supply. The bus power supply supplied from the USB cable is not available.

12.6 Notes on Using DMA

- PKTE2 Operation (EP2)

  When DMA transfer is performed on EP2 transmit data, the USB module automatically performs the same processing as writing the number of transmit data to PKTE2 if one data FIFO (maximum packet size) becomes full. Accordingly, to transfer data of integral multiples of the maximum packet size, the user needs not to write the number of transmit data in PKTE2. To transfer data of less than the maximum packet size, the user must write the number of transmit data in PKTE2 using the DMA transfer end interrupt of the DMAC. If the user writes the number of transmit data in PKTE2 in cases other than the case when data of less than the maximum packet size is transferred, excess transfer occurs and correct operation cannot be guaranteed.

  Figure 12.15 shows an example for transmitting 1050 bytes of data from EP1 to the host in high-speed mode. In this case, internal processing as the same as writing the number of transmit data to PKTE2 is automatically performed twice. This kind of internal processing is performed when the currently selected data FIFO becomes full. Accordingly, this processing is automatically performed only when the maximum packet size of data is transmitted. This processing is not performed automatically when data less than the maximum packet size is transmitted.

In this example, when the first 512 bytes of data has not been transmitted to the host at the end of DMA transfer on the first 512 bytes of data and second 512 bytes of data, the DMA request does not occur. If the transmission has been completed, the DMA request occurrs because there is space in the one FIFO.

When the last 26 bytes of data has been transferred, there is no data to be transferred in the firmware. However, the DMA request ocurrs if there is space in the FIFO. Therefore, when DMA transfer is completed on all data, the DMA enable state should be cleared by writing 0 to the EP2DMAE bit in DMA0. Note that over-sampling should not be performed by the DMAC. Generally the number of transmit data is set as the number of DMAC transfers and the number of data less than the maximum packet size is written in PKTE2 using the DMA transfer end interrupt. If the number of transmit data is an integral multiple of the maximum packet size (for example, 1024 bytes or 2048 bytes), the number of transmit data is automatically written in PKTE2. In this case, the user must not write the number of transmit data in PKTE2 using the DMA transfer end interrupt. If the writing is performed, correct operation cannot be guaranteed.

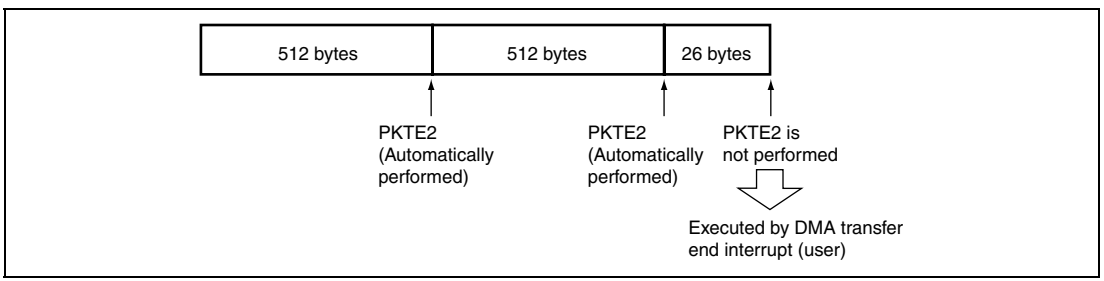


Figure 12.15 PKTE2 Operation for EP2

12.7 Transition to USB Suspend Mode

12.7.1 Suspend Signal Output

If the USB continues to be idle for the specified time, the USB enters the suspend state. When the USB enters the suspend state, the $\overline{\text{USSUSP}}$ signal is asserted to low and the suspend state can be notified to the external transceiver.

When the external transceiver receives the $\overline{\text{USSUSP}}$ signal, the clock oscillation is stopped and the power-down state is entered. Therefore the USCLK supplied to this LSI is stopped. To operate the USB module, the USCLK supplied by the external transceiver and system clock (ϕ) should be supplied. While the USCLK is stopped, programming is required so that access to the USB module does not occur.

To control this, the USB suspend interrupt (USBI2) which notifies that the USB module enters or recovers the suspend state to the CPU and a register bit (USUSOUT bit in USBSUSP) which controls the external $\overline{\text{USSUSP}}$ output are provided.

Figure 12.16 shows the enter/recover sequence of suspend mode and operation in each point is explained in the following 1 to 6 descriptions.

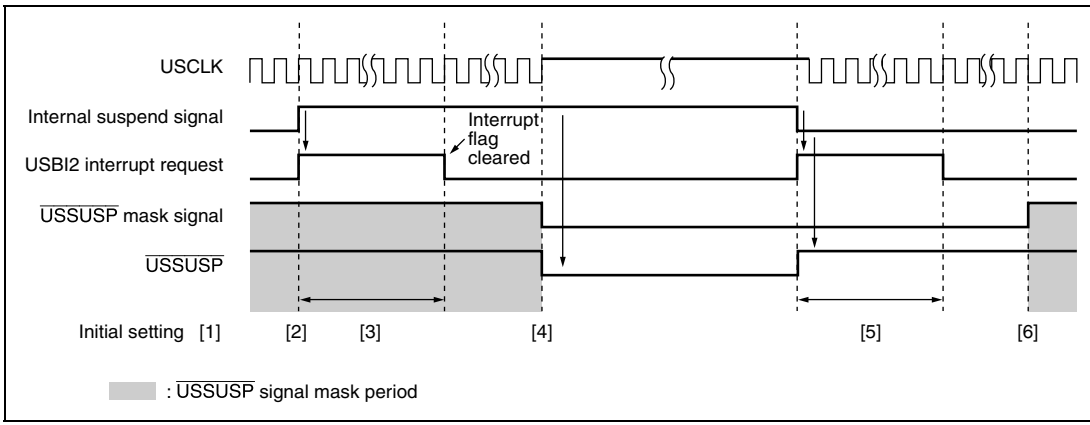


Figure 12.16 Enter/Recover Sequence of USB Suspend State

1. $\overline{\text{USBSUSP}}$ Output Mask

When the USUSOUT bit in USBSUSP is cleared to 0, output of the $\overline{\text{USSUSP}}$ pin is masked (fixed to 1) and notification of the suspend state to the external transceiver is disabled.

2. Entering Suspend State by USB Module

If the USB continues to be idle for the specified time, the USB module asserts the internal suspend signal and enters the suspend state. Since the $\overline{\text{USSUSP}}$ output is masked by means of the USUSOUT bit in USBSUSP, the external pin, $\overline{\text{USSUSP}}$, is not changed.

3. USB Suspend Interrupt (USBI2)

When the USUSFGE bit in USBSUSP is set to 1 and the internal suspend signal of the USB module is changed, an interrupt (USBI2) can be requested to the CPU.

As the initial setting, the USUSFG bit (interrupt flag) in USBSUSP should be cleared to 0 and then the USUSFGE bit in USBSUSP should be set to 1.

The USBI2 interrupt request occurs when the internal suspend signal of the USB module is changed. After the interrupt flag is cleared in the interrupt routine, monitor the USUSMONI bit in USBSUSP and confirm that the interrupt occurs because of transition to the suspend state. Note that an interrupt occurs when transition is made in the following order: normal mode, suspend mode, and normal mode.

4. Clearing Suspend Signal Output Mask

To prevent access to the USB module by the CPU or DMAC from occurring in the interrupt processing routine, the USUSOUT bit in USBSUSP is set to 1 after the setting such as to stop the DMAC is made. Then output of the $\overline{\text{USSUSP}}$ pin goes low and the USCLK input from the external transceiver is stopped.

After the USCLK input is stopped, do not access the USB module registers by the CPU or DMAC.

5. USB Suspend Interrupt (USBI2)

Even if the USCLK is stopped, the internal suspend signal is negated after the USB bus state is recovered from the suspend state.

Since the USBI2 interrupt is requested when the internal suspend signal is changed, if interrupts are enabled, an interrupt is requested to the CPU even when the internal suspend signal is negated.

6. Suspend Signal Output Mask

In the interrupt processing routine, output of the $\overline{\text{USSUSP}}$ pin should be masked (fixed to 1) by clearing the USUSOUT bit in USBSUSP to 0.

Then access to the USB module registers can be performed by the CPU or DMAC.

12.7.2 Software Standby in Suspend Mode

In the firmware mainly for the USB, if the USB enters suspend mode, this LSI enters software standby mode which is the power-down state because power consumption of the system can be reduced. Since the system clock (ϕ) is stopped in software standby mode, the suspend recover interrupt (SUSRI) is used to recover from software standby mode.

Figure 12.17 shows the enter/recover sequence of suspend mode and software standby mode and operation in each point is explained in the following 1 to 8 descriptions.

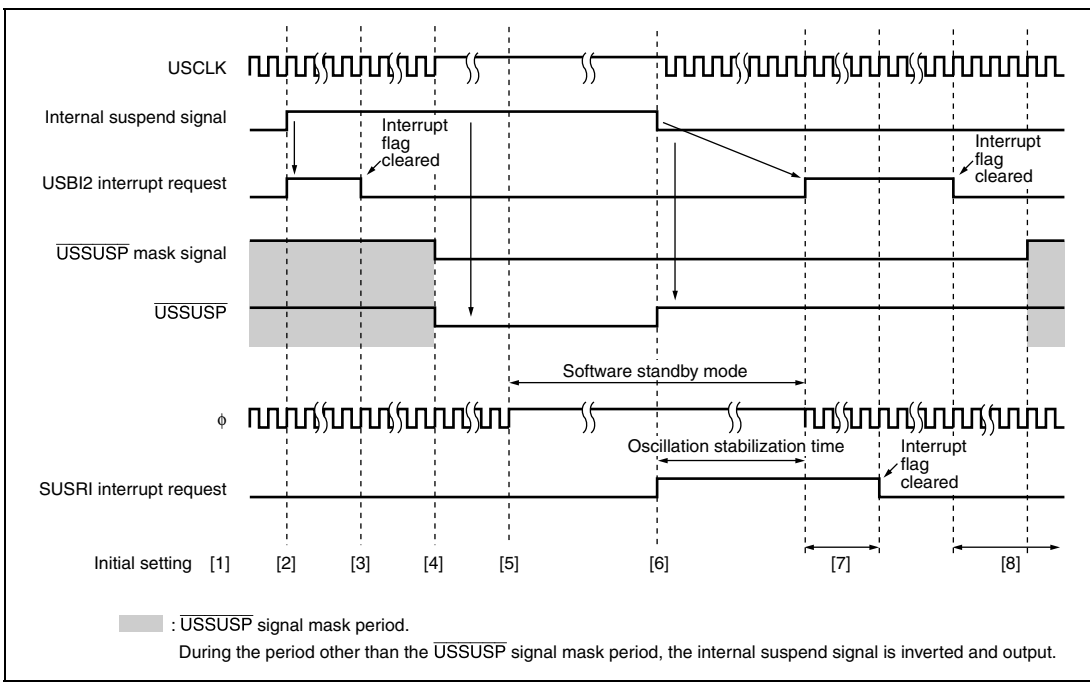


Figure 12.17 Enter/Recover Sequence of USB Suspend State and Software Standby Mode

1. $\overline{\text{USSUSP}}$ Output Mask

When the USUSOUT bit in USBSUSP is cleared to 0, output of the $\overline{\text{USSUSP}}$ pin is masked (fixed to 1) and notification of the suspend state to the external transceiver is disabled.

2. Entering Suspend State by USB Module

If the USB continues to be idle for the specified time, the USB module asserts the internal suspend signal and enters the suspend state. Since the $\overline{\text{USSUSP}}$ output is masked by means of the USUSOUT bit in USBSUSP, the external pin, $\overline{\text{USSUSP}}$, is not changed.

3. USB Suspend Interrupt (USBI2)

When the USUSFGE bit in USBSUSP is set to 1 and the internal suspend signal of the USB module is changed, an interrupt (USBI2) can be requested to the CPU.

As the initial setting, the USUSFG bit (interrupt flag) in USBSUSP should be cleared to 0 and then the USUSFGE bit in USBSUSP should be set to 1.

The USBI2 interrupt request occurs when the internal suspend signal of the USB module is changed. After the interrupt flag is cleared in the interrupt routine, monitor the USUSMONI bit in USBSUSP and confirm that the interrupt occurs because of transition to the suspend state. Note that an interrupt occurs when transition is made in the following order: normal mode, suspend mode, and normal mode.

4. Clearing Suspend Signal Output Mask

To prevent access to the USB module by the CPU or DMAC from occurring in the interrupt processing routine, the USUSOUT bit in USBSUSP is set to 1 after the setting such as to stop the DMAC is made. Then output of the $\overline{\text{USSUSP}}$ pin goes low and the USCLK input from the external transceiver is stopped.

After the USCLK input is stopped, do not access the USB module registers by the CPU or DMAC.

5. Entering Software Standby Mode

When the USB bus state enters the state other than the USB idle state before software standby mode is entered, the suspend recover interrupt (SUSRI) is set in order to clear software standby mode.

To enable the suspend recover interrupt (SUSRI), clear the SUSRIF bit in PFCR3 to 0 and then set the SUSRIE bit in PFCR3 to 1.

After that, the SLEEP instruction is executed while the SSBY bit in SBYCR is set to 1 in order to enter software standby mode.

The system clock (ϕ) is stopped and this LSI enters the power-down state.

6. **Recover Event from Suspend**

   If the USB bus state recovers from the suspend state, the suspend recover interrupt (SUSRI) is requested. By this interrupt request, software standby mode can be cancelled.

   If the USB bus state recovers from the suspend state, the $\overline{\text{USSUSP}}$ pin is negated and this LSI starts oscillation. After the specified oscillation stabilization time is elapsed, the internal system clock is supplied and the CPU executes the suspend recover interrupt (SUSRI) processing.

7. **Suspend Recover Interrupt (SUSRI)**

   In the interrupt processing routine, clear the SUSRIF bit in PFCR3 to 0 and then clear the SUSRIE bit in PFCR3 to 0 in order to disable the suspend recover interrupt (SUSRI).

   At this time, the USBI2 interrupt is requested. The priority of the SUSRI interrupt must be higher than that of the USBI2 interrupt.

8. **USB Suspend Interrupt (USBI2)**

   After recovering from the SUSRI interrupt processing, the pending USBI2 interrupt processing is executed.

   To mask output of the $\overline{\text{USSUSP}}$ pin (fix to 1), clear the USUSOUT bit in USBSUSP to 0 in the interrupt processing routine.

   Then access to the USB registers can be performed by the CPU or DMAC.

12.8 Usage Notes

12.8.1 Setup Data Reception

The following must be noted for the EP0s FIFO used to receive 8-byte setup data.

The USB is designed to always receive setup requests. Accordingly, write from the USB bus has higher priority than read from the LSI. If the reception of the next setup request starts while the LSI is reading data after completing reception, this data read from the LSI is forcibly cancelled and the next setup request write starts. After the next setup request write, data read from the LSI is thus undefined.

12.8.2 FIFO Clear

If the USB cable is disconnected during communication, old data may be contained in the FIFO. Accordingly, FIFOs must be cleared immediately after USB cable connection.

Note, however, that FIFOs that are currently used for data transfer to or from the host must not be cleared.

12.8.3 Operating Frequency

The system clock (ϕ) must range from 31 MHz to 33 MHz.

12.8.4 Interrupts

This module uses signals in which each bit in IFR0 is logical-ORed as interrupt requests. Thus, even if a flag is cleared, the request is not negated while at least one of other flags is set.

12.8.5 Register Access Size

In this module, all registers should be accessed in 32-bit units using the MOV.L instruction. They cannot be accessed in 8- or 16-bit units using instructions such as MOV.W, MOV.B, or bit manipulation instruction.

12.8.6 Data Register Overread or Overwrite

When data registers in this module are read from or written to, the following must be noted:

1. **Receive Data Register**

   Receive data registers must not read a data size that is greater than the effective size of the read data item. In other words, receive data registers must not read data with data size larger than that specified by the receive data size register. For the receive data register of EP1 having a dual-FIFO configuration, data to be read at any time must be within the maximum packet size. Make sure to confirm that the EP1 FULL bit in IFR0 is set to 1 before reading from data from a single FIFO, because data registers cannot be accessed while FIFOs are switched.

2. **Transmit Data Register**

   Data to be written to the transmit data registers must be within the maximum packet size. For the transmit data register of EP2 having a dual-FIFO configuration, data to be written at any time must be within the maximum packet size. In this case, after a data write, the FIFO is switched to the other FIFO, enabling an further data write, when the number of transmit data is written in PKTE2. Accordingly, data of size corresponding to two FIFOs must not be written to the transmit data registers at a time. Make sure to confirm that the EP2EMPTY bit in IFR0 is set to 1 before writing data to a single FIFO, because data registers cannot be accessed while FIFOs are switched.

12.8.7 EP0 Interrupt Sources Assignment

EP0 interrupt sources assigned to bits 3 to 0 in IFR0 must be assigned to the same interrupt signal by setting ISR0. There are no other restrictions on interrupt sources.

12.8.8 FIFO Size at Full Speed

This module operates in high-speed or full-speed mode. The FIFO size to be used in each mode is shown below. Therefore, data more than 64 bytes cannot be read from or written to the one FIFO for EP1 or EP2 in full-speed mode. If the read or write is performed, correct transfer is impossible. Data less than 64 bytes should be read from or written to.

Table 12.4 FIFO Size in Each Transfer Mode

Endpoint	FIFO Size		FIFO Configuration	Transfer Method	Transfer Direction
	Full Speed	High Speed			
EP0s	8 bytes	8 bytes	Single	Setup	Out
EP0i	64 bytes	64 bytes	Single	Control	In
EP0o	64 bytes	64 bytes	Single	Control	Out
EP1	64 bytes	512 bytes	Dual	Bulk	Out
EP2	64 bytes	512 bytes	Dual	Bulk	In
EP3	64 bytes	64 bytes	Single	Interrupt	In

12.8.9 Level Shifter for VBUS Pin

The USVBUS pin of this LSI must be connected to the USB connector's VBUS pin via a level shifter. (Make sure the polarity is correct.) This is because the USB module has a circuit that operates by detecting USB cable connection or disconnection.

Even if the power of the device incorporating this module is turned off, 5-V power is applied to the USB connector's VBUS pin while the USB cable is connected to the device set. To protect the LSI from destruction, use an external level shifter which allows voltage application to the pin even when the power is off.

12.8.10 USB 2.0 Transceiver (Physical Layer)

This module does not include the USB 2.0 transceiver. Therefore an external transceiver should be used.

12.8.11 EPDR0s Read

EPDR0s must be read in 8-byte units. If read is suspended, data received in the next setup cannot be read normally.

12.8.12 USB Bus Idle in High-Speed Mode

When the USB bus enters the idle state in high-speed mode, this module enters full-speed mode. However, the MODE MN1 and MODE MN0 bits in IFR0 are not changed. Therefore, the MODEF bit in IFR0 is not set.

12.8.13 Note on USB Bus Disconnection

If the connector is disconnected by the user regardless of whether communication is in progress with the host or the idle state is entered, the USB cable may be disconnected. When the USB cable is disconnected and data transfer is performed between the USB module and the host, the correct transfer cannot be performed after the cable is connected again. Therefore, when the cable disconnection is detected, the USB module must be reset. To reset this, set the USBSWRST bit in PFCR3 to 1. The CPU must access PFCR3. After the USBSWRST bit is set to 1 and one clock is elapsed, clear the USBSWRST bit to 0. Then, the USB module operates normally. After a reset, values of all registers in the USB module are returned to the initial values so they should be set again.

12.8.14 Example of External Circuit

Since this LSI does not contain the on-chip USB physical layer, the physical layer LSI in the UTMI specification should be connected externally. Figure 12.19 shows a connection example of the external circuit. Figure 12.19 is only an example. The actual connecting differs according to the specifications of the external physical layer LSI in the UTMI specification.

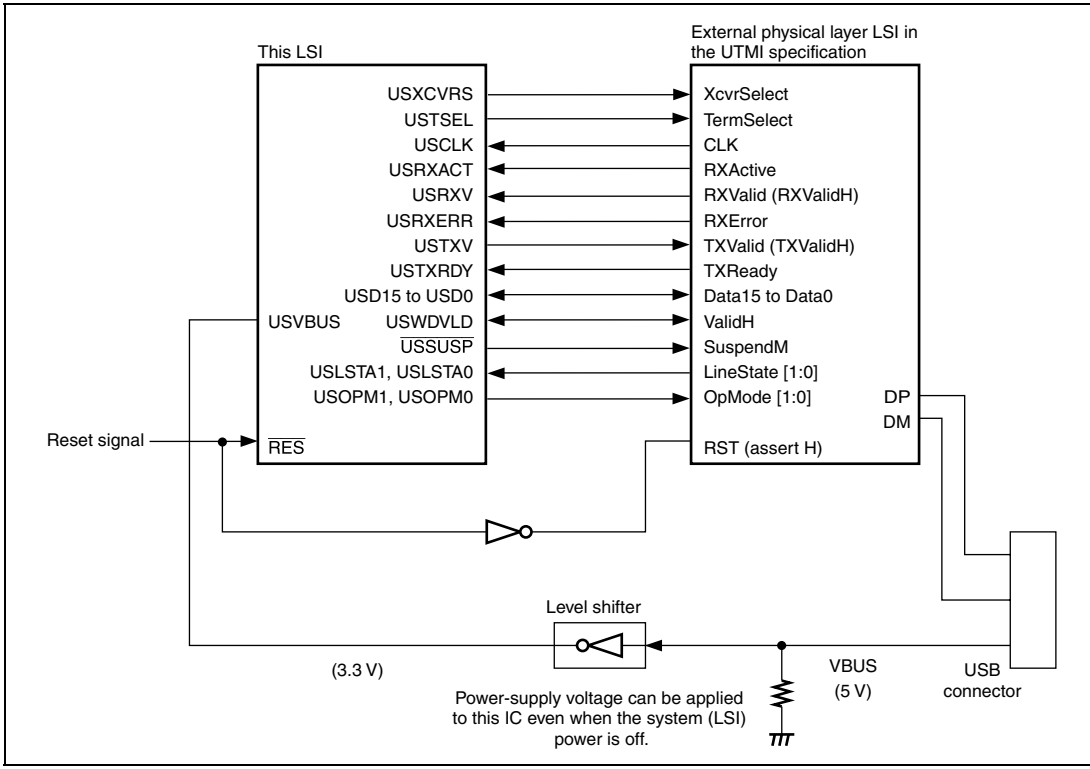


Figure 12.18 Connection Example of External Circuit

12.8.15 External Physical Layer LSI

The pin name, usage, and type may differ according to each of external physical layer LSI. Check and confirm the specifications of it before connecting to this LSI.

12.8.16 Operation at the Bus Reset Reception

When a bus reset is received from the host, there will be a defect in the operation described below. When the first bus reset from the host completes, and then the second bus reset is received without following bus accesses of SOF or data transfer, the second bus reset will not be received successfully (shown in case 2 in figure 12.19).

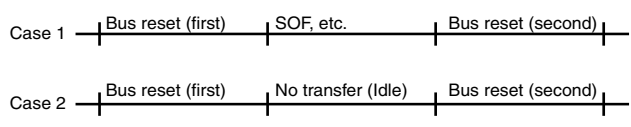


Figure 12.19 Bus Reset Following Completion of First Bus Reset

Note: Since the successive bus resets are not normally required, the bus reset does not occur successively with following no data transfer.

12.8.17 Usage Notes in Control IN Transfer

In the status stage of the control IN transfer, 0-length packet is only received.

Section 13 RAM

This LSI has an on-chip high-speed static RAM. The RAM is connected to the CPU by a 16-bit data bus and is connected to the DMAC by a 32-bit data bus, enabling one-state access to byte data, word data, and longword data (note that two-state access is used when the RAM is accessed by using the longword access instruction such as MOV.L by the CPU).

The on-chip RAM can be enabled or disabled by means of the RAME bit in the system control register (SYSCR). For details on the system control register (SYSCR), refer to section 3.2.2, System Control Register (SYSCR).

Product Type		ROM Type	RAM Capacity	RAM Address
H8S/2170	HD64F2170	Flash memory version	32 kbytes	H′FF7000 to H′FFEFFF

Section 14 Flash Memory (0.18-µm F-ZTAT Version)

The flash memory has the following features. Figure 14.1 shows a block diagram of the flash memory.

14.1 Features

- Size

Product Classification		ROM Size	ROM Address
H8S/2170	HD64F2170	256 kbytes	H′000000 to H′03FFFF

- Two flash-memory MATs according to LSI initiation mode
  The on-chip flash memory has two memory spaces in the same address space (hereafter referred to as memory MATs). The mode setting in the initiation determines which memory MAT is initiated first. The MAT can be switched by using the bank-switching method after initiation.
  - The user memory MAT is initiated at a power-on reset in user mode: 256 kbytes
  - The user boot memory MAT is initiated at a power-on reset in user boot mode: 8 kbytes
- Programming/erasing interface by the download of on-chip program
  This LSI has a dedicated programming/erasing program. After downloading this program to the on-chip RAM, programming/erasing can be performed by setting the argument parameter.
- Programming/erasing time
  The flash memory programming time is 3 ms (typ) in 128-byte simultaneous programming and approximately 25 µs per byte. The erasing time is 1000 ms (typ) per 64-kbyte block.
- Number of programming
  The number of flash memory programming can be up to 100 times at the minimum. (The value ranged from 1 to 100 is guaranteed.)
- Three on-board programming modes
  - Boot mode
    This mode is a program mode that uses an on-chip SCI interface. The user MAT and user boot MAT can be programmed. This mode can automatically adjust the bit rate between host and this LSI.
  - User program mode
    The user MAT can be programmed by using the optional interface.
  - User boot mode
    The user boot program of the optional interface can be made and the user MAT can be programmed.

- Programming/erasing protection

  Sets protection against flash memory programming/erasing via hardware, software, or error protection.
- Programmer mode

  This mode uses the PROM programmer. The user MAT and user boot MAT can be programmed.

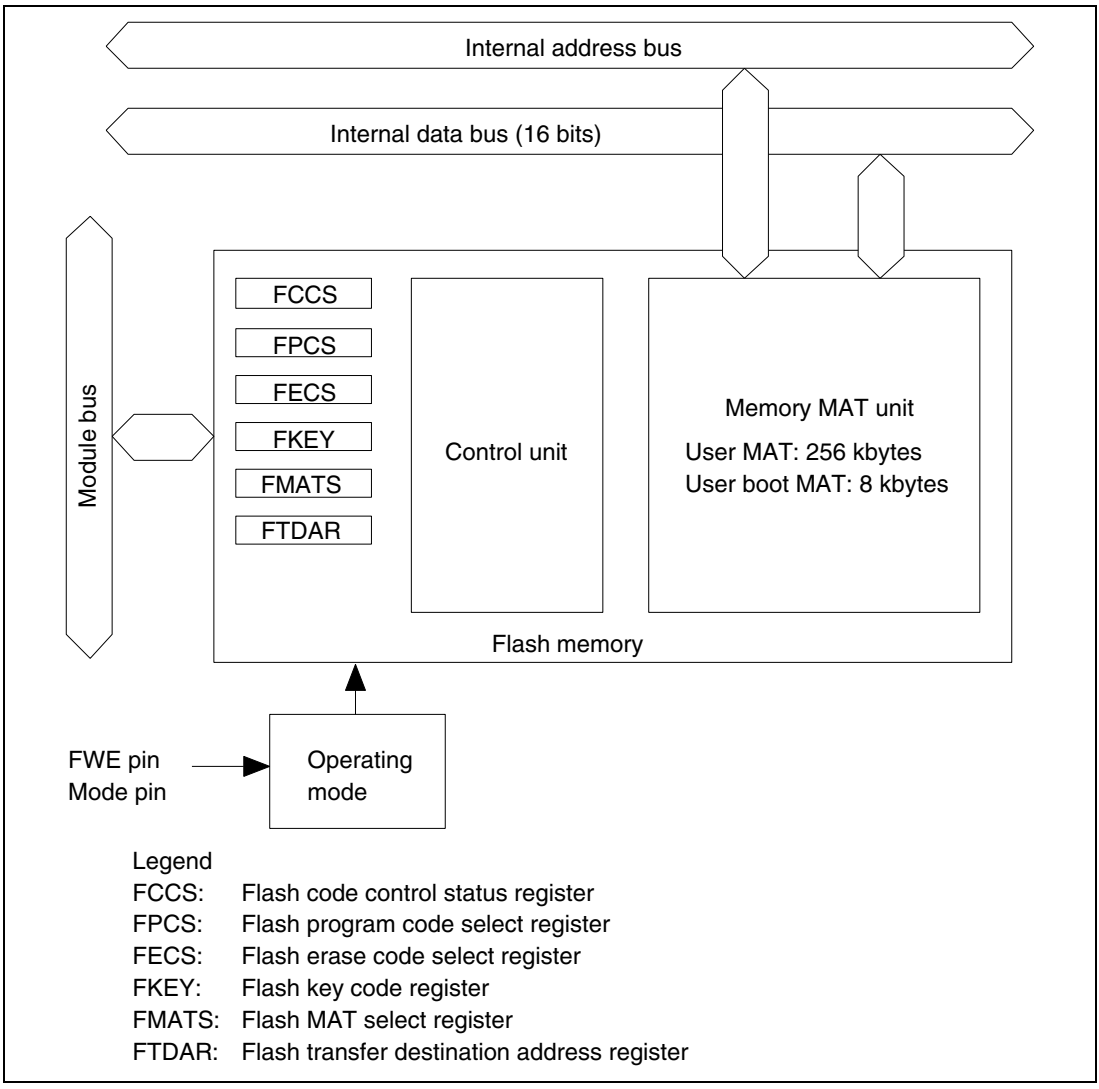


Figure 14.1 Block Diagram of Flash Memory

14.1.1 Operating Mode

When each mode pin and the FWE pin are set in the reset state and reset start is performed, this LSI enters each operating mode as shown in figure 14.2.

- Flash memory can be read in user mode, but cannot be programmed or erased.
- Flash memory can be read, programmed, or erased on the board only in user program mode, user boot mode, and boot mode.
- Flash memory can be read, programmed, or erased by means of the PROM programmer in programmer mode.

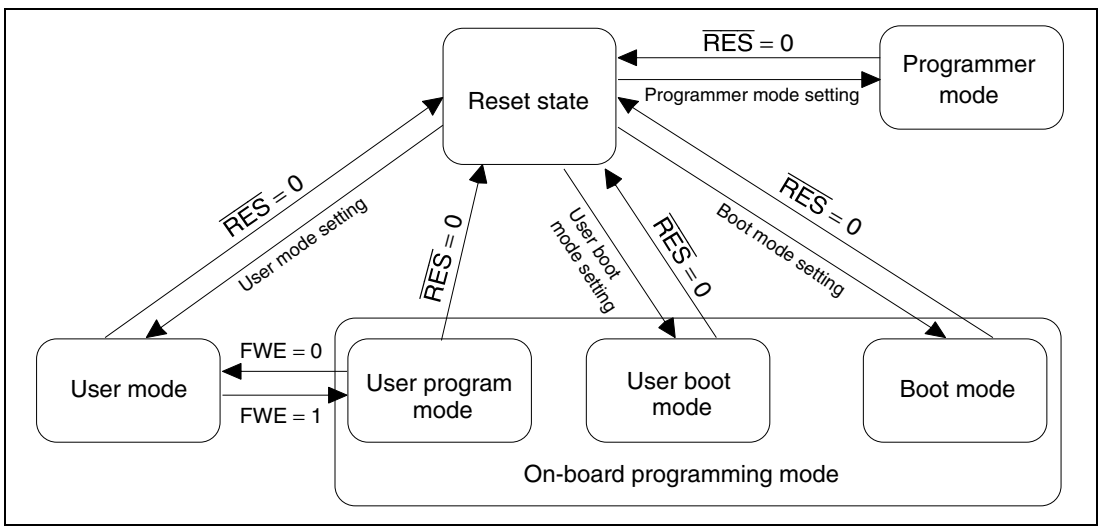


Figure 14.2 Mode Transition of Flash Memory

14.1.2 Mode Comparison

The comparison table of programming and erasing related items about boot mode, user program mode, user boot mode, and programmer mode is shown in table 14.1.

Table 14.1 Comparison of Programming Modes

	Boot mode	User program mode	User boot mode	Programmer mode
Programming/ erasing environment	On-board	On-board	On-board	PROM programmer
Programming/ erasing enable MAT	User MAT User boot MAT	User MAT	User MAT	User MAT User boot MAT
All erasure	○ (Automatic)	○	○	○ (Automatic)
Block division erasure	○ *1	○	○	×
Program data transfer	From host via SCI	From optional device via RAM	From optional device via RAM	Via programmer
Reset initiation MAT	Embedded program storage MAT	User MAT	User boot MAT*2	—
Transition to user mode	Changing mode setting and reset	Changing FLSHE bit and FWE pin	Changing mode setting and reset	—

Notes: 1. All-erasure is performed. After that, the specified block can be erased.
2. Firstly, the reset vector is fetched from the embedded program storage MAT. After the flash memory related registers are checked, the reset vector is fetched from the user boot MAT.

- The user boot MAT can be programmed or erased only in boot mode and programmer mode.
- The user MAT and user boot MAT are erased in boot mode. Then, the user MAT and user boot MAT can be programmed by means of the command method. However, the contents of the MAT cannot be read until this state.
  Only user boot MAT is programmed and the user MAT is programmed in user boot mode or only user MAT is programmed because user boot mode is not used.
- The boot operation of the optional interface can be performed by the mode pin setting different from user program mode in user boot mode.

14.1.3 Flash MAT Configuration

This LSI's flash memory is configured by the 256-kbyte user MAT and 8-kbyte user boot MAT.

The start address is allocated to the same address in the user MAT and user boot MAT. Therefore, when the program execution or data access is performed between two MATs, the MAT must be switched by using FMATS.

The user MAT or user boot MAT can be read in all modes. However, the user boot MAT can be programmed only in boot mode and programmer mode.

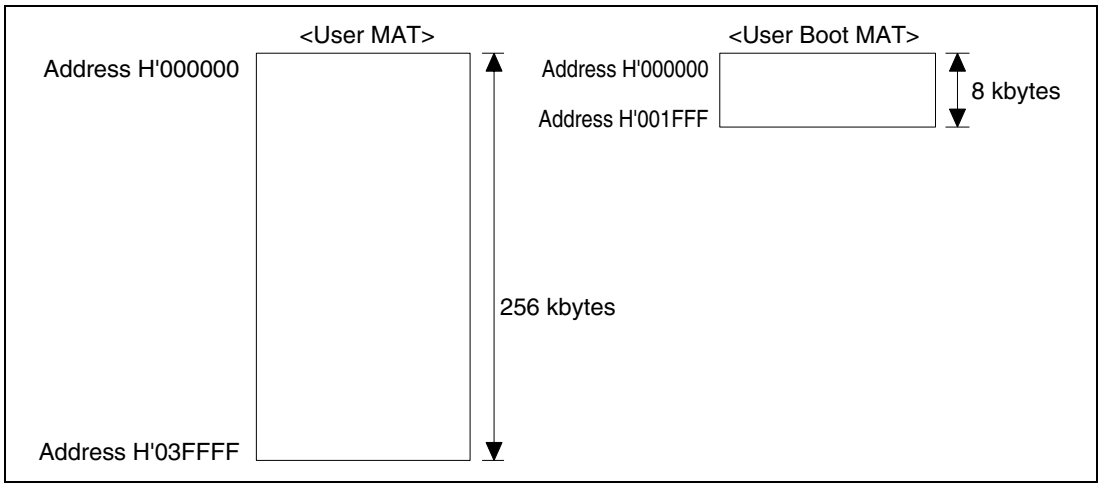


Figure 14.3 Flash Memory Configuration

The size of the user MAT is different from that of the user boot MAT. An address which exceeds the size of the 8-kbyte user boot MAT should not be accessed. If the attempt is made, data is read as undefined value.

14.1.4 Block Division

The user MAT is divided into 64 kbytes (three blocks), 32 kbytes (one block), and 4 kbytes (eight blocks) as shown in figure 14.4. The user MAT can be erased in this divided-block units and the erase-block number of EB0 to EB11 is specified when erasing.

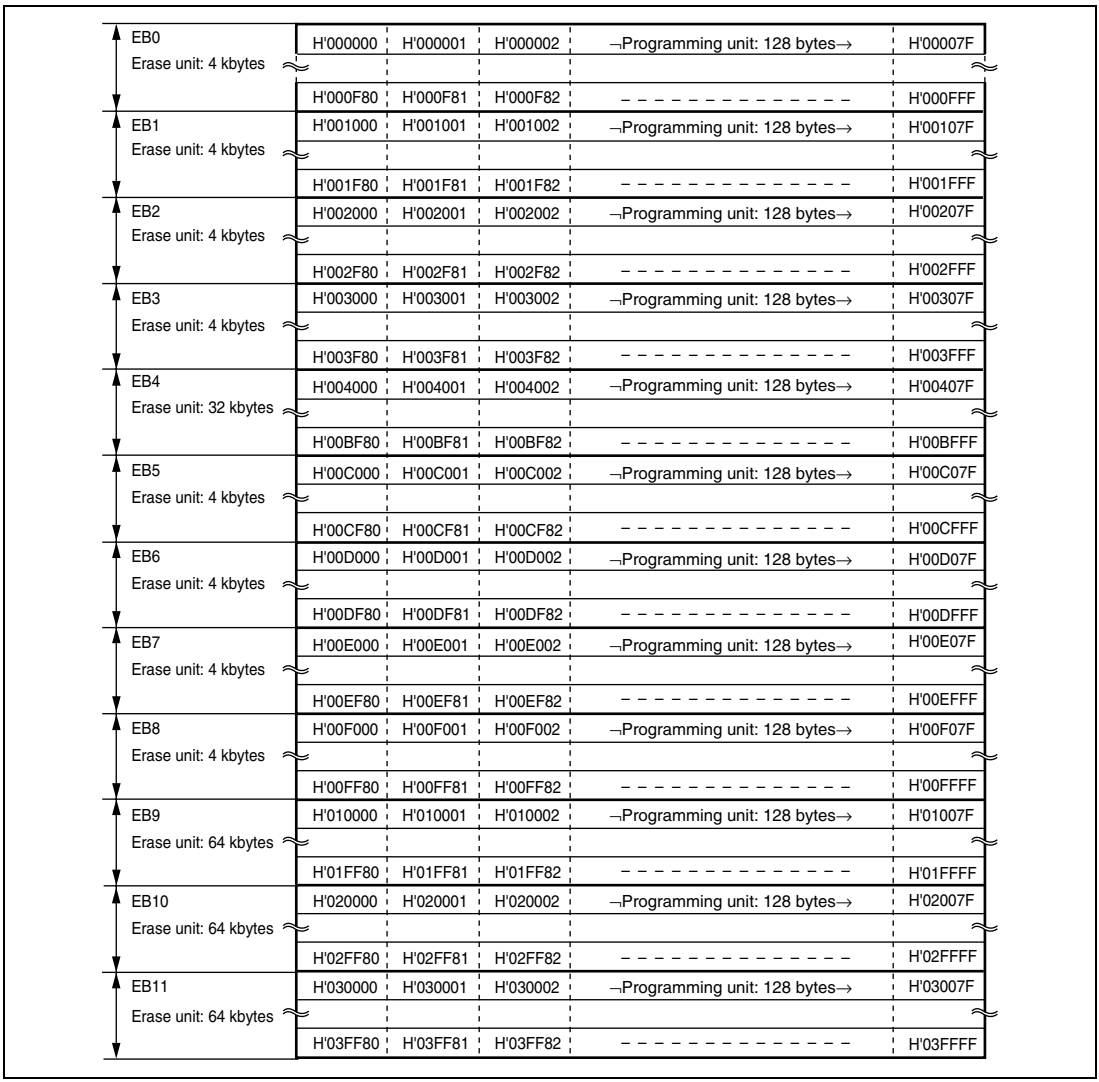


Figure 14.4 Block Division of User MAT

14.1.5 Programming/Erasing Interface

Programming/erasing is executed by downloading the on-chip program to the on-chip RAM and specifying the program address/data and erase block by using the interface register/parameter.

The procedure program is made by the user in user program mode and user boot mode. An overview of the procedure is given as follows. For details, see section 14.4.2, User Program Mode.

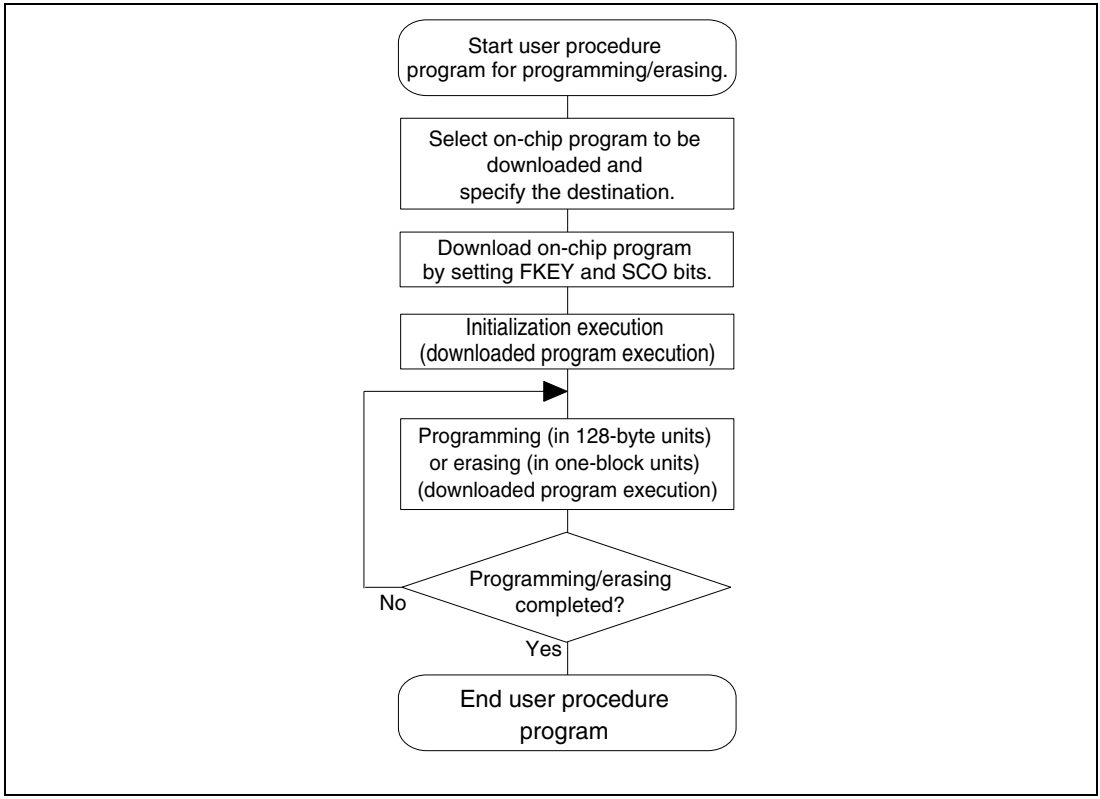


Figure 14.5 Overview of User Procedure Program

1. Selection of on-chip program to be downloaded

   For programming/erasing execution, the FWE pin must be set to 1 to transition to user program mode.

   This LSI has programming/erasing programs which can be downloaded to the on-chip RAM. The on-chip program to be downloaded is selected by setting the corresponding bits in the programming/erasing interface register. The address of the programming destination is specified by the flash transfer destination address register (FTDAR).

2. Download of on-chip program

   The on-chip program is automatically downloaded by setting the flash key register (FKEY) and the SCO bit in the flash control register (FCCS), which are programming/erasing interface registers.

   The flash memory is replaced to the embedded program storage area when downloading. Since the flash memory cannot be read when programming/erasing, the procedure program, which is working from download to completion of programming/erasing, must be executed in the space other than the flash memory to be programmed/erased (for example, on-chip RAM).

   Since the result of download is returned to the programming/erasing interface parameter, whether the normal download is executed or not can be confirmed.

3. Initialization of programming/erasing

   The operating frequency is set before execution of programming/erasing. This setting is performed by using the programming/erasing interface parameter.

4. Programming/erasing execution

   For programming/erasing execution, the FWE pin must be set to 1 to transition to user program mode.

   The program data/programming destination address is specified in 128-byte units when programming.

   The block to be erased is specified in erase-block units when erasing.

   These specifications are set by using the programming/erasing interface parameter and the on-chip program is initiated. The on-chip program is executed by using the JSR or BSR instruction and performing the subroutine call of the specified address in the on-chip RAM. The execution result is returned to the programming/erasing interface parameter.

   The area to be programmed must be erased in advance when programming flash memory.

   All interrupts are prohibited during programming and erasing. Interrupts must be masked within the user system.

5. When programming/erasing is executed consecutively

   When the processing is not ended by the 128-byte programming or one-block erasure, the program address/data and erase-block number must be updated and consecutive programming/erasing is required.

   Since the downloaded on-chip program is left in the on-chip RAM after the processing, download and initialization are not required when the same processing is executed consecutively.

14.2 Input/Output Pins

Table 14.2 shows the flash memory pin configuration.

Table 14.2 Pin Configuration

Pin Name	Input/Output	Function
$\overline{\text{RES}}$	Input	Reset
FWE	Input	Flash memory programming/erasing enable pin
$\overline{\text{MD2}}$	Input	Sets operating mode of this LSI
MD1	Input	Sets operating mode of this LSI
TxD0	Output	Serial transmit data output (used in boot mode)
RxD0	Input	Serial receive data input (used in boot mode)

14.3 Register Descriptions

The registers/parameters which control flash memory are shown in the following.

- Flash code control status register (FCCS)
- Flash program code select register (FPCS)
- Flash erase code select register (FECS)
- Flash key code register (FKEY)
- Flash MAT select register (FMATS)
- Flash transfer destination address register (FTDAR)
- Download pass and fail result (DPFR)
- Flash pass and fail result (FPFR)
- Flash multipurpose address area (FMPAR)
- Flash multipurpose data destination area (FMPDR)
- Flash erase Block select (FEBS)
- Flash program and erase frequency control (FPEFEQ)

There are several operating modes for accessing flash memory, for example, read mode/program mode.

There are two memory MATs: user MAT and user boot MAT. The dedicated registers/parameters are allocated for each operating mode and MAT selection. The correspondence of operating modes and registers/parameters for use is shown in table 14.3.

Table 14.3 Register/Parameter and Target Mode

		Download	Initialization	Programming	Erasure	Read
Programming/ Erasing Interface Register	FCCS	○	—	—	—	—
	FPCS	○	—	—	—	—
	FECS	○	—	—	—	—
	FKEY	○	—	○	○	—
	FMATS	—	—	○ *1	○ *1	○ *2
	FTDAR	○	—	—	—	—
Programming/ Erasing Interface Parameter	DPFR	○	—	—	—	—
	FPFR	—	○	○	○	—
	FPEFEQ	—	○	—	—	—
	FUBRA	—	○	—	—	—
	FMPAR	—	—	○	—	—
	FMPDR	—	—	○	—	—
	FEBS	—	—	—	○	—

Notes: 1. The setting is required when programming or erasing user MAT in user boot mode.
2. The setting may be required according to the combination of initiation mode and read target MAT.

14.3.1 Programming/Erasing Interface Register

The programming/erasing interface registers are as described below. They are all 8-bit registers that can be accessed in byte. These registers are initialized at a reset or in hardware standby mode.

- Flash Code Control and Status Register (FCCS)

FCCS is configured by bits which request the monitor of the FWE pin state and error occurrence during programming or erasing flash memory and the download of on-chip program.

Bit	Bit Name	Initial Value	R/W	Description
7	FWE	1/0	R	Flash Program Enable Monitors the signal level input to the FWE pin. 0: A low level signal is input to the FWE pin. (Hardware protection state) 1: A high level signal is input to the FWE pin.
6, 5	—	All 0	R/W	Reserved The initial value should not be changed.
4	FLER	0	R	Flash Memory Error Indicates an error occurs during programming and erasing flash memory. When FLER is set to 1, flash memory enters the error protection state. When FLER is set to 1, high voltage is applied to the internal flash memory. To reduce the damage to flash memory, the reset must be released after the reset period of 100 μs which is longer than normal. 0: Flash memory operates normally. Programming/erasing protection for flash memory (error protection) is invalid. [Clearing condition] • At a reset or in hardware standby mode 1: An error occurs during programming/erasing flash memory. Programming/erasing protection for flash memory (error protection) is valid. [Setting conditions] • When an interrupt, such as NMI, occurs during programming/erasing flash memory. • When the flash memory is read during programming/erasing flash memory (including a vector read or an instruction fetch). • When the SLEEP instruction is executed during programming/erasing flash memory (including software-standby mode) • When a bus master other than the CPU, such as the DMAC, gets bus mastership during programming/erasing flash memory.

Bit	Bit Name	Initial Value	R/W	Description
3	WEINTE	0	R/W	Program/Erase Enable Modifies the space for the interrupt vector table, when interrupt vector data is not read successfully during programming/erasing flash memory or switching between a user MAT and a user boot MAT. When this bit is set to 1, interrupt vector data is read from address spaces H′FF8000 to H′FF807F (on-chip RAM space), instead of from address spaces H′000000 to H′00017F (up to vector number 31). Therefore, make sure to set the vector table in the on-chip RAM space before setting this bit to 1. The interrupt exception handling on and after vector number 32 should not be used because the correct vector is not read, resulting in the CPU runaway. 0: The space for the interrupt vector table is not modified. When interrupt vector data is not read successfully, the operation for the interrupt exception handling cannot be guaranteed. An occurrence of any interrupts should be masked. 1: The space for the interrupt vector table is modified. Even when interrupt vector data is not read successfully, the interrupt exception handling up to vector number 31 is enabled.
2, 1	—	All 0	R/W	Reserved The initial value should not be changed.
0	SCO	0	(R)/W*	Source Program Copy Operation Requests the on-chip programming/erasing program to be downloaded to the on-chip RAM. When this bit is set to 1, the on-chip program which is selected by FPCS/FECS is automatically downloaded in the on-chip RAM specified by FTDAR. In order to set this bit to 1, H′A5 must be written to FKEY and this operation must be executed in the on-chip RAM. Four NOP instructions must be executed immediately after setting this bit to 1. Since this bit is cleared to 0 when download is completed, this bit cannot be read as 1. All interrupts must be disabled. This should be made in the user system.

Bit	Bit Name	Initial Value	R/W	Description
0	SCO	0	(R)/W*	0: Download of the on-chip programming/erasing program to the on-chip RAM is not executed. [Clearing condition] When download is completed 1: Request that the on-chip programming/erasing program is downloaded to the on-chip RAM is occurred. [Setting conditions] When all of the following conditions are satisfied and 1 is set to this bit • H′A5 is written to FKEY • During execution in the on-chip RAM

Note:* This bit is a write only bit. This bit is always read as 0.

- Flash Program Code Select Register (FPCS)

FPCS selects the on-chip programming program to be downloaded.

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R/W	Reserved The initial value should not be changed.
0	PPVS	0	R/W	Program Pulse Verify Selects the programming program. 0: On-chip programming program is not selected. [Clearing condition] When transfer is completed 1: On-chip programming program is selected.

- Flash Erase Code Select Register (FECS)

FECS selects download of the on-chip erasing program.

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R/W	Reserved The initial value should not be changed.
0	EPVB	0	R/W	Erase Pulse Verify Block Selects the erasing program. 0: On-chip erasing program is not selected. [Clearing condition] When transfer is completed 1: On-chip erasing program is selected.

- Flash Key Code Register (FKEY)

FKEY is a register for software protection that enables download of on-chip program and programming/erasing of flash memory. Before setting the SCO bit to 1 in order to download on-chip program or executing the downloaded programming/erasing program, these processing cannot be executed if the key code is not written.

Bit	Bit Name	Initial Value	R/W	Description
7	K7	0	R/W	Key Code
6	K6	0	R/W	Only when H′A5 is written, writing to the SCO bit is valid. When the value other than H′A5 is written to FKEY, 1 cannot be set to the SCO bit. Therefore downloading to the on-chip RAM cannot be executed. Only when H′5A is written, programming/erasing can be executed. Even if the on-chip programming/erasing program is executed, the flash memory cannot be programmed or erased when the value other than H′5A is written to FKEY. H′A5: Writing to the SCO bit is enabled. (The SCO bit cannot be set by the value other than H′A5.) H′5A: Programming/erasing is enabled. (The value other than H′A5 is in software protection state.) H′00: Initial value
5	K5	0	R/W	
4	K4	0	R/W	
3	K3	0	R/W	
2	K2	0	R/W	
1	K1	0	R/W	
0	K0	0	R/W	

- Flash MAT Select Register (FMATS)

FMATS specifies whether user MAT or user boot MAT is selected.

Bit	Bit Name	Initial Value	R/W	Description
7	MS7	0/1*	R/W	MAT Select These bits are in user-MAT selection state when the value other than H′AA is written and in user-boot-MAT selection state when H′AA is written. The MAT is switched by writing the value in FMATS. When the MAT is switched, follow section 14.6, Switching between User MAT and User Boot MAT. (The user boot MAT cannot be programmed in user program mode if user boot MAT is selected by FMATS. The user boot MAT must be programmed in boot mode or in programmer mode.) H′AA: The user boot MAT is selected (in user-MAT selection state when the value of these bits are other than H′AA) Initial value when these bits are initiated in user boot mode. H′00: Initial value when these bits are initiated in a mode except for user boot mode (in user-MAT selection state) [Programmable condition] These bits are in the execution state in the on-chip RAM.
6	MS6	0	R/W	
5	MS5	0/1*	R/W	
4	MS4	0	R/W	
3	MS3	0/1*	R/W	
2	MS2	0	R/W	
1	MS1	0/1*	R/W	
0	MS0	0	R/W	

Note:* Set to 1 when in user boot mode, otherwise set to 0.

- Flash Transfer Destination Address Register (FTDAR)

FTDAR specifies the on-chip RAM address where an on-chip program is downloaded. This register must be specified before setting the SCO bit in FCCS to 1.

Bit	Bit Name	Initial Value	R/W	Description
7	TDER	0	R/W	Transfer Destination Address Setting Error This bit is set to 1 when the address specified by bits TDA6 to TDA0, which is the start address where an on-chip program is downloaded, is over the range. Whether or not the address specified by bits TDA6 to TDA0 is within the range of H′00 to H′03 is determined when an on-chip program is downloaded by setting the SCO bit in FCCS to 1. Make sure that this bit is cleared to 0 and the value specified by bits TDA6 to TDA0 is within the range of H′00 to H′03 before setting the SCO bit to 1. 0: The value specified by bits TDA6 to TDA0 is within the range. 1: The value specified by bits TDA6 to TDA0 is over the range (H′04 to H′FF) and download is stopped.
6	TDA6	0	R/W	Transfer Destination Address Specifies the start address where an on-chip program is downloaded. A value from H′00 to H′03 can be specified as the download start address in the on-chip RAM. H′00: H′FF7000 is specified as the download start address. H′01: H′FFB000 is specified as the download start address. H′02: H′FFD000 is specified as the download start address. H′03: H′FFE800 is specified as the download start address. H′04 to H′FF: Setting prohibited. Specifying this value sets the TDRE bit to 1 during downloading and stops the download.
5	TDA5	0	R/W	
4	TDA4	0	R/W	
3	TDA3	0	R/W	
2	TDA2	0	R/W	
1	TDA1	0	R/W	
0	TDA0	0	R/W	

- RAM Emulation Register (RAMER)

RAMER specifies the area of flash memory to be overlapped with part of RAM when emulating realtime flash memory programming. RAMER is initialized to H′00 by a power-on 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.

Flash memory area divisions are shown in table 14.4. 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	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 0	—	Reserved These bits are always read as 0. The write value should always be 0.
3	RAMS	0	R/W	RAM Select Specifies selection or non-selection of flash memory emulation in RAM. When RAMS is 1, all flash memory blocks are program/erase-protected. 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.
2	RAM2	0	R/W	Flash Memory Area Selection These bits are used together with bit 3 to select the flash memory area to be overlapped with RAM (see table 14.4).
1	RAM1	0	R/W	
0	RAM0	0	R/W	

Table 14.4 Flash Memory Area Divisions

RAM Area	Block Name	RAM2	RAM1	RAM0
H′000000 to H′000FFF	EB0 (4 kbytes)	0	0	0
H′001000 to H′001FFF	EB1 (4 kbytes)	0	0	1
H′002000 to H′002FFF	EB2 (4 kbytes)	0	1	0
H′003000 to H′003FFF	EB3 (4 kbytes)	0	1	1
H′004000 to H′004FFF	EB4 (4 kbytes)	1	0	0
H′005000 to H′005FFF	EB5 (4 kbytes)	1	0	1
H′006000 to H′006FFF	EB6 (4 kbytes)	1	1	0
H′007000 to H′007FFF	EB7 (4 kbytes)	1	1	1

14.3.2 Programming/Erasing Interface Parameter

The programming/erasing interface parameter specifies the operating frequency, storage place for program data, programming destination address, and erase block and exchanges the processing result for the downloaded on-chip program. This parameter uses the general registers of the CPU (ER0 and ER1) or the on-chip RAM area. The initial value is undefined at a reset or in hardware standby mode.

When download, initialization, or on-chip program is executed, registers of the CPU except for R0L are stored. The return value of the processing result is written in R0L. Since the stack area is used for storing the registers except for R0L, the stack area must be saved at the processing start. (A maximum size of a stack area to be used is 132 bytes.)

The programming/erasing interface parameter is used in the following four items.

1. Download control
2. Initialization before programming or erasing
3. Programming
4. Erasing

These items use different parameters. The correspondence table is shown in table 14.5. The meaning of the bits in FPFR varies in each processing program: initialization, programming, or erasure. For details, see descriptions of FPFR for each process.

Table 14.5 Parameters and Target Modes

Name of Parameter	Abbreviation	Down Load	Initialization	Programming	Erasure	R/W	Initial Value	Allocation
Download pass and fail result	DPFR	○	—	—	—	R/W	Undefined	On-chip RAM*
Flash pass and fail result	FPFR	—	○	○	○	R/W	Undefined	R0L of CPU
Flash programming/ erasing frequency control	FPEFEQ	—	○	—	—	R/W	Undefined	R0 of CPU
Flash multipurpose address area	FMPAR	—	—	○	—	R/W	Undefined	On-chip RAM
Flash multipurpose data destination area	FMPDR	—	—	○	—	R/W	Undefined	On-chip RAM
Flash erase block select	FEBS	—	—	—	○	R/W	Undefined	R0L of CPU

Note:* A single byte of the start address to download an on-chip program, which is specified by FTDAR

(1) Download Control

The on-chip program is automatically downloaded by setting the SCO bit to 1. The on-chip RAM area to be downloaded is the 2-kbyte area starting from the address specified by FTDAR.

Download control is set by the program/erase interface registers, and the DPFR parameter indicates the return value.

(a) Download pass/fail result parameter (DPFR: single byte of start address specified by FTDAR)

This parameter indicates the return value of the download result. The value of this parameter can be used to determine if downloading is executed or not. Since the confirmation whether the SCO bit is set to 1 is difficult, the certain determination must be performed by writing the single byte of the start address specified by FTDAR to the value other than the return value of download (for example, H′FF) before the download start (before setting the SCO bit to 1).

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	—	—	Unused Return 0
2	SS	—	R/W	Source Select Error Detect Only one type for the on-chip program which can be downloaded can be specified. When more than two types of the program are selected, the program is not selected, or the program is selected without mapping, error is occurred. 0: Download program can be selected normally 1: Download error is occurred (multi-selection or program which is not mapped is selected)
1	FK	—	R/W	Flash Key Register Error Detect Returns the check result whether the value of FKEY is set to H′A5. 0: KEY setting is normal (FKEY = H′A5) 1: Setting value of FKEY becomes error (FKEY = value other than H′A5)
0	SF	—	R/W	Success/Fail Returns the result whether download is ended normally or not. The determination result whether program that is downloaded to the on-chip RAM is read back and then transferred to the on-chip RAM is returned. 0: Downloading on-chip program is ended normally (no error) 1: Downloading on-chip program is ended abnormally (error occurs)

(2) Programming/Erasing Initialization

The on-chip programming/erasing program to be downloaded includes the initialization program.

The specified period pulse must be applied when programming or erasing. The specified pulse width is made by the method in which wait loop is configured by the CPU instruction. The operating frequency of the CPU must be set.

The initial program is set as a parameter of the programming/erasing program which has downloaded these settings.

(a) Flash programming/erasing frequency parameter (FPEFEQ: general register ER0 of CPU)

This parameter sets the operating frequency of the CPU. The settable range of the operating frequency in this LSI is 10 to 33 MHz.

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	—	—	—	Unused This bit should be cleared to 0.
15 to 0	F15 to F0	—	R/W	Frequency Set Set the operating frequency of the CPU. With the PLL multiplication function, set the frequency multiplied. The setting value must be calculated as the following methods. 1. The operating frequency which is shown in MHz units must be rounded in a number to three decimal places and be shown in a number of two decimal places. 2. The value multiplied by 100 is converted to the binary digit and is written to the FPEFEQ parameter (general register ER0). For example, when the operating frequency of the CPU is 33.000 MHz, the value is as follows. 1. The number to three decimal places of 33.000 is rounded and the value is thus 33.00. 2. The formula that 33.00 × 100 = 3300 is converted to the binary digit and B′0000,1100,1110,0100 (H′0CE4) is set to ER0.

(b) Flash pass/fail parameter (FPFR: general register R0L of CPU)

This parameter indicates the return value of the initialization result.

Bit	Bit Name	Initial Value	R/W	Description
7 to 2	—	—	—	Unused Return 0
1	FQ	—	R/W	Frequency Error Detect Returns the check result whether the specified operating frequency of the CPU is in the range of the supported operating frequency. 0: Setting of operating frequency is normal 1: Setting of operating frequency is abnormal
0	SF	—	R/W	Success/Fail Indicates whether initialization is completed normally. 0: Initialization is ended normally (no error) 1: Initialization is ended abnormally (error occurs)

(3) Programming Execution

When flash memory is programmed, the programming destination address on the user MAT must be passed to the programming program in which the program data is downloaded.

1. The start address of the programming destination on the user MAT must be stored in a general register ER1. This parameter is called as flash multipurpose address area parameter (FMPAR).
   Since the program data is always in units of 128 bytes, the lower eight bits (A7 to A0) must be H′00 or H′80 as the boundary of the programming start address on the user MAT.
2. The program data for the user MAT must be prepared in the consecutive area. The program data must be in the consecutive space which can be accessed by using the MOV.B instruction of the CPU and in other than the flash memory space.
   When data to be programmed does not satisfy 128 bytes, the 128-byte program data must be prepared by filling with the dummy code H′FF.
   The start address of the area in which the prepared program data is stored must be stored in a general register ER0. This parameter is called as flash multipurpose data destination area parameter (FMPDR).

For details on the program processing procedure, see section 14.4.2, User Program Mode.

(a) Flash multipurpose address area parameter (FMPAR: general register ER1 of CPU)

This parameter stores the start address of the programming destination on the user MAT.

When the address in the area other than flash memory space is set, an error occurs.

The start address of the programming destination must be at the 128-byte boundary. If this boundary condition is not satisfied, an error occurs. The error occurrence is indicated by the WA bit (bit 1) in FPFR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOA31 to MOA0	—	R/W	Store the start address of the programming destination on the user MAT. The consecutive 128-byte programming is executed starting from the specified start address of the user MAT. Therefore, the specified programming start address becomes a 128-byte boundary and MOA6 to MOA0 are always 0.

(b) Flash multipurpose data destination parameter (FMPDR: general register ER0 of CPU):

This parameter stores the start address in the area which stores the data to be programmed in the user MAT. When the storage destination of the program data is in flash memory, an error occurs. The error occurrence is indicated by the WD bit in FPFR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOD31 to MOD0	—	R/W	Store the start address of the area which stores the program data for the user MAT. The consecutive 128-byte data is programmed to the user MAT starting from the specified start address.

(c) Flash pass/fail parameter (FPFR: general register R0L of CPU)

This parameter indicates the return value of the program processing result.

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused Return 0.

Bit	Bit Name	Initial Value	R/W	Description
6	MD	—	R/W	Programming Mode Related Setting Error Detect Returns the check result that a high level signal is input to the FWE pin and the error protection state is not entered. When the low level signal is input to the FWE pin or the error protection state is entered, 1 is written to this bit. The state can be confirmed with the FWE and FLER bits in FCCS. For conditions to enter the error protection state, see section 14.5.3, Error Protection. 0: FWE and FLER settings are normal (FWE = 1, FLER = 0) 1: Programming cannot be performed (FWE = 0 or FLER = 1)
5	EE	—	R/W	Programming Execution Error Detect 1 is returned to this bit when the specified data could not be written because the user MAT was not erased. If this bit is set to 1, there is a high possibility that the user MAT is partially rewritten. In this case, after removing the error factor, erase the user MAT. If FMATS is set to H'AA and the user boot MAT is selected, an error occurs when programming is performed. In this case, both the user MAT and user boot MAT are not rewritten. Programming of the user boot MAT should be performed in boot mode or programmer mode. 0: Programming has ended normally 1: Programming has ended abnormally (programming result is not guaranteed)
4	FK	—	R/W	Flash Key Register Error Detect Returns the check result of the value of FKEY before the start of the programming processing. 0: FKEY setting is normal (FKEY = H′5A) 1: FKEY setting is error (FKEY = value other than H′5A)
3	—	—	—	Unused Returns 0.
2	WD	—	R/W	Write Data Address Detect When the address in the flash memory area is specified as the start address of the storage destination of the program data, an error occurs. 0: Setting of write data address is normal 1: Setting of write data address is abnormal

Bit	Bit Name	Initial Value	R/W	Description
1	WA	—	R/W	Write Address Error Detect When the following items are specified as the start address of the programming destination, an error occurs. • When the programming destination address in the area other than flash memory is specified • When the specified address is not in a 128-byte boundary. (The lower eight bits of the address are other than H′00 and H′80.) 0: Setting of programming destination address is normal 1: Setting of programming destination address is abnormal
0	SF	—	R/W	Success/Fail Indicates whether the program processing is ended normally or not. 0: Programming is ended normally (no error) 1: Programming is ended abnormally (error occurs)

(4) Erasure Execution

When flash memory is erased, the erase-block number on the user MAT must be passed to the erasing program which is downloaded. This is set to the FEBS parameter (general register ER0).

One block is specified from the block number 0 to 11.

For details on the erasing processing procedure, see section 14.4.2, User Program Mode.

(a) Flash erase block select parameter (FEBS: general register ER0 of CPU)

This parameter specifies the erase-block number. The several block numbers cannot be specified.

Bit	Bit Name	Initial Value	R/W	Description
31 to 8	—	—	—	Unused These bits should be cleared to H′0.
7	EB7	—	R/W	Erase Block Set the erase-block number in the range from 0 to 11. 0 corresponds to the EB0 block and 11 corresponds to the EB11 block. An error occurs when the number other than 0 to 11 is set.
6	EB6	—	R/W	
5	EB5	—	R/W	
4	EB4	—	R/W	
3	EB3	—	R/W	
2	EB2	—	R/W	
1	EB1	—	R/W	
0	EB0	—	R/W	

(b) Flash pass/fail parameter (FPFR: general register R0L of CPU)

This parameter returns value of the erasing processing result.

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused Return 0.
6	MD	—	R/W	Programming Mode Related Setting Error Detect Returns the check result that a high level signal is input to the FWE pin and the error protection state is not entered. When the low level signal is input to the FWE pin or the error protection state is entered, 1 is written to this bit. The state can be confirmed with the FWE and FLER bits in FCCS. For conditions to enter the error protection state, see section 14.5.3, Error Protection. 0: FWE and FLER settings are normal (FWE = 1, FLER = 0) 1: Programming cannot be performed (FWE = 0 or FLER = 1)
5	EE	—	R/W	Erasure Execution Error Detect 1 is returned to this bit when the user MAT could not be erased or when flash-memory related register settings are partially changed. If this bit is set to 1, there is a high possibility that the user MAT is partially erased. In this case, after removing the error factor, erase the user MAT. If FMATS is set to H′AA and the user boot MAT is selected, an error occurs when erasure is performed. In this case, both the user MAT and user boot MAT are not erased. Erasing of the user boot MAT should be performed in boot mode or programmer mode. 0: Erasure has ended normally 1: Erasure has ended abnormally (erasure result is not guaranteed)
4	FK	—	R/W	Flash Key Register Error Detect Returns the check result of FKEY value before start of the erasing processing. 0: FKEY setting is normal (FKEY = H′5A) 1: FKEY setting is error (FKEY = value other than H′5A)
3	EB	—	R/W	Erase Block Select Error Detect Returns the check result whether the specified erase-block number is in the block range of the user MAT. 0: Setting of erase-block number is normal 1: Setting of erase-block number is abnormal

Bit	Bit Name	Initial Value	R/W	Description
2, 1	—	—	—	Unused Return 0.
0	SF	—	R/W	Success/Fail Indicates whether the erasing processing is ended normally or not. 0: Erasure is ended normally (no error) 1: Erasure is ended abnormally (error occurs)

14.4 On-Board Programming Mode

When the pin is set in on-board programming mode and the reset start is executed, the on-board programming state that can program/erase the on-chip flash memory is entered. On-board programming mode has three operating modes: boot mode, user program mode, and user boot mode.

For details of the pin setting for entering each mode, see table 14.6. For details of the state transition of each mode for flash memory, see figure 14.2.

Table 14.6 Setting On-Board Programming Mode

Mode Setting	FWE	$\overline{\text{MD2}}$	MD1	NMI
Boot mode	1	0	0	1
User program mode	1	1	1	0/1
User boot mode	1	0	0	0

14.4.1 Boot Mode

Boot mode executes programming/erasing user MAT and user boot MAT by means of the control command and program data transmitted from the host using the on-chip SCI. The tool for transmitting the control command and program data must be prepared in the host. The SCI communication mode is set to asynchronous mode. When reset start is executed after this LSI's pin is set in boot mode, the boot program in the microcomputer is initiated. After the SCI bit rate is automatically adjusted, the communication with the host is executed by means of the control command method.

The system configuration diagram in boot mode is shown in figure 14.6. For details on the pin setting in boot mode, see table 14.6. The NMI and other interrupts are ignored in boot mode. However, the NMI and other interrupts should be disabled in the user system.

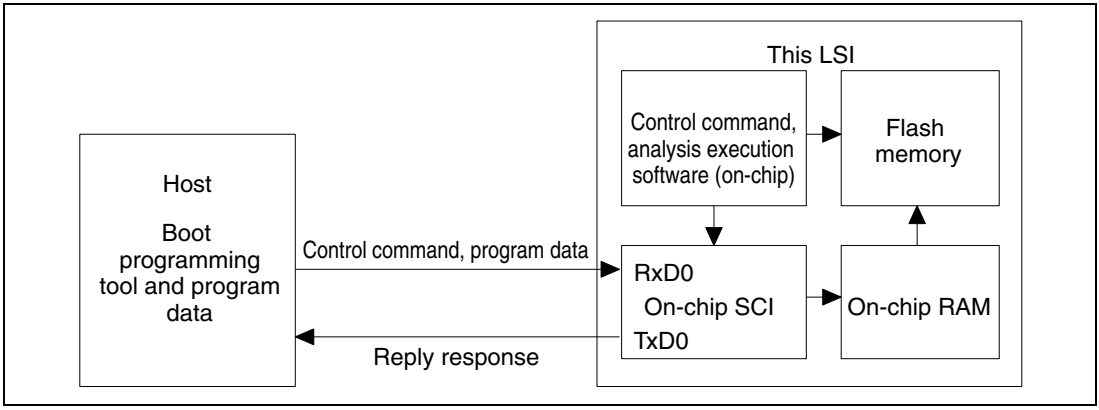


Figure 14.6 System Configuration in Boot Mode

(1) SCI Interface Setting by Host

When boot mode is initiated, this LSI measures the low period of asynchronous SCI-communication data (H′00), which is transmitted consecutively by the host. The SCI transmit/receive format is set to 8-bit data, 1 stop bit, and no parity. This LSI calculates the bit rate of transmission by the host by means of the measured low period and transmits the bit adjustment end sign (1 byte of H′00) to the host. The host must confirm that this bit adjustment end sign (H′00) has been received normally and transmits 1 byte of H′55 to this LSI. When reception is not executed normally, boot mode is initiated again (reset) and the operation described above must be executed. The bit rate between the host and this LSI is not matched by the bit rate of transmission by the host and system clock frequency of this LSI. To operate the SCI normally, the transfer bit rate of the host must be set to 4,800 bps, 9,600 bps, or 19,200 bps.

The system clock frequency, which can automatically adjust the transfer bit rate of the host and the bit rate of this LSI, is shown in table 14.7. Boot mode must be initiated in the range of this system clock.

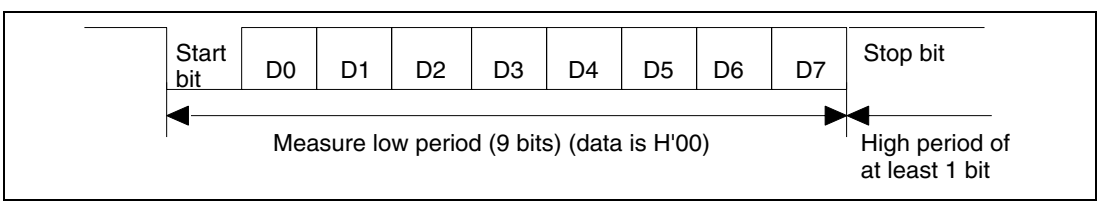


Figure 14.7 Automatic-Bit-Rate Adjustment Operation of SCI

Table 14.7 System Clock Frequency for Automatic-Bit-Rate Adjustment by This LSI

Bit Rate of Host	System Clock Frequency
4,800 bps	5 to 33 MHz
9,600 bps	5 to 33 MHz
19,200 bps	8 to 33 MHz

(2) State Transition Diagram

The overview of the state transition diagram after boot mode is initiated is shown in figure 14.8.

1. Bit rate adjustment

   After boot mode is initiated, the bit rate of the SCI interface is adjusted with that of the host.
2. Waiting for inquiry set command

   For inquiries about user-MAT size and configuration, MAT start address, and support state, the required information is transmitted to the host.
3. Automatic erasure of all user MAT and user boot MAT

   After inquiries have finished, all user MAT and user boot MAT are automatically erased.
4. Waiting for programming/erasing command
   - When the program preparation notice is received, the state for waiting program data is entered. The programming start address and program data must be transmitted following the programming command. When programming is finished, the programming start address must be set to H′FFFFFFFF and transmitted. Then the state for waiting program data is returned to the state of programming/erasing command wait.
   - When the erasure preparation notice is received, the state for waiting erase-block data is entered. The erase-block number must be transmitted following the erasing command. When the erasure is finished, the erase-block number must be set to H′FF and transmitted. Then the state for waiting erase-block data is returned to the state for waiting programming/erasing command. The erasure must be used when the specified block is programmed without a reset start after programming is executed in boot mode. When programming can be executed by only one operation, all blocks are erased before the state for waiting programming/erasing/other command is entered. The erasing operation is not required.
   - There are many commands other than programming/erasing. Examples are sum check, blank check (erasure check), and memory read of the user MAT/user boot MAT and acquisition of current status information.

Note that memory read of the user MAT/user boot MAT can only read the programmed data after all user MAT/user boot MAT has automatically been erased.

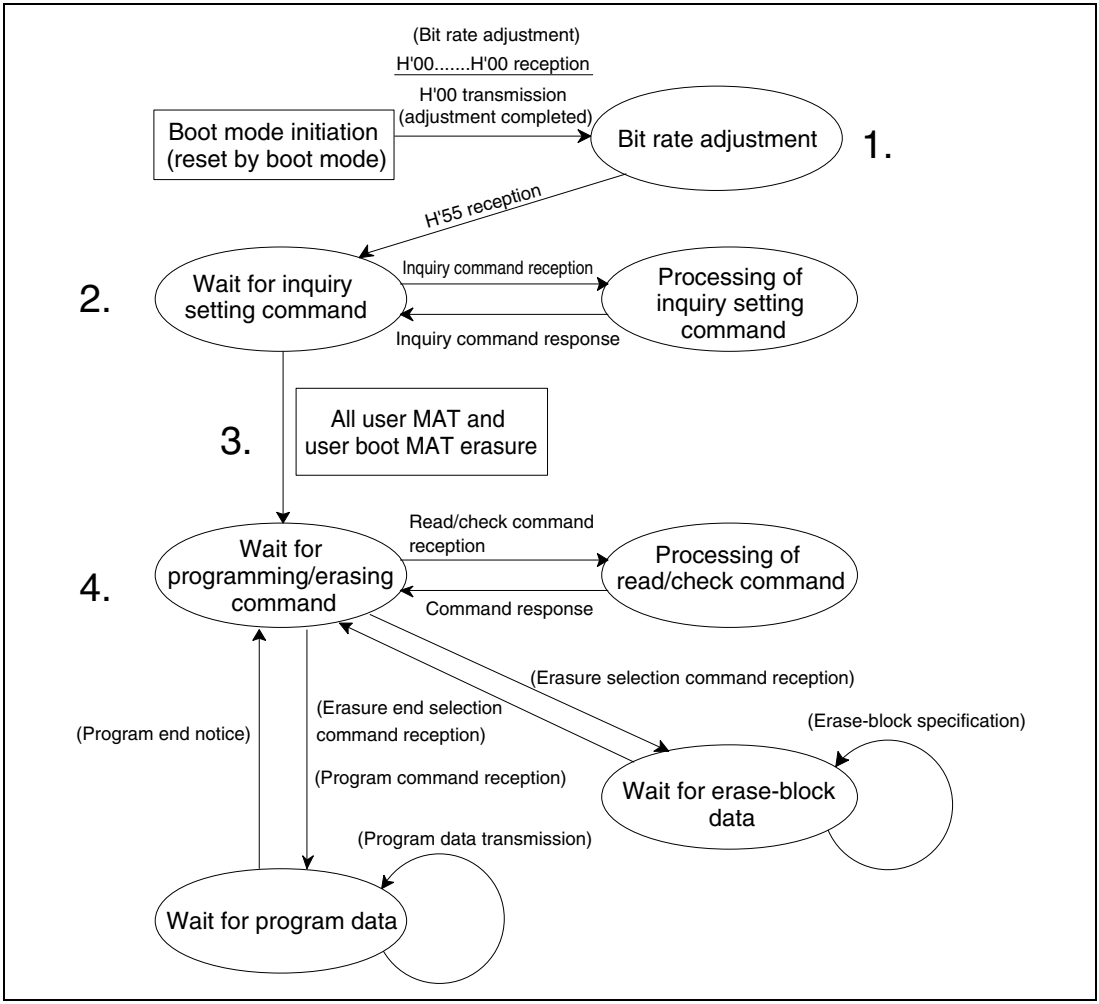


Figure 14.8 Overview of Boot Mode State Transition Diagram

14.4.2 User Program Mode

The user MAT can be programmed/erased in user program mode. (The user boot MAT cannot be programmed/erased.)

Programming/erasing is executed by downloading the program in the microcomputer.

The overview flow is shown in figure 14.9.

High voltage is applied to internal flash memory during the programming/erasing processing. Therefore, transition to reset or hardware standby must not be executed. Doing so may damage or destroy flash memory. If reset is executed accidentally, reset must be released after the reset input period of 100 µs which is longer than normal.

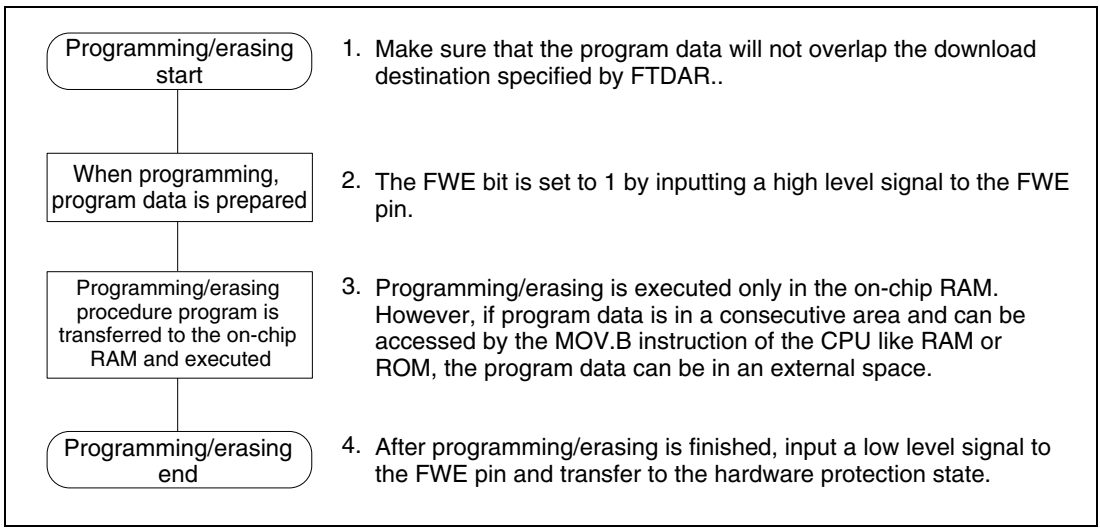


Figure 14.9 Programming/Erasing Overview Flow

(1) On-chip RAM Address Map when Programming/Erasing is Executed

Parts of the procedure program that are made by the user, like download request, programming/erasing procedure, and determination of the result, must be executed in the on-chip RAM. The on-chip program that is to be downloaded is all in the on-chip RAM. Note that area in the on-chip RAM must be controlled so that these parts do not overlap.

Figure 14.10 shows the program area to be downloaded.

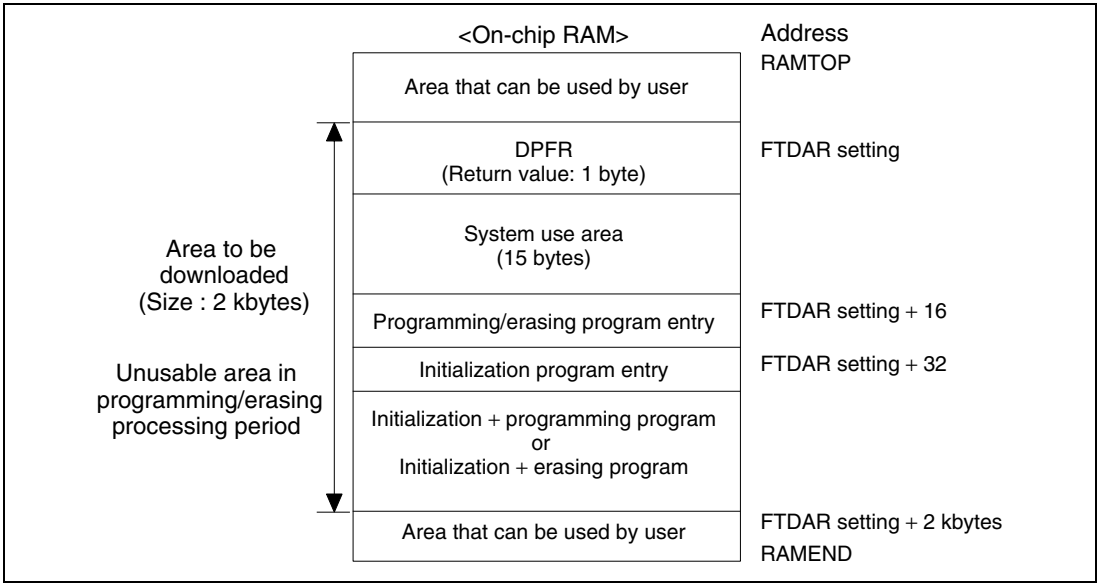


Figure 14.10 RAM Map When Programming/Erasing is Executed

(2) Programming Procedure in User Program Mode

The procedures for download, initialization, and programming are shown in figure 14.11.

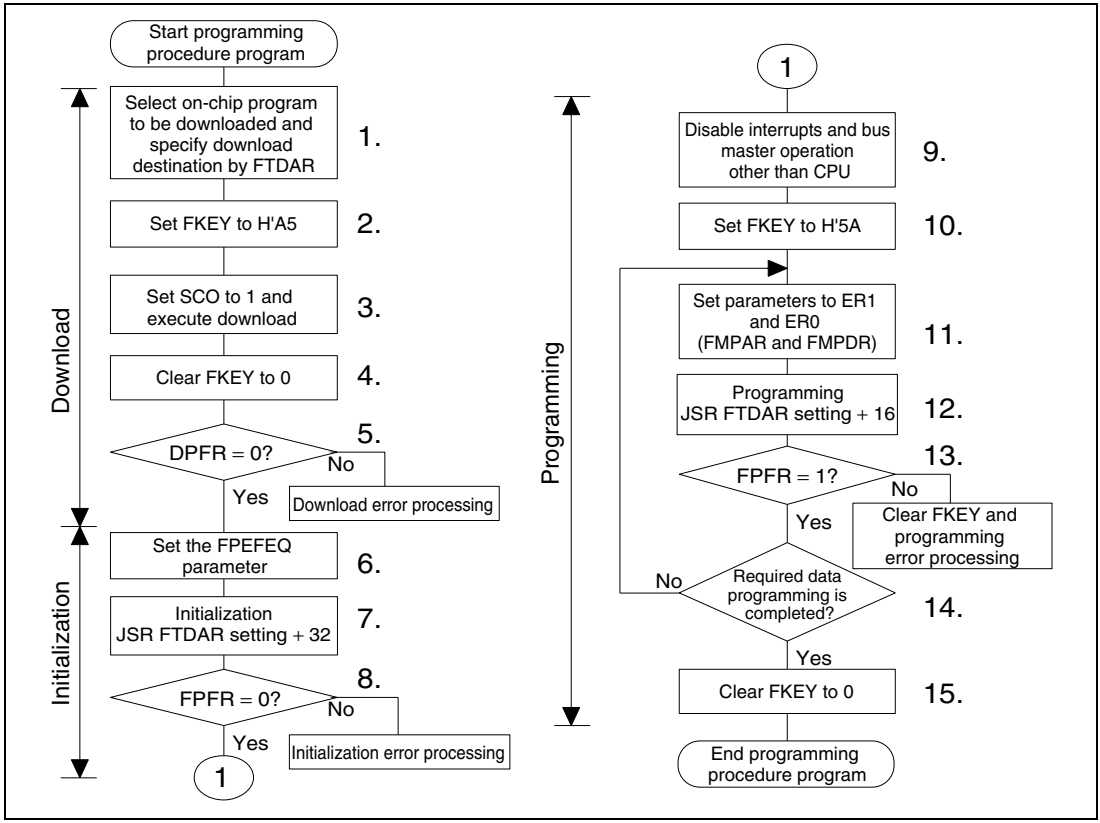


Figure 14.11 Programming Procedure

The procedure program must be executed in an area other than the flash memory to be programmed. Especially the part where the SCO bit in FCCS is set to 1 for downloading must be executed in the on-chip RAM.

The area that can be executed in the steps of the user procedure program (on-chip RAM, user MAT, and external space) is shown in section 14.4.4, Procedure Program and Storable Area for Programming Data.

The following description assumes the area to be programmed on the user MAT is erased and program data is prepared in the consecutive area. When erasing is not executed, erasing is executed before writing.

128-byte programming is performed in one program processing. When more than 128-byte programming is performed, programming destination address/program data parameter is updated in 128-byte units and programming is repeated.

When less than 128-byte programming is performed, data must total 128 bytes by adding the invalid data. If the dummy data to be added is H′FF, the program processing period can be shortened.

1. Select the on-chip program to be downloaded and specify a download destination

   When the PPVS bit of FPCS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are set, download is not performed and a download error is returned to the SS bit in DPFR. The start address of a download destination is specified by FTDAR.

2. Program H′A5 in FKEY

   If H′A5 is not written to FKEY for protection, 1 cannot be set to the SCO bit for download request.

3. 1 is set to the SCO bit of FCCS and then download is executed.

   To set 1 to the SCO bit, the following conditions must be satisfied.

   — H′A5 is written to FKEY.

   — The SCO bit writing is executed in the on-chip RAM.

   When the SCO bit is set to 1, download is started automatically. When the SCO bit is returned to the user procedure program, the SCO is cleared to 0. Therefore, the SCO bit cannot be confirmed to be 1 in the user procedure program.

   The download result can be confirmed only by the return value of DPFR. Before the SCO bit is set to 1, incorrect determination must be prevented by setting the one byte of the start address (to be used as DPFR) specified by FTDAR to a value other than the return value (H′FF).

   When download is executed, particular interrupt processing, which is accompanied by the bank switch as described below, is performed as an internal microcomputer processing. Four NOP instructions are executed immediately after the instructions that set the SCO bit to 1.

   — The user-MAT space is switched to the on-chip program storage area.

   — After the selection condition of the download program and the FTDAR setting are checked, the transfer processing to the on-chip RAM specified by FTDAR is executed.

   — The SCO bit in FCCS is cleared to 0.

   — The return value is set to the DPFR parameter.

   — After the on-chip program storage area is returned to the user-MAT space, the user procedure program is returned.

   — In the download processing, the values of general registers of the CPU are held.

— In the download processing, any interrupts are not accepted. However, interrupt requests are held. Therefore, when the user procedure program is returned, the interrupts occur.

— When the level-detection interrupt requests are to be held, interrupts must be input until the download is ended.

— When hardware standby mode is entered during download processing, the normal download cannot be guaranteed in the on-chip RAM. Therefore, download must be executed again.

— Since a stack area of 128 bytes at the maximum is used, the area must be allocated before setting the SCO bit to 1.

— If a flash memory access by the DMAC signal is requested during downloading, the operation cannot be guaranteed. Therefore, an access request by the DMAC signal must not be generated.

4. FKEY is cleared to H′00 for protection.

5. The value of the DPFR parameter must be checked and the download result must be confirmed.

   — Check the value of the DPFR parameter (one byte of start address of the download destination specified by FTDAR). If the value is H'00, download has been performed normally. If the value is not H'00, the source that caused download to fail can be investigated by the description below.

   — If the value of the DPFR parameter is the same as before downloading (e.g. H'FF), the address setting of the download destination in FTDAR may be abnormal. In this case, confirm the setting of the TDER bit (bit 7) in FTDAR.

   — If the value of the DPFR parameter is different from before downloading, check the SS bit (bit 2) and the FK bit (bit 1) in the DPFR parameter to ensure that the download program selection and FKEY setting were normal, respectively.

6. The operating frequency and user branch destination are set to the FPEFEQ and FUBRA parameters for initialization.

   — The current frequency of the CPU clock is set to the FPEFEQ parameter (general register ER0).

     The settable range of the FPEFEQ parameter is 5 to 33 MHz. When the frequency is set to out of this range, an error is returned to the FPFR parameter of the initialization program and initialization is not performed. For details on the frequency setting, see the description in 14.3.2 (2) (a), Flash programming/erasing frequency parameter (FPEFEQ).

7. Initialization

   When a programming program is downloaded, the initialization program is also downloaded to the on-chip RAM. There is an entry point of the initialization program in the area from the start address specified by FTDAR + 32 bytes of the on-chip RAM. The subroutine is called and initialization is executed by using the following steps.

```
MOV.L   #DLTOP+32,ER2       ; Set entry address to ER2
JSR     @ER2                ; Call initialization routine
NOP
```

— The general registers other than R0L are held in the initialization program.

— R0L is a return value of the FPFR parameter.

— Since the stack area is used in the initialization program, 128-byte stack area at the maximum must be allocated in RAM.

— Interrupts can be accepted during the execution of the initialization program. The program storage area and stack area in the on-chip RAM and register values must not be destroyed.

8. The return value in the initialization program, FPFR (general register R0L) is determined.

9. All interrupts and the use of a bus master (DMAC) other than the CPU are prohibited.
   The specified voltage is applied for the specified time when programming or erasing. If interrupts occur or the bus mastership is moved to other than the CPU during this time, the voltage for more than the specified time will be applied and flash memory may be damaged. Therefore, interrupts and bus mastership to the DMAC are prohibited.

   To prohibit the interrupt, bit 7 (I) in the condition code register (CCR) of the CPU should be set to B′1 in interrupt control mode 0 or bits 2 to 0 (I2 to I0) in the extend control register of the CPU should be set to B′111 in interrupt control mode 2. Then interrupts other than NMI are held and are not executed.

   The NMI interrupts must be masked within the user system.

   The interrupts that are held must be executed after all program processing.

   When the bus mastership is moved to the DMAC, the error protection state is entered. Therefore, taking bus mastership by the DMAC is prohibited.

10. FKEY must be set to H′5A and the user MAT must be prepared for programming.

11. The parameter which is required for programming is set.
    The start address of the programming destination of the user MAT (FMPAR) is set to general register ER1. The start address of the program data area (FMPDR) is set to general register ER0.

    — Example of the FMPAR setting

      FMPAR specifies the programming destination address. When an address other than one in the user MAT area is specified, even if the programming program is executed, programming is not executed and an error is returned to the return value parameter FPFR. Since the unit is 128 bytes, the lower eight bits of the address must be H′00 or H′80 as the boundary of 128 bytes.

— Example of the FMPDR setting

When the storage destination of the program data is flash memory, even if the program execution routine is executed, programming is not executed and an error is returned to the FPFR parameter. In this case, the program data must be transferred to the on-chip RAM and then programming must be executed.

12. Programming

There is an entry point of the programming program in the area from the start address specified by FTDAR + 16 bytes of the on-chip RAM. The subroutine is called and programming is executed by using the following steps.

```
MOV.L   #DLTOP+16,ER2       ; Set entry address to ER2
JSR     @ER2                ; Call programming routine
NOP
```

— The general registers other than R0L are held in the programming program.

— R0L is a return value of the FPFR parameter.

— Since the stack area is used in the programming program, a stack area of 128 bytes at the maximum must be allocated in RAM.

13. The return value in the programming program, FPFR (general register R0L) is determined.

14. Determine whether programming of the necessary data has finished.
If more than 128 bytes of data are to be programmed, specify FMPAR and FMPDR in 128-byte units, and repeat steps 12 to 14. Increment the programming destination address by 128 bytes and update the programming data pointer correctly. If an address which has already been programmed is written to again, not only will a programming error occur, but also flash memory will be damaged.

15. After programming finishes, clear FKEY and specify software protection.
If this LSI is restarted by a reset immediately after user MAT programming has finished, secure the reset period (period of $\overline{RES}$ = 0) of 100 µs which is longer than normal.

(3) Erasing Procedure in User Program Mode

The procedures for download, initialization, and erasing are shown in figure 14.12.

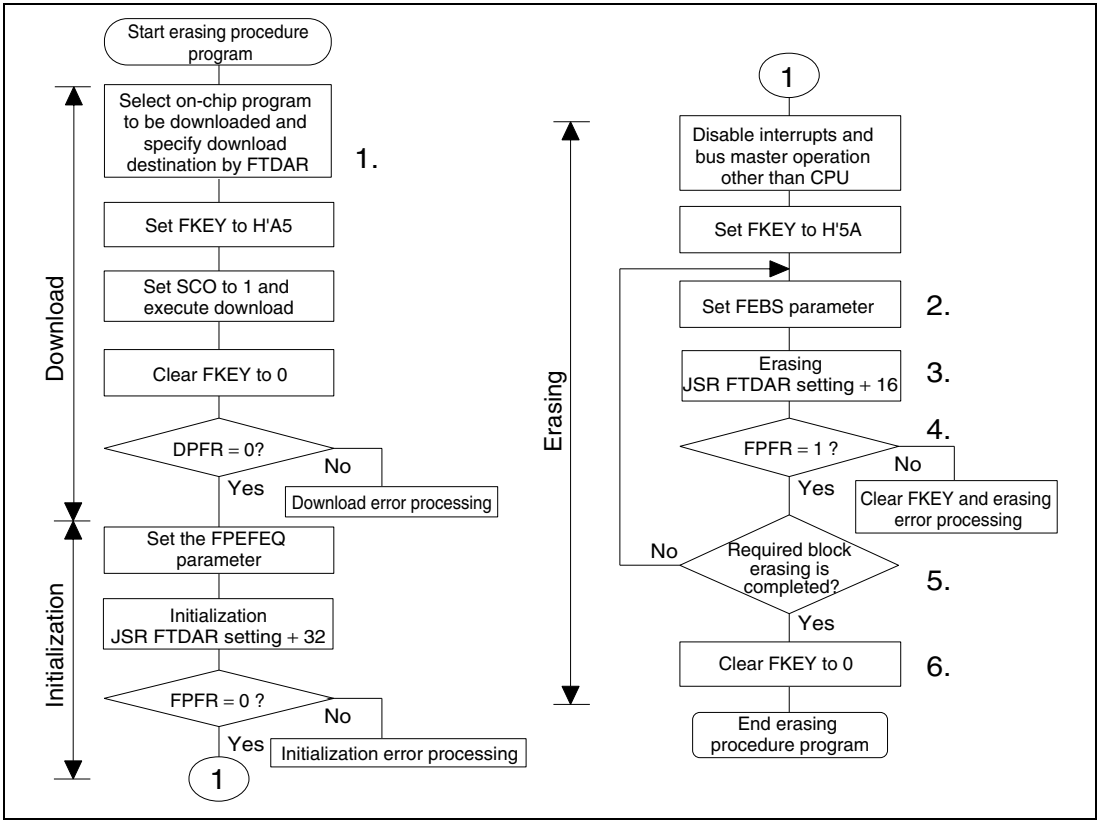


Figure 14.12 Erasing Procedure

The procedure program must be executed in an area other than the user MAT to be erased. Especially the part where the SCO bit in FCCS is set to 1 for downloading must be executed in the on-chip RAM.

The area that can be executed in the steps of the user procedure program (on-chip RAM, user MAT, and external space) is shown in section 14.4.4, Procedure Program and Storable Area for Programming Data.

For the downloaded on-chip program area, refer to the RAM map for programming/erasing in figure 14.10.

A single divided block is erased by one erasing processing. For block divisions, refer to figure 14.4. To erase two or more blocks, update the erase block number and perform the erasing processing for each block.

1. Select the on-chip program to be downloaded
   Set the EPVB bit in FECS to 1.

Several programming/erasing programs cannot be selected at one time. If several programs are set, download is not performed and a download error is reported to the SS bit in the DPFR parameter.

Specify the start address of a download destination by FTDAR.

The procedures to be carried out after setting FKEY, e.g. download and initialization, are the same as those in the programming procedure. For details, refer to Programming Procedure in User Program Mode in section 14.4.2, sub-section (2).

The procedures after setting parameters for erasing programs are as follows:

2. Set the FEBS parameter necessary for erasure

   Set the erase block number of the user MAT in the flash erase block select parameter FEBS (general register ER0). If a value other than an erase block number of the user MAT is set, no block is erased even though the erasing program is executed, and an error is returned to the return value parameter FPFR.

3. Erasure

   Similar to as in programming, there is an entry point of the erasing program in the area from the start address of a download destination specified by FTDAR + 16 bytes of on-chip RAM. The subroutine is called and erasing is executed by using the following steps.

```
MOV.L   #DLTOP+16,ER2       ; Set entry address to ER2
JSR     @ER2                ; Call erasing routine
NOP
```

   - The general registers other than R0L are held in the erasing program.
   - R0L is a return value of the FPFR parameter.
   - Since the stack area is used in the erasing program, a stack area of 128 bytes at the maximum must be allocated in RAM.

4. The return value in the erasing program, FPFR (general register R0L) is determined.

5. Determine whether erasure of the necessary blocks has completed.

   If more than one block is to be erased, update the FEBS parameter and repeat steps 2 to 5. Blocks that have already been erased can be erased again.

6. After erasure completes, clear FKEY and specify software protection.

   If this LSI is restarted by a reset immediately after user MAT erasure has completed, secure the reset period (period of $\overline{RES}$ = 0) of 100 µs which is longer than normal.

(4) Erasing and Programming Procedure in User Program Mode

By changing the on-chip RAM address of the download destination in FTDAR, the erasing program and programming program can be downloaded to separate on-chip RAM areas.

Figure 14.13 shows a repeating procedure of erasing and programming.

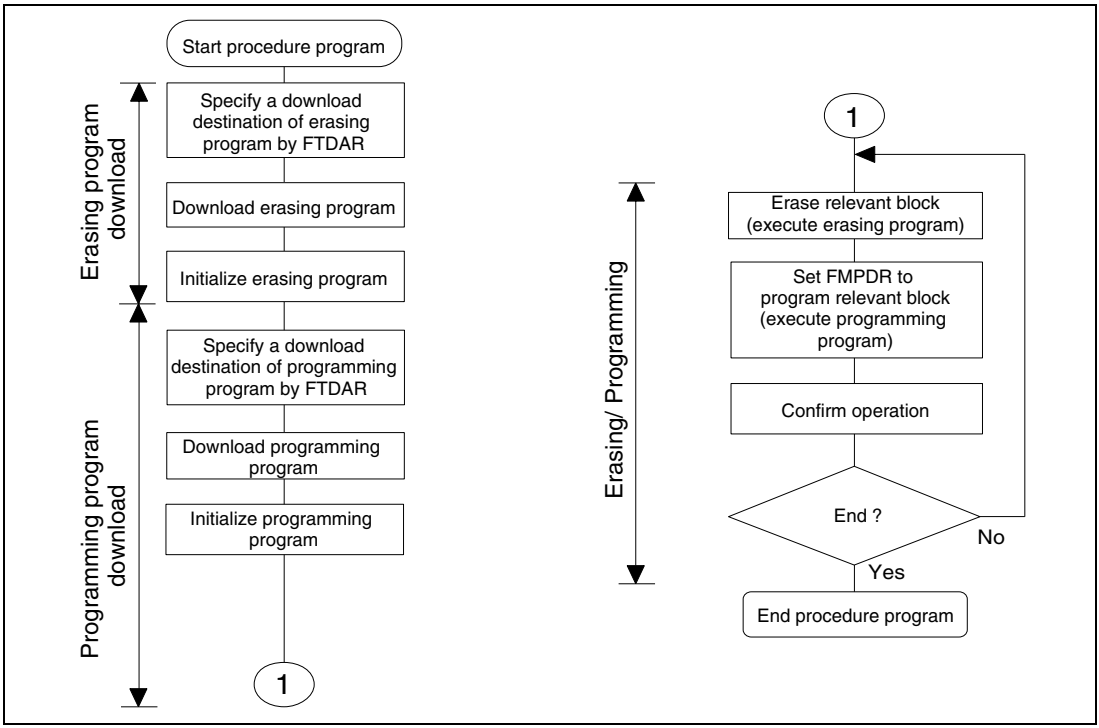


Figure 14.13 Repeating Procedure of Erasing and Programming

In the above procedure, download and initialization are performed only once at the beginning.

In this kind of operation, note the following:

— Be careful not to damage on-chip RAM with overlapped settings.

In addition to the erasing program area and programming program area, areas for the user procedure programs, work area, and stack area are reserved in on-chip RAM. Do not make settings that will overwrite data in these areas.

— Be sure to initialize both the erasing program and programming program.

Initialization by setting the FPEFEQ parameter must be performed for both the erasing program and the programming program. Initialization must be executed for both entry addresses: (download start address for erasing program) + 32 bytes and (download start address for programming program) + 32 bytes.

14.4.3 User Boot Mode

This LSI has user boot mode which is initiated with different mode pin settings than those in boot mode or user program mode. User boot mode is a user-arbitrary boot mode, unlike boot mode that uses the on-chip SCI.

Only the user MAT can be programmed/erased in user boot mode. Programming/erasing of the user boot MAT is only enabled in boot mode or programmer mode.

(1) User Boot Mode Initiation

For the mode pin settings to start up user boot mode, see table 14.6.

When the reset start is executed in user boot mode, the built-in check routine runs. The user MAT and user boot MAT states are checked by this check routine.

While the check routine is running, NMI and all other interrupts cannot be accepted.

Next, processing starts from the execution start address of the reset vector in the user boot MAT. At this point, H′AA is set to FMATS because the execution MAT is the user boot MAT.

(2) User MAT Programming in User Boot Mode

For programming the user MAT in user boot mode, additional processing made by setting FMATS are required: switching from user-boot-MAT selection state to user-MAT selection state, and switching back to user-boot-MAT selection state after programming completes.

Figure 14.14 shows the procedure for programming the user MAT in user boot mode.

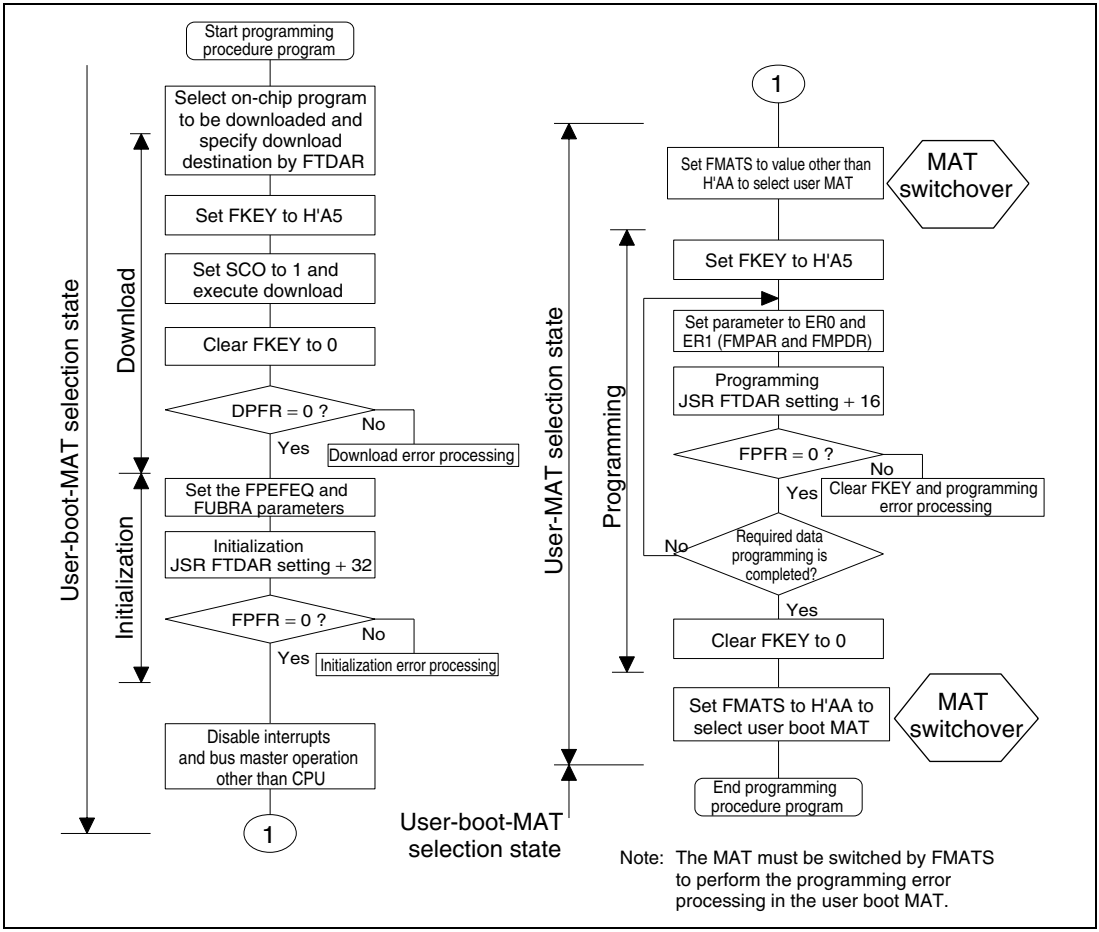


Figure 14.14 Procedure for Programming User MAT in User Boot Mode

The difference between the programming procedures in user program mode and user boot mode is whether the MAT is switched or not as shown in figure 14.14.

In user boot mode, the user boot MAT can be seen in the flash memory space with the user MAT hidden in the background. The user MAT and user boot MAT are switched only while the user MAT is being programmed. Because the user boot MAT is hidden while the user MAT is being programmed, the procedure program must be located in an area other than flash memory. After programming completes, switch the MATs again to return to the first state.

MAT switching is enabled by writing a specific value to FMATS. However note that while the MATs are being switched, the LSI is in an unstable state, e.g. access to a MAT is not allowed until MAT switching is completed, and if an interrupt occurs, from which MAT the interrupt vector is read is undetermined. Perform MAT switching in accordance with the description in section 14.6, Switching between User MAT and User Boot MAT.

Except for MAT switching, the programming procedure is the same as that in user program mode.

The area that can be executed in the steps of the user procedure program (on-chip RAM, user MAT, and external space) is shown in section 14.4.4, Procedure Program and Storable Area for Programming Data.

(3) User MAT Erasing in User Boot Mode

For erasing the user MAT in user boot mode, additional processing made by setting FMATS are required: switching from user-boot-MAT selection state to user-MAT selection state, and switching back to user-boot-MAT selection state after erasing completes.

Figure 14.15 shows the procedure for erasing the user MAT in user boot mode.

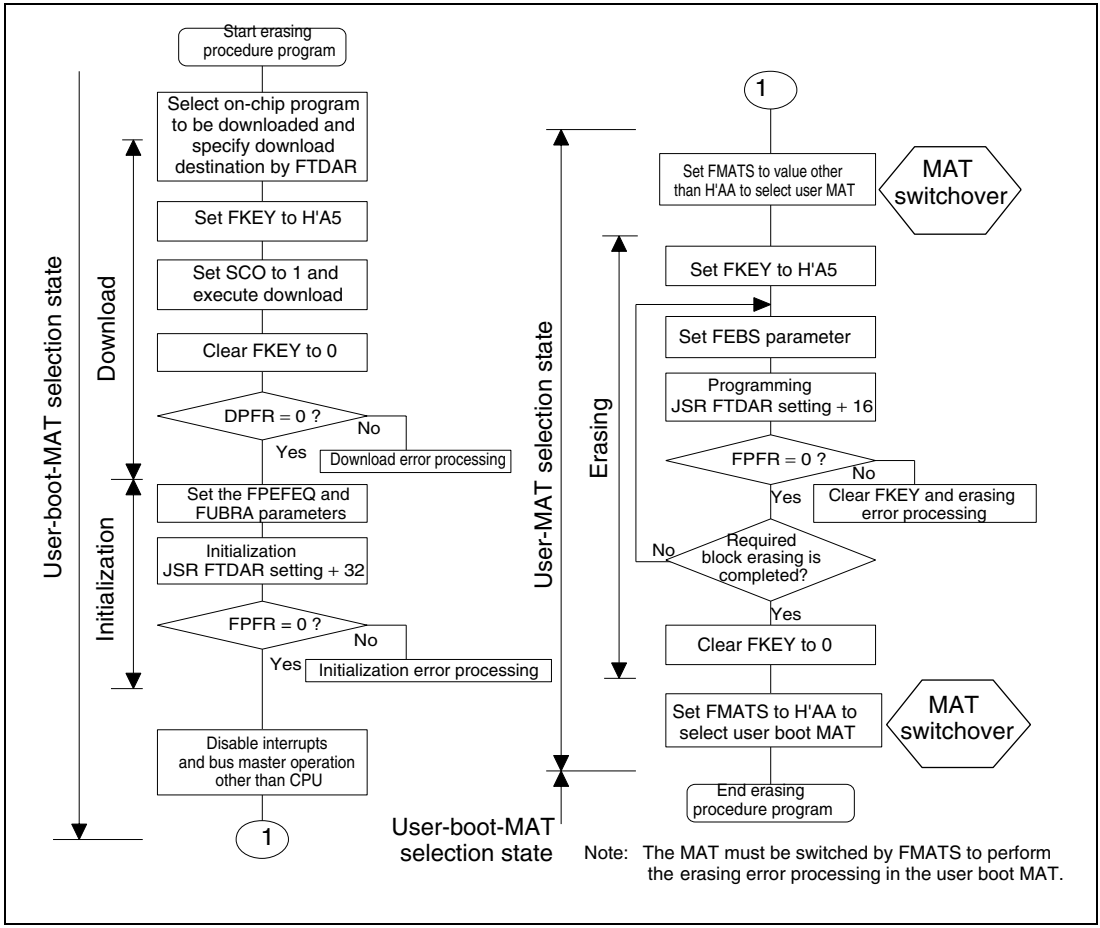


Figure 14.15 Procedure for Erasing User MAT in User Boot Mode

The difference between the erasing procedures in user program mode and user boot mode depends on whether the MAT is switched or not as shown in figure 14.15.

MAT switching is enabled by writing a specific value to FMATS. However note that while the MATs are being switched, the LSI is in an unstable state, e.g. access to a MAT is not allowed until MAT switching is completed, and if an interrupt occurs, from which MAT the interrupt vector is read is undetermined. Perform MAT switching in accordance with the description in section 14.6, Switching between User MAT and User Boot MAT.

Except for MAT switching, the erasing procedure is the same as that in user program mode.

The area that can be executed in the steps of the user procedure program (on-chip RAM, user MAT, and external space) is shown in section 14.4.4, Procedure Program and Storable Area for Programming Data.

14.4.4 Procedure Program and Storable Area for Programming Data

In the descriptions in the previous section, the programming/erasing procedure programs and storable areas for program data are assumed to be in the on-chip RAM. However, the program and the data can be stored in and executed from other areas, such as part of flash memory which is not to be programmed or erased, or somewhere in the external address space.

(1) Conditions that Apply to Programming/Erasing

1. The on-chip programming/erasing program is downloaded from the address in the on-chip RAM specified by FTDAR, therefore, this area is not available for use.
2. The on-chip programming/erasing program will use 128 bytes at the maximum as a stack. So, make sure that this area is secured.
3. Download by setting the SCO bit to 1 will lead to switching of the MAT. If, therefore, this operation is used, it should be executed from the on-chip RAM.
4. The flash memory is accessible until the start of programming or erasing, that is, until the result of downloading has been determined. When in a mode in which the external address space is not accessible, such as single-chip mode, the required procedure programs, NMI handling vector and NMI handler should be transferred to the on-chip RAM before programming/erasing of the flash memory starts.
5. The flash memory is not accessible during programming/erasing operations, therefore, the operation program is downloaded to the on-chip RAM to be executed. The NMI-handling vector and programs such as that which activate the operation program, and NMI handler should thus be stored in on-chip memory other than flash memory or the external address space.
6. After programming/erasing, the flash memory should be inhibited until FKEY is cleared.
   The reset state ($\overline{RES}$ = 0) must be in place for more than 100 µs when the LSI mode is changed to reset on completion of a programming/erasing operation.

Transitions to the reset state, and hardware standby mode are inhibited during programming/erasing. When the reset signal is accidentally input to the chip, a longer period in the reset state than usual (100 µs) is needed before the reset signal is released.

7. Switching of the MATs by FMATS should be needed when programming/erasing of the user boot MAT is operated in user-boot mode. The program which switches the MATs should be executed from the on-chip RAM. See section 14.6, Switching between User MAT and User Boot MAT. Please make sure you know which MAT is selected when switching between them.
8. When the data storable area indicated by programming parameter FMPDR is within the flash memory area, an error will occur even when the data stored is normal. Therefore, the data should be transferred to the on-chip RAM to place the address that FMPDR indicates in an area other than the flash memory.

In consideration of these conditions, there are three factors; operating mode, the bank structure of the user MAT, and operations.

The areas in which the programming data can be stored for execution are shown in tables.

Table 14.8 Executable MAT

	Initiated Mode	
Operation	**User Program Mode**	**User Boot Mode***
Programming	Table 14.9 (1)	Table 14.9 (3)
Erasing	Table 14.9 (2)	Table 14.9 (4)

Note:* Programming/Erasing is possible to user MATs.

Table 14.9 (1) Useable Area for Programming in User Program Mode

	Storable /Executable Area			Selected MAT	
Item	On-chip RAM	User MAT	External Space (Expanded Mode)	User MAT	Embedded Program Storage Area
Storage Area for Program Data	○	×*	○	—	—
Operation for Selection of On-chip Program to be Downloaded	○	○	○	○	
Operation for Writing H′A5 to FKEY	○	○	○	○	
Execution of Writing SC0 = 1 to FCCS (Download)	○	×	×		○
Operation for FKEY Clear	○	○	○	○	
Determination of Download Result	○	○	○	○	
Operation for Download Error	○	○	○	○	
Operation for Settings of Initial Parameter	○	○	○	○	
Execution of Initialization	○	×	×	○	
Determination of Initialization Result	○	○	○	○	
Operation for Initialization Error	○	○	○	○	
NMI Handling Routine	○	×	○	○	
Operation for Inhibit of Interrupt	○	○	○	○	
Operation for Writing H′5A to FKEY	○	○	○	○	
Operation for Settings of Program Parameter	○	×	○	○	

Item	Storable /Executable Area			Selected MAT	
	On-chip RAM	Target Flash Memory	External Space (Expanded Mode)	User MAT	Embedded Program Storage Area
Execution of Programming	○	×	×	○	
Determination of Program Result	○	×	○	○	
Operation for Program Error	○	×	○	○	
Operation for FKEY Clear	○	×	○	○	

Note:* Transferring the data to the on-chip RAM enables this area to be used.

Table 14.9 (2) Useable Area for Erasure in User Program Mode

	Storable /Executable Area			Selected MAT	
Item	On-chip RAM	User MAT	External Space (Expanded Mode)	User MAT	Embedded Program Storage Area
Operation for Selection of On-chip Program to be Downloaded	○	○	○	○	
Operation for Writing H′A5 to FKEY	○	○	○	○	
Execution of Writing SC0 = 1 to FCCS (Download)	○	×	×		○
Operation for FKEY Clear	○	○	○	○	
Determination of Download Result	○	○	○	○	
Operation for Download Error	○	○	○	○	
Operation for Settings of Initial Parameter	○	○	○	○	
Execution of Initialization	○	×	×	○	
Determination of Initialization Result	○	○	○	○	
Operation for Initialization Error	○	○	○	○	
NMI Handling Routine	○	×	○	○	
Operation for Inhibit of Interrupt	○	○	○	○	
Operation for Writing H′5A to FKEY	○	○	○	○	
Operation for Settings of Erasure Parameter	○	×	○	○	
Execution of Erasure	○	×	×	○	
Determination of Erasure Result	○	×	○	○	

Item	Storable /Executable Area			Selected MAT	
	On-chip RAM	User MAT	External Space (Expanded Mode)	User MAT	Embedded Program Storage Area
Operation for Erasure Error	○	×	○	○	
Operation for FKEY Clear	○	×	○	○	

Table 14.9 (3) Useable Area for Programming in User Boot Mode

	Storable/Executable Area			Selected MAT		
Item	On-chip RAM	User Boot MAT	External Space (Expanded Mode)	User MAT	User Boot MAT	Embedded Program Storage Area
Storage Area for Program Data	○	×*[1]	○	—	—	—
Operation for Selection of On-chip Program to be Downloaded	○	○	○		○	
Operation for Writing H′A5 to FKEY	○	○	○		○	
Execution of Writing SC0 = 1 to FCCS (Download)	○	×	×			○
Operation for FKEY Clear	○	○	○		○	
Determination of Download Result	○	○	○		○	
Operation for Download Error	○	○	○		○	
Operation for Settings of Initial Parameter	○	○	○		○	
Execution of Initialization	○	×	×		○	
Determination of Initialization Result	○	○	○		○	
Operation for Initialization Error	○	○	○		○	
NMI Handling Routine	○	×	○		○	
Operation for Interrupt Inhibit	○	○	○		○	
Switching MATs by FMATS	○	×	×	○		
Operation for Writing H′5A to FKEY	○	×	○	○		

	Storable/Executable Area			Selected MAT		
Item	On-chip RAM	User Boot MAT	External Space (Expanded Mode)	User MAT	User Boot MAT	Embedded Program Storage Area
Operation for Settings of Program Parameter	○	×	○	○		
Execution of Programming	○	×	×	○		
Determination of Program Result	○	×	○	○		
Operation for Program Error	○	×*2	○	○		
Operation for FKEY Clear	○	×	○	○		
Switching MATs by FMATS	○	×	×		○	

Notes: 1. Transferring the data to the on-chip RAM enables this area to be used.
2. Switching FMATS by a program in the on-chip RAM enables this area to be used.

Table 14.9 (4) Useable Area for Erasure in User Boot Mode

	Storable/Executable Area			Selected MAT		
Item	On-chip RAM	User Boot MAT	External Space (Expanded Mode)	User MAT	User Boot MAT	Embedded Program Storage Area
Operation for Selection of On-chip Program to be Downloaded	○	○	○		○	
Operation for Writing H′A5 to FKEY	○	○	○		○	
Execution of Writing SC0 = 1 to FCCS (Download)	○	×	×			○
Operation for FKEY Clear	○	○	○		○	
Determination of Download Result	○	○	○		○	
Operation for Download Error	○	○	○		○	
Operation for Settings of Initial Parameter	○	○	○		○	
Execution of Initialization	○	×	×		○	
Determination of Initialization Result	○	○	○		○	
Operation for Initialization Error	○	○	○		○	
NMI Handling Routine	○	×	○		○	
Operation for Interrupt Inhibit	○	○	○		○	
Switching MATs by FMATS	○	×	×		○	
Operation for Writing H′5A to FKEY	○	×	○	○		
Operation for Settings of Erasure Parameter	○	×	○	○		

Item	Storable/Executable Area			Selected MAT		
	On-chip RAM	User Boot MAT	External Space (Expanded Mode)	User MAT	User Boot MAT	Embedded Program Storage Area
Execution of Erasure	○	×	×	○		
Determination of Erasure Result	○	×	○	○		
Operation for Erasure Error	○	×*	○	○		
Operation for FKEY Clear	○	×	○	○		
Switching MATs by FMATS	○	×	×	○		

Note:* Switching FMATS by a program in the on-chip RAM enables this area to be used.

14.5 Protection

There are two kinds of flash memory program/erase protection: hardware and software protection.

14.5.1 Hardware Protection

Programming and erasing of flash memory is forcibly disabled or suspended by hardware protection. In this state, the downloading of an on-chip program and initialization are possible. However, an activated program for programming or erasure cannot program or erase locations in a user MAT, and the error in programming/erasing is reported in the parameter FPFR.

Table 14.10 Hardware Protection

		Function to be Protected	
Item	**Description**	**Download**	**Program/Erase**
FWE pin protection	• When a low level signal is input to the FWE pin, the FWE bit in FCCS is cleared and the program/erase-protected state is entered.	—	○
Reset/standby protection	• The program/erase interface registers are initialized in the reset state (including a reset by the WDT) and standby mode and the program/erase-protected state is entered. • The reset state will not be entered by a reset using the $\overline{\text{RES}}$ pin unless the RES pin is held low until oscillation has stabilized after power is initially supplied. In the case of a reset during operation, hold the RES pin low for the RES pulse width that is specified in the section on AC characteristics. If a reset is input during programming or erasure, data values in the flash memory are not guaranteed. In this case, execute erasure and then execute program again.	○	○

14.5.2 Software Protection

Software protection is set up in any of two ways: by disabling the downloading of on-chip programs for programming and erasing and by means of a key code.

Table 14.11 Software Protection

Item	Description	Function to be Protected	
		Download	Program/Erase
Protection by the SCO bit	• The program/erase-protected state is entered by clearing the SCO bit in FCCS which disables the down-loading of the programming/erasing programs.	○	○
Protection by the FKEY register	• Downloading and programming/erasing are disabled unless the required key code is written in FKEY. Different key codes are used for downloading and for programming/erasing.	○	○

14.5.3 Error Protection

Error protection is a mechanism for aborting programming or erasure when an error occurs, in the form of the microcomputer entering runaway during programming/erasing of the flash memory or operations that are not according to the established procedures for programming/erasing. Aborting programming or erasure in such cases prevents damage to the flash memory due to excessive programming or erasing.

If the microcomputer malfunctions during programming/erasing of the flash memory, the FLER bit in the FCCS register is set to 1 and the error-protection state is entered, and this aborts the programming or erasure.

The FLER bit is set in the following conditions:

1. When an interrupt such as NMI occurs during programming/erasing.
2. When the flash memory is read during programming/erasing (including a vector read or an instruction fetch).
3. When a SLEEP instruction (including software-standby mode) is executed during programming/erasing.
4. When a bus master other than the CPU, such as the DMAC, gets bus mastership during programming/erasing.

Error protection is cancelled only by a reset or by hardware-standby mode. Note that the reset should be released after the reset period of 100 µs which is longer than normal. Since high voltages are applied during programming/erasing of the flash memory, some voltage may remain after the error-protection state has been entered. For this reason, it is necessary to reduce the risk of damage to the flash memory by extending the reset period so that the charge is released.

The state-transition diagram in figure 14.16 shows transitions to and from the error-protection state.

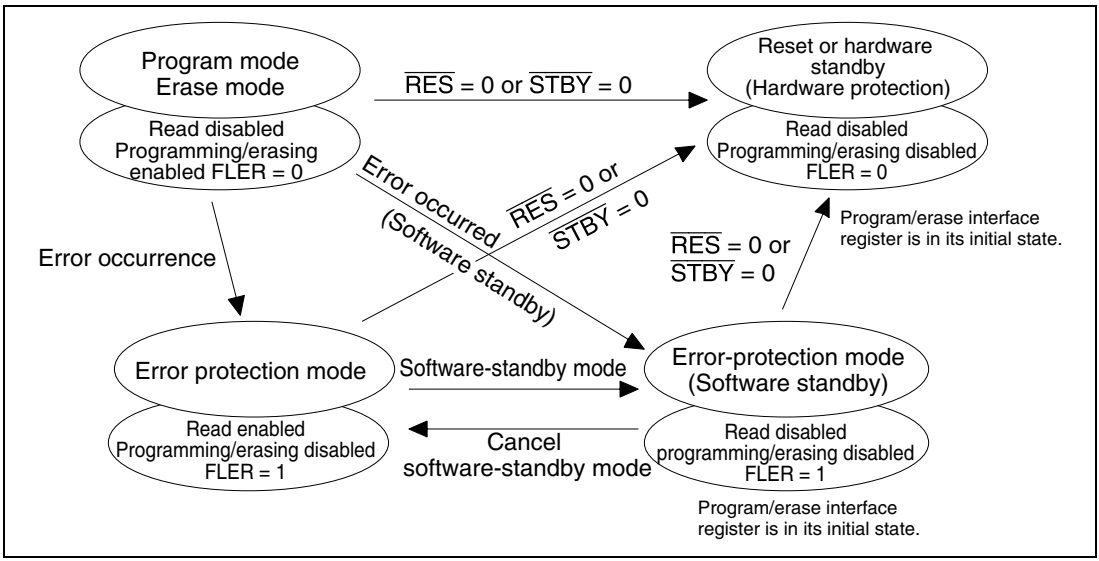


Figure 14.16 Transitions to Error-Protection State

14.6 Switching between User MAT and User Boot MAT

It is possible to alternate between the user MAT and user boot MAT. However, the following procedure is required because these MATs are allocated to address 0.
(Switching to the user boot MAT disables programming and erasing. Programming of the user boot MAT should take place in boot mode or programmer mode.)

1. MAT switching by FMATS should always be executed from the on-chip RAM.
2. To ensure that the MAT that has been switched to is accessible, execute four NOP instructions in the on-chip RAM immediately after writing to FMATS of the on-chip RAM (this prevents access to the flash memory during MAT switching).
3. If an interrupt has occurred during switching, there is no guarantee of which memory MAT is being accessed. Always mask the maskable interrupts before switching between MATs. In addition, configure the system so that NMI interrupts do not occur during MAT switching.

4. After the MATs have been switched, take care because the interrupt vector table will also have been switched. If interrupt processing is to be the same before and after MAT switching, transfer the interrupt-processing routines to the on-chip RAM and set the WEINTE bit in FCCS to place the interrupt-vector table in the on-chip RAM.
5. Memory sizes of the user MAT and user boot MAT are different. When accessing the user boot MAT, do not access addresses above the top of its 8-kbyte memory space. If access goes beyond the 8-kbyte space, the values read are undefined.

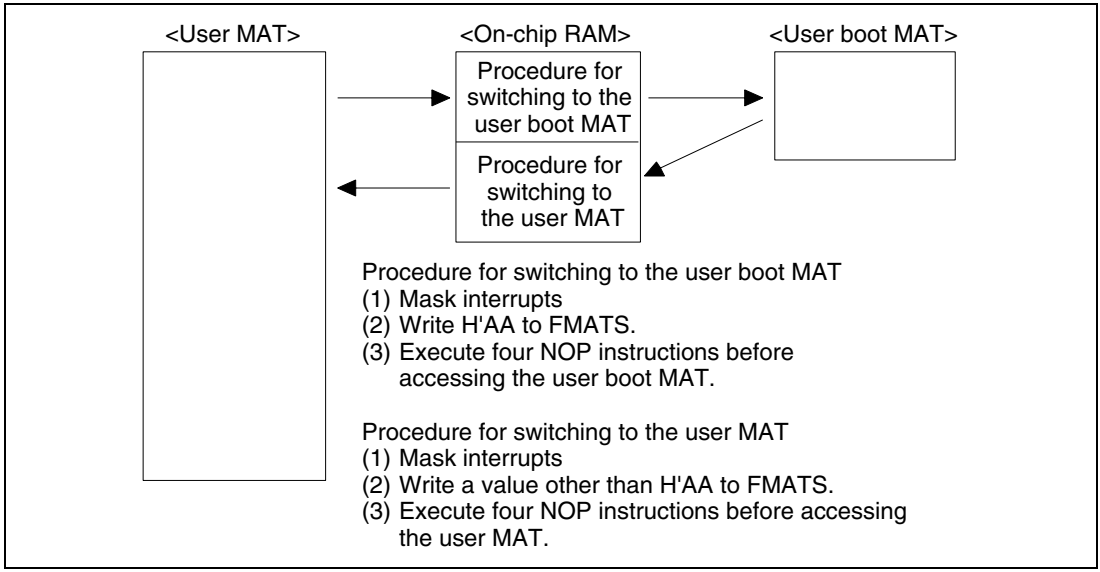


Figure 14.17 Switching between the User MAT and User Boot MAT

14.7 Flash Memory Emulation in RAM

14.7.1 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 realtime. After the RAMER setting has been made, accesses can 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 14.18 shows an example of emulation of realtime flash memory.

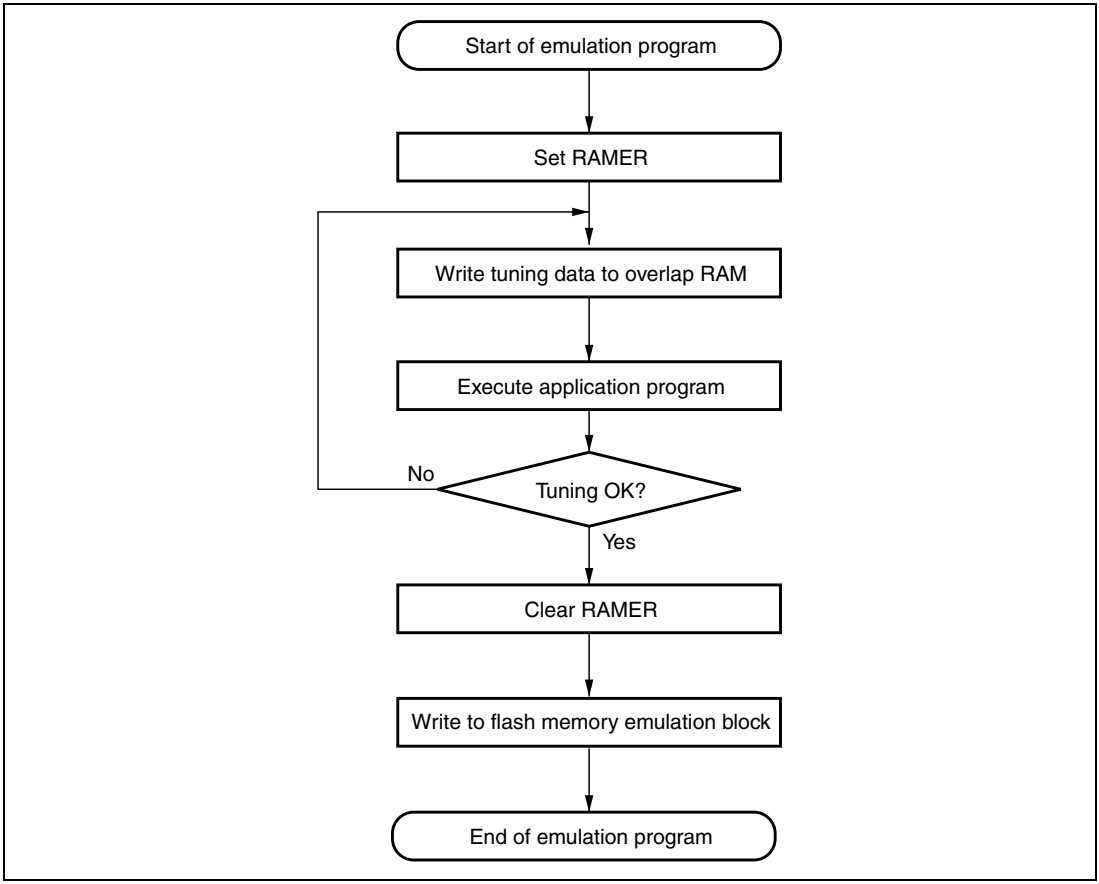


Figure 14.18 Flowchart for Flash Memory Emulation in RAM

14.7.2 RAM Overlap

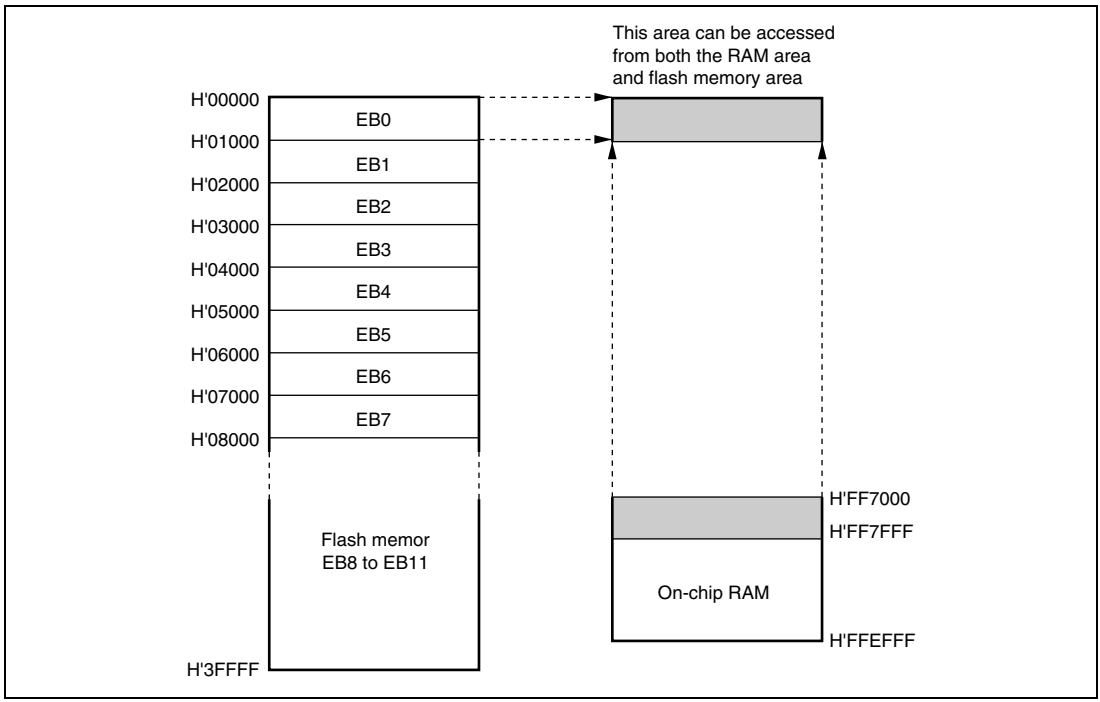


Figure 14.19 Example of RAM Overlap Operation (256-kbyte Flash Memory)

As the flash memory area to be emulated, bits RAM2 to RAM0 select one area among eight areas, EB0 to EB7, in bank 1 of user MAT.

Figure 14.19 shows an example in which flash memory block area, EB1, is overlapped.

1. Set bits RAMS, RAM2, RAM1, and RAM0 in RAMER to 1, 0, 0, 1, to overlap part of RAM onto the area (EB1) for which realtime programming is required.
2. Realtime 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 (EB1).

Notes: 1. When the RAMS bit is set to 1, program/erase-protection is enables for all flash memory blocks regardless of the value of RAM2, RAM1, and RAM0 (emulation protection). When actually programming a flash memory area, the RAMS bit should be cleared to 0.

2. The RAM area cannot be erased by execution of software in accordance with the erase algorithm while flash memory emulation in RAM is being used.

3. Block area EB0 includes the vector table. When performing RAM emulation, the vector table is needed by the overlap RAM.

14.8 Programmer Mode

Along with its on-board programming mode, this LSI also has a programmer mode as a further mode for the writing and erasing of programs and data. In the programmer mode, a general-purpose PROM programmer can freely be used to write programs to the on-chip ROM. Program/erase is possible on the user MAT and user boot MAT. The PROM programmer must support Hitachi microcomputers with 258-kbyte flash memory as a device type*. Figure 14.20 shows a memory map in programmer mode.

A status-polling system is adopted for operation in automatic program, automatic erase, and status-read modes. In the status-read mode, details of the system's internal signals are output after execution of automatic programming or automatic erasure. In programmer mode, provide a 12-MHz input-clock signal.

Note:* In this LSI, set the programming voltage of the PROM programmer to 3.3 V.

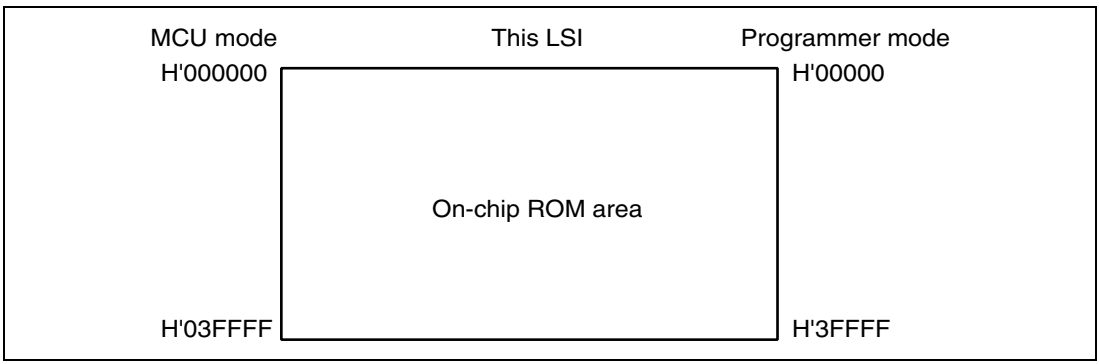


Figure 14.20 Memory Map in Programmer Mode

14.9 Serial Communication Interface Specification for Boot Mode

Initiating boot mode enables the boot program to communicate with the host by using the internal SCI. The serial communication interface specification is shown below.

(1) Status

The boot program has three states.

1. Bit-Rate-Adjustment State

   In this state, the boot program adjusts the bit rate to communicate with the host. Initiating boot mode enables starting of the boot program and entry to the bit-rate-adjustment state. The program receives the command from the host to adjust the bit rate. After adjusting the bit rate, the program enters the inquiry/selection state.
2. Inquiry/Selection State

   In this state, the boot program responds to inquiry commands from the host. The device name, clock mode, and bit rate are selected. After selection of these settings, the program is made to enter the programming/erasing state by the command for a transition to the programming/erasing state. The program transfers the libraries required for erasure to the on-chip RAM and erases the user MATs and user boot MATs before the transition.
3. Programming/erasing state

   Programming and erasure by the boot program take place in this state. The boot program is made to transfer the programming/erasing programs to the RAM by commands from the host. Sum checks and blank checks are executed by sending these commands from the host.

These boot program states are shown in figure 14.21.

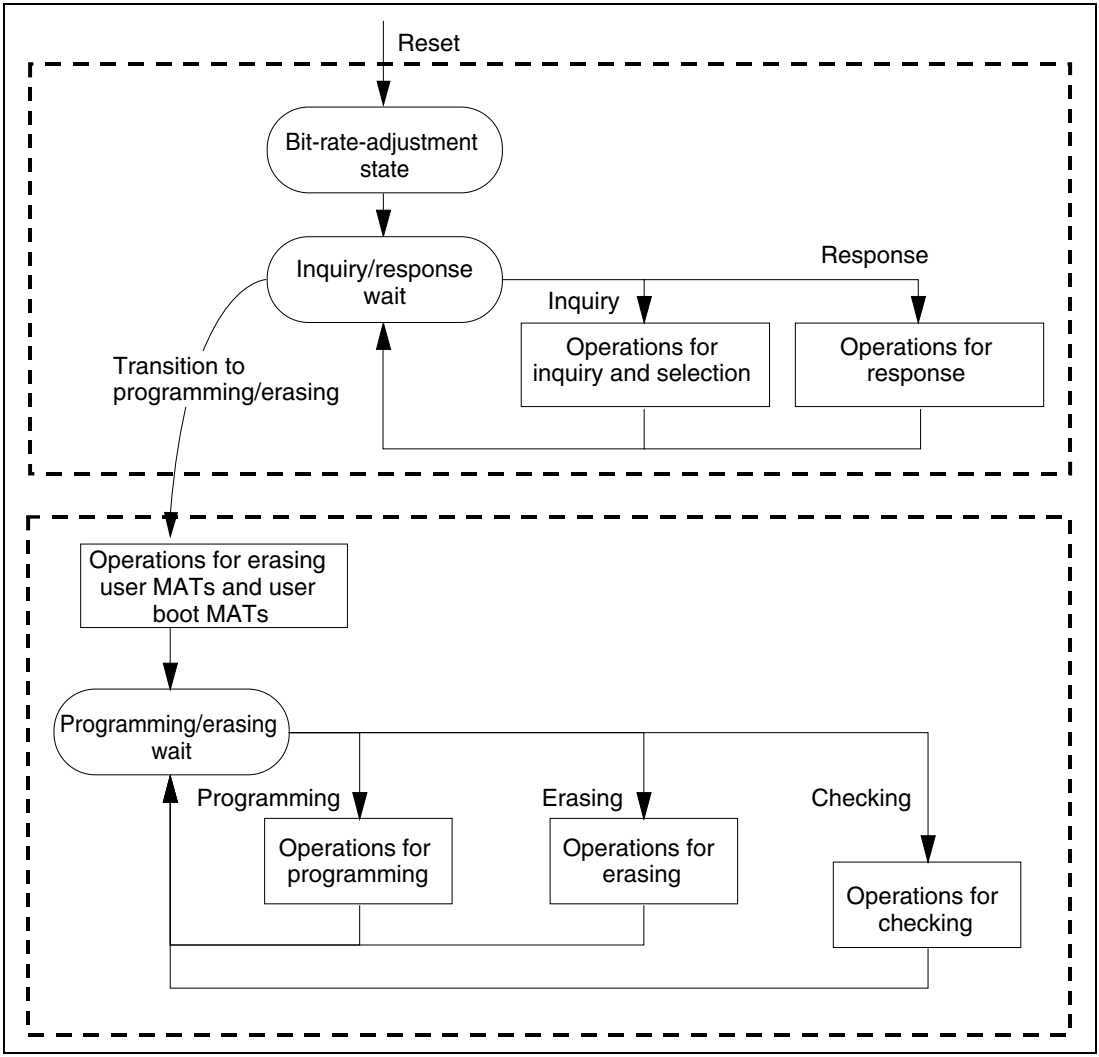


Figure 14.21 Boot Program States

(2) Bit-Rate-Adjustment State

The bit rate is calculated by measuring the period of transfer of a low-level byte (H′00) from the host. The bit rate can be changed by the command for a new bit rate selection. After the bit rate has been adjusted, the boot program enters the inquiry and selection state. The bit-rate-adjustment sequence is shown in figure 14.22.

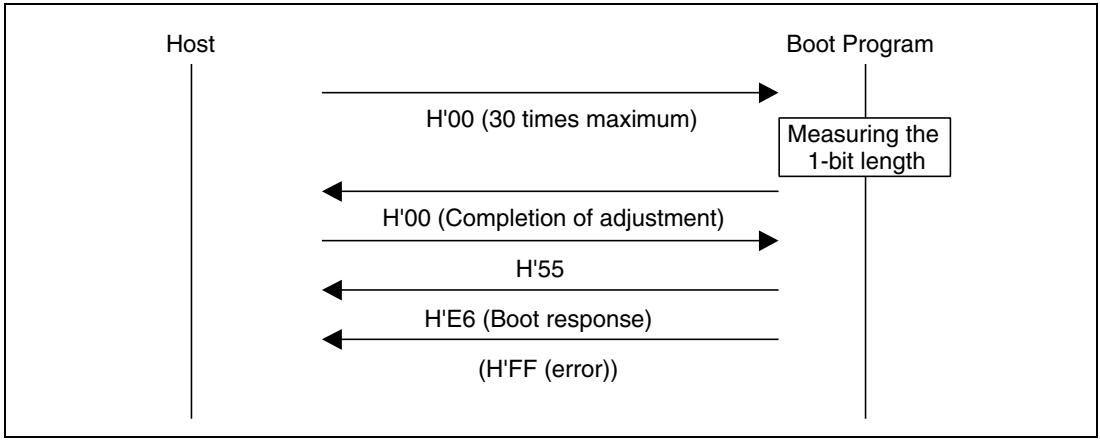


Figure 14.22 Bit-Rate-Adjustment Sequence

(3) Communications Protocol

After adjustment of the bit rate, the protocol for communications between the host and the boot program is as shown below.

1. One-byte commands and one-byte responses

   These commands and responses are comprised of a single byte. These are consists of the inquiries and the ACK for successful completion.
2. n-byte commands or n-byte responses

   These commands and responses are comprised of n bytes of data. These are selections and responses to inquiries.

   The amount of programming data is not included under this heading because it is determined in another command.
3. Error response

   The error response is a response to inquiries. It consists of an error response and an error code and comes two bytes.
4. Programming of 128 bytes

   The size is not specified in commands. The size of n is indicated in response to the programming unit inquiry.
5. Memory read response

   This response consists of four bytes of data.

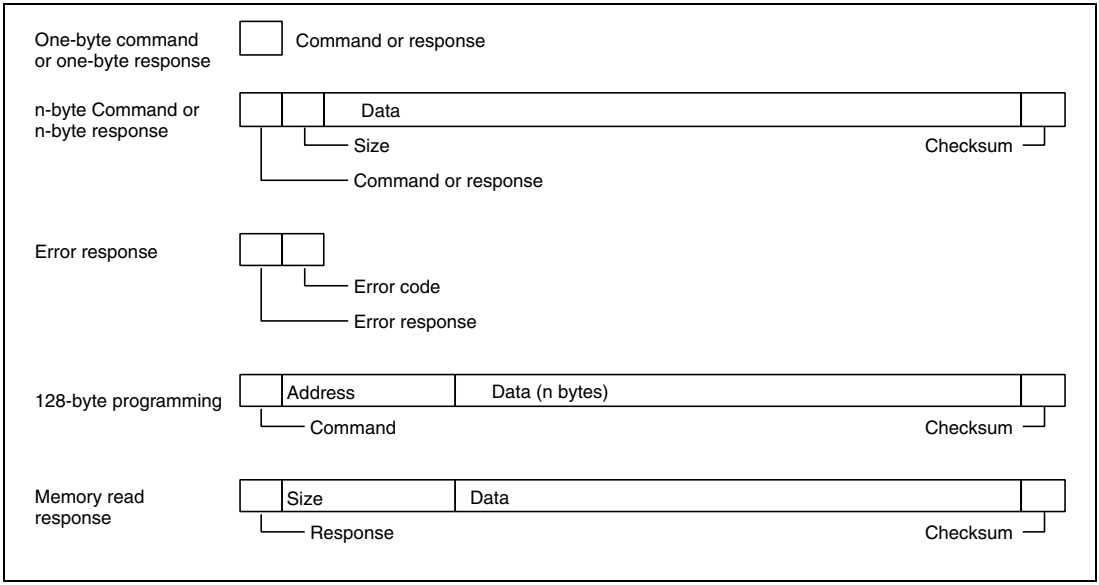


Figure 14.23 Communication Protocol Format

- Command (one byte): Commands including inquiries, selection, programming, erasing, and checking
- Response (one byte): Response to an inquiry
- Size (one byte): The amount of data for transmission excluding the command, amount of data, and checksum
- Checksum (one byte): The checksum is calculated so that the total of all values from the command byte to the SUM byte becomes H′00.
- Data (n bytes): Detailed data of a command or response
- Error response (one byte): Error response to a command
- Error code (one byte): Type of the error
- Address (four bytes): Address for programming
- Data (n bytes): Data to be programmed (the size is indicated in the response to the programming unit inquiry.)
- Size (four bytes): Four-byte response to a memory read

(4) Inquiry and Selection States

The boot program returns information from the flash memory in response to the host's inquiry commands and sets the device code, clock mode, and bit rate in response to the host's selection command.

Inquiry and selection commands are listed below.

Table 14.12 Inquiry and Selection Commands

Command	Command Name	Description
H′20	Supported Device Inquiry	Inquiry regarding device codes
H′10	Device Selection	Selection of device code
H′21	Clock Mode Inquiry	Inquiry regarding numbers of clock modes and values of each mode
H′11	Clock Mode Selection	Indication of the selected clock mode
H′22	Multiplication Ratio Inquiry	Inquiry regarding the number of frequency-multiplied clock types, the number of multiplication ratios, and the values of each multiple
H′23	Operating Clock Frequency Inquiry	Inquiry regarding the maximum and minimum values of the main clock and peripheral clocks
H′24	User Boot MAT Information Inquiry	Inquiry regarding the number of user boot MATs and the start and last addresses of each MAT
H′25	User MAT Information Inquiry	Inquiry regarding the a number of user MATs and the start and last addresses of each MAT
H′26	Block for Erasing Information Inquiry	Inquiry regarding the number of blocks and the start and last addresses of each block
H′27	Programming Unit Inquiry	Inquiry regarding the unit of programming data
H′3F	New Bit Rate Selection	Selection of new bit rate
H′40	Transition to Programming/Erasing State	Erasing of user MAT and user boot MAT, and entry to programming/erasing state
H′4F	Boot Program Status Inquiry	Inquiry into the operated status of the boot program

The selection commands, which are device selection (H′10), clock mode selection (H′11), and new bit rate selection (H′3F), should be sent from the host in that order. These commands will certainly be needed. When two or more selection commands are sent at once, the last command will be valid.

All of these commands, except for the boot program status inquiry command (H′4F), will be valid until the boot program receives the programming/erasing transition (H′40). The host can choose the needed commands out of the commands and inquiries listed above. The boot program status inquiry command (H′4F) is valid after the boot program has received the programming/erasing transition command (H′40).

(a) Supported Device Inquiry

The boot program will return the device codes of supported devices and the product code in response to the supported device inquiry.

Command	H′20

- Command, H′20, (one byte): Inquiry regarding supported devices

Response	H′30	Size	Number of devices	
	Number of characters	Device code		Product name
	...			
	SUM			

- Response, H′30, (one byte): Response to the supported device inquiry
- Size (one byte): Number of bytes to be transmitted, excluding the command, size, and checksum, that is, the amount of data contributes by the number of devices, characters, device codes and product names
- Number of devices (one byte): The number of device types supported by the boot program
- Number of characters (one byte): The number of characters in the device codes and boot program's name
- Device code (four bytes): ASCII code of the supporting product
- Product name (n bytes): Type name of the boot program in ASCII-coded characters
- SUM (one byte): Checksum

  The checksum is calculated so that the total number of all values from the command byte to the SUM byte becomes H′00.

(b) Device Selection

The boot program will set the supported device to the specified device code. The program will return the selected device code in response to the inquiry after this setting has been made.

Command	H′10	Size	Device code	SUM

- Command, H′10, (one byte): Device selection
- Size (one byte): Amount of device-code data
  This is fixed at 2

- Device code (four bytes): Device code (ASCII code) returned in response to the supported device inquiry
- SUM (one byte): Checksum

Response	H′06

- Response, H′06, (one byte): Response to the device selection command
  ACK will be returned when the device code matches.

Error response	H′90	ERROR

- Error response, H′90, (one byte): Error response to the device selection command
  ERROR : (one byte): Error code
  H′11: Sum check error
  H′21: Device code error, that is, the device code does not match

(c) Clock Mode Inquiry

The boot program will return the supported clock modes in response to the clock mode inquiry.

Command	H′21

- Command, H′21, (one byte): Inquiry regarding clock mode

Response	H′31	Size	Number of modes	Mode	···	SUM

- Response, H′31, (one byte): Response to the clock-mode inquiry
- Size (one byte): Amount of data that represents the number of modes and modes
- Number of clock modes (one byte): The number of supported clock modes
  H′00 indicates no clock mode or the device allows to read the clock mode.
- Mode (one byte): Values of the supported clock modes (i.e. H′01 means clock mode 1.)
- SUM (one byte): Checksum

(d) Clock Mode Selection

The boot program will set the specified clock mode. The program will return the selected clock-mode information after this setting has been made.

The clock-mode selection command should be sent after the device-selection commands.

Command	H′11	Size	Mode	SUM

- Command, H′11, (one byte): Selection of clock mode
- Size (one byte): Amount of data that represents the modes
- Mode (one byte): A clock mode returned in reply to the supported clock mode inquiry.
- SUM (one byte): Checksum

Response	H′06

- Response, H′06, (one byte): Response to the clock mode selection command
  ACK will be returned when the clock mode matches.

Error Response	H′91	ERROR

- Error response, H′91, (one byte) : Error response to the clock mode selection command
- ERROR : (one byte): Error code

  H′11: Checksum error
  H′22: Clock mode error, that is, the clock mode does not match.

Even if the clock mode numbers are H′00 and H′01 by a clock mode inquiry, the clock mode must be selected using these respective values.

(e) Multiplication Ratio Inquiry

The boot program will return the supported multiplication and division ratios.

Command	H′22

- Command, H′22, (one byte): Inquiry regarding multiplication ratio

Response	H′32	Size	Number of types					
	Number of multiplication ratios	Multiplica-tion ratio	···					
	...							
	SUM							

- Response, H′32, (one byte): Response to the multiplication ratio inquiry
- Size (one byte): The amount of data that represents the number of clock sources and multiplication ratios and the multiplication ratios
- Number of types (one byte): The number of supported multiplied clock types
  (e.g. when there are two multiplied clock types, which are the main and peripheral clocks, the number of types will be H′02.)
- Number of multiplication ratios (one byte): The number of multiplication ratios for each type
  (e.g. the number of multiplication ratios to which the main clock can be set and the peripheral clock can be set.)
- Multiplication ratio (one byte)

  Multiplication ratio: The value of the multiplication ratio (e.g. when the clock-frequency multiplier is four, the value of multiplication ratio will be H′04.)

  Division ratio: The inverse of the division ratio, i.e. a negative number (e.g. when the clock is divided by two, the value of division ratio will be H′FE. H′FE = D′-2)

The number of multiplication ratios returned is the same as the number of multiplication ratios and as many groups of data are returned as there are types.

- SUM (one byte): Checksum

(f) Operating Clock Frequency Inquiry

The boot program will return the number of operating clock frequencies, and the maximum and minimum values.

Command

H′23

- Command, H′23, (one byte): Inquiry regarding operating clock frequencies

Response

H′33	Size	Number of operating clock frequencies
Minimum value of operating clock frequency		Maximum value of operating clock frequency
...		
SUM		

- Response, H′33, (one byte): Response to operating clock frequency inquiry
- Size (one byte): The number of bytes that represents the minimum values, maximum values, and the number of frequencies.
- Number of operating clock frequencies (one byte): The number of supported operating clock frequency types
  (e.g. when there are two operating clock frequency types, which are the main and peripheral clocks, the number of types will be H′02.)
- Minimum value of operating clock frequency (two bytes): The minimum value of the multiplied or divided clock frequency.

  The minimum and maximum values represent the values in MHz, valid to the hundredths place of MHz, and multiplied by 100. (e.g. when the value is 20.00 MHz, it will be 2000, which is H′07D0.)
- Maximum value (two bytes): Maximum value among the multiplied or divided clock frequencies.

  There are as many pairs of minimum and maximum values as there are operating clock frequencies.
- SUM (one byte): Checksum

(g) User Boot MAT Information Inquiry

The boot program will return the number of user boot MATs and their addresses.

Command	H′24

- Command, H′24, (one byte): Inquiry regarding user boot MAT information

Response	H′34	Size	Number of areas	
	Area-start address			Area-last address
	...			
	SUM			

- Response, H′34, (one byte): Response to user boot MAT information inquiry
- Size (one byte): The number of bytes that represents the number of areas, area-start addresses, and area-last address
- Number of Areas (one byte): The number of consecutive user boot MAT areas
  When user boot MAT areas are consecutive, the number of areas returned is H′01.
- Area-start address (four byte): Start address of the area
- Area-last address (four byte): Last address of the area
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(h) User MAT Information Inquiry

The boot program will return the number of user MATs and their addresses.

Command	H′25

- Command, H′25, (one byte): Inquiry regarding user MAT information

Response	H′35	Size	Number of areas	
	Start address area			Last address area
	...			
	SUM			

- Response, H′35, (one byte): Response to the user MAT information inquiry
- Size (one byte): The number of bytes that represents the number of areas, area-start address and area-last address
- Number of areas (one byte): The number of consecutive user MAT areas
  When the user MAT areas are consecutive, the number of areas is H′01.
- Area-start address (four bytes): Start address of the area
- Area-last address (four bytes): Last address of the area
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(i) Erased Block Information Inquiry

The boot program will return the number of erased blocks and their addresses.

Command	H′26

- Command, H′26, (two bytes): Inquiry regarding erased block information

Response	H′36	Size	Number of blocks	
	Block start address			Block last address
	...			
	SUM			

- Response, H′36, (one byte): Response to the number of erased blocks and addresses
- Size (three bytes): The number of bytes that represents the number of blocks, block-start addresses, and block-last addresses.
- Number of blocks (one byte): The number of erased blocks
- Block start address (four bytes): Start address of a block
- Block last Address (four bytes): Last address of a block
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(j) Programming Unit Inquiry

The boot program will return the programming unit used to program data.

Command	H′27

- Command, H′27, (one byte): Inquiry regarding programming unit

Response	H′37	Size	Programming unit	SUM

- Response, H′37, (one byte): Response to programming unit inquiry
- Size (one byte): The number of bytes that indicate the programming unit, which is fixed to 2
- Programming unit (two bytes): A unit for programming
  This is the unit for reception of programming.
- SUM (one byte): Checksum

(k) New Bit-Rate Selection

The boot program will set a new bit rate and return the new bit rate.

This selection should be sent after sending the clock mode selection command.

Command	H′3F	Size	Bit rate	Input frequency
	Number of multiplication ratios	Multiplication ratio 1	Multiplication ratio 2	
	SUM			

- Command, H′3F, (one byte): Selection of new bit rate
- Size (one byte): The number of bytes that represents the bit rate, input frequency, number of multiplication ratios, and multiplication ratio
- Bit rate (two bytes): New bit rate
  One hundredth of the value (e.g. when the value is 19200 bps, it will be 192, which is H′00C0.)
- Input frequency (two bytes): Frequency of the clock input to the boot program
  This is valid to the hundredths place and represents the value in MHz multiplied by 100. (E.g. when the value is 20.00 MHz, it will be 2000, which is H′07D0.)
- Number of multiplication ratios (one byte): The number of multiplication ratios to which the device can be set.
- Multiplication ratio 1 (one byte) : The value of multiplication or division ratios for the main operating frequency
  Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H′04.)
  Division ratio: The inverse of the division ratio, as a negative number (e.g. when the clock frequency is divided by two, the value of division ratio will be H′FE. H′FE = D′-2)
- Multiplication ratio 2 (one byte): The value of multiplication or division ratios for the peripheral frequency
  Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H′04.)
  (Division ratio: The inverse of the division ratio, as a negative number (E.g. when the clock is divided by two, the value of division ratio will be H′FE. H′FE = D′-2)
- SUM (one byte): Checksum

Response	H′06

- Response, H′06, (one byte): Response to selection of a new bit rate
  When it is possible to set the bit rate, the response will be ACK.

Error Response	H′BF	ERROR

- Error response, H′BF, (one byte): Error response to selection of new bit rate
- ERROR: (one byte): Error code

H′11: Sum checking error
H′24: Bit-rate selection error
The rate is not available.
H′25: Error in input frequency
This input frequency is not within the specified range.
H′26: Multiplication-ratio error
The ratio does not match an available ratio.
H′27: Operating frequency error
The frequency is not within the specified range.

(5) Received Data Check

The methods for checking of received data are listed below.

1. Input frequency

The received value of the input frequency is checked to ensure that it is within the range of minimum to maximum frequencies which matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.

2. Multiplication ratio

The received value of the multiplication ratio or division ratio is checked to ensure that it matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.

3. Operating frequency error

Operating frequency is calculated from the received value of the input frequency and the multiplication or division ratio. The input frequency is input to the LSI and the LSI is operated at the operating frequency. The expression is given below.

Operating frequency = Input frequency × Multiplication ratio, or
Operating frequency = Input frequency ÷ Division ratio

The calculated operating frequency should be checked to ensure that it is within the range of minimum to maximum frequencies which are available with the clock modes of the specified device. When it is out of this range, an operating frequency error is generated.

4. Bit rate

To facilitate error checking, the value (n) of clock select (CKS) in the serial mode register (SMR), and the value (N) in the bit rate register (BRR), which are found from the peripheral operating clock frequency (ϕ) and bit rate (B), are used to calculate the error rate to ensure that it is less than 4%. If the error is more than 4%, a bit rate error is generated. The error is calculated using the following expression:

$$\text{Error (\%)} = \{[\frac{\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{(2 \times n - 1)}}] - 1\} \times 100$$

When the new bit rate is selectable, the rate will be set in the register after sending ACK in response. The host will send an ACK with the new bit rate for confirmation and the boot program will response with that rate.

Confirmation | H′06 |

- Confirmation, H′06, (one byte): Confirmation of a new bit rate

Response | H′06 |

- Response, H′06, (one byte): Response to confirmation of a new bit rate

The sequence of new bit-rate selection is shown in figure 14.24.

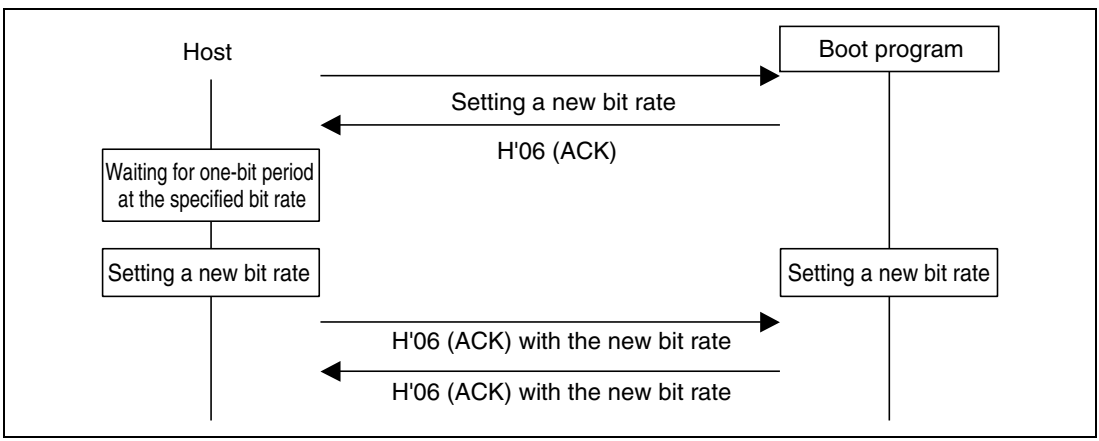


Figure 14.24 New Bit-Rate Selection Sequence

(6) Transition to Programming/Erasing State

The boot program will transfer the erasing program, and erase the user MATs and user boot MATs in that order. On completion of this erasure, ACK will be returned and will enter the programming/erasing state.

The host should select the device code, clock mode, and new bit rate with device selection, clock-mode selection, and new bit-rate selection commands, and then send the command for the transition to programming/erasing state. These procedures should be carried out before sending of the programming selection command or program data.

Command | H′40 |

- Command, H′40, (one byte): Transition to programming/erasing state

Response	H′06

- Response, H′06, (one byte): Response to transition to programming/erasing state
  The boot program will send ACK when the user MAT and user boot MAT have been erased by the transferred erasing program.

Error Response	H′C0	H′51

- Error response, H′C0, (one byte): Error response for user boot MAT blank check
- Error code, H′51, (one byte): Erasing error
  An error occurred and erasure was not completed.

(7) Command Error

A command error will occur when a command is undefined, the order of commands is incorrect, or a command is unacceptable. Issuing a clock-mode selection command before a device selection or an inquiry command after the transition to programming/erasing state command, are examples.

Error Response	H′80	H′xx

- Error response, H′80, (one byte): Command error
- Command, H′xx, (one byte): Received command

(8) Command Order

The order for commands in the inquiry selection state is shown below.

1. A supported device inquiry (H′20) should be made to inquire about the supported devices.
2. The device should be selected from among those described by the returned information and set with a device-selection (H′10) command.
3. A clock-mode inquiry (H′21) should be made to inquire about the supported clock modes.
4. The clock mode should be selected from among those described by the returned information and set.
5. After selection of the device and clock mode, inquiries for other required information should be made, such as the multiplication-ratio inquiry (H′22) or operating frequency inquiry (H′23), which are needed for a new bit-rate selection.
6. A new bit rate should be selected with the new bit-rate selection (H′3F) command, according to the returned information on multiplication ratios and operating frequencies.
7. After selection of the device and clock mode, the information of the user boot MAT and user MAT should be made to inquire about the user boot MATs information inquiry (H′24), user MATs information inquiry (H′25), erased block information inquiry (H′26), and programming unit inquiry (H′27).
8. After making inquiries and selecting a new bit rate, issue the transition to programming/erasing state command (H′40). The boot program will then enter the programming/erasing state.

(9) Programming/Erasing State

A programming selection command makes the boot program select the programming method, a 128-byte programming command makes it program the memory with data, and an erasing selection command and block erasing command make it erase the block. The programming/erasing commands are listed below.

Table 14.13 Programming/Erasing Command

Command	Command Name	Description
H′42	User boot MAT programming selection	Transfers the user boot MAT programming program
H′43	User MAT programming selection	Transfers the user MAT programming program
H′50	128-byte programming	Programs 128 bytes of data
H′48	Erasing selection	Transfers the erasing program
H′58	Block erasing	Erases a block of data
H′52	Memory read	Reads the contents of memory
H′4A	User boot MAT sum check	Checks the checksum of the user boot MAT
H′4B	User MAT sum check	Checks the checksum of the user MAT
H′4C	User boot MAT blank check	Checks the blank data of the user boot MAT
H′4D	User MAT blank check	Checks the blank data of the user MAT
H′4C	User boot MAT blank check	Checks whether the contents of the user boot MAT are blank
H′4D	User MAT blank check	Checks whether the contents of the user MAT are blank
H′4F	Boot program status inquiry	Inquires into the boot program's status

- Programming

  Programming is executed by a programming-selection command and a 128-byte programming command.

  Firstly, the host should send the programming-selection command and select the programming method and programming MATs. There are two programming selection commands, and selection is according to the area and method for programming.

  1. User boot MAT programming selection
  2. User MAT programming selection

After issuing the programming selection command, the host should send the 128-byte programming command. The 128-byte programming command that follows the selection command represents the data programmed according to the method specified by the selection command. When more than 128-byte data is programmed, 128-byte commands should repeatedly be executed. Sending a 128-byte programming command with H′FFFFFFFF as the address will stop the programming. On completion of programming, the boot program will wait for selection of programming or erasing.

Where the sequence of programming operations that is executed includes programming with another method or of another MAT, the procedure must be repeated from the programming selection command.

The sequence for programming-selection and 128-byte programming commands is shown in figure 14.25.

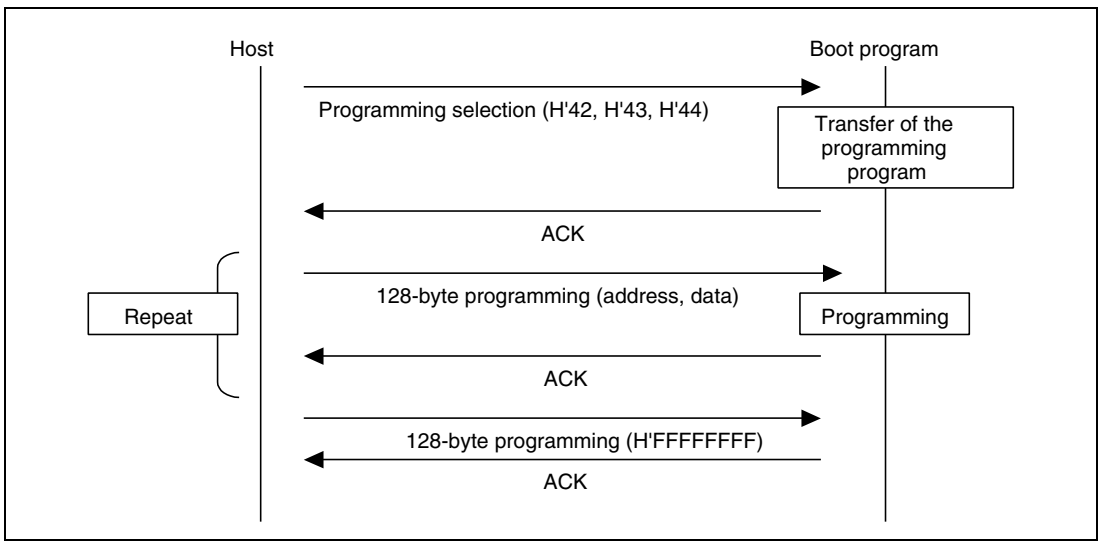


Figure 14.25 Programming Sequence

(a) User boot MAT programming selection

The boot program will transfer a programming program. The data is programmed to the user boot MATs by the transferred programming program.

Command | H′42

- Command, H′42, (one byte): User boot-program programming selection

Response | H′06

- Response, H′06, (one byte): Response to user boot-program programming selection
  When the programming program has been transferred, the boot program will return ACK.

Error Response	H′C2	ERROR

- Error response : H′C2 (1 byte): Error response to user boot MAT programming selection
- ERROR : (1 byte): Error code
  H′54: Selection processing error (transfer error occurs and processing is not completed)

- User-program programming selection
  The boot program will transfer a program for programming. The data is programmed to the user MATs by the transferred program for programming.

Command	H′43

- Command, H′43, (one byte): User-program programming selection

Response	H′06

- Response, H′06, (one byte): Response to user-program programming selection
  When the programming program has been transferred, the boot program will return ACK.

Error Response	H′C3	ERROR

- Error response : H'C3 (1 byte): Error response to user MAT programming selection
- ERROR : (1 byte): Error code
  H'54: Selection processing error (transfer error occurs and processing is not completed)

(b) 128-byte programming

The boot program will use the programming program transferred by the programming selection to program the user boot MATs or user MATs in response to 128-byte programming.

Command	H′50	Address						
	Data	···						
	···							
	SUM							

- Command, H′50, (one byte): 128-byte programming
- Programming Address (four bytes): Start address for programming
  Multiple of the size specified in response to the programming unit inquiry
  (i.e. H′00, H′01, H′00, H′00 : H′01000000)
- Programming Data (128 bytes): Data to be programmed
  The size is specified in the response to the programming unit inquiry.
- SUM (one byte): Checksum

Response	H′06

- Response, H′06, (one byte): Response to 128-byte programming
  On completion of programming, the boot program will return ACK.

Error Response	H′D0	ERROR

- Error response, H′D0, (one byte): Error response for 128-byte programming
- ERROR: (one byte): Error code

  H′11: Checksum Error

  H′53: Programming error

  A programming error has occurred and programming cannot be continued.

The specified address should match the unit for programming of data. For example, when the programming is in 128-byte units, the lower eight bits of the address should be H′00 or H′80. When there are less than 128 bytes of data to be programmed, the host should fill the rest with H′FF.

Sending the 128-byte programming command with the address of H′FFFFFFFF will stop the programming operation. The boot program will interpret this as the end of the programming and wait for selection of programming or erasing.

Command	H′50	Address	SUM

- Command, H′50, (one byte): 128-byte programming
- Programming Address (four bytes): End code is H′FF, H′FF, H′FF, H′FF.
- SUM (one byte): Checksum

Response	H′06

- Response, H′06, (one byte): Response to 128-byte programming
  On completion of programming, the boot program will return ACK.

Error Response	H′D0	ERROR

- Error Response, H′D0, (one byte): Error response for 128-byte programming
- ERROR: (one byte): Error code

  H′11: Checksum error

  H′53: Programming error

  An error has occurred in programming and programming cannot be continued.

(10) Erasure

Erasure is performed with the erasure selection and block erasure command.

Firstly, erasure is selected by the erasure selection command and the boot program then erases the specified block. The command should be repeatedly executed if two or more blocks are to be erased. Sending a block-erasure command from the host with the block number H′FF will stop the erasure operating. On completion of erasing, the boot program will wait for selection of programming or erasing.

The sequences of the issuing of erasure selection commands and the erasure of data are shown in figure 14.26.

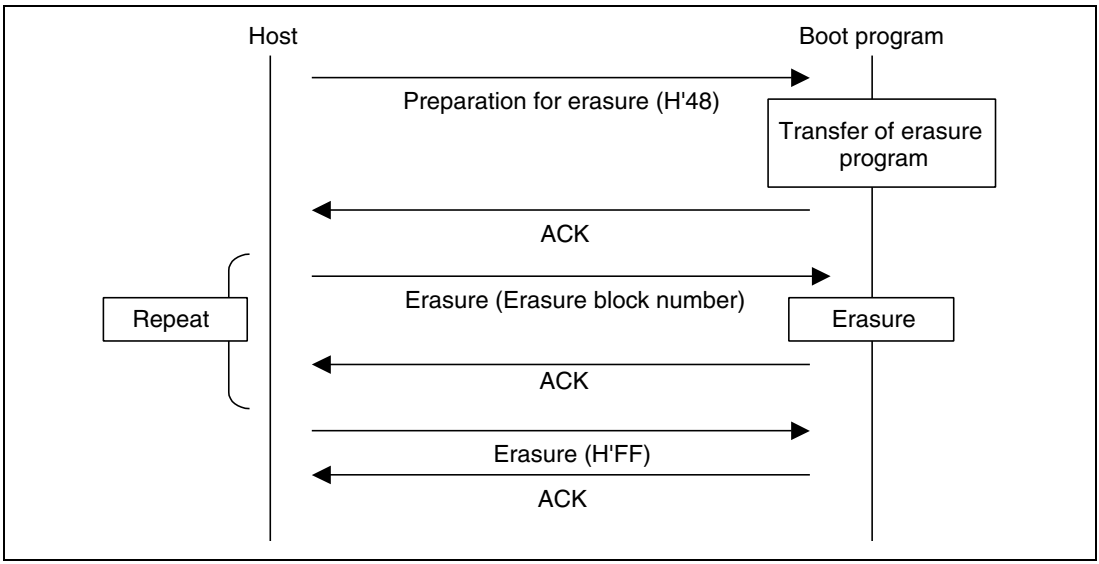


Figure 14.26 Erasure Sequence

(a) Erasure Selection

The boot program will transfer the erasure program. User MAT data is erased by the transferred erasure program.

Command	H′48

- Command, H′48, (one byte): Erasure selection

Response	H′06

- Response, H′06, (one byte): Response for erasure selection
  After the erasure program has been transferred, the boot program will return ACK.

Error Response	H′C8	ERROR

- Error Response, H′C8, (one byte): Error response to erasure selection
- ERROR: (one byte): Error code
  H′54: Selection processing error (transfer error occurs and processing is not completed)

(b) Block Erasure

The boot program will erase the contents of the specified block.

Command	H′58	Size	Block number	SUM

- Command, H′58, (one byte): Erasure

- Size (one byte): The number of bytes that represents the erasure block number
  This is fixed to 1.
- Block number (one byte): Number of the block to be erased
- SUM (one byte): Checksum

Response	H′06

- Response, H′06, (one byte): Response to Erasure
  After erasure has been completed, the boot program will return ACK.

Error Response	H′D8	ERROR

- Error Response, H′D8, (one byte): Response to Erasure
- ERROR (one byte): Error code

  H′11: Sum check error
  H′29: Block number error
  Block number is incorrect.
  H′51: Erasure error
  An error has occurred during erasure.

On receiving block number H′FF, the boot program will stop erasure and wait for a selection command.

Command	H′58	Size	Block number	SUM

- Command, H′58, (one byte): Erasure
- Size, (one byte): The number of bytes that represents the block number
  This is fixed to 1.
- Block number (one byte): H′FF
  Stop code for erasure
- SUM (one byte): Checksum

Response	H′06

- Response, H′06, (one byte): Response to end of erasure (ACK)
  When erasure is to be performed after the block number H′FF has been sent, the procedure should be executed from the erasure selection command.

(11)· Memory read

The boot program will return the data in the specified address.

Command	H′52	Size	Area	Read address	
	Read size				SUM

- Command: H'52 (1 byte): Memory read
- Size (1 byte): Amount of data that represents the area, read address, and read size (fixed at 9)

- Area (1 byte)
  H'00: User boot MAT
  H'01: User MAT
  An address error occurs when the area setting is incorrect.
- Read address (4 bytes): Start address to be read from
- Read size (4 bytes): Size of data to be read
- SUM (1 byte): Checksum

Response

H′52	Read size						
Data	···						
SUM							

- Response: H'52 (1 byte): Response to memory read
- Read size (4 bytes): Size of data to be read
- Data (n bytes): Data for the read size from the read address
- SUM (1 byte): Checksum

Error Response

H′D2	ERROR

- Error response: H'D2 (1 byte): Error response to memory read
- ERROR: (1 byte): Error code
  H'11: Sum check error
  H'2A: Address error
  The read address is not in the MAT.
  H'2B: Size error
  The read size exceeds the MAT.

(12) User-Boot Program Sum Check

The boot program will return the byte-by-byte total of the contents of the bytes of the user-boot program, as a four-byte value.

Command

H′4A

- Command, H′4A, (one byte): Sum check for user-boot program

Response

H′5A	Size	Checksum of user boot program	SUM

- Response, H′5A, (one byte): Response to the sum check of user-boot program
- Size (one byte): The number of bytes that represents the checksum
  This is fixed to 4.
- Checksum of user boot program (four bytes): Checksum of user boot MATs
  The total of the data is obtained in byte units.
- SUM (one byte): Sum check for data being transmitted

(13) User-Program Sum Check

The boot program will return the byte-by-byte total of the contents of the bytes of the user program.

Command	H′4B

- Command, H′4B, (one byte): Sum check for user program

Response	H′5B	Size	Checksum of user program	SUM

- Response, H′5B, (one byte): Response to the sum check of the user program
- Size (one byte): The number of bytes that represents the checksum
  This is fixed to 4.
- Checksum of user boot program (four bytes): Checksum of user MATs
  The total of the data is obtained in byte units.
- SUM (one byte): Sum check for data being transmitted

(14) User Boot MAT Blank Check

The boot program will check whether or not all user boot MATs are blank and return the result.

Command	H′4C

- Command, H′4C, (one byte): Blank check for user boot MAT

Response	H′06

- Response, H′06, (one byte): Response to the blank check of user boot MAT
  If all user MATs are blank (H′FF), the boot program will return ACK.

Error Response	H′CC	H′52

- Error Response, H′CC, (one byte): Response to blank check for user boot MAT
- Error Code, H′52, (one byte): Erasure has not been completed.

(15) User MAT Blank Check

The boot program will check whether or not all user MATs are blank and return the result.

Command	H′4D

- Command, H′4D, (one byte): Blank check for user MATs

Response	H′06

- Response, H′06, (one byte): Response to the blank check for user boot MATs
  If the contents of all user MATs are blank (H′FF), the boot program will return ACK.

Error Response	H′CD	H′52

- Error Response, H′CD, (one byte): Error response to the blank check of user MATs.

- Error code, H′52, (one byte): Erasure has not been completed.

(16) Boot Program State Inquiry

The boot program will return indications of its present state and error condition. This inquiry can be made in the inquiry/selection state or the programming/erasing state.

Command	H′4F

- Command, H′4F, (one byte): Inquiry regarding boot program's state

Response	H′5F	Size	Status	ERROR	SUM

- Response, H′5F, (one byte): Response to boot program state inquiry
- Size (one byte): The number of bytes. This is fixed to 2.
- Status (one byte): State of the boot program
- ERROR (one byte): Error status

  ERROR = 0 indicates normal operation.
  ERROR = 1 indicates error has occurred.

- SUM (one byte): Sum check

Table 14.14 Status Code

Code	Description
H′11	Device Selection Wait
H′12	Clock Mode Selection Wait
H′13	Bit Rate Selection Wait
H′1F	Programming/Erasing State Transition Wait (Bit rate selection is completed)
H′31	Programming State for Erasure
H′3F	Programming/Erasing Selection Wait (Erasure is completed)
H′4F	Programming Data Receive Wait
H′5F	Erasure Block Specification Wait (Erasure is completed)

Table 14.15 Error Code

Code	Description
H′00	No Error
H′11	Sum Check Error
H′12	Program Size Error
H′21	Device Code Mismatch Error
H′22	Clock Mode Mismatch Error
H′24	Bit Rate Selection Error
H′25	Input Frequency Error
H′26	Multiplication Ratio Error
H′27	Operating Frequency Error
H′29	Block Number Error
H′2A	Address Error
H′2B	Data Length Error
H′51	Erasure Error
H′52	Erasure Incomplete Error
H′53	Programming Error
H′54	Selection Processing Error
H′80	Command Error
H′FF	Bit-Rate-Adjustment Confirmation Error

14.10 Usage Notes

1. The initial state of the Hitachi product at its shipment is in the erased state. For the product whose revision of erasing is undefined, we recommend to execute automatic erasure for checking the initial state (erased state) and compensating.
2. For the PROM programmer suitable for programmer mode in this LSI and its program version, refer to the instruction manual of the socket adapter.
3. If the socket, socket adapter, or product index does not match the specifications, too much current flows and the product may be damaged.
4. If a voltage higher than the rated voltage is applied, the product may be fatally damaged. Use a PROM programmer that supports the Hitachi 256 kbytes flash memory on-chip MCU device at 3.3 V. Do not set the programmer to HN28F101 or the programming voltage to 5.0 V. Use only the specified socket adapter. If other adapters are used, the product may be damaged.
5. Do not remove the chip from the PROM programmer nor input a reset signal during programming/erasing. As a high voltage is applied to the flash memory during programming/erasing, doing so may damage or destroy flash memory permanently. If reset is executed accidentally, reset must be released after the reset input period of 100 µs which is longer than normal.
6. The flash memory is not accessible until FKEY is cleared after programming/erasing completes. If this LSI is restarted by a reset immediately after programming/erasing has finished, secure the reset period (period of $\overline{RES}$ = 0) of more than 100 µs. Though transition to the reset state or hardware standby state during programming/erasing is prohibited, if reset is executed accidentally, reset must be released after the reset input period of 100 µs which is longer than normal.
7. At powering on or off the Vcc power supply, fix the $\overline{RES}$ pin to low and set the flash memory to hardware protection state. This power on/off timing must also be satisfied at a power-off and power-on caused by a power failure and other factors.
8. Program the area with 128-byte programming-unit blocks in on-board programming or programmer mode only once. Perform programming in the state where the programming-unit block is fully erased.
9. When the chip is to be reprogrammed with the programmer after execution of programming or erasure in on-board programming mode, it is recommend that automatic programming is performed after execution of automatic erasure.
10. To write data or programs to the flash memory, data or programs must be allocated to addresses higher than that of the external interrupt vector table (H'000040) and H'FF must be written to the areas that are reserved for the system in the exception handling vector table.
11. If data other than H'FF (four bytes) is written to the key code area (H'00003C to H'00003F) of flash memory, reading cannot be performed in programmer mode. (In this case, data is read as H'00. Rewrite is possible after erasing the data.) For reading in programmer mode, make sure to write H'FF to the entire key code area. If data other than H'FF is to be written to the key

code area in programmer mode, a verification error will occur unless a software countermeasure is taken for the PROM programmer.

12. The programming program that includes the initialization routine and the erasing program that includes the initialization routine are each 2 kbytes or less. Accordingly, when the CPU clock frequency is 10 MHz, the download for each program takes approximately 600 μs at the maximum.
13. While an instruction in on-chip RAM is being executed, the DMAC can write to the SCO bit in FCCS that is used for a download request or FMATS that is used for MAT switching. Make sure that these registers are not accidentally written to, otherwise an on-chip program may be downloaded and damage RAM or a MAT switchover may occur and the CPU get out of control. Do not use DMAC to program flash related registers.
14. A programming/erasing program for flash memory used in the conventional F-ZTAT H8S microcomputer which does not support download of the on-chip program by a SCO transfer request cannot run in this LSI. Be sure to download the on-chip program to execute programming/erasing of flash memory in this LSI.
15. Unlike the conventional F-ZTAT H8S microcomputer, no countermeasures are available for a runaway by WDT during programming/erasing. Prepare countermeasures (e.g. use of the periodic timer interrupts) for WDT with taking the programming/erasing time into consideration as required.

Section 15 Clock Pulse Generator

This LSI has an on-chip clock pulse generator that generates the system clock (ϕ) and internal clocks. The clock pulse generator consists of an oscillator circuit and PLL circuit. Figure 15.1 shows a block diagram of the clock pulse generator.

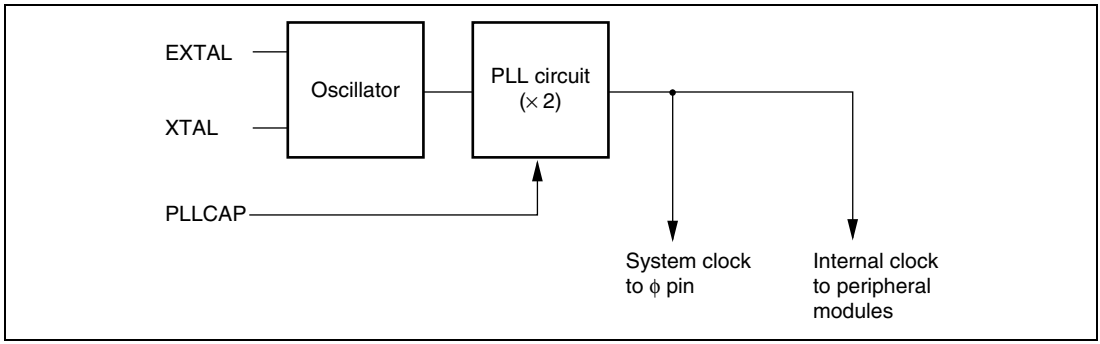


Figure 15.1 Block Diagram of Clock Pulse Generator

The frequency can be multiplied by two by means of the PLL circuit.

15.1 Oscillator

Clock pulses can be supplied by connecting a crystal resonator, or by input of an external clock.

15.1.1 Connecting Crystal Resonator

A crystal resonator can be connected as shown in the example in figure 15.2. Select the damping resistance R_d according to table 15.1. An AT-cut parallel-resonance type should be used. When a crystal resonator is used, the range of usable frequencies is from 5 to 16.5 MHz and a crystal resonator with half frequency of the system clock (ϕ) should be used.

Figure 15.3 shows the equivalent circuit of the crystal resonator. Use a crystal resonator that has the characteristics shown in table 15.2.

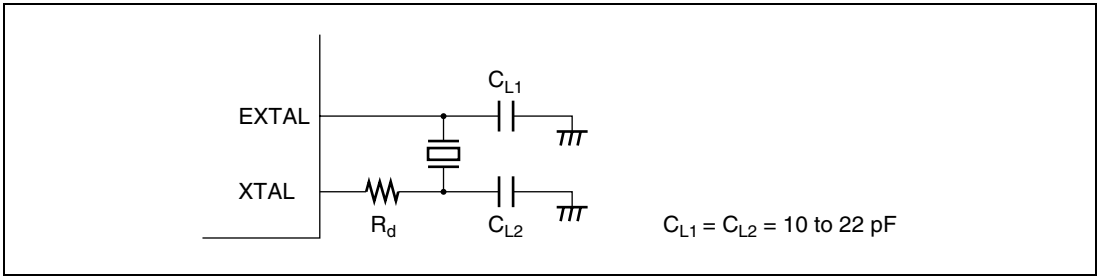


Figure 15.2 Connection of Crystal Resonator (Example)

Table 15.1 Damping Resistance Value

Frequency (MHz)	5	8	12	16.5
R_d (Ω)	425	200	0	0

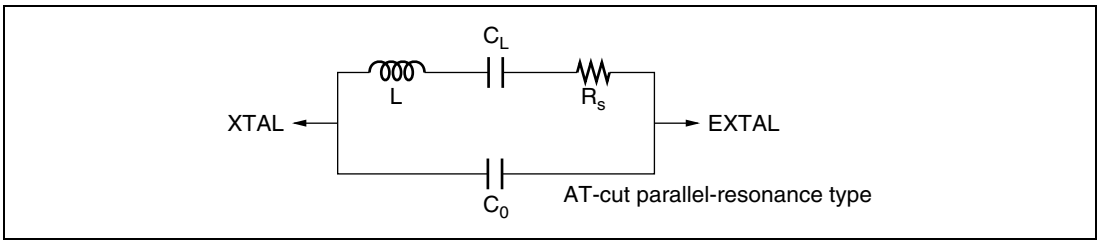


Figure 15.3 Crystal Resonator Equivalent Circuit

Table 15.2 Crystal Resonator Characteristics

Frequency (MHz)	5	8	12	16.5
R_S max (Ω)	110	80	60	50
C_0 max (pF)	7	7	7	7

15.1.2 External Clock Input

An external clock signal can be input as shown in the examples in figure 15.4. If the XTAL pin is left open, make sure that parasitic capacitance is no more than 10 pF. When the counter clock is input to the XTAL pin, make sure that the external clock is held high in standby mode.

Table 15.3 shows the input conditions for the external clock. When an external clock is used, the range of usable frequencies is from 5 to 16.5 MHz and an external clock with half frequency of the system clock (ϕ) should be used.

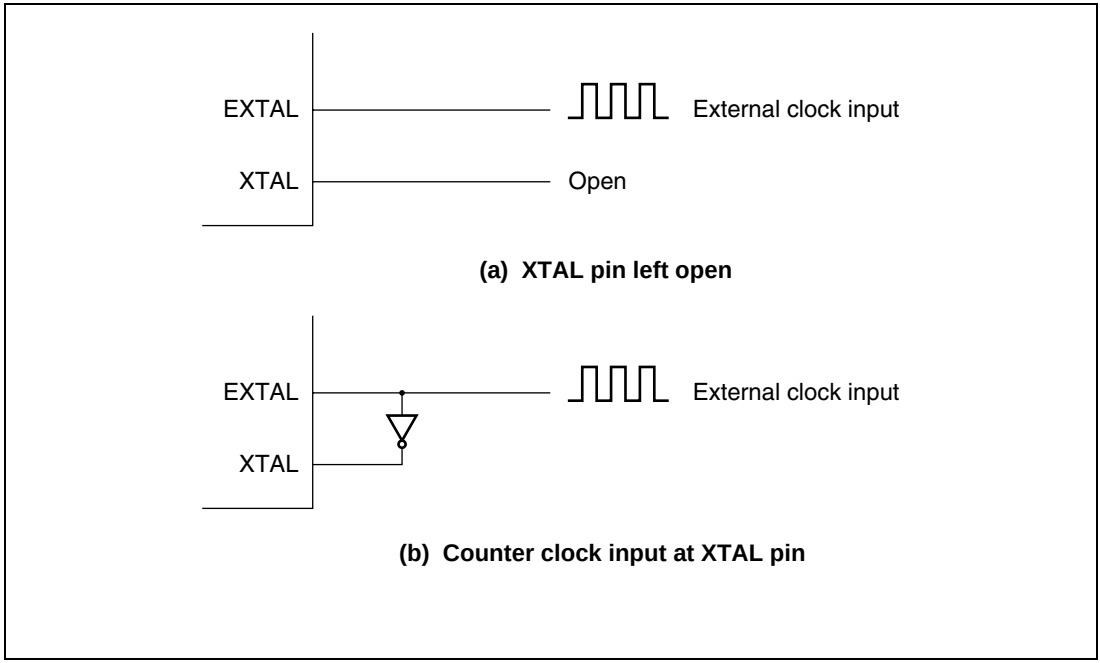


Figure 15.4 External Clock Input (Examples)

Table 15.3 External Clock Input Conditions

Item	Symbol	V_{cc} = 3.0 V to 3.6 V Min.	Max.	Unit	Test Conditions
External clock input low pulse width	t_{EXL}	10	—	ns	Figure 15.5
External clock input high pulse width	t_{EXH}	10	—	ns	
External clock rise time	t_{EXr}	—	5	ns	
External clock fall time	t_{EXf}	—	5	ns	
Clock low pulse width	t_{CL}	0.4	0.6	t_{cyc}	
Clock high pulse width	t_{CH}	0.4	0.6	t_{cyc}	

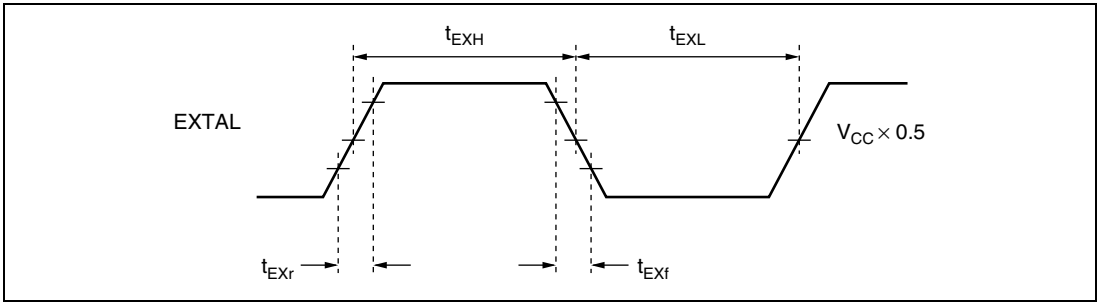


Figure 15.5 External Clock Input Timing

When the specified clock signal is input to the EXTAL pin, an internal clock signal output is ensured after the external clock output stabilization delay time (t_{DEXT}) is passed. Since the clock signal output is not ensured during the t_{DEXT} period, the reset signal should be set to low and the reset state should be retained. Table 15.4 shows the external clock output stabilization delay time and figure 15.6 shows the timing of the external clock output stabilization delay time.

Table 15.4 External Clock Output Stabilization Delay Time

Conditions: Vcc = 3.0 V to 3.6 V, Vss = 0 V

Item	Symbol	Min.	Max.	Unit	Remark
External clock output stabilization delay time	t_{DEXT}*	500	—	µs	Figure 15.6

Note: * t_{DEXT} includes the $\overline{RES}$ pulse width (t_{RESW}).

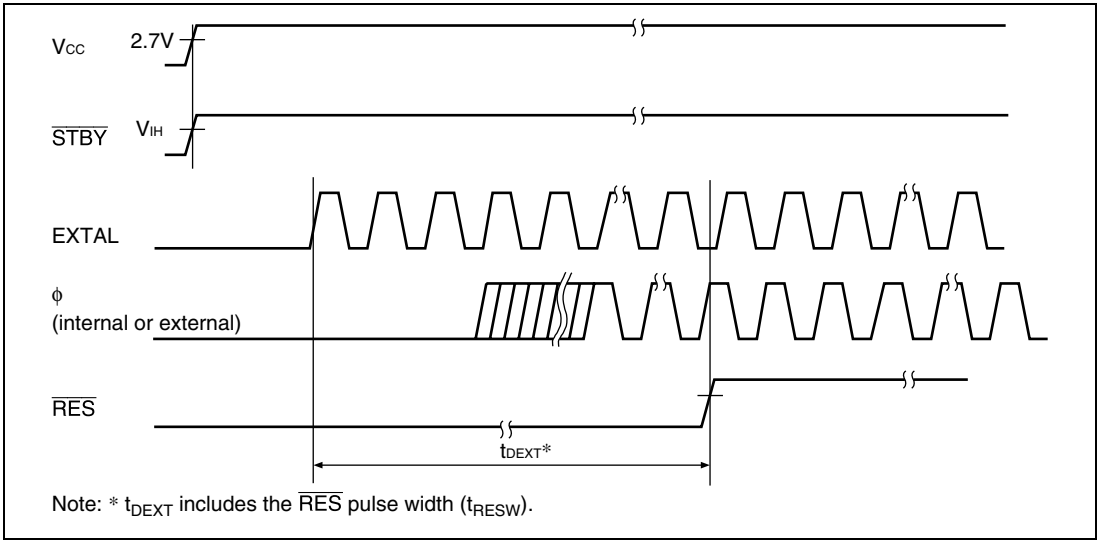


Figure 15.6 Timing of External Clock Output Stabilization Delay Time

15.2 PLL Circuit

The PLL circuit has the function of multiplying the frequency of the clock from the oscillator by a factor of 2. Therefore, a 16.5-MHz clock should be input to realize the internal 33-MHz operation. The phase of the rising edge of the internal clock is controlled so as to match that of the rising edge of the EXTAL pin.

15.3 Usage Notes

15.3.1 Notes on Resonator

Since various characteristics related to the resonator are closely linked to the user's board design, thorough evaluation is necessary on the user's part, using the resonator connection examples shown in this section as a guide. As the parameters for the oscillation circuit will depend on the floating capacitance of the resonator and the user board, the parameters should be determined in consultation with the resonator manufacturer. The design must ensure that a voltage exceeding the maximum rating is not applied to the resonator pin.

15.3.2 Notes on Board Design

When using the crystal resonator, place the crystal resonator and its load capacitors as close as possible to the XTAL and EXTAL pins. Other signal lines should be routed away from the oscillation circuit to prevent induction from interfering with correct oscillation. See figure 15.7.

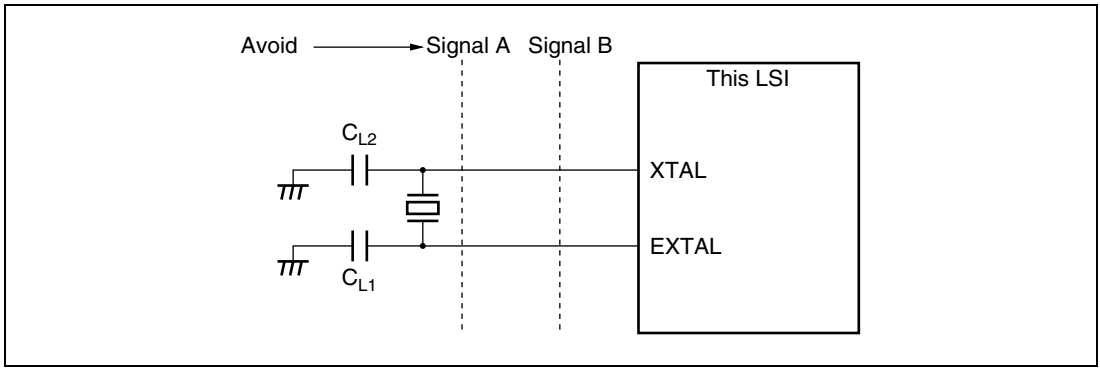


Figure 15.7 Note on Board Design for Oscillation Circuit

Section 16 Power-Down Modes

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

This LSI's operating modes are high-speed mode and four power down modes:

- Sleep mode
- Module stop mode
- Software standby mode
- Hardware standby mode

Sleep mode is a CPU state and module stop mode is an on-chip peripheral function state. A combination of these modes can be set.

After a reset, this LSI is in high-speed mode.

Table 16.1 shows the internal states of this LSI in each mode. Figure 16.1 shows the mode transition diagram.

Table 16.1 Operating Modes and Internal States of LSI

Operating State		High-Speed Mode	Sleep Mode	Module Stop Mode	Software Standby Mode	Hardware Standby Mode
Clock pulse generator		Functions	Functions	Functions	Halted	Halted
CPU	Instruction execution	Functions	Halted	Functions	Halted	Halted
	Register		Retained		Retained	Undefined
External interrupts	NMI	Functions	Functions	Functions	Functions	Halted
	IRQ0 to IRQ7					
Peripheral functions	WDT	Functions	Functions	Functions	Halted (Retained)	Halted (Reset)
	TMR	Functions	Functions	Halted (Retained)	Halted (Retained)	Halted (Reset)
	DMAC	Functions	Functions	Functions	Halted (Retained)	Halted (Reset)
	SCI	Functions	Functions	Halted (Reset)	Halted (Reset)	Halted (Reset)
	USB2	Functions	Functions	Halted (Retained)	Halted (Retained)	Halted (Reset)
	RAM	Functions	Functions	Functions	Retained	Retained
	I/O	Functions	Functions	Functions	Retained	High impedance

Note: Halted (Retained) in the table means that internal register values are retained and internal operations are suspended.

Halted (Reset) in the table means that internal register values and internal states are initialized.

In module stop mode, only modules for which a stop setting has been made are halted (reset or retained).

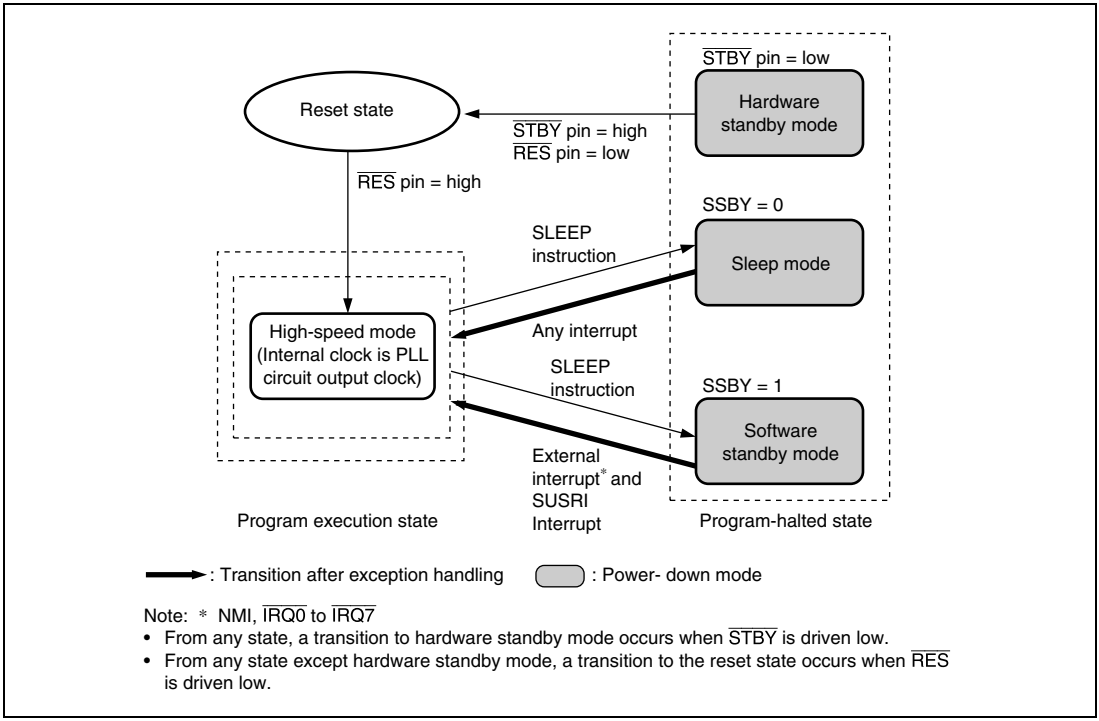


Figure 16.1 Mode Transitions

16.1 Register Descriptions

The registers relating to the power-down mode are shown below.

- Standby control register (SBYCR)
- Module stop control registers H and L (MSTPCRH, MSTPCRL)

16.1.1 Standby Control Register (SBYCR)

SBYCR performs software standby mode control.

Bit	Bit Name	Initial Value	R/W	Description
7	SSBY	0	R/W	Software Standby This bit specifies the transition mode after executing the SLEEP instruction 0: Shifts to sleep mode after the SLEEP instruction is executed 1: Shifts to software standby mode after the SLEEP instruction is executed This bit does not change when clearing the software standby mode by using external interrupts and shifting to normal operation. This bit should be written to 0 when clearing.
6	STS2	0	R/W	Standby Timer Select 2 to 0 These bits select the time the MCU waits for the clock to stabilize when software standby mode is cleared. Make a selection according to the operating frequency so that the standby time is at least oscillation stabilization time. Table 16.2 shows the relationship between setting values and number of wait states. With an external clock, any selection is possible. Normally the minimum value is recommended.
5	STS1	0	R/W	
4	STS0	0	R/W	
3 to 0	—	All 0	R/W	Reserved The initial value should not be changed.

16.1.2 Module Stop Control Registers H and L (MSTPCRH, MSTPCRL)

MSTPCR performs module stop mode control. Setting a bit to 1, the corresponding module enters module stop mode, while clearing the bit to 0 clears the module stop mode.

- MSTPCRH

Bit	Bit Name	Initial Value	R/W	Module
7	—	0*1	R/W	—
6	—	0	R/W	—
5	—	1*2	R/W	—
4	TMRCKSTP	1	R/W	8-bit timer (TMR)
3	—	1*2	R/W	—
2	—	1*2	R/W	—
1	—	1*2	R/W	—
0	—	1*2	R/W	—

- MSTPCRL

Bit	Bit Name	Initial Value	R/W	Module
7	SCICKSTP	1	R/W	Serial communication interface (SCI)
6	—	1*2	R/W	—
5	—	1*2	R/W	—
4	USBCKSTP	1	R/W	Universal serial bus interface 2 (USB2)
3	—	1*2	R/W	—
2	—	1*2	R/W	—
1	—	1*2	R/W	—
0	—	1*2	R/W	—

Note: *1 This bit must not set to 1.
*2 These bits must not be cleared to 0.

16.2 Operation

16.2.1 Sleep Mode

Transition to Sleep Mode: When the SLEEP instruction is executed while the SSBY bit in SBYCR is set to 0, the CPU enters the sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other peripheral functions do not stop.

Exiting Sleep Mode: Sleep mode is exited by any interrupt, or signals at the $\overline{\text{RES}}$, or $\overline{\text{STBY}}$ pins.

- Exiting sleep mode by interrupts

  When an interrupt occurs, sleep mode is exited and interrupt exception processing starts. Sleep mode is not exited if the interrupt is disabled, or interrupts other than NMI are masked by the CPU.
- Exiting sleep mode by $\overline{\text{RES}}$ pin

  Setting the $\overline{\text{RES}}$ pin level low selects the reset state. After the stipulated reset input duration, driving the $\overline{\text{RES}}$ pin high starts the CPU performing reset exception processing.
- Exiting sleep mode by $\overline{\text{STBY}}$ pin

  When the $\overline{\text{STBY}}$ pin level is driven low, a transition is made to hardware standby mode.

16.2.2 Software Standby Mode

Transition to Software Standby Mode: If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, software standby mode is entered. In this mode, the CPU, on-chip peripheral functions, and oscillator all stop. However, the contents of the CPU's internal registers, RAM data, and the states of on-chip peripheral functions other than the SCI, and I/O ports, are retained. In this mode the oscillator stops, and therefore power consumption is significantly reduced.

Clearing Software Standby Mode: Software standby mode is cleared by an external interrupt (NMI pin, or pins IRQ0 to IRQ7), or by means of the $\overline{\text{RES}}$ pin or $\overline{\text{STBY}}$ pin.

- Clearing with an interrupt

  When an NMI, IRQ0 to IRQ7 or SUSRI interrupt request signal is input, clock oscillation starts, and after the elapse of the time set in bits STS2 to STS0 in SBYCR, stable clocks are supplied to the entire LSI, software standby mode is cleared, and interrupt exception handling is started.

  When clearing software standby mode with an IRQ0 to IRQ7 or SUSRI interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than interrupts IRQ0 to IRQ7 or SUSRI is generated. Software standby mode cannot be cleared if the interrupt has been masked on the CPU side.

- Clearing with the $\overline{\text{RES}}$ pin

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

  When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode.

Setting Oscillation Stabilization Time after Clearing Software Standby Mode: Bits STS2 to STS0 in SBYCR should be set as described below.

- Using a crystal resonator

  Set bits STS2 to STS0 so that the standby time is more than the oscillation stabilization time.

  Table 16.2 shows the standby times for operating frequencies and settings of bits STS2 to STS0.
- Using an external clock

  A PLL circuit stabilization time is necessary. Refer to table 16.2 to set the standby time.

Table 16.2 Operating Frequency and Standby Time

STS2	STS1	STS0	Standby Time	10 MHz	20 MHz	33 MHz	Unit
0	0	0	8192 states	0.8	0.4	0.2	ms
		1	16384 states	1.6	0.8	0.4	
	1	0	32768 states	3.2	1.6	0.9	
		1	65536 states	6.5	3.2	1.9	
1	0	0	131072 states	13.1	6.5	3.9	
		1	262144 states	26.2	13.1	7.9	
	1	0	Reserved	—	—	—	—
		1	16 states*	1.6	0.8	0.4	µs

: Recommended setting time

Note: * This setting must not be used in the flash memory version.

Software Standby Mode Application Example: Figure 16.2 shows an example in which a transition is made to software standby mode at the falling edge on the NMI pin, and software standby mode is cleared at the rising edge on the NMI pin.

In this example, an NMI interrupt is accepted with the NMIEG bit in INTCR cleared to 0 (falling edge specification), then the NMIEG bit is set to 1 (rising edge specification), the SSBY bit is set to 1, and a SLEEP instruction is executed, causing a transition to software standby mode.

Software standby mode is then cleared at the rising edge on the NMI pin.

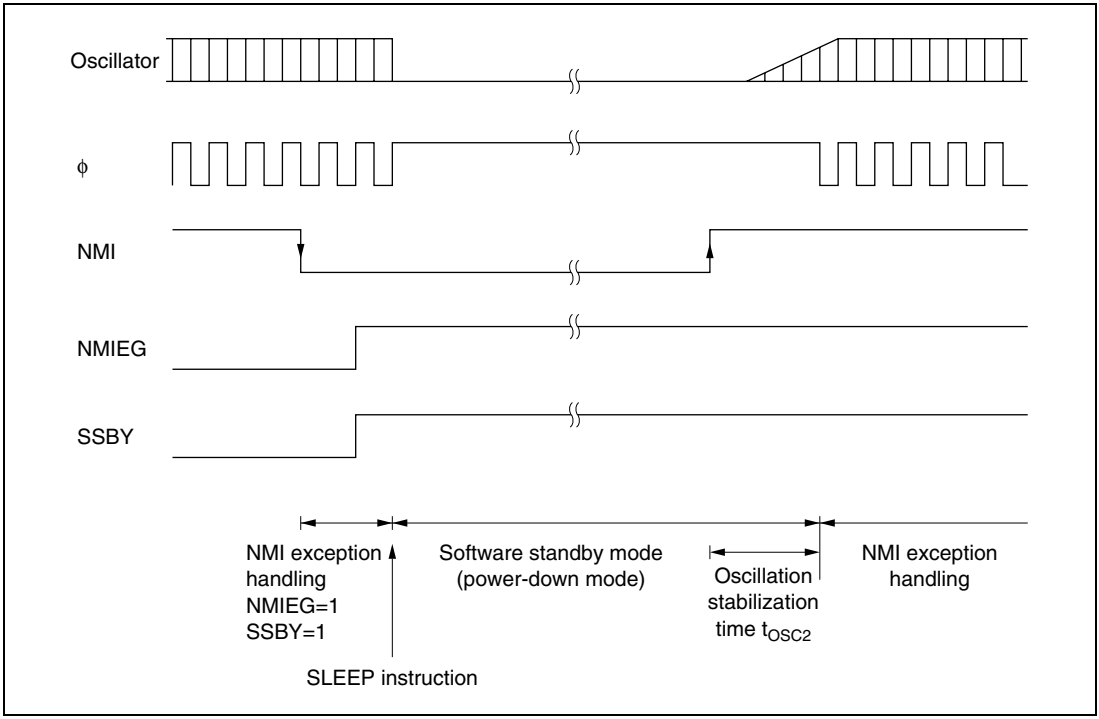


Figure 16.2 Software Standby Mode Application Example

16.2.3 Hardware Standby Mode

Transition to Hardware Standby Mode: When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode from any mode.

In hardware standby mode, all functions enter the reset state and stop operation, resulting in a significant reduction in power consumption. As long as the prescribed voltage is supplied, on-chip RAM data is retained. I/O ports are set to the high-impedance state.

In order to retain on-chip RAM data, the RAME bit in SYSCR should be cleared to 0 before driving the $\overline{\text{STBY}}$ pin low. Do not change the state of the mode pins ($\overline{\text{MD2}}$, MD1) while this LSI is in hardware standby mode.

Clearing Hardware Standby Mode: Hardware standby mode is cleared by means of the $\overline{\text{STBY}}$ pin and the $\overline{\text{RES}}$ pin. When the $\overline{\text{STBY}}$ pin is driven high while the $\overline{\text{RES}}$ pin is low, the reset state is set and clock oscillation is started. Ensure that the $\overline{\text{RES}}$ pin is held low until the clock oscillator stabilizes (for details on the oscillation stabilization time, refer to table 16.2). When the $\overline{\text{RES}}$ pin is subsequently driven high, a transition is made to the program execution state via the reset exception handling state.

Hardware Standby Mode Timing: Figure 16.3 shows an example of hardware standby mode timing. When the $\overline{\text{STBY}}$ pin is driven low after the $\overline{\text{RES}}$ pin has been driven low, a transition is made to hardware standby mode. Hardware standby mode is cleared by driving the $\overline{\text{STBY}}$ pin high, waiting for the oscillation stabilization time, then changing the $\overline{\text{RES}}$ pin from low to high.

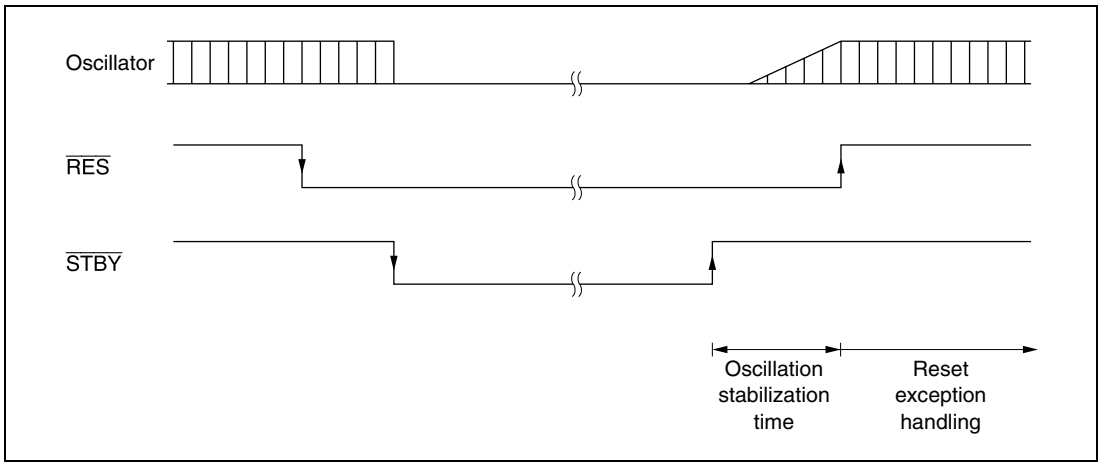


Figure 16.3 Hardware Standby Mode Timing

16.2.4 Module Stop Mode

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

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

When the corresponding bit is cleared to 0, module stop mode is cleared and the module starts operating at the end of the bus cycle. In module stop mode, the internal states of modules other than the SCI are retained.

After reset clearance, all modules other than the DMAC are in module stop mode.

The module registers which are set in module stop mode cannot be read or written to.

16.3 Usage Notes

16.3.1 I/O Port Status

In software standby mode, I/O port states are retained. Therefore, there is no reduction in current consumption for the output current when a high-level signal is output.

16.3.2 Current Consumption during Oscillation Stabilization Standby Period

Current consumption increases during the oscillation stabilization standby period.

16.3.3 On-Chip Peripheral Module Interrupts

Relevant interrupt operations cannot be performed in module stop mode. Consequently, if module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source activation source. Interrupts should therefore be disabled before entering module stop mode.

16.3.4 Writing to MSTPCR

MSTPCR should only be written to by the CPU.

Section 17 List of Registers

This section gives information on the on-chip I/O register addresses, how the register bits are configured, and the register states in each operating mode. The information is given as shown below.

1. Register addresses (address order)
- Registers are listed from the lower allocation addresses.
- Registers are classified by functional modules.
- The access size is indicated.

2. Register bits
- Bit configurations of the registers are described in the same order as the register addresses.
- Reserved bits are indicated by — in the bit name column.
- The bit number in the bit-name column indicates that the whole register is allocated as a counter or for holding data.
- For the registers of 16 or 24 bits, the MSB is described first.

3. Register states in each operating mode
- Register states are described in the same order as the register addresses.
- The register states described here are for the basic operating modes. If there is a specific reset for an on-chip peripheral module, refer to the section on that on-chip peripheral module.

17.1 Register Addresses (Address Order)

The data bus width indicates the numbers of bits by which the register is accessed.

The number of access states indicates the number of states based on the specified reference clock.

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access States
Interrupt flag register 0	IFR0	32	H'FD00	USB2	32	2
Interrupt select register 0	ISR0	32	H'FD04	USB2	32	2
Interrupt enable register 0	IER0	32	H'FD08	USB2	32	2
EP0o receive data size register	EPSZ0o	32	H'FD0C	USB2	32	2
EP1 receive data size register	EPSZ1	32	H'FD10	USB2	32	2
EP0i data register	EPDR0i	32	H'FD14	USB2	32	2
EP0o data register	EPDR0o	32	H'FD18	USB2	32	2
EP0s data register	EPDR0s	32	H'FD1C	USB2	32	2
EP1 data register	EPDR1	32	H'FD20	USB2	32	2
EP2 data register	EPDR2	32	H'FD30	USB2	32	2
EP3 data register 3	EPDR3	32	H'FD40	USB2	32	2
Data status register 0	DASTS0	32	H'FD44	USB2	32	2
Packet enable register 0i	PKTE0i	32	H'FD48	USB2	32	2
Packet enable register 2	PKTE2	32	H'FD4C	USB2	32	2
Packet enable register 3	PKTE3	32	H'FD50	USB2	32	2
FIFO clear register 0	FCLR0	32	H'FD54	USB2	32	2
Endpoint stall register 0	EPSTL0	32	H'FD58	USB2	32	2
DMA setting register	DMA0	32	H'FD5C	USB2	32	2
Control register	CTRL	32	H'FD60	USB2	32	2
Standby control register	SBYCR	8	H'FE14	SYSTEM	8	2
Module stop control register H	MSTPCRH	8	H'FE16	SYSTEM	8	2
Module stop control register L	MSTPCRL	8	H'FE17	SYSTEM	8	2
System control register	SYSCR	8	H'FE1C	SYSTEM	8	2
Mode control register	MDCR	8	H'FE1D	SYSTEM	8	2
USB suspend status register	USBSUSP	8	H'FE1E	USB2	8	2
Access control register	ACSCR	8	H'FE20	BSC	16	2
CS assertion period control register	CSACR	8	H'FE21	BSC	16	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access States
Wait control register	WTCR	16	H'FE22	BSC	16	2
Bus control register	BCR	16	H'FE24	BSC	16	2
Read strobe timing control register	RDNCR	8	H'FE26	BSC	16	2
RAM emulation register	RAMER	8	H'FE27	FLASH	16	2
DRAM control register	DRAMCR	16	H'FE28	BSC	16	2
DRAM access control register	DRACCR	8	H'FE2A	BSC	16	2
Refresh control register	REFCR	16	H'FE2C	BSC	16	2
Refresh timer counter	RTCNT	8	H'FE2E	BSC	16	2
Refresh time constant register	RTCOR	8	H'FE2F	BSC	16	2
Timer control/status register	TCSR	8	H'FE70	WDT	16	2
Timer counter	TCNT	8	H'FE71	WDT	16	2
Flash code control/status register	FCCS	8	H'FE88	FLASH	8	2
Flash program code select register	FPCS	8	H'FE89	FLASH	8	2
Flash erase code select register	FECS	8	H'FE8A	FLASH	8	2
Flash key code register	FKEY	8	H'FE8C	FLASH	8	2
Flash MAT select register	FMATS	8	H'FE8D	FLASH	8	2
Flash transfer destination address register	FTDAR	8	H'FE8E	FLASH	8	2
DMA source address register_0	DMSAR_0	32	H'FEB0	DMAC_0	16	2
DMA destination address register_0	DMDAR_0	32	H'FEB4	DMAC_0	16	2
DMA transfer count register_0	DMTCR_0	32	H'FEB8	DMAC_0	16	2
DMA mode control register_0	DMMDR_0	16	H'FEBC	DMAC_0	16	2
DMA address control register_0	DMACR_0	16	H'FEBE	DMAC_0	16	2
DMA source address register_1	DMSAR_1	32	H'FEC0	DMAC_1	16	2
DMA destination address register_1	DMDAR_1	32	H'FEC4	DMAC_1	16	2
DMA transfer count register_1	DMTCR_1	32	H'FEC8	DMAC_1	16	2
DMA mode control register_1	DMMDR_1	16	H'FECC	DMAC_1	16	2
DMA address control register_1	DMACR_1	16	H'FECE	DMAC_1	16	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access States
DMA source address register_2	DMSAR_2	32	H'FED0	DMAC_2	16	2
DMA destination address register_2	DMDAR_2	32	H'FED4	DMAC_2	16	2
DMA transfer count register_2	DMTCR_2	32	H'FED8	DMAC_2	16	2
DMA mode control register_2	DMMDR_2	16	H'FEDC	DMAC_2	16	2
DMA address control register_2	DMACR_2	16	H'FEDE	DMAC_2	16	2
DMA source address register_3	DMSAR_3	32	H'FEE0	DMAC_3	16	2
DMA destination address register_3	DMDAR_3	32	H'FEE4	DMAC_3	16	2
DMA transfer count register_3	DMTCR_3	32	H'FEE8	DMAC_3	16	2
DMA mode control register_3	DMMDR_3	16	H'FEEC	DMAC_3	16	2
DMA address control register_3	DMACR_3	16	H'FEEE	DMAC_3	16	2
USB transfer control register	USTCR	16	H'FEF0	DMAC	16	2
IRQ enable register	IER	8	H'FF46	INT	8	2
Interrupt control register A	ICRA	8	H'FF48	INT	8	2
Interrupt control register B	ICRB	8	H'FF49	INT	8	2
Interrupt control register C	ICRC	8	H'FF4A	INT	8	2
IRQ status register	ISR	8	H'FF4B	INT	8	2
IRQ sense control register H	ISCRH	8	H'FF4C	INT	8	2
IRQ sense control register L	ISCRL	8	H'FF4D	INT	8	2
Address break control register	ABRKCR	8	H'FF54	INT	8	2
Break address register A	PBARA	8	H'FF55	INT	8	2
Break address register B	PBARB	8	H'FF56	INT	8	2
Break address register C	PBARC	8	H'FF57	INT	8	2
IRQ sense port select register	ISSR	8	H'FF5D	PORT	8	2
Port function control register 1	PFCR1	8	H'FF60	PORT	8	2
Port function control register 3	PFCR3	8	H'FF62	PORT	8	2
Port 1 data direction register	P1DDR	8	H'FF70	PORT	8	2
Port 2 data direction register	P2DDR	8	H'FF71	PORT	8	2
Port 3 data direction register	P3DDR	8	H'FF72	PORT	8	2
Port 4 data direction register	P4DDR	8	H'FF73	PORT	8	2
Port 5 data direction register	P5DDR	8	H'FF74	PORT	8	2
Port 6 data direction register	P6DDR	8	H'FF75	PORT	8	2
Port 7 data direction register	P7DDR	8	H'FF76	PORT	8	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access States
Port 8 data direction register	P8DDR	8	H'FF77	PORT	8	2
Port 9 data direction register	P9DDR	8	H'FF78	PORT	8	2
Port A data direction register	PADDR	8	H'FF79	PORT	8	2
Serial mode register	SMR	8	H'FF88	SCI	8	2
Bit rate register	BRR	8	H'FF89	SCI	8	2
Serial control register	SCR	8	H'FF8A	SCI	8	2
Transmit data register	TDR	8	H'FF8B	SCI	8	2
Serial status register	SSR	8	H'FF8C	SCI	8	2
Receive data register	RDR	8	H'FF8D	SCI	8	2
Port 1 register	PORT1	8	H'FFC0	PORT	8	2
Port 2 register	PORT2	8	H'FFC1	PORT	8	2
Port 3 register	PORT3	8	H'FFC2	PORT	8	2
Port 4 register	PORT4	8	H'FFC3	PORT	8	2
Port 5 register	PORT5	8	H'FFC4	PORT	8	2
Port 6 register	PORT6	8	H'FFC5	PORT	8	2
Port 7 register	PORT7	8	H'FFC6	PORT	8	2
Port 8 register	PORT8	8	H'FFC7	PORT	8	2
Port 9 register	PORT9	8	H'FFC8	PORT	8	2
Port A register	PORTA	8	H'FFC9	PORT	8	2
Port 1 data register	P1DR	8	H'FFD0	PORT	8	2
Port 2 data register	P2DR	8	H'FFD1	PORT	8	2
Port 3 data register	P3DR	8	H'FFD2	PORT	8	2
Port 4 data register	P4DR	8	H'FFD3	PORT	8	2
Port 5 data register	P5DR	8	H'FFD4	PORT	8	2
Port 6 data register	P6DR	8	H'FFD5	PORT	8	2
Port 7 data register	P7DR	8	H'FFD6	PORT	8	2
Port 8 data register	P8DR	8	H'FFD7	PORT	8	2
Port 9 data register	P9DR	8	H'FFD8	PORT	8	2
Port A data register	PADR	8	H'FFD9	PORT	8	2
Timer control register 0	TCR_0	8	H'FFE0	TMR_0	16	2
Timer control register 1	TCR_1	8	H'FFE1	TMR_1	16	2
Timer control/status register 0	TCSR_0	8	H'FFE2	TMR_0	16	2
Timer control/status register 1	TCSR_1	8	H'FFE3	TMR_1	16	2

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access States
Time constant register A0	TCORA_0	8	H'FFE4	TMR_0	16	2
Time constant register A1	TCORA_1	8	H'FFE5	TMR_1	16	2
Time constant register B0	TCORB_0	8	H'FFE6	TMR_0	16	2
Time constant register B1	TCORB_1	8	H'FFE7	TMR_1	16	2
Timer counter 0	TCNT_0	8	H'FFE8	TMR_0	16	2
Timer counter 1	TCNT_1	8	H'FFE9	TMR_1	16	2

17.2 Register Bits

Register bit names of the on-chip peripheral modules are described below.

Each line covers eight bits, and 16-bit and 32-bit registers are shown as 2 or 4 lines, respectively.

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
IFR0	—	—	—	—	—	MODEMN1	MODEMN0	MODEF	USB2
	—	—	—	—	—	—	VBUSMN	VBUSF	
	—	—	—	—	—	—	EP3TR	EP3TS	
	BRST	EP2TR	EP2EMPTY	EP1FULL	SETUPTS	EP0oTS	EP0ITR	EP0ITS	
ISR0	—	—	—	—	—	—	—	MODEF	
	—	—	—	—	—	—	—	VBUSF	
	—	—	—	—	—	—	EP3TR	EP3TS	
	BRST	EP2TR	EP2EMPTY	EP1FULL	SETUPTS	EP0oTS	EP0iTR	EP0iTS	
IER0	—	—	—	—	—	—	—	MODEF	
	—	—	—	—	—	—	—	VBUSF	
	—	—	—	—	—	—	EP3TR	EP3TS	
	BRST	EP2TR	EP2EMPTY	EP1FULL	SETUPTS	EP0oTS	EP0iTR	EP0iTS	
EPSZ0o	D31	D30	D29	D28	D27	D26	D25	D24	
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	
EPSZ1	D31	D30	D29	D28	D27	D26	D25	D24	
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR0i	D31	D30	D29	D28	D27	D26	D25	D24	
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR0o	D31	D30	D29	D28	D27	D26	D25	D24	
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
EPDR0s	D31	D30	D29	D28	D27	D26	D25	D24	USB2
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR1	D31	D30	D29	D28	D27	D26	D25	D24	
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR2	D31	D30	D29	D28	D27	D26	D25	D24	
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR3	D31	D30	D29	D28	D27	D26	D25	D24	
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	
DASTS0	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	EP3DE	EP2DE	—	—	—	EP0IDE	
PKTE0i	D31	D30	D29	D28	D27	D26	D25	D24	
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	
PKTE2	D31	D30	D29	D28	D27	D26	D25	D24	
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	
PKTE3	D31	D30	D29	D28	D27	D26	D25	D24	
	D23	D22	D21	D20	D19	D18	D17	D16	
	D15	D14	D13	D12	D11	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
FCLR0	—	—	—	—	—	—	—	—	USB2
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	EP3CLR	EP2CLR	EP1CLR	—	—	EP0oCLR	EP0iCLR	
EPSTL0	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	EP3STL	EP2STL	EP1STL	EP0STL	
DMA0	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	EP2DMAE	EP1DMAE	
CTRL	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	ASCE	PULLUPE	
SBYCR	SSBY	STS2	STS1	STS0	—	—	—	—	SYSTEM
MSTPCRH	—	—	—	TMRCKSTP	—	—	—	—	
MSTPCRL	SCICKSTP	—	—	USBCKSTP	—	—	—	—	
SYSCR	—	—	INTM1	INTM0	XRST	NMIEG	—	RAME	
MDCR	EXPE	—	—	—	—	MDS2	MDS1	MDS0	
USBSUSP	USUSMONI	USUSFG	USUSFGE	USUSOUT	—	—	—	—	USB2
ACSCR	ABW3	ABW2	ABW1	ABW0	AST3	AST2	AST1	AST0	BSC
CSACR	CSXH3	CSXH2	CSXH1	CSXH0	CSXT3	CSXT2	CSXT1	CSXT0	
WTCR	—	W32	W31	W30	—	W22	W21	W20	
	—	W12	W11	W10	—	W02	W01	W00	
BCR	—	—	—	—	—	—	—	WDBE	
	—	—	—	—	IDLE1	IDLE0	IDLC1	IDLC0	
RDNCR	RDN3	RDN2	RDN1	RDN0	—	—	—	—	
RAMER	—	—	—	—	RAMS	RAM2	RAM1	RAM0	FLASH
DRAMCR	—	RAST	—	CAST	—	—	—	DSET	BSC
	BE	RCDM	DDS	—	—	MXC2	MXC1	MXC0	
DRACCR	—	—	TPC1	TPC0	—	—	RCD1	RCD0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
REFCR	CMF	CMIE	RCW1	RCW0	—	RTCK2	RTCK1	RTCK0	BSC
	RFSHE	—	RLW1	RLW0	SLFRF	TPCS2	TPCS1	TPCS0	
RTCNT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
RTCOR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
TCSR	OVF	WT/$\overline{IT}$	TME	—	RST/$\overline{NMI}$	CKS2	CKS1	CKS0	WDT
TCNT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
FCCS	FWE	—	—	FLER	WEINTE	—	—	SCO	FLASH
FPCS	—	—	—	PPVD	—	—	—	PPVS	
FECS	—	—	—	—	—	—	—	EPVB	
FKEY	K7	K6	K5	K4	K3	K2	K1	K0	
FMATS	MS7	MS6	MS5	MS4	MS3	MS2	MS1	MS0	
FTDAR	TDER	TDA6	TDA5	TDA4	TDA3	TDA2	TDA1	TDA0	
DMSAR_0	—	—	—	—	—	—	—	—	DMAC_0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
DMDAR_0	—	—	—	—	—	—	Bit25	Bit24	
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
DMTCR_0	—	—	—	—	—	—	—	—	
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
DMMDR_0	DA	BEF	DRAKE	TENDE	DREQS	AMS	MDS1	MDS0	
	DIE	IRF	TCEIE	SDIR	DTSIZE	—	LWSIZE	—	
DMACR_0	SAT1	SAT0	SARIE	SARA4	SARA3	SARA2	SARA1	SARA0	
	DAT1	DAT0	DARIE	DARA4	DARA3	DARA2	DARA1	DARA0	
DMSAR_1	—	—	—	—	—	—	—	—	DMAC_1
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
DMDAR_1	—	—	—	—	—	—	—	—	DMAC_1
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
DMTCR_1	—	—	—	—	—	—	—	—	
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
DMMDR_1	DA	BEF	DRAKE	TENDE	DREQS	AMS	MDS1	MDS0	
	DIE	IRF	TCEIE	SDIR	DTSIZE	—	LWSIZE	—	
DMACR_1	SAT1	SAT0	SARIE	SARA4	SARA3	SARA2	SARA1	SARA0	
	DAT1	DAT0	DARIE	DARA4	DARA3	DARA2	DARA1	DARA0	
DMSAR_2	—	—	—	—	—	—	—	—	DMAC_2
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
DMDAR_2	—	—	—	—	—	—	Bit25	Bit24	
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
DMTCR_2	—	—	—	—	—	—	—	—	
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
DMMDR_2	DA	BEF	DRAKE	TENDE	DREQS	AMS	MDS1	MDS0	
	DIE	IRF	TCEIE	SDIR	DTSIZE	—	LWSIZE	—	
DMACR_2	SAT1	SAT0	SARIE	SARA4	SARA3	SARA2	SARA1	SARA0	
	DAT1	DAT0	DARIE	DARA4	DARA3	DARA2	DARA1	DARA0	
DMSAR_3	—	—	—	—	—	—	—	—	DMAC_3
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
DMDAR_3	—	—	—	—	—	—	Bit25	Bit24	DMAC_3
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
DMTCR_3	—	—	—	—	—	—	—	—	
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
DMMDR_3	DA	BEF	DRAKE	TENDE	DREQS	AMS	MDS1	MDS0	
	DIE	IRF	TCEIE	SDIR	DTSIZE	—	LWSIZE	—	
DMACR_3	SAT1	SAT0	SARIE	SARA4	SARA3	SARA2	SARA1	SARA0	
	DAT1	DAT0	DARIE	DARA4	DARA3	DARA2	DARA1	DARA0	
USTCR	EP1DMAE	URCHS1	URCHS0	—	EP2DMAE	UWCHS1	UWCHS0	—	DMAC
	—	—	—	—	—	—	—	—	
IER	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E	INT
ICRA	ICRA7	ICRA6	ICRA5	ICRA4	ICRA3	ICRA2	ICRA1	ICRA0	
ICRB	ICRB7	ICRB6	ICRB5	ICRB4	ICRB3	ICRB2	ICRB1	ICRB0	
ICRC	ICRC7	ICRC6	ICRC5	ICRC4	ICRC3	ICRC2	ICRC1	ICRC0	
ISR	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F	
ISCRH	IRQ7SCB	IRQ7SCA	IRQ6SCB	IRQ6SCA	IRQ5SCB	IRQ5SCA	IRQ4SCB	IRQ4SCA	
ISCRL	IRQ3SCB	IRQ3SCA	IRQ2SCB	IRQ2SCA	IRQ1SCB	IRQ1SCA	IRQ0SCB	IRQ0SCA	
ABRKCR	CMF	—	—	—	—	—	—	BIE	
PBARA	A23	A22	A21	A20	A19	A18	A17	A16	
PBARB	A15	A14	A13	A12	A11	A10	A9	A8	
PBARC	A7	A6	A5	A4	A3	A2	A1	—	
ISSR	ISS7	ISS6	ISS5	ISS4	ISS3	ISS2	ISS1	ISS0	PORT
PFCR1	—	CS1E	CS2E	CS3E	CKOE	ALOE	AMOE	AHOE	
PFCR3	SUSRIF	SUSRIE	—	—	—	—	—	USBSWRST	
P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR	
P2DDR	P27DDR	P26DDR	P25DDR	P24DDR	P23DDR	P22DDR	P21DDR	P20DDR	
P3DDR	P37DDR	P36DDR	P35DDR	P34DDR	P33DDR	P32DDR	P31DDR	P30DDR	
P4DDR	P47DDR	P46DDR	P45DDR	P44DDR	P43DDR	P42DDR	P41DDR	P40DDR	
P5DDR	P57DDR	P56DDR	P55DDR	P54DDR	P53DDR	P52DDR	P51DDR	P50DDR	
P6DDR	P67DDR	P66DDR	P65DDR	P64DDR	P63DDR	P62DDR	P61DDR	P60DDR	

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
P7DDR	P77DDR	P76DDR	P75DDR	P74DDR	P73DDR	P72DDR	P71DDR	P70DDR	PORT
P8DDR	P87DDR	P86DDR	P85DDR	P84DDR	P83DDR	P82DDR	P81DDR	P80DDR	
P9DDR	P97DDR	P96DDR	P95DDR	P94DDR	P93DDR	P92DDR	P91DDR	P90DDR	
PADDR	—	—	—	—	PA3DDR	PA2DDR	PA1DDR	PA0DDR	
SMR	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	—	CKS1	CKS0	SCI
BRR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
SCR	TIE	RIE	TE	RE	—	TEIE	CKE1	CKE0	
TDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
SSR	TDRE	RDRF	ORER	FER	PER	TEND	—	—	
RDR	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	
PORT1	P17	P16	P15	P14	P13	P12	P11	P10	PORT
PORT2	P27	P26	P25	P24	P23	P22	P21	P20	
PORT3	P37	P36	P35	P34	P33	P32	P31	P30	
PORT4	P47	P46	P45	P44	P43	P42	P41	P40	
PORT5	P57	P56	P55	P54	P53	P52	P51	P50	
PORT6	P67	P66	P65	P64	P63	P62	P61	P60	
PORT7	P77	P76	P75	P74	P73	P72	P71	P70	
PORT8	P87	P86	P85	P84	P83	P82	P81	P80	
PORT9	P97	P96	P95	P94	P93	P92	P91	P90	
PORTA	RxDMON	—	—	—	PA3	PA2	PA1	PA0	
P1DR	P17DR	P16DR	P15DR	P14DR	P13DR	P12DR	P11DR	P10DR	
P2DR	P27DR	P26DR	P25DR	P24DR	P23DR	P22DR	P21DR	P20DR	
P3DR	P37DR	P36DR	P35DR	P34DR	P33DR	P32DR	P31DR	P30DR	
P4DR	P47DR	P46DR	P45DR	P44DR	P43DR	P42DR	P41DR	P40DR	
P5DR	P57DR	P56DR	P55DR	P54DR	P53DR	P52DR	P51DR	P50DR	
P6DR	P67DR	P66DR	P65DR	P64DR	P63DR	P62DR	P61DR	P60DR	
P7DR	P77DR	P76DR	P75DR	P74DR	P73DR	P72DR	P71DR	P70DR	
P8DR	P87DR	P86DR	P85DR	P84DR	P83DR	P82DR	P81DR	P80DR	
P9DR	P97DR	P96DR	P95DR	P94DR	P93DR	P92DR	P91DR	P90DR	
PADR	—	—	—	—	PA3DR	PA2DR	PA1DR	PA0DR	
TCR_0	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_0
TCR_1	CMIEB	CMIEA	OVIE	CCLR1	CCLR0	CKS2	CKS1	CKS0	TMR_1
TCSR_0	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	TMR_0

Register Abbreviation	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
TCSR_1	CMFB	CMFA	OVF	—	OS3	OS2	OS1	OS0	TMR_1
TCORA_0	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	TMR_0
TCORA_1	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	TMR_1
TCORB_0	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	TMR_0
TCORB_1	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	TMR_1
TCNT_0	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	TMR_0
TCNT_1	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	TMR_1

17.3 Register States in Each Operating Mode

Register Abbreviation	Power-on Reset	Normal Operation	Sleep	Module Stop	Software Standby	Hardware Standby	Module
IFR0	Initialized	—	—	—	—	Initialized	USB2
ISR0	Initialized	—	—	—	—	Initialized	
IER0	Initialized	—	—	—	—	Initialized	
EPSZ0o	Initialized	—	—	—	—	Initialized	
EPSZ1	Initialized	—	—	—	—	Initialized	
EPDR0i	Initialized	—	—	—	—	Initialized	
EPDR0o	Initialized	—	—	—	—	Initialized	
EPDR0s	Initialized	—	—	—	—	Initialized	
EPDR1	Initialized	—	—	—	—	Initialized	
EPDR2	Initialized	—	—	—	—	Initialized	
EPDR3	Initialized	—	—	—	—	Initialized	
DASTS0	Initialized	—	—	—	—	Initialized	
PKTE0i	Initialized	—	—	—	—	Initialized	
PKTE2	Initialized	—	—	—	—	Initialized	
PKTE3	Initialized	—	—	—	—	Initialized	
FCLR0	Initialized	—	—	—	—	Initialized	
EPSTL0	Initialized	—	—	—	—	Initialized	
DMA0	Initialized	—	—	—	—	Initialized	
CTRL	Initialized	—	—	—	—	Initialized	
SBYCR	Initialized	—	—	—	—	Initialized	SYSTEM
MSTPCRH	Initialized	—	—	—	—	Initialized	
MSTPCRL	Initialized	—	—	—	—	Initialized	
SYSCR	Initialized	—	—	—	—	Initialized	
MDCR	Initialized	—	—	—	—	Initialized	
USBSUSP	Initialized	—	—	—	—	Initialized	USB2
ACSCR	Initialized	—	—	—	—	Initialized	BSC
CSACR	Initialized	—	—	—	—	Initialized	
WTCR	Initialized	—	—	—	—	Initialized	
BCR	Initialized	—	—	—	—	Initialized	
RDNCR	Initialized	—	—	—	—	Initialized	
RAMER	Initialized	—	—	—	—	Initialized	
DRAMCR	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Power-on Reset	Normal Operation	Sleep	Module Stop	Software Standby	Hardware Standby	Module
DRACCR	Initialized	—	—	—	—	Initialized	BSC
REFCR	Initialized	—	—	—	—	Initialized	
RTCNT	Initialized	—	—	—	—	Initialized	
RTCOR	Initialized	—	—	—	—	Initialized	
TCSR	Initialized	—	—	—	—	Initialized	WDT
TCNT	Initialized	—	—	—	—	Initialized	
FCCS	Initialized	—	—	—	—	Initialized	FLASH
FPCS	Initialized	—	—	—	—	Initialized	
FECS	Initialized	—	—	—	—	Initialized	
FKEY	Initialized	—	—	—	—	Initialized	
FMATS	Initialized	—	—	—	—	Initialized	
FTDAR	Initialized	—	—	—	—	Initialized	
DMSAR_0	Initialized	—	—	—	—	Initialized	DMAC_0
DMDAR_0	Initialized	—	—	—	—	Initialized	
DMTCR_0	Initialized	—	—	—	—	Initialized	
DMMDR_0	Initialized	—	—	—	—	Initialized	
DMACR_0	Initialized	—	—	—	—	Initialized	
DMSAR_1	Initialized	—	—	—	—	Initialized	DMAC_1
DMDAR_1	Initialized	—	—	—	—	Initialized	
DMTCR_1	Initialized	—	—	—	—	Initialized	
DMMDR_1	Initialized	—	—	—	—	Initialized	
DMACR_1	Initialized	—	—	—	—	Initialized	
DMSAR_2	Initialized	—	—	—	—	Initialized	DMAC_2
DMDAR_2	Initialized	—	—	—	—	Initialized	
DMTCR_2	Initialized	—	—	—	—	Initialized	
DMMDR_2	Initialized	—	—	—	—	Initialized	
DMACR_2	Initialized	—	—	—	—	Initialized	
DMSAR_3	Initialized	—	—	—	—	Initialized	DMAC_3
DMDAR_3	Initialized	—	—	—	—	Initialized	
DMTCR_3	Initialized	—	—	—	—	Initialized	
DMMDR_3	Initialized	—	—	—	—	Initialized	
DMACR_3	Initialized	—	—	—	—	Initialized	

Register Abbreviation	Power-on Reset	Normal Operation	Sleep	Module Stop	Software Standby	Hardware Standby	Module
USTCR	Initialized	—	—	—	—	Initialized	DMAC
IER	Initialized	—	—	—	—	Initialized	INT
ICRC	Initialized	—	—	—	—	Initialized	
ICRB	Initialized	—	—	—	—	Initialized	
ICRA	Initialized	—	—	—	—	Initialized	
ISR	Initialized	—	—	—	—	Initialized	
ISCRH	Initialized	—	—	—	—	Initialized	
ISCRL	Initialized	—	—	—	—	Initialized	
ABRKCR	Initialized	—	—	—	—	Initialized	
PBARA	Initialized	—	—	—	—	Initialized	
PBARB	Initialized	—	—	—	—	Initialized	
PBARC	Initialized	—	—	—	—	Initialized	
ISSR	Initialized	—	—	—	—	Initialized	
PFCR1	Initialized	—	—	—	—	Initialized	PORT
PFCR3	Initialized	—	—	—	—	Initialized	
P1DDR	Initialized	—	—	—	—	Initialized	
P2DDR	Initialized	—	—	—	—	Initialized	
P3DDR	Initialized	—	—	—	—	Initialized	
P4DDR	Initialized	—	—	—	—	Initialized	
P5DDR	Initialized	—	—	—	—	Initialized	
P6DDR	Initialized	—	—	—	—	Initialized	
P7DDR	Initialized	—	—	—	—	Initialized	
P8DDR	Initialized	—	—	—	—	Initialized	
P9DDR	Initialized	—	—	—	—	Initialized	
PADDR	Initialized	—	—	—	—	Initialized	
SMR	Initialized	—	—	—	—	Initialized	SCI
BRR	Initialized	—	—	—	—	Initialized	
SCR	Initialized	—	—	—	—	Initialized	
TDR	Initialized	—	—	Initialized	Initialized	Initialized	
SSR	Initialized	—	—	Initialized	Initialized	Initialized	
RDR	Initialized	—	—	Initialized	Initialized	Initialized	
PORT1	—	—	—	—	—	—	PORT
PORT2	—	—	—	—	—	—	

Register Abbreviation	Power-on Reset	Normal Operation	Sleep	Module Stop	Software Standby	Hardware Standby	Module
PORT3	—	—	—	—	—	—	PORT
PORT4	—	—	—	—	—	—	
PORT5	—	—	—	—	—	—	
PORT6	—	—	—	—	—	—	
PORT7	—	—	—	—	—	—	
PORT8	—	—	—	—	—	—	
PORT9	—	—	—	—	—	—	
PORTA	—	—	—	—	—	—	
P1DR	Initialized	—	—	—	—	Initialized	
P2DR	Initialized	—	—	—	—	Initialized	
P3DR	Initialized	—	—	—	—	Initialized	
P4DR	Initialized	—	—	—	—	Initialized	
P5DR	Initialized	—	—	—	—	Initialized	
P6DR	Initialized	—	—	—	—	Initialized	
P7DR	Initialized	—	—	—	—	Initialized	
P8DR	Initialized	—	—	—	—	Initialized	
P9DR	Initialized	—	—	—	—	Initialized	
PADR	Initialized	—	—	—	—	Initialized	
TCR_0	Initialized	—	—	—	—	Initialized	TMR_0
TCR_1	Initialized	—	—	—	—	Initialized	TMR_1
TCSR_0	Initialized	—	—	—	—	Initialized	TMR_0
TCSR_1	Initialized	—	—	—	—	Initialized	TMR_1
TCORA_0	Initialized	—	—	—	—	Initialized	TMR_0
TCORA_1	Initialized	—	—	—	—	Initialized	TMR_1
TCORB_0	Initialized	—	—	—	—	Initialized	TMR_0
TCORB_1	Initialized	—	—	—	—	Initialized	TMR_1
TCNT_0	Initialized	—	—	—	—	Initialized	TMR_0
TCNT_1	Initialized	—	—	—	—	Initialized	TMR_1

Note: — is not initialized.

Section 18 Electrical Characteristics

18.1 Absolute Maximum Ratings

Table 18.1 lists the absolute maximum ratings.

Table 18.1 Absolute Maximum Ratings

Item	Symbol	Value	Unit
Power supply voltage	V_{CC}*[2]	–0.3 to +4.3	V
Input voltage	V_{in}	–0.3 to V_{CC} +0.3	V
Operating temperature	T_{opr}	Regular specifications: –20 to +75*[1]	°C
		Wide-range specifications: –40 to +85*[1]	°C
Storage temperature	T_{stg}	–55 to +125	°C

Caution: Permanent damage to the LSI may result if absolute maximum ratings are exceeded.

Notes: 1. Ranges of operating temperature when flash memory is programmed/erased:
Regular specifications: 0 to +75°C
Wide-range specifications: 0 to +85°C

2. Do not apply the power supply voltage to the VCL pin. If applied, permanent damage to the LSI may result. Connect the external capacitor between this pin and GND.

18.2 DC Characteristics

Table 18.2 DC Characteristics (1)

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, T_a = –20°C to +75°

Item		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Schmitt trigger input voltage	P34*[1], P35*[1], P52*[1], P53*[1], P56*[1], P57*[1], port 7*[1], PA0*[1], PA1*[1]	VT^-	$V_{CC} \times 0.2$	—	—	V	
		VT^+	—	—	$V_{CC} \times 0.7$	V	
		$VT^+ - VT^-$	$V_{CC} \times 0.05$	—	—	V	
Input high voltage	$\overline{STBY}$, MD2, MD1	V_{IH}	$V_{CC} \times 0.9$	—	$V_{CC} + 0.3$	V	
	$\overline{RES}$, NMI, FWE		$V_{CC} \times 0.9$	—	$V_{CC} + 0.3$	V	
	EXTAL		$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$	V	
	Port 3*[2], port 5*[2], port 6, port 7*[2], port 8, port A*[2], port B		2.2	—	$V_{CC} + 0.3$	V	
Input low voltage	$\overline{RES}$, $\overline{STBY}$, $\overline{MD2}$, MD1, FWE	V_{IL}	–0.3	—	$V_{CC} \times 0.1$	V	
	NMI, EXTAL		–0.3	—	$V_{CC} \times 0.2$	V	
	Ports 3 to 8, port A*[2], port B		–0.3	—	$V_{CC} \times 0.2$	V	
Output high voltage	All output pins	V_{OH}	V_{CC} – 0.5	—	—	V	I_{OH} = –200 µA
			V_{CC} – 1.0	—	—	V	I_{OH} = –1 mA
Output low voltage	All output pins	V_{OL}	—	—	0.4	V	I_{OL} = 1.6 mA
Input leakage current	$\overline{RES}$	$\lvert I_{in} \rvert$	—	—	10.0	µA	V_{in} = 0.5 to V_{CC} - 0.5 V
	$\overline{STBY}$, NMI, MD2, MD1, FWE		—	—	1.0	µA	

Notes: 1. When used as $\overline{IRQ0}$ to $\overline{IRQ7}$.
2. When used as other than $\overline{IRQ0}$ to $\overline{IRQ7}$.

Table 18.3 DC Characteristics (2)

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, T_a = –20°C to +75°C

Item		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Three-state leakage current (off state)	Ports 1 to 8, port A, port B	$\lvert I_{TSI} \rvert$	—	—	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5 V
Input capacitance	$\overline{RES}$	C_{in}	—	—	10	pF	V_{in} = 0 V
	NMI		—	—	10	pF	f = 1 MHz
	All input pins except $\overline{RES}$ and NMI		—	—	10	pF	T_a = 25°C
Current consumption*[1]	Normal operation	I_{CC}*[3]	—	50	65	mA	f = 33 MHz
	Sleep mode		—	38	48	mA	f = 33 MHz
	Standby mode*[2]		—	30	90	µA	$T_a \leq$ 50°C
			—	—	120	µA	50°C < T_a
RAM standby voltage		V_{RAM}	3.0	—	—	V	
Vcc start power supply		$V_{CCSTART}$	—	0	0.8	V	*[4]
Vcc rising gradient		SV_{CC}	—	—	20	ms/V	*[4]

Notes: 1. Current consumption values are for V_{IH}min = V_{CC} – 0.2 V and V_{IL}max = 0.2 V with all output pins unloaded.
2. The values are for V_{CC} = 3.0 V, V_{IH}min = $V_{CC} \times 0.9$, and V_{IL}max = 0.3 V.
3. I_{CC} depends on V_{CC} and f as follows:
   I_{CC}max = 6.5 (mA) + 0.49 (mA/(MHz × V)) × V_{CC} × f (normal operation)
   I_{CC}max = 6.5 (mA) + 0.35 (mA/(MHz × V)) × V_{CC} × f (sleep mode)
4. These values are measured when the $\overline{RES}$ pin is low.

Table 18.4 Permissible Output Currents

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, T_a = –20°C to +75°C

Item		Symbol	Min.	Typ.	Max.	Unit
Permissible output low current (per pin)	Ports 1 and 4	I_{OL}	—	—	1.6	mA
	All output pins execept ports 1 and 4		—	—	2	
Permissible output low current (total)	Total of all output pins	ΣI_{OL}	—	—	60	mA

Item		Symbol	Min.	Typ.	Max.	Unit
Permissible output high current (per pin)	All output pins	$-I_{OH}$	—	—	2	mA
Permissible output high current (total)	Total of all output pins	$\Sigma-I_{OH}$	—	—	30	mA

Caution: 1. To protect the LSI's reliability, do not exceed the output current values in table 18.4.
2. To drive the Darlington transistor directly, inset a current-limit resister between the LSI and the transistor, as shown in figure 18.1.

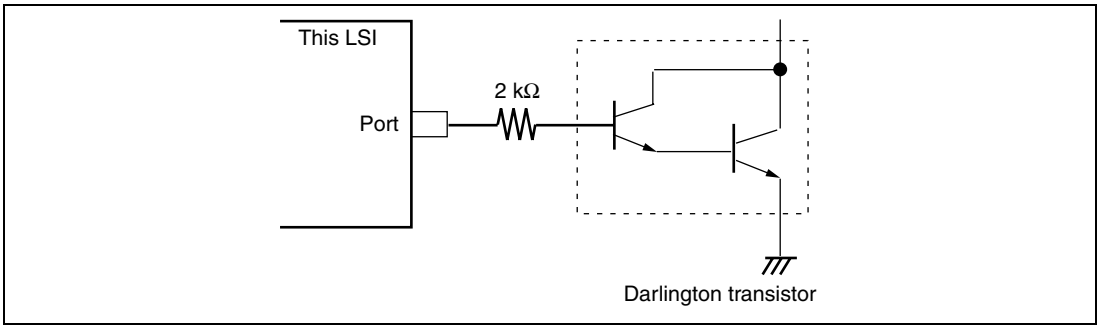


Figure 18.1 Sample of Dalington Transistor Drive Circuit

18.3 AC Characteristics

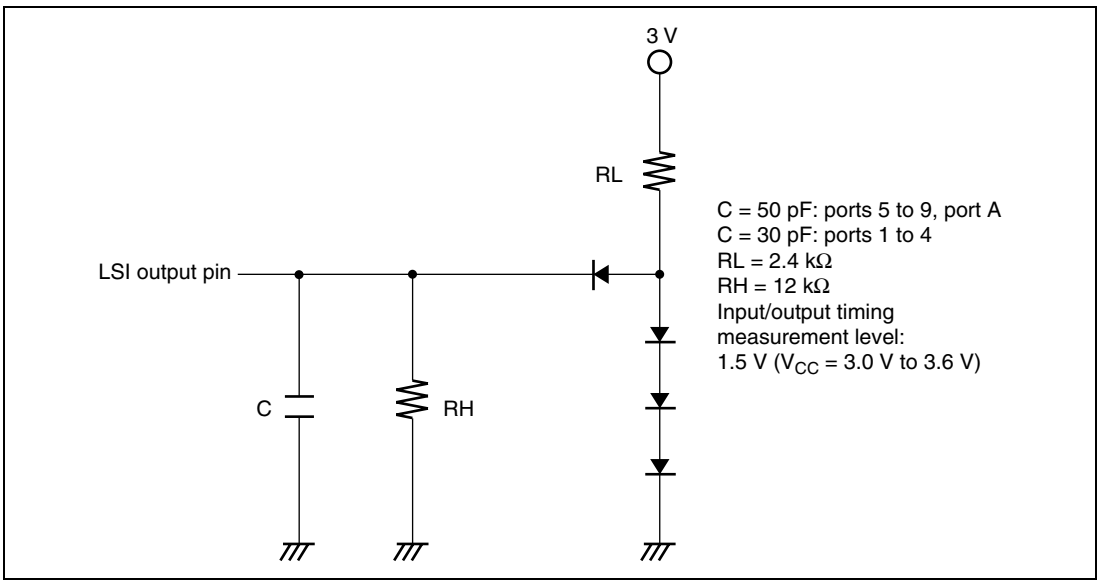


Figure 18.2 Output Load Circuit

18.3.1 Clock Timing

Table 18.5 Clock Timing

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, ϕ = 10 MHz to 33 MHz, T_a = –20°C to +75°C

Item	Symbol	Min.	Max.	Unit		Test Conditions
Clock cycle time	t_{cyc}	30.3	100	ns		Figure 18.3
Clock pulse high width	t_{CH}	10	—	ns		
Clock pulse low width	t_{CL}	10	—	ns		
Clock rise time	t_{Cr}	—	5	ns		
Clock fall time	t_{Cf}	—	5	ns		
Reset oscillation stabilization time (crystal)	t_{OSC1}	10	—	ms		Figure 18.4
Software standby oscillation stabilization time (crystal)	t_{OSC2}	8	—	ms	ϕ < 30 MHz	Figure 18.5
		3.9	—	ms	$\phi \geq$ 30 MHz	
External clock output delay stabilization time	t_{DEXT}	500	—	µs		Figure 18.4

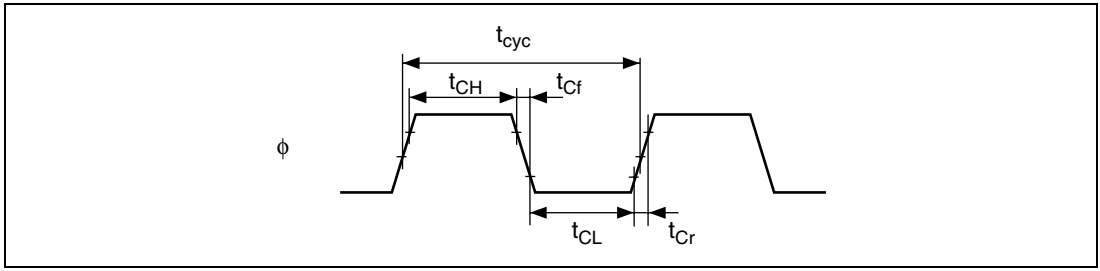


Figure 18.3 System Clock Timing

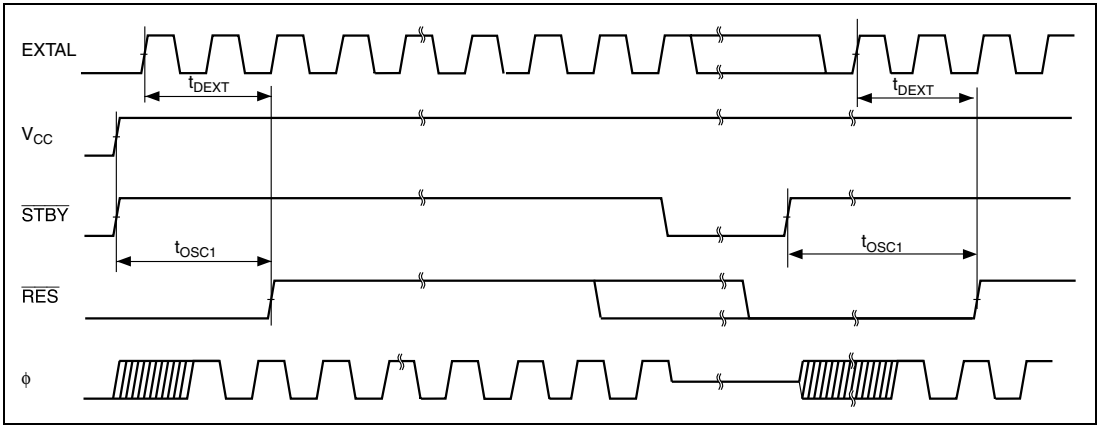


Figure 18.4 Oscillation Stabilization Timing (1)

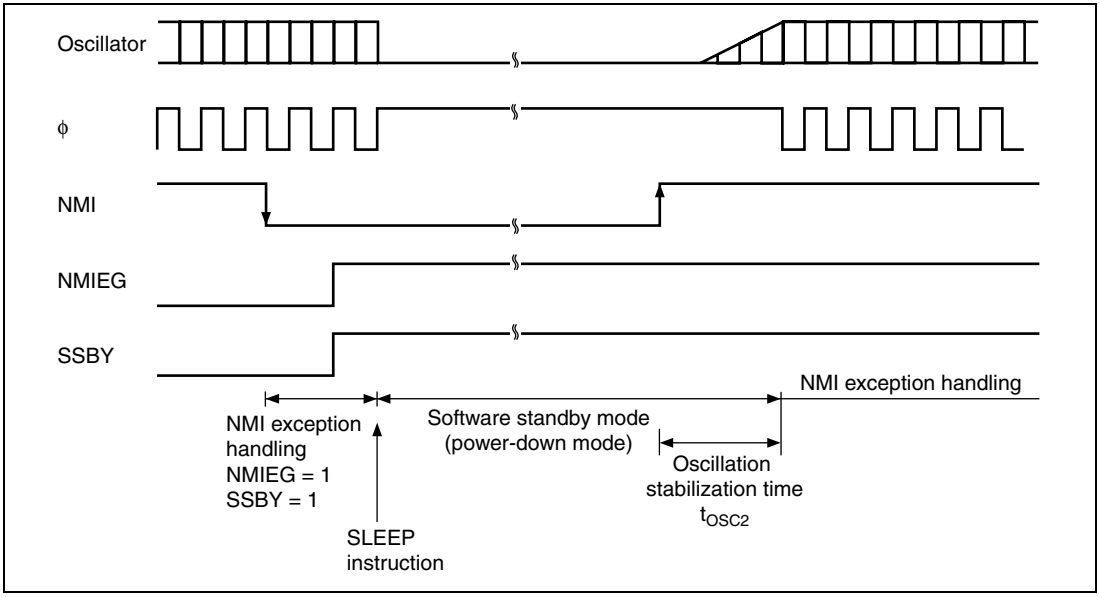


Figure 18.5 Oscillation Stabilization Timing (2)

18.3.2 Control Signal Timing

Table 18.6 Control Signal Timing

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, ϕ = 10 MHz to 33 MHz, T_a = –20°C to +75°C (regular specifications), T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{RES}$ setup time	t_{RESS}	200	—	ns	Figure 18.6
$\overline{RES}$ pulse width	t_{RESW}	20	—	t_{cyc}	
NMI setup time	t_{NMIS}	150	—	ns	Figure 18.7
NMI hold time	t_{NMIH}	10	—		
NMI pulse width (in recovery from software standby mode)	t_{NMIW}	200	—		
$\overline{IRQ}$ setup time	t_{IRQS}	150	—	ns	
$\overline{IRQ}$ hold time	t_{IRQH}	10	—		
$\overline{IRQ}$ pulse width (in recovery from software standby mode)	t_{IRQW}	200	—		

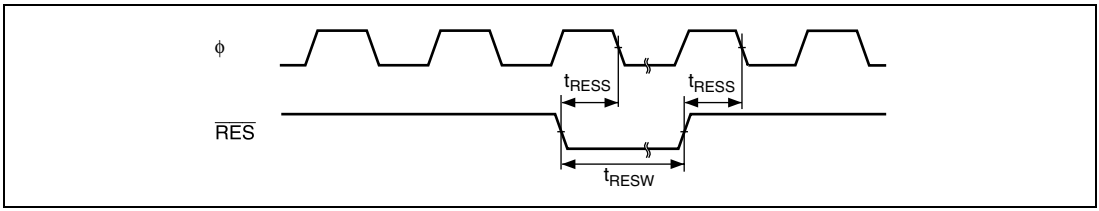


Figure 18.6 Reset Input Timing

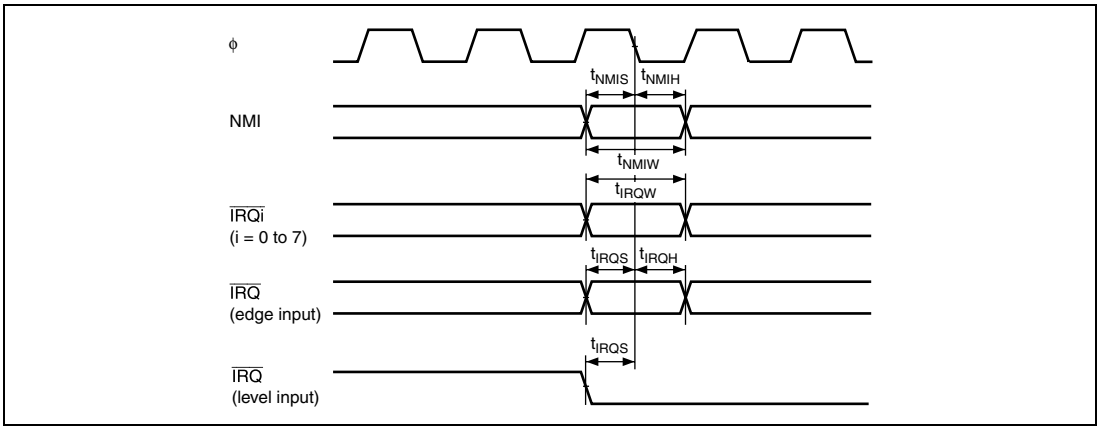


Figure 18.7 Interrupt Input Timing

18.3.3 Bus Timing

Table 18.7 Bus Timing (1)

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, ϕ = 10 MHz to 33 MHz, T_a = –20°C to +75°C

Item	Symbol	Min.	Max.	Unit	Test Conditions
Address delay time	TA_D	—	20	ns	Figures 18.8 to 18.17
Address setup time 1	TA_{S1}	$0.5 \times t_{cyc} - 13$	—	ns	
Address setup time 2	TA_{S2}	$1.0 \times t_{cyc} - 13$	—	ns	
Address setup time 3	TA_{S3}	$1.5 \times t_{cyc} - 13$	—	ns	
Address setup time 4	TA_{S4}	$2.0 \times t_{cyc} - 13$	—	ns	
Address hold time 1	TA_{H1}	$0.5 \times t_{cyc} - 8$	—	ns	
Address hold time 2	TA_{H2}	$1.0 \times t_{cyc} - 8$	—	ns	
Address hold time 3	TA_{H3}	$1.5 \times t_{cyc} - 8$	—	ns	
$\overline{CS}$ delay time 1	t_{CSD1}	—	15	ns	
$\overline{CS}$ delay time 2	t_{CSD2}	—	15	ns	
$\overline{CS}$ delay time 3	t_{CSD3}	—	20	ns	
$\overline{AS}$ delay time	TA_{SD}	—	15	ns	
$\overline{RD}$ delay time 1	t_{RSD1}	—	15	ns	
$\overline{RD}$ delay time 2	t_{RSD2}	—	15	ns	
Read data setup time 1	t_{RDS1}	15	—	ns	
Read data setup time 2	t_{RDS2}	15	—	ns	
Read data hold time 1	t_{RDH1}	0	—	ns	
Read data hold time 2	t_{RDH2}	0	—	ns	
Read data access time 1	T_{AC1}	—	$1.0 \times t_{cyc} - 25$	ns	
Read data access time 2	T_{AC2}	—	$1.5 \times t_{cyc} - 25$	ns	
Read data access time 3	T_{AC3}	—	$2.0 \times t_{cyc} - 25$	ns	
Read data access time 4	T_{AC4}	—	$2.5 \times t_{cyc} - 25$	ns	
Read data access time 5	T_{AC5}	—	$1.0 \times t_{cyc} - 25$	ns	
Read data access time 6	T_{AC6}	—	$2.0 \times t_{cyc} - 25$	ns	
Read data access time 7	T_{AC7}	—	$4.0 \times t_{cyc} - 25$	ns	
Read data access time 8	T_{AC8}	—	$3.0 \times t_{cyc} - 25$	ns	
Address read data access time 2	T_{AA2}	—	$1.5 \times t_{cyc} - 25$	ns	
Address read data access time 3	T_{AA3}	—	$2.0 \times t_{cyc} - 25$	ns	
Address read data access time 4	T_{AA4}	—	$2.5 \times t_{cyc} - 25$	ns	
Address read data access time 5	T_{AA5}	—	$3.0 \times t_{cyc} - 25$	ns	

Table 18.8 Bus Timing (2)

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, ϕ = 10 MHz to 33 MHz, T_a = –20°C to +75°C (regular specifications), T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{WR}$ delay time 1	t_{WRD1}	—	15	ns	Figures 18.8 to 18.19
$\overline{WR}$ delay time 2	t_{WRD2}	—	15	ns	
$\overline{WR}$ pulse width 1	t_{WSW1}	$1.0 \times t_{cyc} - 13$	—	ns	
$\overline{WR}$ pulse width 2	t_{WSW2}	$1.5 \times t_{cyc} - 13$	—	ns	
Write data delay time	t_{WDD}	—	20	ns	
Write data setup time 1	t_{WDS1}	$0.5 \times t_{cyc} - 13$	—	ns	
Write data setup time 2	t_{WDS2}	$1.0 \times t_{cyc} - 13$	—	ns	
Write data setup time 3	t_{WDS3}	$1.5 \times t_{cyc} - 13$	—	ns	
Write data hold time 1	t_{WDH1}	$0.5 \times t_{cyc} - 8$	—	ns	
Write data hold time 2	t_{WDH2}	$1.0 \times t_{cyc} - 8$	—	ns	
Write data hold time 3	t_{WDH3}	$1.5 \times t_{cyc} - 8$	—	ns	
Write command setup time 1	t_{WCS1}	$0.5 \times t_{cyc} - 10$	—	ns	
Write command setup time 2	t_{WCS2}	$1.0 \times t_{cyc} - 10$	—	ns	
Write command hold time 1	t_{WCH1}	$0.5 \times t_{cyc} - 10$	—	ns	
Write command hold time 2	t_{WCH2}	$1.0 \times t_{cyc} - 10$	—	ns	
Read command setup time 1	t_{RCS1}	$1.5 \times t_{cyc} - 10$	—	ns	
Read command setup time 2	t_{RCS2}	$2.0 \times t_{cyc} - 10$	—	ns	
Read command hold time	t_{RCH}	$0.5 \times t_{cyc} - 10$	—	ns	
$\overline{CAS}$ delay time 1	t_{CASD1}	—	15	ns	
$\overline{CAS}$ delay time 2	t_{CASD2}	—	15	ns	
$\overline{CAS}$ setup time 1	t_{CSR1}	$0.5 \times t_{cyc} - 10$	—	ns	
$\overline{CAS}$ setup time 2	t_{CSR2}	$1.5 \times t_{cyc} - 10$	—	ns	
$\overline{CAS}$ pulse width 1	t_{CASW1}	$1.0 \times t_{cyc} - 20$	—	ns	
$\overline{CAS}$ pulse width 2	t_{CASW2}	$1.5 \times t_{cyc} - 20$	—	ns	
$\overline{CAS}$ precharge time 1	t_{CPW1}	$1.0 \times t_{cyc} - 20$	—	ns	
$\overline{CAS}$ precharge time 2	t_{CPW2}	$1.5 \times t_{cyc} - 20$	—	ns	
$\overline{OE}$ delay time 1	t_{OED1}	—	15	ns	
$\overline{OE}$ delay time 2	t_{OED2}	—	15	ns	
Precharge time 1	t_{PCH1}	$1.0 \times t_{cyc} - 20$	—	ns	
Precharge time 2	t_{PCH2}	$1.5 \times t_{cyc} - 20$	—	ns	
Self-refresh precharge time 1	t_{RPS1}	$1.5 \times t_{cyc} - 20$	—	ns	
Self-refresh precharge time 2	t_{RPS2}	$2.0 \times t_{cyc} - 20$	—	ns	

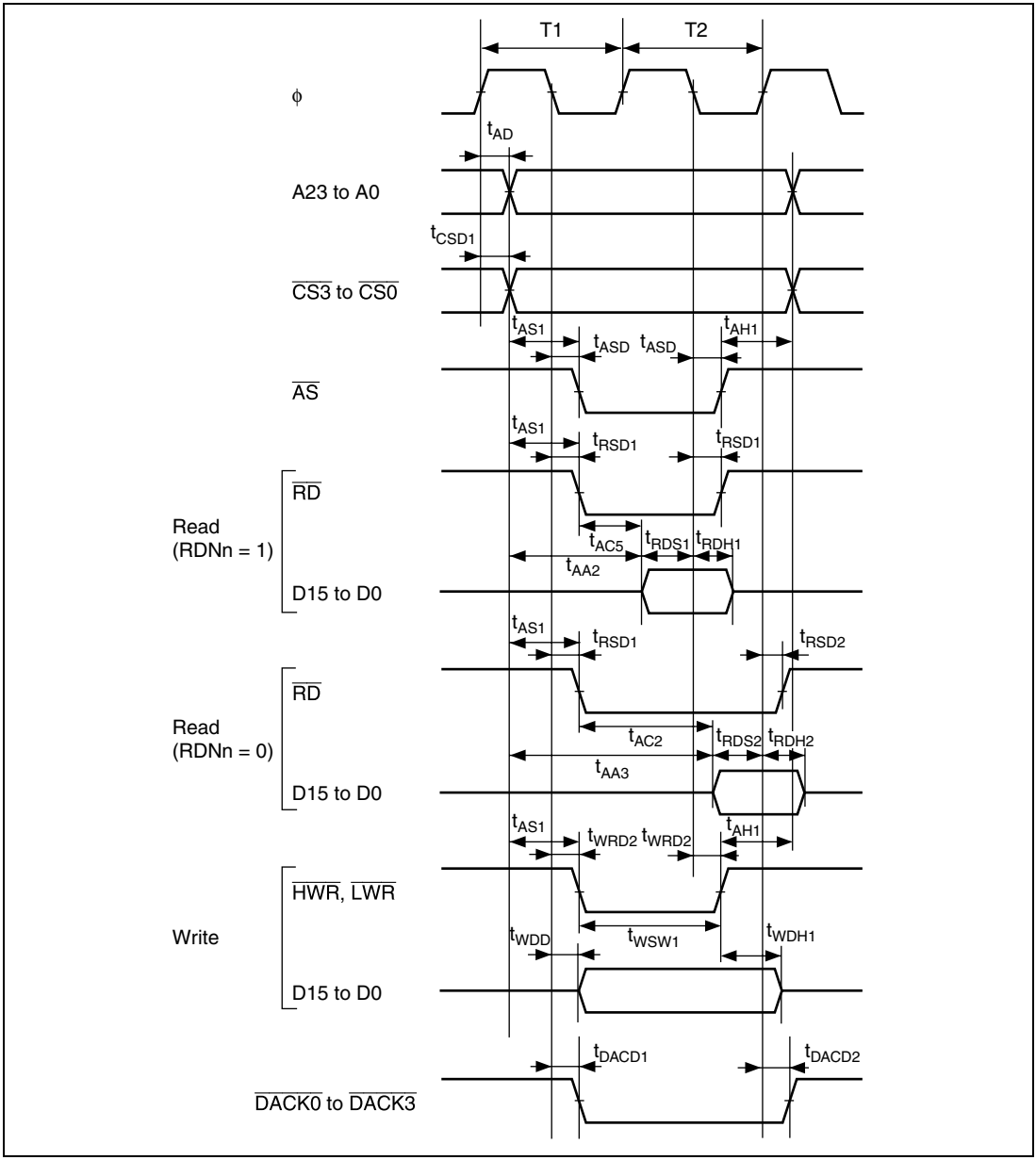


Figure 18.8 Basic Bus Timing: Two-State Access

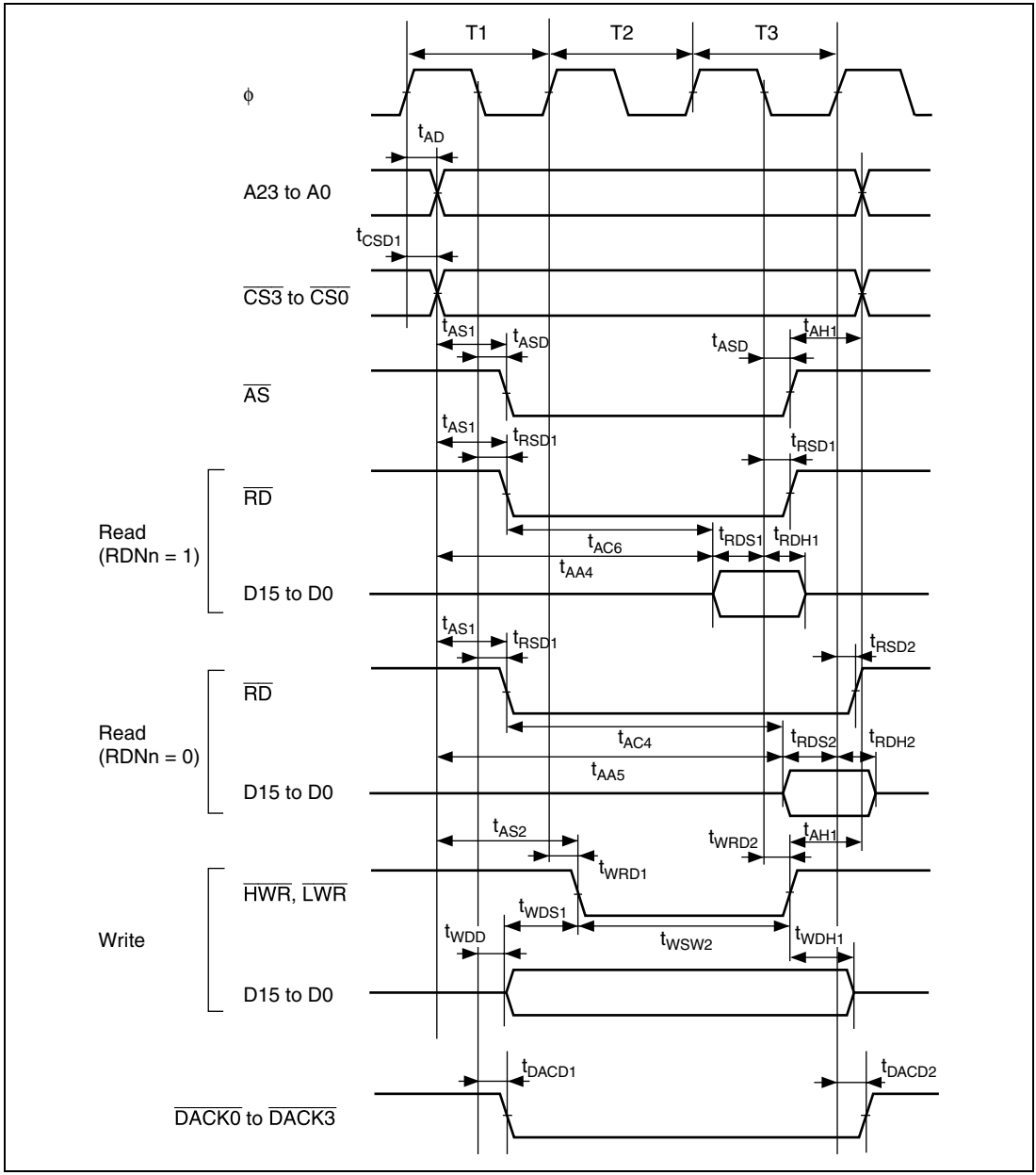


Figure 18.9 Basic Bus Timing: Three-State Access

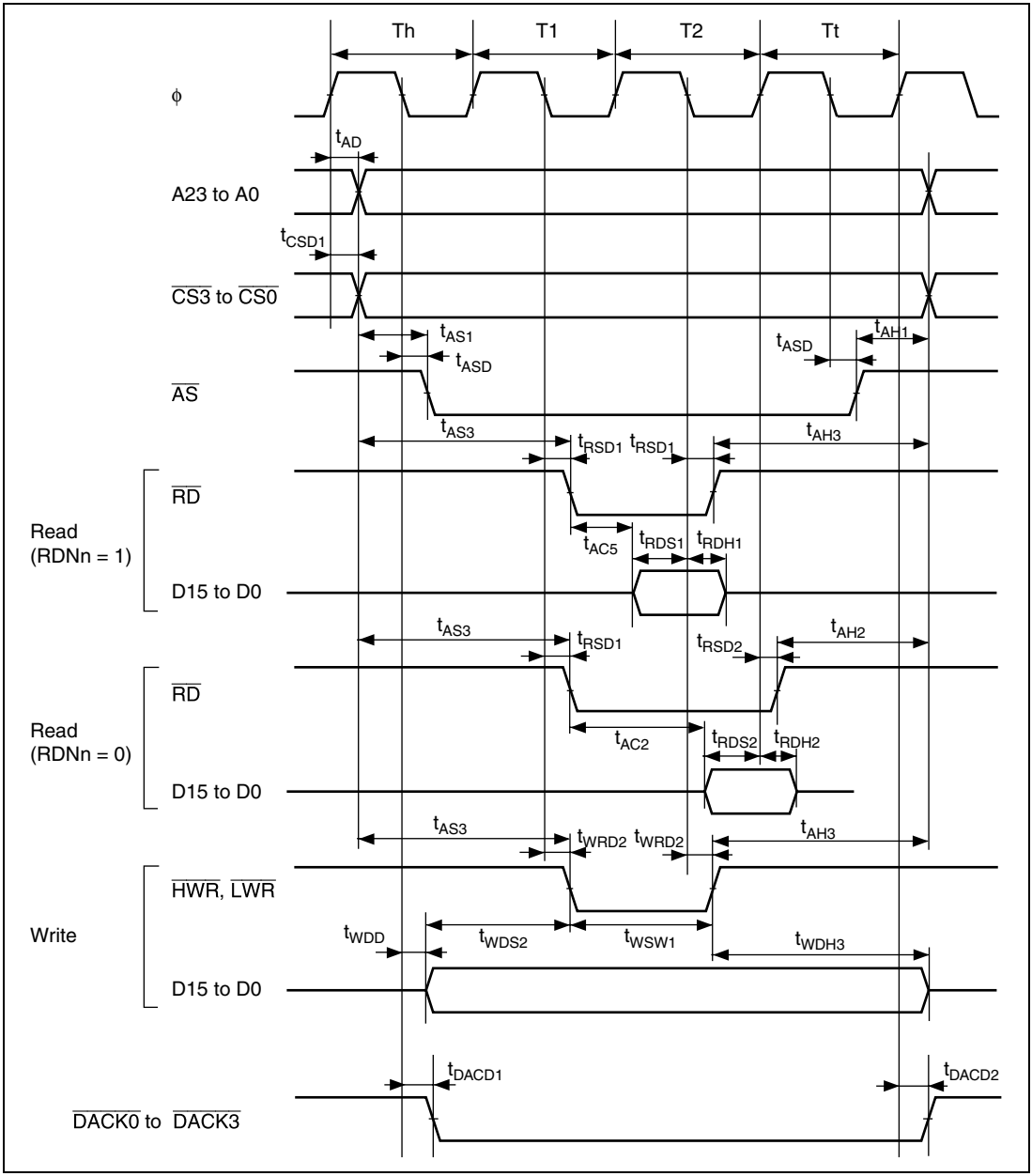


Figure 18.10 Basic Bus Timing: Two-State Access ($\overline{CS}$ Assertion Period Extended)

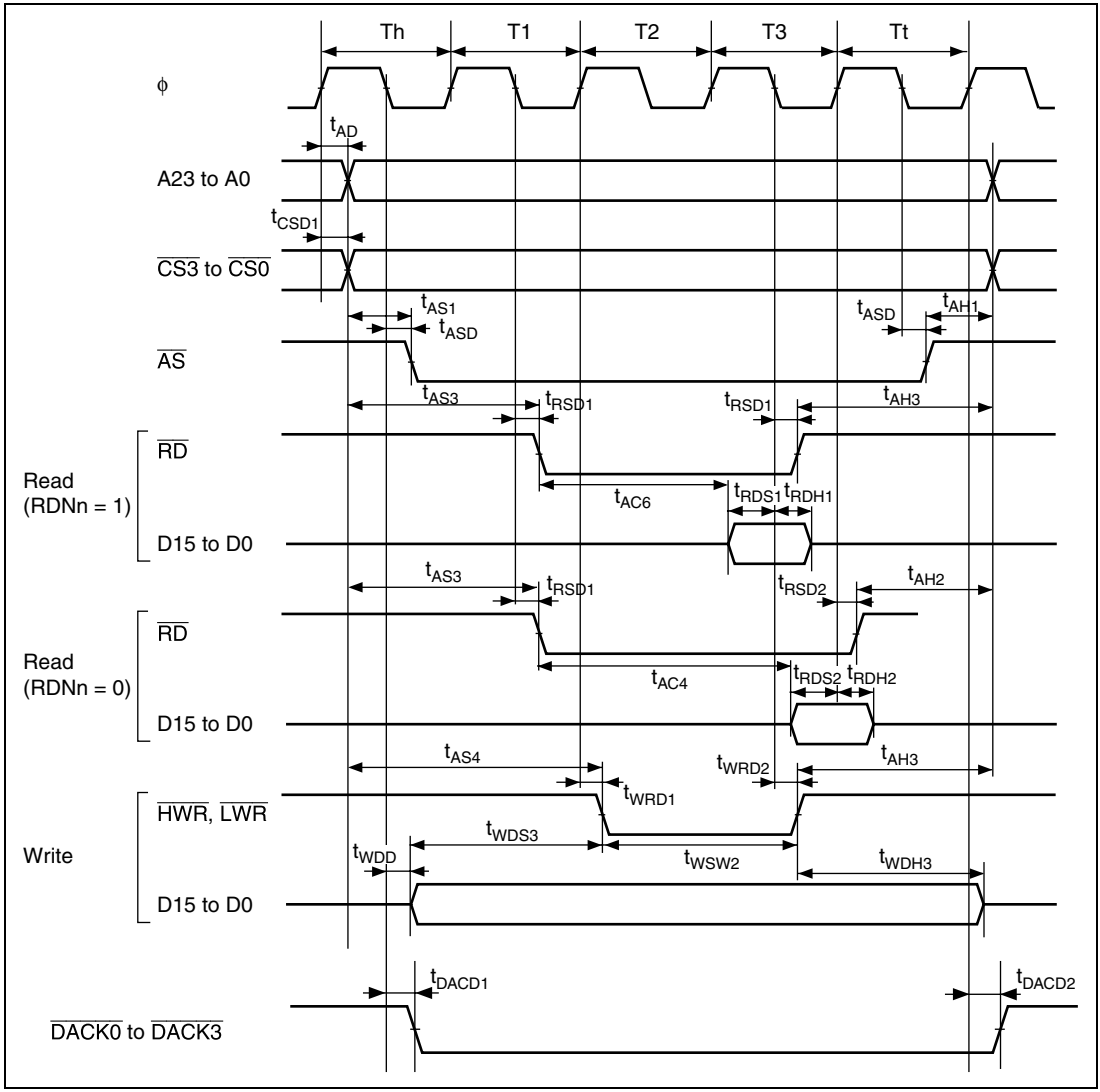


Figure 18.11 Basic Bus Timing: Three-State Access ($\overline{CS}$ Assertion Period Extended)

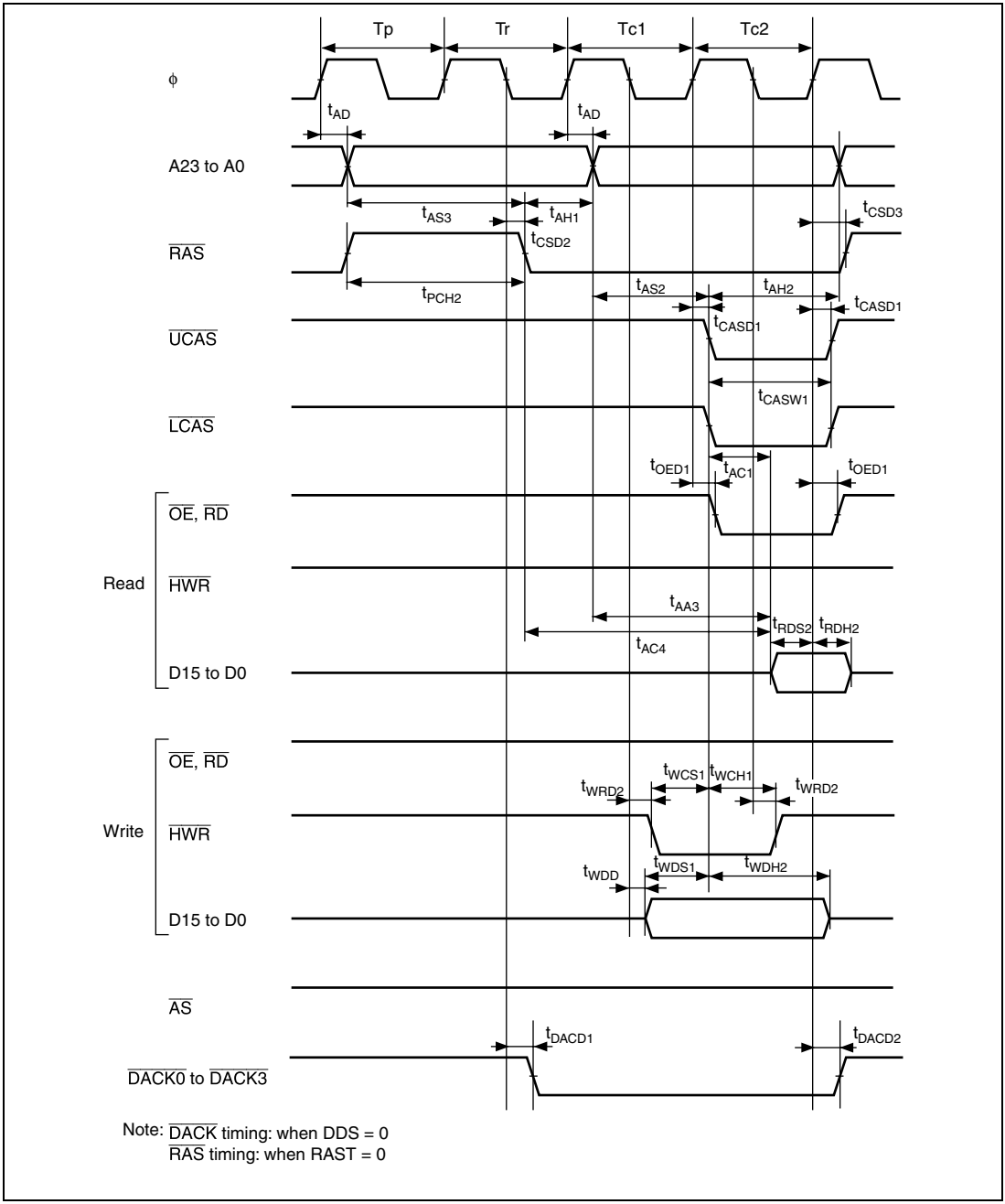
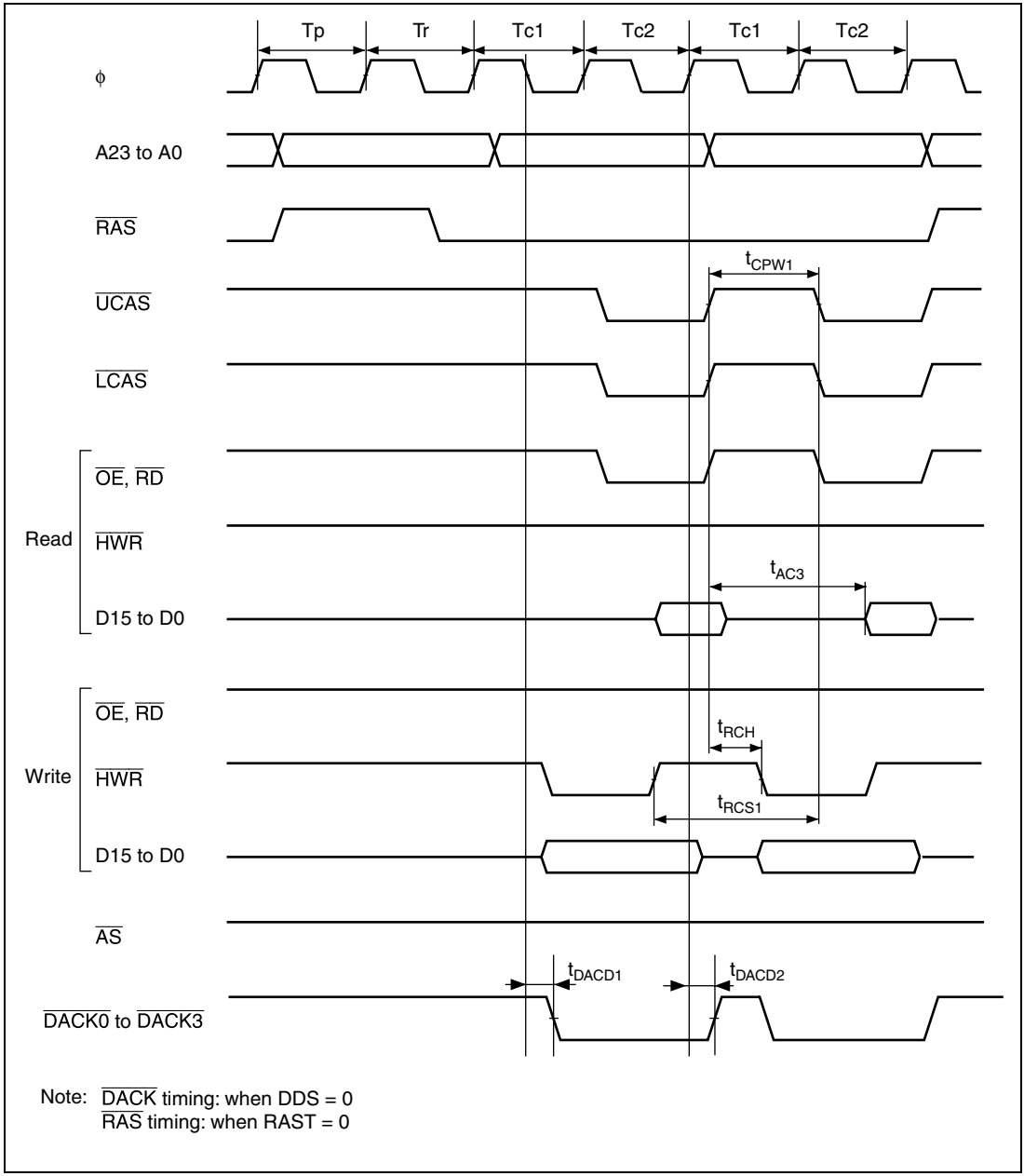
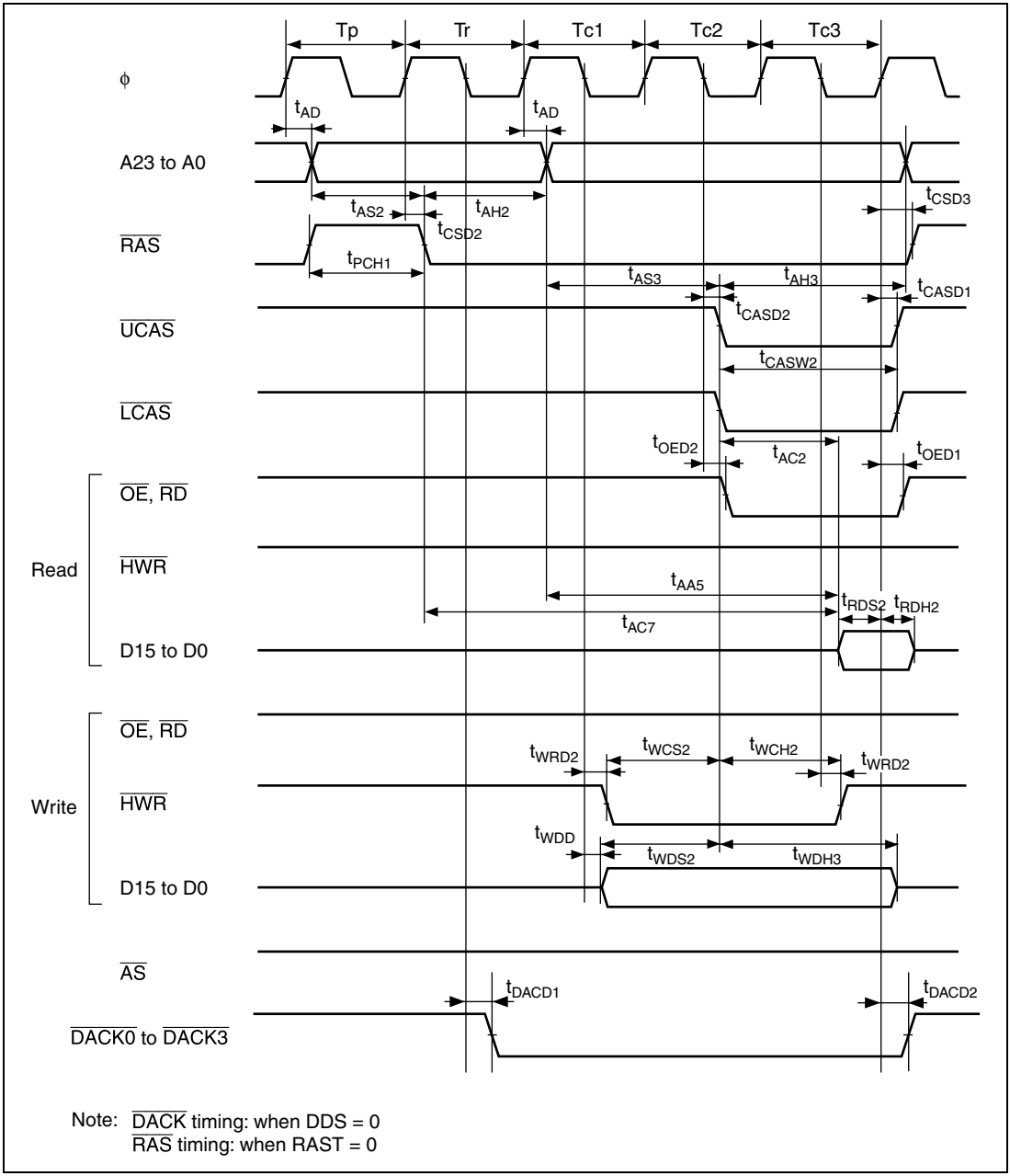


Figure 18.12 DRAM Access Timing: Two-State Access



Note: DACK timing: when DDS = 0
RAS timing: when RAST = 0

Figure 18.13 DRAM Access Timing: Two-State Burst Access



Note: DACK timing: when DDS = 0
RAS timing: when RAST = 0

Figure 18.14 DRAM Access Timing: Three-State Access (RAST = 1)

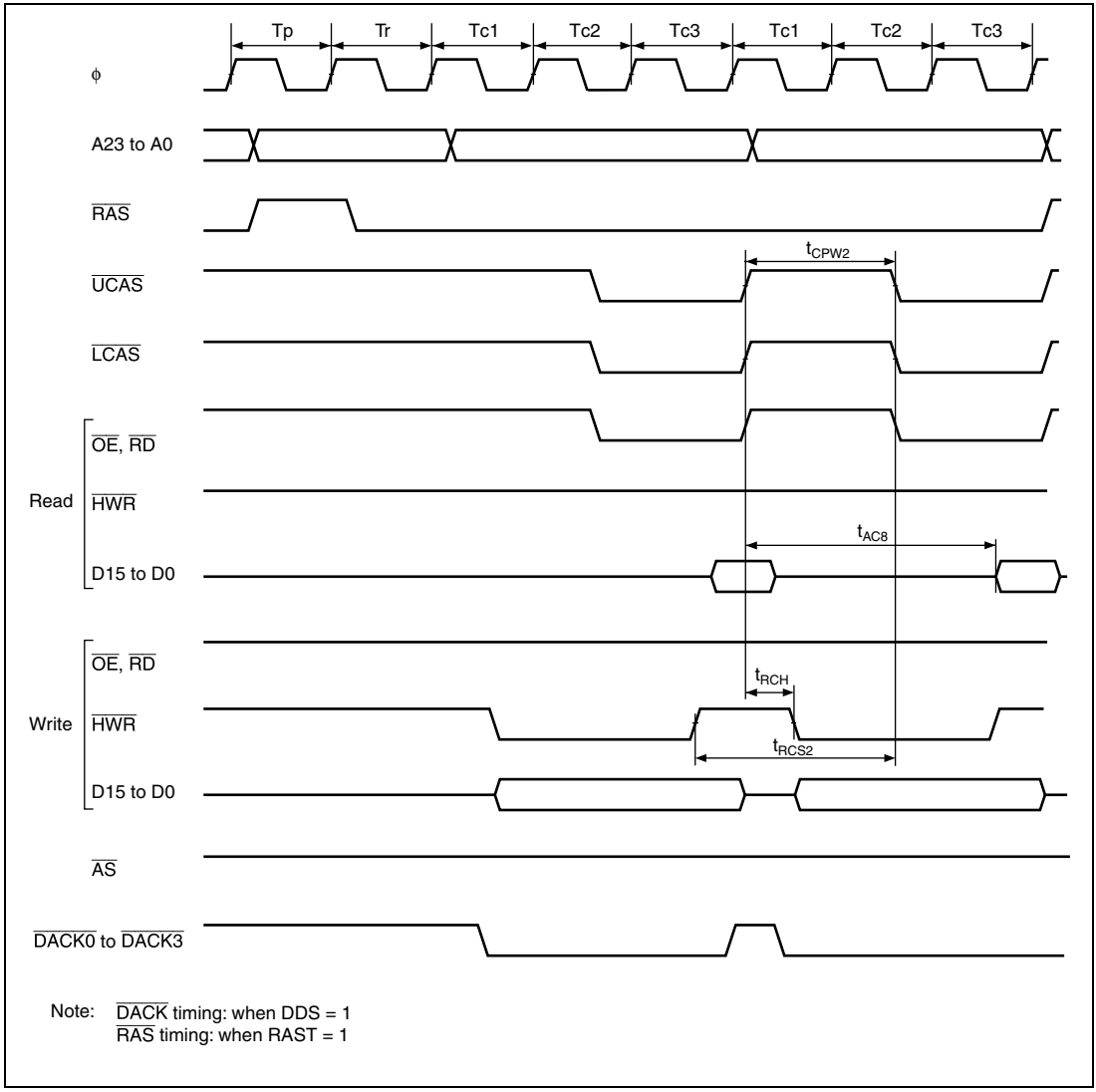


Figure 18.15 DRAM Access Timing: Three-State Burst Access

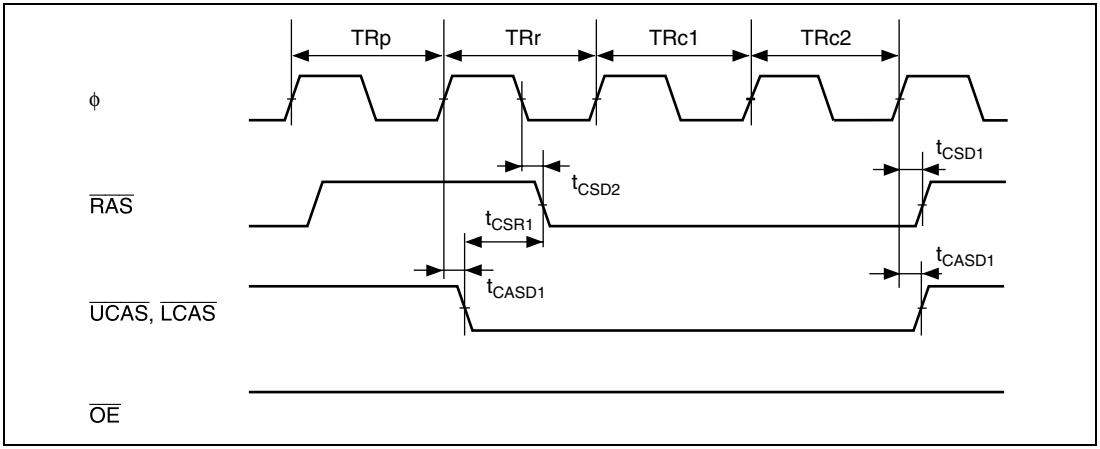


Figure 18.16 CAS-Before-RAS Refresh Timing

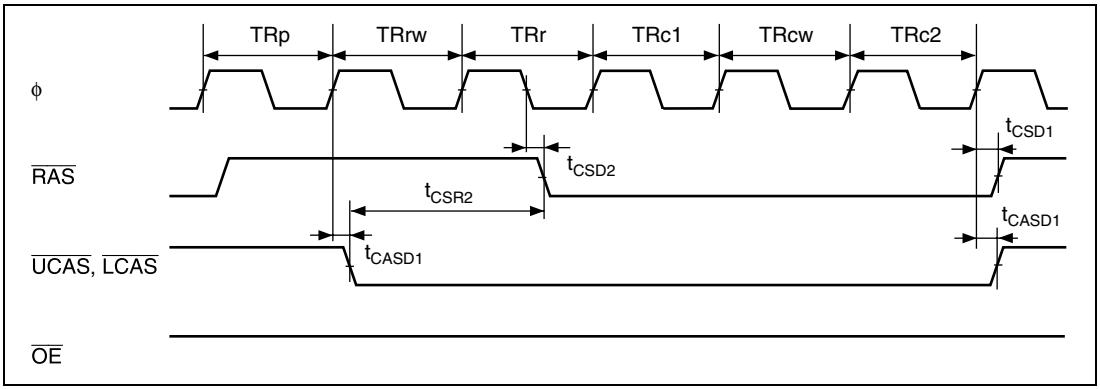


Figure 18.17 CAS-Before-RAS Refresh Timing (with Wait Cycle Insertion)

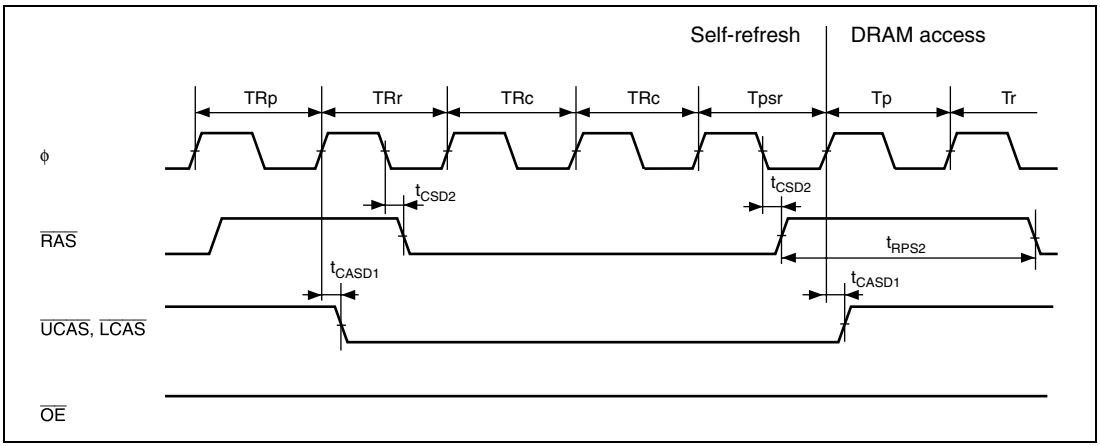


Figure 18.18 Self-Refresh Timing (Return from Software Standby Mode: RAST = 0)

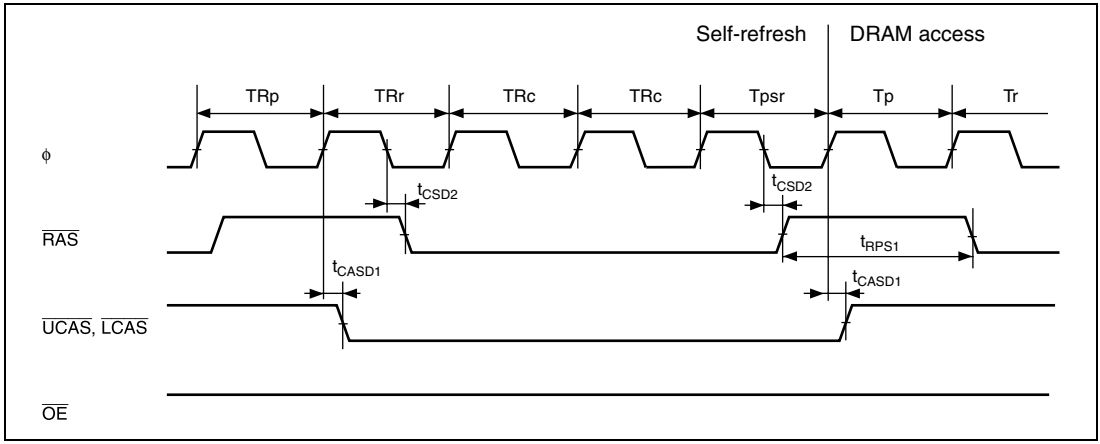


Figure 18.19 Self-Refresh Timing (Return from Software Standby Mode: RAST = 1)

18.3.4 DMAC Timing

Table 18.9 DMAC Timing

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, φ = 10 MHz to 33 MHz, T_a = –20°C to +75°C (regular specifications), T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{\text{DREQ}}$ setup time	t_{DRQS}	25	—	ns	Figure 18.23
$\overline{\text{DREQ}}$ hold time	t_{DRQH}	10	—		
$\overline{\text{TEND}}$ delay time	t_{TED}	—	18	ns	Figure 18.22
$\overline{\text{DACK}}$ delay time 1	t_{DACD1}	—	18		Figures 18.20 and 18.21
$\overline{\text{DACK}}$ delay time 2	t_{DACD2}	—	18		

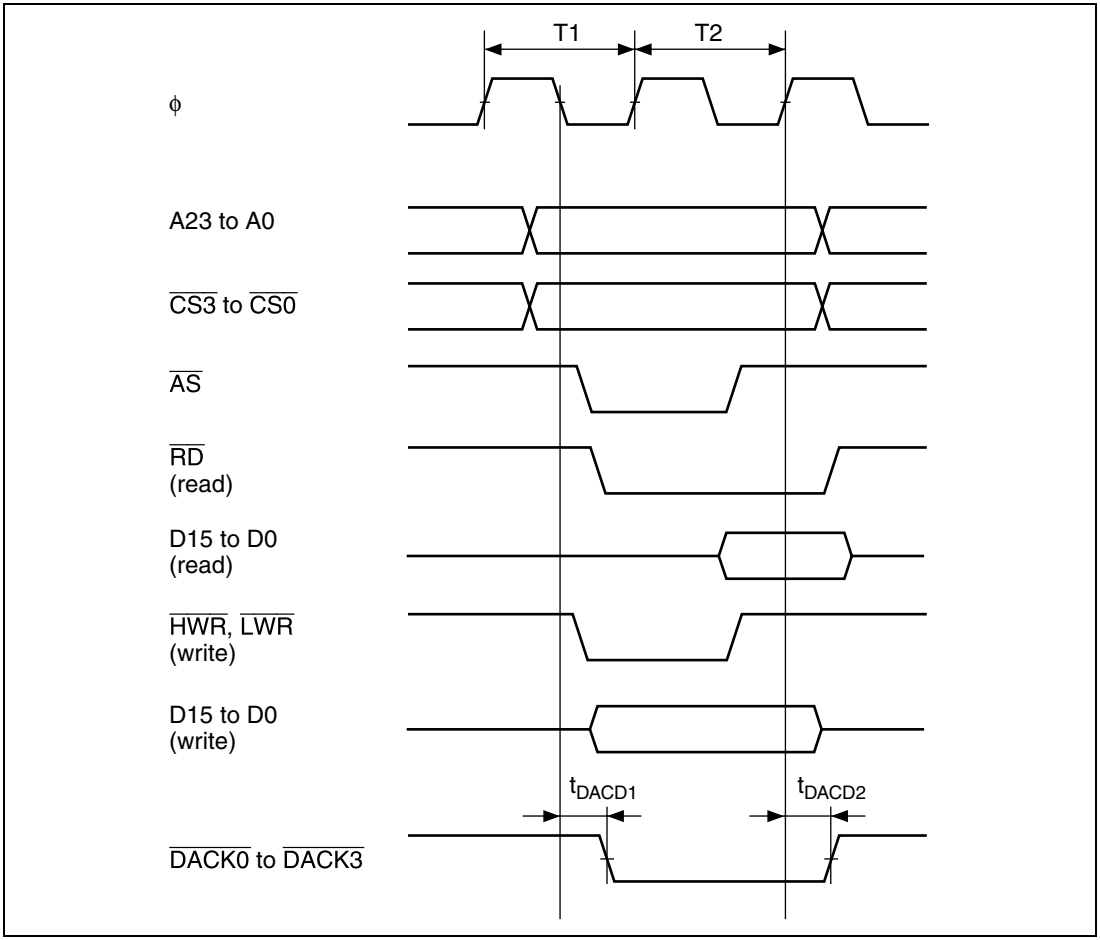


Figure 18.20 DMAC Single Address Transfer Timing: Two-State Access

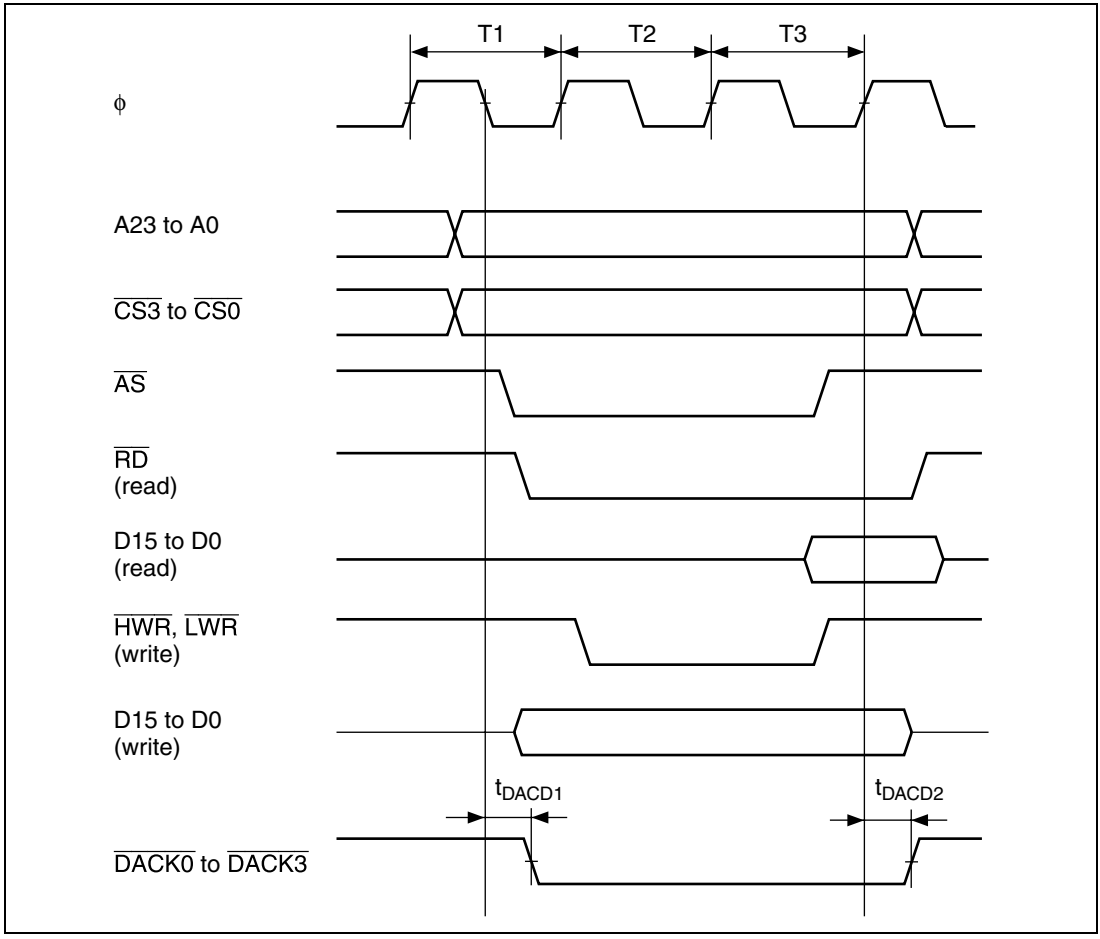


Figure 18.21 DMAC Single Address Transfer Timing: Three-State Access

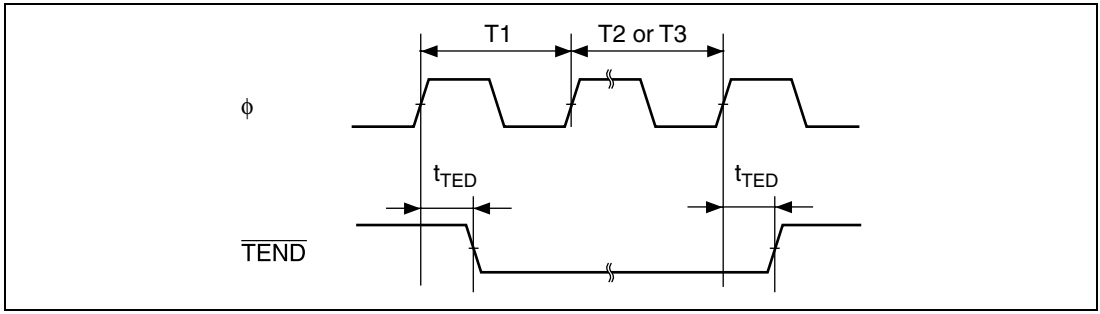


Figure 18.22 DMAC TEND Output Timing

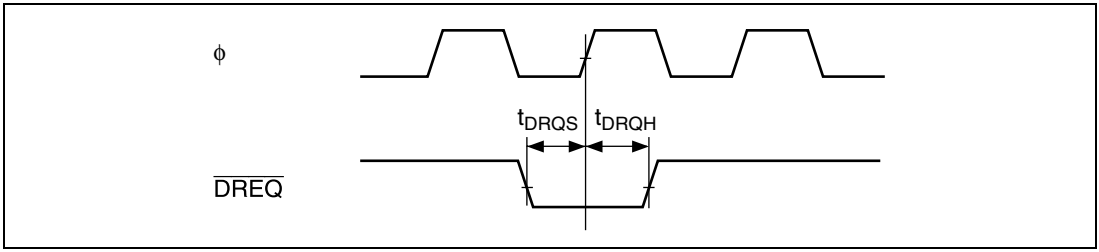


Figure 18.23 DMAC $\overline{\text{DREQ}}$ Input Timing

18.3.5 Timing of On-Chip Peripheral Modules

Table 18.10 Timing of On-Chip Peripheral Modules

Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V, ϕ = 10 MHz to 33 MHz, T_a = –20°C to +75°C

Item			Symbol	Min.	Max.	Unit	Test Conditions
I/O ports	Output data delay time		t_{PWD}	—	40	ns	Figure 18.24
	Input data setup time		t_{PRS}	25	—	ns	
	Input data hold time		t_{PRH}	25	—	ns	
8-bit timer	Timer output delay time		t_{TMOD}	—	40	ns	Figure 18.25
	Timer reset input setup time		t_{TMRS}	25	—	ns	Figure 18.27
	Timer clock input setup time		t_{TMCS}	25	—	ns	Figure 18.26
	Timer clock pulse width	Single-edge specification	t_{TMCWH}	1.5	—	t_{cyc}	
		Both-edge specification	t_{TMCWL}	2.5	—	t_{cyc}	
USB2	Data output delay time		t_{UDO}	—	18	ns	Figure 18.28
	USWDVLD output delay time		t_{UWO}	—	18		
	Control output delay time		t_{UCO}	—	18		
	Data input setup time		t_{UDS}	12	—		
	Data input hold time		t_{UDH}	2	—		
	USWDVLD input setup time		t_{UWS}	12	—		
	USWDVLD input hold time		t_{UDH}	2	—		
	Control input setup time		t_{UUCS}	12	—		
	Control input hold time		t_{UDCH}	2	—		

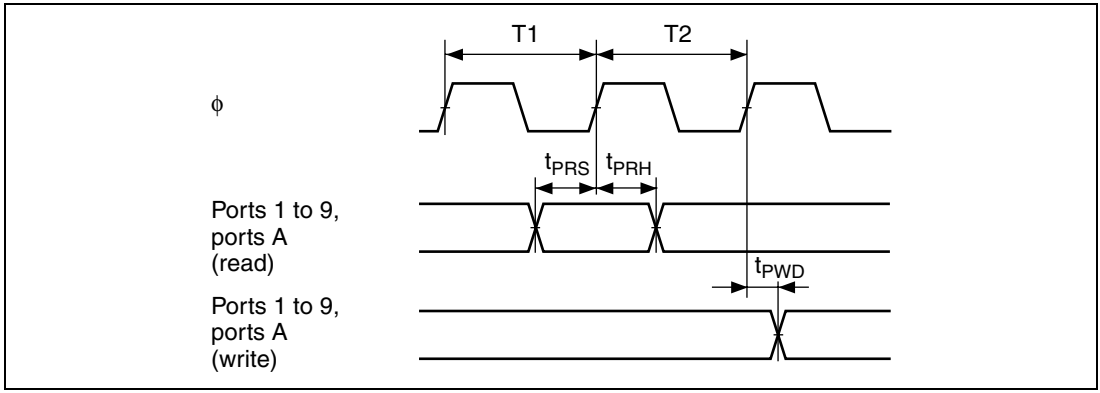


Figure 18.24 I/O Port Input/Output Timing

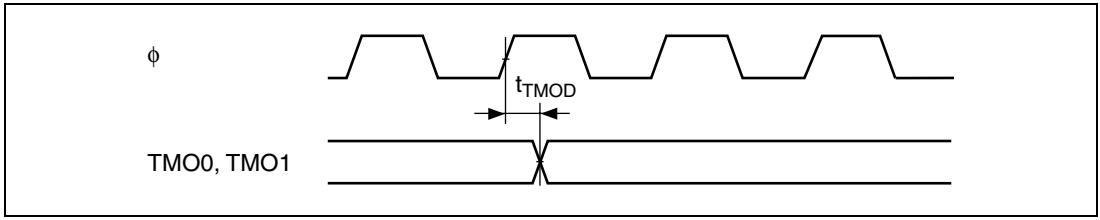


Figure 18.25 8-Bit Timer Output Timing

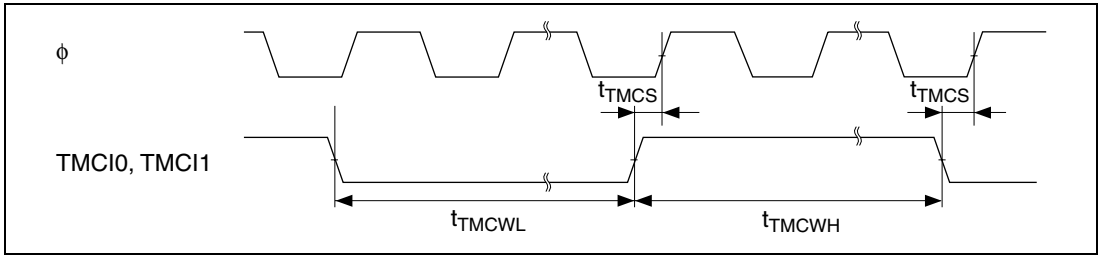


Figure 18.26 8-Bit Timer Clock Input Timing

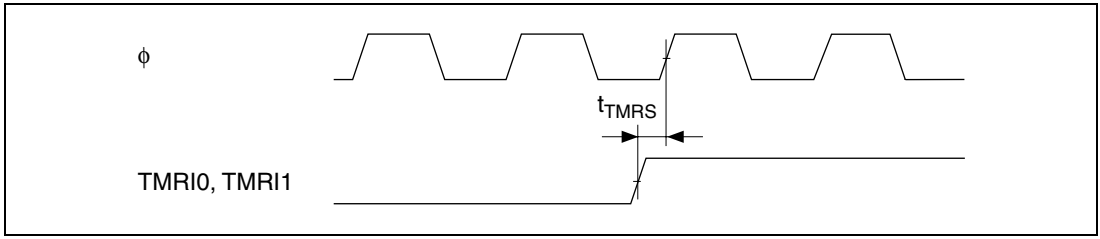


Figure 18.27 8-Bit Timer Reset Input Timing

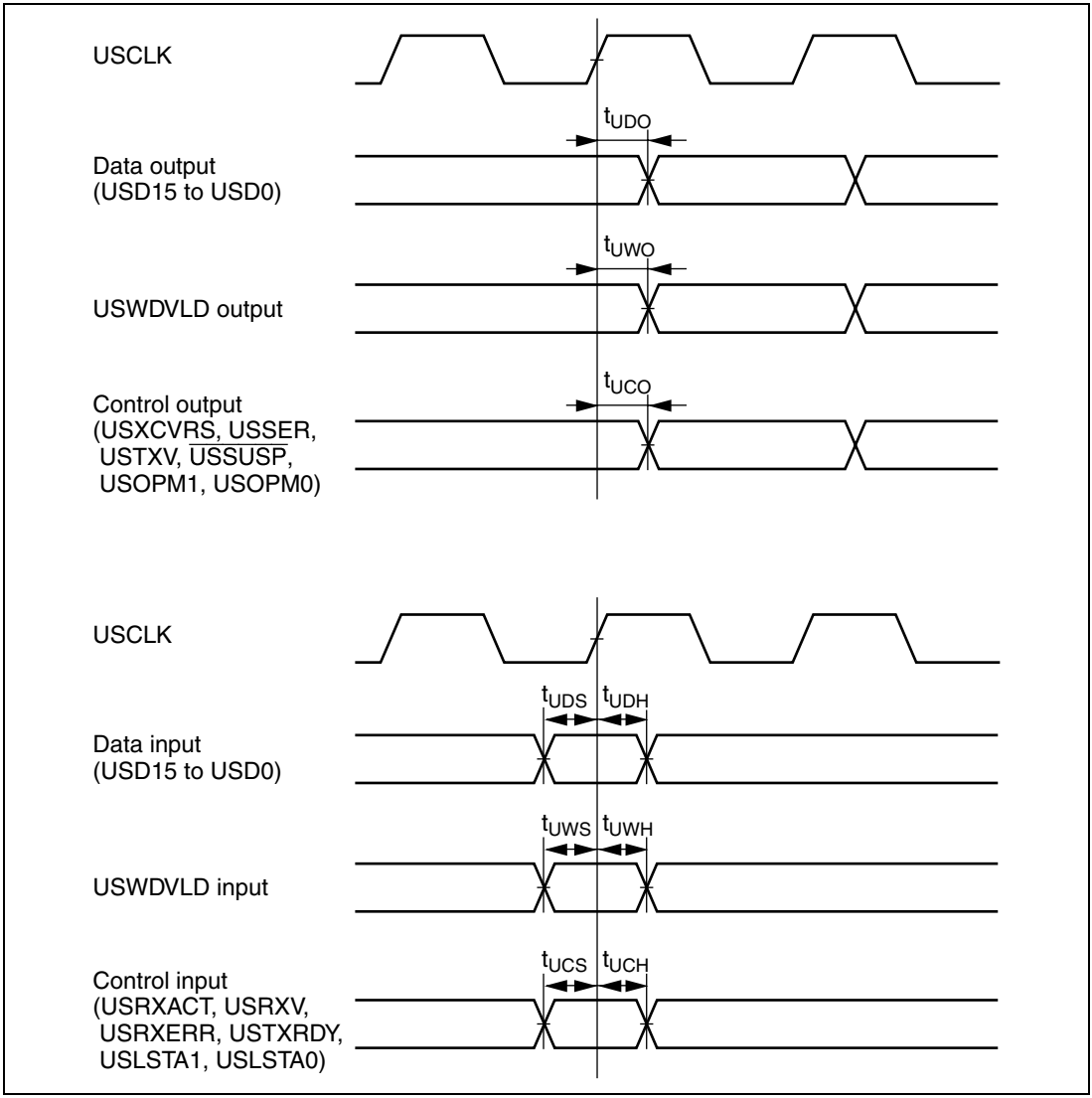


Figure 18.28 USB2 Input/output Timing

18.4 Flash Memory Characteristics

18.4.1 Flash Memory Characteristics

Table 18.11 Flash Memory Characteristics

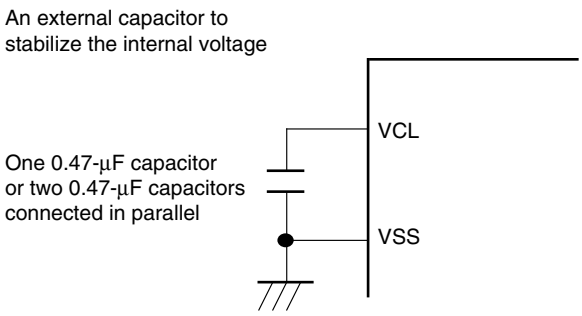
Conditions: V_{CC} = 3.0 V to 3.6 V, V_{SS} = 0 V,
T_a = 0°C to 75°C (program/erase operating temperature range)

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Programming time*1,*2,*4	t_P	—	3	30	ms/128 bytes	
Erase time*1,*2,*4	t_E	—	80	800	ms/4 kbytes	
		—	500	5000	ms/32 kbytes	
		—	1000	10000	ms/64 kbytes	
Rewrite times (total)*1,*2,*4	Σ_{tP}	—	5	15	s/256 kbytes	Ta = 25°C
Erase time (total)*1,*2,*4	Σ_{tE}	—	5	15	s/256 kbytes	Ta = 25°C
Programming and erase time (total)*1,*2,*4	Σ_{tPE}	—	10	30	s/256 kbytes	Ta = 25°C
Count of rewriting	N_{WEC}	100*3	—	—	Times	
Data hold time*4	t_{DRP}	10	—	—	Year	

Notes: 1. The programming and erase time depends on the data.
2. The programming and erase time does not include the data transfer time.
3. The minimum times that all characteristics after rewriting are guaranteed. (A range between 1 and minimum value is guaranteed.)
4. Data hold characteristics when rewriting is performed within the range of specifications including minimum value.

18.5 Use Note (Internal Voltage Step Down)

The H8S/2170 F-ZTAT have a voltage step down circuit that automatically lowers the power supply voltage, inside tha microcomputer, to an adequate level. A capacitor (one 0.47-mF capacitor ot two 0.47-mF capacitors connected in parallel) should be connected between the VCL pin (a pin for internal voltage step down circuit) and VSS pin to stabilize the internal voltage. Figure 18.29 shows how to connect the capacitor. Do not connect the V_{CC} power-supply to the VCL pin. Doing so could permanently damage the LSI. (Connect the V_{CC} power-supply to the V_{CC} pin, in the usual way.)



VDo not connect the Vcc power-supply to the VCL pin.
If connected, the LSI may be permanently damaged.
Connect the VCC power-Supply to the other Vcc pin in the usual way.
Use a multilayer ceramic capacitor (one 0.47-µF capacitor or two
0.47-µF capacitors connected in parallel) for this circuit, and place it/them
near the VCL pin.

Figure 18.29 VCL Capacitor Connection Method

A. Port States in Each Processing State

Port Name Pin Name	MCU Operating Mode	Reset	Hardware Standby Mode	Software Standby Mode	Program Execution State Sleep Mode
Port 1	2	T	T	Keep	I/O port
Port 2	2	T	T	Keep	I/O port
Port 3	2	T	T	Keep	I/O port
Port 4	2	T	T	Keep	I/O port
Port 5	2	T	T	Keep	I/O port
Port 6	2	L	T	Keep	[Address output] A15 to A8
Port 7	2	L	T	Keep	[Address output] A7 to A0
Port 8	2	T	T	Keep	I/O port
P97/ϕ	2	Clock output	T	[Clock output] H [Other than above] Keep	[Clock output] Clock output [Other than above] Input port
P96/$\overline{AS}$	2	H	T	Keep	[$\overline{AS}$ output] $\overline{AS}$ [Other than above] I/O port
P95/$\overline{RD}$	2	H	T	H	$\overline{RD}$, $\overline{HWR}$
P94/$\overline{HWR}$	2	H	T	H	$\overline{RD}$, $\overline{HWR}$
P93/$\overline{LWR}$	2	H	T	[$\overline{LWR}$ output] H [Other than above] Keep	[$\overline{LWR}$ output] $\overline{LWR}$ [Other than above] I/O port

Port Name Pin Name	MCU Operating Mode	Reset	Hardware Standby Mode	Software Standby Mode	Program Execution State Sleep Mode
P92/$\overline{CS2}$	2	H	T	[$\overline{CS}$ output] H [Other than above] Keep	[$\overline{CS}$ output] $\overline{CS}$ [Other than above] I/O port
P91/$\overline{CS1}$	2	H	T	[$\overline{CS}$ output] H [Other than above] Keep	[$\overline{CS}$ output] $\overline{CS}$ [Other than above] I/O port
P90/$\overline{CS0}$	2	H	T	[$\overline{CS}$ output] H [Other than above] Keep	[$\overline{CS}$ output] $\overline{CS}$ [Other than above] I/O port
PA3/A19/$\overline{CS3}$	2	T	T	Keep	[Address output] A19 [$\overline{CS}$ output] $\overline{CS}$ [Other than above] I/O port
PA2/A18/ $\overline{UCAS}$	2	T	T	Keep	[$\overline{UCAS}$ and address output] $\overline{UCAS}$, A18 [Other than above] I/O port
PA1/A17/ $\overline{LCAS}$	2	T	T	Keep	[$\overline{LCAS}$ and address output] $\overline{LCAS}$, A17 [Other than above] I/O port

Port Name Pin Name	MCU Operating Mode	Reset	Hardware Standby Mode	Software Standby Mode	Program Execution State Sleep Mode
PA0/A16	2	T	T	Keep	[Address output] A16 [Other than above] I/O port

Legend:

L: Low level

H: High level

Keep: Input port becomes high-impedance, output port retains state

T: High impedance

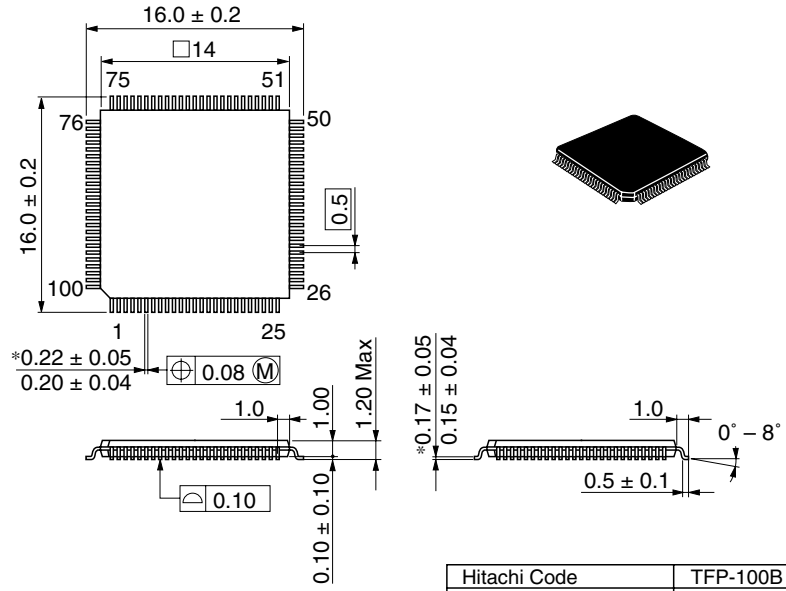
DDR: Data direction register

OPE: Output port enable

B. Product Lineup

Product Classification		Type Name	Model Marking	Package (Code)
H8S/2170	F-ZTAT version	HD64F2170	HD64F2170	100-pin TQFP (TFP-100B)

C. Package Dimensions



Hitachi Code	TFP-100B
JEDEC	—
JEITA	Conforms
Mass (reference value)	0.5 g

*Dimension including the plating thickness
Base material dimension

Figure C.1 Package Dimensions (TFP-100B)

Index

8-Bit Timer 265
16-Bit Counter Mode 276
Cascaded Connection 276
Compare Match Count Mode 277
Pulse Output 272
TCNT Incrementation Timing 273
Toggle output 281
Address Map 53
Address Space 16, 20
Addressing Mode 40
Absolute Address 41
Immediate 42
Memory Indirect 42
Program-Counter Relative 42
Register Direct 41
Register Indirect 41
Register Indirect with Displacement 41
Register Indirect with Post-Increment 41
Register indirect with pre-decrement 41
Bcc 29, 37
Boot Mode 401
Bulk-In Transfer 352
Bulk-Out Transfer 350
Bus Arbitration 150
Bus Controller 83
Basic Timing 109
Chip Select Signals 107
Data Size and Data Alignment 108
Extension of Chip Select ($\overline{CS}$) Assertion Period 120
Read Strobe (RD) Timing 119
Valid Strobes 109
Wait Control 118
Clock Pulse Generator 463
Communications Protocol 437
Condition Field 39
Condition-Code Register (CCR) 24
Control Transfer 344
CPU Operating Modes 16
Advanced Mode 18
Normal Mode 16
Crystal Resonator 463
Data direction register 219
Data register 219
DMA Controller
Channel Priority 186
Dual Address Mode 189
Repeat Area Function 179
Single Address Mode 196
Transfer Modes 171
DMA Controller (DMAC) 153
DMA Transfer Specifications 359, 360
DRAM Interface 121
Effective Address 40, 44
effective address extension 39
Effective Address Extension 39
Error Protection 429
Exception Handling 55
Exception Handling Vector Table 56
Extended Control Register (EXR) 23
FCCS 383
FECS 386
FKEY 387
Flash MAT Configuration 377
FMATS 388
FPCS 386
FTDAR 389, 390
General Registers 22
Hardware Protection 428
Hardware Standby Mode 476
Idle Cycle 141
Instruction Set 29
Arithmetic Operations Instructions 32
Bit Manipulation Instructions 35
Block Data Transfer Instruction 39
Branch Instructions 37
Data Transfer Instructions 31
Logic Operations Instructions 34
Shift Instructions 34
System Control Instruction 38

Interrupt Control Modes 72
Interrupt Controller 63
Interrupt Exception Handling 59
Interrupt Exception Handling Sequence ... 78
Interrupt Exception Handling Vector Table 70
Interrupt Mask Bit 24
Interrupt Request Mask Level 23
Interrupt-In Transfer 354
Interrupts
CMIA 277
CMIB 277
ERI0 316
IRQ7 to IRQ0 Interrupts 69
NMI Interrupt 69
OVI 277
RXI0 316
TEI0 316
TXI0 316
WOVI 290
Interval Timer Mode 289
List of Registers 479
Register Addresses 480
Register Bits 485
Register States in Each Operating Mode 493
MDA Controller
Interrupt Sources 213
Mode Comparison 376
Module Stop Mode 477
On-Board Programming 401
on-board programming mode 373
Operating Modes 49
Operation Field 39
overflow 288
PLL Circuit 467
Port register 219
Power-Down Modes 469
Procedure Program 418
Processing of USB Standard Requests and Class/Vendor Requests 355
Program Counter (PC) 23
Programmer Mode 434
Programming/Erasing Interface Parameter
Download pass/fail result parameter ... 393
Flash erase block select parameter 398
Flash multipurpose address area parameter 396
Flash multipurpose data destination parameter 396
Flash pass/fail parameter 400
Flash programming/erasing frequency parameter 394
Programming/Erasing Interface Register 382
Protection 428
RAM 371
Register Field 39
Registers
ABRKCR 65, 482, 490, 495
ABWCR 87
ACSCR 480, 487, 493
BARC 66
BCR 92, 481, 487, 493
BRR 302, 483, 491, 495
CSACR 88, 480, 487, 493
CTRL 338, 480, 487, 493
DASTS0 ..... 333, 480, 486, 493
DMA0 ..... 337, 480, 487, 493
DMACR 165, 481, 488, 494
DMDAR 158, 481, 488, 494
DMMDR 160, 481, 488, 494
DMSAR 158, 481, 488, 494
DMTCR 158, 481, 488, 494
DRACCR 97, 481, 487, 494
DRAMCR 94, 481, 487, 493
EPDR0i 331, 480, 485, 493
EPDR0o 331, 480, 485, 493
EPDR0s 332, 480, 486, 493
EPDR1 ..... 332, 480, 486, 493
EPDR2 ..... 332, 480, 486, 493
EPDR3 ..... 333, 480, 486, 493
EPSTL0 ..... 337, 480, 487, 493
EPSZ0o 330, 480, 485, 493
EPSZ1 ..... 331, 480, 485, 493
FCLR0 ..... 335, 480, 487, 493
ICR 64, 482, 490, 495

IER 68, 482, 490, 495
IER0 ..... 330, 480, 485, 493
IFR0 ..... 325, 480, 485, 493
ISCR 67, 482, 490, 495
ISR 68, 482, 490, 495
ISR0 ..... 329, 480, 485, 493
ISSR 263, 482, 490, 495
MDCR 50, 480, 487, 493
MSTPCR 295, 473, 487, 493
MSTPCRH 480
P1DDR 222, 482, 490, 495
P1DR 223, 483, 491, 496
P2DDR 226, 482, 490, 495
P2DR 226, 483, 491, 496
P3DDR 230, 482, 490, 495
P3DR 230, 483, 491, 496
P4DDR 233, 482, 490, 495
P4DR 234, 483, 491, 496
P5DDR 237, 482, 490, 495
P5DR 237, 483, 491, 496
P6DDR 242, 482, 490, 495
P6DR 242, 483, 491, 496
P7DDR 246, 482, 491, 495
P7DR 246, 483, 491, 496
P8DDR 251, 483, 491, 495
P8DR 252, 483, 491, 496
P9DDR 255, 483, 491, 495
P9DR 255, 483, 491, 496
PADDR 258, 483, 491, 495
PADR 259, 483, 491, 496
PBAR 66, 490
PBARA 482, 495
PBARB 482
PBARC 482
PFCR1 ..... 261, 482, 490, 495
PFCR3 ..... 339, 482, 490, 495
PKTE0i 334, 480, 486, 493
PKTE2 ..... 334, 480, 486, 493
PKTE3 ..... 334, 480, 486, 493
PORT1 ..... 223, 483, 491, 495
PORT2 ..... 227, 483, 491, 495
PORT3 ..... 231, 483, 491, 496
PORT4 ..... 234, 483, 491, 496
PORT5 ..... 238, 483, 491, 496
PORT6 ..... 243, 483, 491, 496
PORT7 ..... 247, 483, 491, 496
PORT8 ..... 252, 483, 491, 496
PORT9 ..... 256, 483, 491, 496
PORTA 259, 295, 483, 491, 496
RDNCR 93, 481, 487, 493
RDR 295, 483, 491, 495
REFCR 98, 481, 488, 494
RSR 295
RTCNT 101, 481, 488, 494
RTCOR 101, 481, 488, 494
SBYCR 472, 480, 487, 493
SCR 297, 483, 491, 495
SMR 296, 483, 491, 495
SSR 299, 483, 491, 495
SYSCR 51, 480, 487, 493
TCNT ..... 267, 286, 481, 484, ... 488, 492, 494, 496
TCORA 268, 484, 492, 496
TCORB 268, 484, 492, 496
TCR 268, 483, 491, 496
TCSR ..... 270, 286, 481, 483, ... 488, 491, 494, 496
TDR 295, 483, 491, 495
TSR 296
USBSUSP 339, 480, 487, 493
USTCR 169, 482, 490, 495
WTCR 90, 481, 487, 493
Reset 57
Reset Exception Handling 57
Serial Communication Interface 293
Asynchronous Mode 306
Bit rate 302
Framing error 312
Overrun error 312
Parity error 312
Serial Communication Interface Specification 435
Sleep Mode 474
Software Protection 429
Software Standby Mode 474
Stack Pointer (SP) 22

Stack Status 60
Stall Operations 356
Trace Bit 23
Trap Instruction Exception Handling 59
TRAPA instruction 42, 59
Universal Serial Bus 321
USB Cable Connection 342
USB Cable Disconnection 343
user boot MAT 430
user boot memory MAT 373
User Boot Mode 415
user MAT 430
user memory MAT 373
User Program Mode 405
Watchdog Timer (WDT) 285
Watchdog Timer Mode 288
Write Data Buffer Function 149

H8S/2172 Series H8S/2170 F-ZTAT Hardware Manual

Publication Date: 1st Edition, March 2003
Published by: Business Operation Division
Semiconductor & Integrated Circuits
Hitachi, Ltd.
Edited by: Technical Documentation Group
Hitachi Kodaira Semiconductor Co., Ltd.
Copyright © Hitachi, Ltd., 2003. All rights reserved. Printed in Japan.