Everywhere you imagine.

32

SH7660

Hardware Manual

Renesas 32-Bit RISC Microcomputer
HD6417660

User's Manual

Rev.1.00
2004.2.6

Renesas Technology
www.renesas.com

Cautions

Keep safety first in your circuit designs!

1. Renesas Technology Corp. 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 Corp. 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 Corp. or a third party.
2. Renesas Technology Corp. 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 Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. 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 Corp. 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 Corp. by various means, including the Renesas Technology Corp. 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 Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. 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 Corp. or an authorized Renesas Technology Corp. 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 Corp. 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 Corp. for further details on these materials or the products contained therein.

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 they 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.

Preface

The SH7660 RISC (Reduced Instruction Set Computer) microcomputer includes a Renesas Technology original RISC CPU as its core, and the peripheral functions required to configure a system.

Target Users: This manual was written for users who will be using this LSI in the design of application systems. Users of this manual 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 above users.
Refer to the SH-3/SH-3E/SH3-DSP Programming Manual for a detailed description of the instruction set.

Notes on reading this manual:

Product names

The following products are covered in this manual.

Product Classifications and Abbreviations

Basic Classification	Product Code
SH7660	HD6417660

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 SH-3/SH-3E/SH3-DSP Programming Manual.

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

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.renesas.com/eng/

SH7660 manuals:

Document Title	Document No.
SH7660 Hardware Manual	This manual
SH-3/SH-3E/SH3-DSP Programming Manual	ADE-602-096

Users manuals for development tools:

Document Title	Document No.
SH Series C/C++ Compiler, Assembler, Optimizing Linkage Editor User's Manual	ADE-702-246
SH Series Simulator/Debugger (for Windows) User's Manual	ADE-702-186
SH Series Simulator/Debugger (for UNIX) User's Manual	ADE-702-203
High-performance Embedded Workshop User's Manual	ADE-702-201
SH Series High-performance Embedded Workshop, High-performance Debugging Interface Tutorial	ADE-702-230

Abbreviations

ALU	Arithmetic Logic Unit
ASE	Adaptive System Evaluator
AUD	Advanced User Debugger
bps	bit per second
BSC	Bus State Controller
CODEC	Coder-Decoder
CPG	Clock Pulse Generator
CPU	Central Processing Unit
DAC	Digital to Analog Converter
DMAC	Direct Memory Access Controller
DSP	Digital Signal Processor
ESD	Electrostatic Discharge
FIFO	First-In First-Out
Hi-Z	High Impedance
H-UDI	User Debugging Interface
INTC	Interrupt Controller
LSB	Least Significant Bit
MSB	Most Significant Bit
PC	Program Counter
PFC	Pin Function Controller
PLL	Phase Locked Loop
RAM	Random Access Memory
RF	Ratio Frequency
RISC	Reduced Instruction Set Computer
ROM	Read Only Memory
SIOF	Serial Input Output with FIFO
SCIF	Serial Communication Interface with FIFO
SOF	Start Of Frame
TAP	Test Access Port
T.B.D.	To Be Determined
TLB	Translation Lookaside Buffer
UBC	User Break Controller
USB	Universal Serial Bus
WDT	Watch Dog Timer

Contents

Section 1 Overview 1
1.1 SH7660 Features 1
1.2 Block Diagram 8
1.3 Pin Assignment 9
1.4 Pin Functions 21

Section 2 CPU 29
2.1 Processing States and Processing Modes 29
2.1.1 Processing States 29
2.1.2 Processing Modes (User Mode/Privileged Mode) 30
2.2 Memory Map 31
2.2.1 Logical Address Space 31
2.2.2 Physical Address Space 33
2.2.3 External Address Space 34
2.3 Register Descriptions 36
2.3.1 General Registers 38
2.3.2 System Registers 39
2.3.3 Program Counter 40
2.3.4 Control Registers 41
2.4 Data Formats 45
2.4.1 Register Data Format 45
2.4.2 Memory Data Formats 45
2.5 Features of Instructions 47
2.5.1 Instruction Execution Method 47
2.5.2 Addressing Modes 49
2.5.3 Instruction Formats 53
2.6 Instruction Set 56
2.6.1 Instruction Set Based on Functions 56
2.6.2 Operation Code Map 70

Section 3 DSP Operating Unit 73
3.1 DSP Extended Functions 73
3.2 DSP Mode Resources 75
3.2.1 Processing Modes 75
3.2.2 DSP Mode Memory Map 75
3.2.3 CPU Register Sets 76
3.2.4 DSP Registers 80
3.3 CPU Extended Instructions 81
3.3.1 DSP Repeat Control 81

3.3.2 Extended Repeat Control Instructions 91
3.4 DSP Data Transfer Instructions 96
3.4.1 General Registers 99
3.4.2 DSP Data Addressing 101
3.4.3 Modulo Addressing 103
3.4.4 Memory Data Formats 105
3.4.5 Instruction Formats of Double and Single Data Transfer Instructions 105
3.5 DSP Data Operation Instructions 107
3.5.1 DSP Registers 107
3.5.2 DSP Instruction Set 111
3.5.3 DSP-Type Data Formats 116
3.5.4 ALU Fixed-Point Arithmetic Operations 118
3.5.5 ALU Integer Operations 123
3.5.6 ALU Logical Operations 125
3.5.7 Fixed-Point Multiply Operation 126
3.5.8 Shift Operations 129
3.5.9 Most Significant Bit Detection Operation 132
3.5.10 Rounding Operation 135
3.5.11 Overflow Protection 136
3.5.12 Local Data Move Instruction 137
3.5.13 Operand Conflict 138
3.6 DSP Extended Function Instruction Set 139
3.6.1 CPU Extended Instruction Set 139
3.6.2 Double-Data Transfer Instruction Set 141
3.6.3 Single-Data Transfer Instruction Set 142
3.6.4 DSP Data Operation Instruction Set 144
3.6.5 Operation Code Map in DSP Mode 150

Section 4 Exception Handling 155
4.1 Register Descriptions 155
4.1.1 TRAPA Exception Register (TRA) 156
4.1.2 Exception Event Register (EXPEVT) 157
4.1.3 Interrupt Event Register 2 (INTEVT2) 157
4.1.4 Exception Address Register (TEA) 157
4.2 Exception Handling Function 158
4.2.1 Exception Handling Flow 158
4.2.2 Exception Vector Addresses 159
4.2.3 Exception Codes 159
4.2.4 Exception Request and BL Bit (Multiple Exception Prevention) 159
4.2.5 Exception Source Acceptance Timing and Priority 160
4.3 Individual Exception Operations 164
4.3.1 Resets 164
4.3.2 General Exceptions 165

4.4 Exception Processing While DSP Extension Function is Valid 168
4.4.1 Illegal Instruction Exception and Slot Illegal Instruction Exception 168
4.4.2 CPU Address Error 168
4.4.3 Exception in Repeat Control Period 168
4.5 Usage Notes 175

Section 5 Cache 177
5.1 Features 177
5.1.1 Cache Structure 177
5.2 Register Descriptions 179
5.2.1 Cache Control Register 1 (CCR1) 179
5.2.2 Cache Control Register 2 (CCR2) 180
5.3 Operation 183
5.3.1 Searching the Cache 183
5.3.2 Read Access 184
5.3.3 Prefetch Operation 184
5.3.4 Write Access 184
5.3.5 Write-Back Buffer 185
5.3.6 Coherency of Cache and External Memory 185
5.4 Memory-Mapped Cache 186
5.4.1 Address Array 186
5.4.2 Data Array 187
5.4.3 Usage Examples 189

Section 6 X/Y Memory 191
6.1 Features 191
6.2 Operation 192
6.2.1 Access from CPU 192
6.2.2 Access from DSP 192
6.2.3 Access from I Bus Master Module 193
6.3 Usage Notes 193
6.3.1 Page Conflict 193
6.3.2 Bus Conflict 193
6.3.3 Cache Settings 193
6.3.4 Sleep Mode 194

Section 7 U Memory 195
7.1 Features 195
7.2 Operation 196
7.2.1 Access from CPU 196
7.2.2 Access from DSP 196
7.2.3 Access from I Bus Master Module 196
7.3 Usage Notes 197

7.3.1 Page Conflict 197
7.3.2 Bus Conflict 197
7.3.3 Cache Settings 197
7.3.4 sleep mode 198

Section 8 Interrupt Controller (INTC) 199
8.1 Features 199
8.2 Input/Output Pins 201
8.3 Register Descriptions 201
8.3.1 Interrupt Priority Registers A to H (IPRA to IPRH) 202
8.3.2 Interrupt Control Register 0 (ICR0) 204
8.3.3 Interrupt Control Register 1 (ICR1) 205
8.3.4 Interrupt Control Register 2 (ICR2) 207
8.3.5 Interrupt Request Register 0 (IRR0) 208
8.3.6 Interrupt Mask Registers 0, 1, 4 to 6, and 9 (IMR0, IMR1, IMR4 to IMR6, and IMR9) 208
8.3.7 Interrupt Mask Clear Registers 0, 1, 4 to 6, and 9 (IMCR0, IMCR1, IMCR4 to IMCR6, and IMCR9) 210
8.4 Interrupt Sources 211
8.4.1 NMI Interrupt 211
8.4.2 IRQ Interrupts 211
8.4.3 On-Chip Peripheral Module Interrupts 212
8.4.4 Interrupt Exception Handling and Priority 212
8.5 Operation 214
8.5.1 Interrupt Sequence 214
8.5.2 Multiple Interrupts 216

Section 9 Bus State Controller (BSC) 217
9.1 Overview 217
9.1.1 Features 217
9.2 Input/Output Pins 220
9.3 Area Overview 221
9.3.1 Area Division 221
9.3.2 Shadow Area 222
9.3.3 Address Map 223
9.3.4 Data Bus Width 224
9.4 Register Descriptions 224
9.4.1 Common Control Register (CMNCR) 225
9.4.2 CSn Space Bus Control Register (CSnBCR) (n=0, 3, 4) 227
9.4.3 CSn Space Wait Control Register (CSnWCR) (n=0, 3, 4) 231
9.4.4 SDRAM Control Register (SDCR) 242
9.4.5 Refresh Timer Control/Status Register (RTCSR) 245
9.4.6 Refresh Timer Counter (RTCNT) 246

9.4.7 Refresh Time Constant Register (RTCOR) 247
9.4.8 Reset Wait Counter (RWTCNT) 247
9.5 Operating Description 248
9.5.1 Endian/Access Size and Data Alignment 248
9.5.2 Normal Space Interface 251
9.5.3 Access Wait Control 256
9.5.4 $\overline{\text{CSn}}$ Assert Period Expansion 258
9.5.5 SDRAM Interface 259
9.5.6 Burst ROM Interface 285
9.5.7 Byte-Selection SRAM Interface 287
9.5.8 Wait between Access Cycles 291
9.5.9 Bus Arbitration 291
9.5.10 Usage Notes 295

Section 10 Direct Memory Access Controller (DMAC) 297
10.1 Features 297
10.2 Input/Output Pins 299
10.3 Register Descriptions 300
10.3.1 DMA Source Address Register (SAR) 301
10.3.2 DMA Destination Address Register (DAR) 301
10.3.3 DMA Transfer Count Register (DMATCR) 301
10.3.4 DMA Channel Control Register (CHCR) 302
10.3.5 DMA Initial Address Register (IAR) 308
10.3.6 DMA Operation Register (DMAOR) 308
10.3.7 DMA Extension Resource Selector 0 and 1 (DMARS0 and DMARS1) 310
10.4 Operation 313
10.4.1 DMA Transfer Flow 313
10.4.2 Repeat Mode Transfer 315
10.4.3 DMA Transfer Requests 315
10.4.4 Channel Priority 318
10.4.5 DMA Transfer Types 321
10.4.6 Number of Bus Cycle States and DREQ Pin Sampling Timing 328
10.5 Usage Notes 332

Section 11 Clock Pulse Generator (CPG) 333
11.1 Features 333
11.2 Input/Output Pins 336
11.3 Clock Operating Modes 337
11.4 Register Descriptions 340
11.4.1 Frequency Control Register (FRQCR) 340
11.5 Changing the Frequency 343
11.5.1 Changing the Multiplication Rate 343
11.5.2 Changing the Division Ratio 343

11.6 Notes on Board Design 344

Section 12 Watchdog Timer (WDT) 347
12.1 Features 347
12.2 Register Descriptions 349
12.2.1 Watchdog Timer Counter (WTCNT) 349
12.2.2 Watchdog Timer Control/Status Register (WTCSR) 349
12.2.3 Notes on Register Access 351
12.3 Operation 352
12.3.1 Canceling Software Standby 352
12.3.2 Changing the Frequency 352
12.3.3 Using Watchdog Timer Mode 353
12.3.4 Using Interval Timer Mode 353

Section 13 Power-Down Modes 355
13.1 Features 355
13.1.1 Power-Down Modes 355
13.1.2 Reset 357
13.1.3 Input/Output Pins 358
13.2 Register Descriptions 359
13.2.1 Standby Control Register (STBCR) 360
13.2.2 Standby Control Register 2 (STBCR2) 361
13.2.3 Standby Control Register 3 (STBCR3) 362
13.2.4 Standby Control Register 4 (STBCR4) 363
13.2.5 Memory Clock Control Register (MCCR) 365
13.3 Operation 366
13.3.1 Sleep Mode 366
13.3.2 Software Standby Mode 366
13.3.3 Module Standby Function 368
13.3.4 State Transitions between Modes 369
13.3.5 Method of turning off the built-in regulator (external V_{DD} used) 369
13.4 Usage note 370
13.4.1 Reset by the $\overline{\text{RESETP}}$ pin 370

Section 14 Timer Unit (TMU) 371
14.1 Features 371
14.2 Register Descriptions 373
14.2.1 Timer Start Register (TSTR) 374
14.2.2 Timer Control Registers (TCR) 374
14.2.3 Timer Constant Registers (TCOR) 376
14.2.4 Timer Counters (TCNT) 376
14.3 Operation 377
14.3.1 Counter Operation 377

14.4 Interrupts 379
14.4.1 Status Flag Set Timing 379
14.4.2 Status Flag Clear Timing 379
14.4.3 Interrupt Sources and Priorities 380
14.5 Usage Notes 380

Section 15 Serial I/O with FIFO (SIOF) 381
15.1 Features 381
15.2 Input/Output Pins 383
15.3 Register Descriptions 384
15.3.1 Mode Register (SIMDR) 384
15.3.2 Control Register (SICTR) 387
15.3.3 Transmit Data Register (SITDR) 390
15.3.4 Receive Data Register (SIRDR) 391
15.3.5 Transmit Control Data Register (SITCR) 392
15.3.6 Receive Control Data Register (SIRCR) 393
15.3.7 Status Register (SISTR) 394
15.3.8 Interrupt Enable Register (SIIER) 399
15.3.9 FIFO Control Register (SIFCTR) 401
15.3.10 Clock Select Register (SISCR) 403
15.3.11 Transmit Data Assign Register (SITDAR) 404
15.3.12 Receive Data Assign Register (SIRDAR) 405
15.3.13 Control Data Assign Register (SICDAR) 406
15.3.14 SPI Control Register (SPICR) 408
15.4 Operation 410
15.4.1 Serial Clocks 410
15.4.2 Serial Timing 411
15.4.3 Transfer Data Format 412
15.4.4 Register Allocation of Transfer Data 413
15.4.5 Control Data Interface 416
15.4.6 FIFO 417
15.4.7 Transmit and Receive Procedures 419
15.4.8 Interrupts 424
15.4.9 Transmit and Receive Timing 426
15.4.10 SPI Mode 430
15.5 Usage Notes 433

Section 16 Serial Communication Interface with FIFO (SCIF) 435
16.1 Features 435
16.2 Input/Output Pins 438
16.3 Register Descriptions 439
16.3.1 Receive Shift Register (SCRSR) 440
16.3.2 Receive FIFO Data Register (SCFRDR) 440

16.3.3 Transmit Shift Register (SCTSR) 440
16.3.4 Transmit FIFO Data Register (SCFTDR) 441
16.3.5 Serial Mode Register (SCSMR) 442
16.3.6 Serial Control Register (SCSCR) 445
16.3.7 FIFO Error Count Register (SCFER) 449
16.3.8 Serial Status Register (SCSSR) 450
16.3.9 Bit Rate Register (SCBRR) 456
16.3.10 FIFO Control Register (SCFCR) 458
16.3.11 FIFO Data Count Register (SCFDR) 461
16.3.12 Transmit Data Stop Register (SCTDSR) 461
16.4 Operation 462
16.4.1 Overview 462
16.4.2 Asynchronous Mode 462
16.4.3 Serial Operation in Asynchronous Mode 464
16.4.4 Clock Synchronous Mode 475
16.4.5 Serial Operation in Clock Synchronous Mode 476
16.5 SCIF Interrupt Sources and DMAC 485
16.6 Notes on Usage 487

Section 17 Boot Function (BOOT) 491
17.1 Features 491
17.2 Input/Output Pin 493
17.3 Register Description 493
17.4 Operation 494
17.4.1 Address Space in Boot Mode 494
17.4.2 Procedure for Execution of Boot Processing 495
17.5 Usage Note 498
17.5.1 Endian when Using Boot Function 498
17.5.2 Clock Frequency and Data Transfer Rate when Using Boot Function 498

Section 18 USB Pin Multiplex Controller (USBPM) 499
18.1 Feature 499
18.2 Input/Output Pins 501
18.3 Register Description 502
18.3.1 EXCPG Control Register (EXCPGCR) 502
18.4 Examples of External Circuit 505
18.4.1 Example of the Connection between USB Function Controller and External Circuit 505
18.4.2 Example of the Connection between USB Host Controller and External Circuit 507
18.5 Usage Notes 508

Section 19 USB Host Controller (USBH) 509
19.1 Features 509
19.2 Input/Output Pins 510
19.3 Register Descriptions 511
19.3.1 HcRevision Register (HREVR) 512
19.3.2 HcControl Register (HCTLR) 513
19.3.3 HcCommandStatus Register (HCSR) 516
19.3.4 HcInterruptStatus Register (HISR) 518
19.3.5 HcInterruptEnable Register (HIER) 521
19.3.6 HcInterruptDisable Register (HIDR) 524
19.3.7 HcHCCA Register (HHCCAR) 526
19.3.8 HcPeriodCurrentED Register (HPCEDR) 527
19.3.9 HcControlHeadED Register (HCHEDR) 527
19.3.10 HcControlCurrentED Register (HCCEDR) 527
19.3.11 HcBulkHeadED Register (HBHEDR) 528
19.3.12 HcBulkCurrentED Register (HBCEDR) 528
19.3.13 HcDoneHeadED Register (HDHEDR) 528
19.3.14 HcFmInterval Register (HFIR) 529
19.3.15 HcFmRemaining Register (HFRR) 531
19.3.16 HcFmNumber Register (HFNR) 532
19.3.17 HcPeriodicStart Register (HPSR) 533
19.3.18 HcLSThreshold Register (HLSTR) 534
19.3.19 HcRhDescriptorA Register (HRDRA) 535
19.3.20 HcRhDescriptorB Register (HRDRB) 537
19.3.21 HcRhStatus Register (HRSR) 538
19.3.22 HcRhPortStatus Register (HRPSR) 540
19.4 Data Storage Format of USB Host Controller 546
19.4.1 Storage Format of Transferred Data 546
19.4.2 Storage Format of Descriptor 546
19.5 Usage Restrictions of USB Host Controller 546
19.5.1 Restriction of Reset Control 546

Section 20 USB Function Controller (USBF) 547
20.1 Features 547
20.2 Input/Output Pins 549
20.3 Register Descriptions 550
20.3.1 Interrupt Flag Register 0 (IFR0) 551
20.3.2 Interrupt Select Register 0 (ISR0) 558
20.3.3 Interrupt Enable Register 0 (IER0) 563
20.3.4 EP0i Data Register (EPDR0i) 566
20.3.5 EP0o Data Register (EPDR0o) 567
20.3.6 EP0s Data Register (EPDR0s) 567

20.3.7 EP1 Data Register (EPDR1) 567
20.3.8 EP2i Data Register (EPDR2i) 568
20.3.9 EP2o Data Register (EPDR2o) 568
20.3.10 EP3i Data Register (EPDR3i) 568
20.3.11 EP3o Data Register (EPDR3o) 569
20.3.12 EP4 Data Register (EPDR4) 569
20.3.13 EP5 Data Register (EPDR5) 569
20.3.14 EP6 Data Register (EPDR6) 570
20.3.15 EP0o Receive Data Size Register (EPSZ0o) 570
20.3.16 EP2o Receive Data Size Register (EPSZ2o) 570
20.3.17 EP3o Receive Data Size Register (EPSZ3o) 570
20.3.18 EP6 Receive Data Size Register (EPSZ6) 571
20.3.19 Trigger Register (TRG) 571
20.3.20 Data Status Register (DASTS) 572
20.3.21 FIFO Clear Register (FCLR) 572
20.3.22 DMA Transfer Setting Register (DMA) 573
20.3.23 Endpoint Stall Register (EPSTL) 574
20.3.24 Configuration Value Register (CVR) 575
20.3.25 Time Stamp Register (TSR) 576
20.3.26 Control Register (CLTR) 576
20.3.27 Endpoint Information Register (EPIRn0 to EPIRn5) 578
20.4 Operation 586
20.4.1 Cable Connection 586
20.4.2 Cable Disconnection 587
20.4.3 Control Transfer 587
20.4.4 EP1, 4 Interrupt-In Transfer 593
20.4.5 EP2i, 5 Bulk-In Transfer (Dual FIFOs) 594
20.4.6 EP2o, 6 Bulk-Out Transfer (Dual FIFOs) 595
20.4.7 EP3i Isochronous-In Transfer 597
20.4.8 EP3o Isochronous Out Transfer 599
20.5 Processing of USB Standard Commands and Class/Vendor Commands 601
20.5.1 Processing of Commands Transmitted by Control Transfer 601
20.6 Stall Operations 602
20.6.1 Overview 602
20.6.2 Forcible Stall by Application 602
20.6.3 Automatic Stall by USB Function Module 604
20.7 Example of External Circuitry for USB Function Controller 605
20.8 Usage Notes 606
20.8.1 Setup Data Reception 606
20.8.2 FIFO Clear 606
20.8.3 Overreading/Overwriting of Data Register 606
20.8.4 Assigning EP0 Interrupt Sources 607
20.8.5 FIFO Clear when DMA Transfer is Set 607

20.8.6 Note on Using TR Interrupt 607
20.8.7 Note on Clearing and Transition to Software Standby Mode 609
20.8.8 Main Clock Usage when Using Bus Password Function 610
20.8.9 Peripheral clock for USB 610

Section 21 Bluetooth Interface (BT) 611
21.1 Features 611
21.2 Input/Output Pins 613
21.3 Register Description 614
21.4 RF-IC Interface 614
21.4.1 Connection with RF ICs 614
21.5 Voice Codec IC Interface 616
21.5.1 Voice Codec (STLC7550) Interface 616
21.5.2 Voice Codec (MC145483) Interface 617
21.6 Low-Frequency Clock Oscillator Interface 618
21.6.1 Using RTCSEL1 Bit to Select Clock Function 618
21.6.2 Frequency-Conversion Circuit Operations 619
21.6.3 Note on Connecting External Crystal Resonator 620
21.7 Power-On Reset/Clock-Resume Control Circuit 621
21.7.1 Power-On Reset 621
21.7.2 Clock-Resume Control 622
21.8 Bluetooth HCI/TCI Commands and API 622

Section 22 D/A Converter (DAC) 623
22.1 Features 623
22.2 Input/Output Pins 624
22.3 Register Descriptions 624
22.3.1 D/A Data Registers 0 and 1 (DADR_0, DADR_1) 624
22.3.2 D/A Control Register (DACR) 625
22.4 Operation 625

Section 23 I/O Ports 627
23.1 Port A 627
23.1.1 Register Description 627
23.1.2 Port A Data Register (PADR) 628
23.2 Port B 629
23.2.1 Register Description 629
23.2.2 Port B Data Register (PBDR) 629
23.3 Port D 631
23.3.1 Register Description 631
23.3.2 Port D Data Register (PDDR) 631
23.4 Port E 633
23.4.1 Register Description 633

23.4.2 Port E Data Register (PEDR) ... 633
23.5 Port G ... 635
23.5.1 Register Description ... 635
23.5.2 Port G Data Register (PGDR) ... 636
23.6 Port H ... 637
23.6.1 Register Description ... 637
23.6.2 Port H Data Register (PHDR) ... 637

Section 24 Pin Function Controller (PFC) ... 639
24.1 Overview ... 639
24.2 Register Descriptions ... 641
24.2.1 Port A Control Register (PACR) ... 641
24.2.2 Port B Control Register (PBCR) ... 643
24.2.3 Port D Control Register (PDCR) ... 644
24.2.4 Port E Control Register (PECR) ... 646
24.2.5 Port G Control Register (PGCR) ... 647
24.2.6 Port H Control Register (PHCR) ... 649
24.2.7 Pin Select Register A (PSELA) ... 650
24.2.8 I/O Buffer Hi-Z Control Register A (HIZCRA) ... 652
24.2.9 Noise Canceller Control Register (NCCR) ... 654

Section 25 User Break Controller ... 655
25.1 Features ... 655
25.2 Register Descriptions ... 657
25.2.1 Break Address Register A (BARA) ... 657
25.2.2 Break Address Mask Register A (BAMRA) ... 658
25.2.3 Break Bus Cycle Register A (BBRA) ... 658
25.2.4 Break Address Register B (BARB) ... 660
25.2.5 Break Address Mask Register B (BAMRB) ... 661
25.2.6 Break Data Register B (BDRB) ... 661
25.2.7 Break Data Mask Register B (BDMRB) ... 662
25.2.8 Break Bus Cycle Register B (BBRB) ... 663
25.2.9 Break Control Register (BRCR) ... 664
25.2.10 Execution Times Break Register (BETR) ... 667
25.2.11 Branch Source Register (BRSR) ... 669
25.2.12 Branch Destination Register (BRDR) ... 670
25.3 Operation ... 671
25.3.1 Flow of the User Break Operation ... 671
25.3.2 Break on Instruction Fetch Cycle ... 672
25.3.3 Break on Data Access Cycle ... 673
25.3.4 Break on X/Y-Memory Bus Cycle ... 674
25.3.5 Sequential Break ... 674
25.3.6 Value of Saved Program Counter ... 675

25.3.7 PC Trace 676
25.3.8 Usage Examples 676
25.4 Usage Notes 681

Section 26 User Debugging Interface (H-UDI) 683
26.1 Features 683
26.2 Input/Output Pins 684
26.3 Register Descriptions 685
26.3.1 Bypass Register (SDBPR) 685
26.3.2 Instruction Register (SDIR) 685
26.3.3 Boundary Scan Register (SDBSR) 686
26.3.4 ID Register (SDID) 691
26.4 Operation 692
26.4.1 TAP Controller 692
26.4.2 Reset Configuration 693
26.4.3 TDO Output Timing 693
26.5 Boundary Scan 694
26.5.1 Supported Instructions 694
26.5.2 Cautions 695
26.6 Usage Notes 696

Section 27 List of Registers 697
27.1 Register Addresses
(by functional module, in order of the corresponding section numbers) 698
27.2 Register Bits 705
27.3 Register States in Each Operating Mode 722

Section 28 Electrical Characteristics 729
28.1 Absolute Maximum Ratings 729
28.2 DC Characteristics 731
28.3 AC Characteristics 735
28.3.1 Clock Timing 736
28.3.2 Control Signal Timing 740
28.3.3 AC Bus Timing 743
28.3.4 Basic Timing 745
28.3.5 Burst ROM Timing 752
28.3.6 Synchronous DRAM Timing 753
28.3.7 Peripheral Module Signal Timing 776
28.3.8 SIOF Module Signal Timing 777
28.3.9 SCIF Module Signal Timing 781
28.3.10 USB Module Signal Timing 782
28.3.11 USB Transceiver Timing 783
28.3.12 Bluetooth Interface (BT) Timing 785

28.3.13 AC Characteristic Test Conditions 789
28.4 D/A Converter Characteristics 790

Appendix 791
A. Pin States 791
B. Usage Notes for Oscillator circuit with External Crystal Resonator 795
B.1 Recommended Oscillator Circuit 795
B.2 Note on PCB design 796
C. Package Dimensions 797

Index 799

Figures

Section 1 Overview
Figure 1.1 Block Diagram of SH7660 8
Figure 1.2 Pin Assignment 9

Section 2 CPU
Figure 2.1 Processing State Transitions 30
Figure 2.2 Logical Address Space 32
Figure 2.3 P4 Area 33
Figure 2.4 Physical Address Space 34
Figure 2.5 External Address Space and Mounted Space (Area 0) 35
Figure 2.6 Register Configuration in Each Processing Mode 37
Figure 2.7 General Registers 39
Figure 2.8 System Registers and Program Counter 40
Figure 2.9 Control Register Configuration 44
Figure 2.10 Data Format on Memory (Big Endian Mode) 46
Figure 2.11 Data Format on Memory (Little Endian Mode) 46

Section 3 DSP Operating Unit
Figure 3.1 DSP Instruction Format 74
Figure 3.2 CPU Registers in DSP Mode 76
Figure 3.3 DSP Register Configuration 80
Figure 3.4 DSP Registers and Bus Connections 97
Figure 3.5 General Registers (DSP Mode) 100
Figure 3.6 Sample Parallel DSP Instruction Program 113
Figure 3.7 Examples of Conditional Operations and Data Transfer Instructions 115
Figure 3.8 Data Formats 117
Figure 3.9 ALU Fixed-Point Arithmetic Operation Flow 118
Figure 3.10 Operation Sequence Example 120
Figure 3.11 DC Bit Generation Examples in Carry or Borrow Mode 121
Figure 3.12 DC Bit Generation Examples in Negative Value Mode 121
Figure 3.13 DC Bit Generation Examples in Overflow Mode 122
Figure 3.14 ALU Integer Arithmetic Operation Flow 123
Figure 3.15 ALU Logical Operation Flow 125
Figure 3.16 Fixed-Point Multiply Operation Flow 127
Figure 3.17 Arithmetic Shift Operation Flow 129
Figure 3.18 Logical Shift Operation Flow 131
Figure 3.19 PDMSB Operation Flow 133
Figure 3.20 Rounding Operation Flow 135
Figure 3.21 Definition of Rounding Operation 136
Figure 3.22 Local Data Move Instruction Flow 137

Section 4 Exception Handling
Figure 4.1 Register Bit Configuration 156

Section 5 Cache
Figure 5.1 Cache Structure 177
Figure 5.2 Cache Search Scheme 183
Figure 5.3 Write-Back Buffer Configuration 185
Figure 5.4 Specifying Address and Data for Memory-Mapped Cache Access 188

Section 8 Interrupt Controller (INTC)
Figure 8.1 Block Diagram of INTC 200
Figure 8.2 Interrupt Operation Flowchart 215

Section 9 Bus State Controller (BSC)
Figure 9.1 BSC Functional Block Diagram 219
Figure 9.2 Logical Address Space and Physical Address Space 221
Figure 9.3 Normal Space Basic Access Timing (Access Wait 0) 251
Figure 9.4 Continuous Access for Normal Space 1 Data Bus Width = 16 bits, Long-Word Access, CSnWCR.WN Bit = 0 (Access Wait = 0, Cycle Wait = 0) 253
Figure 9.5 Continuous Access for Normal Space 2 Data Bus Width = 16 bits, Long-Word Access, CSnWCR.WN Bit = 1 (Access Wait = 0, Cycle Wait = 0) 254
Figure 9.6 Example of 16-Bit Data-Width SRAM Connection 255
Figure 9.7 Example of 8-Bit Data-Width SRAM Connection 255
Figure 9.8 Wait Timing for Normal Space Access (Software Wait Only) 256
Figure 9.9 Wait State Timing for Normal Space Access (Wait State Insertion using $\overline{\text{WAIT}}$ Signal) 257
Figure 9.10 $\overline{\text{CSn}}$ Assert Period Expansion 258
Figure 9.11 Example of 16-Bit Data-Width SDRAM Connection 260
Figure 9.12 Burst Read Basic Timing (Auto Pre-charge) 268
Figure 9.13 Burst Read Wait Specification Timing (Auto Pre-charge) 268
Figure 9.14 Single Read Wait Specification Timing (Auto Pre-charge) 269
Figure 9.15 Basic Timing for Synchronous DRAM Burst Write (Auto Pre-charge) 270
Figure 9.16 Single Write Basic Timing (Auto-Precharge) 271
Figure 9.17 Burst Read Timing (No Auto Precharge) 273
Figure 9.18 Burst Read Timing (Bank Active, Same Row Address) 273
Figure 9.19 Burst Read Timing (Bank Active, Different Row Addresses) 274
Figure 9.20 Single Write Timing (No Auto Precharge) 275
Figure 9.21 Single Write Timing (Bank Active, Same Row Address) 276
Figure 9.22 Single Write Timing (Bank Active, Different Row Addresses) 277
Figure 9.23 Auto-Refresh Timing 278
Figure 9.24 Self-Refresh Timing 280
Figure 9.25 Low-Frequency Mode Access Timing 281
Figure 9.26 Power-Down Mode Access Timing 282
Figure 9.27 Synchronous DRAM Mode Write Timing (Based on JEDEC) 284

Figure 9.28 Burst ROM Access (Data Bus Width 8 Bits, 4-byte Transfer (number of burst 4), Access Wait for the 1st time 2, Access Wait for 2nd Time and after 1) 286
Figure 9.29 Byte-Selection SRAM Basic Access Timing (BAS = 0) 288
Figure 9.30 Byte-Selection RAM Basic Access Timing (BAS = 1) 289
Figure 9.31 Example of Connection with 16-Bit Data-Width Byte-Selection SRAM 289
Figure 9.32 Waveform in the Event of a Problem 290
Figure 9.33 Bus Arbitration Timing (Master Mode) 293
Figure 9.34 Master and Slave Connection Example 294

Section 10 Direct Memory Access Controller (DMAC)

Figure 10.1 Block Diagram of the DMAC 298
Figure 10.2 DMA Transfer Flowchart 314
Figure 10.3 Round-Robin Mode 319
Figure 10.4 Changes in Channel Priority in Round-Robin Mode 320
Figure 10.5 Data Flow of Dual Address Mode 322
Figure 10.6 Example of DMA Transfer Timing in Dual Mode (Source: Ordinary memory, Destination: Ordinary memory) 323
Figure 10.7 Data Flow in Single Address Mode 324
Figure 10.8 Example of DMA Transfer Timing in Single Address Mode 324
Figure 10.9 DMA Transfer Example in the Cycle-Steal Normal Mode (Dual Address, DREQ Low Level Detection) 325
Figure 10.10 Example of DMA Transfer in Cycle Steal Intermittent Mode (Dual address, DREQ low level detection) 326
Figure 10.11 DMA Transfer Example in the Burst Mode (Dual Address, DREQ low level detection) 326
Figure 10.12 Bus State when Multiple Channels Are Operating 328
Figure 10.13 Example of DREQ Input Detection in Cycle Steal Mode Edge Detection 329
Figure 10.14 Example of DREQ Input Detection in Cycle Steal Mode Level Detection 329
Figure 10.15 Example of DREQ Input Detection in Burst Mode Edge Detection 329
Figure 10.16 Example of DREQ Input Detection in Burst Mode Level Detection 330
Figure 10.17 Example of DMA Transfer End Signal Timing in Cycle Steel Mode Level Detection 330
Figure 10.18 BSC Ordinary Memory Access (No wait, Idle Cycle 1, Longword Access to 16-bit Device) 331

Section 11 Clock Pulse Generator (CPG)

Figure 11.1 Block Diagram of Clock Pulse Generator 334
Figure 11.2 Note on Using a Crystal Resonator 344
Figure 11.3 Note on Using a PLL Oscillator Circuit 345

Section 12 Watchdog Timer (WDT)

Figure 12.1 Block Diagram of the WDT 348
Figure 12.2 Writing to WTCNT and WTCSR 352

Section 13 Power-Down Modes
Figure 13.1 Canceling Standby Mode with STBCR.STBY 368
Figure 13.2 Transitions between Modes 369

Section 14 Timer Unit (TMU)
Figure 14.1 TMU Block Diagram 372
Figure 14.2 Setting Count Operation 377
Figure 14.3 Auto-Reload Count Operation 378
Figure 14.4 Count Timing when Internal Clock Is Operating 378
Figure 14.5 UNF Set Timing 379
Figure 14.6 Status Flag Clear Timing 379

Section 15 Serial I/O with FIFO (SIOF)
Figure 15.1 Block Diagram of SIOF 382
Figure 15.2 Serial Clock Supply 410
Figure 15.3 Serial Data Synchronization Timing 411
Figure 15.4 SIOF Transmit/Receive Timing 412
Figure 15.5 Transmit/Receive Data Bit Alignment 414
Figure 15.6 Control Data Bit Alignment 415
Figure 15.7 Control Data Interface (Slot Position) 416
Figure 15.8 Control Data Interface (Secondary FS) 417
Figure 15.9 Example of Transmit Operation in Master Mode 419
Figure 15.10 Example of Receive Operation in Master Mode 420
Figure 15.11 Example of Transmit Operation in Slave Mode 421
Figure 15.12 Example of Receive Operation in Slave Mode 422
Figure 15.13 Transmit and Receive Timing (8-Bit Monaural Data (1)) 426
Figure 15.14 Transmit and Receive Timing (8-Bit Monaural Data (2)) 426
Figure 15.15 Transmit and Receive Timing (16-Bit Monaural Data (1)) 427
Figure 15.16 Transmit and Receive Timing (16-Bit Stereo Data (1)) 427
Figure 15.17 Transmit and Receive Timing (16-Bit Stereo Data (2)) 428
Figure 15.18 Transmit and Receive Timing (16-Bit Stereo Data (3)) 428
Figure 15.19 Transmit and Receive Timing (16-Bit Stereo Data (4)) 429
Figure 15.20 Transmit and Receive Timing (16-Bit Stereo Data) 429
Figure 15.21 Example of Configuration in SPI Mode 430
Figure 15.22 SPI Data/Clock Timing 1 (CPHA = 0) 432
Figure 15.23 SPI Data/Clock Timing 2 (CPHA = 1) 432

Section 16 Serial Communication Interface with FIFO (SCIF)
Figure 16.1 Block Diagram of SCIF 437
Figure 16.2 Sample SCIF Initialization Flowchart 466
Figure 16.3 Sample Serial Transmission Flowchart 467
Figure 16.4 Example of Transmit Operation
(Example with 8-Bit Data, Parity, One Stop Bit) 469
Figure 16.5 Example of Transmit Data Stop Function 469

Figure 16.6 Transmit Data Stop Function Flowchart 470
Figure 16.7 Sample Serial Reception Flowchart (1) 471
Figure 16.8 Sample Serial Reception Flowchart (2) 472
Figure 16.9 Example of SCIF Receive Operation
(Example with 8-Bit Data, Parity, One Stop Bit) 474
Figure 16.10 $\overline{\text{CTS}}$ Control Operation 474
Figure 16.11 $\overline{\text{RTS}}$ Control Operation 475
Figure 16.12 Data Format in Clock Synchronous Communication 476
Figure 16.13 Sample SCIF Initialization Flowchart (1) (Transmission) 477
Figure 16.13 Sample SCIF Initialization Flowchart (2) (Reception) 478
Figure 16.13 Sample SCIF Initialization Flowchart (3)
(Simultaneous Transmission and Reception) 479
Figure 16.14 Sample Serial Transmission Flowchart (1)
(First Transmission after Initialization) 480
Figure 16.14 Sample Serial Transmission Flowchart (2)
(Second and Subsequent Transmission) 480
Figure 16.15 Sample Serial Reception Flowchart (1)
(First Reception after Initialization) 481
Figure 16.15 Sample Serial Reception Flowchart (2)
(Second and Subsequent Reception) 482
Figure 16.16 Sample Simultaneous Serial Transmission and Reception Flowchart (1)
(First Transfer after Initialization) 483
Figure 16.16 Sample Simultaneous Serial Transmission and Reception Flowchart (2)
(Second and Subsequent Transfer) 484
Figure 16.17 Receive Data Sampling Timing in Asynchronous Mode 488

Section 17 Boot Function (BOOT)
Figure 17.1 Block Diagram of Boot Function 492
Figure 17.2 External Memory Space in Boot Mode 494
Figure 17.3 Procedure for Execution of Boot Processing 497

Section 18 USB Pin Multiplex Controller (USBPM)
Figure 18.1 Block Diagram of this USB 500
Figure 18.2 Example 1 of Connection between USB Function Controller
and External Circuit 505
Figure 18.3 Example 2 of Connection between USB Function Controller
and External Circuit 506
Figure 18.4 Example of Connection USB Host Controller and External Circuit 507

Section 20 USB Function Controller (USBF)
Figure 20.1 Block Diagram of USB Function Controller 549
Figure 20.2 Example of Endpoint Configuration (1) 582
Figure 20.3 Example of Endpoint Configuration (2) 585
Figure 20.4 Cable Connection Operation 586

Figure 20.5 Cable Disconnection Operation 587
Figure 20.6 Transfer Stages in Control Transfer 587
Figure 20.7 Setup Stage Operation 588
Figure 20.8 Data Stage (Control-In) Operation 589
Figure 20.9 Data Stage (Control-Out) Operation 590
Figure 20.10 Status Stage (Control-In) Operation 591
Figure 20.11 Status Stage (Control-Out) Operation 592
Figure 20.12 EP1 Interrupt-In Transfer Operation 593
Figure 20.13 EP2 Bulk-In Transfer Operation 594
Figure 20.14 EP2o Bulk-Out Transfer Operation 595
Figure 20.15 Operation of Isochronous-In Transfer 597
Figure 20.16 Operation of EP3o Isochronous-Out Transfer 599
Figure 20.17 Forcible Stall by Application 603
Figure 20.18 Automatic Stall by USB Function Module 604
Figure 20.19 Example of USB Function Module External Circuitry 605
Figure 20.20 Set Timing of TR Interrupt Flag 608
Figure 20.21 Example Flow of Clearing and Transition to Software Standby Mode 609

Section 21 Bluetooth Interface (BT)

Figure 21.1 Block Diagram of Bluetooth Interface (BT) 612
Figure 21.2 Example of Connection to RF IC 614
Figure 21.3 Example of Connection to Voice Codec (STLC7550) 616
Figure 21.4 (1) Example of Connection to Voice Codec (MC145483) 617
Figure 21.4 (2) Timing chart for connection to Voice Codec (MC145483) 617
Figure 21.5 Function of Frequency-Conversion Circuit 619
Figure 21.6 Errors in Pseudo-32-kHz Clock Obtained by Conversion from 32.768-kHz Clock 620
Figure 21.7 Note on Using Crystal Resonator 621
Figure 21.8 Timing Chart of Reset Signal 622

Section 22 D/A Converter (DAC)

Figure 22.1 Block Diagram of DAC 623
Figure 22.2 D/A Converter Operation Example 626

Section 23 I/O Ports

Figure 23.1 Port A 627
Figure 23.2 Port B 629
Figure 23.3 Port D 631
Figure 23.4 Port E 633
Figure 23.5 Port G 635
Figure 23.6 Port H 637

Section 25 User Break Controller
Figure 25.1 Block Diagram of User Break Controller 656

Section 26 User Debugging Interface (H-UDI)
Figure 26.1 Block Diagram of H-UDI 683
Figure 26.2 TAP Controller State Transitions 692
Figure 26.3 H-UDI Data Transfer Timing 693
Figure 26.4 Example of Connecting Reset Signals without Mutual Interference 696

Section 28 Electrical Characteristics
Figure 28.1 EXTAL Clock Input Timing 737
Figure 28.2 CKIO Clock Input Timing 737
Figure 28.3 CKIO Clock Output Timing 737
Figure 28.4 Power-On Oscillation Settling Time 737
Figure 28.5 Oscillation Settling Time on Return from Standby (Return by Reset) 738
Figure 28.6 Oscillation Settling Time on Return from Standby (Return by NMI or IRQ) 738
Figure 28.7 PLL Synchronization Settling Time at Power-On Reset 738
Figure 28.8 PLL Synchronization Settling Time in Case of Reset or NMI Interrupt 739
Figure 28.9 Reset and $\overline{\text{BOOT-E}}$ Input Timing 741
Figure 28.10 Interrupt Signal Input Timing 741
Figure 28.11 $\overline{\text{IRQOUT}}$ Timing 741
Figure 28.12 $\overline{\text{REFOUT}}$ Timing 742
Figure 28.13 Bus Release Timing 742
Figure 28.14 Pin Drive Timing at Standby 742
Figure 28.15 Basic Bus Cycle in Normal Space (No Wait) 745
Figure 28.16 Basic Bus Cycle in Normal Space (Software Wait 1) 746
Figure 28.17 Basic Bus Cycle in Normal Space (Asynchronous External Wait 1 Input) 747
Figure 28.18 Basic Bus Cycle in Normal Space (Software Wait 1, Asynchronous External Wait Valid (WM Bit = 0), No Idle Cycle) 748
Figure 28.19 CS Extended Bus Cycle in Normal Space (SW = 1 Cycle, HW = 1 Cycle, Asynchronous External Wait 1 Input) 749
Figure 28.20 Bus Cycle of SRAM with Byte Selection (SW = 1 Cycle, HW = 1 Cycle, Asynchronous External Wait 1 Input, BAS = 0 (UB and LB in Write Cycle Controlled)) 750
Figure 28.21 Bus Cycle of SRAM with Byte Selection (SW = 1 Cycle, HW = 1 Cycle, Asynchronous External Wait 1 Input, BAS = 1 (WE in Write Cycle Controlled)) 751
Figure 28.22 Read Bus Cycle of Burst ROM (Software Wait 1, Asynchronous External Wait 1 Input, Burst Wait 1, Number of Burst 2) 752
Figure 28.23 Single Read Bus Cycle of Synchronous DRAM (Auto Precharge Mode, CAS Latency 2, TRCD = 1 Cycle, TRP = 1 Cycle) 753
Figure 28.24 Single Read Bus Cycle of Synchronous DRAM (Auto Precharge Mode, CAS Latency 2, TRCD = 2 Cycle, TRP = 2 Cycle) 754

Figure 28.25 Burst Read Bus Cycle of Synchronous DRAM (Single Read × 4)
(Auto Precharge Mode, CAS Latency 2, TRCD = 1 Cycle, TRP = 2 Cycle) 755
Figure 28.26 Burst Read Bus Cycle of Synchronous DRAM (Single Read × 4)
(Auto Precharge Mode, CAS Latency 2, TRCD = 2 Cycle, TRP = 1 Cycle) 756
Figure 28.27 Single Write Bus Cycle of Synchronous DRAM
(Auto Precharge Mode, TRWL = 1 Cycle) 757
Figure 28.28 Single Write Bus Cycle of Synchronous DRAM
(Auto Precharge Mode, TRCD = 3 Cycle, TRWL = 1 Cycle) 758
Figure 28.29 Burst Write Bus Cycle of Synchronous DRAM (Single Write × 4)
(Auto Precharge Mode, TRCD = 1 Cycle, TRWL = 1 Cycle) 759
Figure 28.30 Burst Write Bus Cycle of Synchronous DRAM (Single Write × 4)
(Auto Precharge Mode, TRCD = 1 Cycle, TRWL = 1 Cycle) 760
Figure 28.31 Burst Read Bus Cycle of Synchronous DRAM (Single Read × 4)
(Bank Active Mode: ACTV + READ Command, CAS Latency 2,
TRCD = 1 Cycle) 761
Figure 28.32 Burst Read Bus Cycle of Synchronous DRAM (Single Read × 4)
(Bank Active Mode: READ Command, Same Row Address, CAS Latency 2,
TRCD = 1 Cycle) 762
Figure 28.33 Burst Read Bus Cycle of Synchronous DRAM (Single Read × 4)
(Bank Active Mode: PRE + ACTV + READ Command, Different Row Address,
CAS Latency 2, TRCD = 1 Cycle) 763
Figure 28.34 Burst Write Bus Cycle of Synchronous DRAM (Single Write × 4)
(Bank Active Mode: ACTV + WRIT Command, TRCD = 1 Cycle) 764
Figure 28.35 Burst Write Bus Cycle of Synchronous DRAM (Single Write × 4)
(Bank Active Mode: ACTV + WRIT Command, TRCD = 1 Cycle) 765
Figure 28.36 Burst Write Bus Cycle of Synchronous DRAM (Single Write × 4)
(Bank Active Mode: PRE + ACTV + WRIT Command, TRCD = 1 Cycle) 766
Figure 28.37 Auto Refresh Timing of Synchronous DRAM (TRP = 2 Cycle) 767
Figure 28.38 Self Refresh Timing of Synchronous DRAM (TRP = 2 Cycle) 768
Figure 28.39 Power-On Sequence of Synchronous DRAM
(Mode Write Timing, TRP = 2 Cycle) 769
Figure 28.40 Access Timing in Low-Frequency Mode of Synchronous DRAM
(Auto Precharge Mode, TRWL = 1 Cycle) 770
Figure 28.41 Auto Refresh Timing in Low-Frequency Mode of Synchronous DRAM
(TRP = 2 Cycle) 771
Figure 28.42 Self Refresh Timing in Low-Frequency Mode of Synchronous DRAM
(TRP = 2 Cycle) 772
Figure 28.43 Power-On Sequence in Low-Frequency Mode of Synchronous DRAM
(Mode Write Timing, TRP = 2 Cycle) 773
Figure 28.44 Write to Read Bus Cycle in Power-Down Mode of Synchronous DRAM
(Auto Precharge Mode, TRCD = 1 Cycle, TRP = 1 Cycle, TRWL = 1 Cycle) 774
Figure 28.45 Read to Write Bus Cycle in Power-Down Mode of Synchronous DRAM
(Auto Precharge Mode, TRCD = 1 Cycle, TRP = 1 Cycle, TRWL = 1 Cycle) 775

Figure 28.46 I/O Port Timing 776
Figure 28.47 DREQ Input Timing (DREQ Low Level is Detected) 776
Figure 28.48 DACK and TEND Output Timing 776
Figure 28.49 SIOF_MCLK Input Timing 777
Figure 28.50 SIOF Transmission/Reception Timing (Master Mode 1, Falling Edge Sampling) 778
Figure 28.51 SIOF Transmission/Reception Timing (Master Mode 1, Rising Edge Sampling) 778
Figure 28.52 SIOF Transmission/Reception Timing (Master Mode 2, Falling Edge Sampling) 779
Figure 28.53 SIOF Transmission/Reception Timing (Master Mode 2, Rising Edge Sampling) 779
Figure 28.54 SIOF Transmission/Reception Timing (Slave Mode 1, Slave Mode 2) 780
Figure 28.55 SIOF Transmission/Reception Timing (SPI Mode, CPHA = 0, SSAST1 and SSAST0 = 01) 780
Figure 28.56 SIOF Transmission/Reception Timing (SPI Mode, CPHA = 1, SSAST1 and SSAST0 = 01) 781
Figure 28.57 SCIF Input Clock Cycle 782
Figure 28.58 Input/Output Timing in SCIF Synchronous Mode 782
Figure 28.59 USB Clock Timing 783
Figure 28.60 USB Transceiver Timing 784
Figure 28.61 USB Transceiver Characteristics Testing Circuit (Full-Speed) 784
Figure 28.62 USB Transceiver Characteristics Testing Circuit (Low-Speed) 784
Figure 28.63 Bluetooth Module Clock Timing 785
Figure 28.64 Bluetooth Interface Module Low Power Clock Timing 786
Figure 28.65 Bluetooth Interface (BT) Voice CODEC (STLC7550) Interface Signal Timing 787
Figure 28.66 Bluetooth Interface (BT) Voice CODEC (MC145483) Interface Signal Timing 787
Figure 28.67 SPI Interface Signal Timing for Bluetooth Interface (BT) RF 788
Figure 28.68 Bluetooth Interface (BT) Receive Data Timing 788
Figure 28.69 Output Load Circuit 789

Appendix

Figure C.1 Package Dimensions (TBP-208A) 797

Tables

Section 1 Overview

Table 1.1 SH7660 Features 2
Table 1.2 Pin Functions and Initial Values 10
Table 1.3 Pin Functions 21

Section 2 CPU

Table 2.1 Register Initial Values 36
Table 2.2 Addressing Modes and Effective Addresses 49
Table 2.3 Instruction Formats 53
Table 2.4 CPU Instruction Types 56
Table 2.5 Data Transfer Instructions 60
Table 2.6 Arithmetic Operation Instructions 62
Table 2.7 Logic Operation Instructions 64
Table 2.8 Shift Instructions 65
Table 2.9 Branch Instructions 66
Table 2.10 System Control Instructions 67
Table 2.11 Operation Code Map 70

Section 3 DSP Operating Unit

Table 3.1 Virtual Address Space 75
Table 3.2 Operation of SR Bits in Each Processing Mode 79
Table 3.3 RS and RE Setting Rule 85
Table 3.4 Repeat Control Instructions 85
Table 3.5 Repeat Control Macros 86
Table 3.6 DSP Mode Extended System Control Instructions 87
Table 3.7 PC Value during Repeat Control (When RC[11:0] ≥ 2) 90
Table 3.8 Extended Repeat Control Instructions 94
Table 3.9 Extended System Control Instructions in DSP Mode 99
Table 3.10 Overview of Data Transfer Instructions 101
Table 3.11 Modulo Addressing Control Instructions 103
Table 3.12 Double Data Transfer Instruction Formats 105
Table 3.13 Single Data Transfer Instruction Formats 106
Table 3.14 Destination Register in DSP Instructions 108
Table 3.15 Source Register in DSP Operations 108
Table 3.16 DSR Register Bits 109
Table 3.17 DSP Instruction Formats 112
Table 3.18 Correspondence between DSP Instruction Operands and Registers 112
Table 3.19 DC Bit Update Definitions 114
Table 3.20 Examples of NOPX and NOPY Instruction Codes 116
Table 3.21 Variation of ALU Fixed-Point Operations 119
Table 3.22 Correspondence between Operands and Registers 119

Table 3.23 Variation of ALU Integer Operations 124
Table 3.24 Variation of ALU Logical Operations 125
Table 3.25 Variation of Fixed-Point Multiply Operation 127
Table 3.26 Correspondence between Operands and Registers 127
Table 3.27 Variation of Shift Operations 129
Table 3.28 Operation Definition of PDMSB 134
Table 3.29 Variation of PDMSB Operation 134
Table 3.30 Variation of Rounding Operation 136
Table 3.31 Definition of Overflow Protection for Fixed-Point Arithmetic Operations 136
Table 3.32 Definition of Overflow Protection for Integer Arithmetic Operations 137
Table 3.33 Variation of Local Data Move Operations 137
Table 3.34 Correspondence between Operands and Registers 138
Table 3.35 DSP Mode Extended System Control Instructions 139
Table 3.36 Double Data Transfer Instruction 141
Table 3.37 Single Data Transfer Instructions 142
Table 3.38 Correspondence between DSP Data Transfer Operands and Registers 143
Table 3.39 DSP Data Operation Instructions 144
Table 3.40 Operation Code Map 150

Section 4 Exception Handling

Table 4.1 Exception Event Vectors 162
Table 4.2 Instruction Positions regarding a repeat loop and Restriction Types 169
Table 4.3 SPC Value when a Re-Execution Type Exception Occurs in Repeat Control 171
Table 4.4 Restrictions of Exception Acceptance in the Repeat Loop 173
Table 4.5 Instruction Where a Specific Exception Occurs When a Memory Access Exception Occurs in Repeat Control (SR.RC[11:0] ≥ 1) 174

Section 5 Cache

Table 5.1 LRU and Way Replacement (when Cache Locking Mechanism is Disabled) 178
Table 5.2 Way Replacement when a PREF Instruction Misses the Cache 181
Table 5.3 Way Replacement when Instructions other than the PREF Instruction Miss the Cache 182
Table 5.4 LRU and Way Replacement (when W2LOCK = 1 and W3LOCK =0) 182
Table 5.5 LRU and Way Replacement (when W2LOCK = 0 and W3LOCK =1) 182
Table 5.6 LRU and Way Replacement (when W2LOCK = 1 and W3LOCK =1) 182

Section 6 X/Y Memory

Table 6.1 X/Y Memory Virtual Addresses 191
Table 6.2 Cache Settings 194

Section 7 U Memory

Table 7.1 U Memory Virtual Addresses 195
Table 7.2 Cache Settings 197

Section 8 Interrupt Controller (INTC)
Table 8.1 Pin Configuration ... 201
Table 8.2 Interrupt Sources and IPRA to IPRH ... 203
Table 8.3 Correspondence between Interrupt Sources and IMR0 to IMR9/IMCR0 to IMCR9 ... 209
Table 8.4 Interrupt Exception Handling Sources and Priority ... 213

Section 9 Bus State Controller (BSC)
Table 9.1 Pin Configuration ... 220
Table 9.2 Address Space Map of External Address Space ... 223
Table 9.3 16-Bit External Device/Big Endian Access and Data Alignment ... 248
Table 9.4 8-Bit External Device/Big Endian Access and Data Alignment ... 249
Table 9.5 16-Bit External Device/Little Endian Access and Data Alignment ... 249
Table 9.6 8-Bit External Device/Little Endian Access and Data Alignment ... 250
Table 9.7 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (1)-1 ... 261
Table 9.7 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (1)-2 ... 262
Table 9.8 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (2)-1 ... 263
Table 9.8 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (2)-2 ... 264
Table 9.9 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (3)-1 ... 265
Table 9.9 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (3)-2 ... 266
Table 9.10 Relationship between Access Size and Number of Bursts ... 267
Table 9.11 Access Address in SDRAM Mode Register Write ... 283
Table 9.12 Relationship between Data Bus Width, Access Size, and Number of Bursts ... 285

Section 10 Direct Memory Access Controller (DMAC)
Table 10.1 Pin Configuration ... 299
Table 10.2 DMARS Settings ... 312
Table 10.3 Selecting External Request Modes with the RS Bits ... 315
Table 10.4 Selecting External Request Detection with DL, DS Bits ... 316
Table 10.5 Selecting External Request Detection with DO Bit ... 316
Table 10.6 Selecting On-Chip Peripheral Module Request Modes with the RS3 to RS0 Bits ... 317
Table 10.7 Supported DMA Transfers ... 321
Table 10.8 Relationship of Request Modes and Bus Modes by DMA Transfer Category ... 327

Section 11 Clock Pulse Generator (CPG)
Table 11.1 Clock Pulse Generator Pins Configuration ... 336
Table 11.2 Clock Operating Modes ... 337

Table 11.3 Possible Combination of Clock Modes and FRQCR Values 338

Section 13 Power-Down Modes
Table 13.1 States of Power-Down Modes 356
Table 13.2 Pin Configuration 358
Table 13.3 Register States in Software Standby Mode 367

Section 14 Timer Unit (TMU)
Table 14.1 TMU Interrupt Sources 380

Section 15 Serial I/O with FIFO (SIOF)
Table 15.1 Pin Configuration 383
Table 15.2 Operation in Each Transfer Mode 386
Table 15.3 SIOF Serial Clock Frequency 410
Table 15.4 Serial Transfer Modes 412
Table 15.5 Frame Length 413
Table 15.6 Audio Mode Specification for Transmit Data 414
Table 15.7 Audio Mode Specification for Receive Data 415
Table 15.8 The Number of Channels in Control Data Setting 415
Table 15.9 Conditions to Issue Transmit Request 417
Table 15.10 Conditions to Issue Receive Request 418
Table 15.11 Transmit and Receive Reset 423
Table 15.12 SIOF Interrupt Sources 424
Table 15.13 States of Transmit and Receive Operations in SPI Mode 431

Section 16 Serial Communication Interface with FIFO (SCIF)
Table 16.1 Pin Configuration 438
Table 16.2 SCSMR Settings for Serial Transfer Format Selection 463
Table 16.3 Serial Transfer Formats 464
Table 16.4 SCIF Interrupt Sources 486

Section 17 Boot Function (BOOT)
Table 17.1 Input/Output Pin 493
Table 17.2 Clock frequency and data transfer rate of SCIF when boot function is used 498

Section 18 USB Pin Multiplex Controller (USBPM)
Table 18.1 Pin Configuration (Analog Transceiver Signal) 501
Table 18.2 Pin Configuration (Power Control signal) 501
Table 18.3 Pin Configuration (Clock Control signal) 501

Section 19 USB Host Controller (USBH)
Table 19.1 Pin Configuration 510

Section 20 USB Function Controller (USBF)
Table 20.1 Pin Configuration 549
Table 20.2 Restrictions of Settable Values 581
Table 20.3 Example of Endpoint Configuration (1) 581

Table 20.4 Example of Setting of Endpoint Configuration Information (1) 583
Table 20.5 Example of Endpoint Configuration (2) 584
Table 20.6 Example of Setting of Endpoint Configuration Information (2) 585
Table 20.7 Command Decoding on Application Side 601

Section 21 Bluetooth Interface (BT)
Table 21.1 Input/Output Pins 613
Table 21.2 Settings and Functions 618

Section 22 D/A Converter (DAC)
Table 22.1 Pin Configuration 624

Section 23 I/O Ports
Table 23.1 Port A Data Register (PADR) Read/Write Operations 628
Table 23.2 Port B Data Register (PBDR) Read/Write Operations 630
Table 23.3 Port D Data Register (PDDR) Read/Write Operations 632
Table 23.4 Port E Data Register (PEDR) Read/Write Operations 634
Table 23.5 Port G Data Register (PGDR) Read/Write Operations 636
Table 23.6 Port H Data Register (PHDR) Read/Write Operations 638

Section 24 Pin Function Controller (PFC)
Table 24.1 Multiplex Pins 639

Section 25 User Break Controller
Table 25.1 Specifying Break Address Register 660
Table 25.2 Specifying Break Data Register 662
Table 25.3 Data Access Cycle Addresses and Operand Size Comparison Conditions 673

Section 26 User Debugging Interface (H-UDI)
Table 26.1 Pin Configuration 684
Table 26.2 JTAG Commands 686
Table 26.3 Correspondence between Pins of SH7660 and Boundary Scan Register 687
Table 26.4 Reset Configuration 693

Section 28 Electrical Characteristics
Table 28.1 Absolute Maximum Ratings 729
Table 28.2 DC Characteristics (1) Common Items 731
Table 28.2 DC Characteristics (2-a) Except DAC and USB Related Pins 732
Table 28.2 DC Characteristics (2-b) USB Related Pins* 733
Table 28.2 DC Characteristics (2-c) USB Transceiver Related Pins* 734
Table 28.3 Permissible Output Current Values 734
Table 28.4 Operating Frequencies 735
Table 28.5 Clock Timing 736
Table 28.6 Control Signal Timing 740
Table 28.7 Bus Timing 743
Table 28.8 Peripheral Module Signal Timing 776
Table 28.9 SIOF Module Signal Timing 777

Table 28.10 SCIF Module Signal Timing 781
Table 28.11 USB Module Clock Timing 782
Table 28.12 USB Module Clock Timing 782
Table 28.13 USB Transceiver Timing (Full-Speed) 783
Table 28.14 USB Transceiver Timing (Low-Speed) 783
Table 28.15 Bluetooth Interface Module Clock Timing 785
Table 28.16 Bluetooth Interface Module Clock Timing 785
Table 28.17 Bluetooth Interface Module Low Power Clock Timing 785
Table 28.18 Bluetooth Interface Module Low Power Clock Timing 786
Table 28.19 Bluetooth Interface (BT) Voice CODEC Interface Signal Timing 786
Table 28.20 SPI Interface Signal Timing for Bluetooth Interface (BT) RF 787
Table 28.21 Receive Data Timing 788
Table 28.22 D/A Converter Characteristics 790

Appendix

Table A.1 791

Section 1 Overview

1.1 SH7660 Features

This LSI is a RISC (Reduced Instruction Set Computer) microcomputer that integrates a 32-bit RISC-type SuperH architecture CPU and digital signal processing (DSP) extended function as its core, together with 16-kbyte cache memory, 16-kbyte X/Y memory, and 128-kbyte U memory, as well as peripheral functions required for system configuration such as an interrupt controller.

High-speed data transfers can be performed by an on-chip direct memory access controller (DMAC) and a direct connection to various memories can be performed by the external memory access support function. Moreover, this LSI also includes powerful peripheral functions that are essential to system configuration, such as the USB (host/function), high-speed (921 kbits/s) asynchronous serial interface circuit, voice/audio CODEC serial interface circuit, and D/A converter.

This LSI also supports the Bluetooth™ interface as one of peripheral functions and provides the Bluetooth protocol stack function including firmware up to the HCI layer as a standard function. Moreover, this LSI has extra processing power to perform the upper protocol stack and various application profiles to implement the whole Bluetooth baseband functions. The API is also provided to handle the protocol stack function below the HCI layer so that this Bluetooth interface can easily be handled like the conventional peripheral functions. This Bluetooth interface can be connected directly to the RF chip such as the HD157100NP or HD157102NP (manufactured by Renesas Technology).

Note: Bluetooth is a registered trademark of Bluetooth SIG, Inc., USA. Renesas Technology uses Bluetooth by entering into licensing agreements.

The features of this LSI are listed in table 1.1.

Table 1.1 SH7660 Features

Item	Features
CPU	• Renesas Technology Original SuperH architecture • Compatible with SH-1, SH-2 and SH-3 at object code level • 32-bit internal data bus • Supports various registers Sixteen 32-bit general registers (including eight 32-bit bank registers) Five 32-bit control registers Four 32-bit system registers • Supports RISC-type instruction set Instruction length: 16-bit fixed length for improved code efficiency Load/store architecture Delayed branch instructions Instruction set suitable for C language • Supports barrel shift instructions and multiply-and-accumulate instructions • Instruction execution time: One instruction/cycle for basic instructions • Logical address space: 4 Gbytes • Five-stage pipeline
DSP unit	• Mixture of 16-bit and 32-bit instructions • 32-/40-bit internal data bus • Employs multiplier, ALU, and barrel shifter • 16-bit × 16-bit → 32-bit one cycle multiplier • Employs large-capacity DSP data register files Six 32-bit data registers Two 40-bit data registers • Extended harvard architecture for DSP data bus Two data buses One instruction bus • Maximum four parallel operations: ALU, multiply, and two load or store • Two addressing units to generate addresses for two memory access • DSP data addressing modes: Increment and indexing (with or without modulo addressing) • Zero-overhead repeat loop control • Conditional execution instructions • User DSP mode and privileged DSP mode

Item	Features
Cache memory	• 16-kbyte cache, unified of instructions and data • 256 entries, 4-way set associative, and 16-byte block length • Write-back, write-through, and LRU replacement algorithm • 1-stage write-back buffer • Maximum 2 ways can be protected
X/Y memory	• Three independent read/write ports 8-/16-/32-bit access from the CPU Maximum two 16-bit accesses from the DSP 8-/16-/32-bit access from the DMAC 8-/32-bit access from the USBH • A total of 16 kbytes (4 kbytes × 4) • No interrupt requests • No DMA transfer requests (accessible as a transfer source or destination)
U memory	• Two independent read/write ports 8-/16-/32-bit access from the CPU 16-/32-bit access from the DSP 8-/16-/32-bit access from the DMAC 8-/32-bit access from the USBH • As large as 128-kbyte memory • No interrupt requests • No DMA transfer requests (accessible as a transfer source or destination)
Interrupt controller (INTC)	• Five external interrupt pins (NMI, IRQ4 to IRQ2, IRQ0) • One interrupt request output pin ($\overline{\text{IRQOUT}}$) • On-chip peripheral interrupt: Priority is independently selected for each module • Selection of falling/rising edge sense and high/low level sense

Item	Features
Bus state controller (BSC)	• Physical address space is divided into three areas: Area 0, area 3, area 4; each a maximum of 16 Mbytes • The following features are settable for each area: Bus size (8 or 16 bits); The supported bus size may differ for each area Number of access wait cycles (Numbers of wait-state cycles during reading and writing are independently selectable for some areas.) Setting of idle wait cycles (for the same area or different area) Specifying the memory to be connected to each area allows direct connection to SRAM, SRAM with byte selection, SDRAM, or burst ROM Outputs chip select signals ($\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$) for corresponding area (Can be turned by the program $\overline{CSn}$ assert/negate timing.) • SDRAM refresh function Auto-refresh and self-refresh modes • SDRAM burst access function • Big endian or little endian can be set.
Direct memory access controller (DMAC)	• Four channels (for one channel, external requests can be accepted) • Burst mode and cycle-steal mode • Intermittent cycle-steal mode
Clock pulse generator (CPG)	• Clocks can be input from an external pin (EXTAL or CKIO) or crystal unit • Three clocks generated: Internal clock: 120 MHz Bus clock: 60 MHz Peripheral clock: 30 MHz • Supports power-down modes: Software standby mode Sleep mode Module standby mode • Three clock modes (PLL1 and PLL2 multiplication ratio, clock, and crystal unit can be selected) • One frequency-divider mode
Watchdog timer (WDT)	• One-channel watchdog timer • Watchdog timer mode and interval timer mode can be selected • In interval timer mode, interrupts can be generated

Item	Features
Timer unit (TMU)	• Three channels of 32-bit timer • Four input clocks can be selected for each channel counter • 32-bit down counter of the auto-reload type • On-chip prescaling by Pϕ • Interrupt requests: Interrupt requests are generated by underflow of 32-bit down counter
Serial I/O with FIFO (SIOF)	• One channel • 16-stage 32-bit FIFOs (independent for reception and transmission) • 8-/16-/16-bit stereo-audio input/output supported • Synchronization method: Frame synchronized pulse and switching between left and right channels • Supports a CODEC control data interface • Can be connected to linear, audio, A-Law, or μ-Law CODEC chip • Supports master and slave modes • Sampling rate clock can be generated from Pϕ or be input from external pin (max. 48 kHz) • On-chip prescaler from Pϕ • Module standby function • Capable of interrupt requests and DMAC requests
Serial communication interface with FIFO (SCIF0, SCIF1)	• Two channels • Supports asynchronous mode and clocked synchronous mode • 64-byte transmit/receive FIFOs • High-speed UART • Supports $\overline{\text{CTS}}$/$\overline{\text{RTS}}$ • On-chip prescaler from Pϕ • Capable of interrupt requests and DMAC requests
Boot function (BOOT)	• Normal mode or boot mode at a power-on reset • Automatic fetching of initial write program from SCIF0 Downloading of program to on-chip memory (U memory) • Auto-running function of downloaded program

Item	Features
USB host controller (USBH)	• Register set conforming to OHCI version 1.0 • Conforms to USB 1.1 • Supports 127 endpoints • Supports control/bulk/interrupt/isochronous modes • Bus master controller • One-port USB transceiver (sharing with USB function controller) • Selectable module input clock: 48-MHz external input or on-chip DLL • Capable of issuing interrupt requests
USB function controller (USBF)	• Conforms to USB 1.1 • One-port USB transceiver (sharing with USB host controller) • Supports six endpoints, the number of endpoints is selectable • Supports isochronous, totally 2 endpoints • Supports control (endpoint 0)/bulk (2 endpoints)/interrupt (1 endpoint) • The USB standard commands are supported, and class and vendor commands are handled by firmware • On-chip FIFO buffer for endpoints (bulk, isochronous: 128 bytes/endpoint) • Selectable module input clock: 48-MHz external input or on-chip DLL • Capable of issuing interrupt requests and DMAC requests
Bluetooth interface (BT)	• Supports Bluetooth version 1.1 • Supports direct interface with the HD157100NP or HD157102NP (Renesas) RF IC • Supports direct interface with the STLC7550 or MC145483 voice CODEC IC • Supports four voice CODEC formats (A-law/μ-law/CVSD/linear PCM) • Supports ACL and SCO links (SCI over HCI possible) • Frequency hopping within 79 channels as a form of spread-spectrum modulation • Use of the 32.768-kHz clock for RTC (Realtime Clock) as the clock for power-down mode • Selectable clock for power-down mode: Crystal oscillator mode or external direct input mode • Supports three power-down modes: Hold/sniff/park
D/A converter (DAC)	• Two channels of 8 bit D/A converters • Output range: 0 to AVcc
I/O port/pin function controller (PFC)	• Bitwise selection of input/output

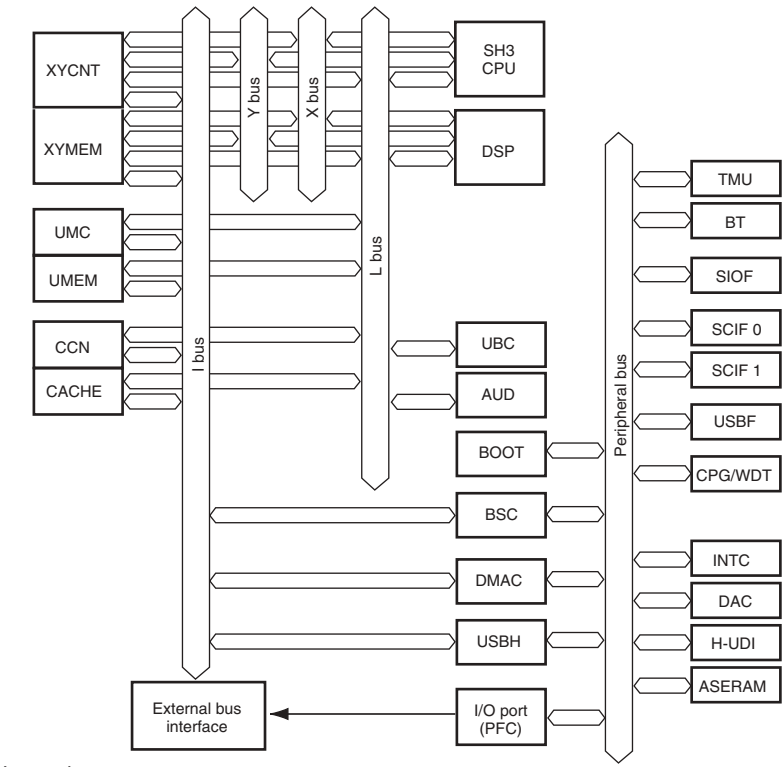
Item	Features
User break controller (UBC)	• Address, data value, access type, and data size are available for setting as break conditions • Supports the sequential break function • Two break channels
User debugging interface (H-UDI)	• Supports the E10A emulator • Realtime branch trace • 1-kbyte of on-chip RAM for executing the high-speed emulation program
Package	• 208-pin BGA (pin pitch: 0.65 mm)
Power-supply voltage	• I/O: 3.0 to 3.6 V (some parts: 2.7 to 3.0 V) • Internal: on-chip regulator provides, or 1.4 to 1.6 V external power supply can be selected.
Operating temperature range	• –40°C to +85°C (except USB) • –20°C to +85°C (for USB only)

Notes: 1. This LSI does not support the MMU and TLB functions, therefore, the LDTLB instruction provided by the SH-3 CPU is not available.

2. Since the on-chip AUD and ASERAM modules are used for debugging, related descriptions are omitted in this manual. For details, refer to the user's manual related to development tools.

1.2 Block Diagram

Figure 1.1 shows an internal block diagram of the SH7660.



Legend:

ASERAM :	ASE memory	H-UDI :	User debugging interface
AUD :	Advanced user debugger	INTC :	Interrupt controller
BOOT :	Boot loader	PFC :	Pin function controller
BSC :	Bus state controller	SCIF :	Serial communication interface with FIFO
BT :	Bluetooth interface	SIOF :	Serial I/O with FIFO
CACHE :	Cache memory	TMU :	16-bit Timer unit
CCN :	Cache memory controller	UBC :	User break controller
CPG/WDT :	Clock pulse generator/watchdog timer	UMC :	U memory controller
CPU :	Central processing unit	UMEM :	U memory
DAC :	D/A converter	USBH/USBF :	Universal serial bus host/universal serial bus function
DMAC :	Direct memory access controller	XYCNT:	X/Y memory controller
DSP:	Digital signal processor	XYMEM:	X/Y memory

Note: Since the on-chip AUD and ASERAM modules are used for debugging, related descriptions are omitted in this manual. For details, refer to the user's manual related to development tools.

Figure 1.1 Block Diagram of SH7660

1.3 Pin Assignment

Figure 1.2 shows the pin assignment. Table 1.2 shows pin functions and initial values.

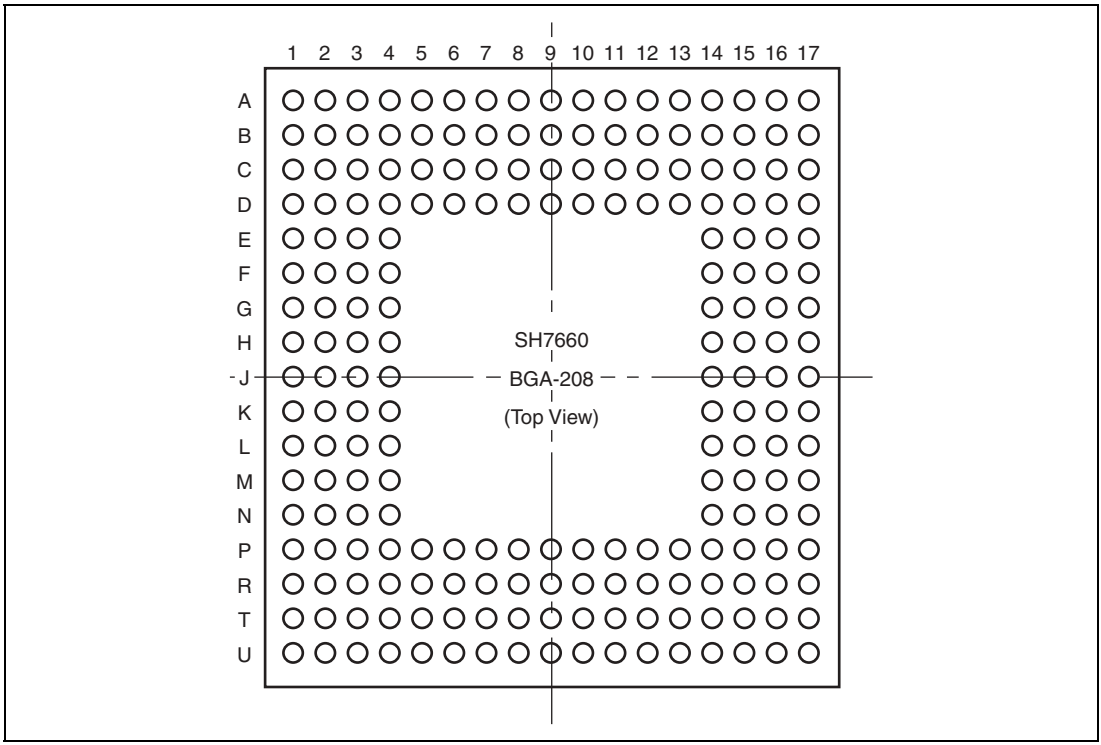


Figure 1.2 Pin Assignment

Table 1.2 Pin Functions and Initial Values

Pin No.	Pin Name	Description
A1	PTH4/SCIF1_$\overline{CTS}$	Port H/$\overline{CTS}$ input for SCIF1 Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
A2	NMI	Nonmaskable interrupt request When this pin is not used, it should be fixed to high.
A3	PTB1/IRQ2	Port B/external interrupt request input Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
A4	RTCSEL0	Test pin This pin must be fixed to low.
A5	NC	Reserved This pin should be open.
A6	NC	Reserved This pin should be open.
A7	AVss (DAC)	Analog power supply for D/A converter (DAC) (0 V)
A8	XTAL	Crystal unit pin
A9	EXTAL	External clock input/crystal unit pin
A10	EXTAL2	External clock input/crystal unit pin for Bluetooth power-down state When RTCSEL0 = high, this pin enters the non-active state.
A11	RDI_TXTRDATA*[1]	Transmit/receive data I/O for RF IC The initial state is output.
A12	Vss_28	Power supply for I/O pins with RF IC (0 V)
A13	RDI_REFCLK_IN*[1]	Clock input for data I/O with RF IC This pin is always input.
A14	RCI_SPI_TXRX*[1]	SPI serial data I/O with RF IC The initial state is output.
A15	USB_N	D– I/O for on-chip USB transceiver The initial state is Hi-Z.
A16	USB_P	D+ I/O for on-chip USB transceiver The initial state is Hi-Z.
A17	AVss (USB)	Analog power supply for USB transceiver (0 V)

Pin No.	Pin Name	Description
B1	NC	Reserved This pin should be open.
B2	$\overline{\text{ASEMD0}}$	ASE mode control input This pin should be fixed to high in the normal state.
B3	PTG0/SCIF0_SCK/IRQ3	Port G/clock I/O for SCIF0/external interrupt request input Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
B4	RREF	Test pin This pin should be pulled-up.
B5	NC	Reserved This pin should be open.
B6	Vss (DLL)	Power supply for internal regulator (DLL) (0 V)
B7	DA0	D/A converter (DAC) output (channel 0)
B8	Vcc (I/O)	Power supply for I/O (3.3 V)
B9	Vss	Power supply (0 V)
B10	XTAL2	Crystal unit pin for Bluetooth power-down state When RTCSEL0 = high, this pin enters the non-active state.
B11	RDI_RXBDW_OUT*[1]	Packet control output for RF IC
B12	Vcc_28	Power supply for I/O pin with RF IC (2.8 V/3.3 V)
B13	RCI_SPI_CLK*[1]	SPI interface clock output with RF IC
B14	NC	Reserved This pin should be open.
B15	NC	Reserved This pin should be open.
B16	NC	Reserved This pin should be open.
B17	NC	Reserved This pin should be open.
C1	PTH1/SCIF1_TxD	Port H/transmit data output for SCIF1 This pin is fixed to input and enters the output Hi-Z state in the initial state.

Pin No.	Pin Name	Description
C2	PTH2/SCIF1_RxD	Port H/receive data input for SCIF1 Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
C3	PTH3/SCIF1_$\overline{RTS}$	Port H/$\overline{RTS}$ output for SCIF1 Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
C4	PTB0/IRQ0	Port B/external interrupt request input Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
C5	Vss (SREG)	Power supply for internal regulator (sub) (0 V)
C6	V_{DD} (DLL)	Power-supply output from internal regulator / power-supply input (DLL) (1.5 V)
C7	DA1	D/A converter (DAC) output (channel 1)
C8	Vss	Power supply (0 V)
C9	NC	Reserved This pin should be open.
C10	Vss	Power supply (0 V)
C11	RDI_CTRL3*[1]	Strobe output for enabling RF-IC oscillator
C12	Vss_28	Power supply for I/O pin with RF IC (0 V)
C13	RCI_SPI_ENB*[1]	SPI interface enable output with RF IC
C14	AVcc (USB)	Analog power supply for USB transceiver (3.3 V)
C15	NC	Reserved This pin should be open.
C16	UCLK	External clock input for USB When initialized, the internal DLL clock is valid and this pin is internally fixed to input.
C17	$\overline{USB_OVR_CRNT}$/ USB_VBUS	Over current detection input for USB/USB-cable connection monitor input Even if this pin is not used, this pin should be driven by an appropriate level.
D1	PTG2/SCIF0_RxD	Port G/receive data input for SCIF0 Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.

Pin No.	Pin Name	Description
D2	PTG3/SCIF0_ $\overline{\text{RTS}}$	Port G/$\overline{\text{RTS}}$ output for SCIF0 Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
D3	PTG4/SCIF0_ $\overline{\text{CTS}}$	Port G/$\overline{\text{CTS}}$ input for SCIF0 Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
D4	PTH0/SCIF1_SCK/IRQ4	Port H/clock I/O for SCIF1/external interrupt request input Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
D5	Vcc (SREG)	Power supply for internal regulator (sub) (3.3 V)
D6	Vcc (DLL)	Power supply for internal regulator (DLL) (3.3 V)
D7	Vss	Power supply (0 V)
D8	AVcc (DAC)	Analog power supply for D/A converter (DAC) (3.3 V)
D9	Vcc (I/O)	Power supply for I/O (3.3 V)
D10	NC	Reserved This pin should be open.
D11	RDI_CTRL4*[1]	Reset control for RF IC and power-down control output
D12	Vcc_28	Power supply for I/O pin with RF IC (2.8 V/3.3 V)
D13	$\overline{\text{RESETP}}$	Power-on reset input
D14	$\overline{\text{USB_PWR_EN}}$/ USB_PULLUP	USB power-supply application enable control output/pull-up control output for USB The initial state is low-level output.
D15	Vcc (PLL)	Power supply for internal regulator (PLL1/PLL2) (3.3 V)
D16	V_{DD} (PLL)	Power supply output from internal regulator/power-supply input (PLL1/PLL2) (1.5 V)
D17	Vss (PLL)	Power supply for internal regulator (PLL1/PLL2) (0 V)
E1	PTA6/SIOF_$\overline{\text{SS2}}$	Port A/SPI mode slave device select for SIOF Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
E2	PTG1/SCIF0_TxD	Port G/transmit data output for SCIF0 This pin is fixed to input and enters the output Hi-Z state in the initial state.
E3	Vss	Power supply (0 V)

Pin No.	Pin Name	Description
E4	Vcc (I/O)	Power supply for I/O (3.3 V)
E14	Vcc (internal)	Power supply for internal regulator (3.3 V)
E15	V_{DD}	Power-supply output from internal regulator/power-supply input (1.5 V)
E16	Vss	Power supply (0 V)
E17	CKIO	System clock I/O
F1	PTA5/SIOF_$\overline{SS1}$	Port A/SPI mode slave device select for SIOF Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
F2	Vss	Power supply (0 V)
F3	V_{DD}	Power-supply output from internal regulator/power-supply input (1.5 V)
F4	Vcc (internal)	Power supply for internal regulator (3.3 V)
F14	Vcc (I/O)	Power supply for I/O (3.3 V)
F15	Vss	Power supply (0 V)
F16	NC	Reserved This pin should be open.
F17	MD5	Endian set input
G1	PTA2/SIOF_RXD (MISO)	Port A/receive data input for SIOF Though this pin is fixed to input and enters the output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
G2	PTA3/SIOF_SYNC (SIOF_$\overline{SS0}$)	Port A/frame synchronization signal output for SIOF (SPI mode slave device select for SIOF) Though this pin is fixed to input and enters output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
G3	PTA4/SIOF_SCK (SCK)	Port A/serial clock I/O for SIOF Though this pin is fixed to input and enters output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
G4	NC	Reserved This pin should be open.
G14	MD2	Clock mode set input
G15	MD1	Clock mode set input

Pin No.	Pin Name	Description
G16	PTE6/RDI_CTRL2*[1] /IRQOUT	Port E/external power amplifier control output for RF IC /IRQOUT Though this pin is fixed to input and enters output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
G17	PTE0 /VCI_CODEC_PWRDWN*[2] /TEND	Port E/power-down control output for voice CODEC IC /DMA transfer end flag output Though this pin is fixed to input and enters output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
H1	TEST_REG	Built-in regulator selection pin. The built-in power supply regulator is used when this pin is high, external power input is used when this pin is low for the internal power (VDD=1.5V).
H2	NC	Reserved This pin should be open.
H3	PTA0/SIOF_MCLK	Port A/master clock input for SIOF Though this pin is fixed to input and enters output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
H4	PTA1/SIOF_TXD (MOSI)	Port A/transmit data output for SIOF Though this pin is fixed to input and enters output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
H14	Vcc (internal)	Power supply for internal regulator (3.3 V)
H15	V_{DD}	Power-supply output from internal regulator/power-supply input (1.5 V)
H16	Vss	Power supply (0 V)
H17	PTE1/VCI_SCO_TX*[2]/DACK	Port E/transmit data input from voice CODEC IC /DMA transfer request accept Though this pin is fixed to input and enters output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
J1	TRST	H-UDI reset input This pin is internally pulled up. When a boundary scan function is not used in normal mode (ASEMD0 = high), this pin should be fixed to low.
J2	ASEBRKAK	ASE break acknowledge output
J3	TDO	H-UDI data output

Pin No.	Pin Name	Description
J4	$\overline{\text{TEST}}$	Test input This pin should always be fixed to high.
J14	Vcc (I/O)	Power supply for I/O (3.3 V)
J15	PTE2/VCI_SCO_RX*[2] /DREQ	Port E/receive data output to voice CODEC IC /DMA transfer request input Though this pin is fixed to input and enters output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
J16	PTE4 /VCI_SCO_SYNC_OUT*[2] /$\overline{\text{BREQ}}$	Port E/frame synchronization signal output for voice CODEC IC/bus mastership request input The bus mastership request input is valid in the initial state.
J17	PTE3 /VCI_SCO_CLK_OUT*[2] /$\overline{\text{BACK}}$	Port E/clock output for voice CODEC IC/bus release acknowledge output The bus release acknowledge output is valid in the initial state.
K1	TDI	H-UDI test data input This pin is internally pulled up.
K2	TCK	H-UDI clock input This pin is internally pulled up.
K3	Vss	Power supply (0 V)
K4	Vcc (I/O)	Power supply for I/O (3.3 V)
K14	AUDATA3	AUD data bus (bit 3) output
K15	$\overline{\text{AUDSYNC}}$	AUD synchronization signal output
K16	PTE5/VCI_HWC*[2]	Port E/operating mode select output for voice CODEC IC Though this pin is fixed to input and enters output Hi-Z state in the initial state, the internal pull-up MOS is turned on.
K17	Vss	Power supply (0 V)
L1	TMS	H-UDI test mode switch This pin is internally pulled up.
L2	Vss	Power supply (0 V)
L3	V_{DD}	Power-supply output from internal regulator/Power-supply input (1.5 V)
L4	Vcc (internal)	Power supply for internal regulator (3.3 V)
L14	Vcc (I/O)	Power supply for I/O (3.3 V)

Pin No.	Pin Name	Description
L15	Vss	Power supply (0 V)
L16	AUDATA1	AUD data bus (bit 1) output
L17	AUDATA2	AUD data bus (bit 2) output
M1	NC	Reserved This pin should be open.
M2	$\overline{\text{BOOT_E}}$	Boot mode enable control input This pin is internally pulled up.
M3	$\overline{\text{CS0}}$	Chip select 0 signal output
M4	$\overline{\text{CS3}}$	Chip select 3 signal output
M14	PTB7/$\overline{\text{BS}}$	Port B/bus cycle start signal output The bus cycle start signal output is valid in the initial state.
M15	$\overline{\text{RD}}$	Read strobe signal output
M16	AUDCK	AUD clock output
M17	AUDATA0	AUD data bus (bit 0) output
N1	$\overline{\text{CS4}}$	Chip select 4 signal output
N2	D0	Data bus (bit 0)
N3	D1	Data bus (bit 1)
N4	D2	Data bus (bit 2)
N14	Vcc (internal)	Power supply for internal regulator (3.3 V)
N15	V_{DD}	Power-supply output from internal regulator/Power-supply input (1.5 V)
N16	Vss	Power supply (0 V)
N17	PTB6/$\overline{\text{REFOUT}}$	Port B/bus release request output The bus release request output is valid in the initial state.
P1	D3	Data bus (bit 3)
P2	D4	Data bus (bit 4)
P3	Vcc (I/O)	Power supply for I/O (3.3 V)
P4	D5	Data bus (bit 5)
P5	D13	Data bus (bit 13)
P6	Vcc (I/O)	Power supply for I/O (3.3 V)
P7	A3	Address bus (bit 3)
P8	Vcc (I/O)	Power supply for I/O (3.3 V)
P9	A8	Address bus (bit 8)

Pin No.	Pin Name	Description
P10	Vcc (I/O)	Power supply for I/O (3.3 V)
P11	A15	Address bus (bit 15)
P12	Vcc (I/O)	Power supply for I/O (3.3 V)
P13	PTD3/A21	Port D/address bus (bit 21) The A21 output is valid in the initial state.
P14	PTB4/$\overline{\text{RAS}}$	Port B/$\overline{\text{RAS}}$ (SDRAM) output The $\overline{\text{RAS}}$ output is valid in the initial state.
P15	Vcc (I/O)	Power supply for I/O (3.3 V)
P16	Vss	Power supply (0 V)
P17	$\overline{\text{WAIT}}$	Hardware wait request This pin is internally pulled up.
R1	Vss	Power supply (0 V)
R2	REG_PD_MAIN	This pin should be fixed to high when the external power supply is used for the internal power (VDD=1.5V). This pin should be open when the internal power supply regulator is used.
R3	D10	Data bus (bit 10)
R4	Vcc (I/O)	Power supply for I/O (3.3 V)
R5	D14	Data bus (bit 14)
R6	Vss	Power supply (0 V)
R7	A4	Address bus (bit 4)
R8	Vss	Power supply (0 V)
R9	A11	Address bus (bit 11)
R10	REG_PD_BGR	This pin should be fixed to high when the external power supply is used for the internal power (VDD=1.5V). This pin should be open when the internal power supply regulator is used.
R11	A14	Address bus (bit 14)
R12	A18	Address bus (bit 18)
R13	PTD2/A20	Port D/address bus (bit 20) The A20 output is valid in the initial state.
R14	Vcc (I/O)	Power supply for I/O (3.3 V)
R15	PTB2/CKE	Port B/clock enable (SDRAM) output The CKE output is valid in the initial state.
R16	$\overline{\text{WE0}}$/DQM0	D7 to D0 select signal/DQM (SDRAM) output
R17	$\overline{\text{WE1}}$/DQM1	D15 to D8 select signal/DQM (SDRAM) output

Pin No.	Pin Name	Description
T1	D6	Data bus (bit 6)
T2	D7	Data bus (bit 7)
T3	NC	Reserved This pin should be open.
T4	D11	Data bus (bit 11)
T5	D15	Data bus (bit 15)
T6	A1	Address bus (bit 1)
T7	A5	Address bus (bit 5)
T8	REG_PD_VREF	This pin should be fixed to high when the external power supply is used for the internal power (VDD=1.5V). This pin should be open when the internal power supply regulator is used.
T9	A9	Address bus (bit 9)
T10	NC	Reserved This pin should be open.
T11	A13	Address bus (bit 13)
T12	A17	Address bus (bit 17)
T13	PTD1/A19	Port D/address bus (bit 19) The A19 output is valid in the initial state.
T14	PTD5/A23	Port D/address bus (bit 23) The A23 output is valid in the initial state.
T15	PTB5/RD/$\overline{WR}$	Port B/read/write output The read/write output is valid in the initial state.
T16	VBB	Test pin This pin should be connected to Vss.
T17	NC	Reserved This pin should be open.
U1	D8	Data bus (bit 8)
U2	D9	Data bus (bit 9)
U3	Vss	Power supply (0 V)
U4	D12	Data bus (bit 12)
U5	PTD0/A0	Port D/address bus (bit 0) The A0 output is valid in the initial state.
U6	A2	Address bus (bit 2)
U7	A6	Address bus (bit 6)

Pin No.	Pin Name	Description
U8	A7	Address bus (bit 7)
U9	A10	Address bus (bit 10)
U10	A12	Address bus (bit 12)
U11	Vss	Power supply (0 V)
U12	A16	Address bus (bit 16)
U13	Vss	Power supply (0 V)
U14	PTD4/A22	Port D/address bus (bit 22) The A22 output is valid in the initial state.
U15	Vss	Power supply (0 V)
U16	PTB3/$\overline{\text{CAS}}$	Port B/$\overline{\text{CAS}}$ (SDRAM) output The $\overline{\text{CAS}}$ output is valid in the initial state.
U17	VBBENB	Test pin This pin should be connected to Vss.

Notes: 1. These pins are connected to the RF IC. For details on the connection, refer to section 21, Bluetooth Interface (BT).

2. These pins are connected to the voice CODEC IC. For details on the connection, refer to section 21, Bluetooth Interface (BT).

1.4 Pin Functions

Table 1.3 lists the pin functions.

Table 1.3 Pin Functions

Classification	Symbol	I/O	Name	Function
Power supply	V_{DD}*[1]	Output/ Input	Power supply	Power output of the built-in regulator, or power input from an external power supply (for internal circuit). A pin function is specified with the TEST_REG pin.
	Vcc (internal)	Input	Power supply	Power supply for internal regulator. Connect all Vcc pins to the system power supply. There will be no operation if any pins are open.
	Vcc (I/O)	Input	Power supply	Power supply for I/O pins. Connect all Vcc pins to the system power supply. There will be no operation if any pins are open.
	Vss	Input	Ground	Ground pin. Connect all Vss pins to the system power supply (0 V). There will be no operation if any pins are open.
	TEST_REG	Input	Power supply selection	Built-in regulator selection pin. The built-in power supply regulator is used when this pin is high, external power input is used when this pin is low for the internal power (V_{DD}=1.5V). The level of this pin should not be changed during operation
	REG_PD_MAIN	Input/NC	Power supply control	Built-in regulator control. This pin should be fixed to high when the external power supply is used for the internal power (V_{DD}=1.5V). This pin should be open when the built-in power supply regulator is used.
	REG_PD_ BGR	Input/NC	Power supply control	Built-in regulator control. This pin should be fixed to high when the external power supply is used for the internal power (V_{DD}=1.5V). This pin should be open when the built-in power supply regulator is used.

Classification	Symbol	I/O	Name	Function
Power supply	REG_PD_VREF	Input/NC	Power supply control	Built-in regulator control. This pin should be fixed to high when the external power supply is used for the internal power (V_{DD}=1.5V). This pin should be open when the built-in power supply regulator is used.
Clock	Vcc (PLL)	Input	PLL1/PLL2 power supply	Power supply for on-chip PLL1/PLL2 oscillator
	V_{DD} (PLL)*[1]	Output/ Input	Power supply	Power output of the built-in regulator for PLL1/PLL2 oscillator, or power input from an external power supply. A pin function is specified with the TEST_REG pin.
	Vss (PLL)	Input	PLL1/PLL2 ground	Ground pin for on-chip PLL1/PLL2 oscillator
	EXTAL	Input	Crystal input (clock input)	For connection to the crystal resonator. Also, this pin can input external clocks. For connection to the crystal resonator and connection of the input of external clocks, refer to section 11, Clock Pulse Generator (CPG).
	XTAL	Output	Crystal output	For connection to the crystal resonator. For connection to the crystal resonator and connection of the input of external clocks, refer to section 11, Clock Pulse Generator (CPG).
	CKIO	I/O	Clock I/O	Supplies system clocks to external devices or used for external clock input.
Operating mode control	MD5, MD2, MD1	Input	Mode control	Sets the operating mode. The level on these pins should not be changed during operation. MD2 and MD1 set the clock mode and MD5 sets the endian.

Classification	Symbol	I/O	Name	Function
System control	$\overline{\text{RESETP}}$	Input	Power-on reset	When this pin goes low, the system enters the power-on reset state.
	$\overline{\text{BOOT_E}}$	Input	Boot control	Boot mode enable control input. When this pin goes low at a power-on reset, the system enters the boot mode and runs the boot loader after reset processing. When this pin goes high at a power-on reset, the normal reset sequence is executed.
	$\overline{\text{BREQ}}$	Input	Bus mastership request	This pin is set to low when an external device requests the bus mastership release.
	$\overline{\text{BACK}}$	Output	Bus mastership request acknowledge	Indicates that bus mastership is externally released. A device which has output the BREQ signal knows that the bus mastership is acquired by receiving the $\overline{\text{BACK}}$ signal.
Interrupts	NMI	Input	Nonmaskable interrupt request	Nonmaskable interrupt request pin. This pin should be fixed to high when not in use.
	IRQ4 to IRQ2, IRQ0	Input	Interrupt requests 4 to 2, 0	Maskable interrupt request pins. Selectable as level input or edge input. The rising edge, falling edge, and both edges are selectable as edges.
	$\overline{\text{IRQOUT}}$	Output	Bus mastership request	Notifies an external device of interrupt source occurrence.
Address bus	A23 to A0	Output	Address bus	Outputs addresses.
Data bus	D15 to D0	I/O	Data bus	16-bit bidirectional bus
Bus control	$\overline{\text{CS0}}$, $\overline{\text{CS3}}$, $\overline{\text{CS4}}$	Output	Chip select 0, 3, 4	Chip select signal for external memory or devices
	$\overline{\text{RD}}$	Output	Read	Indicates reading of data from external devices.
	RD/$\overline{\text{WR}}$	Output	Read/write	Read/write signal pin
	$\overline{\text{BS}}$	Output	Bus start	Bus cycle start signal pin
	$\overline{\text{WE1}}$	Output	Write upper bits	Indicates that bits 15 to 8 of the data in the external memory or device are being written.

Classification	Symbol	I/O	Name	Function
Bus control	$\overline{WE0}$	Output	Write lower bits	Indicates that bits 7 to 0 of the data in the external memory or device are being written.
	$\overline{CAS}$	Output	Data enable	$\overline{CAS}$ signal of SDRAM
	$\overline{RAS}$	Output	Data enable	$\overline{RAS}$ signal of SDRAM
	CKE	Output	Clock enable	Clock enable signal pin of SDRAM
	DQM1	Output	DQ mask	Indicates that bits 15 to 8 of the data in SDRAM are selected.
	DQM0	Output	DQ mask	Indicates that bits 7 to 0 of the data in SDRAM are selected.
	$\overline{REFOUT}$	Output	Bus release request	Indicates that a refresh request has occurred during bus mastership release.
	$\overline{WAIT}$	Input	Wait	Inserts a wait cycle into the bus cycles during access to the external space.
Direct memory access controller (DMAC)	DREQ	Input	DMA transfer request	Input pin for an external DMA transfer request
	DACK	Output	DMA transfer request acceptance	Output pin for external DMA transfer request acceptance
	TEND	Output	DMA transfer end	Indicates that DMA transfer ends.
Serial I/O with FIFO (SIOF)	SIOF_TXD (MOSI)	Output	Transmit data	Transmit data pin
	SIOF_RXD (MISO)	Input	Receive data	Receive data pin
	SIOF_SCK (SCK)	I/O	Serial clock	Clock I/O pin
	SIOF_MCLK	Input	Master clock	Master clock input pin
	SIOF_SYNC (SIOF_$\overline{SS0}$)	I/O	Frame synchronization signal (slave device 0 select)	Frame synchronization signal pin (In SPI mode, selects the slave device 0.)
	SIOF_$\overline{SS1}$	Output	Slave device 1 select	In SPI mode, selects the slave device 1.
	SIOF_$\overline{SS2}$	Output	Slave device 2 select	In SPI mode, selects the slave device 2.

Classification	Symbol	I/O	Name	Function
Serial communication interface with FIFO (SCIF0, SCIF1)	SCIF0_TxD, SCIF1_TxD	Output	Transmit data	Transmit data pin
	SCIF0_RxD, SCIF1_RxD	Input	Receive data	Receive data pin
	SCIF0_SCK, SCIF1_SCK	I/O	Serial clock	Clock I/O pin
	SCIF0_$\overline{\text{RTS}}$, SCIF1_$\overline{\text{RTS}}$	Output	Transmit request	Modem control pin
	SCIF0_$\overline{\text{CTS}}$, SCIF1_$\overline{\text{CTS}}$	Input	Transmit enable	Modem control pin
USB	UCLK	Input	USB clock	USB clock input pin (48-MHz input)
	$\overline{\text{USB_PWR_EN}}$/ USB_PULLUP	Output	USB power supply control	Host: USB power supply application enable control Function: USB pull-up control
	$\overline{\text{USB_OVR_CRNT}}$/USB_VBUS	Input	USB power supply detection	Host: Over current detection Function: USB-cable connection monitor pin
	USB_N	I/O	D– I/O	D– I/O for on-chip USB transceiver
	USB_P	I/O	D+ I/O	D+ I/O for on-chip USB transceiver
	AVcc (USB)	Input	Analog power supply for USB transceiver	Analog power supply pin for USB transceiver
	AVss (USB)	Input	Analog ground for USB transceiver	Analog ground pin for USB transceiver Connect to the system ground (Vss).
	Vcc (DLL)	Input	3.3-V power supply for USB clock multiplication circuit	Power supply pin for the 48-MHz clock multiplication circuit of the USB Connect to the system power supply (Vcc).
	V_{DD} (DLL)*[1]	Output/ Input	Power supply	Power output of the built-in regulator for 48MHz clock frequency multiplier circuit of USB, or power input from an external power supply. A pin function is specified with the TEST_REG pin.
	Vss (DLL)	Input	Ground for USB clock multiplication circuit	Ground pin for the 48-MHz clock multiplication circuit of the USB Connect to the system ground (Vss).

Classification	Symbol	I/O	Name	Function
Bluetooth interface (BT)	RDI_ TXTRDATA*[2]	I/O	Data bus for transmission/ reception	Bus for data transmission/reception with the RF IC
	RDI_RXBDW_ OUT*[2]	Output	Packet control	Signal for notifying the RF IC of the packet processing state of the BT
	RDI_REFCLK_ IN*[2]	Input	BT clock input	Input the BT operating clock.
	RDI_CTRL2*[2]	Output	RF IC control 2	Used for external power amplifier control when supporting class 1.
	RDI_CTRL3*[2]	Output	RF IC control 3	Strobe signal for enabling the RF IC oscillator
	RDI_CTRL4*[2]	Output	RF IC control 4	Reset control and power-down control output for RF IC
	RCI_SPI_CLK*[2]	Output	SPI clock	Serial interface clock
	RCI_SPI_ TXRX*[2]	I/O	SPI data	Serial interface data
	RCI_SPI_ENB*[2]	Output	SPI enable	Serial interface enable signal
	VCI_SCO_ CLK_OUT*[3]	Output	VCI clock	Supplies a clock to the voice CODEC IC.
	VCI_SCO_ SYNC_OUT*[3]	Output	VCI synchronization	Supplies a frame-synchronization signal to the voice CODEC IC.
	VCI_SCO_TX*[3]	Input	SCO transmit data	Inputs SCO data to be transmitted from the voice CODEC IC.
	VCI_SCO_RX*[3]	Output	SCO receive data	Outputs received SCO data to the voice CODEC IC.
	VCI_HWC*[3]	Output	VCI mode select	Selects operating mode of the voice CODEC IC (STLC7550).
	$\overline{\text{VCI_CODEC_ PWRDWN}}$*[3]	Output	VCI power down	Controls power-down for the voice CODEC IC.
	RTCSEL0	Input	Test pin	Test pin This pin should be fixed to low.
	RREF	Input	Test pin	Test pin This pin should be pulled-up.
	EXTAL2 XTAL2	Input Output	Crystal resonator connection for low-power clock	For connection to a crystal resonator for a low-power clock. When RTCSEL0 = high, these pins are inactive.

Classification	Symbol	I/O	Name	Function
Bluetooth interface (BT)	Vcc_28	Input	Power supply	Power supply pin for the RF-IC connection pin The same level should be supplied as the RF IC.
	Vss_28	Input	Ground	Ground pin for the RF-IC connection pin
D/A converter (DAC)	DA1, DA0	Output	Analog output pin	Analog output pin for the D/A converter
	AVcc (DAC)	Input	D/A analog power supply	Power supply pin for the D/A converter. When the D/A converter is not in use, connect this pin to the port power supply (Vcc).
	AVss (DAC)	Input	D/A analog ground	Ground pin for the D/A converter. Connect this pin to the system power supply (Vss).
I/O port	PTA6 to PTA0	I/O	General port	7-bit general I/O port pin
	PTB7 to PTB0	I/O	General port	8-bit general I/O port pin
	PTD5 to PTD0	I/O	General port	6-bit general I/O port pin
	PTE6 to PTE0	I/O	General port	7-bit general I/O port pin
	PTG4 to PTG0	I/O	General port	5-bit general I/O port pin
	PTH4 to PTH0	I/O	General port	5-bit general I/O port pin
User debugging interface (H-UDI)	TCK	Input	Test clock	Test-clock input pin
	TMS	Input	Test mode select	Inputs the test-mode select signal.
	TDI	Input	Test data input	Serial input pin for instructions and data
	TDO	Output	Test data output	Serial output pin for instructions and data
	$\overline{\text{TRST}}$	Input	Test reset	Initialization-signal input pin
Advanced user debugger (AUD)	AUDATA3 to AUDATA0	Output	AUD data	Destination-address output pin for branch instruction
	AUDCK	Output	AUD clock	Synchronization clock output pin
	$\overline{\text{AUDSYNC}}$	Output	AUD synchronization signal	Data start-position acknowledge-signal output pin

Classification	Symbol	I/O	Name	Function
E10A interface	$\overline{\text{ASEBRKAK}}$	Output	Break mode acknowledge	Indicates that the E10A emulator has entered its break mode. For the connection with the E10A, refer to the SH7660 E10A Emulator User's Manual (tentative title).
	$\overline{\text{ASEMD0}}$	Input	ASE mode	Sets the ASE mode.

Notes: 1. V_{DD}, V_{DD} (PLL), and V_{DD} (DLL) should not connect mutually in the short distance, should separate as much as possible, and give the noise measures.

2. These pins are connected to the RF IC. For details on the connection, refer to section 21, Bluetooth Interface (BT).

3. These pins are connected to the voice CODEC IC. For details on the connection, refer to section 21, Bluetooth Interface (BT).

Section 2 CPU

2.1 Processing States and Processing Modes

2.1.1 Processing States

This LSI supports four types of processing states: a reset state, an exception handling state, a program execution state, and a power-down state, according to the CPU processing states.

Reset State: In the reset state, the CPU is reset. The LSI supports two types of resets: power-on reset and manual reset. For details on resets, refer to section 4, Exception Handling.

In a power-on reset, the registers and internal statuses of all LSI on-chip modules are initialized. In a manual reset, the register contents of a part of the LSI on-chip modules, such as the bus state controller (BSC), are retained. For details, refer to section 27, List of Registers.

The CPU internal statuses and registers are initialized both in a power-on reset and manual reset. After initialization, the program branches to address H'A0000000 to pass control to the reset processing program defined by the user to be executed.

Exception Handling State: In the exception handling state, the CPU processing flow is changed temporarily by a general exception or interrupt exception processing. The program counter (PC) and status register (SR) are saved in the save program counter (SPC) and save status register (SSR), respectively. The program branches to an address obtained by adding a vector offset to the vector base register (VBR) and passes control to the exception handling program defined by the user to be executed.

For details on exception handling states, refer to section 4, Exception Handling.

Program Execution State: The CPU executes programs sequentially.

Power-Down State: The CPU stops operation to reduce power consumption. The power-down state can be entered by executing the SLEEP instruction. For details on the power-down state, refer to section 13, Power-Down Modes.

Figure 2.1 shows a status transition diagram.

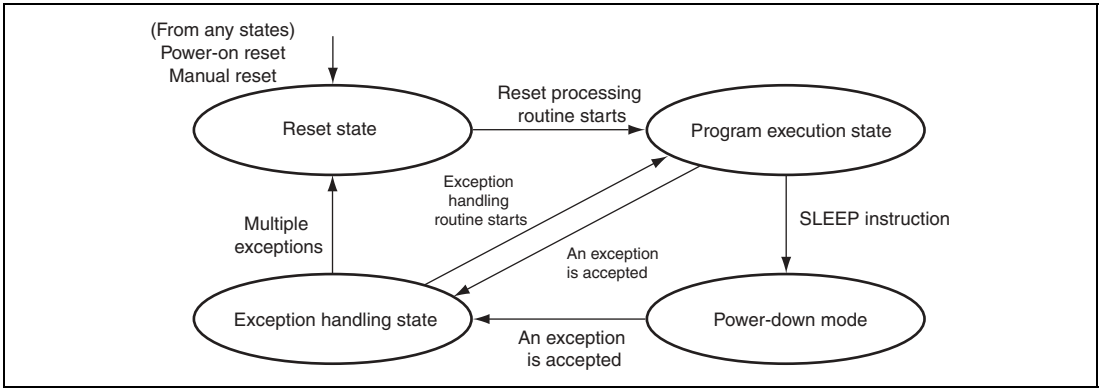


Figure 2.1 Processing State Transitions

2.1.2 Processing Modes (User Mode/Privileged Mode)

This LSI supports two processing modes: user mode and privileged mode. These processing modes can be determined by the processing mode bit (MD) in the status register (SR). If the MD bit is cleared to 0, user mode is selected. If the MD bit is set to 1, privileged mode is selected. The CPU automatically enters privileged mode by a transition to the reset state or exception handling state. In privileged mode, any registers and resources in address spaces can be accessed. For details on differences of registers and address spaces which can be accessed by the CPU in each processing mode, refer to section 2.2, Memory Map and section 2.3, Register Descriptions.

Writing 0 to the MD bit in SR puts the CPU in user mode. In user mode, some of the registers, including SR, and some of the address spaces cannot be accessed by the user program and system control instructions cannot be executed. This function effectively protects the system resources from the user program. To change the processing mode from user to privileged mode, a transition to the exception handling state is required.*[1]*[2]

Notes: 1. To call a service routine used in privileged mode from user mode, the LSI supports an unconditional trap instruction (TRAPA).
2. When a transition from user mode to privileged mode occurs, the contents of SR and PC are saved. A program execution in user mode can be resumed by restoring the contents of SR and PC. To return from an exception handling program, the LSI supports an RTE instruction.

2.2 Memory Map

The address spaces that can be accessed by the CPU mounted on this LSI differ depending on the processing mode as described above, and the address space handled by the CPU may not match the physical address space. To avoid misunderstandings, the address space which is output by the CPU is referred to as the logical address space while the actual physical address space is referred to as the physical address space. Furthermore, among the physical address space, the space to which this LSI can make external access is referred to as the external address space. These terms are used hereafter in this manual.

2.2.1 Logical Address Space

The address space which is output by the CPU is referred to as the logical address space. The CPU of this LSI supports a 32-bit logical address space and accesses system resources using the 4 Gbytes of logical address space. User programs and data are accessed from the logical address space. From the hardware view, the addresses output from the CPU are used on the L bus (see figure 1.1 in section 1, Overview). The logical address space is divided into several areas and managed as shown in figure 2.2.

In privileged mode, 4 Gbytes of space from P0 area to P4 area can be accessed.

In user mode, 2 Gbytes of space in U0 area can be accessed. When the DSP bit in SR is set to 1, 16 Mbytes of space in Uxy area can be also accessed. If areas other than U0 and Uxy are accessed in user mode, an address error occurs.

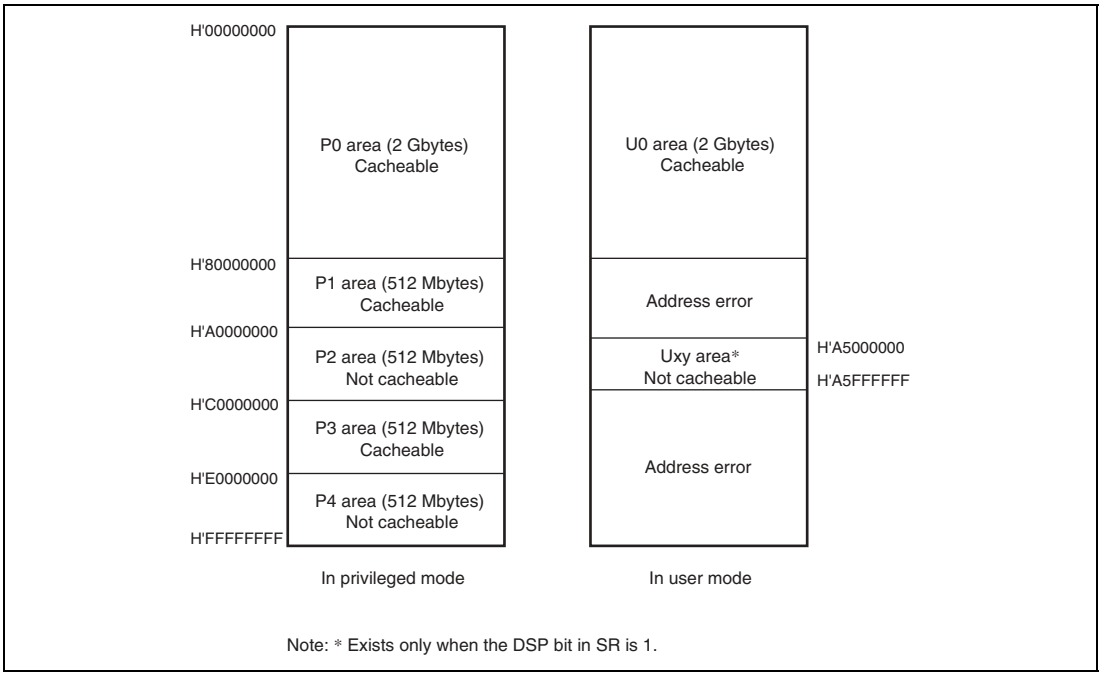


Figure 2.2 Logical Address Space

P0/U0 Area: This area is called the P0 area when the CPU is in privileged mode and the U0 area when in user mode. For the P0 and U0 areas, access using the cache is enabled.*

P1 Area: The P1 area is defined as a privileged area that is cacheable.* Normally, this area contains programs that operate at high-speed in privileged mode, such as the kernel of the operating system (OS) and the exception handler.

P2 Area: The P2 area is defined as a privileged area that is not cacheable. The reset processing program that is initiated when a transition is made to the reset state is written from the start address of the P2 area (H'A0000000). Normally, this area contains programs that are needed to initiate the OS, such as the system initial setting routine. Note that access to several on-chip I/O registers requires the program to be saved in the P2 area.

P3 Area: The P3 area is defined as a privileged area that is cacheable.*

P4 Area: The P4 area is a control space that is not cacheable and can be accessed only in privileged mode. Figure 2.3 shows the P4 area in detail. Several on-chip I/O registers are assigned to this area. For details on I/O registers which are assigned to this area, see section 27, List of Registers.

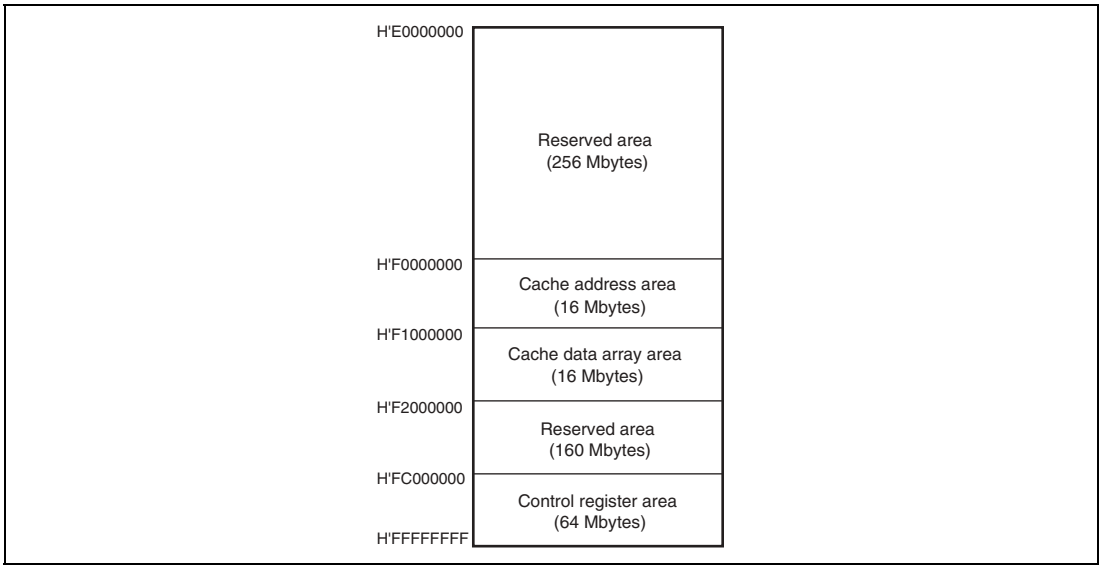


Figure 2.3 P4 Area

Uxy Area: The Uxy area that can be used when the DSP bit in SR is set to 1 in user mode is mapped to the on-chip memory of this LSI. Accessing this area when the DSP bit is 0 in user mode will result in an address error. This area cannot be accessed through the cache. For details on the Uxy area, see section 6, X/Y Memory and section 7, U Memory. For details on the DSP bit, see section 3, DSP Operating Unit.

Note: * Whether the cache is used or not is determined according to the CE bit in the cache control register (CCR1).

2.2.2 Physical Address Space

The physical address space obtained by address translation of addresses output from the CPU is referred to as the physical address space. From the hardware view, the addresses of the physical address space are used on the I bus. Similar to the logical address space, this LSI supports a 32-bit physical address space. However, as shown in figure 2.4, the upper three bits of the 32 bits are masked and handled as a shadow. Therefore, only 29 bits are actually used to access the physical address space and 0.5 Gbytes of physical memory can be accessed. Replacing the upper three bits of an address in this area with 0s makes the address in the corresponding physical address space.* For details on the physical address space, see section 9, Bus State Controller (BSC).

Bus masters other than the CPU, e.g. DMAC, are directly connected to the I bus. Therefore, instead of handling the logical address space, they directly handle the physical address space.

Note: * Since the on-chip I/O registers can be accessed only if the upper three bits of a logical address are set to 111 (corresponding to the P4 area in privileged mode), the upper three bits are output with the same values of 111 on the I bus.

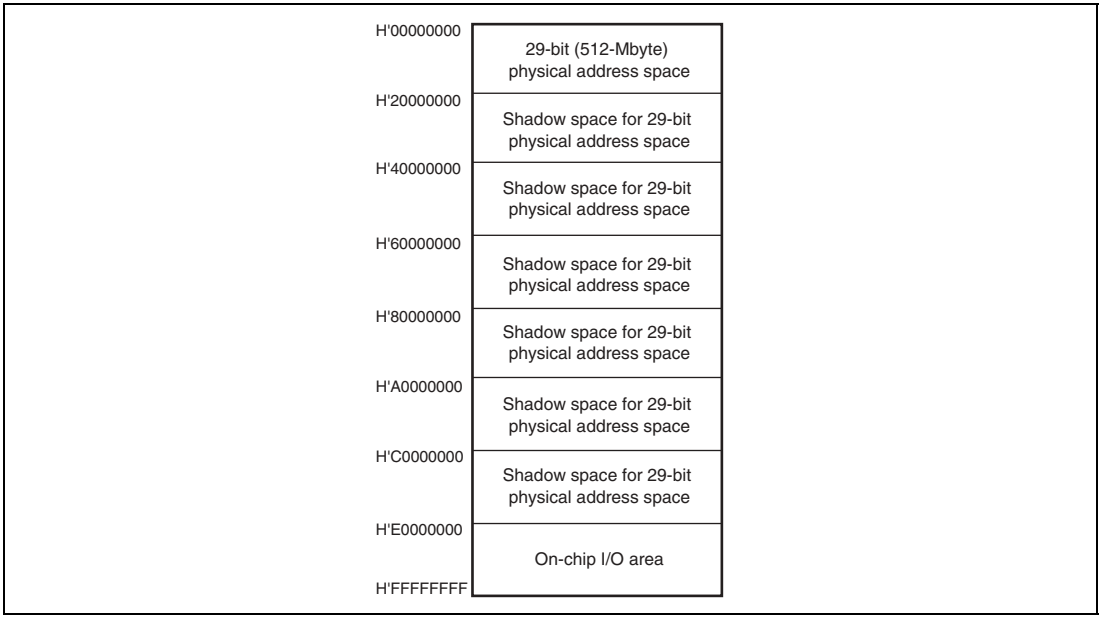


Figure 2.4 Physical Address Space

2.2.3 External Address Space

The physical address space is divided into eight areas as shown in figure 2.5. Area 1 is used as the on-chip I/O space, and most of the on-chip registers of this LSI are mapped to this area*. In this LSI, areas 2 and 5 to 7 are reserved areas. The remaining three areas, areas 0, 3, and 4, are used as the external address space. Each area of the external address space can be connected to different types of memory (for details, see section 9, Bus State Controller (BSC)).

The upper three bits of a logical address are normally masked from the CPU and treated as a shadow. In privileged mode, for example, logical addresses H'00000100 in the P0 area, H'80000100 in the P1 area, H'A0000100 in the P2 area, and H'C0000100 in the P3 area are all mapped to the same external address H'00000100 in area 0. However, since addresses cannot be mapped to the P4 area, the external address space cannot be accessed by an access to the P4 area.

The three areas (areas 0, 3, and 4) to which the external memory can be connected are each a 64-Mbyte external address space. Since this LSI has 24 address pins, a maximum of 16 Mbytes of memory can be mounted for each area. Figure 2.5 shows the mounted space corresponding to area 0.

Note: * To access an on-chip I/O register mapped to area 1 of the external address space, make an access from the P2 area of the logical address space that is not cacheable.

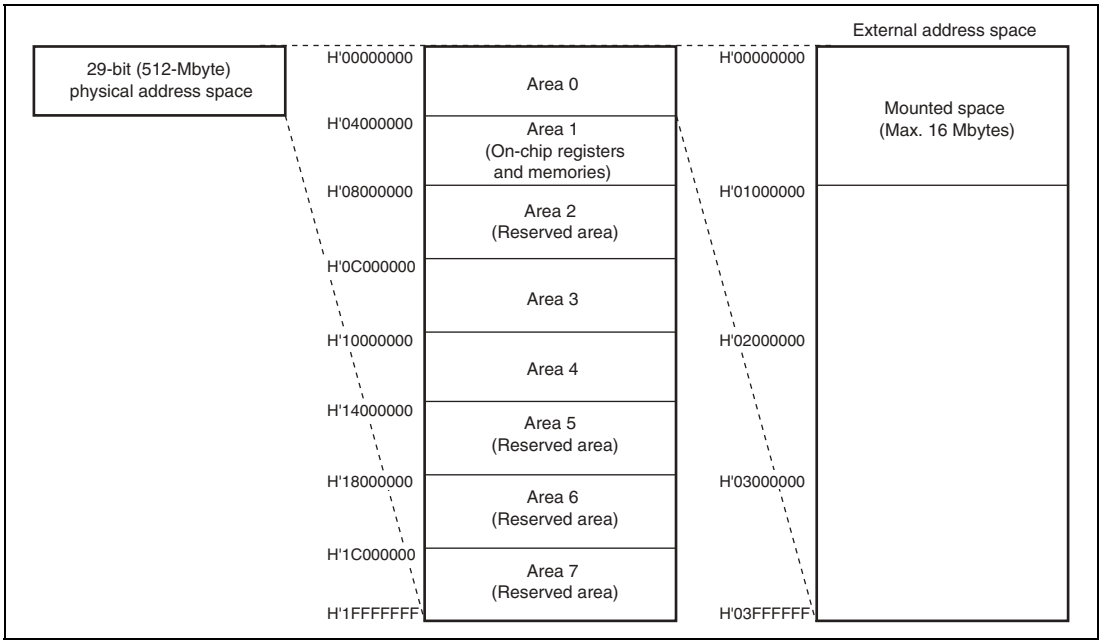


Figure 2.5 External Address Space and Mounted Space (Area 0)

2.3 Register Descriptions

The CPU of this LSI provides thirty-three 32-bit registers: 24 general registers, five control registers, three system registers, and one program counter.

General Registers: This LSI incorporates 24 general registers: R0_BANK0 to R7_BANK0, R0_BANK1 to R7_BANK1, and R8 to R15. R0 to R7 are banked. The processing mode and the register bank (RB) bit in the status register (SR) determine which set of banked registers (R0_BANK0 to R7_BANK0 or R0_BANK1 to R7_BANK1) are accessed as general registers.

System Registers: This LSI incorporates the multiply and accumulate registers (MACH/MACL) and procedure register (PR) as system registers. These registers can be accessed regardless of the processing mode.

Program Counter: The program counter (PC) stores the value obtained by adding 4 to the current instruction address.

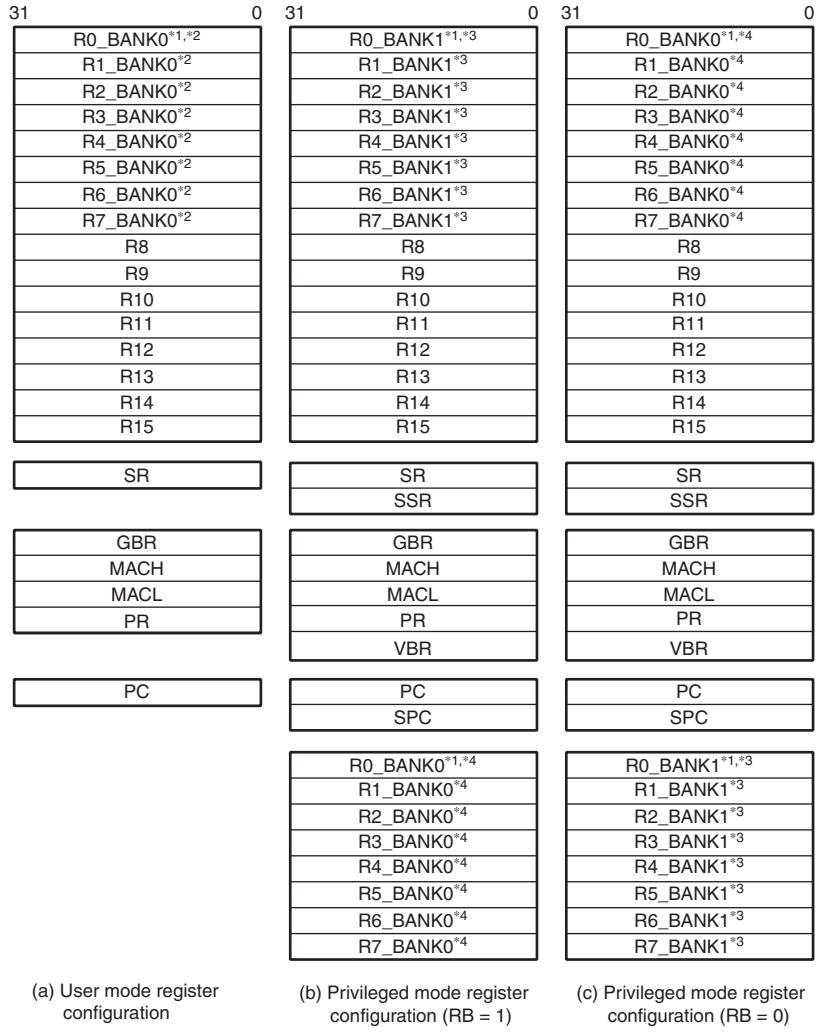
Control Registers: This LSI incorporates the status register (SR), global base register (GBR), save status register (SSR), save program counter (SPC), and vector base register (VBR) as control registers. Only GBR can be accessed in user mode. Control registers other than GBR can be accessed only in privileged mode.

Table 2.1 shows the register values after a reset. Figure 2.6 shows the register configurations in each processing mode.

Table 2.1 Register Initial Values

Register Type	Registers	Initial Values*
General registers	R0_BANK0 to R7_BANK0, R0_BANK1 to R7_BANK1, R8 to R15	Undefined
System registers	MACH, MACL, PR	Undefined
Program counter	PC	H'A0000000
Control registers	SR	MD bit = 1, RB bit = 1, BL bit = 1, I3 to I0 bits = B'1111 (H'F), DSP bit = 0, reserved bits = all 0, other bits = undefined
	GBR, SSR, SPC	Undefined
	VBR	H'00000000

Note: * Initialized by a power-on reset or manual reset.



Notes: 1. The R0 register is used as an index register in indexed register indirect addressing mode and indexed GBR indirect addressing mode.
2. Bank register
3. Bank register
Accessed as a general register when the RB bit is set to 1 in SR.
Accessed only by LDC/STC instructions when the RB bit is cleared to 0.
4. Bank register
Accessed as a general register when the RB bit is cleared to 0 in SR.
Accessed only by LDC/STC instructions when the RB bit is set to 1.

Figure 2.6 Register Configuration in Each Processing Mode

2.3.1 General Registers

There are twenty-four general registers: R0_BANK0 to R7_BANK0, R0_BANK1 to R7_BANK1, and R8 to R15. Figure 2.7 shows the general register configuration. R0 to R7 are banked. The processing mode and the register bank (RB) bit in the status register (SR) determine which set of banked registers (R0_BANK0 to R7_BANK0 or R0_BANK1 to R7_BANK1) are accessed as general registers. R0 to R7 registers in the selected bank are accessed as R0 to R7. R0 to R7 in the non-selected bank is accessed as R0_BANK to R7_BANK by the control register load instruction (LDC) and control register store instruction (STC).

In user mode, bank 0 is selected regardless of the RB bit value. Sixteen registers: R0_BANK0 to R7_BANK0 and R8 to R15 are accessed as general registers R0 to R15. R0_BANK1 to R7_BANK1 registers in bank 1 cannot be accessed.

In privileged mode that is entered by a transition to the exception handling state, the RB bit is set to 1 to select bank 1. In this case, sixteen registers: R0_BANK1 to R7_BANK1 in bank 1 and R8 to R15 are accessed as general registers R0 to R15. A bank is switched automatically when an exception handling state is entered, registers R0 to R7 need not be saved by the exception handling routine. R0_BANK0 to R7_BANK0 in bank 0 can be accessed as R0_BANK to R7_BANK by the LDC or STC instructions.

In privileged mode, bank 0 can also be used as general registers by clearing the RB bit to 0. In this case, sixteen registers: R0_BANK0 to R7_BANK0 in bank 0 and R8 to R15 are accessed as general registers R0 to R15. R0_BANK1 to R7_BANK1 in bank 1 can be accessed as R0_BANK to R7_BANK by the LDC or STC instructions.

The general registers R0 to R15 are used as equivalent registers for almost all instructions. In some instructions, R0 is implicitly used or only R0 may be used as source or destination registers.

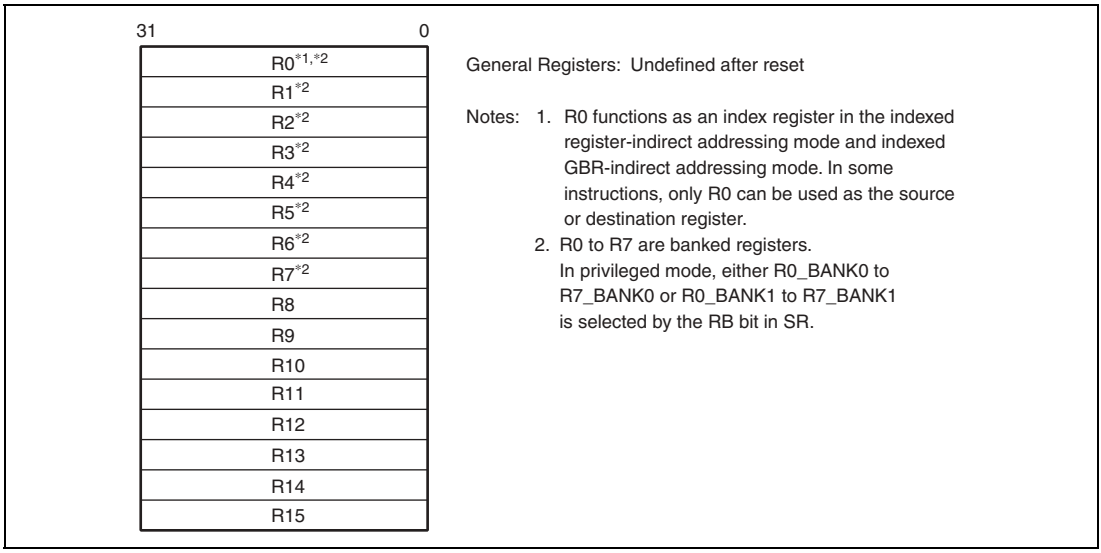


Figure 2.7 General Registers

2.3.2 System Registers

The system registers which are the following two registers can be accessed by the LDS or STS instructions. Figure 2.8 shows the system register configuration.

Multiply and Accumulate Registers: The multiply and accumulate registers store the results of multiplication and accumulation instructions or multiplication instructions. These registers also store additional values for the multiplication and accumulation instructions. After a reset, these registers are undefined.

The multiply and accumulate registers consist of the multiply and accumulate high register (MACH) which stores the upper 32 bits and multiply and accumulate low register (MACL) which stores lower 32 bits.

Procedure Register: The procedure register (PR) stores the return address for a subroutine call using the BSR, BSRF, or JSR instruction. The return address stored in PR is restored to the program counter (PC) by the RTS (return from the subroutine) instruction. After a reset, this register is undefined.

2.3.3 Program Counter

The program counter (PC) stores the value obtained by adding 4 to the current instruction address. Figure 2.8 shows the PC configuration. There is no instruction to read PC directly. Before an exception handling state is entered, the PC is saved in the save program counter (SPC). Before a subroutine call is executed, PC is saved in the procedure register (PR). In addition, PC can be used for PC relative addressing mode.

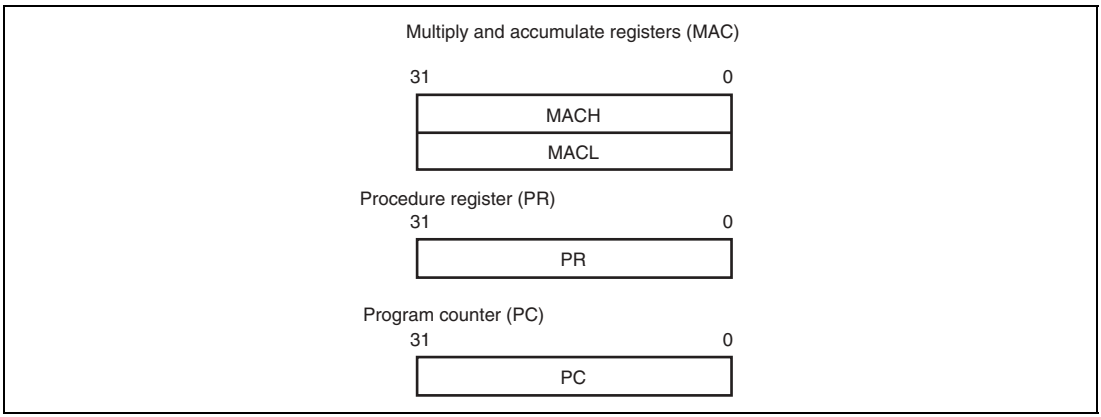


Figure 2.8 System Registers and Program Counter

2.3.4 Control Registers

The control registers can be accessed by the LDC or STC instruction in privileged mode. The global base register (GBR) can also be accessed in user mode. The control registers are the following five registers.

Status Register (SR): SR stores various information which indicates the system status. SR can be accessed only in privileged mode.

Bit	Bit Name	Initial Value	R/W	Description
31	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. When 1 is written to this bit, operation cannot be guaranteed.
30	MD	1	R/W	Processing Mode Indicates the CPU processing mode. 0: User mode 1: Privileged mode The MD bit is set to 1 after a reset or in an exception handling state.
29	RB	1	R/W	Register Bank The general registers R0 to R7 are banked registers. The RB bit selects a bank used in privileged mode. 0: Selects bank 0 registers. In this case, R0_BANK0 to R7_BANK0 and R8 to R15 are used as general registers. R0_BANK1 to R7_BANK1 can be accessed by the LDC or STC instruction. 1: Selects bank 1 registers. In this case, R0_BANK1 to R7_BANK1 and R8 to R15 are used as general registers. R0_BANK0 to R7_BANK0 can be accessed by the LDC or STC instruction. The RB bit is set to 1 after a reset or in an exception handling state.
28	BL	1	R/W	Block 0: Enables an interrupt or user break. 1: Disables an interrupt or user break. The BL bit is set to 1 after a reset or in an exception handling state.
27 to 10	—	All 0	R	Reserved*2 These bits are always read as 0. The write value should always be 0. When 1 is written to these bits, operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
9	M	—	R/W	M Bit*[1]
8	Q	—	R/W	Q Bit*[1] These bits are used by the DIV0S, DIV0U, and DIV1 instructions. These bits can be changed even in user mode by executing these instructions. These bits are undefined at a reset. These bits do not change in an exception handling state.
7	I3	1	R/W	Interrupt Mask Bits
6	I2	1	R/W	The 4-bit data indicates the interrupt mask level. These bits do not change even if an interrupt occurs. At a reset, these bits are initialized to B'1111. These bits are not affected in an exception handling state.
5	I1	1	R/W	
4	I0	1	R/W	
3, 2	—	All 0	R	Reserved*[2] These bits are always read as 0. The write value should always be 0. When 1 is written to these bits, operation cannot be guaranteed.
1	S	—	R/W	Saturation Mode*[1] Specifies saturation mode for multiply instructions or multiply and accumulate instructions. This bit can be specified by the SETS and CLRS instructions in user mode. At a reset, this bit is undefined. This bit is not affected in an exception handling state.
0	T	—	R/W	T Bit*[1] Indicates true or false for compare instructions or carry or borrow occurrence for an operation instruction with carry or borrow. This bit can be specified by the SETT and CLRT instructions in user mode. At a reset, this bit is undefined. This bit is not affected in an exception handling state.

Notes: 1. The M, Q, S, and T bits can be set/cleared by the user mode specific instructions. Other bits can be read or written in privileged mode.
2. When DSP is used, this bit is extended. Refer to section 3.2.3, CPU Register Sets.

Save Status Register (SSR): The save status register (SSR) can be accessed only in privileged mode. Before entering the exception handling state, the contents of SR are saved. At a reset, the SSR initial value is undefined.

Save Program Counter (SPC): The save program counter (SPC) can be accessed only in privileged mode. Before entering the exception handling state, the contents of PC are saved. At a reset, the SPC initial value is undefined.

Global Base Register (GBR): The global base register (GBR) is referenced as a base register in GBR indirect addressing mode. At a reset, the GBR initial value is undefined.

Vector Base Register (VBR): The vector base register (VBR) can be accessed only in privileged mode. If a transition from a state other than a reset state to exception handling state occurs, this register is referenced as a base address of the branch destination. For details, refer to section 4, Exception Handling. At a reset, VBR is initialized to H'00000000.

Figure 2.9 shows the control register configuration.

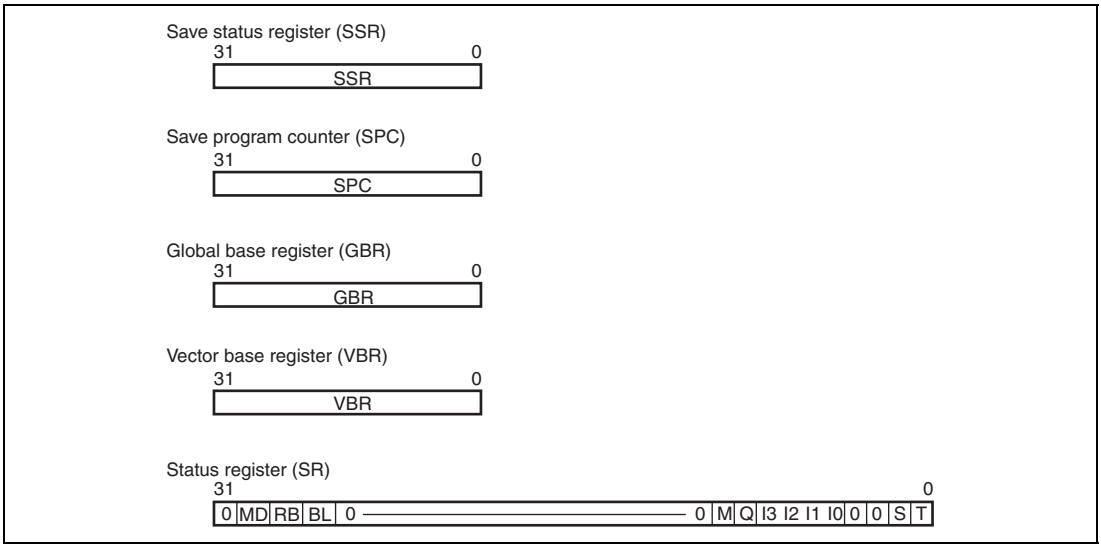
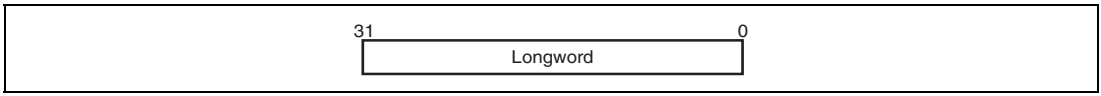


Figure 2.9 Control Register Configuration

2.4 Data Formats

2.4.1 Register Data Format

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



2.4.2 Memory Data Formats

Memory data formats are classified into byte, word, and longword. Memory can be accessed in bytes, words, and longwords. When the memory operand is only a byte (8 bits) or a word (16 bits), it is sign-extended into a longword when loaded into a register.

Word operand must be accessed from word boundary (even address in two bytes: address 2n) and longword operand must be accessed from longword boundary (even address in four bytes: address 4n). Otherwise, an address error will occur and an exception handling state will be entered. Byte operand can be accessed from any address.

When a word or longword operand is accessed, the byte positions on the memory corresponding to the word or longword data on the register is determined according to the specified endian mode (big endian or little endian).

Figure 2.10 shows a byte correspondence in big endian mode. In big endian mode, the MSB byte in the register corresponds to the lowest address in the memory, and the LSB byte in the register corresponds to the highest address. For example, if the contents of the general register R0 is stored at an address indicated by the general register R1 in longwords, the MSB byte in R0 is stored at the address indicated by the R1 and the LSB byte in R0 is stored at the address indicated by the (R1 + 3).

The on-chip device registers assigned to memory are accessed in big endian mode. Note that the available access size (byte, word, or longword) differs in each register.

Note: The CPU instruction codes of this LSI must be stored in words. In big endian mode, the instruction code must be stored from upper byte to lower byte in this order from the word boundary on the memory.

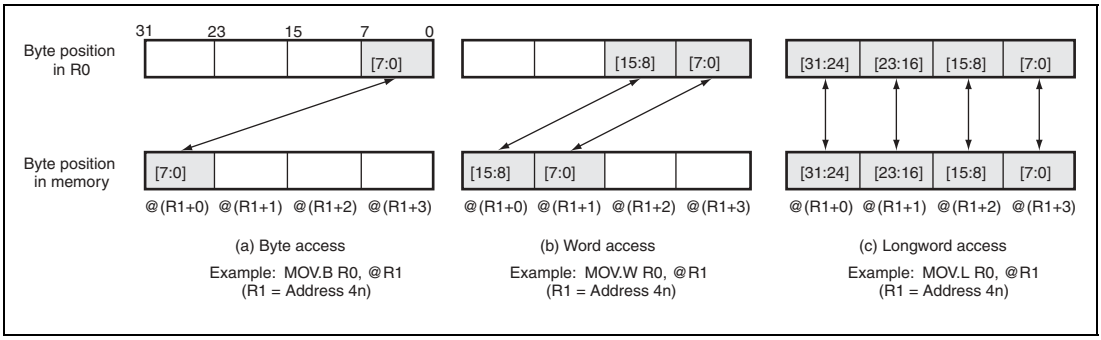


Figure 2.10 Data Format on Memory (Big Endian Mode)

The little endian mode can also be specified as data format. Either big endian or little endian mode should be selected according to the external pin (MD5 pin) at a power-on reset. When the MD5 pin goes low, the processor operates in big endian mode. When the MD5 pin goes high, the processor operates in little endian mode. The endian mode cannot be changed dynamically.

In little endian mode, the MSB byte in the register corresponds to the highest address in the memory, and the LSB byte in the register corresponds to the lowest address (figure 2.11). For example, if the contents of the general register R0 is stored at an address indicated by the general register R1 in longwords, the MSB byte of R0 is stored at the address indicated by the (R1 + 3) and the LSB byte of R0 is stored at the address indicated by R1.

If little endian mode is selected, the on-chip memory of this LSI is accessed in little endian mode. However, the on-chip device registers assigned to memory are accessed in big endian mode. Note that the available access size (byte, word, or longword) differs in each register.

Note: The CPU instruction codes of this LSI must be stored in words. In little endian mode, the instruction code must be stored from lower byte to upper byte in this order from the word boundary on the memory.

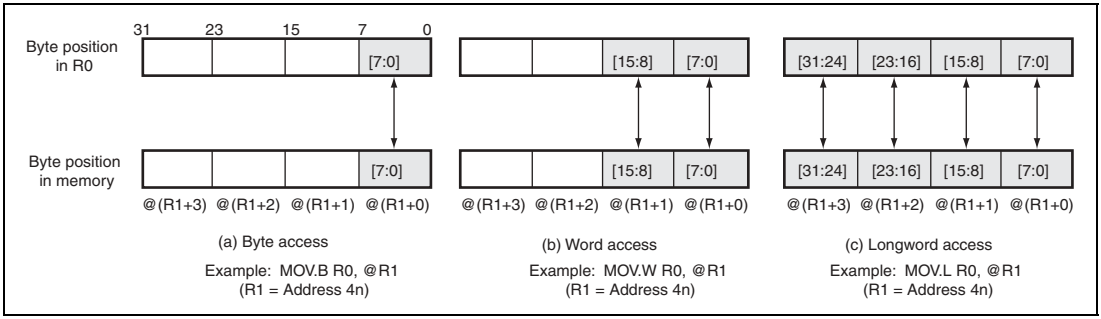


Figure 2.11 Data Format on Memory (Little Endian Mode)

2.5 Features of Instructions

2.5.1 Instruction Execution Method

Instruction Length: All instructions have a fixed length of 16 bits and are executed in the sequential pipeline. In the sequential pipeline, almost all instructions can be executed in one cycle. All data are handled in longwords (32 bits). Memory can be accessed in bytes, words, or longwords. In this case, byte or word data is sign-extended and handled as longword data. Immediate data is sign-extended to longword size for arithmetic operations (MOV, ADD, and CMP/EQ instructions) or zero-extended to longword size for logical operations (TST, AND, OR, and XOR instructions).

Load/Store Architecture: Basic operations are executed between registers. In operations involving memory, data is first loaded into a register (load/store architecture). However, bit manipulation instructions such as AND are executed directly on memory.

Delayed Branching: Unconditional branch instructions are executed as delayed branches. With a delayed branch instruction, the branch is made after execution of the instruction immediately following the delayed branch instruction. This minimizes disruption of the pipeline when a branch is made. An example is shown below. This LSI supports two types of conditional branch instructions: delayed branch instruction and normal branch instruction.

```
        BRA     TRGET
        ADD     R1, R0      ; ADD instruction is executed before
                                branching to the TRGET.
```

T Bit: The result of a comparison is indicated by the T bit in the status register (SR), and a conditional branch is performed according to whether the result is True or False. Processing speed has been improved by keeping the number of instructions that modify the T bit to a minimum. An example of using the T bit is shown below.

```
        ADD     #1, R0      ; The T bit cannot be modified by the
ADD                         instruction.
        CMP/EQ  #0, R0      ; The T bit is set to 1 if R0 is 0.
        BT      TRGET       ; Branch to the TRGET if the T bit is
set                             to 1 (R0 = 0).
```

Literal Constant: Byte literal constant is placed inside the instruction code as immediate data. Since the instruction length is fixed to 16 bits, word or longword literal constant is not placed inside the instruction code, but stored as a table in main memory. The table in memory is referenced with a MOV instruction using PC-relative addressing mode with displacement. An example is shown below.

```
MOV.W     @(disp, PC), R0
```

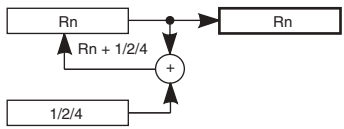
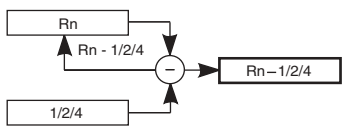
Absolute Addresses: The absolute address value should be placed in a table in main memory as well as word or longword literal constant. This value is transferred to a register and the operand access is specified using indexed register indirect addressing mode. The absolute address value is stored in a register during execution of an instruction. This is also applied to the word or longword immediate data.

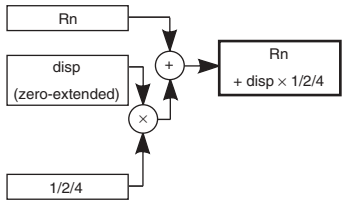
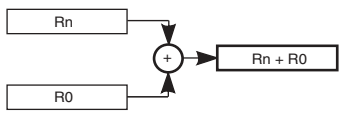
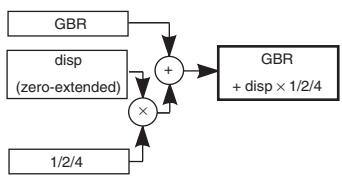
16-Bit/32-Bit Displacement: When data is referenced with a 16- or 32-bit displacement, the displacement value is placed in a table in memory beforehand. As well as the absolute addresses, this value is transferred to a register and the operand access is specified using indexed register indirect addressing mode. The displacement value is stored in a register during execution of an instruction. This is also applied to the word or longword immediate data.

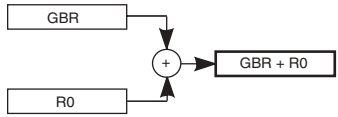
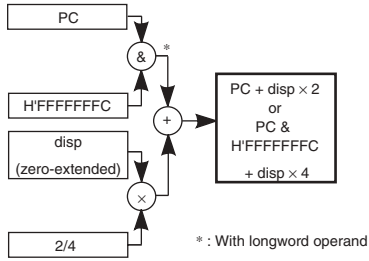
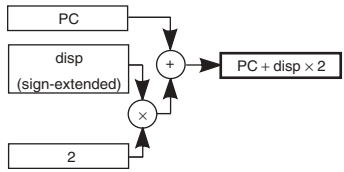
2.5.2 Addressing Modes

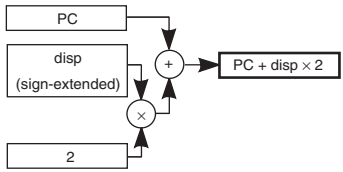
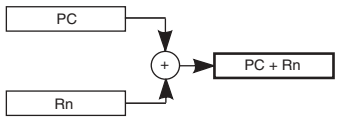
Table 2.2 shows addressing modes and effective address calculation methods.

Table 2.2 Addressing Modes and Effective Addresses

Addressing Mode	Instruction Format	Effective Address Calculation Method	Calculation Formula
Register direct	Rn	Effective address is register Rn. (Operand is register Rn contents.)	—
Register indirect	@Rn	Effective address is register Rn contents. 	Rn
Register indirect with post-increment	@Rn+	Effective address is register Rn contents. A constant is added to Rn after instruction execution: 1 for a byte operand, 2 for a word operand, or 4 for a longword operand. 	Rn After instruction execution Byte: Rn + 1 → Rn Word: Rn + 2 → Rn Longword: Rn + 4 → Rn
Register indirect with pre-decrement	@–Rn	Effective address is register Rn contents, decremented by a constant beforehand: 1 for a byte operand, 2 for a word operand, or 4 for a longword operand. 	Byte: Rn – 1 → Rn Word: Rn – 2 → Rn Longword: Rn – 4 → Rn (Instruction executed with Rn after calculation)

Addressing Mode	Instruction Format	Effective Address Calculation Method	Calculation Formula
Register indirect with displacement	@(disp:4, Rn)	Effective address is register Rn contents with 4-bit displacement disp added. After disp is zero-extended, it is multiplied by 1 (byte), 2 (word), or 4 (longword), according to the operand size. 	Byte: Rn + disp Word: Rn + disp × 2 Longword: Rn + disp × 4
Indexed register indirect	@(R0, Rn)	Effective address is sum of register Rn and R0 contents. 	Rn + R0
GBR indirect with displacement	@(disp:8, GBR)	Effective address is register GBR contents with 8-bit displacement disp added. After disp is zero-extended, it is multiplied by 1 (byte), 2 (word), or 4 (longword), according to the operand size. 	Byte: GBR + disp Word: GBR + disp × 2 Longword: GBR + disp × 4

Addressing Mode	Instruction Format	Effective Address Calculation Method	Calculation Formula
Indexed GBR indirect	@(R0, GBR)	Effective address is sum of register GBR and R0 contents. 	GBR + R0
PC-relative with displacement	@(disp:8, PC)	Effective address is register PC with 8-bit displacement disp added. After disp is zero-extended, it is multiplied by 2 (word) or 4 (longword), according to the operand size. With a longword operand, the lower 2 bits of PC are masked. 	Word: PC + disp × 2 Longword: PC&H'FFFFFFFC + disp × 4
PC-relative	disp:8	Effective address is register PC with 8-bit displacement disp added after being sign-extended and multiplied by 2. 	PC + disp × 2

Addressing Mode	Instruction Format	Effective Address Calculation Method	Calculation Formula
PC-relative	disp:12	Effective address is register PC with 12-bit displacement disp added after being sign-extended and multiplied by 2. 	PC + disp × 2
	Rn	Effective address is sum of register PC and Rn contents. 	PC + Rn
Immediate	#imm:8	8-bit immediate data imm of TST, AND, OR, or XOR instruction is zero-extended.	—
	#imm:8	8-bit immediate data imm of MOV, ADD, or CMP/EQ instruction is sign-extended.	—
	#imm:8	8-bit immediate data imm of TRAPA instruction is zero-extended and multiplied by 4.	—

Note: For addressing modes with displacement (disp) as shown below, the assembler description in this manual indicates the value before it is scaled (x 1, x 2, or x 4) according to the operand size to clarify the LSI operation. For details on actual assembler description, refer to the description rules in each assembler.

@ (disp:4, Rn) ; Register indirect with displacement
@ (disp:8, GBR) ; GBR indirect with displacement
@ (disp:8, PC) ; PC relative with displacement
disp:8, disp:12 ; PC relative

2.5.3 Instruction Formats

Table 2.3 shows the instruction formats and the meaning of the source and destination operands. The meaning of the operands depends on the instruction code. The following symbols are used in the table.

xxxx: Instruction code

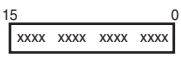
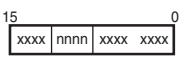
mmmm: Source register

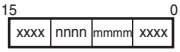
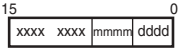
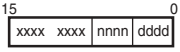
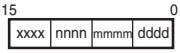
nnnn: Destination register

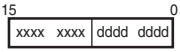
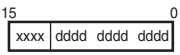
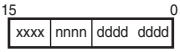
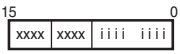
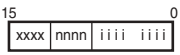
iiii: Immediate data

dddd: Displacement

Table 2.3 Instruction Formats

Instruction Format	Source Operand	Destination Operand	Sample Instruction
0 type 	—	—	NOP
n type 	—	nnnn: register direct	MOVT Rn
	Control register or system register	nnnn: register direct	STS MACH,Rn
	Control register or system register	nnnn: pre-decrement register indirect	STC.L SR,@-Rn
m type 	mmmm: register direct	Control register or system register	LDC Rm,SR
	mmmm: post-increment register indirect	Control register or system register	LDC.L @Rm+,SR
	mmmm: register indirect	—	JMP @Rm
	mmmm: PC-relative using Rm	—	BRAF Rm

Instruction Format	Source Operand	Destination Operand	Sample Instruction
nm type 	mmmm: register direct	nnnn: register direct	ADD Rm,Rn
	mmmm: register direct	nnnn: register indirect	MOV.L Rm,@Rn
	mmmm: post-increment register indirect (multiply-and-accumulate operation) nnnn: *post-increment register indirect (multiply-and-accumulate operation)	MACH, MACL	MAC.W @Rm+,@Rn+
	mmmm: post-increment register indirect	nnnn: register direct	MOV.L @Rm+,Rn
	mmmm: register direct	nnnn: pre-decrement register indirect	MOV.L Rm,@-Rn
	mmmm: register direct	nnnn: indexed register indirect	MOV.L Rm,@(R0,Rn)
md type 	mmmmdddd: register indirect with displacement	R0 (register direct)	MOV.B @(disp,Rm),R0
nd4 type 	R0 (register direct)	nnnndddd: register indirect with displacement	MOV.B R0,@(disp,Rn)
nmd type 	mmmm: register direct	nnnndddd: register indirect with displacement	MOV.L Rm,@(disp,Rn)
	mmmmdddd: register indirect with displacement	nnnn: register direct	MOV.L @(disp,Rm),Rn

Instruction Format	Source Operand	Destination Operand	Sample Instruction
d type 	ddddddddd: GBR indirect with displacement	R0 (register direct)	MOV.L @(disp,GBR),R0
	R0 (register direct)	ddddddddd: GBR indirect with displacement	MOV.L R0,@(disp,GBR)
	ddddddddd: PC-relative with displacement	R0 (register direct)	MOVA @(disp,PC),R0
	ddddddddd: PC-relative	—	BF label
d12 type 	ddddddddddddd: PC-relative	—	BRA label (label=disp+PC)
nd8 type 	ddddddddd: PC-relative with displacement	nnnn: register direct	MOV.L @(disp,PC),Rn
i type 	iiiiiiii: immediate	Indexed GBR indirect	AND.B #imm,@(R0,GBR)
	iiiiiiii: immediate	R0 (register direct)	AND #imm,R0
	iiiiiiii: immediate	—	TRAPA #imm
ni type 	iiiiiiii: immediate	nnnn: register direct	ADD #imm,Rn

Note: * In multiply-and-accumulate instructions, nnnn is the source register.

2.6 Instruction Set

2.6.1 Instruction Set Based on Functions

Table 2.4 lists the instruction types based on functions.

Table 2.4 CPU Instruction Types

Type	Kinds of Instruction	Op Code	Function	Number of Instructions
Data transfer instructions	5	MOV	Data transfer	39
		MOVA	Effective address transfer	
		MOVT	T bit transfer	
		SWAP	Upper/lower swap	
		XTRCT	Extraction of middle of linked registers	
Arithmetic operation instructions	21	ADD	Binary addition	33
		ADDC	Binary addition with carry	
		ADDV	Binary addition with overflow	
		CMP/cond	Comparison	
		DIV1	Division	
		DIV0S	Signed division initialization	
		DIV0U	Unsigned division initialization	
		DMULS	Signed double-precision multiplication	
		DMULU	Unsigned double-precision multiplication	
		DT	Decrement and test	
		EXTS	Sign extension	
		EXTU	Zero extension	
		MAC	Multiply-and-accumulate, double-precision multiply-and-accumulate	
		MUL	Double-precision multiplication (32×32 bits)	

Type	Kinds of Instruction	Op Code	Function	Number of Instructions
Arithmetic operation instructions	21	MULS	Signed multiplication (16 × 16 bits)	33
		MULU	Unsigned multiplication (16 × 16 bits)	
		NEG	Sign inversion	
		NEGC	Sign inversion with borrow	
		SUB	Binary subtraction	
		SUBC	Binary subtraction with borrow	
		SUBV	Binary subtraction with underflow	
Logic operation instructions	6	AND	Logical AND	14
		NOT	Bit inversion	
		OR	Logical OR	
		TAS	Memory test and bit setting	
		TST	Logical AND and T bit setting	
		XOR	Exclusive logical OR	
Shift instructions	12	ROTCL	1-bit left rotate with T bit	16
		ROTCR	1-bit right rotate with T bit	
		ROTL	1-bit left rotate	
		ROTR	1-bit right rotate	
		SHAD	Arithmetic dynamic shift	
		SHAL	Arithmetic 1-bit left shift	
		SHAR	Arithmetic 1-bit right shift	
		SHLD	Logical dynamic shift	
		SHLL	Logical 1-bit left shift	
		SHLLn	Logical n-bit left shift	
		SHLR	Logical 1-bit right shift	
		SHLRn	Logical n-bit right shift	

Type	Kinds of Instruction	Op Code	Function	Number of Instructions
Branch instructions	9	BF	Conditional branch, delayed conditional branch (T = 0)	11
		BT	Conditional branch, delayed conditional branch (T = 1)	
		BRA	Unconditional branch	
		BRAF	Unconditional branch	
		BSR	Branch to subroutine procedure	
		BSRF	Branch to subroutine procedure	
		JMP	Unconditional branch	
		JSR	Branch to subroutine procedure	
		RTS	Return from subroutine procedure	
System control instructions	14	CLRMAC	MAC register clear	74
		CLRS	S bit clear	
		CLRT	T bit clear	
		LDC	Load into control register	
		LDS	Load into system register	
		NOP	No operation	
		PREF	Data prefetch to cache	
		RTE	Return from exception handling	
		SETS	S bit setting	
		SETT	T bit setting	
		SLEEP	Transition to power-down mode	
		STC	Store from control register	
		STS	Store from system register	
		TRAPA	Trap exception handling	
Total:	67			187

The instruction code, operation, and number of execution states of the CPU instructions are shown in tables 2.5 to 2.10, classified by instruction type, using the format shown below.

Instruction	Instruction Code	Operation	Privilege	Execution States	T Bit
Indicated by mnemonic.	Indicated in MSB ↔ LSB order.	Indicates summary of operation.	Indicates a privileged instruction.	Value when no wait states are inserted*1	Value of T bit after instruction is executed
Legend	Legend	Legend			Legend
OP.Sz SRC, DEST OP: Operation code Sz: Size SRC: Source DEST: Destination	mmmm: Source register nnnn: Destination register 0000: R0 0001: R1 ..... 1111: R15	→, ←: Transfer direction (xx): Memory operand M/Q/T: Flag bits in SR &: Logical AND of each bit |: Logical OR of each bit ^: Exclusive logical OR of each bit ~: Logical NOT of each bit <<n: n-bit left shift >>n: n-bit right shift			—: No change
Rm: Source register	iiii: Immediate data				
Rn: Destination register	dddd: Displacement*2				
imm: Immediate data					
disp: Displacement					

Notes: 1. The table shows the minimum number of execution states. In practice, the number of instruction execution states will be increased in cases such as the following:

a. When there is a conflict between an instruction fetch and a data access

b. When the destination register of a load instruction (memory → register) is also used by the following instruction

2. Scaled (x 1, x 2, or x 4) according to the instruction operand size, etc.

Table 2.5 Data Transfer Instructions

Instruction		Instruction Code	Operation	Privileged Mode	Cycles	T Bit
MOV	#imm,Rn	1110nnnniiiiiiii	imm → Sign extension → Rn	—	1	—
MOV.W	@(disp,PC),Rn	1001nnnndddddddd	(disp x 2+PC)→Sign extension → Rn	—	1	—
MOV.L	@(disp,PC),Rn	1101nnnndddddddd	(disp x 4+PC)→Rn	—	1	—
MOV	Rm,Rn	0110nnnnmmmm0011	Rm→Rn	—	1	—
MOV.B	Rm,@Rn	0010nnnnmmmm0000	Rm→(Rn)	—	1	—
MOV.W	Rm,@Rn	0010nnnnmmmm0001	Rm→(Rn)	—	1	—
MOV.L	Rm,@Rn	0010nnnnmmmm0010	Rm→(Rn)	—	1	—
MOV.B	@Rm,Rn	0110nnnnmmmm0000	(Rm)→Sign extension→Rn	—	1	—
MOV.W	@Rm,Rn	0110nnnnmmmm0001	(Rm)→Sign extension→Rn	—	1	—
MOV.L	@Rm,Rn	0110nnnnmmmm0010	(Rm)→Rn	—	1	—
MOV.B	Rm,@–Rn	0010nnnnmmmm0100	Rn–1→Rn, Rm→(Rn)	—	1	—
MOV.W	Rm,@–Rn	0010nnnnmmmm0101	Rn–2→Rn, Rm→(Rn)	—	1	—
MOV.L	Rm,@–Rn	0010nnnnmmmm0110	Rn–4→Rn, Rm→(Rn)	—	1	—
MOV.B	@Rm+,Rn	0110nnnnmmmm0100	(Rm)→Sign extension→Rn, Rm+1→Rm	—	1	—
MOV.W	@Rm+,Rn	0110nnnnmmmm0101	(Rm)→Sign extension→Rn, Rm+2→Rm	—	1	—
MOV.L	@Rm+,Rn	0110nnnnmmmm0110	(Rm)→Rn, Rm+4→Rm	—	1	—
MOV.B	R0,@(disp,Rn)	10000000nnnndddd	R0→(disp+Rn)	—	1	—
MOV.W	R0,@(disp,Rn)	10000001nnnndddd	R0→(disp x 2+Rn)	—	1	—
MOV.L	Rm,@(disp,Rn)	0001nnnnmmmmdddd	Rm→(disp x 4+Rn)	—	1	—
MOV.B	@(disp,Rm),R0	10000100mmmmdddd	(disp+Rm)→Sign extension→R0	—	1	—
MOV.W	@(disp,Rm),R0	10000101mmmmdddd	(disp x 2+Rm)→Sign extension→R0	—	1	—
MOV.L	@(disp,Rm),Rn	0101nnnnmmmmdddd	(disp x 4+Rm)→Rn	—	1	—
MOV.B	Rm,@(R0,Rn)	0000nnnnmmmm0100	Rm→(R0+Rn)	—	1	—
MOV.W	Rm,@(R0,Rn)	0000nnnnmmmm0101	Rm→(R0+Rn)	—	1	—
MOV.L	Rm,@(R0,Rn)	0000nnnnmmmm0110	Rm→(R0+Rn)	—	1	—

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

Table 2.6 Arithmetic Operation Instructions

Instruction		Instruction Code	Operation	Privileged Mode	Cycles	T Bit
ADD	Rm,Rn	0011nnnnmmmm1100	Rn+Rm→Rn	—	1	—
ADD	#imm,Rn	0111nnnniiiiiiii	Rn+imm→Rn	—	1	—
ADDC	Rm,Rn	0011nnnnmmmm1110	Rn+Rm+T→Rn, Carry→T	—	1	Carry
ADDV	Rm,Rn	0011nnnnmmmm1111	Rn+Rm→Rn, Overflow→T	—	1	Overflow
CMP/EQ	#imm,R0	10001000iiiiiiii	If R0 = imm, 1 → T	—	1	Comparison result
CMP/EQ	Rm,Rn	0011nnnnmmmm0000	If Rn = Rm, 1 → T	—	1	Comparison result
CMP/HS	Rm,Rn	0011nnnnmmmm0010	If Rn ≥ Rm with unsigned data, 1 → T	—	1	Comparison result
CMP/GE	Rm,Rn	0011nnnnmmmm0011	If Rn ≥ Rm with signed data, 1 → T	—	1	Comparison result
CMP/HI	Rm,Rn	0011nnnnmmmm0110	If Rn > Rm with unsigned data, 1 → T	—	1	Comparison result
CMP/GT	Rm,Rn	0011nnnnmmmm0111	If Rn > Rm with signed data, 1 → T	—	1	Comparison result
CMP/PL	Rn	0100nnnn00010101	If Rn > 0, 1 → T	—	1	Comparison result
CMP/PZ	Rn	0100nnnn00010001	If Rn ≥ 0, 1 → T	—	1	Comparison result
CMP/STR	Rm,Rn	0010nnnnmmmm1100	If Rn and Rm have an equivalent byte, 1 → T	—	1	Comparison result
DIV1	Rm,Rn	0011nnnnmmmm0100	Single-step division (Rn/Rm)	—	1	Calculation result
DIV0S	Rm,Rn	0010nnnnmmmm0111	MSB of Rn → Q, MSB of Rm → M, M ^ Q → T	—	1	Calculation result
DIV0U		0000000000011001	0 → M/Q/T	—	1	0
DMULS.L	Rm,Rn	0011nnnnmmmm1101	Signed operation of Rn × Rm → MACH, MACL 32 × 32 → 64 bits	—	2 (to 5)*	—
DMULU.L	Rm,Rn	0011nnnnmmmm0101	Unsigned operation of Rn × Rm → MACH, MACL 32 × 32 → 64 bits	—	2 (to 5)*	—
DT	Rn	0100nnnn00010000	Rn – 1 → Rn, if Rn = 0, 1 → T, else 0 → T	—	1	Comparison result

Instruction		Instruction Code	Operation	Privileged Mode	Cycles	T Bit
EXTS.B	Rm,Rn	0110nnnnmmmm1110	A byte in Rm is sign-extended → Rn	—	1	—
EXTS.W	Rm,Rn	0110nnnnmmmm1111	A word in Rm is sign-extended → Rn	—	1	—
EXTU.B	Rm,Rn	0110nnnnmmmm1100	A byte in Rm is zero-extended → Rn	—	1	—
EXTU.W	Rm,Rn	0110nnnnmmmm1101	A word in Rm is zero-extended → Rn	—	1	—
MAC.L	@Rm+, @Rn+	0000nnnnmmmm1111	Signed operation of (Rn) × (Rm) + MAC → MAC, Rn + 4 → Rn, Rm + 4 → Rm 32 × 32 + 64 → 64 bits	—	2 (to 5)*	—
MAC.W	@Rm+, @Rn+	0100nnnnmmmm1111	Signed operation of (Rn) × (Rm) + MAC → MAC, Rn + 2 → Rn, Rm + 2 → Rm 16 × 16 + 64 → 64 bits	—	2 (to 5)*	—
MUL.L	Rm,Rn	0000nnnnmmmm0111	Rn × Rm → MACL 32 × 32 → 32 bits	—	2 (to 5)*	—
MULS.W	Rm,Rn	0010nnnnmmmm1111	Signed operation of Rn × Rm → MACL 16 × 16 → 32 bits	—	1 (to 3)*	—
MULU.W	Rm,Rn	0010nnnnmmmm1110	Unsigned operation of Rn × Rm → MACL 16 × 16 → 32 bits	—	1 (to 3)*	—
NEG	Rm,Rn	0110nnnnmmmm1011	0–Rm→Rn	—	1	—
NEGC	Rm,Rn	0110nnnnmmmm1010	0–Rm–T→Rn, Borrow→T	—	1	Borrow
SUB	Rm,Rn	0011nnnnmmmm1000	Rn–Rm→Rn	—	1	—
SUBC	Rm,Rn	0011nnnnmmmm1010	Rn–Rm–T→Rn, Borrow →T	—	1	Borrow
SUBV	Rm,Rn	0011nnnnmmmm1011	Rn–Rm→Rn, Underflow→T	—	1	Underflow

Note: * The number of execution cycles indicated within the parentheses ( ) are required when the operation result is read from the MACH/MACL register immediately after the instruction.

Table 2.7 Logic Operation Instructions

Instruction		Instruction Code	Operation	Privileged Mode	Cycles	T Bit
AND	Rm,Rn	0010nnnnmmmm1001	Rn & Rm → Rn	—	1	—
AND	#imm,R0	11001001iiiiiiii	R0 & imm → R0	—	1	—
AND.B	#imm, @(R0, GBR)	11001101iiiiiiii	(R0+GBR) & imm → (R0+GBR)	—	3	—
NOT	Rm,Rn	0110nnnnmmmm0111	~Rm → Rn	—	1	—
OR	Rm,Rn	0010nnnnmmmm1011	Rn | Rm → Rn	—	1	—
OR	#imm,R0	11001011iiiiiiii	R0 | imm → R0	—	1	—
OR.B	#imm, @(R0, GBR)	11001111iiiiiiii	(R0+GBR) | imm → (R0+GBR)	—	3	—
TAS.B	@Rn	0100nnnn00011011	If (Rn) is 0, 1 → T; 1 → MSB of (Rn)	—	4	Test result
TST	Rm,Rn	0010nnnnmmmm1000	Rn & Rm; if the result is 0, 1 → T	—	1	Test result
TST	#imm,R0	11001000iiiiiiii	R0 & imm; if the result is 0, 1 → T	—	1	Test result
TST.B	#imm, @(R0, GBR)	11001100iiiiiiii	(R0 + GBR) & imm; if the result is 0, 1 → T	—	3	Test result
XOR	Rm,Rn	0010nnnnmmmm1010	Rn ^ Rm → Rn	—	1	—
XOR	#imm,R0	11001010iiiiiiii	R0 ^ imm → R0	—	1	—
XOR.B	#imm, @(R0, GBR)	11001110iiiiiiii	(R0+GBR) ^ imm → (R0+GBR)	—	3	—

Table 2.8 Shift Instructions

Instruction		Instruction Code	Operation	Privileged Mode	Cycles	T Bit
ROTL	Rn	0100nnnn00000100	T←Rn←MSB	—	1	MSB
ROTR	Rn	0100nnnn00000101	LSB→Rn→T	—	1	LSB
ROTCL	Rn	0100nnnn00100100	T←Rn←T	—	1	MSB
ROTCR	Rn	0100nnnn00100101	T→Rn→T	—	1	LSB
SHAD	Rm, Rn	0100nnnnmmmm1100	Rm ≥ 0: Rn << Rm → Rn Rm < 0: Rn >> Rm → [MSB → Rn]	—	1	—
SHAL	Rn	0100nnnn00100000	T←Rn←0	—	1	MSB
SHAR	Rn	0100nnnn00100001	MSB→Rn→T	—	1	LSB
SHLD	Rm, Rn	0100nnnnmmmm1101	Rm ≥ 0: Rn << Rm → Rn Rm < 0: Rn >> Rm → [0 → Rn]	—	1	—
SHLL	Rn	0100nnnn00000000	T←Rn←0	—	1	MSB
SHLR	Rn	0100nnnn00000001	0→Rn→T	—	1	LSB
SHLL2	Rn	0100nnnn00001000	Rn<<2 → Rn	—	1	—
SHLR2	Rn	0100nnnn00001001	Rn>>2 → Rn	—	1	—
SHLL8	Rn	0100nnnn00011000	Rn<<8 → Rn	—	1	—
SHLR8	Rn	0100nnnn00011001	Rn>>8 → Rn	—	1	—
SHLL16	Rn	0100nnnn00101000	Rn<<16 → Rn	—	1	—
SHLR16	Rn	0100nnnn00101001	Rn>>16 → Rn	—	1	—

Table 2.9 Branch Instructions

Instruction		Instruction Code	Operation	Privileged Mode	Cycles	T Bit
BF	disp	10001011dddddddd	If T = 0, branch to disp × 2 + PC if T = 1, nop	—	3/1*	—
BF/S	disp	10001111dddddddd	Delayed branch, if T = 0, branch to disp × 2 + PC if T = 1, nop	—	2/1*	—
BT	disp	10001001dddddddd	If T = 1, branch to disp × 2 + PC if T = 0, nop	—	3/1*	—
BT/S	disp	10001101dddddddd	Delayed branch, if T = 1, branch to disp × 2 + PC if T = 0, nop	—	2/1*	—
BRA	disp	1010dddddddddddd	Delayed branch, branch to disp × 2 + PC	—	2	—
BRAF	Rm	0000mmmm00100011	Delayed branch, branch to Rm + PC	—	2	—
BSR	disp	1011dddddddddddd	Delayed branch, reload next instruction address after delayed slot instruction → to PR, and branch to disp × 2 + PC	—	2	—
BSRF	Rm	0000mmmm00000011	Delayed branch, reload next instruction address after delayed slot instruction → to PR, and branch to Rm + PC	—	2	—
JMP	@Rm	0100mmmm00101011	Delayed branch, branch to Rm	—	2	—
JSR	@Rm	0100mmmm00001011	Delayed branch, reload next instruction address after delayed slot instruction → to PR, and branch to Rm	—	2	—
RTS		0000000000001011	Delayed branch, branch to PR	—	2	—

Note: * One cycle when the branch is not executed.

Table 2.10 System Control Instructions

Instruction		Instruction Code	Operation	Privileged Mode	Cycles	T Bit
CLRMAC		0000000000101000	0→MACH,MACL	—	1	—
CLRS		0000000001001000	0→S	—	1	—
CLRT		0000000000001000	0→T	—	1	0
LDC	Rm,SR	0100mmmm00001110	Rm→SR	√	6	LSB
LDC	Rm,GBR	0100mmmm00011110	Rm→GBR	—	4	—
LDC	Rm,VBR	0100mmmm00101110	Rm→VBR	√	4	—
LDC	Rm,SSR	0100mmmm00111110	Rm→SSR	√	4	—
LDC	Rm,SPC	0100mmmm01001110	Rm→SPC	√	4	—
LDC	Rm,R0_BANK	0100mmmm10001110	Rm→R0_BANK	√	4	—
LDC	Rm,R1_BANK	0100mmmm10011110	Rm→R1_BANK	√	4	—
LDC	Rm,R2_BANK	0100mmmm10101110	Rm→R2_BANK	√	4	—
LDC	Rm,R3_BANK	0100mmmm10111110	Rm→R3_BANK	√	4	—
LDC	Rm,R4_BANK	0100mmmm11001110	Rm→R4_BANK	√	4	—
LDC	Rm,R5_BANK	0100mmmm11011110	Rm→R5_BANK	√	4	—
LDC	Rm,R6_BANK	0100mmmm11101110	Rm→R6_BANK	√	4	—
LDC	Rm,R7_BANK	0100mmmm11111110	Rm→R7_BANK	√	4	—
LDC.L	@Rm+,SR	0100mmmm00000111	(Rm)→SR, Rm+4→Rm	√	8	LSB
LDC.L	@Rm+,GBR	0100mmmm00010111	(Rm)→GBR, Rm+4→Rm	—	4	—
LDC.L	@Rm+,VBR	0100mmmm00100111	(Rm)→VBR, Rm+4→Rm	√	4	—
LDC.L	@Rm+,SSR	0100mmmm00110111	(Rm)→SSR,Rm+4→Rm	√	4	—
LDC.L	@Rm+,SPC	0100mmmm01000111	(Rm)→SPC,Rm+4→Rm	√	4	—
LDC.L	@Rm+, R0_BANK	0100mmmm10000111	(Rm)→R0_BANK,Rm+4→Rm	√	4	—
LDC.L	@Rm+, R1_BANK	0100mmmm10010111	(Rm)→R1_BANK,Rm+4→Rm	√	4	—
LDC.L	@Rm+, R2_BANK	0100mmmm10100111	(Rm)→R2_BANK,Rm+4→Rm	√	4	—
LDC.L	@Rm+, R3_BANK	0100mmmm10110111	(Rm)→R3_BANK, Rm+4→Rm	√	4	—
LDC.L	@Rm+, R4_BANK	0100mmmm11000111	(Rm)→R4_BANK, Rm+4→Rm	√	4	—
LDC.L	@Rm+, R5_BANK	0100mmmm11010111	(Rm)→R5_BANK, Rm+4→Rm	√	4	—

Instruction		Instruction Code	Operation	Privileged Mode	Cycles	T Bit
LDC.L	@Rm+, R6_BANK	0100mmmm11100111	(Rm)→R6_BANK, Rm+4→Rm	√	4	—
LDC.L	@Rm+, R7_BANK	0100mmmm11110111	(Rm)→R7_BANK, Rm+4→Rm	√	4	—
LDS	Rm,MACH	0100mmmm00001010	Rm→MACH	—	1	—
LDS	Rm,MACL	0100mmmm00011010	Rm→MACL	—	1	—
LDS	Rm,PR	0100mmmm00101010	Rm→PR	—	1	—
LDS.L	@Rm+,MACH	0100mmmm00000110	(Rm)→MACH, Rm+4→Rm	—	1	—
LDS.L	@Rm+,MACL	0100mmmm00010110	(Rm)→MACL, Rm+4→Rm	—	1	—
LDS.L	@Rm+,PR	0100mmmm00100110	(Rm)→PR, Rm+4→Rm	—	1	—
NOP		0000000000001001	No operation	—	1	—
PREF	@Rm	0000mmmm10000011	(Rm) → cache	—	1	—
RTE		0000000000101011	Delayed branch, SSR → SR, branch to SPC	√	5	—
SETS		0000000001011000	1→S	—	1	—
SETT		0000000000011000	1→T	—	1	1
SLEEP		0000000000011011	Sleep	√	4*[1]	—
STC	SR,Rn	0000nnnn00000010	SR→Rn	√	1	—
STC	GBR,Rn	0000nnnn00010010	GBR→Rn	—	1	—
STC	VBR,Rn	0000nnnn00100010	VBR→Rn	√	1	—
STC	SSR, Rn	0000nnnn00110010	SSR→Rn	√	1	—
STC	SPC,Rn	0000nnnn01000010	SPC→Rn	√	1	—
STC	R0_BANK,Rn	0000nnnn10000010	R0_BANK→Rn	√	1	—
STC	R1_BANK,Rn	0000nnnn10010010	R1_BANK→Rn	√	1	—
STC	R2_BANK,Rn	0000nnnn10100010	R2_BANK→Rn	√	1	—
STC	R3_BANK,Rn	0000nnnn10110010	R3_BANK→Rn	√	1	—
STC	R4_BANK,Rn	0000nnnn11000010	R4_BANK→Rn	√	1	—
STC	R5_BANK,Rn	0000nnnn11010010	R5_BANK→Rn	√	1	—
STC	R6_BANK,Rn	0000nnnn11100010	R6_BANK→Rn	√	1	—
STC	R7_BANK,Rn	0000nnnn11110010	R7_BANK→Rn	√	1	—
STC.L	SR,@–Rn	0100nnnn00000011	Rn–4→Rn, SR→(Rn)	√	1	—
STC.L	GBR,@–Rn	0100nnnn00010011	Rn–4→Rn, GBR→(Rn)	—	1	—
STC.L	VBR,@–Rn	0100nnnn00100011	Rn–4→Rn, VBR→(Rn)	√	1	—

Instruction		Instruction Code	Operation	Privileged Mode	Cycles	T Bit
STC.L	SSR,@–Rn	0100nnnn00110011	Rn–4→Rn, SSR→(Rn)	√	1	—
STC.L	SPC,@–Rn	0100nnnn01000011	Rn–4→Rn, SPC→(Rn)	√	1	—
STC.L	R0_BANK,@–Rn	0100nnnn10000011	Rn–4→Rn, R0_BANK→(Rn)	√	1	—
STC.L	R1_BANK,@–Rn	0100nnnn10010011	Rn–4→Rn, R1_BANK→(Rn)	√	1	—
STC.L	R2_BANK,@–Rn	0100nnnn10100011	Rn–4→Rn, R2_BANK→(Rn)	√	1	—
STC.L	R3_BANK,@–Rn	0100nnnn10110011	Rn–4→Rn, R3_BANK→(Rn)	√	1	—
STC.L	R4_BANK,@–Rn	0100nnnn11000011	Rn–4→Rn, R4_BANK→(Rn)	√	1	—
STC.L	R5_BANK,@–Rn	0100nnnn11010011	Rn–4→Rn, R5_BANK→(Rn)	√	1	—
STC.L	R6_BANK,@–Rn	0100nnnn11100011	Rn–4→Rn, R6_BANK→(Rn)	√	1	—
STC.L	R7_BANK,@–Rn	0100nnnn11110011	Rn–4→Rn, R7_BANK→(Rn)	√	1	—
STS	MACH,Rn	0000nnnn00001010	MACH→Rn	—	1	—
STS	MACL,Rn	0000nnnn00011010	MACL→Rn	—	1	—
STS	PR,Rn	0000nnnn00101010	PR→Rn	—	1	—
STS.L	MACH,@–Rn	0100nnnn00000010	Rn–4→Rn, MACH→(Rn)	—	1	—
STS.L	MACL,@–Rn	0100nnnn00010010	Rn–4→Rn, MACL→(Rn)	—	1	—
STS.L	PR,@–Rn	0100nnnn00100010	Rn–4→Rn, PR→(Rn)	—	1	—
TRAPA	#imm	11000011iiiiiiii	Unconditional trap exception occurs*2	—	8	—

Notes: *1 Minimum number of cycles before the chip enters the sleep state.

*2 For details, refer to section 4, Exception Handling.

1. The table shows the minimum number of cycles required for execution. In practice, the number of execution cycles will be increased in the following conditions.
   a. If there is a conflict between an instruction fetch and a data access
   b. If the destination register of a load instruction (memory → register) is also used by the following instruction.
2. For addressing modes with displacement (disp) as shown below, the assembler description in this manual indicates the value before it is scaled (x 1, x 2, or x 4) according to the operand size to clarify the LSI operation. For details on assembler description, refer to the description rules in each assembler.
   @ (disp:4, Rn) ; Register indirect with displacement
   @ (disp:8, GBR) ; GBR indirect with displacement
   @ (disp:8, PC) ; PC relative with displacement
   disp:8, disp:12; PC relative

2.6.2 Operation Code Map

Table 2.11 shows the operation code map.

Table 2.11 Operation Code Map

Instruction Code				Fx: 0000	Fx: 0001	Fx: 0010	Fx: 0011 to 1111
MSB			LSB	MD: 00	MD: 01	MD: 10	MD: 11
0000	Rn	Fx	0000				
0000	Rn	Fx	0001				
0000	Rn	00MD	0010	STC SR, Rn	STC GBR, Rn	STC VBR, Rn	STC SSR, Rn
0000	Rn	01MD	0010	STC SPC, Rn			
0000	Rn	10MD	0010	STC R0_BANK, Rn	STC R1_BANK, Rn	STC R2_BANK, Rn	STC R3_BANK, Rn
0000	Rn	11MD	0010	STC R4_BANK, Rn	STC R5_BANK, Rn	STC R6_BANK, Rn	STC R7_BANK, Rn
0000	Rm	00MD	0011	BSRF Rm		BRAF Rm	
0000	Rm	10MD	0011	PREF @Rm			
0000	Rn	Rm	01MD	MOV.B Rm, @(R0, Rn)	MOV.W Rm, @(R0, Rn)	MOV.L Rm, @(R0, Rn)	MUL.L Rm, Rn
0000	0000	00MD	1000	CLRT	SETT	CLRMAC	LDTLB
0000	0000	01MD	1000	CLRS	SETS		
0000	0000	Fx	1001	NOP	DIV0U		
0000	0000	Fx	1010				
0000	0000	Fx	1011	RTS	SLEEP	RTE	
0000	Rn	Fx	1000				
0000	Rn	Fx	1001			MOVT Rn	
0000	Rn	Fx	1010	STS MACH, Rn	STS MACL, Rn	STS PR, Rn	
0000	Rn	Fx	1011				
0000	Rn	Rm	11MD	MOV.B @(R0, Rm), Rn	MOV.W @(R0, Rm), Rn	MOV.L @(R0, Rm), Rn	MAC.L @Rm+,@Rn+
0001	Rn	Rm	disp	MOV.L Rm, @(disp:4, Rn)			
0010	Rn	Rm	00MD	MOV.B Rm, @Rn	MOV.W Rm, @Rn	MOV.L Rm, @Rn	
0010	Rn	Rm	01MD	MOV.B Rm, @–Rn	MOV.W Rm, @–Rn	MOV.L Rm, @–Rn	DIV0S Rm, Rn
0010	Rn	Rm	10MD	TST Rm, Rn	AND Rm, Rn	XOR Rm, Rn	OR Rm, Rn
0010	Rn	Rm	11MD	CMP/STR Rm, Rn	XTRCT Rm, Rn	MULU.W Rm, Rn	MULSW Rm, Rn
0011	Rn	Rm	00MD	CMP/EQ Rm, Rn		CMP/HS Rm, Rn	CMP/GE Rm, Rn

Instruction Code				Fx: 0000		Fx: 0001		Fx: 0010		Fx: 0011 to 1111	
MSB			LSB	MD: 00		MD: 01		MD: 10		MD: 11	
0011	Rn	Rm	01MD	DIV1	Rm, Rn	DMULU.L	Rm,Rn	CMP/HI	Rm, Rn	CMP/GT	Rm, Rn
0011	Rn	Rm	10MD	SUB	Rm, Rn			SUBC	Rm, Rn	SUBV	Rm, Rn
0011	Rn	Rm	11MD	ADD	Rm, Rn	DMULS.L	Rm,Rn	ADDC	Rm, Rn	ADDV	Rm, Rn
0100	Rn	Fx	0000	SHLL	Rn	DT	Rn	SHAL	Rn		
0100	Rn	Fx	0001	SHLR	Rn	CMP/PZ	Rn	SHAR	Rn		
0100	Rn	Fx	0010	STS.L	MACH, @–Rn	STS.L	MACL, @–Rn	STS.L	PR, @–Rn		
0100	Rn	00MD	0011	STC.L	SR, @–Rn	STC.L	GBR, @–Rn	STC.L	VBR, @–Rn	STC.L	SSR, @–Rn
0100	Rn	01MD	0011	STC.L	SPC, @–Rn						
0100	Rn	10MD	0011	STC.L	R0_BANK, @–Rn	STC.L	R1_BANK, @–Rn	STC.L	R2_BANK, @–Rn	STC.L	R3_BANK, @–Rn
0100	Rn	11MD	0011	STC.L	R4_BANK, @–Rn	STC.L	R5_BANK, @–Rn	STC.L	R6_BANK, @–Rn	STC.L	R7_BANK, @–Rn
0100	Rn	Fx	0100	ROTL	Rn			ROTCL	Rn		
0100	Rn	Fx	0101	ROTR	Rn	CMP/PL	Rn	ROTCR	Rn		
0100	Rm	Fx	0110	LDS.L	@Rm+, MACH	LDS.L	@Rm+, MACL	LDS.L	@Rm+, PR		
0100	Rm	00MD	0111	LDC.L	@Rm+, SR	LDC.L	@Rm+, GBR	LDC.L	@Rm+, VBR	LDC.L	@Rm+, SSR
0100	Rm	01MD	0111	LDC.L	@Rm+, SPC						
0100	Rm	10MD	0111	LDC.L	@Rm+, R0_BANK	LDC.L	@Rm+, R1_BANK	LDC.L	@Rm+, R2_BANK	LDC.L	@Rm+, R3_BANK
0100	Rm	11MD	0111	LDC.L	@Rm+, R4_BANK	LDC.L	@Rm+, R5_BANK	LDC.L	@Rm+, R6_BANK	LDC.L	@Rm+, R7_BANK
0100	Rn	Fx	1000	SHLL2	Rn	SHLL8	Rn	SHLL16	Rn		
0100	Rn	Fx	1001	SHLR2	Rn	SHLR8	Rn	SHLR16	Rn		
0100	Rm	Fx	1010	LDS	Rm, MACH	LDS	Rm, MACL	LDS	Rm, PR		
0100	Rm/ Rn	Fx	1011	JSR	@Rm	TAS.B	@Rn	JMP	@Rm		

Instruction Code				Fx: 0000		Fx: 0001		Fx: 0010		Fx: 0011 to 1111	
MSB			LSB	MD: 00		MD: 01		MD: 10		MD: 11	
0100	Rn	Rm	1100	SHAD	Rm, Rn						
0100	Rn	Rm	1101	SHLD	Rm, Rn						
0100	Rm	00MD	1110	LDC	Rm, SR	LDC	Rm, GBR	LDC	Rm, VBR	LDC	Rm, SSR
0100	Rm	01MD	1110	LDC	Rm, SPC						
0100	Rm	10MD	1110	LDC	Rm, R0_BANK	LDC	Rm, R1_BANK	LDC	Rm, R2_BANK	LDC	Rm, R3_BANK
0100	Rm	11MD	1110	LDC	Rm, R4_BANK	LDC	Rm, R5_BANK	LDC	Rm, R6_BANK	LDC	Rm, R7_BANK
0100	Rn	Rm	1111	MAC.W	@Rm+, @Rn+						
0101	Rn	Rm	disp	MOV.L	@(disp:4, Rm), Rn						
0110	Rn	Rm	00MD	MOV.B	@Rm, Rn	MOV.W	@Rm, Rn	MOV.L	@Rm, Rn	MOV	Rm, Rn
0110	Rn	Rm	01MD	MOV.B	@Rm+, Rn	MOV.W	@Rm+, Rn	MOV.L	@Rm+, Rn	NOT	Rm, Rn
0110	Rn	Rm	10MD	SWAP.B	Rm, Rn	SWAP.W	Rm, Rn	NEGC	Rm, Rn	NEG	Rm, Rn
0110	Rn	Rm	11MD	EXTU.B	Rm, Rn	EXTU.W	Rm, Rn	EXTS.B	Rm, Rn	EXTS.W	Rm, Rn
0111	Rn	imm		ADD	# imm : 8, Rn						
1000	00MD	Rn	disp	MOV. B	R0, @(disp: 4, Rn)	MOV. W	R0, @(disp: 4, Rn)				
1000	01MD	Rm	disp	MOV.B	@(disp:4, Rm), R0	MOV.W	@(disp: 4, Rm), R0				
1000	10MD	imm/disp		CMP/EQ	#imm:8, R0	BT	disp: 8			BF	disp: 8
1000	11MD	imm/disp				BT/S	disp: 8			BF/S	disp: 8
1001	Rn	disp		MOV.W	@(disp: 8, PC), Rn						
1010	disp			BRA	disp: 12						
1011	disp			BSR	disp: 12						
1100	00MD	imm/disp		MOV.B	R0, @(disp: 8, GBR)	MOV.W	R0, @(disp: 8, GBR)	MOV.L	R0, @(disp: 8, GBR)	TRAPA	#imm: 8
1100	01MD	disp		MOV.B	@(disp: 8, GBR), R0	MOV.W	@(disp: 8, GBR), R0	MOV.L	@(disp: 8, GBR), R0	MOVA	@(disp: 8, PC), R0
1100	10MD	imm		TST	#imm: 8, R0	AND	#imm: 8, R0	XOR	#imm: 8, R0	OR	#imm: 8, R0
1100	11MD	imm		TST.B	#imm: 8, @(R0, GBR)	AND.B	#imm: 8, @(R0, GBR)	XOR.B	#imm: 8, @(R0, GBR)	OR.B	#imm: 8, @(R0, GBR)
1101	Rn	disp		MOV.L	@(disp: 8, PC), Rn						
1110	Rn	imm		MOV	#imm:8, Rn						
1111	**********										

Note: For details, refer to the SH-3/SH-3E/SH3-DSP Programming Manual.

Section 3 DSP Operating Unit

3.1 DSP Extended Functions

This LSI incorporates a DSP unit and XY memory directly connected to the DSP unit. This LSI supports the DSP extended function instruction sets needed to control the DSP unit and XY memory. The DSP extended function instructions are classified into four groups (figure 3.1).

Extended System Control Instructions for the CPU: If the DSP extended function is enabled, the following extended system control instructions can be used for the CPU.

- Repeat loop control instructions and repeat loop control register access instructions are added. Looped programs can be executed efficiently by using the zero-overhead repeat control unit. For details, refer to section 3.3, CPU Extended Instructions.
- Modulo addressing control instructions and control register access instructions are added. Function allows access to data with a circular structure. For details, refer to section 3.4, DSP Data Transfer Instructions.
- DSP unit register access instructions are added. Some of the DSP unit registers can be used in the same way as the CPU system registers. For details, refer to section 3.4, DSP Data Transfer Instructions.

Data Transfer Instructions for Data Transfers between DSP Unit and On-Chip XY Memory: Data transfer instructions for data transfers between the DSP unit and on-chip XY memory are called double-data transfer instructions. Instruction codes for these double-transfer instructions are 16-bit codes as well as CPU instruction codes. These data transfer instructions perform data transfers between the DSP unit and on-chip XY memory that is directly connected to the DSP unit. These data transfer instructions can be described in combination with other DSP unit operation instructions. For details, refer to section 3.4, DSP Data Transfer Instructions.

Data Transfer Instructions for Data Transfers between DSP Unit Registers and All Virtual Address Spaces: Data transfer instructions for data transfers between DSP unit registers and all virtual address spaces are called single-data transfer instructions. Instruction codes for the double-transfer instructions are 16-bit codes as well as CPU instruction codes. These data transfer instructions performs data transfers between the DSP unit registers and all virtual address spaces. For details, refer to section 3.4, DSP Data Transfer Instructions.

DSP Unit Operation Instructions: DSP unit operation instructions are called DSP operation instructions. These instructions are provided to execute digital signal processing operations at high speed using the DSP. Instruction codes for these instructions are 32 bits. The DSP operation instruction fields consist of two fields: field A and field B. In field A, a function for double data transfer instructions can be descried. In field B, ALU operation instructions and multiply instructions can be described. The instructions described in fields A and B can be executed in parallel. A maximum of four instructions (ALU operation, multiply, and two data

transfers) can be executed in parallel. For details, refer to section 3.5, DSP Data Operation Instructions.

Notes: 1. 32-bit instruction codes are handled as two consecutive 16-bit instruction codes. Accordingly, 32-bit instruction codes can be assigned to a word boundary. 32-bit instruction codes must be stored in memory, upper word and lower word, in this order, in word units.

2. In little endian, the upper and lower words must be stored in memory as data to be accessed in word units.

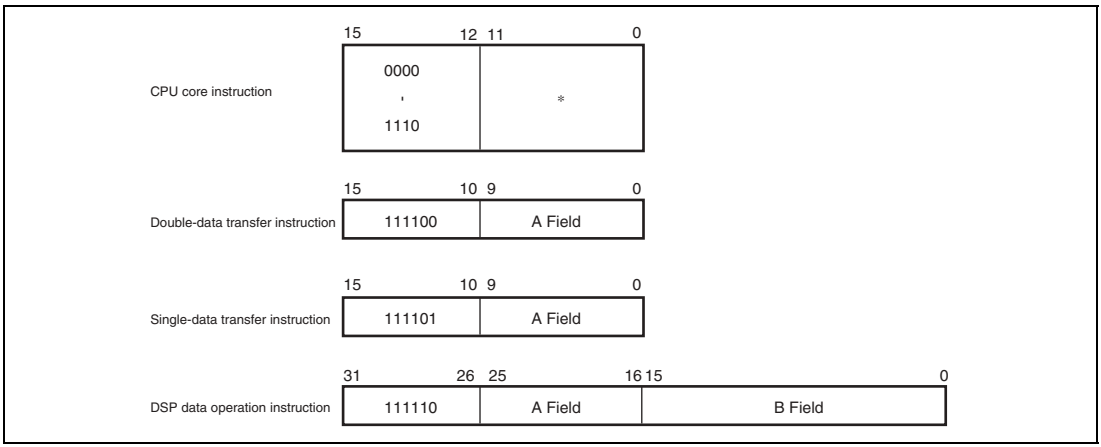


Figure 3.1 DSP Instruction Format

3.2 DSP Mode Resources

3.2.1 Processing Modes

The CPU processing modes can be extended using the mode bit (MD) and DSP bit (DSP) of the status register (SR), as shown below.

			Description	
MD Bit	DSP Bit	Processing Mode	Access of Resources Protected in Privileged Mode or Privileged Instruction Execution	DSP Extended Functions
0	0	User mode	Prohibited	Invalid
0	1	User DSP mode	Prohibited	Valid
1	0	Privileged mode	Allowed	Invalid
1	1	Privileged DSP mode	Allowed	Valid

As shown above, the extension of the DSP function by the DSP bit can be specified independently of the control by the MD bit. Note, however, that the DSP bit can be modified only in privileged mode. Before the DSP bit is modified, a transition to privileged mode or privileged DSP mode is necessary.

3.2.2 DSP Mode Memory Map

In DSP mode, a part of the P2 area in the virtual address space can be accessed in user DSP mode (H'A5000000 to H'A5FFFFFF of H'A0000000 to H'BFFFFFFF can be accessed) as shown in figure 3.1. When this area is accessed in user DSP mode, this area is referred to as a Uxy area. X/Y memory is then assigned to this Uxy area. Accordingly, X/Y memory can also be accessed in user DSP mode.

Table 3.1 Virtual Address Space

Address Range	Name	Protection	Description
H'A5000000 to H'A5FFFFFF	P2/Uxy	Privileged mode or DSP mode	16-Mbyte physical address space, non-cacheable Can be accessed in privileged mode and DSP modes (privileged DSP mode and user DSP mode)

3.2.3 CPU Register Sets

In DSP mode, the status register (SR) in the CPU unit is extended to add control bits and three control registers: a repeat start register (SR), repeat end register (RE), and modulo register are added as control registers (figure 3.2).

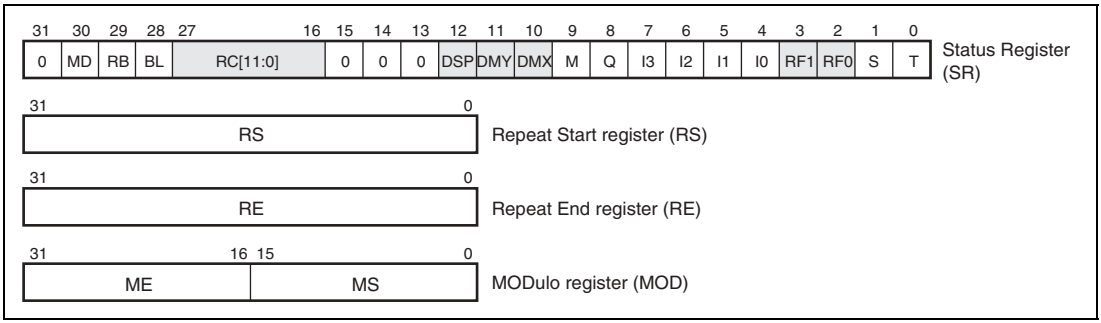


Figure 3.2 CPU Registers in DSP Mode

Extension of Status Register (SR): In DSP mode, the following control bits are added to the status register (SR). These added bits are called DSP extended bits. These DSP extended bits are valid only in DSP mode.

Bit	Bit Name	Initial Value	R/W	Description
31 to 28	—	—	—	For details, refer to section 2, CPU.
27 to 16	RC11 to RC0	All 0	R/W	Repeat Counter Holds the number of repeat times in order to perform loop control, and can be modified in privileged mode, privileged DSP mode, or user DSP mode. At reset, this bit is initialized to 0. This bit is not affected in the exception handling state.
15 to 13	—	—	—	For details, refer to section 2, CPU.
12	DSP	0	R/W	DSP Enables or disables the DSP extended functions. If this bit is set to 1, the DSP extended functions are enabled. This bit can be modified in privileged mode and privileged DSP mode; not in user DSP mode. At reset, this bit is initialized to 0. This bit is not affected in the exception handling state.
11	DMY	0	R/W	Modulo Control
10	DMX	0	R/W	Enable or disable modulo addressing for X/Y memory access. These bits can be modified in privileged mode, privileged DSP mode, or user DSP mode. At reset, these bits are initialized to 0. These bits are not affected in the exception handling state.
9 to 4	—	—	—	For details, refer to section 2, CPU.
3	FR1	0	R/W	Repeat Flag
2	FR0	0	R/W	Used by repeat control instructions. These bits can be modified in privileged mode, privileged DSP mode, or user DSP mode. At reset, these bits are initialized to 0. These bits are not affected in the exception handling state.
1 to 0	—	—	—	For details, refer to section 2, CPU.

Note: When data is written to the SR register, 0 should be written to reserved bits (bits 31, 15 to 13).

Repeat Start Register (RS): RS holds the start address of the repeat loop that is controlled by the zero overhead repeat control function. This register can be accessed in DSP mode. At reset, the initial value of this register is undefined. This register is not affected in the exception handling state.

Repeat End Register (RE): RE holds the address for detecting the execution of the end instruction of repeat loop that is controlled by the zero overhead repeat control function. This register can be accessed in DSP mode. At reset, this register is initialized to 0. This register is not affected in the exception handling state.

Modulo Register (MOD): MOD stores the modulo end address and modulo start address for modulo addressing in upper and lower 16 bits. The upper and lower 16 bits of the MOD register are referred to as the ME register and MS register, respectively. This register can be accessed in DSP mode. At reset, the initial value of this register is undefined. This register is not affected in the exception handling state.

The above registers can be accessed by the control register load instruction (LDC) and store instruction (STC). Note that the LDC and STC instructions for the RS, RE, and MOD registers can be used only in privileged DSP mode and user DSP mode. The LDC and STC instruction for the SR register can be executed only when the MD bit is set to 1 or in user DSP mode. Note, however, that the LDC and STC instructions can modify only the RC11 to RC0, RF1 to RF0, DMX, and DMY bits in the SR, as described below.

- In user mode, if the LCD and STC instructions are used for the SR, an illegal instruction exception occurs.
- In privileged and privileged DSP modes, all SR bits can be modified.
- In user DSP mode, the SR can be read by the STC instruction.

  In user DSP mode, the LDC instruction can be issued to the SR but only the DSP extension bits can be modified.

Table 3.2 Operation of SR Bits in Each Processing Mode

Bit	Privileged Mode MD = 1 & DSP = 0	User Mode MD = 0 & DSP = 0	Privileged DSP Mode MD = 1 & DSP = 1	User DSP Mode MD = 0 & DSP = 1	Access to DSP-Related Bit with Dedicated Instruction	Initial Value after Reset
MD	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	S: OK, L: NG		1
RB	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	S: OK, L: NG		1
BL	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	S: OK, L: NG		1
RC [11:0]	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	R: OK, L: OK	SETRC instruction	B'000000000000
DSP	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	S: OK, L: NG		0
DMY	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	R: OK, L: OK		0
DMX	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	R: OK, L: OK		0
Q	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	S: OK, L: NG		x
M	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	S: OK, L: NG		x
I[3:0]	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	S: OK, L: NG		B'1111
RF[1:0]	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	R: OK, L: OK	SETRC instruction	x
S	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	S: OK, L: NG		x
T	S: OK, L: OK	S, L: Illegal instruction	S: OK, L: OK	S: OK, L: NG		x

Legend

S: STC instruction

L: LDC instruction

OK: STC/LDC operation is enabled.

Invalid instruction: Exception occurs when an illegal instruction is executed.

NG: Previous value is retained. No change.

x: Undefined

Before entering the exception handling state, all bits including the DSP extended bits of the SR registers are saved in the SSR. Before returning from the exception handling, all bits including the DSP extended bits of the SR must be restored. If the repeat control must be recovered before entering the exception handling state, the RS and RE registers must be recovered to the value that existed before exception handling. In addition, if it is necessary to recover modulo control before entering the exception handling state, the MOD register must be recovered to the value that existed before exception handling.

3.2.4 DSP Registers

The DSP unit incorporates eight data registers (A0, A1, X0, X1, Y0, Y1, M0, and M1) and a status register (DSR). Figure 3.3 shows the DSP register configuration. These are 32-bit width registers with the exception of registers A0 and A1. Registers A0 and A1 include 8 guard bits (fields A0G and A1G), giving them a total width of 40 bits. The DSR register stores the DSP data operation result (zero, negative, others). The DSP register has a DC bit whose function is similar to the T bit of the CPU register. For details on DSR bits, refer to section 3.5, DSP Data Operation Instructions.

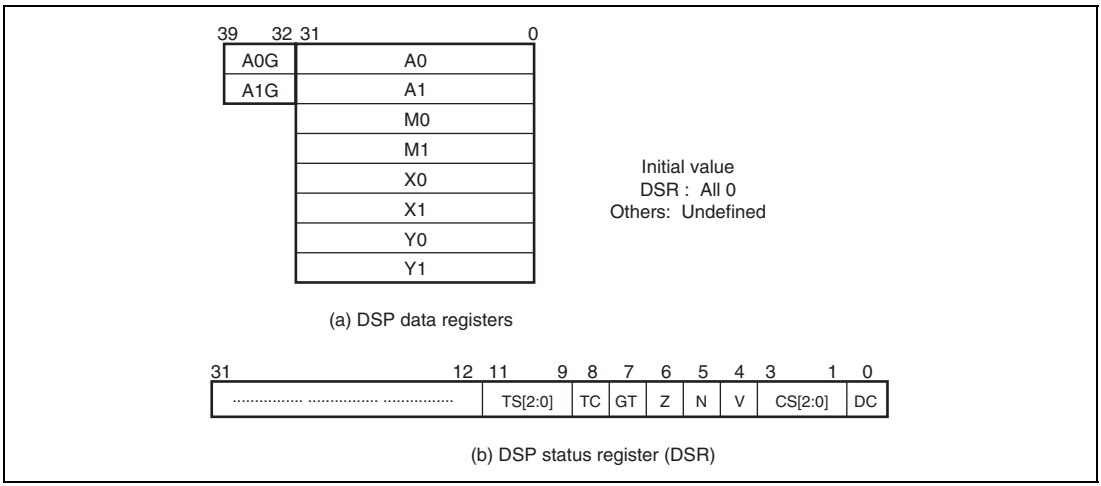


Figure 3.3 DSP Register Configuration

3.3 CPU Extended Instructions

3.3.1 DSP Repeat Control

In DSP mode, a specific function, zero overhead repeat control function, is provided to execute repeat loops efficiently. By using this function, loop programs can be executed without overhead caused by the compare and branch instructions. When describing the repeat loop, the repeat control instructions in table 3.4, the repeat control macro instructions in table 3.5, and the DSP mode extended system control instructions in table 3.6 are used.

Examples of Repeat Loop Programs: Examples of repeat loop programs are shown below.

- Example 1: Repeat loop consisting of 4 or more instructions

```
            LDRS RptStart  ; Sets repeat start instruction address
                             to the RS register
            LDRE RptStart +4      ; Sets (repeat detection instruction
                             address + 4) to the RE register
            SETRC #4       ; Sets the number of repetitions (4) to
                             the RC[11:0] bits of the SR register
            Instr0         ; At least one instruction is required
                             from SETRC instruction to [Repeat start
                             instruction]
RptStart:   instr1         ; [Repeat start instruction]
            ... ...        ;
            ... ...        ;
RptDtct:    instr(N-3)          ; [Repeat start instruction]
                                  Three instruction prior to the repeat
                                end instruction is regarded as repeat
                                  detection instruction
RptEnd2:    instr(N-2)     ;
RptEnd1:    instr(N-1)     ;
RptEnd:     instrN         ; [Repeat end instruction]
```

In the above program example, instructions from the RptStart address (instr1 instruction) to the RptEnd address (instrN instruction) are repeated four times. These repeated instructions in the program are called repeat loop. The start and end instructions of the repeat loop are called the repeat start instruction and repeat end instruction, respectively. The CPU sequentially executes instructions and starts repeat loop control if the CPU detects the completion of a specific instruction. This specific instruction is called the repeat detection instruction. In a repeat loop consisting of four or more instructions, an instruction three instructions prior to the repeat end instruction is regarded as the repeat detection instruction. In a repeat loop consisting of four or

more instructions, the same instruction is regarded as the RptStart instruction and RptDtct instruction.

To control the repeat loop, the DSP extended control registers, such as the RE register and RS register and the RC[11:0] and RF[1:0] bits of the SR register, are used. These registers can be specified by the LDRE, LDRS, and SETRC instructions.

- Repeat end register (RE)

  The RE register is specified by the LDRE instruction. The RE register specifies (repeat detection instruction address +4). In a repeat loop consisting of 4 or more instructions, an instruction three instructions prior to the repeat end instruction is regarded as the repeat detection instruction. A repeat loop consisting of three or less instructions is described later.
- Repeat start register (RS)

  The RE register is specified by the LDRS instruction. In a repeat loop consisting of 4 or more instructions, the RS register specifies the repeat start instruction address. In a repeat loop consisting of three or less instructions, a specific address is specified in the RS. This is described later.
- Repeat counter (RC[11:0] bits of the SR)

  The repeat counter is specifies the number of repetitions by the SETRC instruction. During repeat loop execution, the RC holds the remaining number of repetitions.
- Repeat flags (RF[1:0] bits of the SR)

  The repeat flags are automatically specified according to the RS and RE register values during SETRC instruction execution. The repeat flags store information on the number of instructions included in the repeat loop. Normally, the user cannot modify the repeat flag values.

The CPU always executes instructions by comparing the RE register to program counter values. Because the PC stores (the current instruction address +4), if the RE matches the PC during repeat instruction detection execution, a repeat detection instruction can be detected. If a repeat detection instruction is executed without branching and if RC[11:0] > 0, then repeat control is performed. If RC[11:0] ≥ 2 when the repeat end instruction is completed, the RC[11:0] is decremented by 1 and then control is passed to the address specified by the RS register.

Examples 2 to 4 show program examples of the repeat loop consisting of three instructions, two instructions, and one instruction, respectively. In these examples, an instruction immediately prior to the repeat start instruction is regarded as a repeat detection instruction. The RS register specifies the specific value that indicates the number of repeat instructions.

- Example 2: Repeat loop consisting of three instructions

```
        LDRS RptStart +4   ; Sets (repeat detection instruction
                           address + 4) to the RS register
        LDRE RptStart +4   ; Sets (repeat detection instruction
                           address + 4) to the RE register
        SETRC #4           ; Sets the number of repetitions (4) to
                             the RC[11:0] bits of the SR register
                           ; If RE-RS==0 during SETRC instruction
                           execution, the repeat loop is regarded
                             as three-instruction repeat.
RptDtct:  instr0           ; [Repeat start instruction]
                             An instruction prior to the Repeat
                           start instruction is regarded as a
                           repeat detection instruction.
                             At least one instruction is required
                             from LDRC instruction to [Repeat start
                             instruction]
RptStart: instr1           ; [Repeat start instruction]
          Instr2           ;
RptEnd:   instr3           ; [Repeat end instruction]
```

- Example 3: Repeat loop consisting of two instructions

```
          LDRS RptStart +6      ; Sets (repeat detection instruction
                                  address + 6) to the RS register
          LDRE RptStart +4      ; Sets (repeat detection instruction
                                  address + 4) to the RE register
          SETRC #4         ; Sets the number of repetitions (4) to
                             the RC[11:0] bits of the SR register
                           ; If RE-RS==-2 during SETRC instruction
                         execution, the repeat loop is regarded
                             as two-instruction repeat.
RptDtct:  instr0           ; [Repeat start instruction]
                             An instruction prior to the Repeat
                           start instruction is regarded as a
                           repeat detection instruction.
                             At least one instruction is required
                             from LDRC instruction to [Repeat start
                             instruction]
RptStart: instr1           ; [Repeat start instruction]
RptEnd:   instr2           ; [Repeat end instruction]
```

- Example 4: Repeat loop consisting of one instruction

```
            LDRS RptStart +8      ; Sets (repeat detection instruction
                                    address + 8) to the RS register
            LDRE RptStart +4      ; Sets (repeat detection instruction
                                    address + 4) to the RE register
            SETRC #4             ; Sets the number of repetitions (4) to
                                    the RC[11:0] bits of the SR register
                                 ; If RE-RS==-4 during SETRC instruction
                               execution, the repeat loop is regarded
                                    as one-instruction repeat.
 RptDtct:   instr0               ; [Repeat start instruction]
                                    An instruction prior to the Repeat
                                 start instruction is regarded as a
                                 repeat detection instruction.
                                    At least one instruction is required
                                    from LDRC instruction to [Repeat start
                                    instruction]
 RptStart:
 RptEnd:    instr1               ; [Repeat start instruction] ==
                                 [Repeat end instruction]
```

In repeat loops consisting of three instructions, two instructions and one instruction, specific addresses are specified in the RS register. RE – RS is calculated during SETRC instruction execution, and the number of instructions included in the repeat loop is determined according to the result. A value of 0, –2,and –4 in the result correspond to 3 instructions, two instructions, and one instruction, respectively.

If repeat instruction execution is completed without branching and if RC[11:0] > 0, an instruction following the repeat detection instruction is regarded as a repeat start instruction and instruction execution is repeated for the number of times corresponding to the recognized number of instructions. If RC[11:0] ≥ 2 when the repeat end instruction is completed, the RC[11:0] is decremented by 1 and then control is passed to the address specified by the RS register. If RC[11:0] ==1(or 0) when the repeat end instruction is completed, the RC[11:0] is cleared to 0 and then the control is passed to the next instruction following the repeat end instruction.

Note: If RE – RS is a positive value, the CPU regards the repeat loop as a four-instruction repeat loop. (In a repeat loop consisting of four or more instructions, RE – RS is always a positive value. For details, refer to example 1 above.) If RE – RS is positive, or a value other than 0, –2,and –4, correct operation cannot be guaranteed.

The rule is shown in table 3.3.

Table 3.3 RS and RE Setting Rule

	Number of Instructions in Repeat Loop			
Register	**1**	**2**	**3**	**≥4**
RS	RptStart0 + 8	RptStart0 + 6	RptStart0 + 4	RptStart
RE	RptStart0 + 4	RptStart0 + 4	RptStart0 + 4	RptEnd3 + 4

Note: The terms used above in table 3.3, are defined as follows.
RptStart: Address of the repeat start instruction
RptStart0: Address of the instruction one instruction prior to the repeat start instruction
RptEnd3: Address of the instruction three instructions prior to the repeat end instruction

Repeat Control Instructions and Repeat Control Macros: To describe a repeat loop, the RS and RE registers must be specified appropriately by the LDRS and LDRE instructions and then the number of repetitions must be specified by the SERTC instruction. An 8-bit immediate data or a general register can be used as an operand of the SETRC instruction. To specify the RC as a value greater than 256, use SETRC Rm type instructions.

Table 3.4 Repeat Control Instructions

Instruction	Operation	Number of Execution States
LDRS @(disp,PC)	Calculates (disp x 2 + PC) and stores the result to the RS register	1
LDRE @(disp,PC)	Calculates (disp x 2 + PC) and stores the result to the RE register	1
SETRC #imm	Sets 8-bit immediate data imm to the RC[11:0] bits of the SR register and sets the information related to the number of repetitions to the RF[1:0] bits of the SR. RC[11:0] can be specified as 0 to 255.	1
SETRC Rm	Sets the[11:0] bits of the Rm register to the RC[11:0] bits of the SR register and sets the information related to the number of repetitions to the RF[1:0] bits of the SR. RC[11:0] can be specified as 0 to 4095.	1

The RS and RE registers must be specified appropriately according to the rules shown in table 3.3. The SH assembler supports control macros (REPEAT) as shown in table 3.5 to solve problems.

Table 3.5 Repeat Control Macros

Instruction	Operation	Number of Execution States
REPEAT RptStart, RptEnd, #imm	Specifies RptStart as repeat start instruction, RptEnd as repeat end instruction, and 8-bit immediate data #imm as number of repetitions. This macro is extended to three instructions: LDRS, LDRE, and SETRC which are converted correctly.	3
REPEAT RptStart, RptEnd, Rm	Specifies RptStart as repeat start instruction, RptEnd as repeat end instruction, and the [11:0] bits of Rm as number of repetitions. This macro is extended to three instructions: LDRS, LDRE, and SETRC which are converted correctly.	3

Using the repeat macros shown in table 3.5, examples 1 to 4 shown above can be simplified to examples 5 to 8 as shown below.

- Example 5: Repeat loop consisting of 4 or more instructions (extended to the instruction stream shown in example 1, above)

```
          REPEAT RptStart, RptEnd, #4
          Instr0          ;
RptStart: instr1          ; [Repeat start instruction]
          ... ...         ;
          ... ...         ;
          instr (N-3)     ; [Repeat detection instruction]
          instr (N-2)     ;
          instr (N-1)     ;
RptEnd:   instrN          ; [Repeat end instruction]
```

- Example 6: Repeat loop consisting of three instructions (extended to the instruction stream shown in example 2, above)

```
          REPEAT RptStart, RptEnd, #4
          instr0          ; [Repeat detection instruction]
RptStart: instr1          ; [Repeat start instruction]
          Instr2          ;
RptEnd:   instr3          ; [Repeat end instruction]
```

- Example 7: Repeat loop consisting of two instructions (extended to the instruction stream shown in example 3, above)

```
          REPEAT RptStart, RptEnd, #4
          instr0          ; [Repeat detection instruction]
RptStart: instr1          ; [Repeat start instruction]
RptEnd:   instr2          ; [Repeat end instruction]
```

- Example 8: Repeat loop consisting of one instruction instructions (extended to the instruction stream shown in example 3, above)

```
          REPEAT RptStart, RptEnd, #4
          instr0          ; [Repeat detection instruction]
RptStart:
RptEnd:   instr1          ; [Repeat start instruction] ==
                            [Repeat end instruction]
```

In the DSP mode, the system control instructions (LDC and STC) that handle the RS and RE registers are extended. The RC[11:0] bits and RF[1:0] bits of the SR can be controlled by the LDC and STC instructions for the SR register. These instructions should be used if an exception is enabled during repeat loop execution. The repeat loop can be resumed correctly by storing the RS and RE register values and RC[11:0] bits and RF[1:0] bits of the SR register before exception handling and by restoring the stored values after exception handling. However, note that there are some restrictions on exception acceptance during repeat loop execution. For details refer to section 3.3.1 (3), Restrictions on Repeat Loop Control and section 4, Exception Handling.

Table 3.6 DSP Mode Extended System Control Instructions

Instruction	Operation	Number of Execution States
STC RS, Rn	RS→Rn	1
STC RE, Rn	RE→Rn	1
STC.L RS, @-Rn	Rn-4→Rn, RS→(Rn)	1
STC.L RE, @-Rn	Rn-4→Rn, RE→(Rn)	1
LDC.L @Rn+, RS	(Rn)→RS, Rn+4→Rn	4
LDC.L @Rn+, RE	(Rn)→RE, Rn+4→Rn	4
LDC Rn,RS	Rn→RS	4
LDC Rn, RE	Rn→RE	4

Restrictions on Repeat Loop Control

1. Repeat control instruction assignment

   The SETRC instruction must be executed after executing the LDRS and LDRE instructions. In addition, note that at least one instruction is required between the SETRC instruction and a repeat start instruction.

2. Illegal instruction one or more instructions following the repeat detection instruction

   If one of the following instructions is executed between an instruction following a repeat detection instruction to a repeat end instruction, an illegal instruction exception occurs.

   — Branch instructions

     BRA, BSR, BT, BF, BT/S, BF/S, BSRF, RTS, BRAF, RTE, JSR, JMP, TRAPA

   — Repeat control instructions

     SETRC, LDRS, LDRE

   — Load instructions for SR, RS, and RE registers

     LDC Rn,SR, LDC @Rn+,SR, LDC Rn,RE, LDC @Rn+,RE, LDC Rn,RS, LDC @Rn+,RS

Note: This restriction applies to all instructions for a repeat loop consisting of one to three instructions and to three instructions including a repeat end instruction.

3. Instructions prohibited during repeat loop (In a repeat loop consisting of four or more instructions)

   The following instructions must not be placed between the repeat start instruction and repeat detection instruction in a repeat loop consisting of four or more instructions. Otherwise, the correct operation cannot be guaranteed.

   — Repeat control instructions

     SETRC, LDRS, LDRE

   — Load instructions for SR, RS, and RE registers

     LDC Rn,SR, LDC @Rn+,SR, LDC Rn,RE, LDC @Rn+,RE, LDC Rn,RS, LDC @Rn+,RS

Note: Multiple repeat loops cannot be guaranteed. Describe the inner loop by repeat control instructions, and the external loop by other instructions such as DT or BF/S.

4. Branching to an instruction following the repeat detection instruction and restriction on an exception acceptance

   Execution of a repeat detection instruction must be completed without any branch so that the CPU can recognize the repeat loop. Therefore, when the execution branches to an instruction following the repeat detection instruction, the control will not be passed to a repeat start instruction after executing a repeat end instruction because the repeat loop is not recognized by the CPU. In this case, the RC[11:0] bits of the SR register will not be changed.

   — If a conditional branch instruction is used in the repeat loop, an instruction before a repeat detection instruction must be specified as a branch destination.

— If a subroutine call is used in the repeat loop, a delayed slot instruction of the subroutine call instruction must be placed before a repeat detection instruction.

Here, a branch includes a return from an exception handling routine. If an exception whose return address is placed in an instruction following the repeat detection instruction occurs, the repeat control cannot be returned correctly. Accordingly, an exception acceptance is restricted from the repeat detection instruction to the repeat end instruction. Exceptions such as interrupts that can be retained by the CPU are retained. For exceptions that cannot be retained by the CPU, a transition to an exception occurs but a program cannot be returned to the previous execution state correctly. For details, refer to section 4, Exception Handling.

Notes: 1. If a TRAPA instruction is used as a repeat detection instruction, an instruction following the repeat detection instruction is regarded as a return address. In this case, a control cannot be returned to the repeat control correctly. In a TRAPA instruction, an address of an instruction following the repeat detection address is regarded as return address. Accordingly, to return to the repeat control correctly, place a return address prior to the repeat detection instruction.

2. If a SLEEP instruction is placed following a repeat detection instruction, a transition to the power-down state or an exception acceptance such as interrupts can be performed correctly. In this case, however, the repeat control cannot be returned correctly. To return to the repeat control correctly, the SLEEP instruction must be placed prior to the repeat detection instruction.

5. Branch from a repeat detection instruction

   If a repeat detection instruction is a delayed slot instruction of a delayed branch instruction or a branch instruction, a repeat loop can be acknowledged when a branch does not occur in a branch instruction. If a branch occurs in a branch instruction, a repeat control is not performed and a branch destination instruction is executed.

6. Program counter during repeat control

   If RC[11:0] ≥ 2, the program counter (PC) value is not correct for instructions two instructions following a repeat detection instruction. In a repeat loop consisting of one to three instructions, the PC indicates the correct value (instruction address + 4) for an instruction (repeat start instruction) following a repeat detect ion instruction but the PC continues to indicate the same address (repeat start instruction address) from the subsequent instruction to a repeat end instruction. In a repeat loop consisting of four or more instructions, the PC indicates the correct value (instruction address + 4) for an instruction following a repeat detect ion instruction, but PC indicates the RS and (RS +2) for instructions two and three instructions following the repeat detection instruction. The correct operation cannot be guaranteed for the incorrect PC values.

   Accordingly, PC relative addressing instructions placed two or more instructions following the repeat detection instruction cannot be executed correctly and the correct results cannot be obtained.

— PC relative addressing instructions

MOV.A @(disp, PC), Rn

MOV.W @(disp, PC), Rn

MOV.L @(disp, PC), Rn

(Including the case when the MOV #imm,Rn is extended to MOV.W @(disp, PC), Rn or MOV.L @(disp, PC), Rn)

Table 3.7 PC Value during Repeat Control (When RC[11:0] ≥ 2)

	Number of Instructions in Repeat Loop			
	1	2	3	≥4
RptDtct	RptDtct + 4	RptDtct + 4	RptDtct + 4	RptDtct +4
RptDtct1	RptDtct1 + 4	RptDtct1+ 4	RptDtct1 + 4	RptDtct1 + 4
RptDtct2	—	RptDtct1+ 4	RptDtct1 + 4	RS
RptDtct3	—	—	RptDtct1 + 4	RS + 2

Note: In table 3.7, the following symbols are used.

RptDtct: An address of the repeat detection instruction

RptDtct1: An address of the instruction one instruction following the repeat start instruction (In a repeat loop consisting of one to three instructions, RptStart is a repeat start instruction)

RptDtct2: An address of the instruction two instruction following the repeat start instruction

RptDtct3: An address of the instruction three instruction following the repeat start instruction

7. Repeat counter and repeat control

The CPU always executes a program with comparing the repeat end register (RE) and the program counter (PC). If the PC matches the RE while the RC[11:0] bits of the SR register are other than 0, the repeat control function is initiated.

— If RC ≥ 2, a control is passed to a repeat start instruction after a repeat end instruction has been executed. The RC is decremented by 1 at the completion of the repeat end instruction. In this case, restrictions (a) to (f) are also applied.

— If RC == 1, the RC is decremented to 0 at the completion of the repeat end instruction and a control is passed to the subsequent instruction. In this case, restrictions (a) to (f) are also applied.

— If RC == 0, the repeat control function is not initiated even if a repeat detection instruction is executed. The repeat loop is executed once as normal instructions and a control is not be passed to a repeat start instruction even if a repeat end instruction is executed.

3.3.2 Extended Repeat Control Instructions

In the repeat control function described in section 3.3.1, Repeat Control Instructions, there are some restrictions. To reduce these restrictions, this LSI supports the extended repeat instructions to extend the repeat control function that are shown in table 3.8. These extended repeat control instructions were not supported in the conventional SH-DSP. To keep compatibility with the conventional SH-DSP, use the conventional repeat control instructions called compatible repeat control instructions.

Program Examples Using the Extended Repeat Control Instructions: Examples of repeat loop programs using the extended repeat control instructions are shown below.

- Example 1: Repeat loop consisting of 4 or more instructions

```
        LDRS RptStart     ; Sets repeat start instruction address
                            to the RS register
        LDRE RptEnd       ; Sets repeat end instruction address
                          to the RE register
        LDRC #4           ; Sets the number of repetitions (4) to
                            the RC[11:0] bits of the SR register
        instr0            ; At least one instruction is required
                            from LDRC instruction to [Repeat start
                            instruction]
RptStart: instr1          ; [Repeat start instruction]
        ... ...           ;
        ... ...           ;
        instr(N-3)        ; [Repeat detection instruction]
                            An instruction three instructions prior to the
                          repeat end instruction is regarded as a
                            repeat detection instruction.
        instr(N-2)    ;
        instr(N-1)    ;
RptEnd:   instrN          ; [Repeat end instruction]
```

- Example 2: Repeat loop consisting of three instructions

```
        LDRS RptStart     ; Sets repeat start instruction address
                            to the RS register
        LDRE RptEnd       ; Sets repeat end instruction address
                          to the RE register
        LDRC #4           ; Sets the number of repetitions (4) to
                            the RC[11:0] bits of the SR register
        instr0            ; [Repeat detection instruction]
                            An instruction prior to the Repeat start
                            instruction is regarded as a repeat detection
                            instruction.
                            At least one instruction is required
                            from LDRC instruction to [Repeat start
                            instruction]
RptStart: instr1          ; [Repeat start instruction]
          instr2          ;
RptEnd:   instr3          ; [Repeat end instruction]
```

- Example 3: Repeat loop consisting of two instructions

```
        LDRS RptStart     ; Sets repeat start instruction address
                            to the RS register
        LDRE RptEnd       ; Sets repeat end instruction address
                        to the RE register
        LDRC #4           ; Sets the number of repetitions (4) to
                            the RC[11:0] bits of the SR register
        instr0            ; [Repeat detection instruction]
                            An instruction prior to the Repeat
                            start instruction is regarded as a repeat
                            detection instruction.
                            At least one instruction is required
                            from LDRC instruction to [Repeat start
                            instruction]
RptStart: instr1          ; [Repeat start instruction]
RptEnd:   instr2          ; [Repeat end instruction]
```

- Example 4: Repeat loop consisting of two instructions

```
            LDRS RptStart  ; Sets repeat start instruction address
                             to the RS register
            LDRE RptEnd    ; Sets repeat end instruction address
                           to the RE register
            LDRC #4        ; Sets the number of repetitions (4) to
                             the RC[11:0] bits of the SR register
            instr0         ; [Repeat detection instruction]
                             An instruction prior to the Repeat
                                  start instruction is regarded as a
                                  repeat detection instruction. At least
                                  one instruction is required from LDRC
                                  instruction to [Repeat start
                                  instruction]
  RptStart:
  RptEnd:   instr1       ; [Repeat start instruction]==
                           [Repeat end instruction]
```

In extended repeat control instructions, a repeat start instruction address and a repeat end instruction address are stored in the RS register and RE register, respectively, regardless of the number of repeat instructions. In addition, the extended repeat control can be performed by using the LDRC instruction instead of the SETRC instruction. During the extended repeat control, a repeat loop can be recognized by executing a repeat end instruction. Therefore, there is no restriction on branches or exceptions.

Extended Repeat Control Instructions: To describe the extended repeat loop, the repeat start and end addresses must be specified to the RS and RE registers by the LDRS and LDRE instructions, respectively. For the LDRS and LDRE instructions of the extended repeat control instructions, the LDRS and LDRE instructions of the compatible repeat control instructions are used. The number of repetitions are specified by the LDRC instruction. An 8-bit immediate data or the general register values can be used as an operand of the LDRC instruction. If 256 or greater value is specified to the RC, use the LDRC Rm type instructions.

Table 3.8 Extended Repeat Control Instructions

Instruction	Operation	Number of Execution States
LDRS @(disp,PC)	Calculates (disp × 2 + PC) and stores the result to the RS register	1
LDRE @(disp,PC)	Calculates (disp × 2 + PC) and stores the result to the RE register	1
LDRC #imm	Sets 8-bit immediate data imm to the RC[11:0] bits of the SR register and sets the information related to the number of repetitions to the RF[1:0] bits of the SR. RC[11:0] can be specified as 0 to 255. During extended repeat control, bit 0 of the RE register is set to 1.	1
LDRC Rm	Sets the[11:0] bits of the Rm register to the RC[11:0] bits of the SR register and sets the information related to the number of repetitions to the RF[1:0] bits of the SR. RC[11:0] can be specified as 0 to 4095. During extended repeat control, bit 0 of the RE register is set to 1.	1

By executing the LDRC instruction, the CPU performs the extended repeat control function. To indicate that the CPU is being in extended repeat control, bit 0 of the RE register is set to 1 by executing the LDRC instruction. To change the RE register value by a process such as an exception handling, bit 0 of the RE register must be saved and restored correctly. By saving and restoring the RC[11:0] bits, DSP bit, and RF[1:0] bits of the SR register, RE register, and RS register correctly, a control is returned to the extended repeat function correctly after processing such as exception handling.

Restrictions on Extended Repeat Loop Control:

1. Extended repeat control instruction assignment

   The LDRC instruction must be executed after executing the LDRS and LDRE instructions. In addition, note that at least one instruction is required between the LDRC instruction and a repeat start instruction.

2. Illegal instruction one or more instructions following the repeat end instruction

   If one of the following instructions is executed as a repeat end instruction, an illegal instruction exception occurs.

   — Delayed branch instructions

   BRA, BSR, BT/S, BF/S, BSRF, RTS, BRAF, RTE, JSR, JMP

   — Repeat control instructions

   SETRC, LDRS, LDRE, LDRC

   — Load instructions for SR, RS, and RE registers

   LCD Rn,SR, LDC @Rn+,SR, LDC Rn,RE, LDC @Rn+,RE, LDC Rn,RS, LDC @Rn+,RS

Note: A branch instruction without delay (BT, BF, TRAPA) can be placed as a repeat end instruction. A delay stop of a delayed branch instruction can also be placed as a repeat end instruction. In this case, the RC[11:0] value is decremented by 1 regardless of branch occurrence. If no branch occurs, a control returns to a repeat start instruction. If a branch occurs, a control is passed to a branch destination.

3. Repeat counter and repeat control

   The CPU always execute a program with comparing the repeat end register (RE) and the (PC – 4) (current instruction address). If the (PC – 4) [31:1] matches the RE [31:1] while bit 0 of RE register is set to 1, the extended repeat control function is initiated.

   — If RC ≥ 2, a control is passed to a repeat start instruction after a repeat end instruction has been executed. The RC is decremented by 1 at the completion of the repeat end instruction.

   — If RC == 1, the RC is decremented to 0 at the completion of the repeat end instruction and a control is passed to the subsequent instruction.

   — If RC == 0, the repeat control function is not initiated even if a repeat detection instruction is executed. The repeat loop is executed once as normal instructions and a control is not be passed to a repeat start instruction even if a repeat end instruction is executed.

3.4 DSP Data Transfer Instructions

In DSP mode, data transfer instructions are added for the DSP unit registers. The newly added instructions are classified into the following three groups.

1. Double data transfer instructions

   The DSP unit is connected to the X memory and Y memory via the specific buses called X bus and Y bus. By using the data transfer instructions using the X and Y buses, two data items can be transferred between the DSP unit and X/Y memories simultaneously. These instructions are called double data transfer instructions. These double data transfer instructions can be described in combination with the DSP operation instructions to execute data transfer and data operation in parallel.
2. Single data transfer instructions

   The DSP unit is also connected to the L bus that is used by the CPU. The DSP registers other than the DSR can access any virtual addresses generated by the CPU. In this case, the single data transfer instructions are used. The single data transfer instructions cannot be used in combination with the DSP operation instructions and can access only one data item at a time.
3. System control instructions

   Some of the DSP unit registers are handled as the CPU system registers. To control these system registers, the system control registers are supported. The DSP registers are connected to the CPU general registers via the data transfer bus (C bus).

In any DSP data transfer instructions, an address to be accessed is generated and output by the CPU. For DSP data transfer instructions, some of the CPU general registers are used for address generation and specific addressing modes are used.

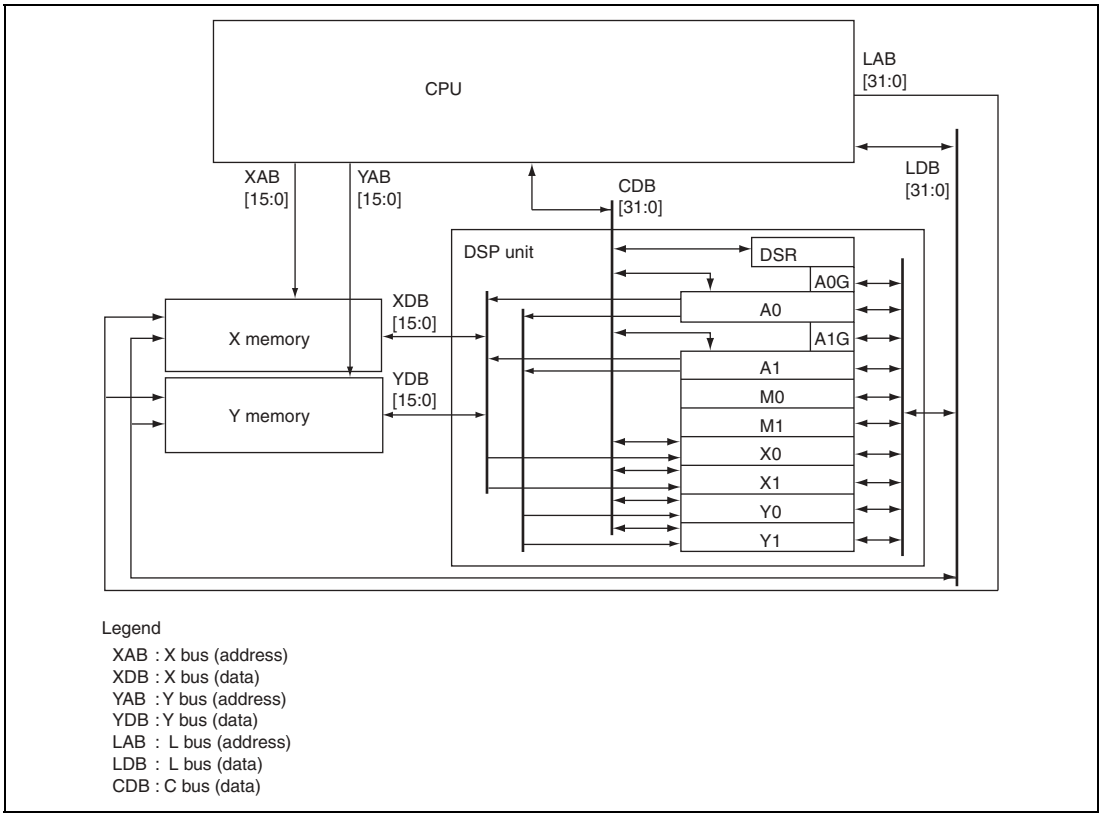


Figure 3.4 DSP Registers and Bus Connections

Double data transfer instructions (MOVX.W, MOVY.W, MOVX.L, and MOVY.L): Instruction formats of double data transfer instructions are shown in table 3.12. With double data transfer group instructions, X memory and Y memory can be accessed in parallel.

In this case, the specific buses called X bus and Y bus are used to access X memory and Y memory, respectively. To fetch the CPU instructions, the L bus is used. Accordingly, no conflict occurs among X, Y, and L buses.

Load instructions for X memory specify the X0 or X1 register as the destination operand. Load instructions for Y memory specify the Y0 or Y1 register as the destination operand. Store registers for X or Y memory specify the A0 or A1 register as the source operand. The double data transfer instructions use only word data (16 bits). When a word data transfer instruction is executed, the upper word of register operand is used. To load word data, data is loaded to the upper word of the destination register and the lower word of the destination register is automatically cleared to 0.

Double data transfer instructions can be described in parallel to the DSP operation instructions. Even if a conditional operation instruction is specified in parallel to a double data transfer

instruction, the specified condition does not affect the data transfer operations. For details, refer to section 3.5, DSP Data Operation Instructions.

Double data transfer instructions can access only the X memory or Y memory and cannot access other memory space. The X bus and Y bus are 16 bits and support 64-kbyte address spaces corresponding to address areas H'A5000000 to H'A500FFFF and H'A5010000 to H'A501FFFF, respectively. Because these areas are included in the P2/Uxy area, they are not affected by the cache and address translation unit.

Single data transfer instructions: The instruction formats of single data transfer instructions are shown in table 3.13. The single data transfer instructions access any memory location. All DSP registers* other than the DSR can be specified as source and destination operands. Guard bit registers A0G and A1G can also be specified as two independent registers. Because the single data transfer instructions use the L bus (LAB and LDB), these instructions can access any virtual space handled by the CPU. If these instructions access the cacheable area while the cache is enabled, the area accessed by these instructions are cached. The X memory and Y memory are mapped to the virtual address space and can also be accessed by the single data transfer instructions. In this case, bus conflict may occur between data transfer and instruction fetch because the CPU also uses the L bus for instruction fetches.

The single data transfer instructions can handle both word and longword data. In word data transfer, only the upper word of the operand register is valid. In word data load, word data is loaded into the upper word of the destination registers and the lower word of the destination is automatically cleared to 0. If the guard bits are supported, the sign bit is extended before storage. In longword data load, longword data is loaded into the upper and lower word of the destination register. If the guard bits are supported, the sign bit is extended before storage. When the guard register is stored, the sign bit is extended to the upper 24 bits of the LDB and are loaded onto the LDB bus.

Notes: * Because the DSR register is defined as the system register, it can be accessed by the LDS or STS instruction.

1. Any data transfer instruction is executed at the MA stage of the pipeline.
2. Any data transfer instruction does not modify the condition code bits of the DSR register.

System control instructions: The DSR, A0, X0, X1, Y0, and Y1 registers in the DSP unit can also be used as the CPU system registers. Accordingly, data transfer operations between these DSP system registers and general registers or memory can be executed by the STS and LDS instructions. These DSP system registers can be treated as the CPU system register such as PR, MACL and MACH and can use the same addressing modes.

Table 3.9 Extended System Control Instructions in DSP Mode

Instruction		Operation	Execution States
STS	DSR,Rn	DSR → Rn	1
STS	A0,Rn	A0 → Rn	1
STS	X0,Rn	X0 → Rn	1
STS	X1,Rn	X1 → Rn	1
STS	Y0,Rn	Y0 → Rn	1
STS	Y1,Rn	Y1 → Rn	1
STS.L	DSR,@-Rn	Rn – 4 → Rn, DSR → (Rn)	1
STS.L	A0,@-Rn	Rn – 4 → Rn, A0 → (Rn)	1
STS.L	X0,@-Rn	Rn – 4 → Rn, X0 → (Rn)	1
STS.L	X1,@-Rn	Rn – 4 → Rn, X1 → (Rn)	1
STS.L	Y0,@-Rn	Rn – 4 → Rn, Y0 → (Rn)	1
STS.L	Y1,@-Rn	Rn – 4 → Rn, Y1 → (Rn)	1
LDS.L	@Rn+,DSR	(Rn) → DSR, Rn + 4 → Rn	1
LDS.L	@Rn+,A0	(Rn) → A0, Rn + 4 → Rn	1
LDS.L	@Rn+,X0	(Rn) → X0, Rn + 4 → Rn	1
LDS.L	@Rn+,X1	(Rn) → X1, Rn + 4 → Rn	1
LDS.L	@Rn+,Y0	(Rn) → Y0, Rn + 4 → Rn	1
LDS.L	@Rn+,Y1	(Rn) → Y1, Rn + 4 → Rn	1
LDS	Rn,DSR	Rn → DSR	1
LDS	Rn,A0	Rn → A0	1
LDS	Rn,X0	Rn → X0	1
LDS	Rn,X1	Rn → X1	1
LDS	Rn,Y0	Rn → Y0	1
LDS	Rn,Y1	Rn → Y1	1

3.4.1 General Registers

The DSP instructions 10 general registers in the 16 general registers as address pointers or index registers for double data transfers and single data transfers. In the following descriptions, another register function in the DSP instructions is also indicated within parentheses [ ].

- Double data transfer instructions (X memory and Y memory are accessed simultaneously)
  In double data transfers, X memory Y memory can be accessed simultaneously. To specify X and Y memory addresses, two address pointers are supported.

Memory to be Accessed	Address Pointer	Index Register
X memory (MOVX.W)	R4,R5[Ax]	R8 [Ix]
Y memory (MOVY.W)	R6,R7[Ay]	R9 [Iy]

- Single data transfer instructions

  In single data transfer, any virtual address space can be accessed via the L bus. The following address pointers and index registers are used.

Address to be Accessed	Address Pointer	Index Register
Any virtual space (MOVS.W/L)	R4,R5, R2, R3[As]	R8 [Is]

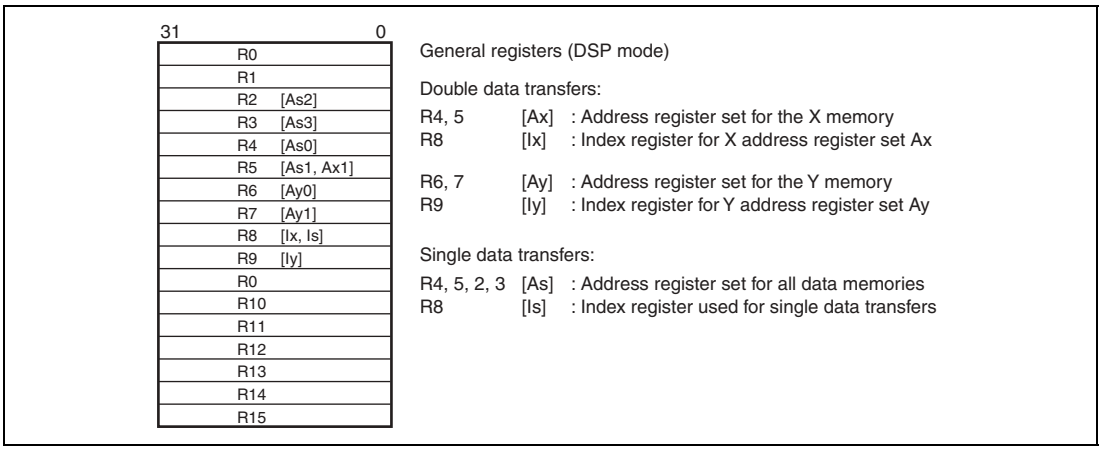


Figure 3.5 General Registers (DSP Mode)

In assembler, R0 to R9 are used as symbols. In the DSP data transfer instructions, the following register names (alias) can also be used. In assembler, described as shown below.

```
Ix: .REG        (R8)
```

Ix indicates the alias of register 8. Other aliases are shown below.

```
Ax0: .REG (R4)

Ax1: .REG (R5)

Ix: .REG (R8)

Ay0: .REG (R6)

Ay1: .REG(R7)
```

Iy: .REG (R9)

As0: .REG (R4); This definition is used for if the alias is required in the single data transfer

As1: .REG (R5) ; This definition is used for if the alias is required in the single data transfer

As2: .REG (R2)

As3: .REG (R3)

Is: .REG (R8) ; This definition is used for if the alias is required in the single data transfer

3.4.2 DSP Data Addressing

Table 3.10 shows the relationship between the double data transfer instructions and single data transfer instructions.

Table 3.10 Overview of Data Transfer Instructions

	Double Data Transfer Instructions	**Single Data Transfer Instructions**
	MOVX.W, MOVY.W	**MOVS.W, MOVS.L**
Address register	Ax: R4, R5, Ay: R6, R7	As: R2, R3, R4, R5
Index register	Ix: R8, Iy: R9	Is: R8
Addressing	Nop/Inc (+2)/index addition: post-increment	Nop/Inc (+2, +4)/index addition: post-increment
	—	Dec (–2, –4): pre-decrement
Modulo addressing	Possible	Not possible
Data bus	XDB, YDB	LDB
Data length	16 bits (word)	16/32 bits (word/longword)
Bus conflict	No	Yes
Memory	X/Y data memory	Entire memory space
Source register	Dx, Dy: A0, A1	Ds: A0/A1, M0/M1, X0/X1, Y0/Y1, A0G, A1G
Destination register	Dx: X0/X1 Dy: Y0/Y1	Ds: A0/A1, M0/M1, X0/X1, Y0/Y1, A0G, A1G

Addressing Mode for Double Data Transfer Instructions: The double data transfer instructions supports the following three addressing modes.

- Non-update address register addressing

  The Ax and Ay registers are address pointers. They are not updated.
- Increment address register addressing

  The Ax and Ay registers are address pointers. After a data transfer, they are each incremented by 2 (post- increment).
- Addition index register addressing

  The Ax and Ay registers are address pointers. After a data transfer, the value of the Ix or Iy register is added to each (post-increment). The double data transfer instructions do not supports decrement addressing mode. To perform decrementing, –2 is set in the index register and addition index register addressing is specified.

When using X/Y memory addressing, bit 0 of the address pointer is invalid. Accordingly, bit 0 of the address pointer and index register must be cleared to 0 in X/Y memory addressing. If 1 is written to the bit, correct operation cannot be guaranteed.

When accessing X and Y memories using the X and Y buses, the upper word of Ax and Ay are ignored. The result of Ay + 2 or Ay + Iy is stored in the lower word of Ay, while the upper word retains its original value. The Ax + 2 and Ax + Ix operations are executed in longword (32 bits) and the upper word may be changed according to the result.

Addressing Mode for Single Data Transfer Instructions: The following four kinds of addressing can be used with single data transfer instructions.

- Non-update address register addressing

  The As register is an address pointer. An access to @As is performed but As is not updated.
- Increment address register addressing:

  The As register is an address pointer. After an access to @As, the As register is incremented by 2 or 4 (post-increment).
- Addition index register addressing:

  The As register is an address pointer. After an access to @As, the value of the Is register is added to the As register (post-increment).
- Decrement address register addressing:

  The As register is an address pointer. Before a data transfer, –2 or –4 is added to the As register (i.e. 2 or 4 is subtracted) (pre-decrement).

In single data transfer instructions, all bits in 32-bit address are valid.

3.4.3 Modulo Addressing

In double data transfer instructions, a modulo addressing can be used. The modulo addressing control instructions are listed in table 3.11. If the address pointer value reaches the preset modulo end address while a modulo addressing mode is specified, the address pointer value becomes the modulo start address.

To control modulo addressing, the modulo register (MOD) extended in the DSP mode and the DMX and DMY bits of the SR register are used.

The MOD register is provided to set the start and end addresses of the modulo address area. The upper and lower words of the MOD register store modulo start address (MS) and modulo end address (ME), respectively. The LDC and STC instructions are extended for MOD register handling.

If the DMX bit of the SR register is set, the modulo addressing is specified for the X address register. If the DMY bit of the SR register is set, the modulo addressing is specified for the Y address register. Modulo addressing is valid for either the X or the Y address register, only; it cannot be set for both at the same time. Therefore, DMX and DMY cannot both be set simultaneously (if they are, the DMY setting will be valid). (In the future, this specification may be changed.) The DMX and MDY bits of the SR can be specified by the STC or LDC instruction for the SR register.

If an exception is accepted during modulo addressing, the MDX and MDY bits of the SR and MOD register must be saved. By restoring these register values, a control is returned to the modulo addressing after an exception handling.

Table 3.11 Modulo Addressing Control Instructions

Instruction		Operation	Execution States
STC	MOD,Rn	MOD → Rn	1
STC.L	MOD,Rn	Rn – 4 → Rn, MOD → (Rn)	1
LDC	@Rn+,MOD	(Rn) → Rn, Rn + 4 → Rn	4
LDC	Rn+,MOD	Rn → MOD	4

An example of the use of modulo addressing is shown below.

```
MOV.L #H'70047000, R10      ;Specify MS=H'7000 ME = H'7004
LDC Rn,MOD            ;Specify ME:MS to MOD register
STC SR, R10           ;
MOV.L #H'FFFFF3FF, R11      ;
MOV.L #H'00000400, R12      ;
AND R11, R10          ;
OR R12, R10           ;
LDC R10, SR           ; Specify SR.MDX=1, SR.MDY=0, and
                        X modulo addressing mode
MOV.L #H'A5007000, R14
MOVX.W @R4+,X0        ; R4: H'A5007000→ H'A5007002
MOVX.W @R4+,X0        ; R4: H'A5007002→ H'A5007004
MOVX.W @R4+,X0        ; R4: H'A5007004→ H'A5007000
                        (Matches to ME and MS is set)
MOVX.W @R4+,X0        ; R4: H'A5007000→ H'A5007002
```

The start and end addresses are specified in MS and ME, then the DMX or DMY bit is set to 1. When the X or Y data transfer instruction specified by the DMX or DMY is executed, the address register contents before updating are compared with ME*, and if they match, start address MS is stored in the address register as the value after updating.

When the addressing type of the X/Y data transfer instruction is no-update, the X/Y data transfer instruction is not returned to MS even if they match ME. When the addressing type of the X/Y data transfer is adding index register, address pointer is may not match and exceed ME. In this case, the modulo start address is not set as the address pointer.

The maximum modulo size is 64 kbytes. This is sufficient to access the X and Y data memory.

Note: Not only with modulo addressing, but when X and Y data addressing is used, bit 0 is ignored. 0 must always be written to bit 0 of the address pointer, index register, MS, and ME.

3.4.4 Memory Data Formats

Memory data formats that can be used in the DSP instructions are classified into word and longword. An address error will occur if word data starting from an address other than 2n is accessed by MOVS.W instruction or longword data starting from an address other than 4n is accessed by MOVS.L, LDS.L, or STS.L instruction. In such cases, the data accessed cannot be guaranteed

An address error will not occur if word data starting from an address other than 2n is accessed by the MOVX.W or MOVY.W instruction. When using the MOVX.W or MOVY.W instruction, an address must be specified on the boundary 2n. If an address is specified other than 2n, the data accessed cannot be guaranteed.

3.4.5 Instruction Formats of Double and Single Data Transfer Instructions

The format of double data transfer instructions is shown in tables 3.12, and that of single data transfer instructions in table 3.13.

Table 3.12 Double Data Transfer Instruction Formats

Type	Mnemonic	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X memory data transfer	NOPX	1	1	1	1	0	0	0		0		0		0	0		
	MOVX.W @Ax,Dx							Ax		Dx		0		0	1		
	MOVX.W @Ax+,Dx													1	0		
	MOVX.W @Ax+Ix,Dx													1	1		
	MOVX.W Da,@Ax									Da		1		0	1		
	MOVX.W Da,@Ax+													1	0		
	MOVX.W Da,@Ax+Ix													1	1		
Y memory data transfer	NOPY	1	1	1	1	0	0		0		0		0			0	0
	MOVY.W @Ay,Dy								Ay		Dy		0			0	1
	MOVY.W @Ay+,Dy															1	0
	MOVY.W @Ay+Iy,Dy															1	1
	MOVY.W Da,@Ay										Da		1			0	1
	MOVY.W Da,@Ay+															1	0
	MOVY.W Da,@Ay+Iy															1	1

Note: Ax: 0 = R4, 1 = R5
Ay: 0 = R6, 1 = R7
Dx: 0 = X0, 1 = X1
Dy: 0 = Y0, 1 = Y1
Da: 0 = A0, 1 = A1

Table 3.13 Single Data Transfer Instruction Formats

Type	Mnemonic	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Single data transfer	MOVS.W @-As,Ds	1	1	1	1	0	1	As		Ds		0:(*)		0	0	0	0
	MOVS.W @As,Ds							0:R4				1:(*)		0	1		
	MOVS.W @As+,Ds							1:R5				2:(*)		1	0		
	MOVS.W @As+Is,Ds							2:R2				3:(*)		1	1		
	MOVS.W Ds,@-As							3:R3				4:(*)		0	0	0	1
	MOVS.W Ds,@As											5:A1		0	1		
	MOVS.W Ds,@As+											6:(*)		1	0		
	MOVS.W Ds,@As+Is											7:A0		1	1		
	MOVS.L @-As,Ds											8:X0		0	0	1	0
	MOVS.L @As,Ds											9:X1		0	1		
	MOVS.L @As+,Ds											A:Y0		1	0		
	MOVS.L @As+Is,Ds											B:Y1		1	1		
	MOVS.L Ds,@-As											C:M0		0	0	1	1
	MOVS.L Ds,@As											D:A1G		0	1		
	MOVS.L Ds,@As+											E:M1		1	0		
	MOVS.L Ds,@As+Is											F:A0G		1	1		

Note: * Codes reserved for system use.

3.5 DSP Data Operation Instructions

3.5.1 DSP Registers

There are eight data registers (A0, A1, X0, X1, Y0, Y1, M0 and M1) and one control register (DSR) as DSP registers (figure 3.3).

Four kinds of operation access the DSP data registers. The first is DSP data processing. When a DSP fixed-point arithmetic operation instruction uses A0 or A1 as the source register, it uses the guard bits (bits 39 to 32). When it uses A0 or A1 as the destination register, guard bits 39 to 32 are valid. When a DSP fixed-point arithmetic operation instruction uses a DSP register other than A0 or A1 as the source register, it sign-extends the source register value to bits 39 to 32. When it uses one of these registers as the destination register, bits 39 to 32 of the result are discarded.

The second kind of operation is an X or Y data transfer operation, MOVX.W, MOVY.W. This operation accesses the X and Y memories through the 16-bit X and Y data buses (figure 3.4). The register to be loaded or stored by this operation always comprises the upper 16 bits (bits 31 to 16). X0 or X1 can be the destination register of an X memory load and Y0 or Y1 can be the destination of a Y memory load, but no other register can be the destination register in this operation. When data is read into the upper 16 bits of a register (bits 31 to 16), the lower 16 bits of the register (bits 15 to 0) are automatically cleared. A0 and A1 can be stored in the X or Y memory by this operation, but no other registers can be stored.

The third kind of operation is a single-data transfer instruction, MOVS.W or MOVS.L. These instructions access any memory location through the LDB (figure 3.4). All DSP registers connect to the LDB and can be the source or destination register of the data transfer. These instructions have word and longword access modes. In word mode, registers to be loaded or stored by this instruction comprise the upper 16 bits (bits 31 to 16) for DSP registers except A0G and A1G. When data is loaded into a register other than A0G and A1G in word mode, the lower half of the register is cleared. When A0 or A1 is used, the data is sign-extended to bits 39–32 and the lower half is cleared. When A0G or A1G is the destination register in word mode, data is loaded into an 8-bit register, but A0 or A1 is not cleared. In longword mode, when the destination register is A0 or A1, it is sign-extended to bits 39 to 32.

The fourth kind of operation is system control instructions such as LDS, STS, LDS.L, or STS.L. The DSR, A0, X0, X1, Y0, and Y1 registers of the DSP register can be treated as system registers. For these registers, data transfer instructions between the CPU general registers and system registers or memory access instructions are supported.

Tables 3.14 and 3.15 show the data type of registers used in DSP instructions. Some instructions cannot use some registers shown in the tables because of instruction code limitations. For example, PMULS can use A1 as the source register, but cannot use A0. These tables ignore details of register selectability.

Table 3.14 Destination Register in DSP Instructions

			Guard Bits	Register Bits	
Registers	**Instructions**		**39 32**	**31 16**	**15 0**
A0, A1	DSP operation	Fixed-point, PSHA, PMULS	Sign-extended	40-bit result	
		Integer, PDMSB	Sign-extended	24-bit result	Cleared
		Logical, PSHL	Cleared	16-bit result	Cleared
	Data transfer	MOVS.W	Sign-extended	16-bit data	Cleared
		MOVS.L	Sign-extended	32-bit data	
A0G, A1G	Data transfer	MOVS.W	8-bit data	No update	
		MOVS.L	8-bit data	No update	
X0, X1 Y0, Y1 M0, M1	DSP operation	Fixed-point, PSHA, PMULS		32-bit result	
		Integer, logical, PDMSB, PSHL		16-bit result	Cleared
	Data transfer	MOVX/Y.W, MOVS.W		16-bit data	Cleared
		MOVS.L		32-bit data	

Table 3.15 Source Register in DSP Operations

			Guard Bits	Register Bits	
Registers	**Instructions**		**39 32**	**31 16**	**15 0**
A0, A1	DSP operation	Fixed-point, PDMSB, PSHA	40-bit data		
		Integer	24-bit data		
		Logical, PSHL, PMULS		16-bit data	
	Data transfer	MOVX/Y.W, MOVS.W		16-bit data	
		MOVS.L		32-bit data	
A0G, A1G	Data transfer	MOVS.W	8-bit data		
		MOVS.L	8-bit data		
X0, X1 Y0, Y1 M0, M1	DSP operation	Fixed-point, PDMSB, PSHA	Sign*	32-bit data	
		Integer	Sign*	16-bit data	
		Logical, PSHL, PMULS		16-bit data	
	Data transfer	MOVS.W		16-bit data	
		MOVS.L		32-bit data	

Note: * The data is sign-extended and input to the ALU.

The DSP unit incorporates the DSP status register (DSR) which is used as a control register. The DSR register stores the DSP data operation result (zero, negative, others). The DSR register also has the DC bit whose function is similar to the T bit of the CPU register. The DC bit functions as status flag. Conditional DSP data operations are controlled based on the DC bit. These operation control affects only the DSP unit instructions. In other words, these operations control affects only the DSP registers and does not affect address register update and CPU instructions such as load and store instructions. A condition to be reflected on the DC bit should be specified to the DC status selection bits (CS[2:0]).

The unconditional DSP type data operation instructions other than PMULS, MOVX, MOVY, and MOVS change the condition flag and DC bit. However, the CPU instructions including the MAC instruction do not modify the DC bit. In addition, conditional DSP data operation instructions do not modify the DSR.

Table 3.16 DSR Register Bits

Bits	Bit Name	Initial Value	R/W	Function
31 to 12	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
11 to 9	TS2 to TS0	All 0	R/W	T Bit Status Selection Specifies the operation result status to be set in the T bit in the SR register if the TC bit is 1. If the S bit of the SR register is set to 1, an overflow is detected. 000: Carry/borrow mode 001: Negative value mode 010: Zero mode 011: Overflow mode 100: Signed greater mode 101: Signed greater than or equal to mode 110: Reserved (setting prohibited) 111: Reserved (setting prohibited)
8	TC	0	R/W	TC Bit 0: The T bit of the SR register is not affected by the DSP instruction. 1: The T bit of the SR register changes according to the TS bit of the DSR register while the DSP instruction is executed. Note, however, the T bit does not change during conditional DSP instruction execution.

Bits	Bit Name	Initial Value	R/W	Function
7	GT	0	R/W	Signed Greater Bit Indicates that the operation result is positive (except 0), or that operand 1 is greater than operand 2 1: Operation result is positive, or operand 1 is greater than operand 2
6	Z	0	R/W	Zero Bit Indicates that the operation result is zero (0), or that operand 1 is equal to operand 2 1: Operation result is zero (0), or operands are equal
5	N	0	R/W	Negative Bit Indicates that the operation result is negative, or that operand 1 is smaller than operand 2 1: Operation result is negative, or operand 1 is smaller than operand 2
4	V	0	R/W	Overflow Bit Indicates that the operation result has overflowed 1: Operation result has overflowed
3 to 1	CS	All 0	R/W	DC Bit Status Selection Designate the mode for selecting the operation result status to be set in the DC bit 000: Carry/borrow mode 001: Negative value mode 010: Zero mode 011: Overflow mode 100: Signed greater mode 101: Signed greater than or equal to mode 110: Reserved (setting prohibited) 111: Reserved (setting prohibited)

Bits	Bit Name	Initial Value	R/W	Function
0	DC	0	R/W	DSP Status Bit Sets the status of the operation result in the mode designated by the CS bits 0: Designated mode status has not occurred (false) 1: Designated mode status has occurred Indicates the operation result by carry or borrow regardless of the CS bit status after the PADDC or PSUBC instruction has been executed.

The DSR is assigned to the system registers. For the DSR, the following load and store instructions are supported.

```
STS DSR,Rn;
STS.L DSR,@-Rn;
LDS Rn,DSR;
LDS.L @Rn+,DSR;
```

If the DSR is read by the STS instruction, upper bits (bits 31 to 16) are all 0

3.5.2 DSP Instruction Set

DSP instructions are instructions for digital signal processing performed by the DSP unit. These instructions have a 32-bit instruction code, and multiple instructions can be executed in parallel. The instruction code is divided into an A field and B field; a parallel data transfer instruction is specified in the A field, and a single or double data operation instruction in the B field. Instructions can be specified independently, and are also executed independently.

B-field data operation instructions are of three kinds: double data operation instructions, conditional single data operation instructions, and unconditional single data operation instructions. The formats of the DSP operation instructions are shown in table 3.17. The respective operands are selected independently from the DSP registers. The correspondence between DSP instruction operands and registers is shown in table 3.18.

Parallel processing only instructions in the A field (without those in the B field) can be executed (transferring data in parallel without DSP data operation instructions).

Table 3.17 DSP Instruction Formats

Type			Instruction Formats
Double data operation instructions			`ALUop. Sx, Sy, Du`
			`MLTop. Se, Df, Dg`
Single data operation instructions	Conditional single data operation instructions	`DCT`	`ALUop. Sx, Sy, Dz`
		`DCF`	`ALUop. Sx, Sy, Dz`
		`DCT`	`ALUop. Sx, Dz`
		`DCF`	`ALUop. Sx, Dz`
		`DCT`	`ALUop. Sy, Dz`
		`DCF`	`ALUop. Sy, Dz`
	Unconditional single data operation instructions		`ALUop. Sx, Sy, Dz`
			`ALUop. Sx, Dz`
			`ALUop. Sy, Dz`
			`MLTop. Se, Sf, Dg`

Table 3.18 Correspondence between DSP Instruction Operands and Registers

	ALU, Shift Operations				Multiply Operations		
Register	**Sx**	**Sy**	**Dz**	**Du**	**Se**	**Sf**	**Dg**
A0	Yes	—	Yes	Yes	—	—	Yes
A1	Yes	—	Yes	Yes	Yes	Yes	Yes
M0	—	Yes	Yes	—	—	—	Yes
M1	—	Yes	Yes	—	—	—	Yes
X0	Yes	—	Yes	Yes	Yes	Yes	—
X1	Yes	—	Yes	—	Yes	—	—
Y0	—	Yes	Yes	Yes	Yes	Yes	—
Y1	—	Yes	Yes	—	—	Yes	—

When writing parallel instructions, the B-field instruction is written first, followed by the A-field instruction. A sample parallel processing program using DSP instructions is shown in figure 3.6. Parallel processing only instructions in the A field (without those in the B field) can be executed (transferring data in parallel without DSP data operation instructions)

```
        PADD  A0, M0, A0   PMULS  X0, Y0, M0   MOVX.W @R4+,     X0   MOVY.W @R6+, Y0
DCF     PINC  M1, A1                          MOVX.W @R5+R8,  X0   MOVY.W @R7+, Y1
        PCMP  M1, M0                          MOVX.W @R4,     X1   [NOPY]
```

Figure 3.6 Sample Parallel DSP Instruction Program

Square brackets mean that the contents can be omitted.

The no operation instructions NOPX and NOPY can be omitted. For details on the B field in DSP instructions, refer to section 3.6.4, DSP Data Operation Instruction Set.

The DSR register condition code bit (DC) is always updated on the basis of the result of an unconditional ALU or shift operation instruction. Conditional instructions do not update the DC bit. Multiply instructions, also, do not update the DC bit. DC bit updating is performed by means of the CS[2:0] bits in the DSR register. The DC bit update rules are shown in table 3.19.

Table 3.19 DC Bit Update Definitions

CS [2:0]			Condition Mode	Description
0	0	0	Carry or borrow mode	The DC bit is set if an ALU arithmetic operation generates a carry or borrow, and is cleared otherwise. When a PSHA or PSHL shift instruction is executed, the last bit data shifted out is copied into the DC bit. When an ALU logical operation is executed, the DC bit is always cleared.
0	0	1	Negative value mode	When an ALU or shift (PSHA) arithmetic operation is executed, the MSB of the result, including the guard bits, is copied into the DC bit. When an ALU or shift (PSHL) logical operation is executed, the MSB of the result, excluding the guard bits, is copied into the DC bit.
0	1	0	Zero value mode	The DC bit is set if the result of an ALU or shift operation is all-zeros, and is cleared otherwise.
0	1	1	Overflow mode	The DC bit is set if the result of an ALU or shift (PSHA) arithmetic operation exceeds the destination register range, excluding the guard bits, and is cleared otherwise. When an ALU or shift (PSHL) logical operation is executed, the DC bit is always cleared.
1	0	0	Signed greater-than mode	This mode is similar to signed greater-or-equal mode, but DC is cleared if the result is all-zeros. DC = ~{(negative value ^ over-range) | zero value}; In case of arithmetic operation DC = 0; In case of logical operation
1	0	1	Signed greater-or-equal mode	If the result of an ALU or shift (PSHA) arithmetic operation exceeds the destination register range, including the guard bits (over-range), the definition is the same as in negative value mode. If the result is not over-range, the definition is the opposite of that in negative value mode. When an ALU or shift (PSHL) logical operation is executed, the DC bit is always cleared. DC = ~(negative value ^ over-range); In case of arithmetic operation DC = 0 ; In case of logical operation
1	1	0	Reserved (setting prohibited)	
1	1	1	Reserved (setting prohibited)	

- Conditional Operations and Data Transfer

  Some instructions belonging to this class can be executed conditionally, as described earlier. The specified condition is valid only for the B field of the instruction, and is not valid for data transfer instructions for which a parallel specification is made. Examples are shown in figure 3.7.

```
DCT PADD X0,Y0,A0  MOVX.W @R4+,X0  MOVY.W A0,@R6+R9 ;

When condition is True

Before execution: X0=H'33333333, Y0=H'55555555, A0=H'123456789A,
                  R4=H'00008000, R6=H'00005000, R9=H'00000004,
                  (R4)=1111  (R6)=2222

After execution:  X0=H'11110000, Y0=H'55555555, A0=H'0088888888,
                  R4=H'00008002, R6=H'00005004, R9=H'00000004,
                  (H'00008000)=H'1111  (H'00008233)=H'2222

When condition is False

Before execution: X0=H'33333333, Y0=H'55555555, A0=H'123456789A,
                  R4=H'00008000, R6=H'00005000, R9=H'00000004,
                  (R4)=1111,  (R6)=2222

After execution:  X0=H'11110000, Y0=H'55555555, A0=H'123456789A,
                  R4=H'00008002, R6=H'00005004, R9=H'00000004,
                  (H'00008000)=H'1111  (H'00008233)=H'2222
```

Figure 3.7 Examples of Conditional Operations and Data Transfer Instructions

- Assignment of NOPX and NOPY Instruction Codes

  When there is no data transfer instruction to be parallel-processed simultaneously with a DSP data operation instruction, an NOPX or NOPY instruction can be written as the data transfer instruction, or the instruction can be omitted. The instruction code is the same whether an NOPX or NOPY instruction is written or the instruction is omitted. When there in no DSP data operation instruction to be parallel-processed simultaneously with a data transfer instruction, an NOPX or NOPY instruction can be written or the instruction can be omitted. Examples of NOPX and NOPY instruction codes are shown in table 3.20.

Table 3.20 Examples of NOPX and NOPY Instruction Codes

Instruction			Code
PADD X0,Y0,A0	MOVX.W @R4+,X0	MOVY.W @R6+R9,Y0	1111100000001011
			1011000100000111
PADD X0,Y0,A0	NOPX	MOVY.W @R6+R9,Y0	1111100000000011
			1011000100000111
PADD X0,Y0,A0	NOPX	NOPY	1111100000000000
			1011000100000111
PADD X0,Y0,A0	NOPX		1111100000000000
			1011000100000111
PADD X0,Y0,A0			1111100000000000
			1011000100000111
	MOVX.W @R4+,X0	MOVY.W @R6+R9,Y0	1111000000001011
	MOVX.W @R4+,X0	NOPY	1111000000001000
	MOVS.W @R4+,X0		1111010010001000
	NOPX	MOVY.W @R6+R9,Y0	1111000000000011
		MOVY.W @R6+R9,Y0	1111000000000011
	NOPX	NOPY	1111000000000000
NOP			0000000000001001

3.5.3 DSP-Type Data Formats

This LSI has several different data formats that depend on the instruction. This section explains the data formats for DSP type instructions.

Figure 3.8 shows three DSP-type data formats with different binary point positions. A CPU-type data format with the binary point to the right of bit 0 is also shown for reference.

The DSP-type fixed point data format has the binary point between bit 31 and bit 30. The DSP-type integer format has the binary point between bit 16 and bit 15. The DSP-type logical format does not have a binary point. The valid data lengths of the data formats depend on the instruction and the DSP register.

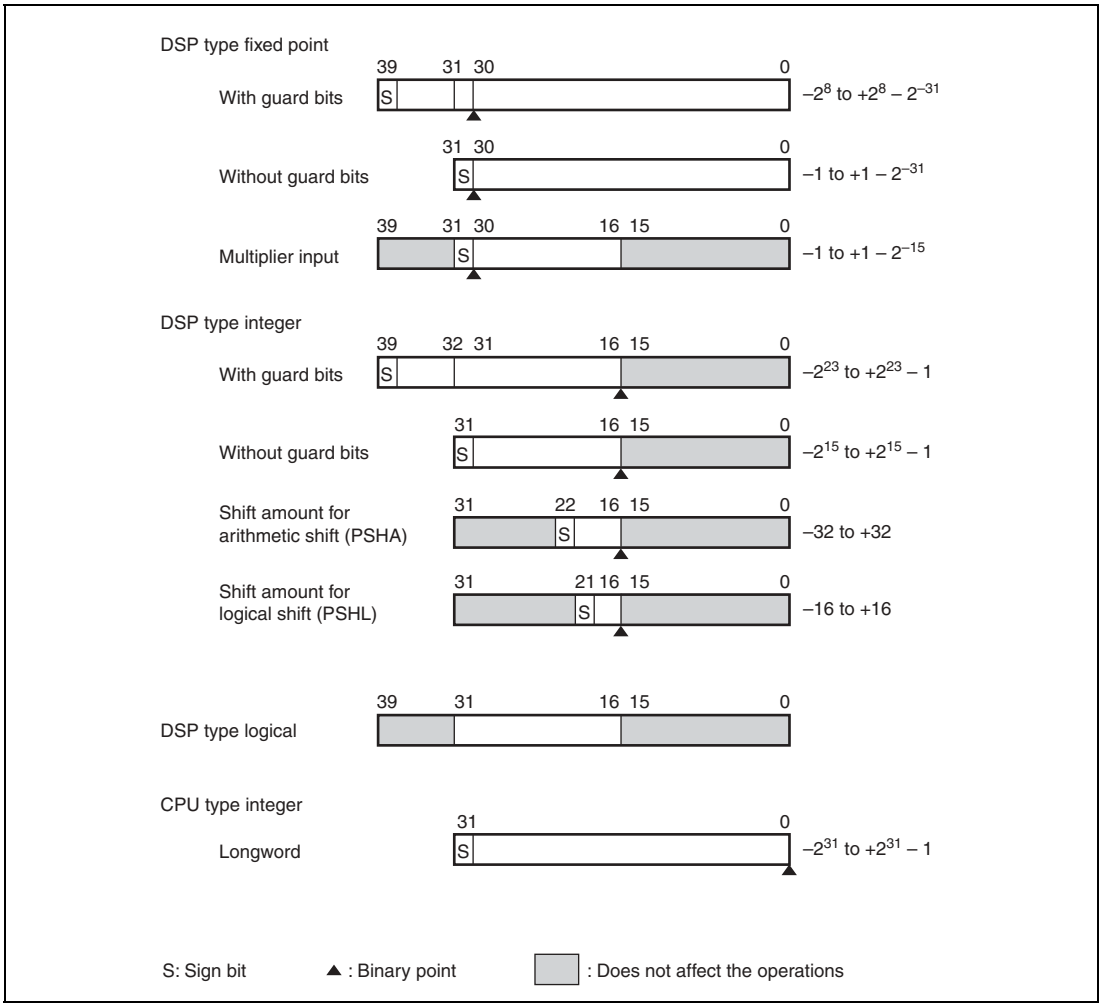


Figure 3.8 Data Formats

The shift amount for the arithmetic shift (PSHA) instruction has a 7-bit field that can represent values from –64 to +63, but –32 to +32 are valid numbers for the instruction. Also the shift amount for a logical shift operation has a 6-bit field, but –16 to +16 are valid numbers for the instruction. The results when an invalid shift amount is specified cannot be guaranteed.

3.5.4 ALU Fixed-Point Arithmetic Operations

Figure 3.9 shows the ALU fixed-point arithmetic operation flow. Table 3.21 shows the variation of this type of operation and table 3.22 shows the correspondence between each operand and registers.

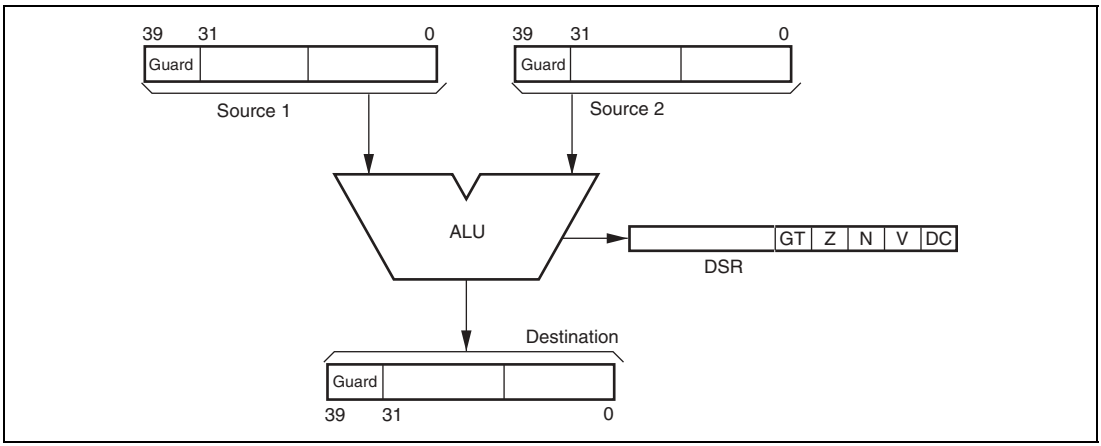


Figure 3.9 ALU Fixed-Point Arithmetic Operation Flow

Note: The ALU fixed-point arithmetic operations are basically 40-bit operation; 32 bits of the base precision and 8 bits of the guard-bit parts. So the signed bit is copied to the guard-bit parts when a register not providing the guard-bit parts is specified as the source operand. When a register not providing the guard-bit parts is specified as a destination operand, the lower 32 bits of the operation result are input into the destination register.

ALU fixed-point operations are executed between registers. Each source and destination operand are selected independently from one of the DSP registers. When a register providing guard bits is specified as an operand, the guard bits are activated for this type of operation. These operations are executed in the DSP stage, as shown in figure 3.10. The DSP stage is the same stage as the MA stage in which memory access is performed.

Table 3.21 Variation of ALU Fixed-Point Operations

Mnemonic	Function	Source 1	Source 2	Destination
PADD	Addition	Sx	Sy	Dz or Du
PSUB	Subtraction	Sx	Sy	Dz or Du
PADDC	Addition with carry	Sx	Sy	Dz
PSUBC	Subtraction with borrow	Sx	Sy	Dz
PCMP	Comparison	Sx	Sy	—
PCOPY	Data copy	Sx	All 0	Dz
		All 0	Sy	Dz
PABS	Absolute	Sx	All 0	Dz
		All 0	Sy	Dz
PNEG	Negation	Sx	All 0	Dz
		All 0	Sy	Dz
PCLR	Clear	All 0	All 0	Dz

Table 3.22 Correspondence between Operands and Registers

Register	Sx	Sy	Dz	Du
A0	Yes	—	Yes	Yes
A1	Yes	—	Yes	Yes
M0	—	Yes	Yes	—
M1	—	Yes	Yes	—
X0	Yes	—	Yes	Yes
X1	Yes	—	Yes	—
Y0	—	Yes	Yes	Yes
Y1	—	Yes	Yes	—

As shown in figure 3.10, data loaded from the memory at the MA stage to X0 by the data transfer instruction (MOVX.W @R4+,X0) which is programmed at the same line as the ALU operation (PADD X0,Y0,A0) , is not used as a source operand for this operation, even though the destination operand X0 of the data load operation is identical to the source operand of the ALU operation. In this case, results of previous operation (MOVX.W @R4+,X0) are used as the source operands for the ALU operation, and then updated as the destination operand X0 of the data transfer instruction (MOVX.W @R4+,X0). .

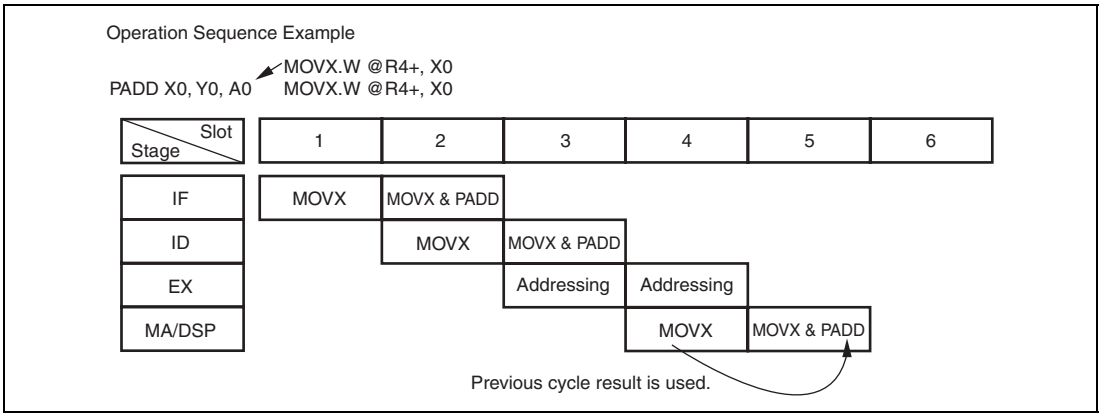


Figure 3.10 Operation Sequence Example

Every time an ALU arithmetic operation is executed, the DC, N, Z, V, and GT bits in DSR are basically updated in accordance with the operation result. However, in case of a conditional operation, they are not updated even though the specified condition is true and the operation is executed. In case of an unconditional operation, they are always updated in accordance with the operation result. The definition of a DC bit is selected by CS[2:0] (condition selection) bits in DSR. The DC bit result is as follows:

Carry or Borrow Mode: CS[2:0] = B'000: The DC bit indicates that carry or borrow is generated from the most significant bit of the operation result, except the guard-bit parts. Some examples are shown in figure 3.11. This mode is the default condition. When the input data is negative in a PABS or PNEG instruction, carry is generated.

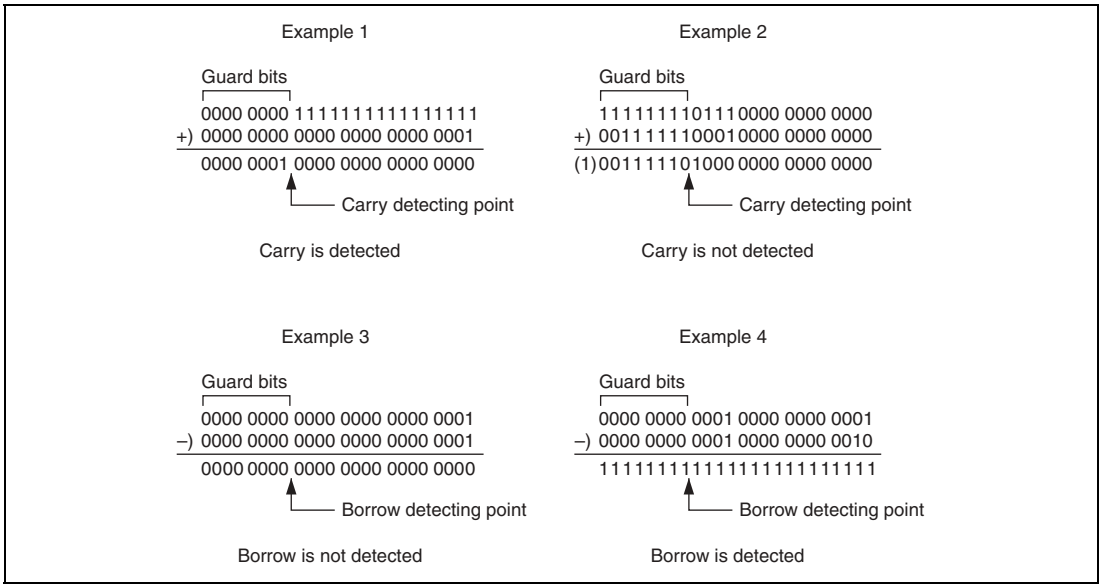


Figure 3.11 DC Bit Generation Examples in Carry or Borrow Mode

Negative Value Mode: CS[2:0] = B'001: The DC flag indicates the same value as the MSB of the operation result. When the result is a negative number, the DC bit shows 1. When it is a positive number, the DC bit shows 0. The ALU always executes 40-bit arithmetic operation, so the sign bit to detect whether positive or negative is always got from the MSB of the operation result regardless of the destination operand. Some examples are shown in figure 3.12.

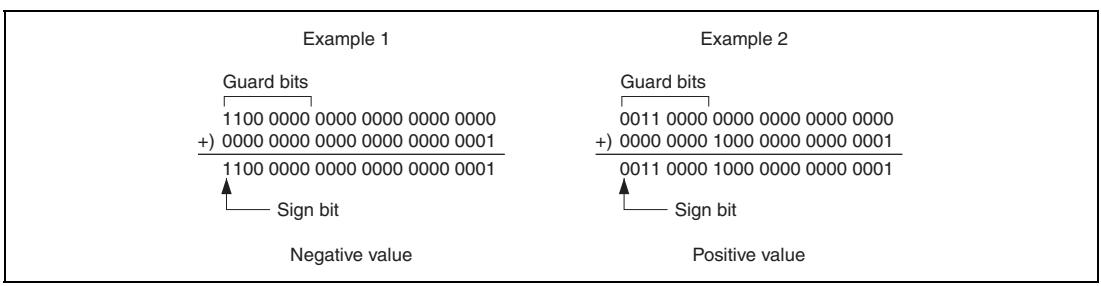


Figure 3.12 DC Bit Generation Examples in Negative Value Mode

Zero Value Mode: CS[2:0] = B'010: The DC flag indicates whether the operation result is 0 or not. When the result is 0, the DC bit shows 1. When it is not 0, the DC bit shows 0.

Overflow Mode: CS[2:0] = B'011: The DC bit indicates whether or not overflow occurs in the result. When an operation yields a result beyond the range of the destination register, except the guard-bit parts, the DC bit is set. Even though guard bits are provided in the destination register, the DC bit always indicates the result of when no guard bits are provided. So, the DC bit is always

set if the guard-bit parts are used for large number representation. Some DC bit generation examples in overflow mode are shown in figure 3.13.

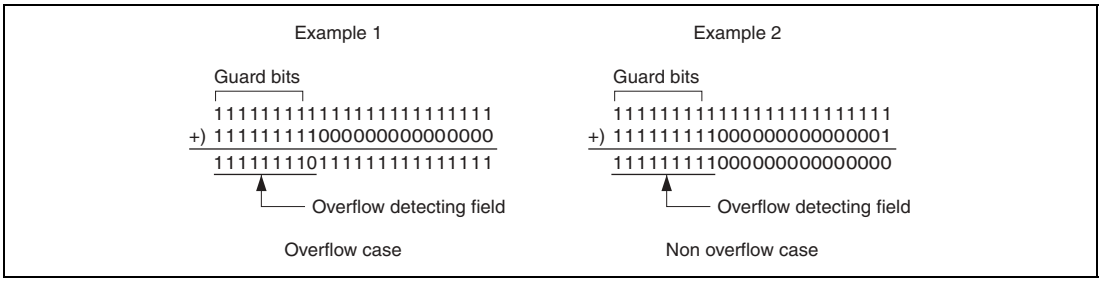


Figure 3.13 DC Bit Generation Examples in Overflow Mode

Signed Greater Than Mode: CS[2:0] = B'100: The DC bit indicates whether or not the source 1 data (signed) is greater than the source 2 data (signed) as the result of compare operation PCMP. Therefore, the PCMP operation should be executed before the conditional operation is executed under this condition mode. This mode is similar to the Negative Value Mode described before, because the result of a compare operation is a positive value if the source 1 data is greater than the source 2 data. However, the signed bit of the result shows a negative value if the compare operation yields a result beyond the range of the destination operand, including the guard-bit parts (called "Over-range"), even though the source 1 data is greater than the source 2 data. The DC bit is updated concerning this type of special case in this condition mode. The equation below shows the definition of getting this condition:

DC = ~ {(Negative ^ Over-range) | Zero}

When the PCMP operation is executed under this condition mode, the result of the DC bit is the same as the T bit's result of the CMP/GT operation of the CPU instruction.

Signed Greater Than or Equal Mode: CS[2:0] = B'101: The DC bit indicates whether the source 1 data (signed) is greater than or equal to the source 2 data (signed) as the result of compare operation PCMP. Therefore, the PCMP operation should be executed before the conditional operation is executed under this condition mode. This mode is similar to the Signed Greater Than Mode described before but the equal case is also included in this mode. The equation below shows the definition of getting this condition:

DC = ~ (Negative ^ Over-range)

When the PCMP operation is executed under this condition mode, the result of the DC bit is the same as the T bit's result of a CMP/GE operation of the CPU instruction.

The N bit always indicates the same state as the DC bit set in negative value mode by the CS[2:0] bits. See the negative value mode part above. The Z bit always indicates the same state as the DC bit set in zero value mode by the CS[2:0] bits. See the zero value mode part above. The V bit

always indicates the same state as the DC bit set in overflow mode by the CS[2:0] bits. See the overflow mode part above. The GT bit always indicates the same state as the DC bit set in signed greater than mode by the CS[2:0] bits. See the signed greater than mode part above.

Note: The DC bit is always updated as the carry flag for ‘PADDC’ and is always updated as the carry/borrow flag for ‘PSUBC’ regardless of the CS[2:0] state.

- Overflow Protection

  The S bit in SR is effective for any ALU fixed-point arithmetic operations in the DSP unit. See section 3.5.11, Overflow Protection, for details.

3.5.5 ALU Integer Operations

Figure 3.14 shows the ALU integer arithmetic operation flow. Table 3.23 shows the variation of this type of operation. The correspondence between each operand and registers is the same as ALU fixed-point operations as shown in table 3.22.

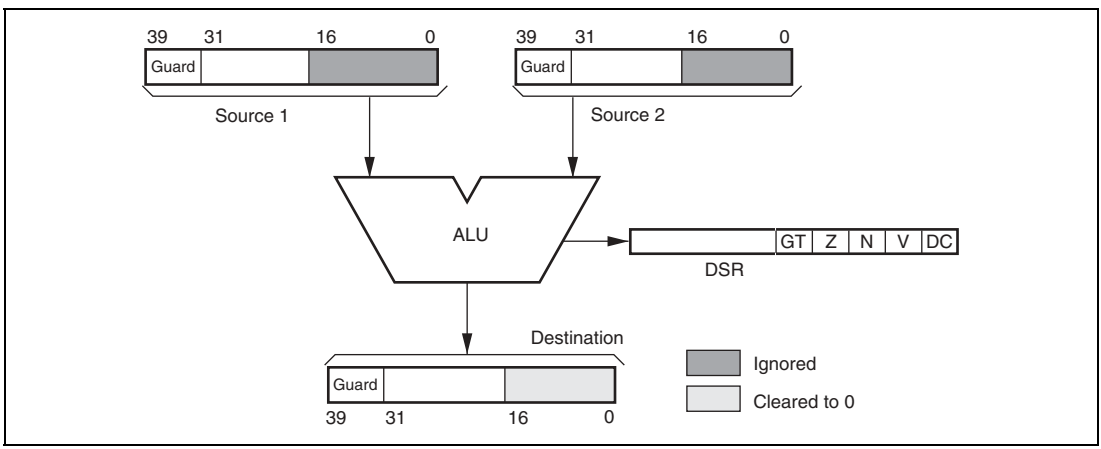


Figure 3.14 ALU Integer Arithmetic Operation Flow

Table 3.23 Variation of ALU Integer Operations

Mnemonic	Function	Source 1	Source 2	Destination
PINC	Increment by 1	Sx	+1	Dz
		+1	Sy	Dz
PDEC	Decrement by 1	Sx	–1	Dz
		–1	Sy	Dz

Note: The ALU integer operations are basically 24-bit operation, the upper 16 bits of the base precision and 8 bits of the guard-bits parts. So the signed bit is copied to the guard-bit parts when a register not providing the guard-bit parts is specified as the source operand. When a register not providing the guard-bit parts is specified as a destination operand, the upper word excluding the guard bits of the operation result are input into the destination register.

In ALU integer arithmetic operations, the lower word of the source operand is ignored and the lower word of the destination operand is automatically cleared. The guard-bit parts are effective in integer arithmetic operations if they are supported. Others are basically the same operation as ALU fixed-point arithmetic operations. As shown in table 3.23, however, this type of operation provides two kinds of instructions only, so that the second operand is actually either +1 or –1. When a word data is loaded into one of the DSP unit's registers, it is input as an upper word data. When a register providing guard bits is specified as an operand, the guard bits are also activated. These operations, as well as ALU fixed-point arithmetic operations, are executed in the DSP stage, as shown in figure 3.10. The DSP stage is the same stage as the MA stage in which memory access is performed.

Every time an ALU arithmetic operation is executed, the DC, N, Z, V, and GT bits in DSR are basically updated in accordance with the operation result. This is the same as ALU fixed-point arithmetic operations but the lower word of each source and destination operand is not used in order to generate them. See section 3.5.4, ALU Fixed-Point Arithmetic Operations, for details.

In case of a conditional operation, they are not updated even though the specified condition is true and the operation is executed. In case of an unconditional operation, they are always updated in accordance with the operation result. See section 3.5.4, ALU Fixed-Point Arithmetic Operations, for details.

- Overflow Protection

  The S bit in SR is effective for any ALU integer arithmetic operations in DSP unit. See section 3.5.11, Overflow Protection, for details.

3.5.6 ALU Logical Operations

Figure 3.15 shows the ALU logical operation flow. Table 3.24 shows the variation of this type of operation. The correspondence between each operand and registers is the same as the ALU fixed-point arithmetic operations as shown in table 3.21.

The ALU logical operation is executed between registers. Each source operand and destination operand is selected independently from one of the DSP registers. As shown in figure 3.15, this type of operation uses only the upper word of each operand. The lower word and guard-bit parts are ignored for the source operand and those of the destination operand are automatically cleared. These operations are also executed in the DSP stage, as shown in figure 3.10. The DSP stage is the same stage as the MA stage in which memory access is performed.

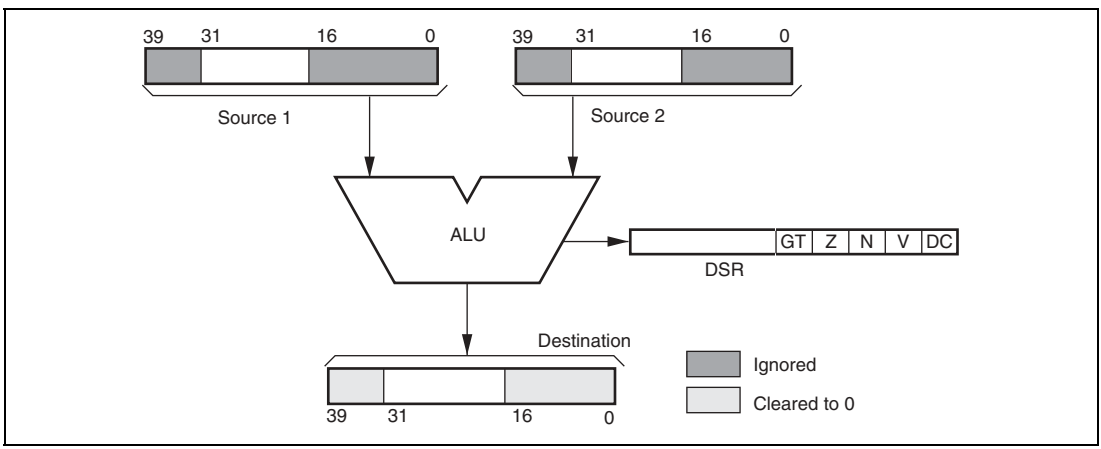


Figure 3.15 ALU Logical Operation Flow

Table 3.24 Variation of ALU Logical Operations

Mnemonic	Function	Source 1	Source 2	Destination
PAND	Logical AND	Sx	Sy	Dz
POR	Logical OR	Sx	Sy	Dz
PXOR	Logical exclusive OR	Sx	Sy	Dz

Every time an ALU logical operation is executed, the DC, N, Z, V, and GT bits in the DSR register are basically updated in accordance with the operation result. In case of a conditional operation, they are not updated even though the specified condition is true and the operation is executed. In case of an unconditional operation, they are always updated in accordance with the operation result. The definition of the DC bit is selected by the CS[2:0] (DC bit condition selection) bits in DSR. The DC bit result is:

Carry or Borrow Mode: CS[2:0] = B'000: The DC bit is always cleared to 0.

Negative Value Mode: CS[2:0] = B'001: Bit 31 of the operation result is loaded into the DC bit.

Zero Value Mode: CS[2:0] = B'010: The DC bit is set to1when the operation result is zero; otherwise it is cleared to 0.

Overflow Mode: CS[2:0] = B'011: The DC bit is always cleared to 0.

Signed Greater Than Mode: CS[2:0] = B'100: The DC bit is always cleared to 0.

Signed Greater Than or Equal Mode: CS[2:0] = B'101: The DC bit is always cleared to 0.

The N bit always indicates the same state as the DC bit set in negative value mode by the CS[2:0] bits. See the negative value mode part above. The Z bit always indicates the same state as the DC bit set in zero value mode by the CS[2:0] bits. See the zero value mode part above. The V bit always indicates the same state as the DC bit set in overflow mode by the CS[2:0] bits. See the overflow mode part above. The GT bit always indicates the same state as the DC bit set in signed greater than mode by the CS[2:0] bits. See the signed greater than mode part above.

3.5.7 Fixed-Point Multiply Operation

Figure 3.16 shows the fixed-point multiply operation flow. Table 3.25 shows the variation of this type of operation and table 3.26 shows the correspondence between each operand and registers. The multiply operation of the DSP unit is single-word signed single-precision multiplication. The fixed-point multiply operations are executed in the DSP stage, as shown in figure 3.10. The DSP stage is the same stage as the MA stage in which memory access is performed.

If a double-precision multiply operation is needed, the CPU double-word multiply instructions can be made of use.

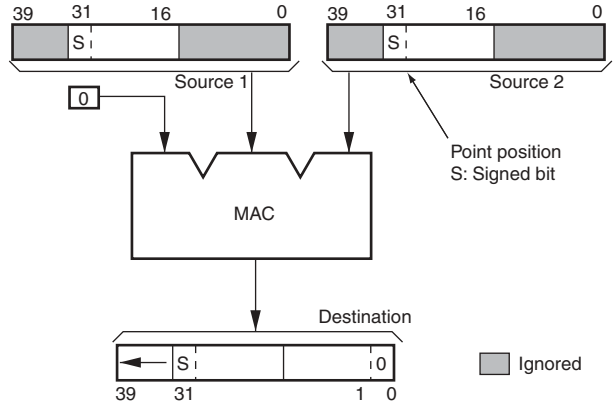


Figure 3.16 Fixed-Point Multiply Operation Flow

Table 3.25 Variation of Fixed-Point Multiply Operation

Mnemonic	Function	Source 1	Source 2	Destination
PMULS	Signed multiplication	Se	Sf	Dg

Table 3.26 Correspondence between Operands and Registers

Register	Se	Sf	Dg
A0	—	—	Yes
A1	Yes	Yes	Yes
M0	—	—	Yes
M1	—	—	Yes
X0	Yes	Yes	—
X1	Yes	—	—
Y0	Yes	Yes	—
Y1	—	Yes	—

Note: The multiply operations basically generate 32-bit operation results. So when a register providing the guard-bit parts are specified as a destination operand, the guard-bit parts will copy bit 31 of the operation result.

The multiply operation of the DSP unit side is not integer but fixed-point arithmetic. So, the upper words of each multiplier and multiplicand are input into a MAC unit as shown in figure 3.16. In the CPU instruction multiply operations, the lower words of both source operands are input into a MAC unit. The operation result is also different from the CPU's case. The CPU instruction multiply operation result is aligned to the LSB of the destination, but the fixed-point multiply operation result of DSP unit is aligned to the MSB, so that the LSB of the fixed-point multiply operation result is always 0.

Multiply is always unconditional, but does not affect any condition code bits, DC, N, Z, V, and GT, in DSR.

- Overflow Protection

  The S bit in SR is effective for this multiply operation in the DSP unit. See section 3.5.11, Overflow Protection, for details.

  If the S bit is 0, overflow occurs only when `H'8000*H'8000 ((-1.0)*(-1.0))` operation is executed as signed fixed-point multiply. The result is `H'00 8000 0000` but it does not mean (`+1.0`). If the S bit is 1, overflow is prevented and the result is `H'00 7FFF FFFF`.

3.5.8 Shift Operations

Shift operations barrel shift and can use either register or immediate value as the shift amount operand. Other source and destination operands are specified by the register. There are two kinds of shift operations of arithmetic and logical shifts. Table 3.27 shows the variation of this type of operation. The correspondence between each operand and registers, except for immediate operands, is the same as the ALU fixed-point operations as shown in table 3.21.

Table 3.27 Variation of Shift Operations

Mnemonic	Function	Source 1	Source 2	Destination
PSHA Sx, Sy, Dz	Arithmetic shift	Sx	Sy	Dz
PSHL Sx, Sy, Dz	Logical shift	Sx	Sy	Dz
PSHA #Imm1, Dz	Arithmetic shift with immediate	Dz	Imm1	Dz
PSHL #Imm2, Dz	Logical shift with immediate	Dz	Imm2	Dz
				$-32 \le Imm1 \le +32$, $-16 \le Imm2 \le +16$

Arithmetic Shift: Figure 3.17 shows the arithmetic shift operation flow.

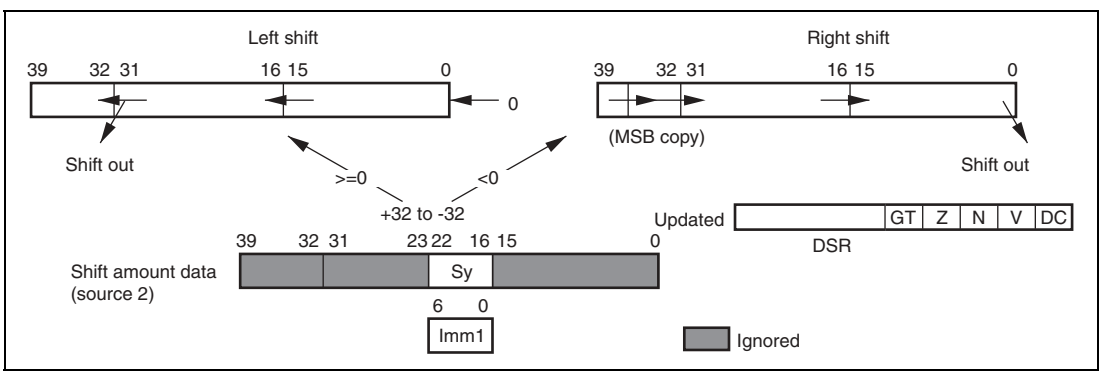


Figure 3.17 Arithmetic Shift Operation Flow

Note: The arithmetic shift operations are basically 40-bit operation, that is, the 32 bits of the base precision and eight bits of the guard-bit parts. So the signed bit is copied to the guard-bit parts when a register not providing the guard-bit parts is specified as the source operand. When a register not providing the guard-bit parts is specified as a destination operand, the lower 32 bits of the operation result are input into the destination register.

In this arithmetic shift operation, all bits of the source 1 and destination operands are activated. The shift amount is specified by the source 2 operand as an integer data. The source 2 operand can be specified by either a register or immediate operand. The available shift range is from –32 to

+32. Here, a negative value means the right shift, and a positive value means the left shift. It is possible for any source 2 operand to specify from –64 to +63 but the result is unknown if an invalid shift value is specified. In case of a shift with an immediate operand instruction, the source 1 operand must be the same register as the destination's. This operation is executed in the DSP stage, as shown in figure 3.10 as well as in ALU fixed-point arithmetic operations. The DSP stage is the same stage as the MA stage in which memory access is performed.

Every time an arithmetic shift operation is executed, the DC, N, Z, V, and GT bits in DSR are basically updated in accordance with the operation result. In case of a conditional operation, they are not updated even though the specified condition is true and the operation is executed. In case of an unconditional operation, they are always updated in accordance with the operation result. The definition of the DC bit is selected by the CS[2:0] (DC bit condition selection) bits in DSR. The DC bit result is:

1. Carry or Borrow Mode: CS[2:0] = B'000
   The DC bit indicates the last shifted out data as the operation result.
2. Negative Value Mode: CS[2:0] = B'001
   The DC bit is set to 1 when the operation result is a negative value, and cleared to 0 when the operation result is zero or a positive value.
3. Zero Value Mode: CS[2:0] = B'010
   The DC bit is set to 1 when the operation result is zero; otherwise it is cleared to 0.
4. Overflow Mode: CS[2:0] = B'011
   The DC bit is set to 1 when an overflow occurs.
5. Signed Greater Than Mode: CS[2:0] = B'100
   The DC bit is always cleared to 0.
6. Signed Greater Than or Equal Mode: CS[2:0] = B'101
   The DC bit is always cleared to 0.

The N bit always indicates the same state as the DC bit set in negative value mode by the CS[2:0] bits. See the negative value mode part above. The Z bit always indicates the same state as the DC bit set in zero value mode by the CS[2:0] bits. See the zero value mode part above. The V bit always indicates the same state as the DC bit set in overflow mode by the CS[2:0] bits. See the overflow mode part above. The GT bit always indicates the same state as the DC bit set in signed greater than mode by the CS[2:0] bits. See the signed greater than mode part above.

- Overflow Protection
  The S bit in SR is also effective for arithmetic shift operation in the DSP unit. See section 3.5.11, Overflow Protection, for details.

Logical Shift: Figure 3.18 shows the logical shift operation flow.

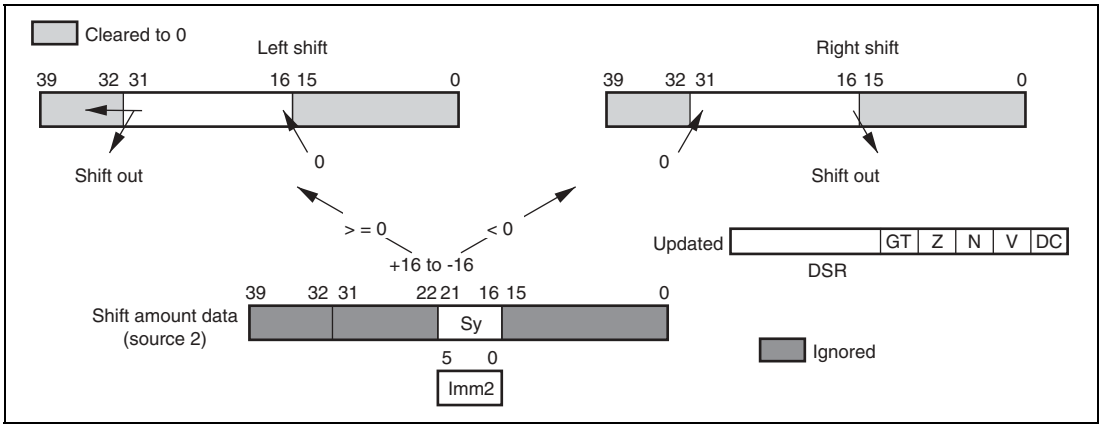


Figure 3.18 Logical Shift Operation Flow

As shown in figure 3.18, the logical shift operation uses the upper word of the source 1 operand and the destination operand. The lower word and guard-bit parts are ignored for the source operand and those of the destination operand are automatically cleared as in the ALU logical operations. The shift amount is specified by the source 2 operand as an integer data. The source 2 operand can be specified by either the register or immediate operand. The available shift range is from –16 to +16. Here, a negative value means the right shift, and a positive value means the left shift. It is possible for any source 2 operand to specify from –32 to +31, but the result is unknown if an invalid shift value is specified. In case of a shift with an immediate operand instruction, the source 1 operand must be the same register as the destination's. These operations are executed in the DSP stage, as shown in figure 3.10. The DSP stage is the same stage as the MA stage in which memory access is performed.

Every time a logical shift operation is executed, the DC, N, Z, V, and GT bits in DSR are basically updated in accordance with the operation result. In case of a conditional operation, they are not updated even though the specified condition is true and the operation is executed. In case of an unconditional operation, they are always updated in accordance with the operation result. The definition of the DC bit is selected by the CS[2:0] (DC bit condition selection) bits in DSR. The DC bit result is:

1. Carry or Borrow Mode: CS[2:0] = B'000
   The DC bit indicates the last shifted out data as the operation result.
2. Negative Value Mode: CS[2:0] = B'001
   Bit 31 of the operation result is loaded into the DC bit.
3. Zero Value Mode: CS[2:0] = B'010
   The DC bit is set to 1 when the operation result is zero; otherwise it is cleared to 0.

4. Overflow Mode: CS[2:0] = B'011
   The DC bit is always cleared to 0.
5. Signed Greater Than Mode: CS[2:0] = B'100
   The DC bit is always cleared to 0.
6. Signed Greater Than or Equal Mode: CS[2:0] = B'101
   The DC bit is always cleared to 0.

The N bit always indicates the same state as the DC bit set in negative value mode by the CS[2:0] bits. See the negative value mode part above. The Z bit always indicates the same state as the DC bit set in zero value mode by the CS[2:0] bits. See the zero value mode part above. The V bit always indicates the same state as the DC bit set in overflow mode by the CS[2:0] bits, but it is always cleared in this operation. So is the GT bit.

3.5.9 Most Significant Bit Detection Operation

The PDMSB, most significant bit detection operation, is used to calculate the shift amount for normalization. Figure 3.19 shows the PDMSB operation flow and table 3.28 shows the operation definition. Table 3.29 shows the possible variations of this type of operation. The correspondence between each operand and registers is the same as for ALU fixed-point operations, as shown in table 3.21.

Note: The result of the MSB detection operation is basically 24 bits as well as ALU integer operation, the upper 16 bits of the base precision and eight bits of the guard-bit parts. When a register not providing the guard-bit parts is specified as a destination operand, the upper word of the operation result is input into the destination register.

As shown in figure 3.19, the PDMSB operation uses all bits as a source operand, but the destination operand is treated as an integer operation result because shift amount data for normalization should be integer data as described in section 3.5.8, Arithmetic Shift. These operations are executed in the DSP stage, as shown in figure 3.10. The DSP stage is the same stage as the MA stage in which memory access is performed.

Every time a PDMSB operation is executed, the DC, N, Z, V, and GT bits in DSR are basically updated in accordance with the operation result. In case of a conditional operation, they are not updated, even though the specified condition is true, and the operation is executed. In case of an unconditional operation, they are always updated with the operation result.

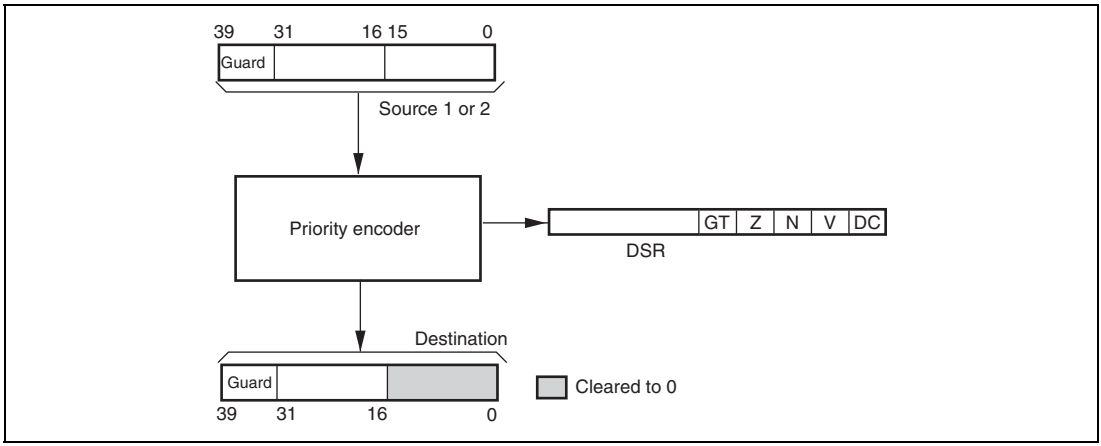


Figure 3.19 PDMSB Operation Flow

The definition of the DC bit is selected by the CS[2:0] (DC bit condition selection) bits in DSR. The DC bit result is

Carry or Borrow Mode: CS[2:0] = B'000: The DC bit is always cleared to 0.

Negative Value Mode: CS[2:0] = B'001: The DC bit is set to 1 when the operation result is a negative value, and cleared to 0 when the operation result is zero or a positive value.

Zero Value Mode: CS[2:0] = B'010: The DC bit is set to 1 when the operation result is zero; otherwise it is cleared to 0.

Overflow Mode: CS[2:0] = B'011: The DC bit is always cleared to 0.

Signed Greater Than Mode: CS[2:0] = B'100: The DC bit is set to 1 when the operation result is a positive value; otherwise it is cleared to 0.

Signed Greater Than or Equal Mode: CS[2:0] = B'101: The DC bit is set to 1 when the operation result is zero or a positive value; otherwise it is cleared to 0.

Table 3.28 Operation Definition of PDMSB

Source Data														Result for DST							
Guard Bit					Upper Word					Lower Word				Guard Bit	Upper Word						
39	38	…	33	32	31	30	29	28	…	3	2	1	0	39–32	31–22	21	20	19	18	17	16	Decimal
0	0	…	0	0	0	0	0	0	…	0	0	0	0	All 0	All 0	0	1	1	1	1	1	+31
0	0	…	0	0	0	0	0	0	…	0	0	0	1	All 0	All 0	0	1	1	1	1	0	+30
0	0	…	0	0	0	0	0	0	…	0	0	1	*	All 0	All 0	0	1	1	1	0	1	+29
0	0	…	0	0	0	0	0	0	…	0	1	*	*	All 0	All 0	0	1	1	1	0	0	+28
		:							:									:				
0	0	…	0	0	0	0	0	1	…	*	*	*	*	All 0	All 0	0	0	0	0	1	0	+2
0	0	…	0	0	0	0	1	*	…	*	*	*	*	All 0	All 0	0	0	0	0	0	1	+1
0	0	…	0	0	0	1	*	*	…	*	*	*	*	All 0	All 0	0	0	0	0	0	0	0
0	0	…	0	0	1	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	1	1	1	–1
0	0	…	0	1	*	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	1	1	0	–2
		:							:									:				
0	1	…	*	*	*	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	0	0	0	–8
1	0	…	*	*	*	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	0	0	0	–8
																		:				
1	1	…	1	0	*	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	1	1	0	–2
1	1	…	1	1	0	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	1	1	1	–1
1	1	…	1	1	1	0	*	*	…	*	*	*	*	All 0	All 0	0	0	0	0	0	0	0
1	1	…	1	1	1	1	0	*	…	*	*	*	*	All 0	All 0	0	0	0	0	0	1	+1
1	1	…	1	1	1	1	1	0	…	*	*	*	*	All 0	All 0	0	0	0	0	1	0	+2
		:							:									:				
1	1	…	1	1	1	1	1	1	…	1	0	*	*	All 0	All 0	0	1	1	1	0	0	+28
1	1	…	1	1	1	1	1	1	…	1	1	0	*	All 0	All 0	0	1	1	1	0	1	+29
1	1	…	1	1	1	1	1	1	…	1	1	1	0	All 0	All 0	0	1	1	1	1	0	+30
1	1	…	1	1	1	1	1	1	…	1	1	1	1	All 0	All 0	0	1	1	1	1	1	+31

Note: * means don't care.

Table 3.29 Variation of PDMSB Operation

Mnemonic	Function	Source	Source 2	Destination
PDMSB	MSB detection	Sx	—	Dz
		—	Sy	Dz

The N bit always indicates the same state as the DC bit set in negative value mode by the CS[2:0] bits. See the negative value mode part above. The Z bit always indicates the same state as the DC bit set in zero value mode by the CS[2:0] bits. See the zero value mode part above. The V bit is always cleared. The GT bit always indicates the same state as the DC bit set in signed greater than mode by the CS[2:0] bits. See the signed greater than mode part above.

3.5.10 Rounding Operation

The DSP unit provides the function that rounds from 32 bits to 16 bits. In case of providing guard-bit parts, it rounds from 40 bits to 24 bits. When a round instruction is executed, H'00008000 is added to the source operand data and then, the lower word is cleared. Figure 3.20 shows the rounding operation flow and figure 3.21 shows the operation definition. Table 3.30 shows the variation rounding operation. The correspondence between each operand and registers is the same as ALU fixed-point operations as shown in table 3.21.

As shown in figure 3.21, the rounding operation uses full-size data for both source and destination operands. These operations are executed in the DSP stage as shown in figure 3.10. The DSP stage is the same stage as the MA stage in which memory access is performed.

The rounding operation is always executed unconditionally, so that the DC, N, Z, V, and GT bits in DSR are always updated in accordance with the operation result. The definition of the DC bit is selected by the CS[2:0] (DC bit condition selection) bits in DSR. The result of these condition code bits is the same as the ALU-fixed point arithmetic operations.

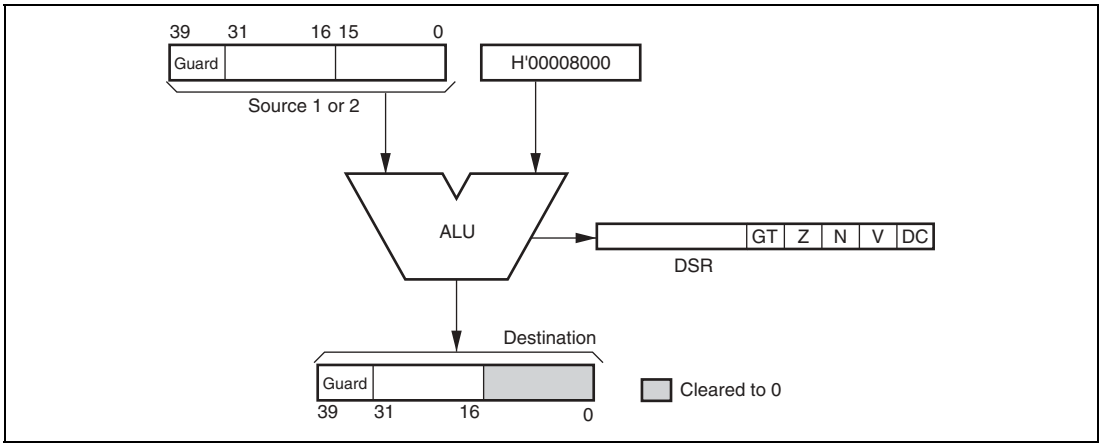


Figure 3.20 Rounding Operation Flow

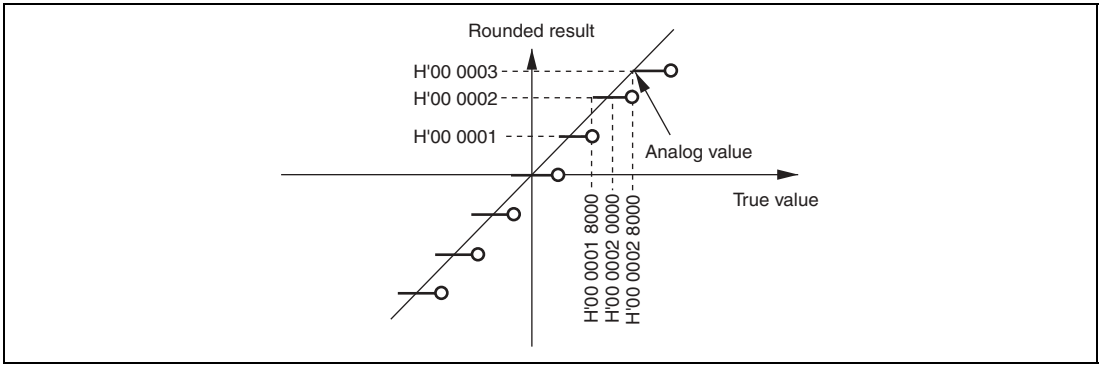


Figure 3.21 Definition of Rounding Operation

Table 3.30 Variation of Rounding Operation

Mnemonic	Function	Source 1	Source 2	Destination
PRND	Rounding	Sx	—	Dz
		—	Sy	Dz

- Overflow Protection

  The S bit in SR is effective for any rounding operations in the DSP unit. See section 3.5.11, Overflow Protection, for details.

3.5.11 Overflow Protection

The S bit in SR is used as the overflow protection enable bit. The S bit is effective for any arithmetic operations executed in the DSP unit, including the CPU instruction multiply and MAC operations. The arithmetic operation overflows when the operation result exceeds the range of two's complement representation without guard-bit parts. Table 3.31 shows the definition of overflow protection for fixed-point arithmetic operations, including fixed-point multiplication described in section 3.5.7, Fixed-Point Multiply Operation. Table 3.32 shows the definition of overflow protection for integer arithmetic operations. The lower word of the saturation value of the integer arithmetic operation is don't care. Lower word value cannot be guaranteed.

When the overflow protection is effective, overflow never occurs. So, the V bit is cleared, and the DC bit is also cleared when the overflow mode is selected by the CS[2:0] bits.

Table 3.31 Definition of Overflow Protection for Fixed-Point Arithmetic Operations

Sign	Overflow Condition	Fixed Value	Hex Representation
Positive	Result > $1 - 2^{-31}$	$1 - 2^{-31}$	00 7FFF FFFF
Negative	Result < –1	–1	FF 8000 0000

Table 3.32 Definition of Overflow Protection for Integer Arithmetic Operations

Sign	Overflow Condition	Fixed Value	Hex Representation
Positive	Result > $2^{15} - 1$	$2^{15} - 1$	00 7FFF ****
Negative	Result < -2^{15}	-2^{15}	FF 8000 ****

Note: * means don't care.

3.5.12 Local Data Move Instruction

The DSP unit of this LSI provides additional two independent registers, MACL and MACH, in order to support CPU instruction multiply/MAC operations. They can be also used as temporary storage registers by local data move instructions between MACH/L and other DSP registers. Figure 3.22 shows the flow of seven local data move instructions. Table 3.33 shows the variation of this type of instruction.

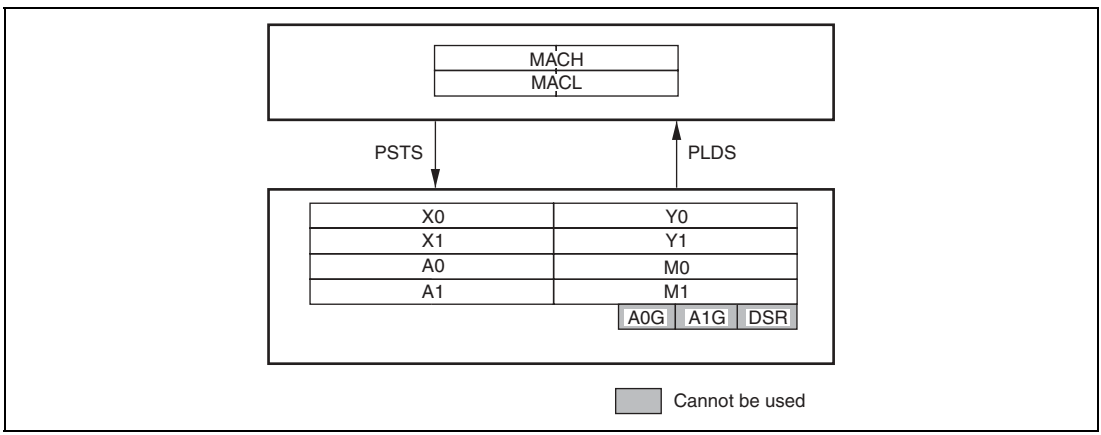


Figure 3.22 Local Data Move Instruction Flow

Table 3.33 Variation of Local Data Move Operations

Mnemonic	Function	Operand
PLDS	Data move from DSP register to MACL/MACH	Dz
PSTS	Data move from MACL/MACH to DSP register	Dz

This instruction is very similar to other transfer instructions. If either the A0 or A1 register is specified as the destination operand of PSTS, the signed bit is sign-extended and copied into the corresponding guard-bit parts, A0G or A1G. The DC bit in DSR and other condition code bits are not updated regardless of the instruction result. This instruction can operate with MOVX and MOVY in parallel.

3.5.13 Operand Conflict

When an identical destination operand is specified with multiple parallel instructions, data conflict occurs. Table 3.34 shows the correspondence between each operand and registers.

Table 3.34 Correspondence between Operands and Registers

		X-Memory Load Instructions			Y-Memory Load Instructions			6-Operand ALU Instructions			3-Operand Multiply Instructions			3-Operand ALU Instructions		
		Ax	Ix	Dx	Ay	Iy	Dy	Sx	Sy	Du	Se	Sf	Dg	Sx	Sy	Dz
DSP Registers	A0							*1		*2			*2	*1		*1
	A1							*1		*2	*1	*1	*2	*1		*1
	M0								*1				*1		*1	*1
	M1								*1				*1		*1	*1
	X0			*2				*1		*2	*1	*1		*1		*2
	X1			*2				*1			*1			*1		*2
	Y0						*2		*1	*2	*1	*1			*1	*2
	Y1						*2		*1			*1			*1	*2

Notes: 1. Registers available for operands
2. Registers available for operands (when there is operand conflict)

There are three cases of operand conflict problems.

- When ALU and multiply instructions specify the same destination operand (Du and Dg)
- When X-memory load and ALU instructions specify the same destination operand (Dx and Du, or Dz)
- When Y-memory load and ALU instructions specify the same destination operand (Dy and Du, or Dz)

In these cases above, the result is not guaranteed.

3.6 DSP Extended Function Instruction Set

3.6.1 CPU Extended Instruction Set

Table 3.35 DSP Mode Extended System Control Instructions

Instruction	Instruction Code	Operation	Execution States	T Bit
SETRC #imm	10000010iiiiiiii	imm→RC (of SR)	1	—
SETRC Rn	0100nnnn00010100	Rn[11:0] →RC (of SR)	1	—
LDRS @(disp,PC)	10001100dddddddd	(disp x 2 + PC) →RS	1	—
LDRE @(disp,PC)	10001110dddddddd	(disp x 2 + PC) →RE	1	—
STC MOD,Rn	0000nnnn01010010	MOD→Rn	1	—
STC RS,Rn	0000nnnn01100010	RS→Rn	1	—
STC RE,Rn	0000nnnn01110010	RE→Rn	1	—
STS DSR,Rn	0000nnnn01101010	DSR→Rn	1	—
STS A0,Rn	0000nnnn01111010	A0→Rn	1	—
STS X0,Rn	0000nnnn10001010	X0→Rn	1	—
STS X1,Rn	0000nnnn10011010	X1→Rn	1	—
STS Y0,Rn	0000nnnn10101010	Y0→Rn	1	—
STS Y1,Rn	0000nnnn10111010	Y1→Rn	1	—
STS.L DSR,@-Rn	0100nnnn01100010	Rn-4→Rn, DSR→(Rn)	1	—
STS.L A0,@-Rn	0100nnnn01110010	Rn-4→Rn, A0→(Rn)	1	—
STS.L X0,@-Rn	0100nnnn10000010	Rn-4→Rn, X0→(Rn)	1	—
STS.L X1,@-Rn	0100nnnn10010010	Rn-4→Rn, X1→(Rn)	1	—
STS.L Y0,@-Rn	0100nnnn10100010	Rn-4→Rn, Y0→(Rn)	1	—
STS.L Y1,@-Rn	0100nnnn10110010	Rn-4→Rn, Y1→(Rn)	1	—
STC.L MOD,@-Rn	0100nnnn01010011	Rn-4→Rn, MOD→(Rn)	1	—
STC.L RS,@-Rn	0100nnnn01100011	Rn-4→Rn, RS→(Rn)	1	—
STC.L RE,@-Rn	0100nnnn01110011	Rn-4→Rn, RE→(Rn)	1	—
LDS.L @Rn + ,DSR	0100nnnn01100110	(Rn) →DSR, Rn + 4→Rn	1	—
LDS.L @Rn + ,A0	0100nnnn01110110	(Rn) →A0, Rn + 4→Rn	1	—
LDS.L @Rn + ,X0	0100nnnn10000110	(Rn) →X0, Rn + 4→Rn	1	—
LDS.L @Rn + ,X1	0100nnnn10010110	(Rn) →X1, Rn + 4→Rn	1	—
LDS.L @Rn + ,Y0	0100nnnn10100110	(Rn) →Y0, Rn + 4→Rn	1	—

Instruction	Instruction Code	Operation	Execution States	T Bit
LDS.L @Rn + ,Y1	0100nnnn10110110	(Rn) →Y1, Rn + 4→Rn	1	—
LDC.L @Rn + ,MOD	0100nnnn01010111	(Rn) →MOD, Rn + 4→Rn	4	—
LDC.L @Rn + ,RS	0100nnnn01100111	(Rn) →RS, Rn + 4→Rn	4	—
LDC.L @Rn + ,RE	0100nnnn01110111	(Rn) →RE, Rn + 4→Rn	4	—
LDS Rn,DSR	0100nnnn01101010	Rn→DSR	1	—
LDS Rn,A0	0100nnnn01111010	Rn→A0	1	—
LDS Rn,X0	0100nnnn10001010	Rn→X0	1	—
LDS Rn,X1	0100nnnn10011010	Rn→X1	1	—
LDS Rn,Y0	0100nnnn10101010	Rn→Y0	1	—
LDS Rn,Y1	0100nnnn10111010	Rn→Y1	1	—
LDC Rn,MOD	0100nnnn01011110	Rn→MOD	4	—
LDC Rn,RS	0100nnnn01101110	Rn→RS	4	—
LDC Rn,RE	0100nnnn01111110	Rn→RE	4	—

3.6.2 Double-Data Transfer Instruction Set

Table 3.36 Double Data Transfer Instruction

Instruction		Instruction Code	Operation	Execution States	DC
X memory data transfer	NOPX	1111000*0*0*00**	X memory no access	1	—
	MOVX.W @Ax,Dx	111100A*D*0*01**	(Ax) →MSW of Dx, 0 →LSW of Dx	1	—
	MOVX.W @Ax + ,Dx	111100A*D*0*10**	(Ax) →MSW of Dx, 0 → LSW of Dx, Ax + 2 →Ax	1	—
	MOVX.W @Ax + Ix,Dx	111100A*D*0*11**	(Ax) → MSW of Dx, 0 →LSW of Dx, Ax + Ix →Ax	1	—
	MOVX.W Da,@Ax	111100A*D*1*01**	MSW of Da →(Ax)	1	—
	MOVX.W Da,@Ax +	111100A*D*1*10**	MSW of Da →(Ax), Ax + 2 →Ax	1	—
	MOVX.W Da,@Ax + Ix	111100A*D*1*11**	MSW of Da →(Ax), Ax + Ix →Ax	1	—
Y memory data transfer	NOPY	111100*0*0*0**00	Y memory no access	1	—
	MOVY.W @Ay,Dy	111100*A*D*0**01	(Ay) →MSW of Dy, 0 →LSW of Dy	1	—
	MOVY.W @Ay + ,Dy	111100*A*D*0**10	(Ay) → MSW of Dy, 0 →LSW of Dy, Ay + 2 →Ay	1	—
	MOVY.W @Ay + Iy,Dy	111100*A*D*0**11	(Ay) →MSW of Dy, 0 →LSW of Dy, Ay + Iy →Ay	1	—
	MOVY.W Da,@Ay	111100*A*D*1**01	MSW of Da →(Ay)	1	—
	MOVY.W Da,@Ay +	111100*A*D*1**10	MSW of Da →(Ay), Ay + 2 ->Ay	1	—
	MOVY.W Da,@Ay + Iy	111100*A*D*1**11	MSW of Da →(Ay), Ay + Iy →Ay	1	—

3.6.3 Single-Data Transfer Instruction Set

Table 3.37 Single Data Transfer Instructions

Instruction	Instruction Code	Operation	Execution States	DC
MOVS.W @-As,Ds	111101AADDDD0000	As-2 →As, (As) → MSW of Ds, 0 -> LSW of Ds	1	—
MOVS.W @As,Ds	111101AADDDD0100	(As) →MSW of Ds, 0 →LSW of Ds	1	—
MOVS.W @As + ,Ds	111101AADDDD1000	(As) →MSW of Ds, 0 →LSW of Ds, As + 2 →As	1	—
MOVS.W @As + Ix,Ds	111101AADDDD1100	(Asc) →MSW of Ds, 0 →LSW of Ds, As + Ix ->As	1	—
MOVS.W Ds,@-As*	111101AADDDD0001	As-2 → MSW of Ds, As→(As)	1	—
MOVS.W Ds,@As*	111101AADDDD0101	MSW of Ds→(As)	1	—
MOVS.W Ds,@As + *	111101AADDDD1001	MSW of Ds →(As), As + 2 →As	1	—
MOVS.W Ds,@As + Ix*	111101AADDDD1101	MSW of Ds →(As), As + Ix →As	1	—
MOVS.L @-As,Ds	111101AADDDD0010	As-4 →As, (As) →Ds	1	—
MOVS.L @As,Ds	111101AADDDD0110	(As) →Ds	1	—
MOVS.L @As + ,Ds	111101AADDDD1010	(As) →Ds, As + 4 →As	1	—
MOVS.L @As + Ix,Ds	111101AADDDD1110	(As) →Ds, As + Ix →As	1	—
MOVS.L Ds,@-As	111101AADDDD0011	As-4 →As, Ds →(As)	1	—
MOVS.L Ds,@As	111101AADDDD0111	Ds →(As)	1	—
MOVS.L Ds,@As +	111101AADDDD1011	Ds →(As), As + 4 V→As	1	—
MOVS.L Ds,@As + Ix	111101AADDDD1111	Ds →(As), As + Ix →As	1	—

Note: * If guard bit registers A0G and A1G are specified in source operand Ds, the data is output to the LDB[7:0] bus and the sign bit is copied into the upper bits, [31:8].

The correspondence between DSP data transfer operands and registers is shown in table 3.38.

Table 3.38 Correspondence between DSP Data Transfer Operands and Registers

Register		Ax	Ix	Dx	Ay	Iy	Dy	Da	As	Ds
CPU	R0	—	—	—	—	—	—	—	—	—
register	R1	—	—	—	—	—	—	—	—	—
	R2 (As2)	—	—	—	—	—	—	—	Yes	—
	R3 (As3)	—	—	—	—	—	—	—	Yes	—
	R4 (Ax0, As0)	Yes	—	—	—	—	—	—	Yes	—
	R5 (Ax1, As1)	Yes	—	—	—	—	—	—	Yes	—
	R6 (Ay0)	—	—	—	Yes	—	—	—	—	—
	R7 (Ay1)	—	—	—	Yes	—	—	—	—	—
	R8 (Ix)	—	Yes	—	—	—	—	—	—	—
	R9 (Iy)	—		—	—	Yes	—	—	—	—
DSP	A0	—	—	—	—	—	—	Yes	—	Yes
register	A1	—	—	—	—	—	—	Yes	—	Yes
	M0	—	—	—	—	—	—	—	—	Yes
	M1	—	—	—	—	—	—	—	—	Yes
	X0	—	—	Yes	—	—	—	—	—	Yes
	X1	—	—	Yes	—	—	—	—	—	Yes
	Y0	—	—	—	—	—	Yes	—	—	Yes
	Y1	—	—	—	—	—	Yes	—	—	Yes
	A0G	—	—	—	—	—	—	—	—	Yes
	A1G	—	—	—	—	—	—	—	—	Yes

3.6.4 DSP Data Operation Instruction Set

Table 3.39 DSP Data Operation Instructions

Instruction	Instruction Code	Operation	Execution States	DC
PMULS Se,Sf,Dg	111110**********	Se*Sf→Dg (Signed)	1	—
	0100eeff0000gg00			
PADD Sx,Sy,Du	111110**********	Sx + Sy→Du Se*Sf→Dg (Signed)	1	*
PMULS Se,Sf,Dg	0111eeffxxyygguu			
PSUB Sx,Sy,Du	111110**********	Sy-Sy→Du Se*Sf→Dg (Signed)	1	*
PMULS Se,Sf,Dg	0110eeffxxyygguu			
PADD Sx,Sy,Dz	111110**********	Sx + Sy→Dz	1	*
	10110001xxyyzzzz			
DCT PADD Sx,Sy,Dz	111110**********	If DC=1, Sx + Sy→Dz If DC=0, nop	1	—
	10110010xxyyzzzz			
DCF PADD Sx,Sy,Dz	111110**********	If DC=0, Sx + Sy→Dz If DC=1, nop	1	—
	10110011xxyyzzzz			
PSUB Sx,Sy,Dz	111110**********	Sx-Sy→Dz	1	*
	10100001xxyyzzzz			
DCT PSUB Sx,Sy,Dz	111110**********	If DC=1, Sx-Sy→Dz If DC=0, nop	1	—
	10100010xxyyzzzz			
DCF PSUB Sx,Sy,Dz	111110**********	If DC=0, Sx-Sy→Dz If DC=1, nop	1	—
	10100011xxyyzzzz			
PSHA Sx,Sy,Dz	111110**********	If Sy>=0, Sx<<Sy→Dz (arithmetic shift)	1	*
	10010001xxyyzzzz	If Sy<0, Sx>>Sy→Dz		
DCT PSHA Sx,Sy,Dz	111110**********	If DC=1 & Sy>=0, Sx<<Sy→Dz (arithmetic shift)	1	—
	10010010xxyyzzzz	If DC=1 & Sy<0, Sx>>Sy→Dz If DC=0, nop		

Instruction	Instruction Code	Operation	Execution States	DC
DCF PSHA Sx,Sy,Dz	111110********** 10010011xxyyzzzz	If DC=0 & Sy>=0, Sx<<Sy→Dz (arithmetic shift) If DC=0 & Sy<0, Sx>>Sy→Dz If DC=1, nop	1	—
PSHL Sx,Sy,Dz	111110********** 10000001xxyyzzzz	If Sy>=0, Sx<<Sy→Dz (logical shift) If Sy<0, Sx>>Sy→Dz	1	*
DCT PSHL Sx,Sy,Dz	111110********** 10000010xxyyzzzz	If DC=1 & Sy>=0, Sx<<Sy→Dz (logical shift) If DC=1 & Sy<0, Sx>>Sy→Dz If DC=0, nop	1	—
DCF PSHL Sx,Sy,Dz	111110********** 10000011xxyyzzzz	If DC=0 & Sy>=0, Sx<<Sy→Dz (logical shift) If DC=0 & Sy<0, Sx>>Sy→Dz If DC=1, nop	1	—
PCOPY Sx,Dz	111110********** 11011001xx00zzzz	Sx→Dz	1	*
PCOPY Sy,Dz	111110********** 1111100100yyzzzz	Sy→Dz	1	*
DCT PCOPY Sx,Dz	111110********** 11011010xx00zzzz	If DC=1, Sx→Dz If DC=0, nop	1	—
DCT PCOPY Sy,Dz	111110********** 1111101000yyzzzz	If DC=1, Sy→Dz If DC=0, nop	1	—
DCF PCOPY Sx,Dz	111110********** 11011011xx00zzzz	If DC=0, Sx→Dz If DC=1, nop	1	—
DCF PCOPY Sy,Dz	111110********** 1111101100yyzzzz	If DC=0, Sy→Dz If DC=1, nop	1	—
PDMSB Sx,Dz	111110********** 10011101xx00zzzz	Sx→Dz normalization count shift value	1	*
PDMSB Sy,Dz	111110********** 1011110100yyzzzz	Sy→Dz normalization count shift value	1	*
DCT PDMSB Sx,Dz	111110********** 10011110xx00zzzz	If DC=1, normalization count shift value Sx→Dz If DC=0, nop	1	—
DCT PDMSB Sy,Dz	111110********** 1011111000yyzzzz	If DC=1, normalization count shift value Sy→Dz If DC=0, nop	1	—
DCF PDMSB Sx,Dz	111110********** 10011111xx00zzzz	If DC=0, normalization count shift value Sx→Dz If DC=1, nop	1	—

Instruction	Instruction Code	Operation	Execution States	DC
DCF PDMSB Sy,Dz	111110********** 1011111100yyzzzz	If DC=0, normalization count shift value Sy→Dz If DC=1, nop	1	—
PINC Sx,Dz	111110********** 10011001xx00zzzz	MSW of Sx+ 1→Dz	1	*
PINC Sy,Dz	111110********** 1011100100yyzzzz	MSW of Sy+ 1→Dz	1	*
DCT PINC Sx,Dz	111110********** 10011010xx00zzzz	If DC=1, MSW of Sx+ 1→Dz If DC=0, nop	1	—
DCT PINC Sy,Dz	111110********** 1011101000yyzzzz	If DC=1, MSW of Sy+ 1→Dz If DC=0, nop	1	—
DCF PINC Sx,Dz	111110********** 10011011xx00zzzz	If DC=0, MSW of Sx + 1→Dz If DC=1, nop	1	—
DCF PINC Sy,Dz	111110********** 1011101100yyzzzz	If DC=0, MSW of Sy+ 1→Dz If DC=1, nop	1	—
PNEG Sx,Dz	111110********** 11001001xx00zzzz	0-Sx→Dz	1	*
PNEG Sy,Dz	111110********** 1110100100yyzzzz	0-Sy→Dz	1	*
DCT PNEG Sx,Dz	111110********** 11001010xx00zzzz	If DC=1, 0-Sx→Dz If DC=0, nop	1	—
DCT PNEG Sy,Dz	111110********** 1110101000yyzzzz	If DC=1, 0-Sy→Dz If DC=0, nop	1	—
DCF PNEG Sx,Dz	111110********** 11001011xx00zzzz	If DC=0, 0-Sx→Dz If DC=1, nop	1	—
DCF PNEG Sy,Dz	111110********** 1110101100yyzzzz	If DC=0, 0-Sy→Dz If DC=1, nop	1	—
POR Sx,Sy,Dz	111110********** 10110101xxyyzzzz	Sx | Sy→Dz	1	*
DCT POR Sx,Sy,Dz	111110********** 10110110xxyyzzzz	If DC=1, Sx | Sy→Dz If DC=0, nop	1	—
DCF POR Sx,Sy,Dz	111110********** 10110111xxyyzzzz	If DC=0, Sx | Sy→Dz If DC=1, nop	1	—

Instruction	Instruction Code	Operation	Execution States	DC
PAND Sx,Sy,Dz	111110********** 10010101xxyyzzzz	Sx & Sy→Dz	1	*
DCT PAND Sx,Sy,Dz	111110********** 10010110xxyyzzzz	If DC=1, Sx & Sy→Dz If DC=0, nop	1	—
DCF PAND Sx,Sy,Dz	111110********** 10010111xxyyzzzz	If DC=0, Sx & Sy→Dz If DC=1, nop	1	—
PXOR Sx,Sy,Dz	111110********** 10100101xxyyzzzz	Sx ^ Sy→Dz	1	*
DCT PXOR Sx,Sy,Dz	111110********** 10100110xxyyzzzz	If DC=1, Sx ^ Sy→Dz If DC=0, nop	1	—
DCF PXOR Sx,Sy,Dz	111110********** 10100111xxyyzzzz	If DC=0, Sx ^ Sy→Dz If DC=0, nop	1	—
PDEC Sx,Dz	111110********** 10001001xx00zzzz	Sx [39:16]-1→Dz	1	*
DCT PDEC Sx,Dz	111110********** 10001010xx00zzzz	If DC=1, Sx [39:16]-1→Dz If DC=0, nop	1	—
DCF PDEC Sx,Dz	111110********** 10001011xx00zzzz	If DC=0, Sx [39:16]-1→Dz If DC=1, nop	1	—
PDEC Sy,Dz	111110********** 1010100100yyzzzz	Sy [31:16]-1→Dz	1	*
DCT PDEC Sy,Dz	111110********** 1010101000yyzzzz	If DC=1, Sy [31:16]-1→Dz If DC=0, nop	1	—
DCF PDEC Sy,Dz	111110********** 1010101100yyzzzz	If DC=0, Sy [31:16]-1→Dz If DC=1, nop	1	—
PCLR Dz	111110********** 1000110100000zzzz	h'00000000→Dz	1	*
DCT PCLR Dz	111110********** 1000111000000zzzz	If DC=1, h'00000000→Dz If DC=0, nop	1	—
DCF PCLR Dz	111110********** 1000111100000zzzz	If DC=0, h'00000000→Dz If DC=1, nop	1	—

Instruction	Instruction Code	Operation	Execution States	DC
PSHA #imm,Dz	111110**********	If imm>=0, Dz<<imm→Dz (arithmetic shift)	1	*
	00010iiiiiiizzzz	If imm<0, Dz>>imm→Dz		
PSHL #imm,Dz	111110**********	If imm>=0, Dz<<imm→Dz (logical shift)	1	*
	00000iiiiiiizzzz	If imm<0, Dz>>imm→Dz		
PSTS MACH,Dz	111110**********	MACH→Dz	1	—
	1100110100000zzzz			
DCT PSTS MACH,Dz	111110**********	If DC=1, MACH→Dz	1	—
	1100111000000zzzz	If DC = 0, nop		
DCF PSTS MACH,Dz	111110**********	If DC=0, MACH→Dz	1	—
	1100111100000zzzz	If DC = 1, nop		
PSTS MACL,Dz	111110**********	MACL→Dz	1	—
	1101110100000zzzz			
DCT PSTS MACL,Dz	111110**********	If DC=1, MACL→Dz	1	—
	1101111000000zzzz	If DC = 0, nop		
DCF PSTS MACL,Dz	111110**********	If DC=0, MACL→Dz	1	—
	1101111100000zzzz	If DC = 1, nop		
PLDS Dz,MACH	111110**********	Dz→MACH	1	–
	1110110100000zzzz			
DCT PLDS Dz,MACH	111110**********	If DC=1, Dz→MACH	1	—
	1110111000000zzzz	If DC = 0, nop		
DCF PLDS Dz,MACH	111110**********	If DC=0, Dz→MACH	1	—
	1110111100000zzzz	If DC = 1, nop		
PLDS Dz,MACL	111110**********	Dz→MACL	1	—
	1111110100000zzzz			
DCT PLDS Dz,MACL	111110**********	If DC=1, Dz→MACL	1	—
	1111111000000zzzz	If DC = 0, nop		
DCF PLDS Dz,MACL	111110**********	If DC=0, Dz→MACL	1	—
	1111111100000zzzz	If DC = 1, nop		
PADDC Sx,Sy,Dz	111110**********	Sx + Sy + DC→Dz Carry→DC	1	Carry
	10110000xxyyzzzz			

Instruction	Instruction Code	Operation	Execution States	DC
PSUBC Sx,Sy, Dz	111110**********	Sx-Sy-DC→Dz Borrow→DC	1	Borrow
	10100000xxyyzzzz			
PCMP Sx,Sy	111110**********	Sx-Sy→DC update*	1	*
	10000100xxyy0000			
PABS Sx,Dz	111110**********	If Sx<0, 0-Sx→Dz If Sx>=0, Sx→Dz	1	*
	10001000xx00zzzz			
PABS Sy,Dz	111110**********	If Sy<0, 0-Sy→Dz If Sy>=0, Sy→Dz	1	*
	1010100000yyzzzz			
PRND Sx,Dz	111110**********	Sx + h'00008000→Dz LSW of Dz→h'0000	1	*
	10011000xx00zzzz			
PRND Sy,Dz	111110**********	Sy + h'00008000→Dz LSW of Dz→h'0000	1	*
	1011100000yyzzzz			

Notes * See table 3.19.

In the instruction column, the asterisks in upper line and the lower line indicate A field and B field, respectively.

3.6.5 Operation Code Map in DSP Mode

Table 3.40 shows the operation code map including an instruction codes extended in the DSP mode.

Table 3.40 Operation Code Map

Instruction Code				Fx: 0000	Fx: 0001	Fx: 0010	Fx: 0011 to 1111
MSB			LSB	MD: 00	MD: 01	MD: 10	MD: 11
0000	Rn	Fx	0000				
0000	Rn	Fx	0001				
0000	Rn	00MD	0010	STC SR, Rn	STC GBR, Rn	STC VBR, Rn	STC SSR, Rn
0000	Rn	01MD	0010	STC SPC, Rn	STC MOD, Rn	STC Rs, Rn	STC RE, Rn
0000	Rn	10MD	0010	STC R0_BANK, Rn	STC R1_BANK, Rn	STC R2_BANK, Rn	STC R3_BANK, Rn
0000	Rn	11MD	0010	STC R4_BANK, Rn	STC R5_BANK, Rn	STC R6_BANK, Rn	STC R7_BANK, Rn
0000	Rm	00MD	0011	BSRF Rm		BRAF Rm	
0000	Rm	10MD	0011	PREF @Rm			
0000	Rn	Rm	01MD	MOV.B Rm, @(R0, Rn)	MOV.W Rm, @(R0, Rn)	MOV.L Rm,@(R0, Rn)	MUL.L Rm, Rn
0000	0000	00MD	1000	CLRT	SETT	CLRMAC	LDTLB
0000	0000	01MD	1000	CLRS	SETS		
0000	0000	10MD	1000				
0000	0000	11MD	1000				
0000	0000	Fx	1001	NOP	DIV0U		
0000	0000	Fx	1010				
0000	0000	Fx	1011	RTS	SLEEP	RTE	
0000	Rn	Fx	1000				
0000	Rn	Fx	1001			MOVT Rn	
0000	Rn	00MD	1010	STS MACH, Rn	STS MACL, Rn	STS PR, Rn	
0000	Rn	01MD	1010			STS DSR, Rn	STS A0, Rn
0000	Rn	10MD	1010	STS X0, Rn	STS X1, Rn	STS Y0, Rn	STS Y1, Rn
0000	Rn	Fx	1011				

Instruction Code				Fx: 0000	Fx: 0001	Fx: 0010	Fx: 0011 to 1111
MSB			**LSB**	**MD: 00**	**MD: 01**	**MD: 10**	**MD: 11**
0000	Rn	Rm	11MD	MOV. B @(R0, Rm), Rn	MOV.W @(R0, Rm), Rn	MOV.L @(R0, Rm), Rn	MAC.L @Rm+,@Rn+
0001	Rn	Rm	disp	MOV.L Rm, @(disp:4, Rn)			
0010	Rn	Rm	00MD	MOV.B Rm, @Rn	MOV.W Rm, @Rn	MOV.L Rm, @Rn	
0010	Rn	Rm	01MD	MOV.B Rm, @–Rn	MOV.W Rm, @–Rn	MOV.L Rm, @–Rn	DIV0S Rm, Rn
0010	Rn	Rm	10MD	TST Rm, Rn	AND Rm, Rn	XOR Rm, Rn	OR Rm, Rn
0010	Rn	Rm	11MD	CMP/STR Rm, Rn	XTRCT Rm, Rn	MULU.W Rm, Rn	MULSW Rm, Rn
0011	Rn	Rm	00MD	CMP/EQ Rm, Rn		CMP/HS Rm, Rn	CMP/GE Rm, Rn
0011	Rn	Rm	01MD	DIV1 Rm, Rn	DMULU.L Rm,Rn	CMP/HI Rm, Rn	CMP/GT Rm, Rn
0011	Rn	Rm	10MD	SUB Rm, Rn		SUBC Rm, Rn	SUBV Rm, Rn
0011	Rn	Rm	11MD	ADD Rm, Rn	DMULS.L Rm,Rn	ADDC Rm, Rn	ADDV Rm, Rn
0100	Rn	Fx	0000	SHLL Rn	DT Rn	SHAL Rn	
0100	Rn	Fx	0001	SHLR Rn	CMP/PZ Rn	SHAR Rn	
0100	Rn	Fx	0010	STS.L MACH, @–Rn	STS.L MACL, @–Rn	STS.L PR, @–Rn	
0100	Rn	00MD	0011	STC.L SR, @–Rn	STC.L GBR, @–Rn	STC.L VBR, @–Rn	STC.L SSR, @–Rn
0100	Rn	01MD	0011	STC.L SPC, @–Rn	STC.L MOD, @–Rn	STC.L RS, @–Rn	STC.L RE, @–Rn
0100	Rn	10MD	0011	STC.L R0_BANK, @–Rn	STC.L R1_BANK, @–Rn	STC.L R2_BANK, @–Rn	STC.L R3_BANK, @–Rn
0100	Rn	11MD	0011	STC.L R4_BANK, @–Rn	STC.L R5_BANK, @–Rn	STC.L R6_BANK, @–Rn	STC.L R7_BANK, @–Rn
0100	Rn	Fx	0100	ROTL Rn	SETRC Rn	ROTCL Rn	
0100	Rn	Fx	0101	ROTR Rn	CMP/PL Rn	ROTCR Rn	
0100	Rm	00MD	0110	LDS.L @Rm+, MACH	LDS.L @Rm+, MACL	LDS.L @Rm+, PR	
0100	Rm	01MD	0110			LDS.L @Rm, DSR	LDS.L @Rm, A0
0100	Rm	10MD	0110	LDS.L @Rm, X0	LDS.L @Rm, X1	LDS.L @Rm, Y0	LDS.L @Rm, Y1

Instruction Code				Fx: 0000	Fx: 0001	Fx: 0010	Fx: 0011 to 1111
MSB			LSB	MD: 00	MD: 01	MD: 10	MD: 11
0100	Rm	00MD	0111	LDC.L @Rm+, SR	LDC.L @Rm+, GBR	LDC.L @Rm+, VBR	LDC.L @Rm+, SSR
0100	Rm	01MD	0111	LDC.L @Rm+, SPC	LDC.L @Rm+, MOD	LDC.L @Rm+, RS	LDC.L @Rm+, RE
0100	Rm	10MD	0111	LDC.L @Rm+, R0_BANK	LDC.L @Rm+, R1_BANK	LDC.L @Rm+, R2_BANK	LDC.L @Rm+, R3_BANK
0100	Rm	11MD	0111	LDC.L @Rm+, R4_BANK	LDC.L @Rm+, R5_BANK	LDC.L @Rm+, R6_BANK	LDC.L @Rm+, R7_BANK
0100	Rn	Fx	1000	SHLL2 Rn	SHLL8 Rn	SHLL16 Rn	
0100	Rn	Fx	1001	SHLR2 Rn	SHLR8 Rn	SHLR16 Rn	
0100	Rm	00MD	1010	LDS Rm, MACH	LDS Rm, MACL	LDS Rm, PR	
0100	Rm	01MD	1010			LDS Rm, DSR	LDS Rm, A0
0100	Rm	10MD	1010	LDS Rm, X0	LDS Rm, X1	LDS Rm, Y0	LDS Rm, Y1
0100	Rm/Rn	Fx	1011	JSR @Rm	TAS.B @Rn	JMP @Rm	
0100	Rn	Rm	1100	SHAD Rm, Rn			
0100	Rn	Rm	1101	SHLD Rm, Rn			
0100	Rm	00MD	1110	LDC Rm, SR	LDC Rm, GBR	LDC Rm, VBR	LDC Rm, SSR
0100	Rm	01MD	1110	LDC Rm, SPC	LDC Rm, MOD	LDC Rm, RS	LDC Rm, RE
0100	Rm	10MD	1110	LDC Rm, R0_BANK	LDC Rm, R1_BANK	LDC Rm, R2_BANK	LDC Rm, R3_BANK
0100	Rm	11MD	1110	LDC Rm, R4_BANK	LDC Rm, R5_BANK	LDC Rm, R6_BANK	LDC Rm, R7_BANK
0100	Rn	Rm	1111	MAC.W @Rm+, @Rn+			
0101	Rn	Rm	disp	MOV.L (disp:4, Rm), Rn			
0110	Rn	Rm	00MD	MOV.B @Rm, Rn	MOV.W @Rm, Rn	MOV.L @Rm, Rn	MOV Rm, Rn
0110	Rn	Rm	01MD	MOV.B @Rm+, Rn	MOV.W @Rm+, Rn	MOV.L @Rm+, Rn	NOT Rm, Rn
0110	Rn	Rm	10MD	SWAP.B Rm, Rn	SWAP.W Rm, Rn	NEGC Rm, Rn	NEG Rm, Rn
0110	Rn	Rm	11MD	EXTU.B Rm, Rn	EXTU.W Rm, Rn	EXTS.B Rm, Rn	EXTS.W Rm, Rn
0111	Rn	imm		ADD #imm : 8, Rn			
1000	00MD	Rn	disp	MOV.B	MOV.W	SETRC #imm	
		imm		R0, @(disp: 4, Rn)	R0, @(disp: 4, Rn)		

Instruction Code			Fx: 0000	Fx: 0001	Fx: 0010	Fx: 0011 to 1111
MSB		LSB	MD: 00	MD: 01	MD: 10	MD: 11
1000	01MD	Rm disp	MOV.B @(disp:4, Rm), R0	MOV.W @(disp: 4, Rm), R0		
1000	10MD	imm/disp	CMP/EQ #imm:8, R0	BT disp: 8		BF disp: 8
1000	11MD	imm/disp	LDRS @(disp:8,PC)	BT/S disp: 8	LDRE @(disp:8,PC)	BF/S disp: 8
1001	Rn	disp	MOV.W (disp : 8, PC), Rn			
1010	disp		BRA disp : 12			
1011	disp		BSR disp: 12			
1100	00MD	imm/disp	MOV.B R0, @(disp: 8, GBR)	MOV.W R0, @(disp: 8, GBR)	MOV.L R0, @(disp: 8, GBR)	TRAPA #imm: 8
1100	01MD	disp	MOV.B @(disp: 8, GBR), R0	MOV.W @(disp: 8, GBR), R0	MOV.L @(disp: 8, GBR), R0	MOVA @(disp: 8, PC), R0
1100	10MD	imm	TST #imm: 8, R0	AND #imm: 8, R0	XOR #imm: 8, R0	OR #imm: 8, R0
1100	11MD	imm	TST.B #imm: 8, @(R0, GBR)	AND.B #imm: 8, @(R0, GBR)	XOR.B #imm: 8, @(R0, GBR)	OR.B #imm: 8, @(R0, GBR)
1101	Rn	disp	MOV.L @(disp: 8, PC), Rn			
1110	Rn	imm	MOV #imm:8, Rn			
1111	00**	********	MOVX.W, MOVY.W Double data transfer instruction			
1111	01**	********	MOVS.W, MOVS.L Single data transfer instruction			
1111	10**	********	MOVX.W, MOVY.W Double data transfer instruction, with DSP parallel operation instruction (32-bit instruction)			
1111	11**	********				

Notes: 1. For details, refer to the SH-3/SH-3E/SH3-DSP Programming Manual.

2. Instructions in the hatched areas are DSP extended instructions. These instructions can be executed only when the DSP bit in the SR register is set to 1.

Section 4 Exception Handling

Exception handling is separate from normal program processing, and is performed by a routine separate from the normal program. For example, if an attempt is made to execute an undefined instruction code or an instruction protected by the CPU processing mode, a control function may be required to return to the source program by executing the appropriate operation or to report an abnormality and carry out end processing. In addition, a function to control processing requested by LSI on-chip modules or an LSI external module to the CPU may also be required.

Transferring control to a user-defined exception handling routine and executing the process to support the above functions are called exception handling. This LSI has two types of exceptions: general exceptions and interrupts. The user can execute the required processing by assigning exception handling routines corresponding to the required exception processing and then return to the source program.

A reset input can terminate the normal program execution and pass control to the reset vector after register initialization. This reset operation can also be regarded as an exception handling. This section describes an overview of the exception handling operation. Here, general exceptions and interrupts are referred to as exception handling. For interrupts, this section describes only the process executed for interrupt requests. For details on how to generate an interrupt request, refer to section 8, Interrupt Controller (INTC).

4.1 Register Descriptions

There are four registers for exception handling. A register with an undefined initial value should be initialized by the software. Refer to section 27, List of Registers, for the register addresses and register states in each process.

- TRAPA exception register (TRA)
- Exception event register (EXPEVT)
- Interrupt event register 2 (INTEVT2)
- Exception address register (TEA)

Figure 4.1 shows the bit configuration of each register.

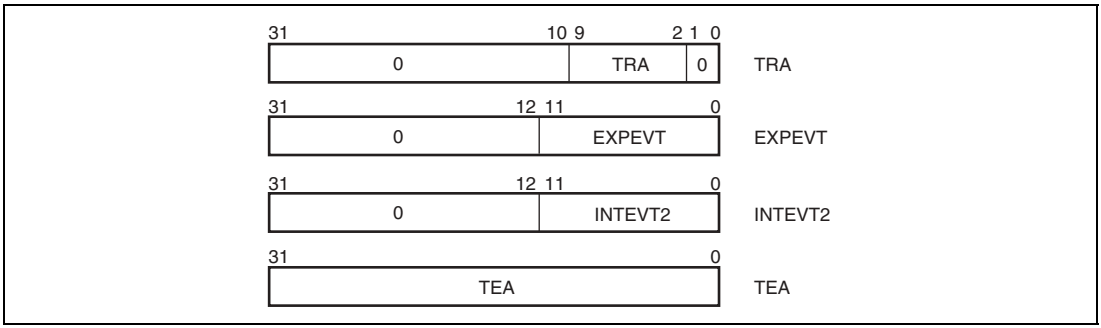


Figure 4.1 Register Bit Configuration

4.1.1 TRAPA Exception Register (TRA)

TRA is assigned to address H'FFFFFFD0 and consists of the 8-bit immediate data (imm) of the TRAPA instruction. TRA is automatically specified by the hardware when the TRAPA instruction is executed. Only bits 9 to 2 of the TRA can be re-written using the software.

Bit	Bit Name	Initial Value	R/W	Description
31 to 10	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
9 to 2	TRA	—	R/W	8-bit Immediate Data
1	—	All 0	R	Reserved
0				These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

4.1.2 Exception Event Register (EXPEVT)

EXPEVT is assigned to address H'FFFFFFD4 and consists of a 12-bit exception code. Exception codes to be specified in EXPEVT are those for resets and general exceptions. These exception codes are automatically specified by the hardware when an exception occurs. Only bits 11 to 0 of EXPEVT can be re-written using the software.

Bit	Bit Name	Initial Value	R/W	Description
31 to 12	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
11 to 0	EXPEVT	*	R/W	12-bit Exception Code

Note: Initialized to H'000 at power-on reset and H'020 at manual reset.

4.1.3 Interrupt Event Register 2 (INTEVT2)

INTEVT2 is assigned to address H'A4000000 and consists of a 12-bit exception code. Exception codes to be specified in INTEVT2 are those for interrupt requests. These exception codes are automatically specified by the hardware when an exception occurs. INTEVT2 cannot be modified using the software.

Bit	Bit Name	Initial Value	R/W	Description
31 to 12	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
11 to 0	INTEVT2	—	R	12-bit Exception Code

4.1.4 Exception Address Register (TEA)

TEA is assigned to address H'FFFFFFFC and stores the virtual address for an exception occurrence when an exception related to memory accesses occurs. TEA can be modified using the software.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	TEA	All 0	R/W	Virtual Address For Exception

4.2 Exception Handling Function

4.2.1 Exception Handling Flow

In exception handling, the contents of the program counter (PC) and status register (SR) are saved in the saved program counter (SPC) and saved status register (SSR), respectively, and execution of the exception handler is invoked from a vector address. The return from exception handler (RTE) instruction is issued by the exception handler routine on completion of the routine, restoring the contents of PC and SR to return to the processor state at the point of interruption and the address where the exception occurred.

A basic exception handling sequence consists of the following operations. If an exception occurs and the CPU accepts it, operations 1 to 8 are executed.

1. The contents of PC are saved in SPC.
2. The contents of SR are saved in SSR.
3. The block (BL) bit in SR is set to 1, masking any subsequent exceptions.
4. The mode (MD) bit in SR is set to 1 to place the privileged mode.
5. The register bank (RB) bit in SR is set to 1.
6. When an exception source is a general exception, an exception code identifying the exception event is written to EXPEVT. When an exception source is an interrupt, an exception code is written to INTEVT2.
7. If a TRAPA instruction is executed, an 8-bit immediate data specified by the TRAPA instruction is set to TRA. For an exception related to memory accesses, the virtual address where the exception occurred is written to TEA.
8. Instruction execution jumps to the designed exception vector address to invoke the handler routine.

The above operations from 1 to 8 are executed in sequence. During these operations, no other exceptions may be accepted unless multiple exception acceptance is enabled.

In an exception handling routine for a general exception, the appropriate exception handling must be executed based on an exception source determined by the EXPEVT. In an interrupt exception handling routine, the appropriate exception handling must be executed based on an exception source determined by the INTEVT2. After the exception handling routine has been completed, program execution can be resumed by executing an RTE instruction. The RTE instruction causes the following operations to be executed.

1. The contents of the SSR are restored into the SR to return to the processing state in effect before the exception handling took place.
2. A delay slot instruction of the RTE instruction is executed.*
3. Control is passed to the address stored in the SPC.

The above operations from 1 to 3 are executed in sequence. During these operations, no other exceptions may be accepted. By changing the SPC and SSR before executing the RTE instruction, a status different from that in effect before the exception handling can also be specified.

Note: * For details on the CPU processing mode in which RTE delay slot instructions are executed, please refer to section 4.5, Usage Notes.

4.2.2 Exception Vector Addresses

A vector address for general exceptions is determined by adding a vector offset to a vector base address. The vector offset for general exceptions is H'00000100. The vector offset for interrupts is H'00000600. The vector base address is loaded into the vector base register (VBR) using the software. The vector base address should reside in the virtual address (P1 or P2).

4.2.3 Exception Codes

The exception codes are written to bits 11 to 0 of the EXPEVT register (for reset or general exceptions) or the INTEVT2 register (for interrupt requests) to identify each specific exception event. See section 8, Interrupt Controller (INTC), for details of the exception codes for interrupt requests. Table 4.1 lists exception codes for resets and general exceptions.

4.2.4 Exception Request and BL Bit (Multiple Exception Prevention)

The BL bit in SR is set to 1 when a reset or exception is accepted. While the BL bit is set to 1, acceptance of general exceptions is restricted as described below, making it possible to effectively prevent multiple exceptions from being accepted.

If the BL bit is set to 1, an interrupt request is not accepted and is retained. The interrupt request is accepted when the BL bit is cleared to 0. If the CPU is in power-down mode, an interrupt is accepted even if the BL bit is set to 1 and the CPU returns from power-down mode.

A DMA address error is not accepted and is retained if the BL bit is set to 1 and accepted when the BL bit is cleared to 0. User break requests generated while the BL bit is set are ignored and are not retained. Accordingly, user breaks are not accepted even if the BL bit is cleared to 0.

If a general exception other than a DMA address error or user break occurs while the BL bit is set to 1, the CPU enters a state similar to that in effect immediately after a reset, and passes control to the reset vector (H'A0000000) (multiple exception). In this case, unlike a normal reset, modules

other than the CPU are not initialized, the contents of EXPEVT, SPC, and SSR are undefined, and this status is not detected by an external device.

To enable acceptance of multiple exceptions, the contents of SPC and SSR must be saved while the BL bit is set to 1 after an exception has been accepted, and then the BL bit must be cleared to 0. Before restoring the SPC and SSR, the BL bit must be set to 1.

4.2.5 Exception Source Acceptance Timing and Priority

Exception Request of Instruction Synchronous Type and Instruction Asynchronous Type: Resets and interrupts are requested asynchronously regardless of the program flow. In general exceptions, a DMA address error and a user break under the specific condition are also requested asynchronously. The user cannot expect on which instruction an exception is requested. For general exceptions other than a DMA address error and a user break under a specific condition, each general exception corresponds to a specific instruction.

Re-execution Type and Processing-completion Type Exceptions: All exceptions are classified into two types: a re-execution type and a processing-completion type. If a re-execution type exception is accepted, the current instruction executed when the exception is accepted is terminated and the instruction address is saved to the SPC. After returning from the exception processing, program execution resumes from the instruction where the exception was accepted. In a processing-completion type exception, the current instruction executed when the exception is accepted is completed, the next instruction address is saved to the SPC, and then the exception processing is executed.

During a delayed branch instruction and delay slot, the following operations are executed. A re-execution type exception detected in a delay slot is accepted before executing the delayed branch instruction. A processing-completion type exception detected in a delayed branch instruction or a delay slot is accepted when the delayed branch instruction has been executed. In this case, the acceptance of delayed branch instruction or a delay slot precedes the execution of the branch destination instruction. In the above description, a delay slot indicates an instruction following an unconditional delayed branch instruction or an instruction following a conditional delayed branch instruction whose branch condition is satisfied. If a branch does not occur in a conditional delayed branch, the normal processing is executed.

Acceptance Priority and Test Priority: Acceptance priorities are determined for all exception requests. The priority of resets, general exceptions, and interrupts are determined in this order: a reset is always accepted regardless of the CPU status. Interrupts are accepted only when resets or general exceptions are not requested.

If multiple general exceptions occur simultaneously in the same instruction, the priority is determined as follows.

1. A processing-completion type exception generated at the previous instruction*
2. A user break before instruction execution (re-execution type)
3. An exception related to an instruction fetch (CPU address error)
4. An exception caused by an instruction decode (General illegal instruction exceptions and slot illegal instruction exceptions: re-execution type, unconditional trap: processing-completion type)
5. An exception related to data access (CPU address error)
6. Unconditional trap (processing-completion type)
7. A user break other than one before instruction execution (processing-completion type)
8. DMA address error (processing-completion type)

Note: * If a processing-completion type exception is accepted at an instruction, exception processing starts before the next instruction is executed. This exception processing executed before an exception generated at the next instruction is detected.

Only one exception is accepted at a time. Accepting multiple exceptions sequentially results in all exception requests being processed.

Table 4.1 Exception Event Vectors

Exception Type	Current Instruction	Exception Event	Priority*[1]	Exception Order	Process at BL=1	Excep-tion Code*[5]	Vector Offset
Reset (Instruction asynchro-nous type)	Aborted	Power-on reset	1	1	Reset	H'000	—
		Manual reset	1	2	Reset	H'020	—
General exception events (Instruction synchro-nous type)	Re-executed	User break(before instruction execution)	2	0	Ignored	H'1E0	H'00000100
		CPU address error (instruction access) *[4]	2	1	Reset	H'0E0	H'00000100
		Illegal general instruction exception	2	2	Reset	H'180	H'00000100
		Illegal slot instruction exception	2	2	Reset	H'1A0	H'00000100
		CPU address error (data read/write)*[4]	2	3	Reset	H'0E0/ H'100	H'00000100
	Completed	Unconditional trap (TRAPA instruction)	2	4	Reset	H'160	H'00000100
		User breakpoint (After instruction execution, address)	2	5	Ignored	H'1E0	H'00000100
General exception events (Instruction asynchro-nous type)	Completed	User breakpoint (Data break, I-BUS break)	2	5	Ignored	H'1E0	H'00000100
		DMA address error	2	6	Retained	H'5C0	H'00000100
Interrupt (Instruction asynchro-nous type)	Completed	Nonmaskable interrupt (NMI)	3*[2]	—	Retained	—*[3]	H'00000600
		External hardware interrupt (IRQ interrupt)	4*[2]	—	Retained	—*[3]	H'00000600
		H-UDI interrupt	4*[2]	—	Retained	—*[3]	H'00000600
		On-chip peripheral module interrupt	4*[2]	—	Retained	—*[3]	H'00000600

Notes: 1. Priorities are indicated from high to low, 1 being the highest and 4 the lowest. A reset has the highest priority. An interrupt is accepted only when general exceptions are not requested.

2. For details on priorities in multiple interrupt sources, refer to section 8, Interrupt Controller (INTC).
3. If an interrupt is accepted, the exception event register (EXPEVT) is not changed. The interrupt source code is specified in interrupt event register 2 (INTEVT2). For details, refer to section 8, Interrupt Controller (INTC).
4. If one of these exceptions occurs in a specific part of the repeat loop, a specific code and vector offset are specified.
5. Exception codes H'040, H'060, H'080, H'0A0, H'0C0, H'0D0, H'120, H'140, and H'3E0 are reserved.

4.3 Individual Exception Operations

This section describes the conditions for specific exception handling, and the processor operations. Resets and general exceptions are explained in this section. For details of interrupts, see section 8, Interrupt Controller (INTC)

4.3.1 Resets

Power-On Reset:

- Conditions
  $\overline{\text{RESETP}}$ is low, power-on reset by WDT is requested, H-UDI reset
- Operations
  Set EXPEVT to H'000, initialize VBR and SR, and branch to PC = H'A0000000.*
  The VBR register is cleared to H'00000000 by the initialization. In the SR register, the bits MD, RB, and BL are set to 1, the DSP bit is cleared to 0, and interrupt mask bits (I3 to I0) are set to B'1111.
  Initialize the CPU and on-chip peripheral modules. For details, refer to the register descriptions in the relevant sections.
  Perform a reset by the $\overline{\text{RESETP}}$ pin low at power supply.

Note: * After a power-on reset, though address is H'A0000000, if the $\overline{\text{BOOT_E}}$ pin is asserted low, a branch is made to the ROM area for the boot function and the boot processing starts (see section 17, Boot Function).
For the details of the H_UDI reset, see section 26, User Debugging Interface (H-UDI).

Manual Reset:

- Conditions
  Manual reset by WDT is request
- Operations
  Set EXPEVT to H′020, initialize VBR and SR, and branch to PC = H'A0000000.*
  The VBR register is cleared to H'00000000 by the initialization. In the SR register, the bits MD, RB, and BL are set to 1, the DSP bit is cleared to 0, and interrupt mask bits (I3 to I0) are set to B'1111. Initialize the CPU and on-chip peripheral modules. A register initialized by a power-on reset is different from that is initialized by a manual reset. For details, refer to the register descriptions in the relevant sections.

Note: * After a manual reset, if the $\overline{\text{BOOT_E}}$ pin is asserted low, a branch is made to the ROM area for the boot function and the boot processing starts as in the case of the power-on reset.

4.3.2 General Exceptions

CPU address error:

- Conditions
  - — Instruction is fetched from odd address (4n + 1, 4n + 3)
  - — Word data is accessed from addresses other than word boundaries (4n + 1, 4n + 3)
  - — Longword is accessed from addresses other than longword boundaries (4n + 1, 4n + 2, 4n + 3)
  - — The area ranging from H'80000000 to H'FFFFFFFF in logical space is accessed in user mode
- Types

  Instruction synchronous, re-execution type
- Save address

  Instruction fetch: An instruction address to be fetched when an exception occurred

  Data access: An instruction address where an exception occurs (a delayed branch instruction address if an instruction is assigned to a delay slot)
- Exception code

  An exception occurred during read: H'0E0

  An exception occurred during write: H'100
- Remarks

  The virtual address (32 bits) that caused the exception is set in TEA.

Illegal general instruction exception:

- Conditions
  - — When undefined code not in a delay slot is decoded

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

Note: For details on undefined code, refer to section 2.6.2, Operation Code Map. When an undefined code other than H'FC00 to H'FFFF is decoded, operation cannot be guaranteed.

  - — When a privileged instruction not in a delay slot is decoded in user mode

    Privileged instructions: LDC, STC, RTE, LDTLB, SLEEP; instructions that access GBR with LDC/STC are not privileged instructions.
- Types

  Instruction synchronous, re-execution type
- Save address

  An instruction address where an exception occurs

- Exception code
  H'180
- Remarks
  None

Illegal slot instruction:

- Conditions
  - When undefined code in a delay slot is decoded
    Delayed branch instructions: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT/S, BF/S
  - When a privileged instruction in a delay slot is decoded in user mode
    Privileged instructions: LDC, STC, RTE, LDTLB, SLEEP; instructions that access GBR with LDC/STC are not privileged instructions.
  - When an instruction that rewrites PC in a delay slot is decoded
    Instructions that rewrite PC: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT, BF, BT/S, BF/S, TRAPA, LDC Rm, SR, LDC.L @Rm+, SR
- Types
  Instruction synchronous, re-execution type
- Save address
  A delayed branch instruction address
- Exception code
  H'1A0
- Remarks
  None

Unconditional trap:

- Conditions
  TRAPA instruction executed
- Types
  Instruction synchronous, processing-completion type
- Save address
  An address of an instruction following TRAPA
- Exception code
  H'160
- Remarks
  The exception is a processing-completion type, so PC of the instruction after the TRAPA instruction is saved to SPC. The 8-bit immediate value in the TRAPA instruction is set in TRA[9:2].

User break point trap:

- Conditions

  When a break condition set in the user break controller is satisfied
- Types

  Break (L bus) before instruction execution: Instruction synchronous, re-execution type

  Operand break (L bus): Instruction synchronous, processing-completion type

  Data break (L bus): Instruction asynchronous, processing-completion type

  I bus break: Instruction asynchronous, processing-completion type
- Save address

  Re-execution type: An address of the instruction where a break occurs (a delayed branch instruction address if an instruction is assigned to a delay slot)

  Completion type: An address of the instruction following the instruction where a break occurs (a delayed branch instruction destination address if an instruction is assigned to a delay slot)
- Exception code

  H'1E0
- Remarks

  For details on user break controller, refer to section 25, User Break Controller (UBC).

DMA address error:

- Conditions
  - Word data accessed from addresses other than word boundaries (4n + 1, 4n + 3)
  - Longword accessed from addresses other than longword boundaries (4n + 1, 4n + 2, 4n + 3)
- Types

  Instruction asynchronous, processing-completion type
- Save address

  An address of the instruction following the instruction where a break occurs (a delayed branch instruction destination address if an instruction is assigned to a delay slot)
- Exception code

  H'5C0
- Remarks

  An exception occurs when a DMA transfer is executed while an illegal instruction address described above is specified in the DMAC. Since the DMAC transfer is performed asynchronously with the CPU instruction operation, an exception is also requested asynchronously with the instruction execution. For details on DMAC, refer to section 10, Direct Memory Access Controller (DMAC).

Note: Address error does not occurred in the USB host since the hardware automatically recognizes the address and performs accessing data of enough bit length.

4.4 Exception Processing While DSP Extension Function is Valid

In DSP mode (the DSP bit of SR is set to 1), some exception processing acceptance conditions or exception processing may be changed.

4.4.1 Illegal Instruction Exception and Slot Illegal Instruction Exception

In the DSP mode, a DSP extension instruction can be executed. If a DSP extension instruction is executed when the DSP bit of SR is cleared to 0 (in a mode other than the DSP mode), an illegal instruction exception occurs.

In the DSP mode, STC and LDC instructions for the SR register can be executed even in user mode. (Note, however, that only the RC[11:0], DMX, DMY, and RF[1:0] bits in the DSP extension bits can be changed.)

4.4.2 CPU Address Error

In the DSP mode, a part of the space P2 (Uxy area: H'A5000000 to H'A5FFFFFF) can be accessed in user mode and no CPU address error will occur even if the area is accessed.

4.4.3 Exception in Repeat Control Period

If an exception is requested or an exception is accepted during repeat control, the exception may not be accepted correctly or a program execution may not be returned correctly from exception processing that is different from the normal state. These restrictions may occur from repeat detection instruction to repeat end instruction while the repeat counter is 1 or more. In this section, this period is called the repeat control period.

The following shows program examples where the number of instructions in the repeat loop is 4 or more, 3, 2, and 1, respectively. In this section, a repeat detection instruction and its instruction address are described as RptDtct. The first, second, and third instructions following the repeat detection instruction are described as RptDtct1, RptDtct2, and RptDtct3. In addition, [A], [B], [C1], and [C2] in the following examples indicate instructions where a restriction occurs. Table 4.2 summarizes the instruction positions regarding a repeat loop and restriction types.

Table 4.2 Instruction Positions regarding a repeat loop and Restriction Types

Instruction Position	SPC*[1]	Illegal Instruction*[2]	Interrupt, Break*[3]	CPU Address Error*[4]
[A]				
[B]			Retained	
[C1]		Added	Retained	Instruction/data
[C2]	Illegal	Added	Retained	Instruction/data

Notes: 1. A specific address is specified in the SPC if an exception occurs while SR.RC[11:0] ≥ 2.
2. Some of instructions are turned to be illegal instructions while SR.RC[11:0] ≥ 1.
3. An interrupt, break or DMA address error request is retained while SR.RC[11:0] ≥ 1.
4. A specific exception code is specified while SR.RC[11:0] ≥ 1.

- Example 1: Repeat loop consisting of four instructions

```
            LDRS RptStart  ; [A]
            LDRE RptDtct + 4      ; [A]
            SETRC #4       ; [A]
            instr0         ; [A]
RptStart:   instr1         ; [A][Repeat start instruction]
            ……             ; [A]
            ……             ; [A]
RptDtct:    RptDtct        ; [B] A repeat detection
                                 instruction is an
                               instruction three
                               instructions before a
                             repeat end instruction
            RptDtct1       ; [C1]
            RptDtct2       ; [C2]
RptEnd:     RptDtct3       ; [C2][Repeat end instruction]
            InstrNext      ; [A]
```

- Example 2: Repeat loop consisting of three instructions

```
            LDRS RptDtct + 4      ; [A]
            LDRE RptDtct + 4      ; [A]
            SETRC #4              ; [A]
RptDtct:    RptDtct               ; [B] A repeat detection
                                        instruction is an
                                        instruction prior to a
                                        repeat start instruction
RptStart:   RptDtct1              ; [C1][Repeat start instruction]
            RptDtct2              ; [C2]
RptEnd:     RptDtct3              ; [C2][Repeat end instruction]
            InstrNext             ; [A]
```

- Example 3: Repeat loop consisting of two instructions

```
            LDRS RptDtct + 6      ; [A]
            LDRE RptDtct + 4      ; [A]
            SETRC #4              ; [A]
RptDtct:    RptDtct               ; [B] A repeat detection
                                        instruction is an
                                      instruction prior to a
                                        repeat start instruction
RptStart:   RptDtct1              ; [C1][Repeat start instruction]
RptEnd:     RptDtct2              ; [C2][Repeat end instruction]
            InstrNext             ; [A]
```

- Example 4: Repeat loop consisting of one instruction

```
            LDRS RptDtct + 8      ; [A]
            LDRE RptDtct + 4      ; [A]
            SETRC #4              ; [A]
RptDtct:    RptDtct               ; [B] A repeat detection
                                        instruction is an
                                        instruction prior to a
                                        repeat start instruction
RptStart:
RptEnd:     RptDtct1              ; [C1][Repeat start instruction]==
                                          [Repeat end instruction]
            InstrNext             ; [A]
```

SPC Saved by an Exception in Repeat Control Period: If an exception is accepted in the repeat control period while the repeat counter (RC[11:0]) in the SR register is two or greater, the program counter to be saved may not indicate the value to be returned correctly. To execute the repeat control after returning from an exception processing, the return address must indicate an instruction prior to a repeat detection instruction. Accordingly, if an exception is accepted in repeat control period, an exception other than re-execution type exception by a repeat detection instruction cannot return to the repeat control correctly.

Table 4.3 SPC Value when a Re-Execution Type Exception Occurs in Repeat Control

Instruction Where an Exception Occurs	Number of Instructions in a Repeat Loop			
	1	2	3	4 or Greater
RptDtct	RptDtct	RptDtct	RptDtct	RptDtct
RptDtct1	RptDtct1	RptDtct1	RptDtct1	RptDtct1
RptDtct2	—	RptDtct1	RptDtct1	RS-4
RptDtct3	—	—	RptDtct1	RS-2

Note: The following symbols are used here.

RptDtct: Repeat detection instruction address

RptDtct1: An instruction address one instruction following the repeat detection instruction (In a repeat loop consisting of one to three instructions, RptDtct1 is equal to RptStart.)

RptDtct2: An instruction address two instruction following the repeat detection instruction

RptDtct3: An instruction address three instruction following the repeat detection instruction

RS: Repeat start instruction address

If a re-execution type exception is accepted at an instruction in the hatched areas above, a return address to be saved in the SPC is incorrect. If SR.RC[11:0] is 1 or 0, a correct return address is saved in the SPC.

Illegal Instruction Exception in Repeat Control Period: If one of the following instructions is executed at the address following RptDtct1, a general illegal instruction exception occurs. For details on an address to be saved in the SPC, refer to section 4.4.3, Exception in Repeat Control Period.

- Branch instructions
  BRA, BSR, BT, BF, BT/S, BF/S, BSRF, RTS, BRAF, RTE, JSR, JMP, TRAPA
- Repeat control instructions
  SETRC, LDRS, LDRE
- Load instructions for SR, RS, and RE
  LDC Rn,SR, LDC @Rn+,SR, LDC Rn,RE, LDC @Rn+,RE, LDC Rn,RS, LDC @Rn+, Rs

Note: In a repeat loop consisting of one to three instructions, some restrictions apply to repeat detection instructions and all the remaining instructions. In a repeat loop consisting of four or more instructions, restrictions apply to only the three instructions that include a repeat end instruction.

An Exception Retained in Repeat Control Period: In the repeat control period, an interrupt or some exception will be retained to prevent an exception acceptance at an instruction where returning from the exception cannot be performed correctly. For details, refer to repeat loop program examples (1) to (4). In the examples, exceptions generated at instructions indicated as [B], and [C] ([C1] or [C2]) the following processing is executed.

- Interrupt, DMA address error
  An exception request is not accepted and retained at instructions [B] and [C]. If an instruction indicates as [A] is executed the next time, an exception request is accepted.* As shown in examples (1) to (4), any interrupt or DMA address error cannot be accepted in a repeat loop consisting of four instructions or less.

Note: An interrupt request or a DMA address error exception request is retained in the interrupt controller (INTC) and the direct memory access controller (DMAC) until the CPU can accept a request.

- User break before instruction execution
  A user break before instruction execution is accepted at instruction [B], and an address of instruction [B] is saved in the SPC. This exception cannot be accepted at instruction [C] but the exception request is retained until an instruction [A] or [B] is executed the next time. Then, the exception request is accepted before an instruction [A] or [B] is executed. In this case, an address of instruction [A] or [B] is saved in the SPC.

- User break after instruction execution

  A user break after instruction execution cannot be accepted at instructions [B] and [C] but the exception request is retained until an instruction [A] or [B] is executed the next time. Then, the exception request is accepted before an instruction [A] or [B] is executed. In this case, an address of instruction [A] or [B] is saved in the SPC.

Table 4.4 Restrictions of Exception Acceptance in the Repeat Loop

Exception Type	Instruction [B]	Instruction [C]
Interrupt	Not accepted	Not accepted
DMA address error	Not accepted	Not accepted
User break before instruction execution	Accepted	Not accepted
User break after instruction execution	Not accepted	Not accepted

CPU Address Error in Repeat Control Period: If a CPU address error occurs in the repeat control period, the exception is accepted but an exception code (H'070) indicating the repeat loop period is specified in the EXPEVT. If a CPU address error occurs in instructions following a repeat detection instruction to repeat end instruction, an exception code for instruction access or data access is specified in the EXPEVT.

The SPC is saved according to the description in section 4.4.3, SPC Saved by an Exception in Repeat Control Period.

After the CPU address error exception processing, the repeat control cannot be returned correctly. To execute a repeat loop correctly, care must be taken not to generate a CPU address error in the repeat control period.

Note: In a repeat loop consisting of one to three instructions, some restrictions apply to repeat detection instructions and all the remaining instructions. In a repeat loop consisting of four or more instructions, restrictions apply to only the three instructions that include a repeat end instruction. The restriction occurs when SR.RC[11:0] ≥ 1.

Table 4.5 Instruction Where a Specific Exception Occurs When a Memory Access Exception Occurs in Repeat Control (SR.RC[11:0] ≥ 1)

Instruction Where an Exception Occurs	Number of Instructions in a Repeat Loop			
	1	2	3	4 or Greater
RptDtct				
RptDtct1	Instruction/data access	Instruction/data access	Instruction/data access	Instruction/data access
RptDtct2	—	Instruction/data access	Instruction/data access	Instruction/data access
RptDtct3	—	—	Instruction/data access	Instruction/data access

Note: The following symbols are used here.

RptDtct: Repeat detection instruction address

RptDtct1: An instruction address one instruction following the repeat detection instruction

RptDtct2: An instruction address two instruction following the repeat detection instruction

RptDtct3: An instruction address three instruction following the repeat detection instruction

4.5 Usage Notes

1. An instruction assigned at a delay slot of the RTE instruction is executed after the contents of the SSR is restored into the SR. An acceptance of an exception related to instruction access is determined according to the SR before restore. An acceptance of other exceptions is determined by the SR after restore, processing mode, and BL bit value. A processing-completion type exception is accepted before an instruction at the RTE branch destination address is executed. However, note that the correct operation cannot be guaranteed if a re-execution type exception occurs.
2. In an instruction assigned at a delay slot of the RTE instruction, a user break cannot be accepted.
3. If the MD and BL bits of the SR register are changed by the LDC instruction, an exception is accepted according to the changed SR value from the next instruction.* A processing-completion type exception is accepted after the next instruction is executed. An interrupt and DMA address error in processing-completion type exceptions are accepted before the next instruction is executed.

Note: * If an LDC instruction is executed for the SR, the following instructions are re-fetched and an instruction fetch exception is accepted according to the modified SR value.

Section 5 Cache

5.1 Features

- Capacity: 16 kbytes
- Structure: Instructions/data mixed, 4-way set associative
- Locking: Way 2 and way 3 are lockable
- Line size: 16 bytes
- Number of entries: 256 entries/way
- Write system: Write-back/write-through is selectable for spaces P0, P1, P3, and U0 individually.
  Group 1 (P0, P3, and U0 areas)
  Group 2 (P1 area)
- Replacement method: Least-recently used (LRU) algorithm

5.1.1 Cache Structure

The cache mixes instructions and data and uses a 4-way set associative system. It is composed of four ways (banks), and each of which is divided into an address section and a data section. Each of the address and data sections is divided into 256 entries. The entry data is called a line. Each line consists of 16 bytes (4 bytes × 4). The data capacity per way is 4 kbytes (16 bytes × 256 entries) in the cache as a whole (4 ways). The cache capacity is 16 kbytes as a whole. Figure 5.1 shows the cache structure.

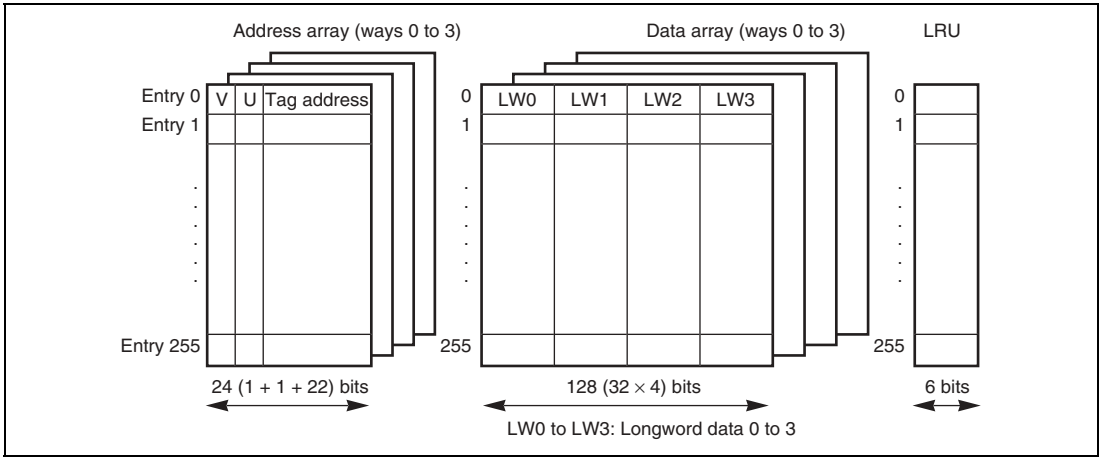


Figure 5.1 Cache Structure

Address Array: The V bit indicates whether the entry data is valid. When the V bit is 1, data is valid; when 0, data is not valid. The U bit indicates whether the entry has been written to in write-back mode. When the U bit is 1, the entry has been written to; when 0, it has not. The tag address holds the physical address used in the external memory access. It is composed of 22 bits (address bits 31 to10 for memory access) used for comparison during cache searches.

In this LSI, the top three of 32 physical address bits are used as shadow bits (see section 9, Bus State Controller (BSC)), and therefore the top three bits of the tag address are cleared to 0.

The V and U bits are initialized to 0 by a power-on reset, but are not initialized by a manual reset. The tag address is not initialized by either a power-on or manual reset.

Data Array: Holds a 16-byte instruction or data. Entries are registered in the cache in line units (16 bytes). The data array is not initialized by a power-on or manual reset.

LRU: With the 4-way set associative system, up to four instructions or data with the same entry address can be registered in the cache. When an entry is registered, LRU shows which of the four ways it is recorded in. There are six LRU bits, controlled by hardware. A least-recently-used (LRU) algorithm is used to select the way.

Six LRU bits indicate the way to be replaced, when a cache miss occurs. Table 5.1 shows the relationship between the LRU bits and the way to be replaced when the cache locking mechanism is disabled. (For the relationship when the cache locking mechanism is enabled, refer to section 5.2.2, Cache Control Register 2.) If a bit pattern other than those listed in table 5.1 is set in the LRU bits by software, the cache will not function correctly. When modifying the LRU bits by software, set one of the patterns listed in table 5.1.

The LRU bits are initialized to B'000000 by a power-on reset, but are not initialized by a manual reset.

Table 5.1 LRU and Way Replacement (when Cache Locking Mechanism is Disabled)

LRU (Bits 5 to 0)	Way to be Replaced
000000, 000100, 010100, 100000, 110000, 110100	3
000001, 000011, 001011, 100001, 101001, 101011	2
000110, 000111, 001111, 010110, 011110, 011111	1
111000, 111001, 111011, 111100, 111110, 111111	0

5.2 Register Descriptions

The cache has the following registers. For details on register addresses and register access size, refer to section 27, List of Registers.

- Cache control register 1 (CCR1)
- Cache control register 2 (CCR2)

5.2.1 Cache Control Register 1 (CCR1)

The cache is enabled or disabled using the CE bit in CCR1. CCR1 also has a CF bit (which invalidates all cache entries), and WT and CB bits (which select either write-through mode or write-back mode). Programs that change the contents of the CCR1 register should be placed in address space that is not cached.

Bit	Bit Name	Initial Value	R/W	Description
31 to 4	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
3	CF	0	R/W	Cache Flush Writing 1 flushes all cache entries (clears the V, U, and LRU bits of all cache entries to 0). This bit is always read as 0. Write-back to external memory is not performed when the cache is flushed.
2	CB	0	R/W	Write-Back Indicates the cache's operating mode for space P1. 0: Write-through mode 1: Write-back mode
1	WT	0	R/W	Write-Through Indicates the cache's operating mode for spaces P0, U0, and P3. 0: Write-back mode 1: Write-through mode
0	CE	0	R/W	Cache Enable Indicates whether the cache function is used. 0: The cache function is not used. 1: The cache function is used.

5.2.2 Cache Control Register 2 (CCR2)

The CCR2 register controls the cache locking mechanism in DSP mode only. The CPU enters DSP mode when the DSP bit (bit 12) in the status register (SR) is set to 1. The cache locking mechanism is disabled in non-DSP mode (DSP bit = 0).

When a prefetch instruction (PREF@Rn) is issued in DSP mode and a cache miss occurs, the line of data pointed to by Rn will be loaded into the cache, according to the setting of bits 9 and 8 (W3LOAD, W3LOCK) and bits 1 and 0 (W2LOAD, W2LOCK in CCR2).

Table 5.2 shows the relationship between the settings of bits and the way that is to be replaced when the cache is missed by a prefetch instruction.

On the other hand, when the cache is hit by a prefetch instruction, new data is not loaded into the cache and the valid entry is held. For example, a prefetch instruction is issued while bits W3LOAD and W3LOCK are set to 1 and the line of data to which Rn points is already in way 0, the cache is hit and new data is not loaded into way 3.

In DSP mode, bits W3LOCK and W2LOCK restrict the way that is to be replaced, when instructions other than the prefetch instruction are issued. Table 5.3 shows the relationship between the settings of bits in CCR2 and the way that is to be replaced when the cache is missed by instructions other than the prefetch instruction.

Programs that change the contents of the CCR2 register should be placed in address space that is not cached.

Bit	Bit Name	Initial Value	R/W	Description
31 to 10	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
9	W3LOAD	0	R/W	Way 3 Load (W3LOAD)
8	W3LOCK	0	R/W	Way 3 Lock (W3LOCK) When the cache is missed by a prefetch instruction while in DSP mode and when bits W3LOAD and W3LOCK in CCR2 are set to 1, the data is always loaded into way 3. Under any other condition, the prefetched data is loaded into the way to which LRU points.
7 to 2	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
1	W2LOAD	0	R/W	Way 2 Load (W2LOAD)
0	W2LOCK	0	R/W	Way 2 Lock (W2LOCK) When the cache is missed by a prefetch instruction while in DSP mode and when bits W2LOAD and W2LOCK in CCR2 are set to 1, the data is always loaded into way 2. Under any other condition, the prefetched data is loaded into the way to which LRU points.

Note: W2LOAD and W3LOAD should not be set to 1 at the same time.

Table 5.2 Way Replacement when a PREF Instruction Misses the Cache

DSP Bit	W3LOAD	W3LOCK	W2LOAD	W2LOCK	Way to be Replaced
0	*	*	*	*	Determined by LRU (table 5.1)
1	*	0	*	0	Determined by LRU (table 5.1)
1	*	0	0	1	Determined by LRU (table 5.4)
1	0	1	*	0	Determined by LRU (table 5.5)
1	0	1	0	1	Determined by LRU (table 5.6)
1	0	*	1	1	Way 2
1	1	1	0	*	Way 3

Note: * Don't care
W3LOAD and W2LOAD should not be set to 1 at the same time.

Table 5.3 Way Replacement when Instructions other than the PREF Instruction Miss the Cache

DSP Bit	W3LOAD	W3LOCK	W2LOAD	W2LOCK	Way to be Replaced
0	*	*	*	*	Determined by LRU (table 5.1)
1	*	0	*	0	Determined by LRU (table 5.1)
1	*	0	*	1	Determined by LRU (table 5.4)
1	*	1	*	0	Determined by LRU (table 5.5)
1	*	1	*	1	Determined by LRU (table 5.6)

Note: * Don't care
W3LOAD and W2LOAD should not be set to 1 at the same time.

Table 5.4 LRU and Way Replacement (when W2LOCK = 1 and W3LOCK =0)

LRU (Bits 5 to 0)	Way to be Replaced
000000, 000001, 000100, 010100, 100000, 100001, 110000, 110100	3
000011, 000110, 000111, 001011, 001111, 010110, 011110, 011111	1
101001, 101011, 111000, 111001, 111011, 111100, 111110, 111111	0

Table 5.5 LRU and Way Replacement (when W2LOCK = 0 and W3LOCK =1)

LRU (Bits 5 to 0)	Way to be Replaced
000000, 000001, 000011, 001011, 100000, 100001, 101001, 101011	2
000100, 000110, 000111, 001111, 010100, 010110, 011110, 011111	1
110000, 110100, 111000, 111001, 111011, 111100, 111110, 111111	0

Table 5.6 LRU and Way Replacement (when W2LOCK = 1 and W3LOCK =1)

LRU (Bits 5 to 0)	Way to be Replaced
000000, 000001, 000011, 000100, 000110, 000111, 001011, 001111, 010100, 010110, 011110, 011111	1
100000, 100001, 101001, 101011, 110000, 110100, 111000, 111001, 111011, 111100, 111110, 111111	0

5.3 Operation

5.3.1 Searching the Cache

If the cache is enabled (the CE bit in CCR1 = 1), whenever instructions or data in spaces P0, P1, P3, and U0 are accessed the cache will be searched to see if the desired instruction or data is in the cache. Figure 5.2 illustrates the method by which the cache is searched. The cache is a physical cache and holds physical addresses in its address section.

Entries are selected using bits 11 to 4 of the address (virtual) of the access to memory and the tag address of that entry is read. The virtual address (bits 31 to 10) of the access to memory and the physical address (tag address) read from the address array are compared. The address comparison uses all four ways. When the comparison shows a match and the selected entry is valid (V = 1), a cache hit occurs. When the comparison does not show a match or the selected entry is not valid (V = 0), a cache miss occurs. Figure 5.2 shows a hit on way 1.

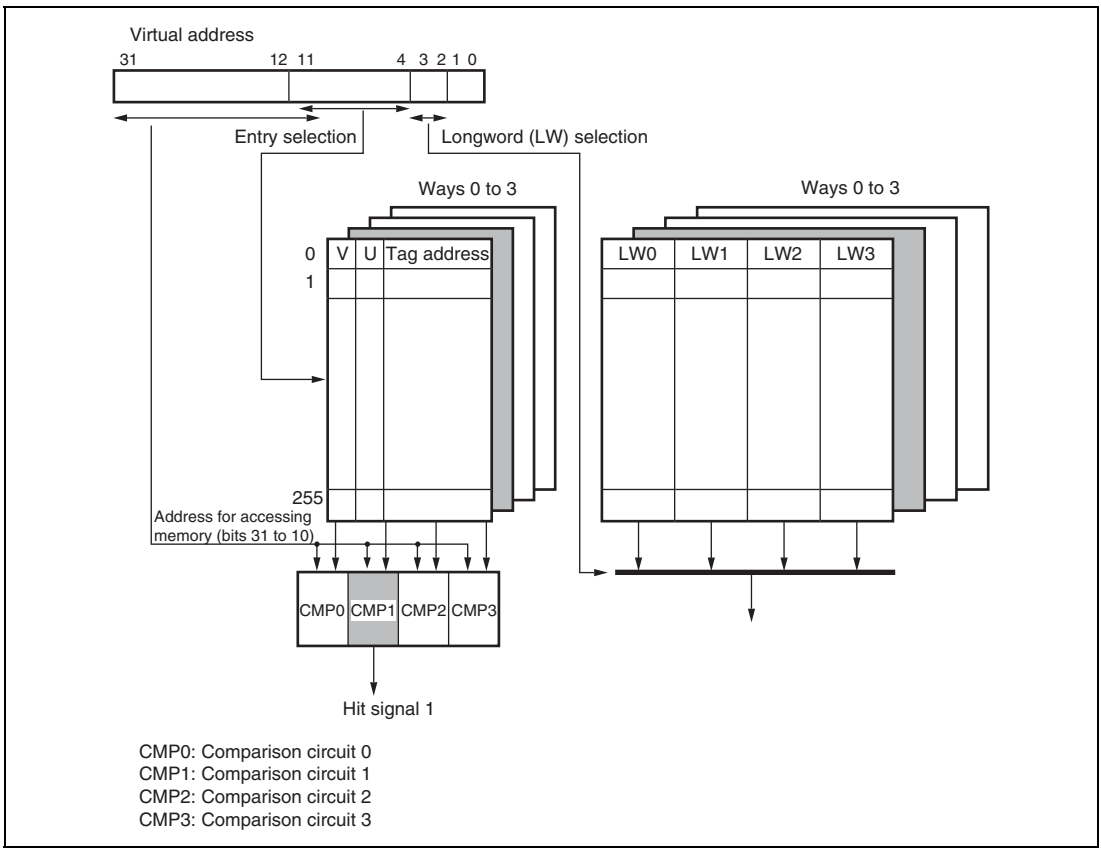


Figure 5.2 Cache Search Scheme

5.3.2 Read Access

Read Hit: In a read access, instructions and data are transferred from the cache to the CPU. The LRU is updated to indicate that the hit way is the most recently hit way.

Read Miss: An external bus cycle starts and the entry is updated. The way to be replaced is shown in table 5.3. Entries are updated in 16-byte units. When the desired instruction or data that caused the miss is loaded from external memory to the cache, the instruction or data is transferred to the CPU in parallel with being loaded to the cache. When it is loaded to the cache, the U bit is cleared to 0 and the V bit is set to 1 to indicate that the hit way is the most recently hit way. When the U bit for the entry which is to be replaced by entry updating in write-back mode is 1, the cache-update cycle starts after the entry is transferred to the write-back buffer. After the cache completes its update cycle, the write-back buffer writes the entry back to the memory. Transfer is in 16-byte units.

5.3.3 Prefetch Operation

Prefetch Hit: The LRU is updated to indicate that the hit way is the most recently hit way. The other contents of the cache are not changed. Instructions and data are not transferred from the cache to the CPU.

Prefetch Miss: Instructions and data are not transferred from the cache to the CPU. The way that is to be replaced is shown in table 5.2. The other operations are the same as those for a read miss.

5.3.4 Write Access

Write Hit: In a write access in write-back mode, the data is written to the cache and no external memory write cycle is issued. The U bit of the entry that has been written to is set to 1, and the LRU is updated to indicate that the hit way is the most recently hit way. In write-through mode, the data is written to the cache and an external memory write cycle is issued. The U bit of the entry that has been written to is not updated, and the LRU is updated to indicate that the hit way is the most recently hit way.

Write Miss: In write-back mode, an external cycle starts when a write miss occurs, and the entry is updated. The way to be replaced is shown in table 5.3. When the U bit of the entry which is to be replaced by entry updating is 1, the cache-update cycle starts after the entry has been transferred to the write-back buffer. Data is written to the cache and the U bit and the V bit are set to 1. The LRU is updated to indicate that the replaced way is the most recently updated way. After the cache has completed its update cycle, the write-back buffer writes the entry back to the memory. Transfer is in 16-byte units. In write-through mode, no write to cache occurs in a write miss; the write is only to the external memory.

5.3.5 Write-Back Buffer

When the U bit of the entry to be replaced in write-back mode is 1, the entry must be written back to the external memory. To increase performance, the entry to be replaced is first transferred to the write-back buffer and fetching of new entries to the cache takes priority over writing back to the external memory. After the fetching of new entries to the cache completes, the write-back buffer writes the entry back to the external memory. During the write-back cycles, the cache can be accessed. The write-back buffer can hold one line of cache data (16 bytes) and its physical address. Figure 5.3 shows the configuration of the write-back buffer.

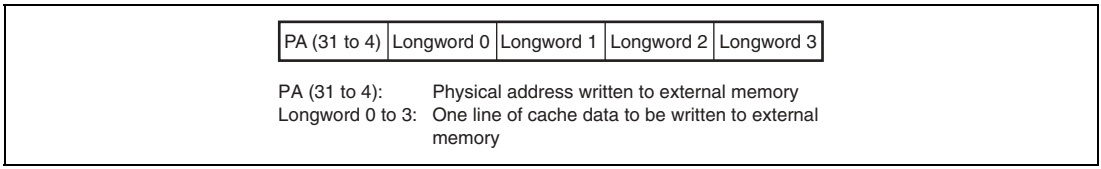


Figure 5.3 Write-Back Buffer Configuration

5.3.6 Coherency of Cache and External Memory

Use software to ensure coherency between the cache and the external memory. When memory shared by this LSI and another device is placed in an address space to which caching applies, use the memory-mapped cache to make the data invalid and written back, as required. Memory that is shared by this LSI's CPU and DMAC should also be handled in this way.

5.4 Memory-Mapped Cache

To allow software management of the cache, cache contents can be read and written by means of MOV instructions in privileged mode. The cache is mapped onto the P4 area in virtual address space. The address array is mapped onto addresses H'F0000000 to H'F0FFFFFF, and the data array onto addresses H'F1000000 to H'F1FFFFFF. Only longword can be used as the access size for the address array and data array, and instruction fetches cannot be performed.

5.4.1 Address Array

The address array is mapped onto H'F0000000 to H'F0FFFFFF. To access an address array, the 32-bit address field (for read/write accesses) and 32-bit data field (for write accesses) must be specified. The address field specifies information for selecting the entry to be accessed; the data field specifies the tag address, V bit, U bit, and LRU bits to be written to the address array.

In the address field, specify the entry address for selecting the entry, W for selecting the way, A for enabling or disabling the associative operation, and H'F0 for indicating address array access. As for W, B'00 indicates way 0, B'01 indicates way 1, B'10 indicates way 2, and B'11 indicates way 3.

In the data field, specify the tag address, LRU bits, U bit, and V bit. Figure 5.4 shows the address and data formats. 0s should be specified in upper three bits (bits 31 to 29) of the tag address. The following three operations are available in the address array.

Address-Array Read: Read the tag address, LRU bits, U bit, and V bit for the entry that corresponds to the entry address and way specified by the address field of the read instruction. In reading, the associative operation is not performed, regardless of whether the associative bit (A bit) specified in the address is 1 or 0.

Address-Array Write (non-Associative Operation): Write the tag address, LRU bits, U bit, and V bit, specified by the data field of the write instruction, to the entry that corresponds to the entry address and way as specified by the address field of the write instruction. Ensure that the associative bit (A bit) in the address field is set to 0. When writing to a cache line for which the U bit = 1 and the V bit =1, write the contents of the cache line back to memory, then write the tag address, LRU bits, U bit, and V bit specified by the data field of the write instruction. When 0 is written to the V bit, 0 must also be written to the U bit for that entry.

Address-Array Write (Associative Operation): When writing with the associative bit (A bit) of the address = 1, the addresses in the four ways for the entry specified by the address field of the write instruction are compared with the tag address that is specified by the data field of the write instruction. Write the U bit and the V bit specified by the data field of the write instruction to the entry of the way that has a hit. However, the tag address and LRU bits remain unchanged. When there is no way that receives a hit, nothing is written and there is no operation. This function is

used to invalidate a specific entry in the cache. When the U bit of the entry that has received a hit is 1 at this point, writing back should be performed. However, when 0 is written to the V bit, 0 must also be written to the U bit of that entry.

5.4.2 Data Array

The data array is mapped onto H'F1000000 to H'F1FFFFFF. To access a data array, the 32-bit address field (for read/write accesses) and 32-bit data field (for write accesses) must be specified. The address field specifies information for selecting the entry to be accessed; the data field specifies the longword data to be written to the data array.

In the address field, specify the entry address for selecting the entry, L for indicating the longword position within the (16-byte) line, W for selecting the way, and H'F1 for indicating data array access. As for L, B'00 indicates longword 0, B'01 indicates longword 1, B'10 indicates longword 2, and B'11 indicates longword 3. As for W, B'00 indicates way 0, B'01 indicates way 1, B'10 indicates way 2, and B'11 indicates way 3).

Since access size of the data array is fixed at longword, bits 1 and 0 of the address field should be set to B'00.

Figure 5.4 shows the address and data formats.

The following two operations on the data array are available. The information in the address array is not affected by these operations.

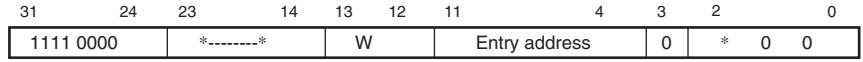
Data-Array Read: Read the data specified by L of the address filed, from the entry that corresponds to the entry address and the way that is specified by the address filed.

Data-Array Write: Write the longword data specified by the data filed, to the position specified by L of the address field, in the entry that corresponds to the entry address and the way specified by the address field.

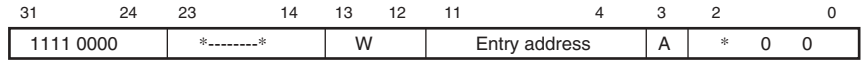
(1) Address array access

(a) Address specification

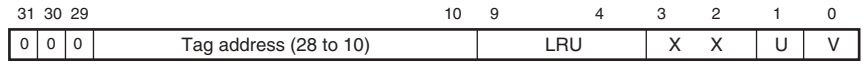
Read access



Write access

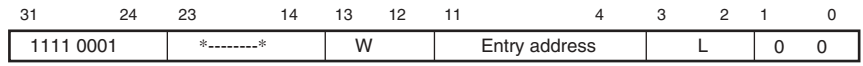


(b) Data specification (both read and write accesses)

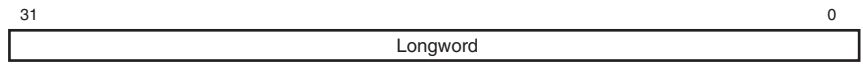


(2) Data array access (both read and write accesses)

(a) Address specification



(b) Data specification



*: Don't care bit
X: 0 for read, don't care for write

Figure 5.4 Specifying Address and Data for Memory-Mapped Cache Access

5.4.3 Usage Examples

Invalidating Specific Entries: Specific cache entries can be invalidated by writing 0 to the entry's V bit in the memory-mapped cache access. When the A bit is 1, the tag address specified by the write data is compared to the tag address within the cache selected by the entry address, and a match is found, the entry is written back if the entry's U bit is 1 and the V bit and U bit specified by the write data are written. If no match is found, there is no operation. In the example shown below, R0 specifies the write data and R1 specifies the address.

```
; R0=H'01100010; specification of data for address array access, tag
address=B'0 0001 0001 0000 0000 00, LRU = B'00 0001, U=0, V=0
; R1=H'F0000088; specification of address to be accessed in address
array access, way = B'00, entry address=B'00001000, A=1
;
MOV.L R0,@R1
```

Reading the Data of a Specific Entry: To read the data field of a specific entry is enabled by the memory-mapped cache access. The longword indicated in the data field of the data array in figure 5.4 is read into the register. In the example shown below, R0 specifies the address and R1 shows what is read.

```
; R0=H'F100 004C; specification of address for data array access,
way=B'00, entry address=B'00000100, L=B'11
;
MOV.L @R0,R1 ; Longword 3 is read.
```

Section 6 X/Y Memory

This LSI has on-chip 8 kbytes of X-memory and Y-memory that can be used to store instructions or data.

6.1 Features

- Page

  There are four pages. The X memory is divided into two pages (pages 0 and 1) and the Y memory is divided into two pages (pages 0 and 1).
- Memory map

  The X/Y memory is located in the virtual address space, physical address space.

  In the virtual address space, this memory is located in the addresses in P2/Uxy that are shown in table 6.1. These addresses are included in space P2 (when SR.MD = 1) or Uxy (when SR.MD = 0 and SR.DSP = 1) according to the CPU operating mode.

Table 6.1 X/Y Memory Virtual Addresses

Page	Memory Size (16 kbytes in Total Four Pages)
Page 0 of X memory	H'A5007000 to H'A5007FFF
Page 1 of X memory	H'A5008000 to H'A5008FFF
Page 0 of Y memory	H'A5017000 to H'A5017FFF
Page 1 of Y memory	H'A5018000 to H'A5018FFF

On the other hand, virtual addresses shown in table 6.1 are located in a part of area 1 in the physical address space. When this memory is accessed from the physical address space, addresses in which the upper three bits are 0 in addresses shown in table 6.1 are used.

- Ports

  Each page has three independent read/write ports and is connected to each bus. The X memory is connected to the I bus, X bus, and L bus. The Y memory is connected to the I bus, Y bus, and L bus. The L bus, X bus, and Y bus are used when this memory is accessed from the virtual address space. The I bus is used when this memory is accessed from the physical address space.
- Priority order

  In the event of simultaneous accesses to the same page from different buses, the accesses are processed according to the priority order. The priority order is: I bus > X bus > L bus in the X memory and I bus > Y bus > L bus in the Y memory.

6.2 Operation

6.2.1 Access from CPU

The 8/16/32-bit access by the CPU can be performed via the L bus or I bus. Methods for accessing by the CPU are via the L bus from the virtual addresses, and via the I bus from the physical addresses. As long as a conflict on the page does not occur, access via the L bus is performed in one cycle. Several cycles are necessary for accessing via the I bus. According to the CPU operating mode, access from the CPU is as follows:

Privileged mode and privileged DSP mode (SR. MD = 1): The X/Y memory can be accessed by the CPU from spaces P0 and P2.

User DSP mode (SR.MD = 0 and SR.DSP = 1): The X/Y memory can be accessed by the CPU from spaces U0 and Uxy.

User mode (SR.MD = 0 and SR.DSP = 0): The X/Y memory can be accessed by the CPU from space U0.

6.2.2 Access from DSP

The16/32-bit access by the DSP can be performed via the L bus or I bus. A 16-bit access by the DSP can be performed via the X and Y buses. Methods for accessing from the DSP differ according to instructions.

With a X data transfer instruction and a Y data transfer instruction, the X/Y memory is always accessed via the X bus or Y bus. As long as a conflict on the page does not occur, access via the X bus or Y bus is performed in one cycle. The X memory access via the X bus and the Y memory access via the Y bus can be performed simultaneously.

In the case of a single data transfer instruction, methods for accessing from the DSP are directly via the L bus from the virtual addresses, and via the I bus from the physical addresses. As long as a conflict on the page does not occur, access via the L bus is performed in one cycle. Several cycles are necessary for accessing via the I bus. According to the CPU operating mode, access from the CPU is as follows:

Privileged DSP mode (SR. MD = 1 and SR.DSP = 1): The X/Y memory can be accessed by the DSP from spaces P0 and P2.

User DSP mode (SR.MD = 0 and SR.DSP = 1): The X/Y memory can be accessed by the DSP from spaces U0 and Uxy.

6.2.3 Access from I Bus Master Module

The X/Y memory is always accessed by I bus master modules such as the DMAC and USBH via the I bus, which is a physical address bus. The 8/16/32-bit access is performed by the DMAC and 8/32-bit access is performed by the USBH. When accessing other than the P4 area (A31 to A29 = B'111), three most significant bits of the address are internally set to B'000 even if the logic addresses are specified other than P4 area. Therefore, access is performed through I bus.

6.3 Usage Notes

6.3.1 Page Conflict

In the event of simultaneous accesses to the same page from different buses, the conflict on the pages occurs. Although each access is completed correctly, this kind of conflict tends to lower X/Y memory accessibility. Therefore it is advisable to provide software measures to prevent such conflict as far as possible. For example, conflict will not arise if different memory or different pages are accessed by each bus.

6.3.2 Bus Conflict

The I bus is shared by several bus master modules. When the X/Y memory is accessed via the I bus, a conflict between the other I-bus master modules may occur on the I bus. This kind of conflict tends to lower X/Y memory accessibility. Therefore it is advisable to provide software measures to prevent such conflict as far as possible. For example, by accessing the X/Y memory by the CPU not via the I bus but from space P2 or Uxy via the L bus, conflict on the I bus can be prevented.

6.3.3 Cache Settings

When the X/Y memory is accessed via the I bus using the cache from the CPU and DSP, correct operation cannot be guaranteed. If the X/Y memory is accessed while the cache is enabled (CCR1.CE = 1), it is advisable to access the X/Y memory via the L bus from space P2 or Uxy. In a program that requires high performance, it is advisable to access the X/Y memory from space P2 or Uxy.

The relationship described above is summarized in table 6.2.

Table 6.2 Cache Settings

Setting	Address Space and Access Enabled or Disabled	
CCR1.CE	P0, U0	P2, Uxy
0	B	A
1	X	A

Note: A: Enabled (recommended)
B: Enabled
X: Disabled

6.3.4 Sleep Mode

The XYMCLKC bit of memory clock control register (MCCR) should be set to 1 when I bus master module such as DMAC accesses this memory in sleep mode.

There is the following restrictions when the X/Ymemory is accessed in the processing the exception handling and interrupt while this function is used.

1. The program of the exception handling and the interrupt processing should not be placed in the X/Y memory and U memory.
2. Eight NOP instructions should be added to the head of the exception handling and the interrupt processing program.
3. When the SLEEP instruction is executed, 16 instructions which include above-mentioned eight NOP instructions for the head of the exception handling and the interrupt processing should be placed outside of cache (Miss hit or non-cacheable area (P2)).

- Example 1

  Take out outside cache by using the address-array write (associative operation).

  (Refer to section 5.4, Memory Mapped Cache in section 5, Cache and Invalidating Specific Entries in section 5.4.3, Usage Examples.)

- Example 2

  The value of vector base register (VBR) is changed to non-cacheable area (P2) before the SLEEP instruction is executed. In this case, execute the SLEEP instruction after confirming the written VBR value (the branch instruction for the flag confirmation is used). After SLEEP ends, the VBR should be restored with the previous value.

Section 7 U Memory

This LSI has on-chip 128-kbyte U memory which can be used to store instructions or data.

7.1 Features

- Page

  There are two pages (pages 0 and 1).
- Memory map

  The U memory is located in both the virtual address space and physical address space.

  In the virtual address space, this memory is located in the addresses in P2/Uxy that are shown in table 7.1. These addresses are included in space P2 (when SR.MD = 1) or Uxy (when SR.MD = 0 and SR.DSP = 1) according to the CPU operating mode.

Table 7.1 U Memory Virtual Addresses

Page	Memory Size (128 kbytes in Total Two Pages)
Page 0 of U memory	H'A55F0000 to H'A55FFFFF
Page 1 of U memory	H'A5600000 to H'A560FFF

On the other hand, virtual addresses shown in table 7.1 are located in a part of area 1 in the physical address space. When this memory is accessed from the physical address space, addresses in which the upper three bits are 0 in addresses shown in table 7.1 are used.

- Ports

  Each page has two independent read/write ports and is connected to the I bus or L bus. The L bus is used when this memory is accessed from the virtual address space. The I bus is used when this memory is accessed from the physical address space.
- Priority order

  In the event of simultaneous accesses to the same page from different buses, the accesses are processed according to the priority order. The priority order is: I bus > L bus.

7.2 Operation

7.2.1 Access from CPU

The 8/16/32-bit access by the CPU can be performed via the L bus or I bus. Methods for accessing by the CPU are directly via the L bus from the virtual addresses, and via the I bus from the physical addresses. As long as a conflict on the page does not occur, access via the L bus is performed in one cycle. Several cycles are necessary for accessing via the I bus. According to the CPU operating mode, access from the CPU is as follows:

Privileged mode and privileged DSP mode (SR. MD = 1): The U memory can be accessed by the CPU from spaces P0 and P2.

User DSP mode (SR.MD = 0 and SR.DSP = 1): The U memory can be accessed by the CPU from spaces U0 and Uxy.

User mode (SR.MD = 0 and SR.DSP = 0): The U memory can be accessed by the CPU from space U0.

7.2.2 Access from DSP

The U memory can be accessed from the DSP only by a single data transfer instruction. The16/32 -bit access by the DSP can be performed via the L bus or I bus. The access cannot be performed by an X data transfer instruction and a Y data transfer instruction.

Methods for accessing from the DSP are via the L bus from the virtual addresses, and via the I bus from the physical addresses. As long as a conflict on the page does not occur, access via the L bus is performed in one cycle. Several cycles are necessary for accessing via the I bus. According to the CPU operating mode, access from the CPU is as follows:

Privileged DSP mode (SR. MD = 1 and SR.DSP = 1): The U memory can be accessed by the DSP from spaces P0 and P2.

User DSP mode (SR.MD = 0 and SR.DSP = 1): The U memory can be accessed by the DSP from spaces U0 and Uxy.

7.2.3 Access from I Bus Master Module

The U memory is always accessed by I bus master modules such as the DMAC and USBH via the I bus, which is a physical address bus. The 8/16/32-bit access can be performed by the DMAC and the 8/32-bit access can be performed by the USBH. When accessing other than the P4 area (A31 to A29 = B'111), three most significant bits of the address are internally set to B'000 even if the logic addresses are specified other than P4 area. Therefore, access is performed through I bus.

7.3 Usage Notes

7.3.1 Page Conflict

In the event of simultaneous accesses to the same page from different buses, the conflict on the pages occurs. Although each access is completed correctly, this kind of conflict tends to lower U memory accessibility. Therefore it is advisable to provide software measures to prevent such conflict as far as possible. For example, conflict will not arise if different memory or different pages are accessed by each bus.

7.3.2 Bus Conflict

The I bus is shared by several bus master modules. When the U memory is accessed via the I bus, a conflict between the other I-bus master modules may occur on the I bus. This kind of conflict tends to lower U memory accessibility. Therefore it is advisable to provide software measures to prevent such conflict as far as possible. For example, by accessing the U memory by the CPU not via the I bus but from space P2 or Uxy via the L bus, conflict on the I bus can be prevented.

7.3.3 Cache Settings

When the U memory is accessed via the I bus using the cache from the CPU and DSP, correct operation cannot be guaranteed. If the U memory is accessed while the cache is enabled (CCR1.CR = 1), it is advisable to access the U memory via the L bus from space P2 or Uxy. In a program that requires high performance, it is advisable to access the U memory from space P2 or Uxy.

The relationship described above is summarized in table 7.2.

Table 7.2 Cache Settings

Setting	Address Space and Access Enabled or Disabled	
CCR1.CE	**P0, U0**	**P2, Uxy**
0	B	A
1	X	A

Note: A: Enabled (recommended)
B: Enabled
X: Disabled

7.3.4 sleep mode

The UMCLKC bit of memory clock control register (MCCR) should be set to 1 when I bus master module such as DMAC accesses this memory in sleep mode.

There is the following restrictions when the U memory is accessed in the processing the exception handling and interrupt while this function is used.

1. The program of the exception handling and the interrupt processing should not be placed in the U memory and X/Y memory.
2. Eight NOP instructions should be added to the head of the exception handling and the interrupt processing program.
3. When the SLEEP instruction is executed, 16 instructions which include above-mentioned eight NOP instructions for the head of the exception handling and the interrupt processing should be placed outside of cache (Miss hit or non-cacheable area (P2)).

- Example 1

  Take out outside cache by using the address-array write (associative operation).

  (Refer to section 5.4, Memory Mapped Cache in section 5, Cache and Invalidating Specific Entries in section 5.4.3, Usage Examples.)
- Example 2

  The value of vector base register (VBR) is changed to non-cacheable area (P2) before the SLEEP instruction is executed. In this case, execute the SLEEP instruction after confirming the written VBR value (the branch instruction for the flag confirmation is used). After SLEEP ends, the VBR should be restored with the previous value.

Section 8 Interrupt Controller (INTC)

The interrupt controller (INTC) ascertains the priority of interrupt sources and controls interrupt requests to the CPU. The INTC registers set the order of priority of each interrupt, allowing the user to process interrupt requests according to the user-set priority.

8.1 Features

- 16 levels of interrupt priority can be set

  By setting the interrupt priority registers, the priorities of on-chip peripheral module and IRQ interrupts can be selected from 16 levels for individual request sources.
- NMI noise canceler function

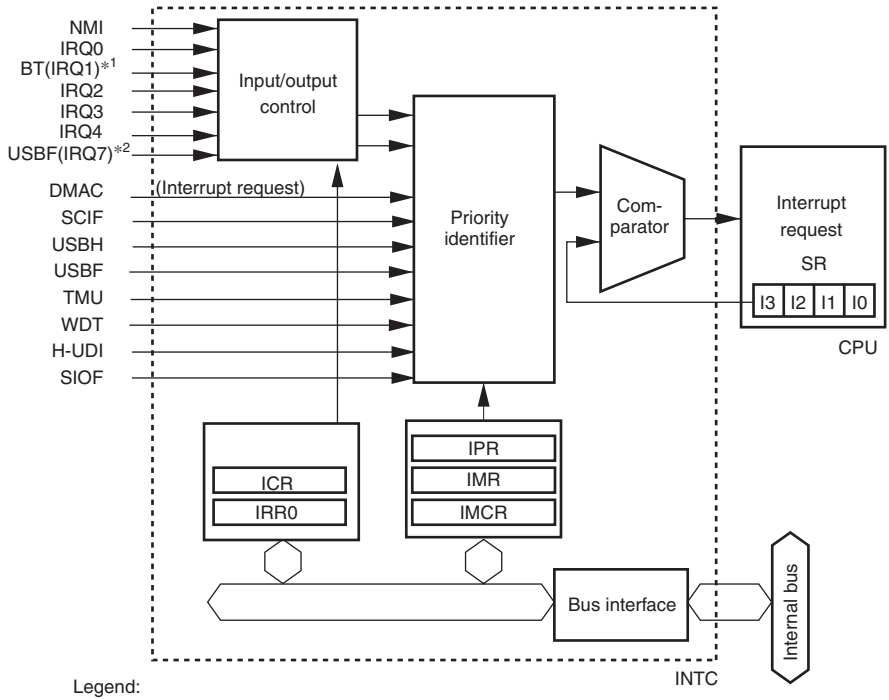
  An NMI input-level bit indicates the NMI pin state. By reading this bit in the interrupt exception service routine, the pin state can be checked, enabling it to be used as a noise canceler.
- IRQ interrupts can be set

  Detection of low level, high level, rising edge, or falling edge
- Interrupt request signal can be output ($\overline{\text{IRQOUT}}$ pin)

  Bus mastership can be required by signaling the external bus master that the external interrupt and on-chip peripheral module interrupt requests have been generated
- Interrupts can be enabled or disabled

  Interrupts can be enabled or disabled individually for each interrupt source with the interrupt mask registers and interrupt mask clear registers.

Figure 8.1 shows a block diagram of the INTC.



Legend:

DMAC : DMA controller
SIOF : Serial I/O with FIFO
SCIF : Serial communication interfaces with FIFO
USBH : USB host controller
USBF : USB function controller
TMU : Timer unit
BT : Bluetooth interface
WDT : Watchdog timer
H-UDI : User-debugging interface
ICR : Interrupt control registers
IPR : Interrupt priority registers
IMR : Interrupt mask registers
IMCR : Interrupt mask clear registers
IRR0 : Interrupt request register 0
SR : Status register

Notes: 1. The interrupt request pin for the BT (Bluetooth interface) is connected to the IRQ1 pin and is handled as an external interrupt.
2. The interrupt request pin for the USBF (USB function controller) standby recovery is connected to the IRQ7 pin and is handled as an external interrupt.

Figure 8.1 Block Diagram of INTC

8.2 Input/Output Pins

Table 8.1 shows the INTC pin configuration.

Table 8.1 Pin Configuration

Pin Name	I/O	Description
NMI	Input	Nonmaskable Interrupt Input
		Input of interrupt request signal which is not maskable
IRQ7, IRQ4 to IRQ0*1	Input	Interrupt Input
		Input of interrupt request signal
$\overline{\text{IRQOUT}}$*2	Output	Bus Mastership Request Output
		Signal of interrupt request generation

Notes: 1. IRQ1 is used as the bluetooth interface interrupt request pin and does not support an external pin function. IRQ7 is used as the standby recovery interrupt request pin and does not support an external pin function.
2. When the NMI or H-UDI interrupt request is generated and the response time of CPU is short, this pin may not be asserted.

8.3 Register Descriptions

The INTC has the following registers. For details on register addresses and register states during each processing, refer to section 27, List of Registers.

- Interrupt control register 0 (ICR0)
- Interrupt control register 1 (ICR1)
- Interrupt control register 2 (ICR2)
- Interrupt priority register A (IPRA)
- Interrupt priority register B (IPRB)
- Interrupt priority register C (IPRC)
- Interrupt priority register D (IPRD)
- Interrupt priority register E (IPRE)
- Interrupt priority register F (IPRF)
- Interrupt priority register G (IPRG)
- Interrupt priority register H (IPRH)
- Interrupt request register 0 (IRR0)
- Interrupt mask register 0 (IMR0)
- Interrupt mask register 1 (IMR1)
- Interrupt mask register 4 (IMR4)
- Interrupt mask register 5 (IMR5)
- Interrupt mask register 6 (IMR6)

- Interrupt mask register 9 (IMR9)
- Interrupt mask clear register 0 (IMCR0)
- Interrupt mask clear register 1 (IMCR1)
- Interrupt mask clear register 4 (IMCR4)
- Interrupt mask clear register 5 (IMCR5)
- Interrupt mask clear register 6 (IMCR6)
- Interrupt mask clear register 9 (IMCR9)

8.3.1 Interrupt Priority Registers A to H (IPRA to IPRH)

IPRA to IPRH are 16-bit readable/writable registers in which priority levels from 0 to 15 are set for on-chip peripheral module and IRQ interrupts.

Bit	Bit Name	Initial Value	R/W	Description
15	IPR15	0	R/W	These bits set the priority level for each interrupt source in 4-bit units. For details, see table 8.2, Interrupt Sources and IPRA to IPRH.
14	IPR14	0	R/W	
13	IPR13	0	R/W	
12	IPR12	0	R/W	
11	IPR11	0	R/W	
10	IPR10	0	R/W	
9	IPR9	0	R/W	
8	IPR8	0	R/W	
7	IPR7	0	R/W	
6	IPR6	0	R/W	
5	IPR5	0	R/W	
4	IPR4	0	R/W	
3	IPR3	0	R/W	
2	IPR2	0	R/W	
1	IPR1	0	R/W	
0	IPR0	0	R/W	

Table 8.2 Interrupt Sources and IPRA to IPRH

Register	Bits 15 to 12	Bits 11 to 8	Bits 7 to 4	Bits 3 to 0
IPRA	TMU0	TMU1	TMU2	Reserved*[1]
IPRB	WDT	Reserved*[1]	Reserved*[1]	Reserved*[1]
IPRC	IRQ3	IRQ2	IRQ1(for BT)*[2]	IRQ0
IPRD	IRQ7(for USBF standby recovery interrupt request signal) *[3]	Reserved*[1]	Reserved*[1]	IRQ4
IPRE	DMAC	Reserved*[1]	Reserved*[1]	Reserved*[1]
IPRF	Reserved*[1]	Reserved*[1]	USBF	USBH
IPRG	SCIF0	SCIF1	Reserved*[1]	Reserved*[1]
IPRH	Reserved*[1]	SIOF	Reserved*[1]	Reserved*[1]

Notes: 1. Always read as 0. The write value should always be 0. If 1 is written to the bit, correct operation cannot be guaranteed.
2. IRQ1 is connected to the bluetooth interface (BT) interrupt request signal and cannot be used for other functions.
3. IRQ7 is connected to the USBF (USB function controller) standby recovery interrupt request signal and cannot be used for other functions.

As shown in table 8.2, on-chip peripheral module or IRQ interrupts are assigned to four 4-bit groups in each register. These 4-bit groups (bits 15 to 12, bits 11 to 8, bits 7 to 4, and bits 3 to 0) are set with values from H'0 (0000) to H'F (1111). Setting H'0 means priority level 0 (masking is requested); H'F means priority level 15 (the highest level).

8.3.2 Interrupt Control Register 0 (ICR0)

ICR0 is a register that sets the input signal detection mode of external interrupt input pin NMI, and indicates the input signal level at the NMI pin.

Bit	Bit Name	Initial Value	R/W	Description
15	NMIL	0/1*	R	NMI Input Level Sets the level of the signal input at the NMI pin. This bit can be read from to determine the NMI pin level. This bit cannot be modified. 0: NMI input level is low 1: NMI input level is high
14 to 9	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
8	NMIE	0	R/W	NMI Edge Select Selects whether the falling or rising edge of the interrupt request signal at the NMI pin is detected. 0: Interrupt request is detected at the falling edge of NMI input 1: Interrupt request is detected at the rising edge of NMI input
7 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

Note: * 1 when NMI input is high, 0 when NMI input is low.

8.3.3 Interrupt Control Register 1 (ICR1)

ICR1 is a 16-bit register that specifies the detection mode for external interrupt input pins IRQ4 to IRQ0 individually: rising edge, falling edge, high level, or low level.

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
14	IRQE*	1	R/W	Interrupt Request Enable Enables or disables the use of pins IRQ7, IRQ4 to IRQ0 as six independent interrupt pins. 0 : Use of pins IRQ7, IRQ4 to IRQ0 as six independent interrupt pins enabled 1 : Use of pins IRQ7, IRQ4 to IRQ0 as interrupt pins disabled
13 to 10	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
9	IRQ41S	0	R/W	IRQn Sense Select
8	IRQ40S	0	R/W	These bits select whether interrupt request signals corresponding to pins IRQ4 to IRQ0 are detected by a rising edge, falling edge, high level, or low level.
7	IRQ31S	0	R/W	
6	IRQ30S	0	R/W	
5	IRQ21S	0	R/W	
4	IRQ20S	0	R/W	
3	IRQ11S	0	R/W	
2	IRQ10S	0	R/W	
1	IRQ01S	0	R/W	
0	IRQ00S	0	R/W	

Bit 2n+1	Bit 2n	
IRQn1S	**IRQn0S**	
0	0	Interrupt request is detected at the falling edge of IRQn input
0	1	Interrupt request is detected at the rising edge of IRQn input
1	0	Interrupt request is detected on low level of IRQn input
1	1	Interrupt request is detected on high level of IRQn input

Legend: n = 0 to 4

Note: * The IRQE bit must be cleared to 0 before the BL bit in the status register (SR) is cleared to 0 in the initialization routine after reset, and must then not be changed. If the IRQE bit is set to 1 when the BL bit is cleared to 0, correct operation cannot be guaranteed.

8.3.4 Interrupt Control Register 2 (ICR2)

ICR2 is a 16-bit register that specifies the detection mode for the external interrupt input pin IRQ7: rising edge, falling edge, high level, or low level.

Bit	Bit Name	Initial Value	R/W	Description
15 to 4	—	All 0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
3	IRQ71S	0	R/W	IRQ7 Sense Select
2	IRQ70S	0	R/W	These bits select whether interrupt request signals corresponding to pin IRQ7 are detected by a rising edge, falling edge, high level, or low level.
1	—	All 0	R	Reserved
0				These bits are always read as 0. The write value should always be 0 or the bit7 in IMR0 should be set to 1. If 1 is written to these bits, correct operation cannot be guaranteed.

Bit 2n+1	Bit 2n	
IRQ71S	IRQ70S	
0	0	Interrupt request (USBF standby recovery interrupt request) is detected at the falling edge of IRQ7 input
0	1	Setting prohibited
1	0	Setting prohibited
1	1	Setting prohibited

Note: When the USBF standby recovery interrupt request need not to be detected, clear the bits 15 to 12 in IPRD to 0 or set the bit 7 in IMR0 to 1, and the IRQ7 interrupt request should be masked.

8.3.5 Interrupt Request Register 0 (IRR0)

IRR0 is an 8-bit register that indicates interrupt requests from external input pins IRQ7, IRQ4* to IRQ0.

Note: * In this LSI, IRQ7 and IRQ1 are used specific to internal module and cannot function as external pins.

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ7R*[2]	0	R/W	IRQn Interrupt Request
6*[1]	—	0	R	Indicates whether there is interrupt request input to the IRQn pin. When edge-detection mode is set for IRQn, an interrupt request is cleared by writing 0 to the IRQnR bit after reading IRQnR = 1.
5*[1]	—	0	R	
4	IRQ4R*[2]	0	R/W	
3	IRQ3R*[2]	0	R/W	When level-detection mode is set for IRQn, these bits indicate whether an interrupt request is input. The interrupt request is set/cleared by only 1/0 input to the IRQn pin.
2	IRQ2R*[2]	0	R/W	
1	IRQ1R*[2]	0	R/W	
0	IRQ0R*[2]	0	R/W	IRQnR
				0: No interrupt request input to IRQn pin
				1: Interrupt request input to IRQn pin
				Legend: n = 0 to 4, 7

Notes: 1. Bits 6 to 5 are reserved bits. These bits are always read as undefined value. The write value should always be 0. If 1 is written to the bits, correct operation cannot be guaranteed.

2. The IRQ7R, IRQ4R to IRQ0R bits are cleared to 0 after the IRQE bit in the interrupt control register 1 (ICR1) is cleared to 0 in the initialization routine after reset.

8.3.6 Interrupt Mask Registers 0, 1, 4 to 6, and 9 (IMR0, IMR1, IMR4 to IMR6, and IMR9)

IMR0, IMR1, IMR4 to IMR6, and IMR9 are 8-bit readable/writable registers that mask the IRQ and on-chip peripheral module interrupts. When an interrupt source is masked, interrupt requests may be mistakenly detected, depending on the operation state of the IRQ pins and on-chip peripheral modules. To prevent this, set IMR0, IMR1, IMR4 to IMR6, and IMR9 while no interrupts are set to be generated, and then read the new settings from these registers.

Table 8.3 shows the relationship between IMR and each interrupt source.

Bit	Bit Name	Initial Value	R/W	Description
7	IM7	0	R/W	Interrupt Mask
6	IM6	0	R/W	Table 8.3 lists the correspondence between the interrupt sources and interrupt mask registers.
5	IM5	0	R/W	
4	IM4	0	R/W	
3	IM3	0	R/W	
2	IM2	0	R/W	
1	IM1	0	R/W	
0	IM0	0	R/W	

IMn	Read	Write
1	Interrupt source of the corresponding bit is masked	
0	Interrupt source of the corresponding bit is not masked	No processing

Legend: n = 7 to 0

Table 8.3 Correspondence between Interrupt Sources and IMR0 to IMR9/IMCR0 to IMCR9

Register Name	Bit Name (Function Name)							
	7	6	5	4	3	2	1	0
IMR0/IMCR0	IRQ7 (USBF standby recovery interrupt request)	—	—	IRQ4	IRQ3	IRQ2	IRQ1 (BT)	IRQ0
IMR1/IMCR1	—	—	—	—	DEI3	DEI2	DEI1	DEI0
			—				(DMAC)	
IMR4/IMCR4	—	TUNI2	TUNI1	TUNI0	ITI	—	—	—
		(TMU2)	(TMU1)	(TMU0)	(WDT)		—	
IMR5/IMCR5	—	—	—	—	—	—	SCIFI1	SCIFI0
	—	—		—		—	(SCIF1)	(SCIF0)
IMR6/IMCR6	—	—	—	—	—	—	SIOFI	—
		—	—	—		—	(SIOF)	—
IMR9/IMCR9	—	—	—	—	—	USBFI1	USBFI0	USBHI
			—		—	(USBF)		(USBH)

Note: —: Reserved. These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

8.3.7 Interrupt Mask Clear Registers 0, 1, 4 to 6, and 9 (IMCR0, IMCR1, IMCR4 to IMCR6, and IMCR9)

IMCR0, IMCR1, IMCR4 to IMCR6, and IMCR9 are 8-bit writable registers that clear the mask settings for the IRQ and on-chip peripheral module interrupts. Table 8.3 shows the relationship between IMCR and each interrupt source.

Bit	Bit Name	Initial Value	R/W	Description
7	IMC7	—	W	Interrupt Mask Clear
6	IMC6	—	W	Table 8.3 lists the correspondence between interrupt sources and interrupt mask clear registers.
5	IMC5	—	W	
4	IMC4	—	W	
3	IMC3	—	W	
2	IMC2	—	W	
1	IMC1	—	W	
0	IMC0	—	W	

IMC n	Write
1	Corresponding bit in interrupt mask register IMRn is cleared
0	No processing

Legend: n = 7 to 0

8.4 Interrupt Sources

There are three types of interrupt sources: NMI, IRQ, and on-chip peripheral modules. Each interrupt has a priority level (0 to 16), with 1 the lowest and 16 the highest. Priority level 0 masks an interrupt, so the interrupt request is ignored.

8.4.1 NMI Interrupt

The NMI interrupt has the highest priority level of 16. When the BL bit in the status register (SR) is 0, NMI interrupts are accepted. NMI interrupts are edge-detected. In sleep or standby mode, the interrupt is accepted regardless of the BL setting. The NMI edge select bit (NMIE) in the interrupt control register 0 (ICR0) is used to select either rising or falling edge detection.

When using edge-input detection for NMI interrupts, a pulse width of at least two Pϕ cycles (peripheral clock) is necessary. NMI interrupt exception handling does not affect the interrupt mask level bits (I3 to I0) in the status register (SR).

It is possible to wake the chip up from sleep mode or standby mode with an NMI interrupt.

8.4.2 IRQ Interrupts

IRQ interrupts are input by level or edge from pins IRQ7, IRQ4 to IRQ0. The priority level can be set by interrupt priority registers C and D (IPRC and IPRD) in a range from 0 to 15.

When using edge-sensing for IRQ interrupts, clear the interrupt source by having software read 1 from the corresponding bit in IRR0, then write 0 to the bit.

When the ICR1 and ICR2 registers are rewritten, IRQ interrupts may be mistakenly detected, depending on the IRQ pin states. To prevent this, rewrite the register while interrupts are masked, then release the mask after clearing the illegal interrupt by writing 0 to IRR0 after reading IRR0.

Edge input interrupt detection requires input of a pulse width of more than two cycles on a peripheral clock (Pϕ) basis.

When using level-sensing for IRQ interrupts, the pin levels must be retained until the CPU samples the pins. Therefore, the interrupt source must be cleared by the interrupt handler.

The interrupt mask bits (I3 to I0) in the status register (SR) are not affected by IRQ interrupt handling.

8.4.3 On-Chip Peripheral Module Interrupts

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

- DMA controller (DMAC)
- Serial I/O with FIFO (SIOF)
- Serial communication interfaces with FIFO (SCIF0 and SCIF1)
- USB host controller (USBH)
- USB function controller (USBF)
- Timer unit (TMU)
- Watchdog timer (WDT)
- User-debugging interface (H-UDI)

Note: * An interrupt generated at the bluetooth interface (BT) is not handles as a on-chip peripheral module interrupt and is connected to the eternal interrupt pin (IRQ1) of this LSI. The USB function controller (USBF) standby recovery interrupt is not handled as a on-chip peripheral module interrupt and is connected to the external interrupt pin (IRQ7) of this LSI.

Not every interrupt source is assigned a different interrupt vector. Sources are reflected in the interrupt event register 2 (INTEVT2). It is easy to identify sources by using the value of the INTEVT2 register as a branch offset.

A priority level (from 0 to 15) can be set for each module except H-UDI by writing to interrupt priority registers A to H (IPRA to IPRH). The priority level of the H-UDI interrupt is 15 (fixed).

The interrupt mask bits (I3 to I0) in the status register are not affected by on-chip peripheral module interrupt handling.

8.4.4 Interrupt Exception Handling and Priority

There are three types of interrupt sources: NMI, IRQ, and on-chip peripheral modules. The priority of each interrupt source is set within priority levels 0 to 15; level 15 is the highest and level 1 is the lowest. When the priority is set to level 0, that interrupt is masked and the interrupt request is ignored.

Table 8.4 lists the codes for the interrupt event register 2 (INTEVT2) and the order of interrupt priority.

Each interrupt source is assigned a unique code by INTEVT2. The start address of the interrupt service routine is common for each interrupt source. This is why, for instance, the value of INTEVT2 is used as an offset at the start of the interrupt service routine and branched to in order to identify the interrupt source.

IRQ interrupt and on-chip peripheral module interrupt priorities can be set properly between 0 and 15 for each module by setting interrupt priority registers A to C and F to H (IPRA to IPRC and IPRF to IPRH). A reset assigns priority level 0 to IRQ and on-chip peripheral module interrupts.

If the same priority level is assigned to two or more interrupt sources and interrupts from those sources occur simultaneously, their priority order is the default priority order indicated at the right in table 8.4.

Table 8.4 Interrupt Exception Handling Sources and Priority

Interrupt Source		Interrupt Code	Interrupt Priority (Initial Value)	IPR (Bit Numbers)	Priority within IPR Setting Unit	Default Priority
NMI		H'1C0	16	—	—	High
H-UDI		H'5E0	15	—	—	↑
IRQ	IRQ0	H'600	0 to 15 (0)	IPRC (3 to 0)	—	
	IRQ1(BT)	H'620	0 to 15 (0)	IPRC (7 to 4)	—	
	IRQ2	H'640	0 to 15 (0)	IPRC (11 to 8)		
	IRQ3	H'660	0 to 15 (0)	IPRC (15 to 12)		
	IRQ4	H'680	0 to 15 (0)	IPRD (3 to 0)		
	IRQ7 (USBF standby recovery)	H'6E0	0 to 15 (0)	IPRD (15 to 12)		
DMAC	DEI0	H'800	0 to 15 (0)	IPRE (15 to 12)	High	
	DEI1	H'820			↕	
	DEI2	H'840				
	DEI3	H'860			Low	
USBH	USBHI	H'A00	0 to 15 (0)	IPRF (3 to 0)	—	
USBF	USI0	H'A20	0 to 15 (0)	IPRF (7to 4)	High	
	USI1	H'A40			Low	
SCIF0	SCIFI0	H'C00	0 to 15 (0)	IPRG (15 to 12)	—	
SCIF1	SCIFI1	H'C20	0 to 15 (0)	IPRG (11 to 8)	—	
SIOF	SIOFI1	H'CA0	0 to 15 (0)	IPRH (11 to 8)	—	
TMU0	TUNI0	H'400	0 to 15 (0)	IPRA (15 to 12)	—	
TMU1	TUNI1	H'420	0 to 15 (0)	IPRA (11 to 8)	—	
TMU2	TUNI2	H'440	0 to 15 (0)	IPRA (8 to 4)	—	
WDT	ITI	H'560	0 to 15 (0)	IPRB (15 to 12)	—	Low

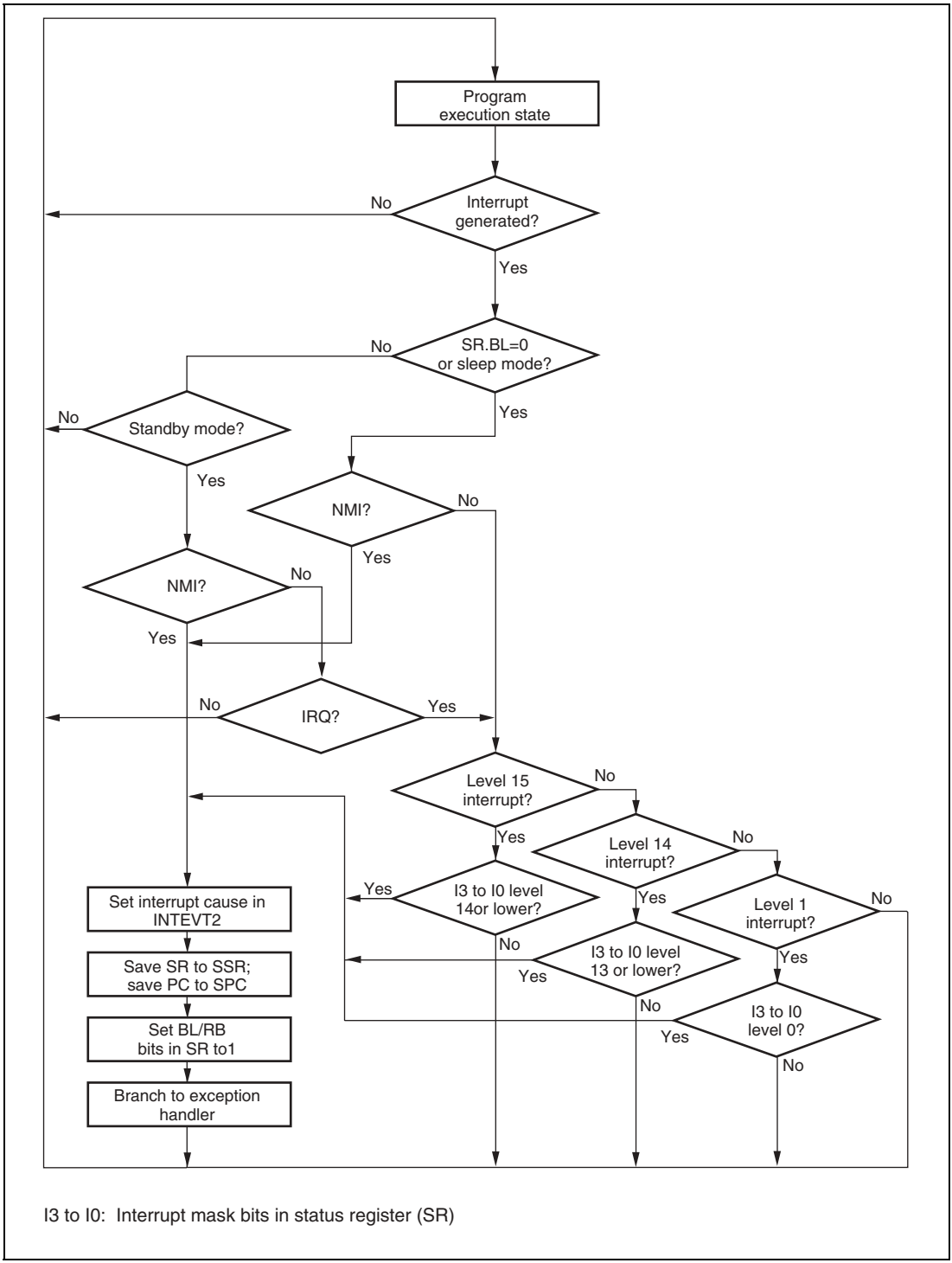
8.5 Operation

8.5.1 Interrupt Sequence

The sequence of interrupt operations is described below. Figure 8.2 is a flowchart of the operations.

1. The interrupt request sources send interrupt request signals to the interrupt controller.
2. The interrupt controller selects the highest-priority interrupt from the interrupt requests sent, following the priority levels set in interrupt priority registers A to H (IPRA to IPRH). Lower priority interrupts are held pending. If two of these interrupts have the same priority level or if multiple interrupts occur within a single module, the interrupt with the highest priority is selected, according to table 8.4, Interrupt Exception Handling Sources and Priority.
3. The priority level of the interrupt selected by the interrupt controller is compared with the interrupt mask bits (I3 to I0) in the status register (SR) of the CPU. If the request priority level is higher than the level in bits I3 to I0, the interrupt controller accepts the interrupt and sends an interrupt request signal to the CPU.
4. Detection timing: The INTC operates, and notifies the CPU of interrupt requests, in synchronization with the peripheral clock (Pϕ). The CPU receives an interrupt at a break in instructions.
5. The interrupt source code is set in the interrupt event register 2 (INTEVT2).
6. The status register (SR) and program counter (PC) are saved to SSR and SPC, respectively.
7. The block bit (BL), register bank bit (RB), and mode bit (MD) in SR are set to 1.
8. The CPU jumps to the start address of the interrupt handler (the sum of the value set in the vector base register (VBR) and H'00000600). This jump is not a delayed branch. The interrupt handler may branch with the INTEVT2 register value as its offset in order to identify the interrupt source. This enables it to branch to the handling routine for the individual interrupt source.

Notes: 1. The interrupt mask bits (I3 to I0) in the status register (SR) are not changed by acceptance of an interrupt in this LSI.
2. The interrupt source flag should be cleared in the interrupt handler. To ensure that an interrupt request that should have been cleared is not inadvertently accepted again, read the interrupt source flag after it has been cleared, and then execute an RTE instruction.



I3 to I0: Interrupt mask bits in status register (SR)

Figure 8.2 Interrupt Operation Flowchart

8.5.2 Multiple Interrupts

When handling multiple interrupts, an interrupt handler should include the following procedures:

1. Branch to a specific interrupt handler corresponding to a code set in INTEVT2. The code in INTEVT2 can be used as an offset for branching to the specific handler.
2. Clear the interrupt source in each specific handler.
3. Save SSR and SPC to memory.
4. Clear the BL bit in SR, and set the accepted interrupt level in the interrupt mask bits in SR.
5. Handle the interrupt.
6. Execute the RTE instruction.

When the sequence of interrupt operations is described above, if an interrupt of higher priority than the one being handled occurs directly after execution of the operation in step 4, it can be accepted.

Section 9 Bus State Controller (BSC)

9.1 Overview

The bus state controller (BSC) outputs control signals for various types of memory that is connected to the external address space and external devices. BSC functions enable this LSI to connect directly with SRAM, SDRAM, and other memory storage devices, and external devices.

9.1.1 Features

The BSC has the following features:

1. Physical address space is divided into eight areas

A maximum 16 Mbytes for each of the three areas, CS0, CS3, and CS4, totally 48 Mbytes.

Can specify the normal space interface, SRAM interface with byte selection, burst ROM interface, and SDRAM interface*[1]

Can select the data bus width (8 or 16 bits) for each address space*[2].

Controls the insertion of the wait state for each address space.

Controls the insertion of the wait state for each read access and write access.

Can set the independent idling cycle in the continuous access for five cases: read-write (in same space/different space), read-read (in same space/different space), the first cycle is a write access.

2. Normal space interface

Supports the interface that can directly connect to the normal memories such as SRAM and ROM.

3. Burst ROM interface

High-speed access to the ROM that has the page mode function (page flash ROM).

4. SDRAM interface

Can set the SDRAM in CS3 area.

Multiplex output for row address/column address.

Efficient access by single read/single write.

High-speed access by the bank-active mode.

Supports an auto-refresh and self-refresh.

5. Byte-selection SRAM interface

Can connect directly to a byte-selection SRAM

6. Bus arbitration

Shares all of the resources with other CPU and outputs the bus enable after receiving the bus request from external devices.

7. Refresh function

Supports the auto-refresh and self-refresh functions.

Specifies the refresh interval using the refresh counter and clock selection

Can execute concentrated refresh by specifying the refresh counts (1, 2, 4, 6, or 8)

Notes: 1. Some CSn arias do not support the some memory interfaces. For the sorts of memory interfaces that are supported by each area, see section 9.3, Area Overview.

2. The data bus width of the CS0 area is 16-bit fixed.

BSC functional block diagram is shown in figure 9.1.

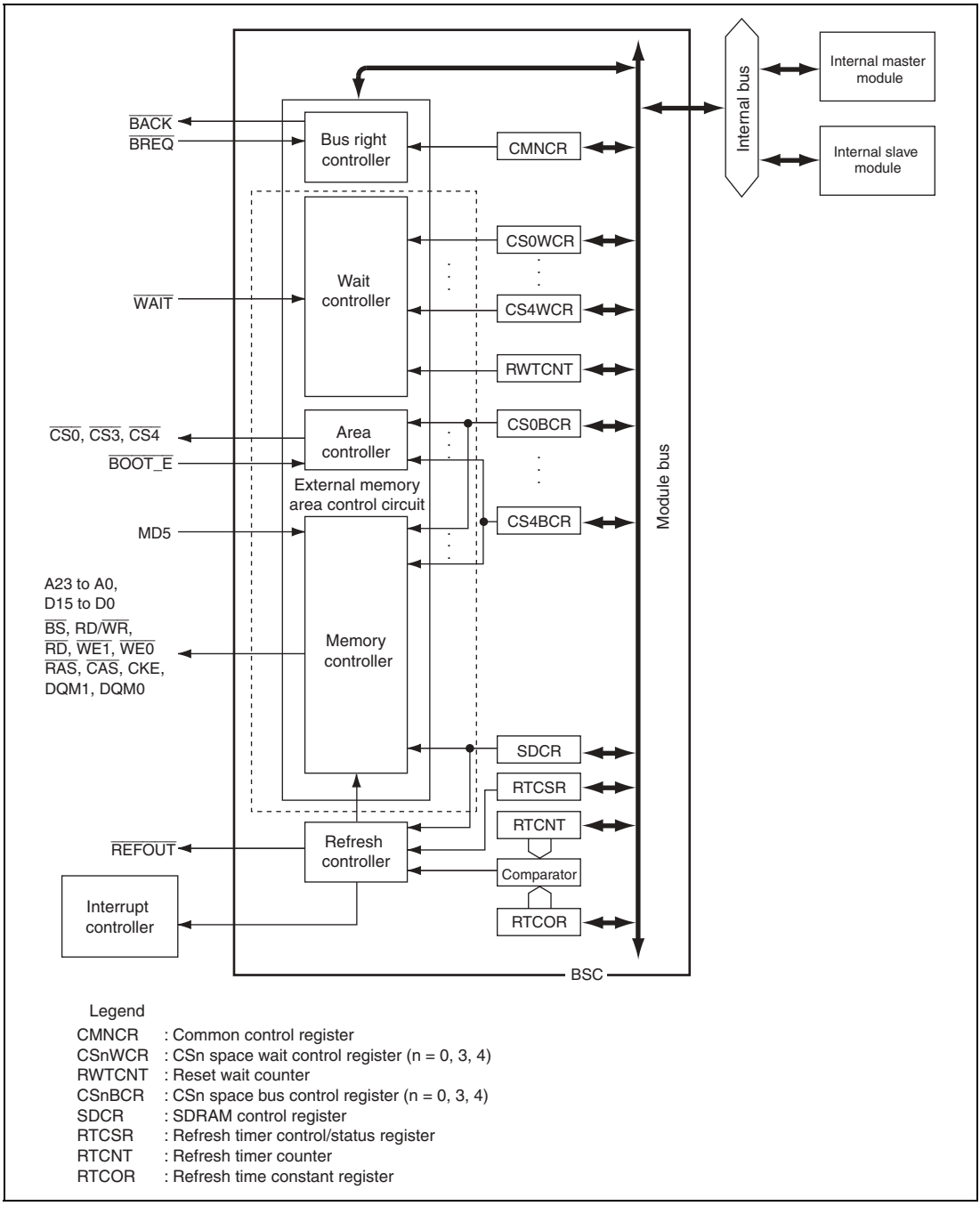


Figure 9.1 BSC Functional Block Diagram

9.2 Input/Output Pins

Table 9.1 shows the pin configuration of BSC.

Table 9.1 Pin Configuration

Name	I/O	Function
A23 to A0	O	Address Bus
D15 to D0	I/O	Data Bus
$\overline{BS}$	O	Bus Cycle Start Asserted when a normal space or burst ROM is accessed. Asserted by the same timing as $\overline{CAS}$ in SDRAM access.
$\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$	O	Chip Select
RD/$\overline{WR}$	O	Read/Write Connects to $\overline{WE}$ pins when SDRAM or byte-selection SRAM is connected.
$\overline{RD}$	O	Read Pulse Signal (read data output enable signal)
$\overline{WE1}$/DQM1	O	$\overline{WE1}$: Controls that D15 to D8 are being written to. Connected to the byte select signal when a byte-selection SRAM is connected. DQM1: Controls that D15 to D8 are being selected. Connected to the DQM1 pin when SDRAM is connected.
$\overline{WE0}$/DQM0	O	$\overline{WE0}$: Controls that D7 to D0 are being written to. Connected to the byte select signal when a byte-selection SRAM is connected. DQM0: Controls that D7 to D0 are being selected. Connected to the DQM0 pin when SDRAM is connected.
$\overline{RAS}$	O	Connects to the $\overline{RAS}$ pin when SDRAM is connected.
$\overline{CAS}$	O	Connects to the $\overline{CAS}$ pin when SDRAM is connected.
CKE	O	Connected to the CKE pin when SDRAM is connected.
$\overline{WAIT}$	I	External Wait Input
$\overline{BREQ}$	I	Bus Request Input
$\overline{BACK}$	O	Bus enable output
MD5	I	Endian Select 0: Big endian 1: Little endian
$\overline{REFOUT}$	O	Refresh request output when a bus is released
$\overline{BOOT_E}$	I	Boot Enable For details see section 17, BOOT Function (BOOT).

9.3 Area Overview

9.3.1 Area Division

In the architecture of this LSI, the logical spaces have 32-bit address spaces. The cache access method is shown by the upper three bits. For details see section 5, Cache. The remaining 29 bits are mapped in 512-Mbyte physical address space that are used for division of the space into eight areas. The BSC performs control for this 29-bit physical space. Figure 9.2 shows the mapping from the logical address space to the physical address space. Area 1 is used for the internal I/O, and area 2 and area 5 to area 7 are reserved. The other three areas (area 0, area 3, and area 4) are used as external address spaces.

As listed in table 9.2, this LSI can connect each type of memories to three area of an external address space among the physical address spaces divided into eight areas, respectively. And it outputs chip select signals ($\overline{CS0}$, $\overline{CS3}$, and $\overline{CS4}$) for each of them. $\overline{CS0}$ is asserted during area 0 access; $\overline{CS4}$ is asserted during area 4 access.

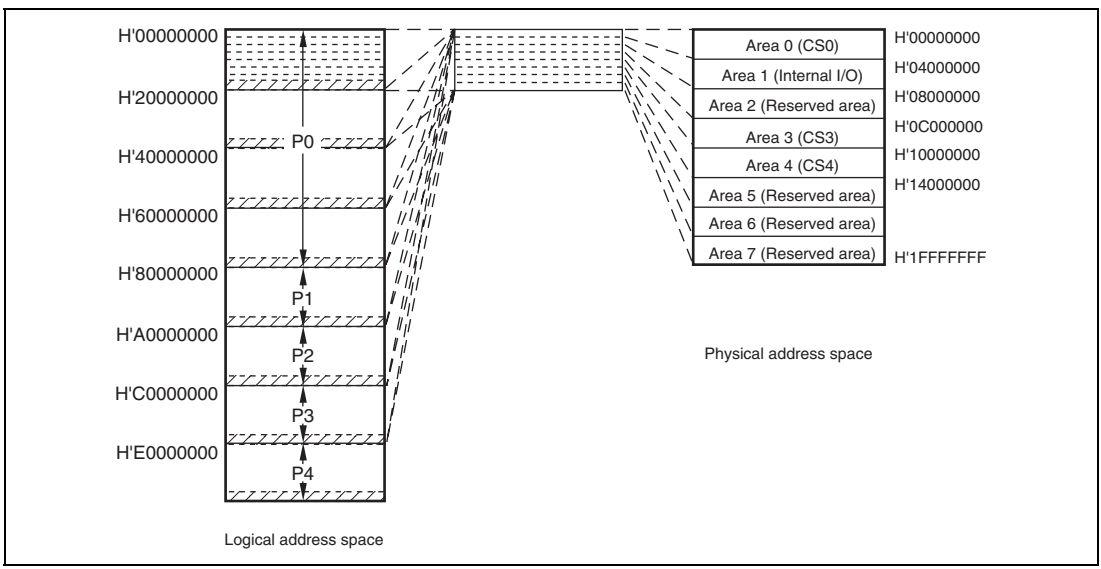


Figure 9.2 Logical Address Space and Physical Address Space

9.3.2 Shadow Area

Areas 0, 3, and 4 are decoded by physical addresses A28 to A26, which correspond to areas B'000, B'011, and B'100. Address bits A31 to A29 are ignored. This means that the range of area 0 addresses, for example, is H'00000000 to H'03FFFFFF, and its corresponding shadow space is the address space obtained by adding to it H'20000000 × n (n = 1 to 6). Addresses A25 and A24 are ignored since addresses A23 to A0 are supported for the external pins of the address bus. Accordingly, for example, H'01000000 to H'03FFFFFF in the area 0 is shadow space.

Area P4 (H'E0000000 to H'FFFFFFFF) is an internal I/O area and is assigned for internal register addresses. Area P4 does not become shadow space.

9.3.3 Address Map

The external address space has a capacity of 48 Mbytes and is used by dividing three partial spaces. The kind of memory to be connected and the data bus width are specified in each partial space. The address map for the external address space is listed in table 9.2.

Table 9.2 Address Space Map of External Address Space

Area	Memory to be Connected	Physical Address	Capacity	Access Size
Area 0 (CS0 area)	Normal memory*[1]	H'00000000 to H'00FFFFFF	16 Mbytes	8, 16, 32*[2]
	Burst ROM	H'01000000 to H'03FFFFFF	Shadow	—
		H'00000000 + H'20000000 × n to H'03FFFFFF + H'20000000 × n	Shadow	(n = 1 to 6)
Area 1	Internal I/O	H'04000000 to H'07FFFFFF		
		H'04000000 + H'20000000 × n to H'07FFFFFF + H'20000000 × n		(n = 1 to 6)
Area 2	Reserved area*[4]	H'08000000 + H'20000000 × n to H'0BFFFFFF + H'20000000 × n		(n = 1 to 6)
Area 3 (CS3 area)	Normal memory*[1]	H'0C000000 to H'0CFFFFFF	16 Mbytes	8, 16, 32*[3]
	Byte-selection SRAM	H'0D000000 to H'0FFFFFFF	Shadow	
	SDRAM	H'0C000000 + H'20000000 × n to H'0FFFFFFF + H'20000000 × n	Shadow	(n = 1 to 6)
Area 4 (CS4 area)	Normal memory*[1]	H'10000000 to H'10FFFFFF	16 Mbytes	8, 16, 32*[3]
	Burst ROM	H'11000000 to H'13FFFFFF	Shadow	
	Byte-selection SRAM	H'10000000 + H'20000000 × n to H'13FFFFFF + H'20000000 × n	Shadow	(n = 1 to 6)
Area 5	Reserved area*[4]	H'14000000 + H'20000000 × n to H'17FFFFFF + H'20000000 × n		(n = 1 to 6)
Area 6	Reserved area*[4]	H'18000000 + H'0000000 × n to H'1BFFFFFF + H'20000000 × n		(n = 0 to 6)
Area 7	Reserved area*[4]	H'1C000000 + H'20000000 × n to H'1FFFFFFF + H'20000000 × n		(n = 0 to 7)

Notes: 1. Normal memory can be connected to the external device which has an interface, such as the SRAM and ROM.

2. In the area 0 (selected when the chip select signal $\overline{CS0}$ is asserted), the data bus width is 16-bit fixed.

3. The data bus width is set by the CSn space bus control register (CSnBCR). The data bus width should be 16 bits when SDRAM is connected in area 3.

4. Do not access the reserved area. If the reserved area is accessed, the correct operation cannot be guaranteed.

9.3.4 Data Bus Width

The data bus width of areas 3 and 4 is set 8 bits or 16 bits by the CSn space bus control register (CSnBCR, n = 3 or 4). The data bus width of area 0 is fixed 16 bits. For details, see section 9.4.2, CSn Space Bus Control Register.

9.4 Register Descriptions

The BSC has the following registers. Refer to section 27, List of Registers, for the register addresses and the register states in each operating mode.

Do not access spaces other than CS0 until the termination of the setting the memory interface. When accessing spaces other than CS0, set the CSn space bus control register (CSnBCR) which corresponds to the area to be accessed before setting the CSn space wait control register (CSnWCR).

- Common control register (CMNCR)
- CS0 space bus control register (CS0BCR)
- CS3 space bus control register (CS3BCR)
- CS4 space bus control register (CS4BCR)
- CS0 space wait control register (CS0WCR)
- CS3 space wait control register (CS3WCR)
- CS4 space wait control register (CS4WCR)
- SDRAM control register (SDCR)
- Refresh timer control/status register (RTCSR)
- Refresh timer counter (RTCNT)
- Refresh time constant register (RTCOR)
- Reset wait counter (RWTCNT)
- SDRAM mode register for CS3 space (SDMR3)*

Note: * Substance of this register is in the SDRAM, and can be written to by accessing the area of this register. For details, see Power-On Sequence in section 9.5.5, SDRAM Interface.

9.4.1 Common Control Register (CMNCR)

CMNCR is a 32-bit register that controls the common items for each area. CMNCR is initialized by a power-on reset, but not by a manual reset or in standby mode and holds the previous value. Do not access external memory other than area 0 until the CMNCR register initialization is complete.

Bit	Bit Name	Initial Value	R/W	Description
31 to 8	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
7	DMAIW1	0	R/W	Wait states between access cycles when DMA single address transfer is performed Specify the number of idle cycles to be inserted after an access to an external device with DACK when DMA single address transfer is performed. The method of inserting idle cycles depends on the contents of DMAIWA. 00: No idle cycle inserted 01: 1 idle cycle inserted 10: 2 idle cycles inserted 11: 4 idle cycled inserted
6	DMAIW0	0	R/W	
5	DMAIWA	0	R/W	Method of inserting wait states between access cycles when DMA single address transfer is performed Specifies the method of inserting the idle cycles specified by the DMAIW1 and DMAIW0 bits. Clearing this bit will make this LSI insert the idle cycles when another device, which includes this LSI, drives the data bus after an external device with DACK drove it. Setting this bit will make this LSI insert the idle cycles even when the continuous accesses to an external device with DACK are performed. 0: Idle cycle is inserted when other device drives the data bus after an external device with DACK drives the data bus. 1: Idle cycle is inserted every time when an external device with DACK is accessed.
4	—	1	R	Reserved This bit is always read as 1. The write value should always be 1. When 0 is written to this bit, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
3	ENDIAN	0/1*	R	Endian Flag Samples the external pin for specifying endian on power-on reset (MD5) and specifies the pin value as the initial value. All address spaces are defined by this bit. This is a read-only bit. 0: The external pin for specifying endian (MD5) was low level on power-on reset. This LSI is being operated as big endian. 1: The external pin for specifying endian (MD5) was high level on power-on reset. This LSI is being operated as little endian.
2	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
1	HIZMEM	0	R/W	High-Z Memory Control Specifies the pin state in standby mode for A23 to A0, $\overline{BS}$, $\overline{CSn}$, RD/$\overline{WR}$, $\overline{WEn}$/DQMn, and $\overline{RD}$. 0: High impedance in standby mode. 1: Driven in standby mode
0	HIZCNT	0	R/W	High-Z Control Specifies the state in standby mode and bus released for CKIO, CKE, $\overline{RAS}$, and $\overline{CAS}$. 0: High impedance in standby mode and bus released for CKIO, CKE, $\overline{RAS}$, and $\overline{CAS}$. 1: Driven in standby mode and bus released for CKIO, CKE, $\overline{RAS}$, and $\overline{CAS}$.

Note: * The external pin for specifying endian (MD5) is sampled on power-on reset. When big endian is specified, this bit is read as 0 and when little endian is specified, this bit is read as 1.

9.4.2 CSn Space Bus Control Register (CSnBCR) (n=0, 3, 4)

CSnBCR is a 32-bit readable/writable register that specifies the memory to be connected to each area, the number of idle cycles between bus cycles, and the bus-width.

CSnBCR is initialized by a power-on reset, but not by a manual reset or in standby mode and holds the previous value. Do not access external memory other than area 0 until CSnBCR register initialization is completed.

Bit	Bit Name	Initial Value	R/W	Description
31	—	All 0	R	Reserved
30				This bit is always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
29	IWW1	1	R/W	Idle Cycles between Write-read Cycles and Write-write Cycles*[1]
28	IWW0	1	R/W	These bits specify the number of idle cycles to be inserted after the access to a memory that is connected to the space. The target access cycles are the write-read cycle and write-write cycle. 00: No idle cycle inserted 01: 1 idle cycle inserted 10: 2 idle cycles inserted 11: 4 idle cycles inserted
27	—	0	R	Reserved
				This bit is always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
26	IWRWD1	1	R/W	Idle Cycles for Another Space Read-Write*[1]
25	IWRWD0	1	R/W	Specify the number of idle cycles to be inserted after the access to a memory that is connected to the space. The target access cycle is a read-write one in which continuous accesses switch between different spaces. 00: No idle cycle inserted 01: 1 idle cycle inserted 10: 2 idle cycles inserted 11: 4 idle cycles inserted

Bit	Bit Name	Initial Value	R/W	Description
24	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
23	IWRWS1	1	R/W	Idle Cycles for Read-Write in the Same Space*[1]
22	IWRWS0	1	R/W	Specify the number of idle cycles to be inserted after the access to a memory that is connected to the space. The target cycle is a read-write cycle of which continuous accesses are for the same space. 00: No idle cycle inserted 01: 1 idle cycle inserted 10: 2 idle cycles inserted 11: 4 idle cycles inserted
21	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
20	IWRRD1	1	R/W	Idle Cycles for Read-Read in Another Space*[1]
19	IWRRD0	1	R/W	Specify the number of idle cycles to be inserted after the access to a memory that is connected to the space. The target cycle is a read-read cycle of which continuous accesses switch between different spaces. 00: No idle cycle inserted 01: 1 idle cycle inserted 10: 2 idle cycles inserted 11: 4 idle cycles inserted
18	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
17	IWRRS1	1	R/W	Idle Cycles for Read-Read in the Same Space
16	IWRRS0	1	R/W	Specify the number of idle cycles to be inserted after the access to a memory that is connected to the space. The target cycle is a read-read cycle of which continuous accesses are for the same space. 00: No idle cycle inserted 01: 1 idle cycle inserted 10: 2 idle cycles inserted 11: 4 idle cycles inserted
15	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
14	TYPE2	0	R/W	Memory Type*[2]
13	TYPE1	0	R/W	Specify the type of memory connected to a space.
12	TYPE0	0	R/W	000: Normal space 001: Burst ROM 010: Setting prohibited 011: Byte-selection SRAM 100: SDRAM 101: Setting prohibited 110: Setting prohibited 111: Setting prohibited For details for memory type in each area, refer to table 9.2.
11	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
10	BSZ1	1	R/W	Data Bus Size*[3]
9	BSZ0	0/1	R/W	Specify the data bus sizes of spaces. The data bus sizes of areas 3 and 4 are shown below. 00: Setting prohibited 01: 8-bit size 10: 16-bit size 11: Setting prohibited Note: The initial values of CS0BCR and other CSnBCR are 10 and 11, respectively. The valid value should be set. The data bus width of the area 0 is fixed 16 bits. Other settings are ignored.
8 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

Notes: 1. In actual operation, same idle cycles might be inserted by factors other than the specification of this bit.
2. If an unappropriated memory is specified to be connected to a space, correct operation cannot be guaranteed.
3. The initial data bus sizes of other than area 0 after a reset cannot be set. Therefore, the data bus sizes should be set 8 or 16 bits before the areas are accessed.

9.4.3 CSn Space Wait Control Register (CSnWCR) (n=0, 3, 4)

CSnWCR is a 32-bit register which specifies various wait cycles for memory accesses. The bit configuration of this register varies as shown below according to the memory type specified by the CSn space bus control register (CSnBCR). Specify CSnBCR register first, then specify the CSnWCR register.

CSnWCR is initialized by a power-on reset, but not by a manual reset or in standby mode and holds the previous value.

Normal Space, Byte-Selection SRAM:

- CS0WCR

Bit	Bit Name	Initial Value	R/W	Description
31 to 13	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. Is 1 is written to these bits, correct operation cannot be guaranteed.
12	SW1	0	R/W	Number of Delay Cycles from Address, $\overline{CSn}$ Assertion to $\overline{RD}$, $\overline{WEn}$ Assertion Specify the number of delay cycles from address and $\overline{CSn}$ assertion to $\overline{RD}$ and $\overline{WEn}$ assertion. 00: 0.5 cycles 01: 1.5 cycles 10: 2.5 cycles 11: 3.5 cycles
11	SW0	0	R/W	

Bit	Bit Name	Initial Value	R/W	Description
10	WR3	1	R/W	Number of Access Wait Cycles
9	WR2	0	R/W	Specify the number of cycles that are necessary for read/write access.
8	WR1	1	R/W	
7	WR0	0	R/W	0000: 0 cycles
				0001: 1 cycle
				0010: 2 cycles
				0011: 3 cycles
				0100: 4 cycles
				0101: 5 cycles
				0110: 6 cycles
				0111: 8 cycles
				1000: 10 cycles
				1001: 12 cycles
				1010: 14 cycles
				1011: 18 cycles
				1100: 24 cycles
				1101: Setting prohibited
				1110: Setting prohibited
				1111: Setting prohibited
6	WM	0	R/W	External Wait Mask Specification
				Specifies whether or not the external wait input is valid. The specification by this bit is valid even when the number of access wait cycle is 0.
				0: External wait input is valid
				1: External wait input is ignored
5 to 2	—	All 0	R	Reserved
				These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
1	HW1	0	R/W	Delay Cycles from $\overline{RD}$, $\overline{WEn}$ negation to Address, $\overline{CSn}$ negation
0	HW0	0	R/W	Specify the number of delay cycles from $\overline{RD}$ and $\overline{WEn}$ negation to address and $\overline{CSn}$ negation.
				00: 0.5 cycles
				01: 1.5 cycles
				10: 2.5 cycles
				11: 3.5 cycles

- CS3WCR

Bit	Bit Name	Initial Value	R/W	Description
31 to 21	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
20	BAS	0	R/W	Byte-Selection SRAM Byte Access Selection Specifies the $\overline{WEn}$ and RD/$\overline{WR}$ signal timing when the byte-selection SRAM interface is used. 0: Asserts the $\overline{WEn}$ signal at the read/write timing and asserts the RD/$\overline{WR}$ signal during the write access cycle. 1: Asserts the $\overline{WEn}$ signal during the read access cycle and asserts the RD/$\overline{WR}$ signal at the write timing.
19 to 11	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
10	WR3	1	R/W	Number of Access Wait Cycles
9	WR2	0	R/W	Specify the number of cycles that are necessary for read/write access.
8	WR1	1	R/W	
7	WR0	0	R/W	0000: 0 cycles 0001: 1 cycle 0010: 2 cycles 0011: 3 cycles 0100: 4 cycles 0101: 5 cycles 0110: 6 cycles 0111: 8 cycles 1000: 10 cycles 1001: 12 cycles 1010: 14 cycles 1011: 18 cycles 1100: 24 cycles 1101: Setting prohibited 1110: Setting prohibited 1111: Setting prohibited

Bit	Bit Name	Initial Value	R/W	Description
6	WM	0	R/W	External Wait Mask Specification Specify whether or not the external wait input is valid. The specification by this bit is valid even when the number of access wait cycle is 0. 0: External wait input is valid 1: External wait input is ignored
5 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

- CS4WCR

Bit	Bit Name	Initial Value	R/W	Description
31 to 21	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
20	BAS	0	R/W	Byte-Selection SRAM Byte Access Selection Specifies the $\overline{\text{WE}}$n and RD/$\overline{\text{WR}}$ signal timing when the byte-selection SRAM interface is used. 0: Asserts the $\overline{\text{WE}}$n signal at the read timing and asserts the RD/$\overline{\text{WR}}$ signal during the write access cycle. 1: Asserts the $\overline{\text{WE}}$n signal during the read access cycle and asserts the RD/$\overline{\text{WR}}$ signal at the write timing.
19	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
18	WW2	0	R/W	Number of Write Access Wait Cycles
17	WW1	0	R/W	Specify the number of cycles that are necessary for write access.
16	WW0	0	R/W	000: The same cycles as WR3 to WR0 setting (read access wait) 001: 0 cycles 010: 1 cycle 011: 2 cycles 100: 3 cycles 101: 4 cycles 110: 5 cycles 111: 6 cycles
15 to 13	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
12	SW1	0	R/W	Number of Delay Cycles from Address, $\overline{CSn}$ Assertion to $\overline{RD}$, $\overline{WEn}$ Assertion
11	SW0	0	R/W	Specify the number of delay cycles from address and $\overline{CSn}$ assertion to $\overline{RD}$ and $\overline{WEn}$ assertion. 00: 0.5 cycles 01: 1.5 cycles 10: 2.5 cycles 11: 3.5 cycles

Bit	Bit Name	Initial Value	R/W	Description
10	WR3	1	R/W	Number of Access Wait Cycles
9	WR2	0	R/W	Specify the number of cycles that are necessary for read access.
8	WR1	1	R/W	
7	WR0	0	R/W	0000: 0 cycles
				0001: 1 cycle
				0010: 2 cycles
				0011: 3 cycles
				0100: 4 cycles
				0101: 5 cycles
				0110: 6 cycles
				0111: 8 cycles
				1000: 10 cycles
				1001: 12 cycles
				1010: 14 cycles
				1011: 18 cycles
				1100: 24 cycles
				1101: Setting prohibited
				1110: Setting prohibited
				1111: Setting prohibited
6	WM	0	R/W	External Wait Mask Specification
				Specifies whether or not the external wait input is valid. The specification by this bit is valid even when the number of access wait cycle is 0.
				0: External wait input is valid
				1: External wait input is ignored
5 to 2	—	All 0	R	Reserved
				These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
1	HW1	0	R/W	Delay Cycles from $\overline{RD}$, $\overline{WEn}$ negation to Address, $\overline{CSn}$ negation
0	HW0	0	R/W	
				Specify the number of delay cycles from $\overline{RD}$ and $\overline{WEn}$ negation to address and $\overline{CSn}$ negation.
				00: 0.5 cycles
				01: 1.5 cycles
				10: 2.5 cycles
				11: 3.5 cycles

Burst ROM:

- CS0WCR

Bit	Bit Name	Initial Value	R/W	Description
31 to 18	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
17	BW1	0	R/W	Number of Burst Wait Cycles
16	BW0	0	R/W	Specify the number of wait cycles to be inserted between the second or later access cycles in burst access. 00: 0 cycles 01: 1 cycle 10: 2 cycles 11: 3 cycles
15 to 11	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
10	W3	1	R/W	Number of Access Wait Cycles
9	W2	0	R/W	Specify the number of wait cycles to be inserted in the write cycle and the first read cycle.
8	W1	1	R/W	
7	W0	0	R/W	0000: 0 cycles 0001: 1 cycle 0010: 2 cycles 0011: 3 cycles 0100: 4 cycles 0101: 5 cycles 0110: 6 cycles 0111: 8 cycles 1000: 10 cycles 1001: 12 cycles 1010: 14 cycles 1011: 18 cycles 1100: 24 cycles 1101: Setting prohibited 1110: Setting prohibited 1111: Setting prohibited

Bit	Bit Name	Initial Value	R/W	Description
6	WM	0	R/W	External Wait Mask Specification Specify whether or not the external wait input is valid. The specification by this bit is valid even when the number of access wait cycle is 0. 0: External wait input is valid 1: External wait input is ignored
5 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

- CS4WCR

Bit	Bit Name	Initial Value	R/W	Description
31 to 18	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
17	BW1	0	R/W	Number of Burst Wait Cycles
16	BW0	0	R/W	Specify the number of wait cycles to be inserted between the second or later access cycles in burst access. 00: 0 cycles 01: 1 cycle 10: 2 cycles 11: 3 cycles
15 to 13	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
12	SW1	0	R/W	Number of Delay Cycles from Address, $\overline{\text{CSn}}$ Assertion to $\overline{\text{RD}}$, $\overline{\text{WE}}$ Assertion
11	SW0	0	R/W	Specify the number of delay cycles from address and $\overline{\text{CSn}}$ assertion to $\overline{\text{RD}}$ and $\overline{\text{WE}}$ assertion. 00: 0.5 cycles 01: 1.5 cycles 10: 2.5 cycles 11: 3.5 cycles

Bit	Bit Name	Initial Value	R/W	Description
10	W3	1	R/W	Number of Access Wait Cycles
9	W2	0	R/W	Specify the number of wait cycles to be inserted in the write cycle and the first read cycle.
8	W1	1	R/W	
7	W0	0	R/W	0000: 0 cycles
				0001: 1 cycle
				0010: 2 cycles
				0011: 3 cycles
				0100: 4 cycles
				0101: 5 cycles
				0110: 6 cycles
				0111: 8 cycles
				1000: 10 cycles
				1001: 12 cycles
				1010: 14 cycles
				1011: 18 cycles
				1100: 24 cycles
				1101: Setting prohibited
				1110: Setting prohibited
				1111: Setting prohibited
6	WM	0	R/W	External Wait Mask Specification
				Specifies whether or not the external wait input is valid. The specification by this bit is valid even when the number of access wait cycle is 0.
				0: External wait input is valid
				1: External wait input is ignored
5 to 2	—	All 0	R	Reserved
				These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
1	HW1	0	R/W	Delay Cycles from $\overline{RD}$, $\overline{WEn}$ negation to Address, $\overline{CSn}$ negation
0	HW0	0	R/W	
				Specify the number of delay cycles from $\overline{RD}$ and $\overline{WEn}$ negation to address and $\overline{CSn}$ negation.
				00: 0.5 cycles
				01: 1.5 cycles
				10: 2.5 cycles
				11: 3.5 cycles

SDRAM:

- CS3WCR

Bit	Bit Name	Initial Value	R/W	Description
31 to 15	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
14	TRP1	0	R/W	Number of Cycles from Auto-precharge/PRE Command to ACTV Command Specify the number of minimum cycles from the start of auto-precharge or issuing of PRE command to the issuing of ACTV command for the same bank. 00: 1 cycle 01: 2 cycles 10: 3 cycles 11: 4 cycles
13	TRP0	0	R/W	
12	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
11	TRCD1	0	R/W	Number of Cycles from ACTV Command to READ(A)/WRIT(A) Command Specify the number of minimum cycles from issuing ACTV command to issuing READ(A)/WRIT(A) command. 00: 1 cycle 01: 2 cycles 10: 3 cycles 11: 4 cycles
10	TRCD0	1	R/W	
9	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
8	A3CL1	1	R/W	CAS Latency
7	A3CL0	0	R/W	Specify the CAS latency. 00: 1 cycle 01: 2 cycles 10: 3 cycles 11: 4 cycles
6, 5	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
4	TRWL1	0	R/W	Number of Cycles from WRITA/WRIT Command to Auto-precharge/PRE Command
3	TRWL0	0	R/W	Specifies the number of cycles from issuing WRITA/WRIT command to the start of auto-precharge or to issuing PRE command. 00: 0 cycles 01: 1 cycle 10: 2 cycles 11: 3 cycles
2	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
1	TRC1	0	R/W	Number of Cycles from REF Command/Self-refresh Release to ACTV Command
0	TRC0	0	R/W	Specify the number of cycles from issuing the REF command or releasing self-refresh to issuing the ACTV command.. 00: 3 cycles 01: 4 cycles 10: 6 cycles 11: 9 cycles

9.4.4 SDRAM Control Register (SDCR)

SDCR is a 32-bit register that specifies the method to refresh and access SDRAM, and the types of SDRAMs to be connected.

SDCR is initialized by a power-on reset, but not by a manual reset or in standby mode and holds the previous value. Bits other than RFSH and RMODE should be written to at the initialization settings and not be changed afterward. When writing to RFSH and RMODE, the values before writing should be written to the other bits. The area 3 should not be accessed until this register setting has been completed while the synchronous DRAM is used.

Bit	Bit Name	Initial Value	R/W	Description
31 to 13	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
12	SLOW	0	R/W	Low-Frequency Mode Specifies the output timing of command, address, and write data for SDRAM and the latch timing of read data from SDRAM. Setting this bit makes the hold time for command, address, write and read data. This mode is suitable for SDRAM with low-frequency clock. 0: Command, address, and write data for SDRAM is output at the rising edge of CKIO. Read data from SDRAM is latched at the rising edge of CKIO. 1: Command, address, and write data for SDRAM is output at the falling edge of CKIO. Read data from SDRAM is latched at the falling edge of CKIO.
11	RFSH	0	R/W	Refresh Control Specifies whether or not the refresh operation of the SDRAM is performed. 0: No refresh 1: Refresh

Bit	Bit Name	Initial Value	R/W	Description
10	RMODE	0	R/W	Refresh Control Specifies whether to perform auto-refresh or self-refresh when the RFSH bit is 1. When the RFSH bit is 1 and this bit is 1, self-refresh starts immediately. When the RFSH bit is 1 and this bit is 0, auto-refresh starts according to the contents that are set in registers RTCSR, RTCNT, and RTCOR. 0: Auto-refresh is performed 1: Self-refresh is performed
9	PDOWN	0	R/W	Power-Down Mode Specifies whether the SDRAM is entered in power-down mode or not after the memory access other than SDRAM is completed. If this bit is set to 1, the CKE pin is pulled to low to place the SDRAM to power-down mode triggered by memory access other than SDRAM after the SDRAM access ends. 0: Does not place the SDRAM in power-down mode. 1: Places the SDRAM in power-down mode.
8	BACTV	0	R/W	Bank Active Mode Specifies to access whether in auto-precharge mode (using READA and WRITA commands) or in bank active mode (using READ and WRIT commands). 0: Auto-precharge mode (using READA and WRITA commands) 1: Bank active mode (using READ and WRIT commands) Note: When bank active mode is specified, the data bus width should be set 16 bits.
7 to 5	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
4	A3ROW1	0	R/W	Number of Bits of Row Address
3	A3ROW0	0	R/W	Specify the number of bits of the row address. 00: 11 bits 01: 12 bits 10: 13 bits 11: Reserved (setting prohibited)

Bit	Bit Name	Initial Value	R/W	Description
2	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
1	A3COL1	0	R/W	Number of Bits of Column Address
0	A3COL0	0	R/W	Specify the number of bits of the column address. 00: 8 bits 01: 9 bits 10: 10 bits 11: Setting prohibited

9.4.5 Refresh Timer Control/Status Register (RTCSR)

RTCSR is a 32-bit readable/writable register that specifies various items about refresh for SDRAM. When the RTCSR is written, the upper 16 bits of the write data must be H′A55A to cancel write protection.

Bit	Bit Name	Initial Value	R/W	Description
31 to 8	—	All 0	R	Write Protection These bits are always read as 0. When writing to these bits, the upper 16-bit of the write data should be H′A55A to cancel write protection. The lower 8-bit is should always be 0. If other value is written to these bits, correct operation cannot be guaranteed.
7	CMF	0	R/W	Compare Match Flag Indicates that a compare match occurs between the refresh timer counter (RTCNT) and refresh time constant register (RTCOR). This bit is set or cleared in the following conditions. 0: Clearing condition: When 0 is written in CMF after reading out RTCSR during CMF = 1. 1: Setting condition: When the condition RTCNT = RTCOR is satisfied.
6	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
5	CKS2	0	R/W	Clock Select Select the clock input to count-up the refresh timer counter (RTCNT). 000: Stop the counting-up 001: Bϕ/4 010: Bϕ/16 011: Bϕ/64 100: Bϕ/256 101: Bϕ/1024 110: Bϕ/2048 111: Bϕ/4096
4	CKS1	0	R/W	
3	CKS0	0	R/W	

Bit	Bit Name	Initial Value	R/W	Description
2	RRC2	0	R/W	Refresh Count
1	RRC1	0	R/W	Specify the number of continuous refresh cycles, when the refresh request occurs after the coincidence of the values of the refresh timer counter (RTCNT) and the refresh time constant register (RTCOR). These bits can make the period of occurrence of refresh long.
0	RRC0	0	R/W	000: Once 001: Twice 010: 4 times 011: 6 times 100: 8 times 101: Setting prohibited 110: Setting prohibited 111: Setting prohibited

9.4.6 Refresh Timer Counter (RTCNT)

RTCNT is an 8-bit counter register that increments using the clock selected by bits CKS2 to CKS0 in RTCSR. When RTCNT matches RTCOR, RTCNT is cleared to 0. The value in RTCNT returns to 0 after counting up to 255. When the RTCNT is written, the upper 16 bits of the write data must be H'A55A to cancel write protection.

Bit	Bit Name	Initial Value	R/W	Description
31 to 8	—	All 0	R/W	Write Protection These bits are always read as 0. When writing to these bits, the upper 16-bit of the write data should be H'A55A to cancel write protection. The lower 8-bit is should always be 0. If other value is written to these bits, correct operation cannot be guaranteed.
7 to 0	—	All 0	R/W	8-bit Counter

9.4.7 Refresh Time Constant Register (RTCOR)

RTCOR is a 32-bit register. When RTCOR matches RTCNT, the CMF bit in RTCSR is set to 1 and RTCNT is cleared to 0.

When the RFSH bit in SDCR is 1, a memory refresh request is issued by this matching signal. This request is maintained until the refresh operation is performed. If the request is not processed when the next matching occurs, the previous request is ignored.

When the RTCOR is written, the upper 16 bits of the write data must be H'A55A to cancel write protection.

Bit	Bit Name	Initial Value	R/W	Description
31 to 8	—	All 0	R	Write Protection These bits are always read as 0. When writing to these bits, the upper 16-bit of the write data should be H′A55A to cancel write protection. The lower 8-bit is should always be 0. If other value is written to these bits, correct operation cannot be guaranteed.
7 to 0	—	All 0	R/W	Refresh Time Specify the upper limit value of the refresh timer counter (RTCNT).

9.4.8 Reset Wait Counter (RWTCNT)

RWTCNT is a 7-bit counter. This counter starts to increment by synchronizing the CKIO after a power-on reset is released, and stops when the value reaches H'007F. External bus access is suspended while the counter is operating. This counter is provided to minimize the time from releasing a reset for flash memory to the first access. This register cannot be read or written to.

9.5 Operating Description

9.5.1 Endian/Access Size and Data Alignment

This LSI supports big endian, in which the 0 address is the most significant byte (MSByte) in the byte data and little endian, in which the 0 address is the least significant byte (LSByte) in the byte data. Endian is specified on power-on reset by the external pin (MD5). When MD5 pin is low level on power-on reset, the endian will become big endian and when MD5 pin is high level on power-on reset, the endian will become little endian.

Two data bus widths (8 bits and 16 bits) are available for normal memory and byte-selection SRAM. Note that the data bus width for the area 0 is fixed 16 bits. The data bus width of SDRAM is fixed 16 bits. Data alignment is performed in accordance with the data bus width of the device and endian. This also means that when longword data is read from a byte-width device, the read operation must be done four times. In this LSI, data alignment and conversion of data length is performed automatically between the respective interfaces.

Table 9.3 through 9.6 show the relationship between endian, device data width, and access unit.

Table 9.3 16-Bit External Device/Big Endian Access and Data Alignment

		Data Bus		Strobe Signals	
Operation		**D15 to D8**	**D7 to D0**	**$\overline{WE1}$/DQM1**	**$\overline{WE0}$/DQM0**
Byte access at 0		Data 7 to 0	—	Assert	—
Byte access at 1		—	Data 7 to 0	—	Assert
Byte access at 2		Data 7 to 0	—	Assert	—
Byte access at 3		—	Data 7 to 0	—	Assert
Word access at 0		Data 15 to 8	Data 7 to 0	Assert	Assert
Word access at 2		Data 15 to 8	Data 7 to 0	Assert	Assert
Longword access at 0	1st time at 0	Data 31 to 24	Data 23 to 16	Assert	Assert
	2nd time at 2	Data 15 to 8	Data 7 to 0	Assert	Assert

Table 9.4 8-Bit External Device/Big Endian Access and Data Alignment

Operation		Data Bus		Strobe Signals	
		D15 to D8	D7 to D0	$\overline{WE1}$/DQM1	$\overline{WE0}$/DQM0
Byte access at 0		—	Data 7 to 0	—	Assert
Byte access at 1		—	Data 7 to 0	—	Assert
Byte access at 2		—	Data 7 to 0	—	Assert
Byte access at 3		—	Data 7 to 0	—	Assert
Word access at 0	1st time at 0	—	Data 15 to 8	—	Assert
	2nd time at 1	—	Data 7 to 0	—	Assert
Word access at 2	1st time at 2	—	Data 15 to 8	—	Assert
	2nd time at 3	—	Data 7 to 0	—	Assert
Longword access at 0	1st time at 0	—	Data 31 to 24	—	Assert
	2nd time at 1	—	Data 23 to 16	—	Assert
	3rd time at 2	—	Data 15 to 8	—	Assert
	4th time at 3	—	Data 7 to 0	—	Assert

Table 9.5 16-Bit External Device/Little Endian Access and Data Alignment

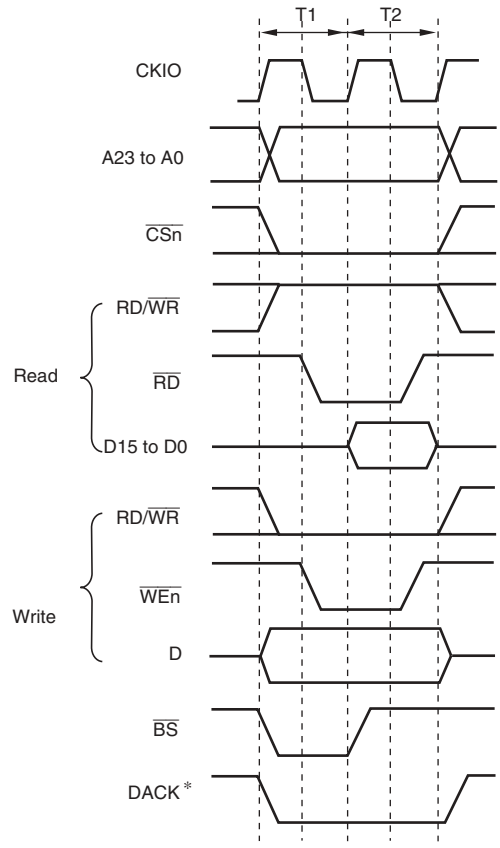
Operation		Data Bus		Strobe Signals	
		D15 to D8	D7 to D0	$\overline{WE1}$/DQM1	$\overline{WE0}$/DQM0
Byte access at 0		—	Data 7 to 0	—	Assert
Byte access at 1		Data 7 to 0	—	Assert	—
Byte access at 2		—	Data 7 to 0	—	Assert
Byte access at 3		Data 7 to 0	—	Assert	—
Word access at 0		Data 15 to 8	Data 7 to 0	Assert	Assert
Word access at 2		Data 15 to 8	Data 7 to 0	Assert	Assert
Longword access at 0	1st time at 0	Data 15 to 8	Data 7 to 0	Assert	Assert
	2nd time at 2	Data 31 to 24	Data 23 to 16	Assert	Assert

Table 9.6 8-Bit External Device/Little Endian Access and Data Alignment

Operation		Data Bus		Strobe Signals	
		D15 to D8	D7 to D0	$\overline{WE1}$/DQM1	$\overline{WE0}$/DQM0
Byte access at 0		—	Data 7 to 0	—	Assert
Byte access at 1		—	Data 7 to 0	—	Assert
Byte access at 2		—	Data 7 to 0	—	Assert
Byte access at 3		—	Data 7 to 0	—	Assert
Word access at 0	1st time at 0	—	Data 7 to 0	—	Assert
	2nd time at 1	—	Data 15 to 8	—	Assert
Word access at 2	1st time at 2	—	Data 7 to 0	—	Assert
	2nd time at 3	—	Data 15 to 8	—	Assert
Longword access at 0	1st time at 0	—	Data 7 to 0	—	Assert
	2nd time at 1	—	Data 15 to 8	—	Assert
	3rd time at 2	—	Data 23 to 16	—	Assert
	4th time at 3	—	Data 31 to 24	—	Assert

9.5.2 Normal Space Interface

Basic Timing: For access to a normal space, this LSI uses strobe signal output in consideration of the fact that mainly static RAM will be directly connected. When using SRAM with a byte-selection pin, see section 9.5.7, Byte-Selection SRAM Interface. Figure 9.3 shows the basic timings of normal space access. A no-wait normal access is completed in two cycles. The $\overline{BS}$ signal is asserted for one cycle to indicate the start of a bus cycle.



Note: * The waveform for DACK is when active low is specified.

Figure 9.3 Normal Space Basic Access Timing (Access Wait 0)

There is no access size specification when reading. The correct access start address is output in the least significant bit of the address, but since there is no access size specification, 16 bits are always read in case of a 16-bit device, and 8 bits in case of an 8-bit device. When writing, only the $\overline{\text{WE}}$n signal for the byte to be written is asserted.

Reading/writing for cache fill or copy back is performed continuously in total of 16 bytes according to the specified data bus width. During the processing, bus is not released. If a cache miss occurs in byte or word operand access or at a branch to an odd word boundary, the CPU performs longword accesses to perform a cache fill operation on the external interface. Writing to the write-through area and reading/writing to the area not to be cached are performed according to the actual access size.

It is necessary to output the data that has been read using the $\overline{\text{RD}}$ signal when a buffer is established in the data bus. The RD/$\overline{\text{WR}}$ signal is in a read state (high output) when no access has been carried out. Therefore, care must be taken when controlling the external data buffer, to avoid collision.

Figures 9.4 and 9.5 show the basic timings of normal space accesses. If the WM bit of the CSnWCR is cleared to 0, a Tnop cycle is inserted to evaluate the external wait (figure 9.4). If the WM bit of the CSnWCR is set to 1, external waits are ignored and no Tnop cycle is inserted (figure 9.5).

Figure 9.6 and figure 9.7 shows the examples of SRAM connection. Note that bits of external memory address and those of this LSI address are connected shifted when using the data in longword units (figure 9.6) and in word units (figure 9.7) since this LSI address is divided in byte units.

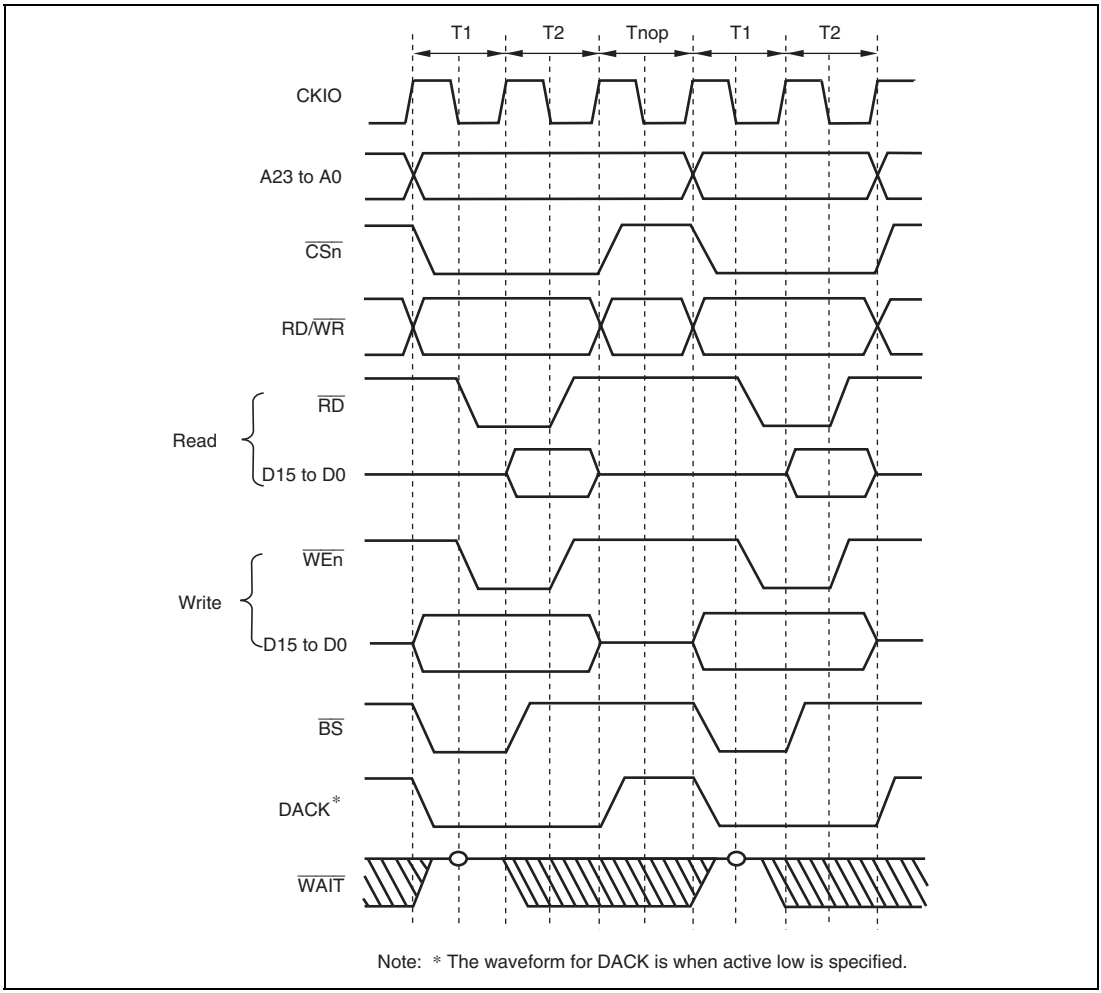


Figure 9.4 Continuous Access for Normal Space 1 Data Bus Width = 16 bits, Long-Word Access, CSnWCR.WN Bit = 0 (Access Wait = 0, Cycle Wait = 0)

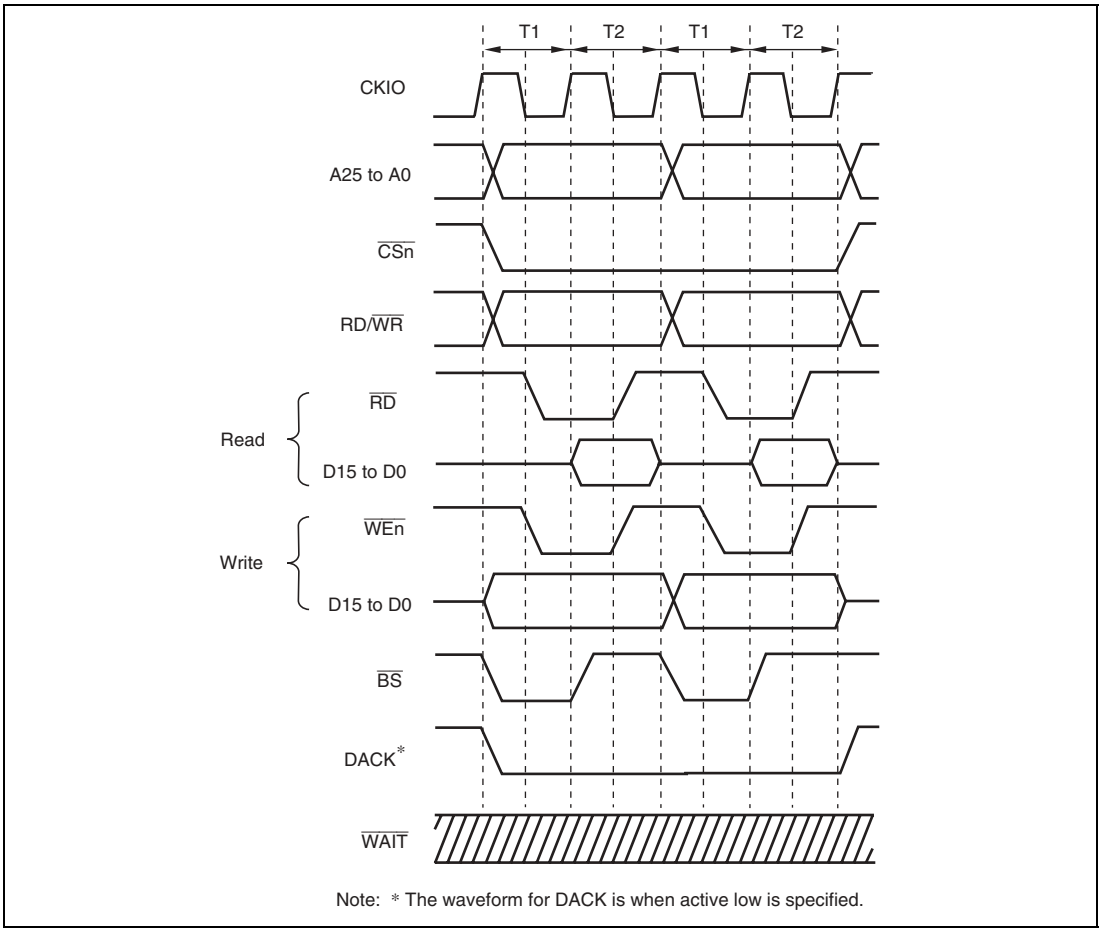


Figure 9.5 Continuous Access for Normal Space 2 Data Bus Width = 16 bits, Long-Word Access, CSnWCR.WN Bit = 1 (Access Wait = 0, Cycle Wait = 0)

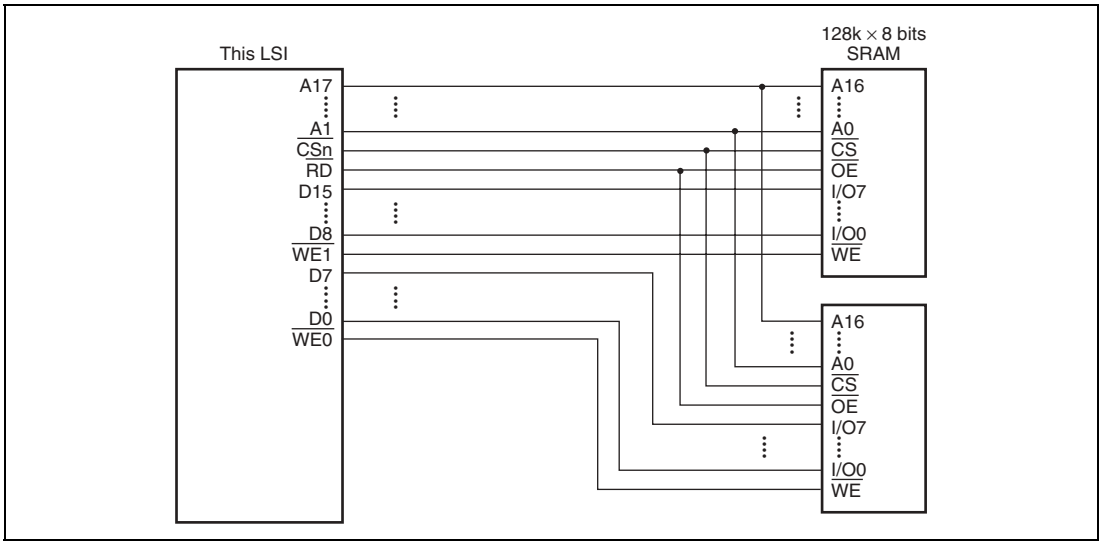


Figure 9.6 Example of 16-Bit Data-Width SRAM Connection

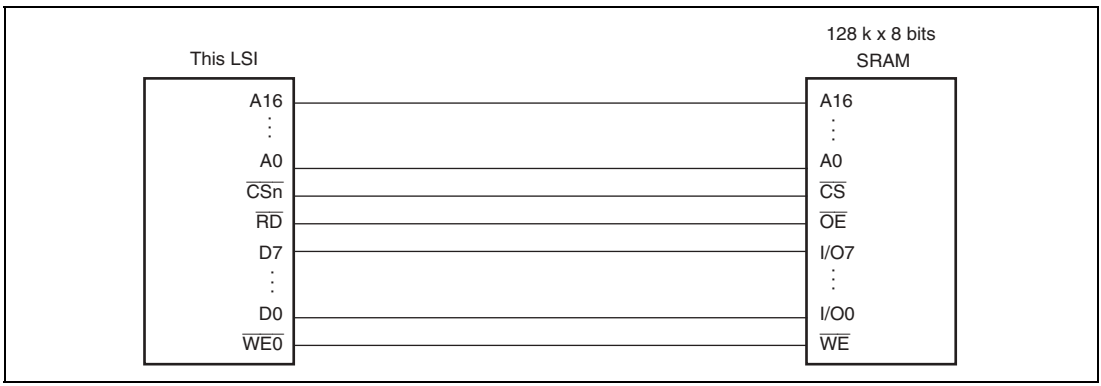


Figure 9.7 Example of 8-Bit Data-Width SRAM Connection

9.5.3 Access Wait Control

Wait cycle insertion on a normal space access can be controlled by the settings of bits WR3 to WR0 in CSnWCR. It is possible for area 4 to insert wait cycles independently in read access and in write access. The areas other than 4 have common access wait for read cycle and write cycle. The specified number of Tw cycles are inserted as wait cycles in a normal space access shown in figure 9.8.

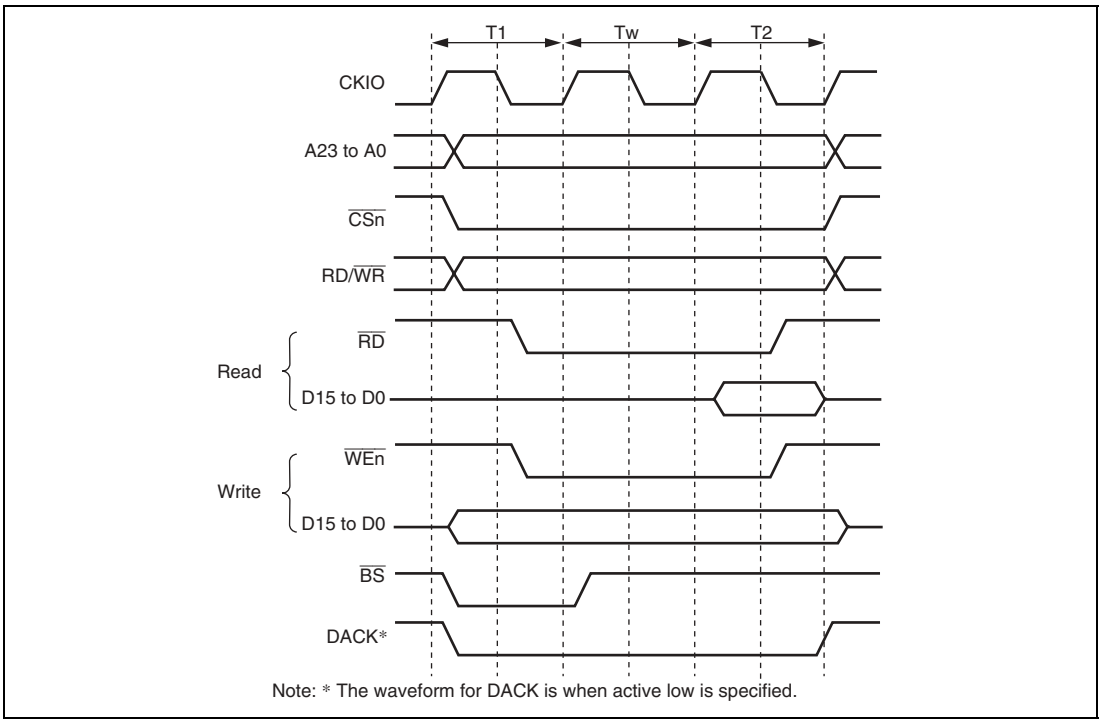


Figure 9.8 Wait Timing for Normal Space Access (Software Wait Only)

When the WM bit in CSnWCR is cleared to 0, the external wait input $\overline{\text{WAIT}}$ signal is also sampled. $\overline{\text{WAIT}}$ pin sampling is shown in figure 9.9. A 2-cycle wait is specified as a software wait. The $\overline{\text{WAIT}}$ signal is sampled on the falling edge of CKIO at the transition from the T1 or Tw cycle to the T2 cycle.

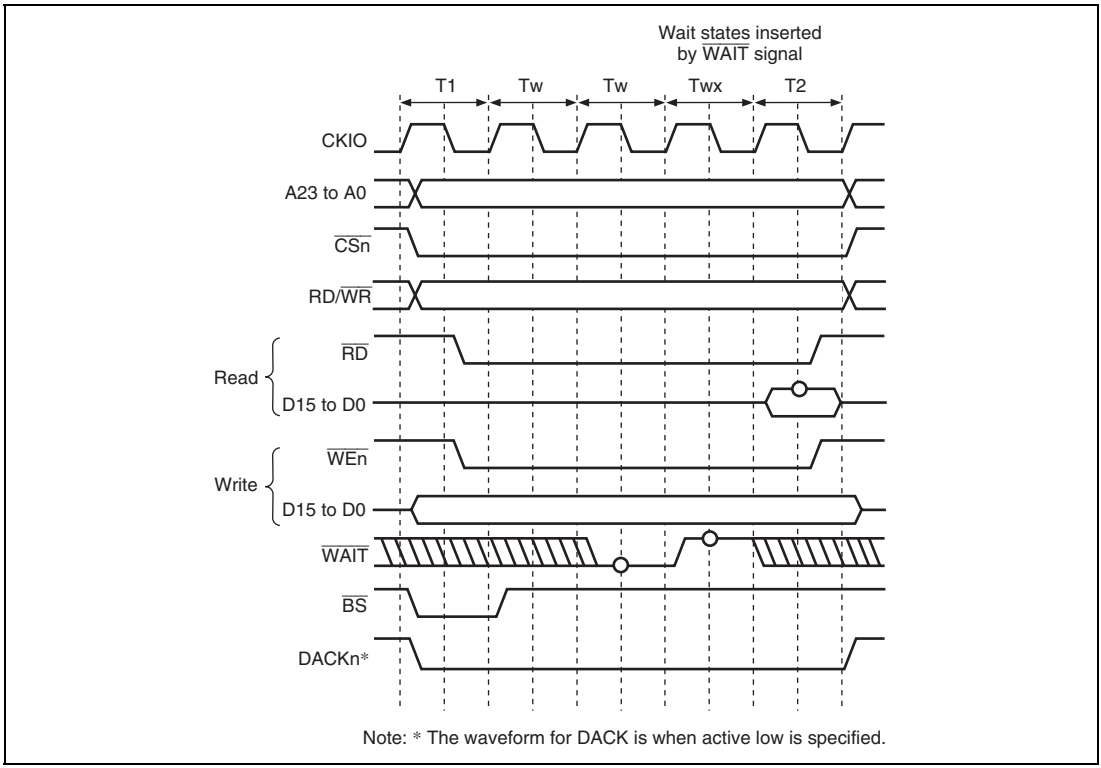


Figure 9.9 Wait State Timing for Normal Space Access (Wait State Insertion using WAIT Signal)

9.5.4 $\overline{\text{CSn}}$ Assert Period Expansion

The number of cycles from $\overline{\text{CSn}}$ assertion to $\overline{\text{RD}}$, $\overline{\text{WEn}}$ assertion can be specified by setting bits SW1 and SW0 in CSnWCR. The number of cycles from $\overline{\text{RD}}$, $\overline{\text{WEn}}$ negation to $\overline{\text{CSn}}$ negation can be specified by setting bits HW1 and HW0. Therefore, a flexible interface to an external device can be obtained. Figure 9.10 shows an example. A Th cycle and a Tf cycle are added before and after an ordinary cycle, respectively. In these cycles, $\overline{\text{RD}}$ and $\overline{\text{WEn}}$ are not asserted, while other signals are asserted. The data output is prolonged to the Tf cycle, and this prolongation is useful for devices with slow writing operations.

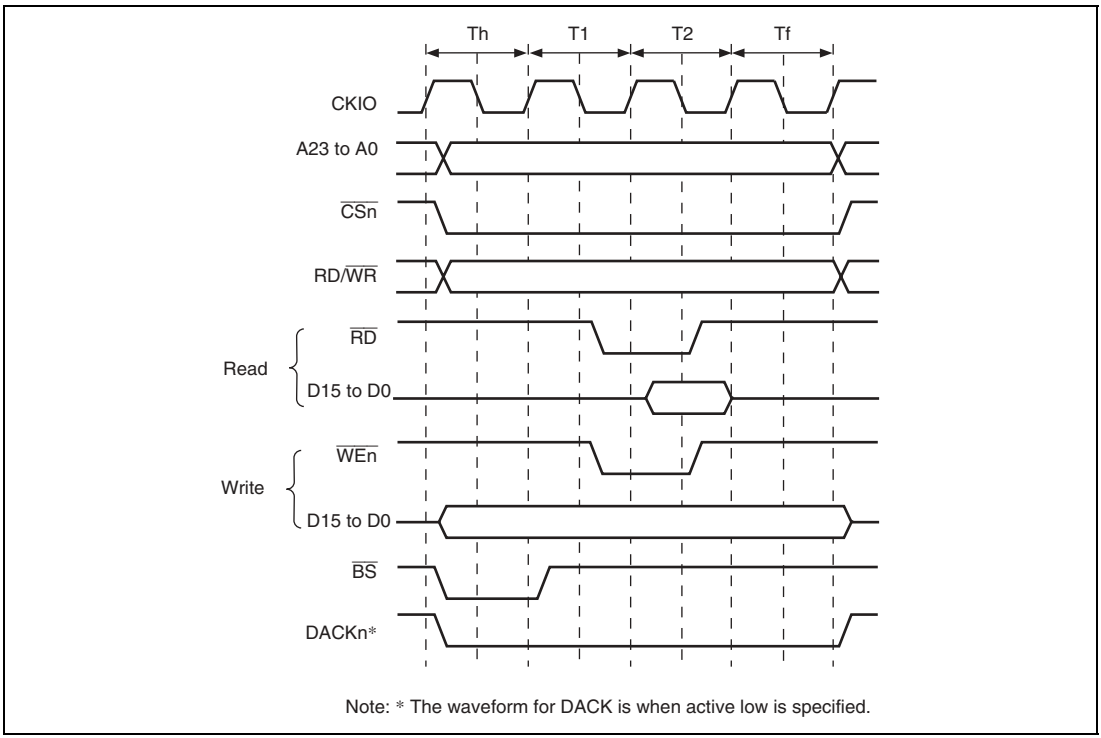


Figure 9.10 $\overline{\text{CSn}}$ Assert Period Expansion

9.5.5 SDRAM Interface

SDRAM Direct Connection: The SDRAM that can be connected to this LSI is a product that has 11/12/13 bits of row address, 8/9/10 bits of column address, 4 or less banks, and uses the A10 pin for setting precharge mode in read and write command cycles. The control signals for direct connection of SDRAM are $\overline{RAS}$, $\overline{CAS}$, RD/$\overline{WR}$, DQM1, DQM0, CKEand $\overline{CS3}$. All the signals other than $\overline{CS3}$ are common to all areas, and signals other than CKE are valid when $\overline{CS3}$ is asserted. The data bus width of the area that is connected to SDRAM should be set to16 bits.

Burst read/single write (burst length 1) and burst read/burst write (burst length 1) are supported as the SDRAM operating mode.

Commands for SDRAM can be specified by $\overline{RAS}$, $\overline{CAS}$, RD/$\overline{WR}$, and specific address signals. These commands are shown below.

- NOP
- Auto-refresh (REF)
- Self-refresh (SELF)
- All banks pre-charge (PALL)
- Specified bank pre-charge (PRE)
- Bank active (ACTV)
- Read (READ)
- Read with pre-charge (READA)
- Write (WRIT)
- Write with pre-charge (WRITA)
- Write mode register (MRS)

The byte to be accessed is specified by DQM1 and DQM0. Reading or writing is performed for a byte whose corresponding DQMn is low. For details on the relationship between DQMn and the byte to be accessed, refer to section 9.5.1, Endian/Access Size and Data Alignment.

Figures 9.11 shows as example of the connection of the SDRAM with this LSI. Address area of this LSI is allocated in byte units. Therefore, note that bit addresses of an external memory and those of this LSI differ when handling data in longword units or word units.

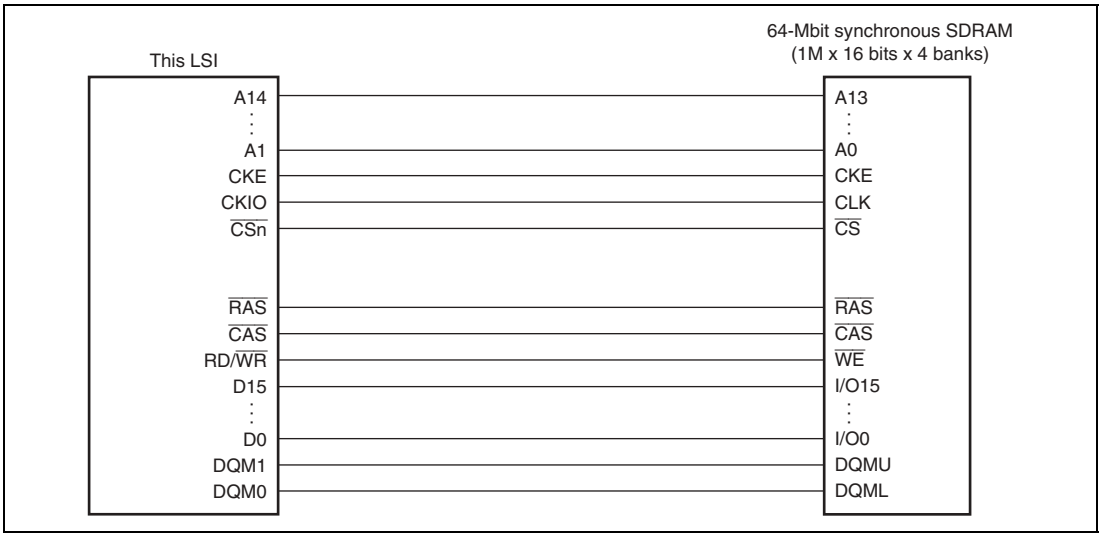


Figure 9.11 Example of 16-Bit Data-Width SDRAM Connection

Address Multiplexing: An address multiplexing is specified so that SDRAM can be connected without external multiplexing circuitry according to the setting of bits BSZ[1:0] in CS3BCR, AxROW[1:0] and AxCOL[1:0] in SDCR. Tables 9.7 to 9.9 show the relationship between the settings of bits BSZ[1:0], A3ROW[1:0], and A3COL[1:0] and the bits output at the address pins. Do not specify those bits in the manner other than this table, otherwise the operation of this LSI is not guaranteed. A23 to A18 are not multiplexed and the original values of address are always output at these pins.

When the data bus width is 16 bits (BSZ[1:0] =B'10), A0 of SDRAM specifies a word address. Therefore, connect this A0 pin of SDRAM to the A1 pin of the LSI; the A1 pin of SDRAM to the A2 pin of the LSI, and so on.

Table 9.7 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (1)-1

Setting				
CS3BCR.BSZ [1:0]	**SDCR.A3ROW [1:0]**	**SDCR.A3COL [1:0]**		
B'10 (Data bus width: 16 bits)	**B'00 (Row address bit count: 11 bits)**	**B'00 (Column address bit count: 8 bits)**		
Output Pin of This LSI	**Row Address Output Cycle**	**Column Address Output Cycle**	**Synchronous DRAM Pin**	**Function**
A17	A25	A17		Unused
A16	A24	A16		
A15	A23	A15		
A14	A22	A14		
A13	A21*2	A21*2	A12 (BA1)	Specifies bank
A12	A20*2	A20*2	A11 (BA0)	
A11	A19	L/H*1	A10/AP	Specifies address /precharge
A10	A18	A10	A9	Address
A9	A17	A9	A8	
A8	A16	A8	A7	
A7	A15	A7	A6	
A6	A14	A6	A5	
A5	A13	A5	A4	
A4	A12	A4	A3	
A3	A11	A3	A2	
A2	A10	A2	A1	
A1	A9	A1	A0	
A0	A8	A0		Unused
Example of connected memory				
16-Mbit product (512 kwords x 16 bits x 2 banks, column 8 bits product): 1				

Notes: 1. L/H is a bit used in the command specification; it is fixed at L or H according to the access mode.
2. Bank address specification

Table 9.7 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (1)-2

Setting				
CS3BCR.BSZ [1:0]	**SDCR.A3ROW [1:0]**	**SDCR.A3COL [1:0]**		
B'10 (Data bus width: 16 bits)	**B'01 (Row address bit count: 12 bits)**	**B'00 (Column address bit count: 8 bits)**		
Output Pin of This LSI	**Row Address Output Cycle**	**Column Address Output Cycle**	**Synchronous DRAM Pin**	**Function**
A17	A25	A17		Unused
A16	A24	A16		
A15	A23	A15		
A14	A22*[2]	A22*[2]	A13 (BA1)	Specifies bank
A13	A21*[2]	A21*[2]	A12 (BA0)	
A12	A20	A12	A11	Address
A11	A19	L/H*[1]	A10/AP	Specifies address /precharge
A10	A18	A10	A9	Address
A9	A17	A9	A8	
A8	A16	A8	A7	
A7	A15	A7	A6	
A6	A14	A6	A5	
A5	A13	A5	A4	
A4	A12	A4	A3	
A3	A11	A3	A2	
A2	A10	A2	A1	
A1	A9	A1	A0	
A0	A8	A0		Unused
Example of connected memory				
64-Mbit product (1 Mword x 16 bits x 4 banks, column 8 bits product): 1				

Notes: 1. L/H is a bit used in the command specification; it is fixed at L or H according to the access mode.
2. Bank address specification

Table 9.8 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (2)-1

Setting				
CS3BCR.BSZ [1:0]	**SDCR.A3ROW [1:0]**	**SDCR.A3COL [1:0]**		
B'10 (Data bus width: 16 bits)	**B'01 (Row address bit count: 12 bits)**	**B'01 (Column address bit count: 9 bits)**		
Output Pin of This LSI	**Row Address Output Cycle**	**Column Address Output Cycle**	**Synchronous DRAM Pin**	**Function**
A17	A26	A17		Unused
A16	A25	A16		
A15	A24	A15		
A14	A23*[2]	A23*[2]	A13 (BA1)	Specifies bank
A13	A22*[2]	A22*[2]	A12 (BA0)	
A12	A21	A12	A11	Address
A11	A20	L/H*[1]	A10/AP	Specifies address /precharge
A10	A19	A10	A9	Address
A9	A18	A9	A8	
A8	A17	A8	A7	
A7	A16	A7	A6	
A6	A15	A6	A5	
A5	A14	A5	A4	
A4	A13	A4	A3	
A3	A12	A3	A2	
A2	A11	A2	A1	
A1	A10	A1	A0	
A0	A9	A0		Unused
Example of connected memory				
128-Mbit product (2 Mwords x 16 bits x 4 banks, column 9 bits product): 1				

Notes: 1. L/H is a bit used in the command specification; it is fixed at L or H according to the access mode.
2. Bank address specification

Table 9.8 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (2)-2

Setting				
CS3BCR.BSZ [1:0]	**SDCR.A3ROW [1:0]**	**SDCR.A3COL [1:0]**		
B'10 (Data bus width: 16 bits)	**B'01 (Row address bit count: 12 bits)**	**B'10 (Column address bit count: 10 bits)**		
Output Pin of This LSI	**Row Address Output Cycle**	**Column Address Output Cycle**	**Synchronous DRAM Pin**	**Function**
A17	A27	A17		Unused
A16	A26	A16		
A15	A25	A15		
A14	A24*[2]	A24*[2]	A13 (BA1)	Specifies bank
A13	A23*[2]	A23*[2]	A12 (BA0)	
A12	A22	A12	A11	Address
A11	A21	L/H*[1]	A10/AP	Specifies address /precharge
A10	A20	A10	A9	Address
A9	A19	A9	A8	
A8	A18	A8	A7	
A7	A17	A7	A6	
A6	A16	A6	A5	
A5	A15	A5	A4	
A4	A14	A4	A3	
A3	A13	A3	A2	
A2	A12	A2	A1	
A1	A11	A1	A0	
A0	A10	A0		Unused
Example of connected memory				
256-Mbit product (4 Mwords x 16 bits x 4 banks, column 10 bits product): 1				

Notes: 1. L/H is a bit used in the command specification; it is fixed at L or H according to the access mode.
2. Bank address specification

Table 9.9 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (3)-1

Setting				
CS3BCR.BSZ [1:0]	**SDCR.A3ROW [1:0]**	**SDCR.A3COL [1:0]**		
B'10 (Data bus width: 16 bits)	**B'10 (Row address bit count: 13 bits)**	**B'01 (Column address bit count: 9 bits)**		
Output Pin of This LSI	**Row Address Output Cycle**	**Column Address Output Cycle**	**Synchronous DRAM Pin**	**Function**
A17	A26	A17		Unused
A16	A25	A16		
A15	A24*[2]	A24*[2]	A14 (BA1)	Specifies bank
A14	A23*[2]	A23*[2]	A13 (BA0)	
A13	A22	A13	A12	Address
A12	A21	A12	A11	
A11	A20	L/H*[1]	A10/AP	Specifies address /precharge
A10	A19	A10	A9	Address
A9	A18	A9	A8	
A8	A17	A8	A7	
A7	A16	A7	A6	
A6	A15	A6	A5	
A5	A14	A5	A4	
A4	A13	A4	A3	
A3	A12	A3	A2	
A2	A11	A2	A1	
A1	A10	A1	A0	
A0	A9	A0		Unused
Example of connected memory				
256-Mbit product (4 Mwords x 16 bits x 4 banks, column 9 bits product): 1				

Notes: 1. L/H is a bit used in the command specification; it is fixed at L or H according to the access mode.
2. Bank address specification

Table 9.9 Relationship between CS3BCR.BSZ[1:0], SDCR.A3ROW[1:0], SDCR.A3COL[1:0], and Address Multiplex Output (3)-2

Setting				
CS3BCR.BSZ [1:0]	**SDCR.A3ROW [1:0]**	**SDCR.A3COL [1:0]**		
B'10 (Data bus width: 16 bits)	**B'10 (Row address bit count: 13 bits)**	**B'10 (Column address bit count: 10 bits)**		
Output Pin of This LSI	**Row Address Output Cycle**	**Column Address Output Cycle**	**Synchronous DRAM Pin**	**Function**
A17	A27	A17		Unused
A16	A26	A16		
A15	A25*[2]	A25*[2]	A14 (BA1)	Specifies bank
A14	A24*[2]	A24*[2]	A13 (BA0)	
A13	A23	A13	A12	Address
A12	A22	A12	A11	
A11	A21	L/H*[1]	A10/AP	Specifies address /precharge
A10	A20	A10	A9	Address
A9	A19	A9	A8	
A8	A18	A8	A7	
A7	A17	A7	A6	
A6	A16	A6	A5	
A5	A15	A5	A4	
A4	A14	A4	A3	
A3	A13	A3	A2	
A2	A12	A2	A1	
A1	A11	A1	A0	
A0	A10	A0		Unused
Example of connected memory				
512-Mbit product (8 Mwords x 16 bits x 4 banks, column 10 bits product): 1				

Notes: 1. L/H is a bit used in the command specification; it is fixed at L or H according to the access mode.
2. Bank address specification

Burst Read: A burst read occurs in the following cases with this LSI.

- Access size in reading is larger than data bus width.
- 16-byte transfer in cache miss.
- 16-byte transfer in DMAC (access to non-cacheable region)

This LSI always accesses the SDRAM with burst length 1. For example, read access of burst length 1 is performed consecutively 8 times to read 16-byte continuous data from the SDRAM that is connected to a 16-bit data bus.

Table 9.10 shows the relationship between the access size and the number of bursts.

Table 9.10 Relationship between Access Size and Number of Bursts

Data Bus Width	Access Size	Number of Bursts
16 bits	8 bits	1
	16 bits	1
	32 bits	2
	16 bits	8

Figures 9.12 and 9.13 show a timing chart in burst read. In burst read, an ACTV command is output in the Tr cycle, the READA command is issued in the Tc1 to Tc7 cycles, the READA command is issued in the Tc8 cycle, and the read data is received at the rising edge of the external clock (CKIO) in the Td1 to Td8 cycles. The Tap cycle is used to wait for the completion of an auto-precharge induced by the READ command in the SDRAM. In the Tap cycle, a new command will not be issued to the same bank. However, access to another CS space or another bank in the same SDRAM space is enabled. The number of Tap cycles is specified by the TRP1 and TRP0 bits of the CS3WCR register.

In this LSI, wait cycles can be inserted by specifying each bit in the CS3WCR register to connect the SDRAM in variable frequencies. Figure 9.13 shows an example in which wait cycles are inserted. The number of cycles from the Tr cycle where the ACTV command is output to the Tc1 cycle where the READA command is output can be specified using the TRCD1 and TRCD0 bits of the CS3WCR register. If the TRCD1 and TRCD0 bits specify two cycles or more, a Trw cycle where the NOT command is issued is inserted between the Tr cycle and Tc1 cycle. The number of cycles from the Tc1 cycle where the READA command is output to the Td1 cycle where the read data is latched can be specified using the A3CL1 and A3CL0 bits of the CS3WCR register. The number of cycles from Tc1 to Td1 corresponds to the synchronous DRAM CAS latency. The CAS latency for the synchronous DRAM is normally defined as up to three cycles. However, the CAS latency in this LSI can be specified as 1 to 4 cycles. This CAS latency can be achieved by connecting a latch circuit between this LSI and the synchronous DRAM.

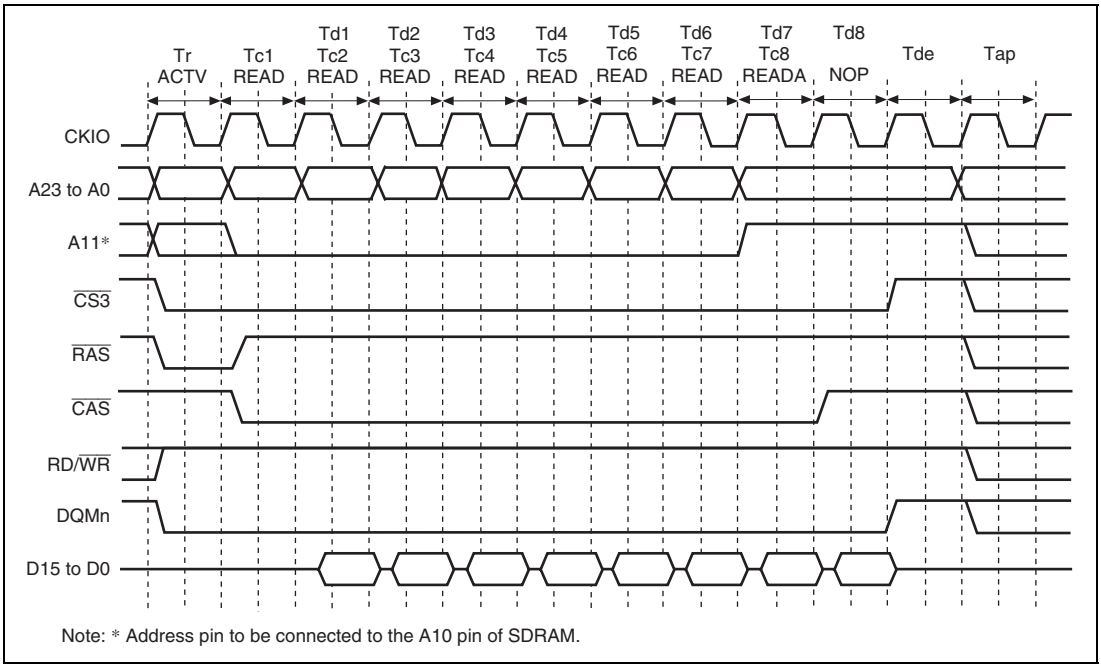


Figure 9.12 Burst Read Basic Timing (Auto Pre-charge)

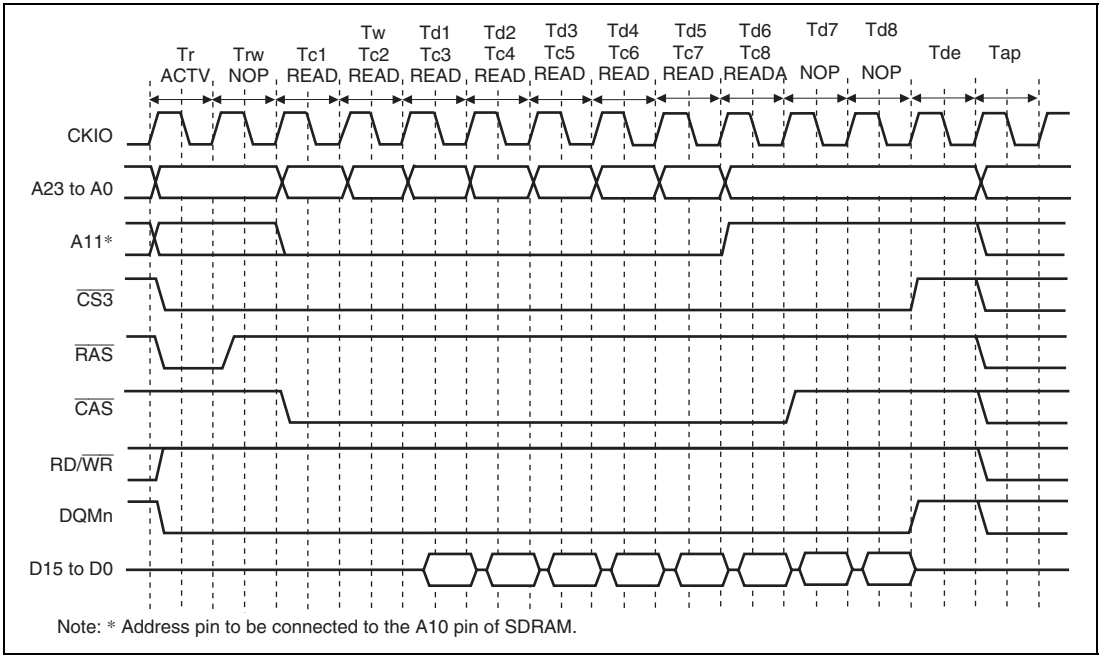
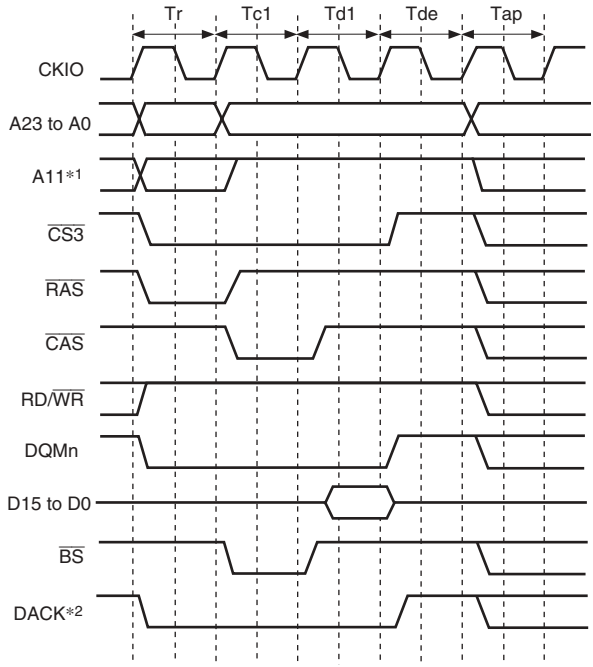


Figure 9.13 Burst Read Wait Specification Timing (Auto Pre-charge)

Single Read: A read access ends in one cycle when data exists in non-cacheable region and the data bus width is larger than or equal to access size. As the burst length is set to 1 in synchronous SDRAM burst read/single write mode, only the required data is output. Consequently, no unnecessary bus cycles are generated even when a cache-through area is accessed.

Figure 9.14 shows the single read basic timing.



Notes: 1. Address pin to be connected to the A10 pin of SDRAM.
2. The waveform for DACK is when active low is specified.

Figure 9.14 Single Read Wait Specification Timing (Auto Pre-charge)

Burst Write: A burst write occurs in the following cases in this LSI.

- Access size in writing is larger than data bus width.
- Copyback of the cache
- 16-byte transfer in DMAC (access to non-cacheable region)

This LSI always accesses SDRAM with burst length 1. For example, write access of burst length 1 is performed continuously 8 times to write 16-byte continuous data to the SDRAM that is connected to a 16-bit data bus.

The relationship between the access size and the number of bursts is shown in table 9.10.

Figure 9.15 shows a timing chart for burst writes. In burst write, an ACTV command is output in the Tr cycle, the WRIT command is issued in the Tc1 to Tc7 cycles, and the WRITA command is issued to execute an auto-precharge in the Tc8 cycle. In the write cycle, the write data is output simultaneously with the write command. After the write command with the auto-precharge is output, the Trw1 cycle that waits for the auto-precharge initiation is followed by the Tap cycle that waits for completion of the auto-precharge induced by the WRITA command in the SDRAM. In the Tap cycle, a new command will not be issued to the same bank. However, access to another CSn space or another bank in the same SDRAM space is enabled. The number of Trw1 cycles is specified by the TRWL1 and TRWL0 bits of the CS3WCR register. The number of Tap cycles is specified by the TRP1 and TRP0 bits of the CS3WCR register.

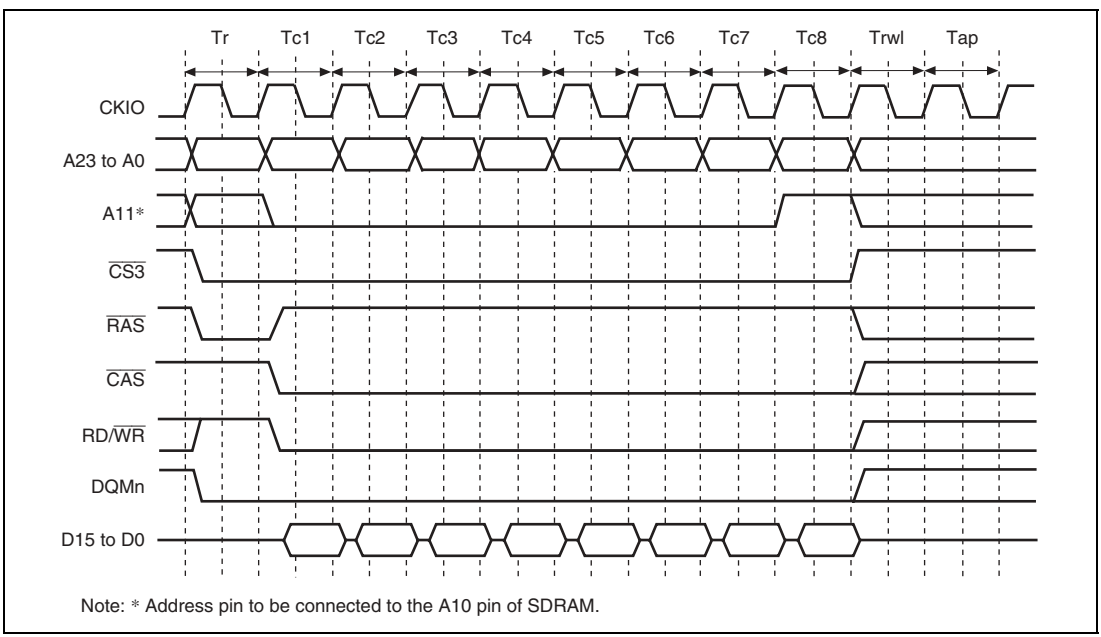
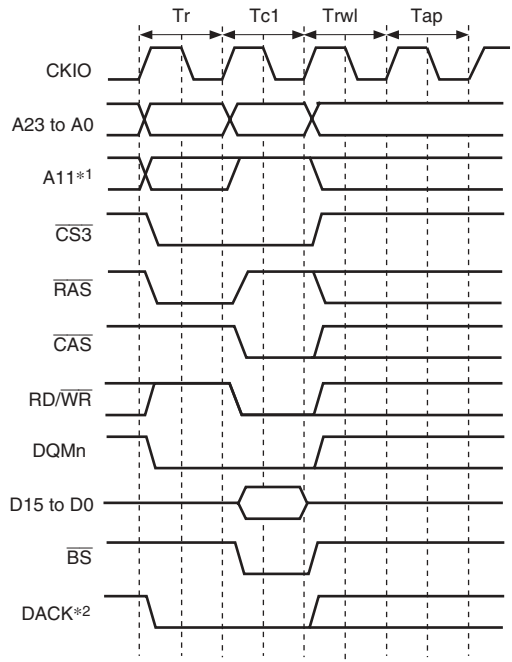


Figure 9.15 Basic Timing for Synchronous DRAM Burst Write (Auto Pre-charge)

Single Write: A write access ends in one cycle when data is written in non-cacheable region and the data bus width is larger than or equal to access size. This is called single write.

Figure 9.16 shows the single write basic timing.



Notes: 1. Address pin to be connected to the A10 pin of SDRAM.
2. The waveform for DACK is when active low is specified.

Figure 9.16 Single Write Basic Timing (Auto-Precharge)

Bank Active: The synchronous DRAM bank function is used to support high-speed accesses to the same row address. When the BACTV bit in SDCR is 1, accesses are performed using commands without auto-precharge (READ or WRIT). This function is called bank-active function. When the bank-active function is used, precharging is not performed when the access ends. When accessing the same row address in the same bank, it is possible to issue the READ or WRIT command immediately, without issuing an ACTV command. As synchronous DRAM is internally divided into several banks, it is possible to activate one row address in each bank. If the next access is to a different row address, a PRE command is first issued to precharge the relevant bank, then when precharging is completed, the access is performed by issuing an ACTV command followed by a READ or WRIT command. If this is followed by an access to a different row address, the access time will be longer because of the precharging performed after the access request is issued. The number of cycles between issuance of the PRE command and the ACTV command is determined by the TRP[1:0] bits in CS3WCR.

In a write, when an auto-precharge is performed, a command cannot be issued to the same bank for a period of Trwl + Tap cycles after issuance of the WRITA command. When bank active mode is used, READ or WRIT commands can be issued successively if the row address is the same. The number of cycles can thus be reduced by Trwl + Tap cycles for each write.

There is a limit on tRAS, the time for placing each bank in the active state. If there is no guarantee that there will not be a cache hit and another row address will be accessed within the period in which this value is maintained by program execution, it is necessary to set auto-refresh and set the refresh cycle to no more than the maximum value of tRAS.

A burst read cycle without auto-precharge is shown in figure 9.17, a burst read cycle for the same row address in figure 9.18, and a burst read cycle for different row addresses in figure 9.19. Similarly, a single write cycle without auto-precharge is shown in figure 9.260, a single write cycle for the same row address in figure 9.21, and a single write cycle for different row addresses in figure 9.22.

In figure 9.18, a Tnop cycle in which no operation is performed is inserted before the Tc cycle that issues the READ command. The Tnop cycle is inserted to acquire two cycles of latency for the DQMn signal that specifies the read byte in the data read from the SDRAM. If the CAS latency is specified as two cycles or more, the Tnop cycle is not inserted because the two cycles of latency can be acquired even if the DQMn signal is asserted after the Tc cycle.

When bank active mode is set, if only accesses to the respective banks in area with the bank-active function set are considered, as long as accesses to the same row address continue, the operation starts with the cycle in figure 9.17 or 9.20, followed by repetition of the cycle in figure 9.18 or 9.21. An access to a different area during this time has no effect. If there is an access to a different row address in the bank active state, after this is detected the bus cycle in figure 9.19 or 9.22 is executed instead of that in figure 9.18 or 9.21. In bank active mode, too, all banks become inactive after a refresh cycle or after the bus is released as the result of bus arbitration.

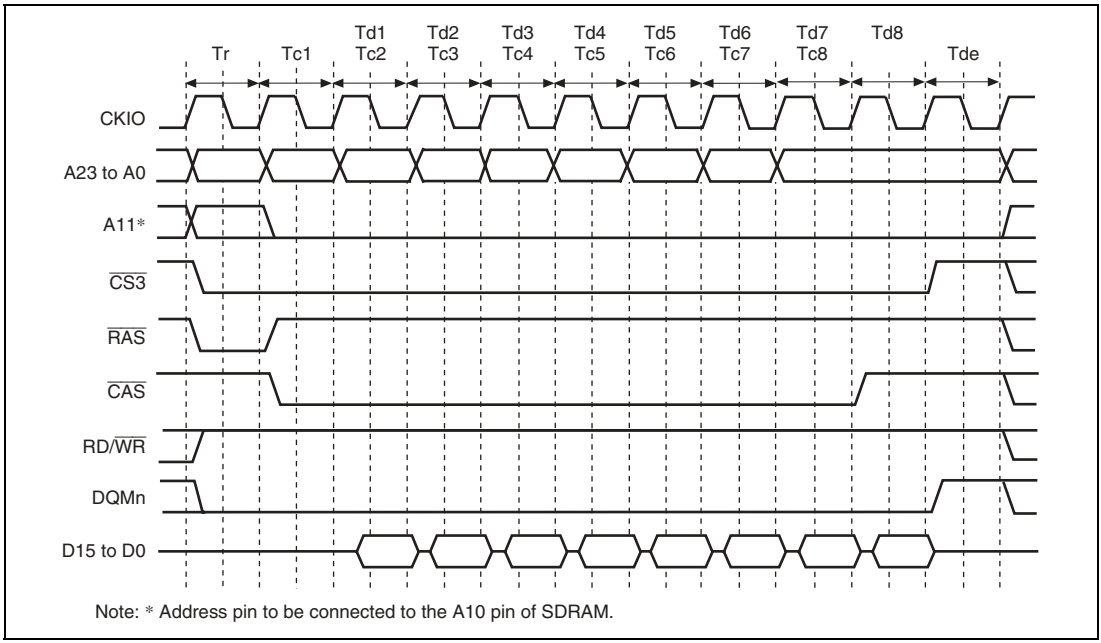


Figure 9.17 Burst Read Timing (No Auto Precharge)

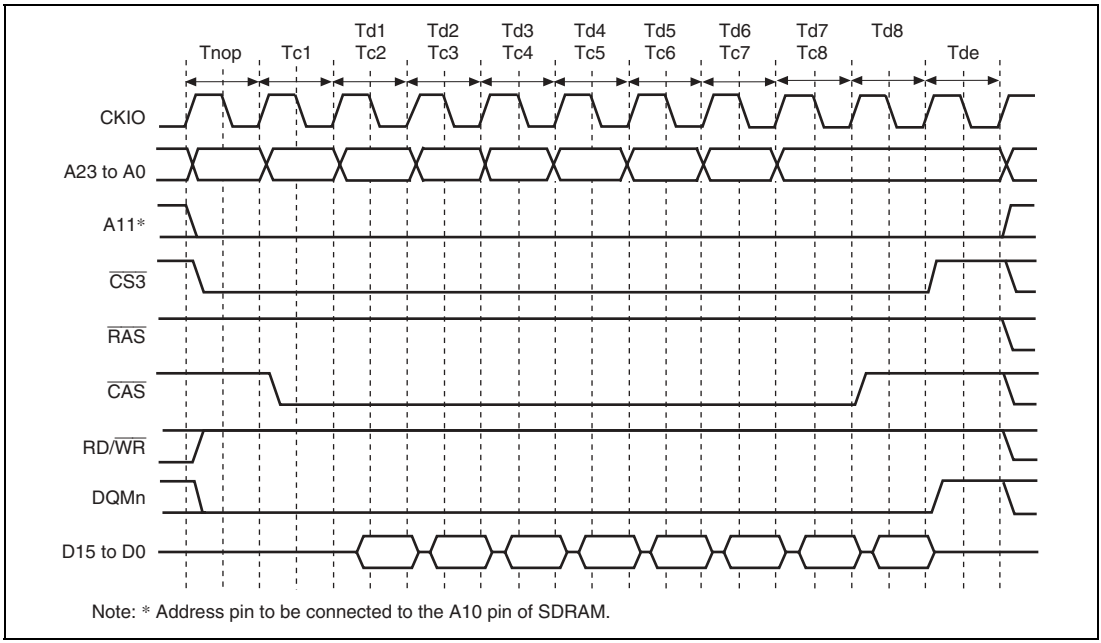


Figure 9.18 Burst Read Timing (Bank Active, Same Row Address)

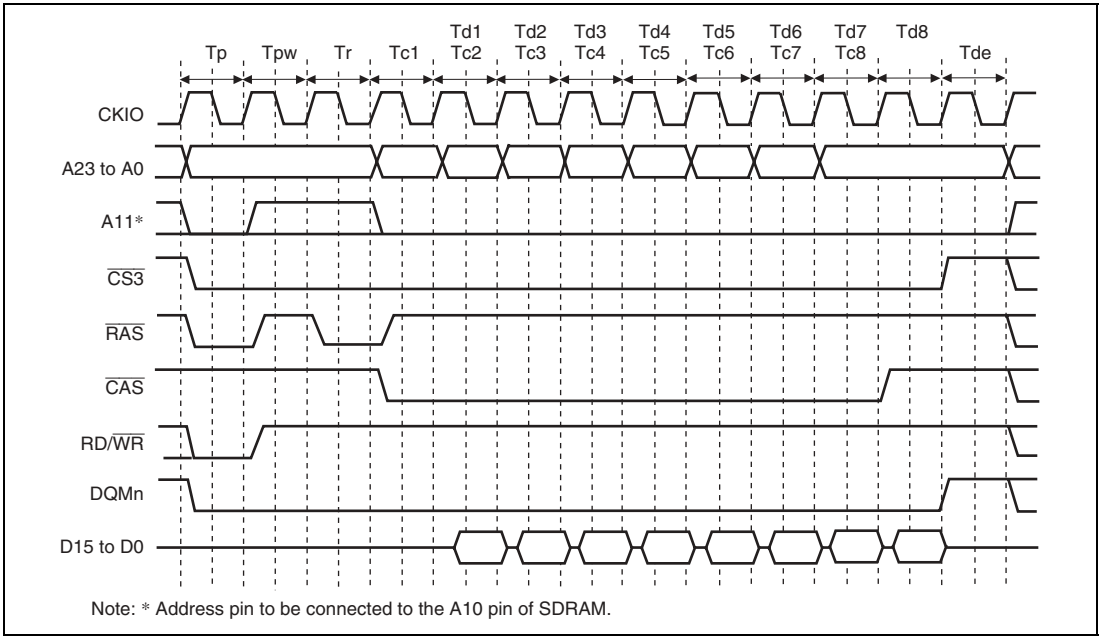
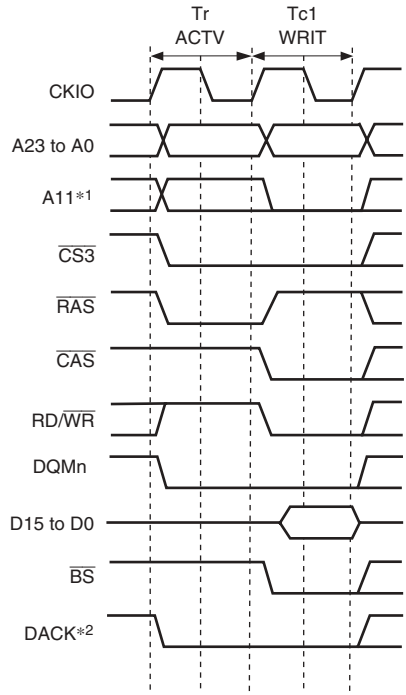
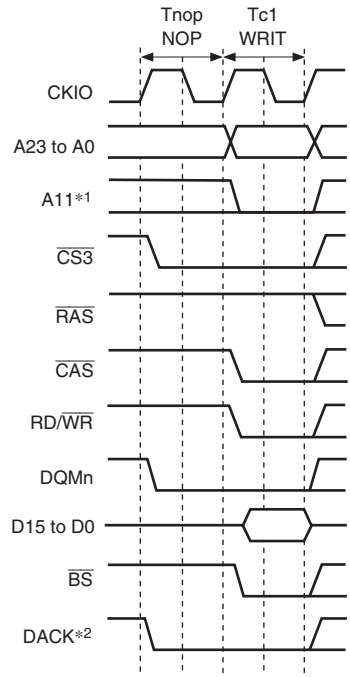


Figure 9.19 Burst Read Timing (Bank Active, Different Row Addresses)



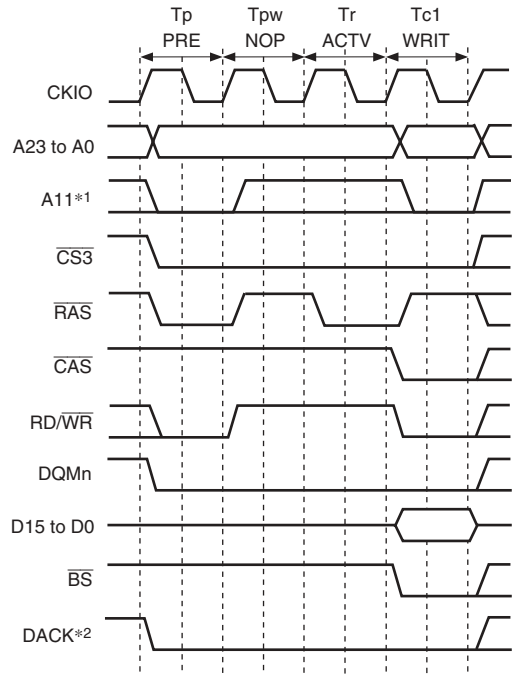
Notes: 1. Address pin to be connected to the A10 pin of SDRAM.
2. The waveform for DACK is when active low is specified.

Figure 9.20 Single Write Timing (No Auto Precharge)



Notes: 1. Address pin to be connected to the A10 pin of SDRAM.
2. The waveform for DACK is when active low is specified.

Figure 9.21 Single Write Timing (Bank Active, Same Row Address)



Notes: 1. Address pin to be connected to the A10 pin of SDRAM.
2. The waveform for DACK is when active low is specified.

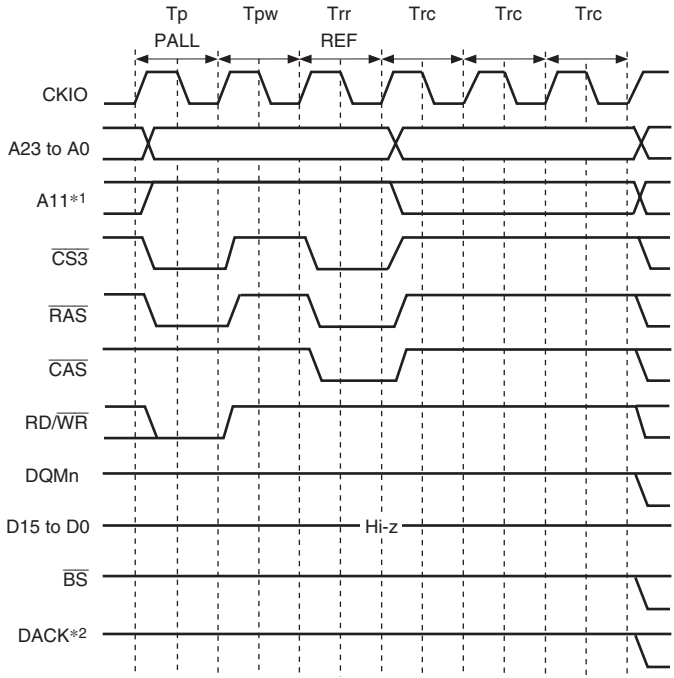
Figure 9.22 Single Write Timing (Bank Active, Different Row Addresses)

Refreshing: This LSI has a function for controlling synchronous DRAM refreshing. Auto-refreshing can be performed by clearing the RMODE bit to 0 and setting the RFSH bit to 1 in SDCR. A continuous refreshing can be performed by setting the RRC[2:0] bits in RTCSR. If SDRAM is not accessed for a long period, self-refresh mode, in which the power consumption for data retention is low, can be activated by setting both the RMODE bit and the RFSH bit to 1.

1. Auto-refreshing

   Refreshing is performed at intervals determined by the input clock selected by bits CKS[2:0] in RTCSR, and the value set by in RTCOR. The each register should be set so as to satisfy the refresh interval stipulation for the SDRAM used. First make the settings for RTCOR, RTCNT, and the RMODE and RFSH bits in SDCR, then make the CKS[2:0] and RRC[2:0] in RTCSR settings. When the clock is selected by bits CKS[2:0], RTCNT starts counting up from the value at that time. The RTCNT value is constantly compared with the RTCOR value, and if the two values are the same, a refresh request is generated and an auto-refresh is performed for the number of times specified by the RRC[2:0]. At the same time, RTCNT is cleared to zero and the count-up is restarted. Figure 9.23 shows the auto-refresh cycle timing.

After starting, the auto refreshing, PALL command is issued in the Tp cycle to make all the banks to pre-charged state from active state after waiting for the completion when some bank is being pre-charged. Then REF command is issued in the Trr cycle after inserting idle cycles of which number is specified by the TRP[1:0] bits in CS3WCR. A new command is not issued for the duration of the number of cycles specified by the TRC[1:0] bits in CS3WCR after the Trr cycle. The TRC[1:0] bits must be set so as to satisfy the SDRAM refreshing cycle time stipulation (tRC). A NOP cycle is inserted between the Tp cycle and Trr cycle when the setting value of the TRP[1:0] bits in CS3WCR is longer than or equal to 2 cycles.



Notes: 1. Address pin to be connected to the A10 pin of SDRAM.
2. The waveform for DACK is when active low is specified.

Figure 9.23 Auto-Refresh Timing

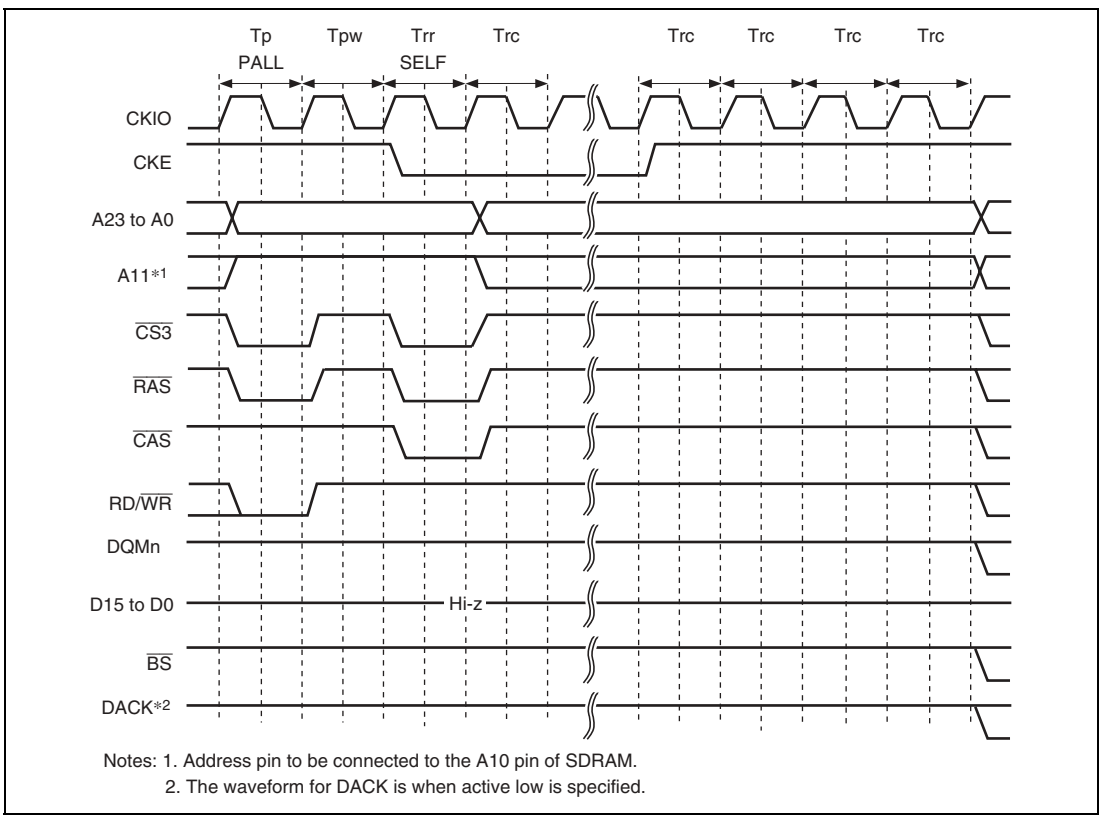
2. Self-refreshing

Self-refresh mode in which the refresh timing and refresh addresses are generated within the SDRAM. Self-refreshing is activated by setting both the RMODE bit and the RFSH bit in SDCR to 1. After starting the self-refreshing, PALL command is issued in Tp cycle after the completion of the pre-charging bank. A SELF command is then issued after inserting idle cycles of which number is specified by the TRP[1:0] bits in CS3WSR. The SDRAM cannot be accessed while in the self-refresh state. Self-refresh mode is cleared by clearing the RMODE bit to 0. After self-refresh mode has been cleared, command issuance is disabled for the number of cycles specified by the TRC[1:0] bits in CS3WCR.

Self-refresh timing is shown in figure 9.24. Settings must be made so that self-refresh clearing and data retention are performed correctly, and auto-refreshing is performed at the correct intervals. When self-refreshing is activated from the state in which auto-refreshing is set, or when exiting standby mode other than through a power-on reset, auto-refreshing is restarted if the RFSH bit is set to 1 and the RMODE bit is cleared to 0 when self-refresh mode is cleared. If the transition from clearing of self-refresh mode to the start of auto-refreshing takes time, this time should be taken into consideration when setting the initial value of RTCNT. Making the RTCNT value 1 less than the RTCOR value will enable refreshing to be started immediately.

After self-refreshing has been set, the self-refresh state continues even if the chip standby state is entered using the LSI standby function, and is maintained even after recovery from standby mode other than through a power-on reset. In case of a power-on reset, the bus state controller's registers are initialized, and therefore the self-refresh state is cleared.

After self-refreshing has been cleared, CKE should be high for a specified time for the SDRAM. When the SDRAM is in power-down mode (PDOWN in SDCR is set to 1), transition to self-refresh mode should be made after clearing power-down mode.



Notes: 1. Address pin to be connected to the A10 pin of SDRAM.
2. The waveform for DACK is when active low is specified.

Figure 9.24 Self-Refresh Timing

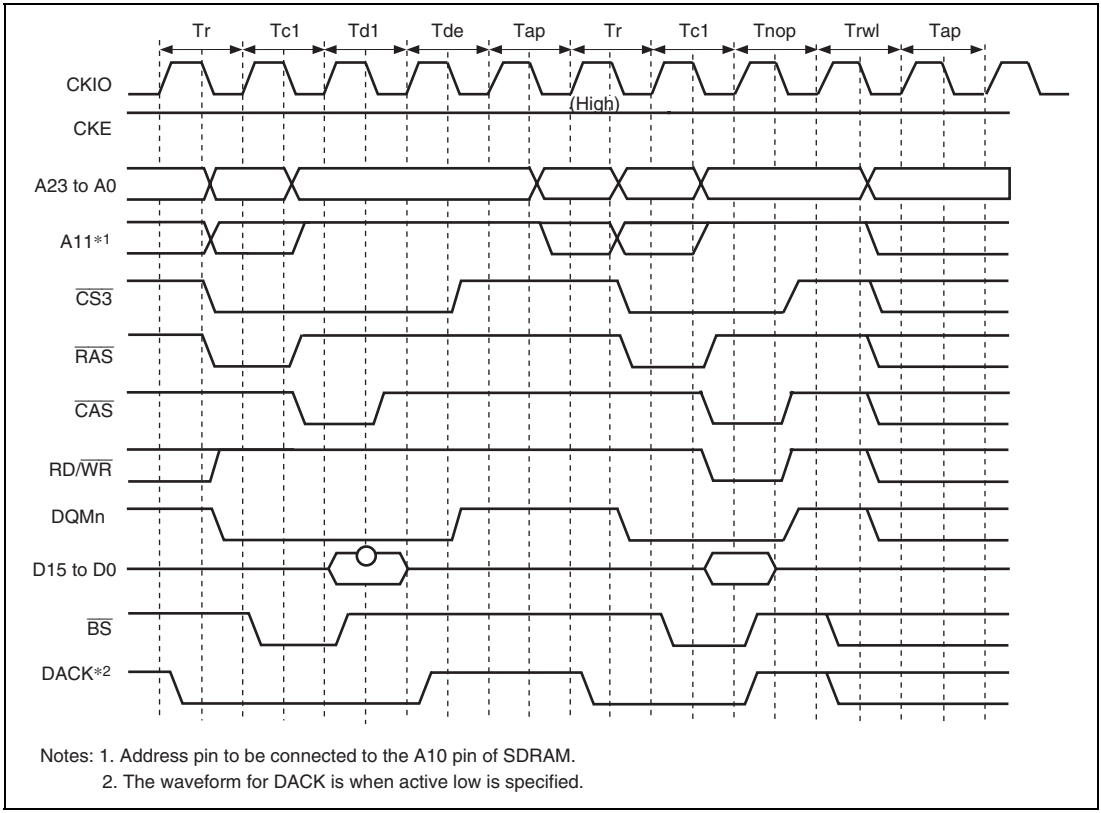
Relationship between Refresh Requests and Bus Cycles: If a refresh request occurs during bus cycle execution, the refresh cycle must wait for the bus cycle to be completed. If a refresh request occurs while the bus is released by the bus arbitration function, the refresh will not be executed until the bus mastership is acquired. This LSI supports requests by the $\overline{\text{REFOUT}}$ pin for the bus mastership while waiting for the refresh request. The $\overline{\text{REFOUT}}$ pin is asserted low until the bus mastership is acquired.

If a new refresh request occurs while waiting for the previous refresh request, the previous refresh request is deleted. To refresh correctly, a bus cycle longer than the refresh interval or the bus mastership occupation must be prevented from occurring. If a bus mastership is requested during self-refresh, the bus will not be released until the refresh is completed.

Low-Frequency Mode: When the SLOW bit in SDCR is set to 1, output of commands, addresses, and write data, and fetch of read data are performed at a timing suitable for operating SDRAM at a low frequency.

Figure 9.25 shows the access timing in low-frequency mode. In this mode, commands, addresses, and write data are output in synchronization with the falling edge of CKIO, which is half a cycle delayed than the normal timing. Read data is fetched at the falling edge of CKIO, which is half a cycle faster than the normal timing. This timing allows the hold time of commands, addresses, write data, and read data to be extended.

If SDRAM is operated at a high frequency with the SLOW bit set to 1, the setup time of commands, addresses, write data, and read data are not guaranteed. Take the operating frequency and timing design into consideration when making the SLOW bit setting.

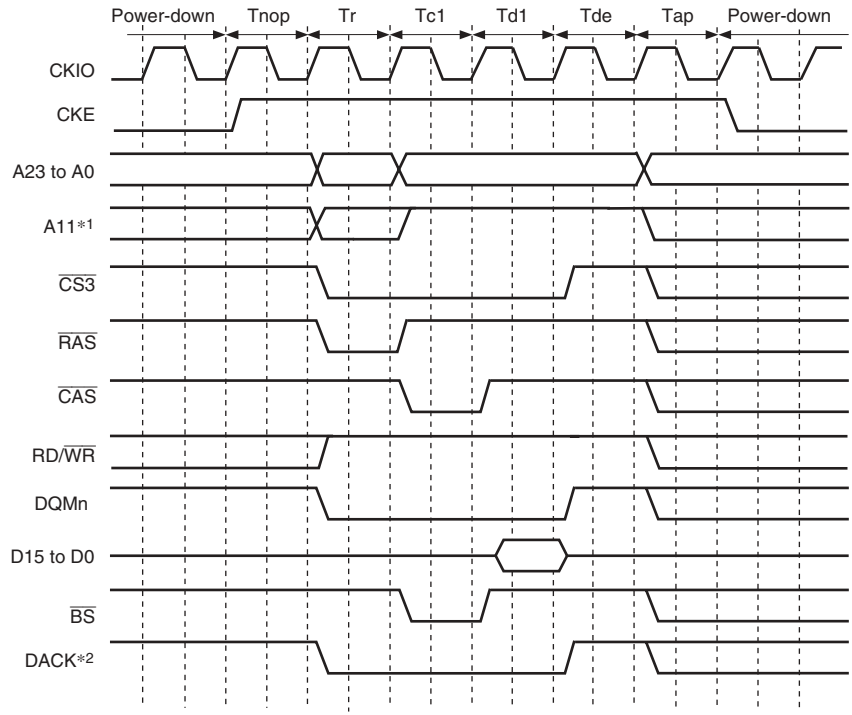


Notes: 1. Address pin to be connected to the A10 pin of SDRAM.
2. The waveform for DACK is when active low is specified.

Figure 9.25 Low-Frequency Mode Access Timing

Power-Down Mode: If the PDOWN bit of the SDCR register is set to 1, the SDRAM is placed in the power-down mode by bringing the CKE signal to the low level in the non-access cycle. This power-down mode can effectively lower the power consumption in the non-access cycle. However, please note that if an access occurs in power-down mode, a cycle of overhead occurs because a cycle is inserted to assert the CKE signal in order to cancel the power-down mode. After the SDRAM access ends, it is possible to shift to the power down mode again by accessing the memory other than SDRAM (CKE becomes Low level).

Figure 9.26 shows the access timing in power-down mode.



Notes: 1. Address pin to be connected to the A10 pin of SDRAM.
2. The waveform for DACK is when active low is specified.

Figure 9.26 Power-Down Mode Access Timing

Power-On Sequence: In order to use the SDRAM, mode setting must first be performed after powering on. To perform SDRAM initialization correctly, the bus state controller registers must first be set, followed by a write to the SDRAM mode register by accessing the SDMR3 register. In SDRAM mode register setting, the address signal value at that time is latched by a combination of the $\overline{CS3}$, $\overline{RAS}$, $\overline{CAS}$, and RD/$\overline{WR}$ signals. If the value to be set is X, the bus state controller provides for value X to be written to the SDRAM mode register by performing a write to address H'A4FD5000 + X for SDRAM. In this operation the data is ignored. To set burst read/single write (burst length 1) or burst read/burst write (burst length 1), CAS latency 2 to 3, wrap type = sequential, and burst length 1 supported by the LSI, arbitrary data is written in a word-size access to the addresses shown in table 9.11. In this time 0 is output at the external address pins of A12 or later.

Table 9.11 Access Address in SDRAM Mode Register Write

- Setting for Area 3 (SDMR3)

  Burst read/single write (burst length 1):

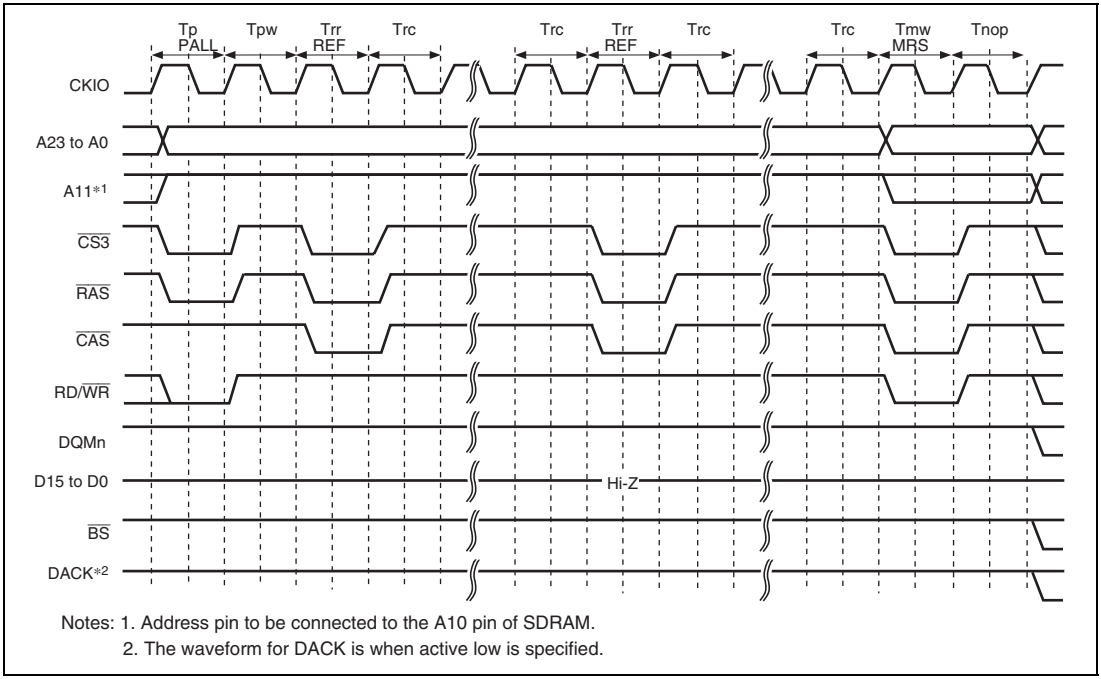
Data Bus Width	CAS Latency	Access Address	External Address Pin
16 bits	2	H'A4FD5440	H'0000440
	3	H'A4FD5460	H'0000460

Burst read/burst write (burst length 1):

Data Bus Width	CAS Latency	Access Address	External Address Pin
16 bits	2	H'A4FD5040	H'0000040
	3	H'A4FD5060	H'0000060

Mode register setting timing is shown in figure 9.27. A PALL command (all bank pre-charge command) is firstly issued. A REF command (auto refresh command) is then issued eight times. An MRS command (mode register write command) is finally issued. Idle cycles, of which number is specified by the TRP[1:0] bits in CS3WCR, are inserted between the PALL and the first REF. Idle cycles, of which number is specified by the TRC[1:0]bits in CS3WCR, are inserted between REF and REF, and between the 8th REF and MRS. Idle cycles, of which number is one or more, are inserted between the MRS and a command to be issued next.

It is necessary to keep idle time of certain cycles for SDRAM before issuing PALL command after power-on. Refer the manual of the SDRAM to be used for the idle time to be needed. When the pulse width of the reset signal is longer then the idle time, mode register setting can be started immediately after the reset, but care should be taken when the pulse width of the reset signal is shorter than the idle time.



Notes: 1. Address pin to be connected to the A10 pin of SDRAM.
2. The waveform for DACK is when active low is specified.

Figure 9.27 Synchronous DRAM Mode Write Timing (Based on JEDEC)

9.5.6 Burst ROM Interface

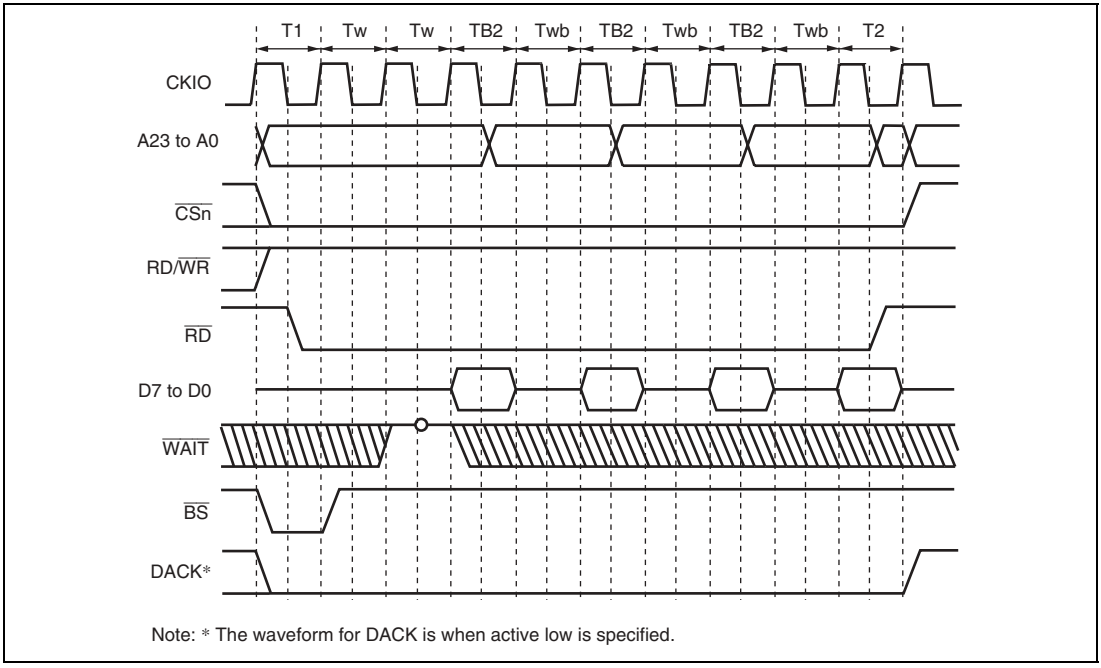
The burst ROM interface is used to access a memory with a high-speed read function using a method of address switching called the burst mode or page mode. In a burst ROM interface, basically the same access as the normal space interface is performed, but the 2nd and subsequent accesses are performed only by changing the address, without negating the $\overline{RD}$ signal at the end of the 1st cycle. In the 2nd and subsequent accesses, addresses are changed at the falling edge of the CKIO signal.

For the 1st access cycle, the number of wait cycles specified by the W[3:0] bits of the CSnWCR register is inserted. For the 2nd and subsequent access cycles, the number of wait cycles specified by the BW[1:0] bits of the CSnWCR register is inserted.

In the access to the burst ROM, the $\overline{BS}$ signal is asserted only to the first access cycle. An external wait input is valid only to the first access cycle. In the single access or write access that do not perform the burst operation in the page flash ROM interface, access timing is same as a normal space interface. Table 9.12 lists a relationship between bus width, access size, and the number of bursts. Figure 9.28 shows a timing chart.

Table 9.12 Relationship between Data Bus Width, Access Size, and Number of Bursts

Data Bus Width	Access Size	Number of Bursts	Number of Accesses
8 bits	8 bits	1	1
	16 bits	2	1
	32 bits	4	1
	16 bytes	16	1
16 bits	8 bits	1	1
	16 bits	1	1
	32 bits	2	1
	16 bytes	8	1



Note: * The waveform for DACK is when active low is specified.

Figure 9.28 Burst ROM Access (Data Bus Width 8 Bits, 4-byte Transfer (number of burst 4), Access Wait for the 1st time 2, Access Wait for 2nd Time and after 1)

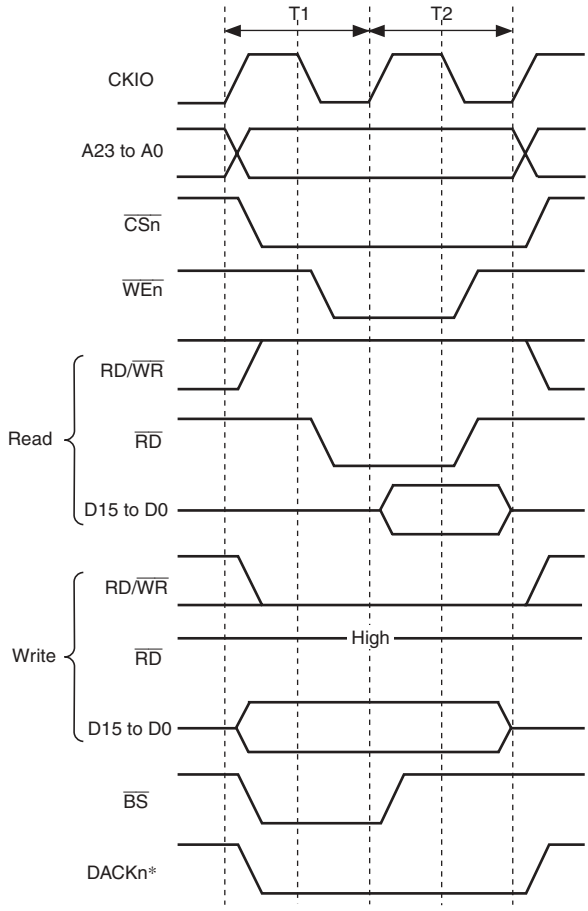
9.5.7 Byte-Selection SRAM Interface

Basic Access Timing:

The byte-selection SRAM interface is a memory interface which outputs a byte-selection pin ($\overline{WE}$) in a read/write bus cycle. This interface has 16-bit data pins and accesses SRAMs having upper and lower byte selection pins, such as UB and LB.

When the BAS bit of the CSnWCR register is cleared to 0 (initial value), the write access timing of the byte-selection SRAM interface is the same as that for the normal space interface. While in read access of a byte-selection SRAM interface, the byte-selection signal is output from the $\overline{WEn}$ pin, which is different from that for the normal space interface. The basic access timing is shown in figure 9.29. In write access, data is written to the memory according to the timing of the byte-selection pin ($\overline{WEn}$). For details, please refer to the Data Sheet for the corresponding memory.

If the BAS bit of the CSnWCR register is set to 1, the $\overline{WEn}$ pin and RD/$\overline{WR}$ pin timings change. Figure 9.30 shows the basic access timing. In write access, data is written to the memory according to the timing of the write enable pin (RD/$\overline{WR}$). The data hold timing from RD/$\overline{WR}$ negation to data write must be acquired by setting the HW[1:0] bits of the CSnWCR register. Figure 9.31 shows an example of Connection with Byte-Selection SRAM.



Note: * The waveform for DACK is when active low is specified.

Figure 9.29 Byte-Selection SRAM Basic Access Timing (BAS = 0)

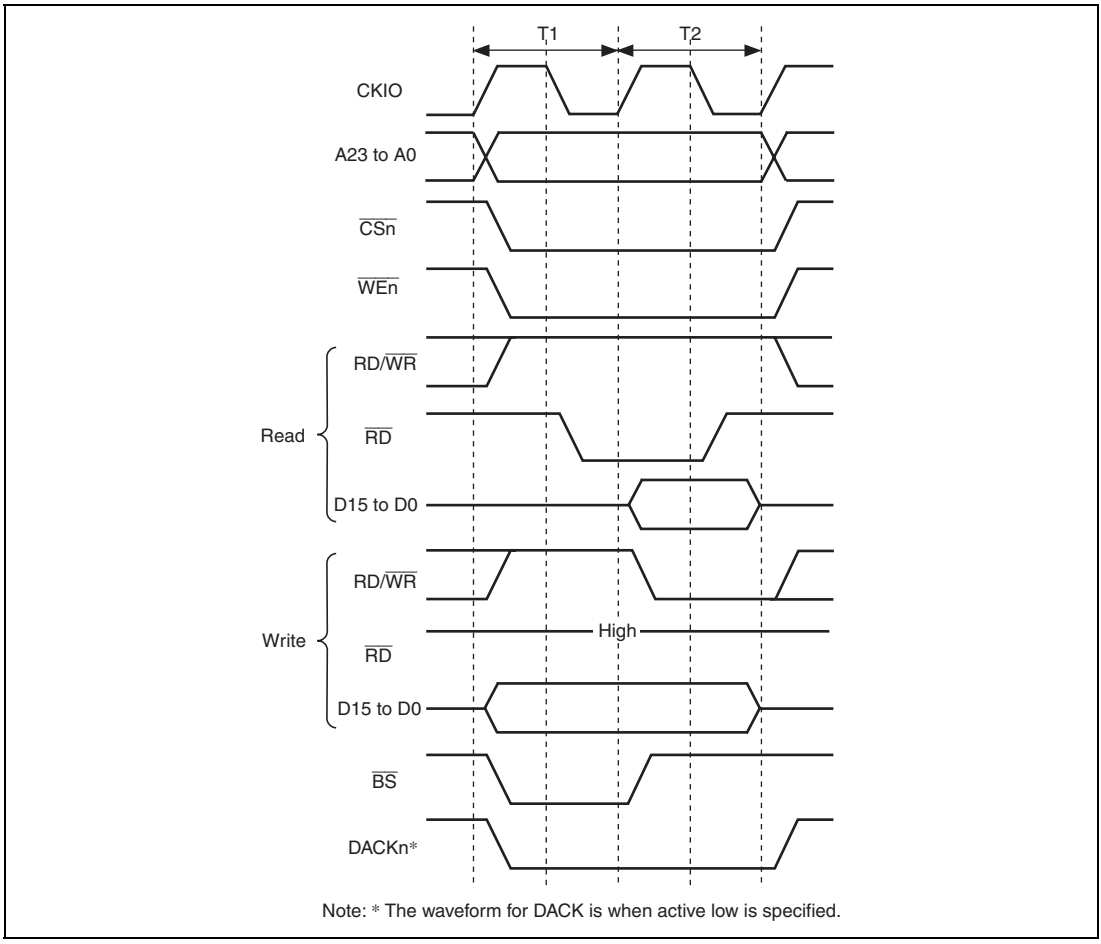


Figure 9.30 Byte-Selection RAM Basic Access Timing (BAS = 1)

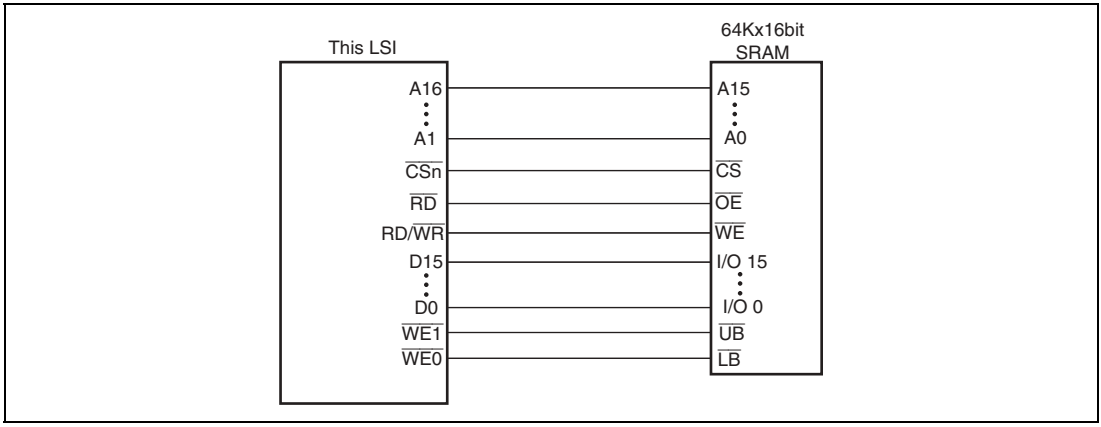


Figure 9.31 Example of Connection with 16-Bit Data-Width Byte-Selection SRAM

Byte-Selection SRAM Interface Restrictions:

The BAS bit should not be set to 1 for the byte-selection SRAM while the SDRAM in auto-refresh is used. If the BAS bit is set to 1, a refresh cannot be issued.

Problems:

When the byte-selection SRAM is accessed (BAS = 1) before a refresh is carried out, the ACTV command is issued instead of the RD/$\overline{WR}$ low output during the PALL command cycle. Therefore, the operation of SDRAM cannot be guaranteed in subsequent REF command.

Avoidance measures:

1. A pseudo SRAM which is mixed with SDRAM should support UB and LB control write.
2. When the $\overline{WE}$ pin of MCP is necessary to be connected to the RD/$\overline{WR}$ pin of this LSI to use MCP incorporating a flash memory, make access according to the following procedure.

   Make SDRAM enter in self-refresh

   Set the functions of CSnBCR and CSnWCR in the flash memory area for using the byte-selection SRAM (BAS = 1): accessing byte-selection SRAM and WE write control.

   Start write access to a flash memory

When a write access to a flash memory ends, set the UB and LB control (BAS = 0) and then clear the self-refresh function.

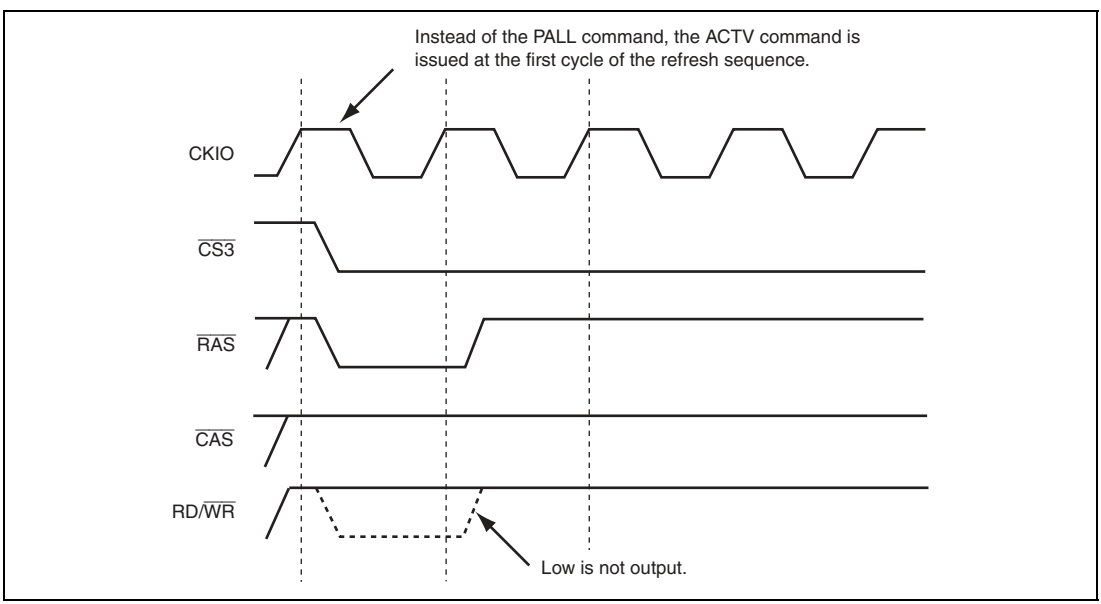


Figure 9.32 Waveform in the Event of a Problem

9.5.8 Wait between Access Cycles

As the operating frequency of LSIs becomes higher, the off-operation of the data buffer often collides with the next data access when the read operation from devices with slow access speed is completed. As a result of these collisions, the reliability of the device is low and malfunctions may occur. A function that avoids data collisions by inserting wait cycles between continuous access cycles has been newly added.

The number of wait cycles between access cycles can be set by bits IWW[1:0], IWRWD[1:0], IWRWS[1:0], IWRRD[1:0], and IWRRS[1:0] in the CSnBCR register , and bits DMAIW[1:0] and DMAIWA in CMNCR. The conditions for setting the wait cycles between access cycles (idle cycles) are shown below.

1. Continuous accesses are write-read or write-write
2. Continuous accesses are read-write for different spaces
3. Continuous accesses are read-write for the same space
4. Continuous accesses are read-read for different spaces
5. Continuous accesses are read-read for the same space
6. Data output from an external device caused by DMA single transfer is followed by data output from another device that includes this LSI (DMAIWA = 0)
7. Data output from an external device caused by DMA single transfer is followed by any type of access (DMAIWA = 1)

9.5.9 Bus Arbitration

In bus arbitration, the LSI operates in two ways;

1. The LSI has the bus mastership in the normal state, and releases the bus mastership after receiving a bus request from another device.
2. The LSI does not have the bus mastership in the normal state; it requests the bus mastership each time that external access is necessary.

In the case of the LSI with bus mastership in the normal state, this is called master mode. This LSI supports master mode.

To prevent device malfunction while the bus mastership is transferred between master and slave, the LSI negates all of the bus control signals before bus release. When the bus mastership is received, all of the bus control signals are first negated and then driven appropriately. In this case, output buffer contention can be prevented because the master and slave drive the same signals with the same values. In addition, to prevent noise while the bus control signal is in the high impedance state, pull-up resistors must be connected to these control signals.

Bus mastership is transferred at the boundary of bus cycles. Namely, bus mastership is released immediately after receiving a bus request when a bus cycle is not being performed. The release of

bus mastership is delayed until the bus cycle is complete when a bus cycle is in progress. Even when from outside the LSI it looks like a bus cycle is not being performed, a bus cycle may be performing internally, started by inserting wait cycles between access cycles. Therefore, it cannot be immediately determined whether or not bus mastership has been released by looking at the $\overline{\text{CSn}}$ signal or other bus control signals. The states that do not allow bus mastership release are shown below.

1. 16-byte transfer because of a cache miss
2. During copyback operation for the cache
3. Between the read and write cycles of a TAS instruction
4. Multiple bus cycles generated when the data bus width is smaller than the access size (for example, between bus cycles when longword access is made to a memory with a data bus width of 8 bits)
5. 16-byte transfer by the DMAC

The bus release by the $\overline{\text{BREQ}}$ and $\overline{\text{BACK}}$ signal handshaking requires some overhead. If the slave has many tasks, multiple bus cycles should be executed in a bus mastership acquisition. Reducing the cycles required for master to slave bus mastership transitions streamlines the system design.

Master Mode: In master mode, the LSI has the bus mastership until a bus request is received from another device. Upon acknowledging the assertion (low level) of the external bus request signal $\overline{\text{BREQ}}$, the LSI releases the bus at the completion of the current bus cycle and asserts the $\overline{\text{BACK}}$ signal. After the LSI acknowledges the negation (high level) of the $\overline{\text{BREQ}}$ signal that indicates the slave has released the bus, it negates the $\overline{\text{BACK}}$ signal and resumes the bus usage.

The SDRAM issues all bank pre-charge commands (PALLs) when active banks exist and releases the bus after completion of a PALL command.

The bus sequence is as follows.

1. The address bus and data bus are placed in a high-impedance state synchronized with the rising edge of CKIO.
2. The bus mastership enable signal is asserted 0.5 cycles after the above timing, synchronized with the falling edge of CKIO.
3. The bus control signals ($\overline{\text{BS}}$, $\overline{\text{CSn}}$, $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, $\overline{\text{WEn}}$/$\overline{\text{DQMn}}$, $\overline{\text{RD}}$, and RD/$\overline{\text{WR}}$) are placed in the high-impedance state at subsequent rising edges of CKIO. These bus control signals go high one cycle before being placed in the high-impedance state.
4. Bus request signals are sampled at the falling edge of CKIO.

The sequence for reclaiming the bus mastership from a slave is described below.

1. 1.5 cycles after the negation of $\overline{\text{BREQ}}$ is detected at the falling edge of CKIO, the bus control signals are driven high.
2. The bus enable signal is negated at the next falling edge of the clock. The address bus and data bus are started to be driven at the next rising edge of CKIO.
3. The fastest timing at which actual bus cycles can be resumed after bus control signal assertion is at the rising edge of the CKIO where address and data signals are driven.

Figure 9.33 shows the bus arbitration timing in master mode.

In an original slave device designed by the user, multiple bus accesses are generated continuously to reduce the overhead caused by bus arbitration. In this case, to execute SDRAM refresh correctly, the slave device must be designed to release the bus mastership within the refresh interval time. To achieve this, the LSI instructs the $\overline{\text{REFOUT}}$ pin to request the bus mastership while the SDRAM waits for the refresh. The LSI asserts the $\overline{\text{REFOUT}}$ pin until the bus mastership is received. If the slave releases the bus, the LSI acquires the bus mastership to execute the SDRAM refresh.

The bus release by the $\overline{\text{BREQ}}$ and $\overline{\text{BACK}}$ signal handshaking requires some overhead. If the slave has many tasks, multiple bus cycles should be executed in a bus mastership acquisition. Reducing the cycles required for master to slave bus mastership transitions streamlines the system design.

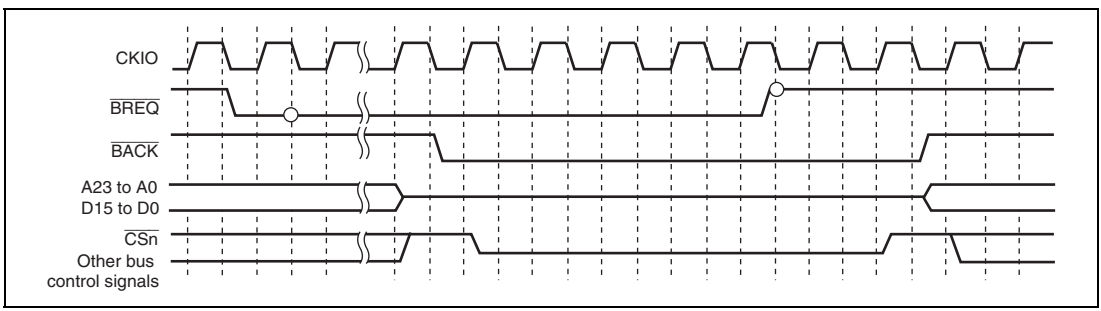


Figure 9.33 Bus Arbitration Timing (Master Mode)

Master and Slave Cooperation: Figure 9.34 shows an example of the master and slave devices. To control system resources correctly between master and slave devices, the roles of the master and slave devices must be defined appropriately. The SDRAM must be initialized before using the devices. To use the standby operation to reduce power consumption, the roles of the master and slave devices must also be defined appropriately.

This LSI is designed to execute all controls such as initialization, refresh, and standby in a device in master mode. If a master device and a slave device are connected directly in a two-processor configuration, the master device performs all processing other than memory access. In this case, the slave device must not access devices such as an SDRAM that require initialization prior to device initialization. The following methods can be used to prevent this.

1. An external circuit can be connected that releases the slave device reset by the master device.
2. The master can set a flag to the devices such as an SRAM that does not require initialization after initialization and the slave initiates access after checking to see that the flag has been set.
3. An external circuit can be connected to generate an interrupt from the master device to a slave device after initialization. The slave is released from the wait state by the interrupt.

If self-refresh mode is specified for the SDRAM, the master device does not release the bus. If a transition is underway to standby mode, or if the master clock stops because of a frequency change, the master device cannot release the bus. To prevent the slave from issuing bus requests in a case such as this, the slave must be put into the sleep state.

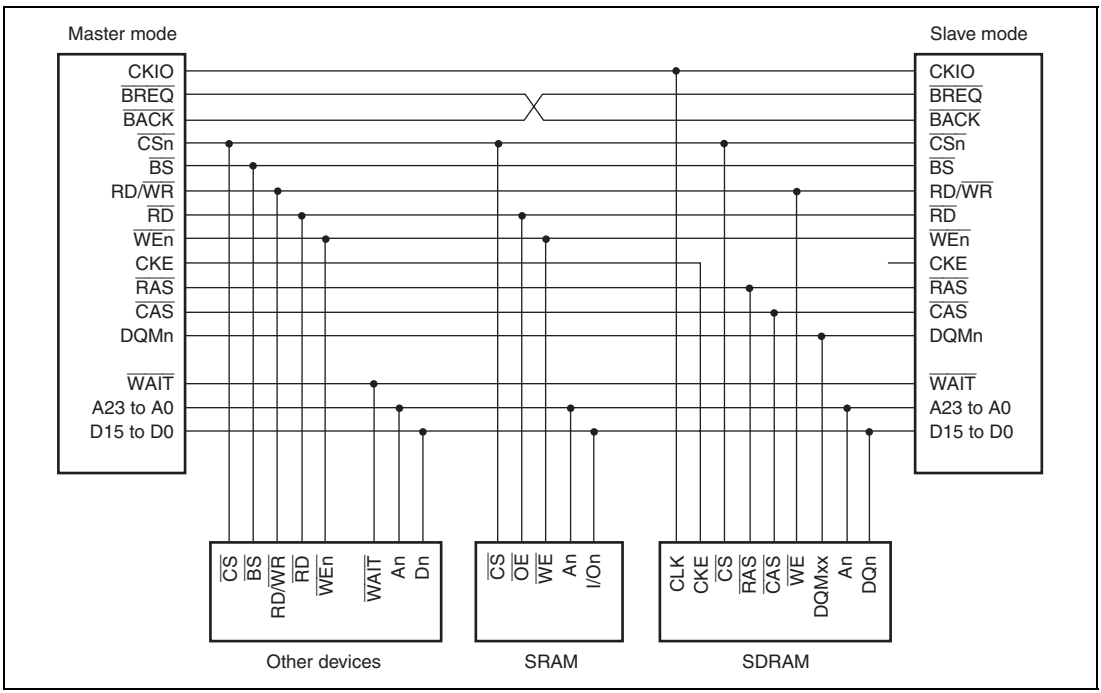


Figure 9.34 Master and Slave Connection Example

9.5.10 Usage Notes

Reset: The bus state controller (BSC) can be initialized completely only at power-on reset. At power-on reset, all signals are negated and output buffers are turned off regardless of the bus cycle state. All control registers are initialized.

In standby, sleep, and manual reset, control registers of the bus state controller are not initialized. At manual reset, the current bus cycle being executed is completed and then the access wait state is entered. If a 16-byte transfer is performed by a cache or if another LSI on-chip bus master module is executed when a manual reset occurs, the current access is cancelled in longword units because the access request is cancelled by the bus master at manual reset. If a manual reset is requested during cache fill operations, the contents of the cache cannot be guaranteed. Since the RTCNT continues counting up during manual reset signal assertion, a refresh request occurs to initiate the refresh cycle. In addition, a bus arbitration request by the $\overline{\text{BREQ}}$ signal can be accepted during manual reset signal assertion.

Some flash memories may specify a minimum time from reset release to the first access. To ensure this minimum time, the bus state controller supports a 7-bit counter (RWTCNT). At power-on reset, the RWTCNT is cleared to 0. After power-on reset, RWTCNT is counted up synchronously together with CKIO and an external access will not be generated until RWTCNT is counted up to H′007F. At manual reset, RWTCNT is not cleared.

Access from the Site of the LSI Internal Bus Master: There are three types of LSI internal buses: a cache bus (L bus), internal bus (I bus), and peripheral bus. The CPU and cache memory are connected to the cache bus. Internal bus masters other than the CPU and bus state controller are connected to the internal bus. Low-speed peripheral modules are connected to the peripheral bus. Internal memories other than the cache memory and debugging modules such as a UBC and AUD are connected bidirectionally to the cache bus and internal bus. Access from the cache bus to the internal bus is enabled but access from the internal bus to the cache bus is disabled. This gives rise to the following problems.

On-chip bus masters such as DMAC other than the CPU can access internal memory other than the cache memory but cannot access the cache memory. If an on-chip bus master other than the CPU writes data to an external memory other than the cache, the contents of the external memory may differ from that of the cache memory. To prevent this problem, if the external memory whose contents is cached is written by an on-chip bus master other than the CPU, the corresponding cache memory should be purged by software.

If the CPU initiates read access for the cache, the cache is searched. If the cache stores data, the CPU latches the data and completes the read access. If the cache does not store data, the CPU performs four contiguous longword read cycles to perform cache fill operations via the internal bus. If a cache miss occurs in byte or word operand access or at a branch to an odd word boundary (4n + 2), the CPU performs four contiguous longword accesses to perform a cache fill operation on the external interface. For a cache-through area, the CPU performs access according to the

actual access addresses. For an instruction fetch to an even word boundary (4n), the CPU performs longword access. For an instruction fetch to an odd word boundary (4n + 2), the CPU performs word access.

For a read cycle of a cache-through area or an on-chip peripheral module, the read cycle is first accepted and then read cycle is initiated. The read data is sent to the CPU via the cache bus.

In a write cycle for the cache area, the write cycle operation differs according to the cache write methods.

In write-back mode, the cache is first searched. If data is detected at the address corresponding to the cache, the data is then re-written to the cache. In the actual memory, data will not be re-written until data in the corresponding address is re-written. If data is not detected at the address corresponding to the cache, the cache is modified. In this case, data to be modified is first saved to the internal buffer, 16-byte data including the data corresponding to the address is then read, and data in the corresponding access of the cache is finally modified. Following these operations, a write-back cycle for the saved 16-byte data is executed.

In write-through mode, the cache is first searched. If data is detected at the address corresponding to the cache, the data is re-written to the cache simultaneously with the actual write via the internal bus. If data is not detected at the address corresponding to the cache, the cache is not modified but an actual write is performed via the internal bus.

Since the bus state controller (BSC) incorporates a one-stage write buffer, the BSC can execute an access via the internal bus before the previous external bus cycle is completed in a write cycle. If the on-chip module is read or written after the external low-speed memory is written, the on-chip module can be accessed before the completion of the external low-speed memory write cycle. In read cycles, the CPU is placed in the wait state until read operation has been completed. To continue the process after the data write to the device has been completed, perform a dummy read to the same address to check for completion of the write before the next process to be executed.

The write buffer of the BSC functions in the same way for an access by a bus master other than the CPU such as the DMAC. Accordingly, to perform dual address DMA transfers, the next read cycle is initiated before the previous write cycle is completed. Note, however, that if both the DMA source and destination addresses exist in external memory space, the next write cycle will not be initiated until the previous write cycle is completed.

On-Chip Peripheral Module Access: To access an on-chip module register, two or more peripheral module clock (Pϕ) cycles are required. Care must be taken in system design.

Section 10 Direct Memory Access Controller (DMAC)

This LSI includes the direct memory access controller (DMAC).

The DMAC can be used in place of the CPU to perform high-speed transfers between external devices that have DACK (transfer request acknowledge signal), external memory, on-chip memory, memory-mapped external devices, and on-chip peripheral modules.

10.1 Features

- Four channels (One channel can receive an external request)
  Channel 0 can receive an external request.
- 4-GB physical address space
- Data transfer unit is selectable: Byte, Word (two bytes), Longword (four bytes), and 16 Bytes (longword × 4)
- Maximum transfer count: 16777216 transfers (24 bits)
- Address mode: Dual address mode and single address mode are supported.
- Transfer requests
  — External request
  — On-chip peripheral module request
  — Auto request
  The following modules can issue an on-chip peripheral module request.
  — SCIF0, SCIF1, SIOF, and USBF
- Selectable bus modes
  — Cycle steal mode (normal mode and intermittent mode)
  — Burst mode
- Selectable channel priority levels: The channel priority levels are selectable between fixed mode and round-robin mode.
- Interrupt request: An interrupt request can be generated to the CPU at the end of the specified counts of data transfer.
- External request detection: There are following four types of DREQ input detection.
  — Low level detection
  — High level detection
  — Rising edge level detection
  — Falling edge level detection
- Transfer request acknowledge and transfer end signals: Active levels for DACK and TEND can be set independently.

Figure 10.1 shows the block diagram of the DMAC.

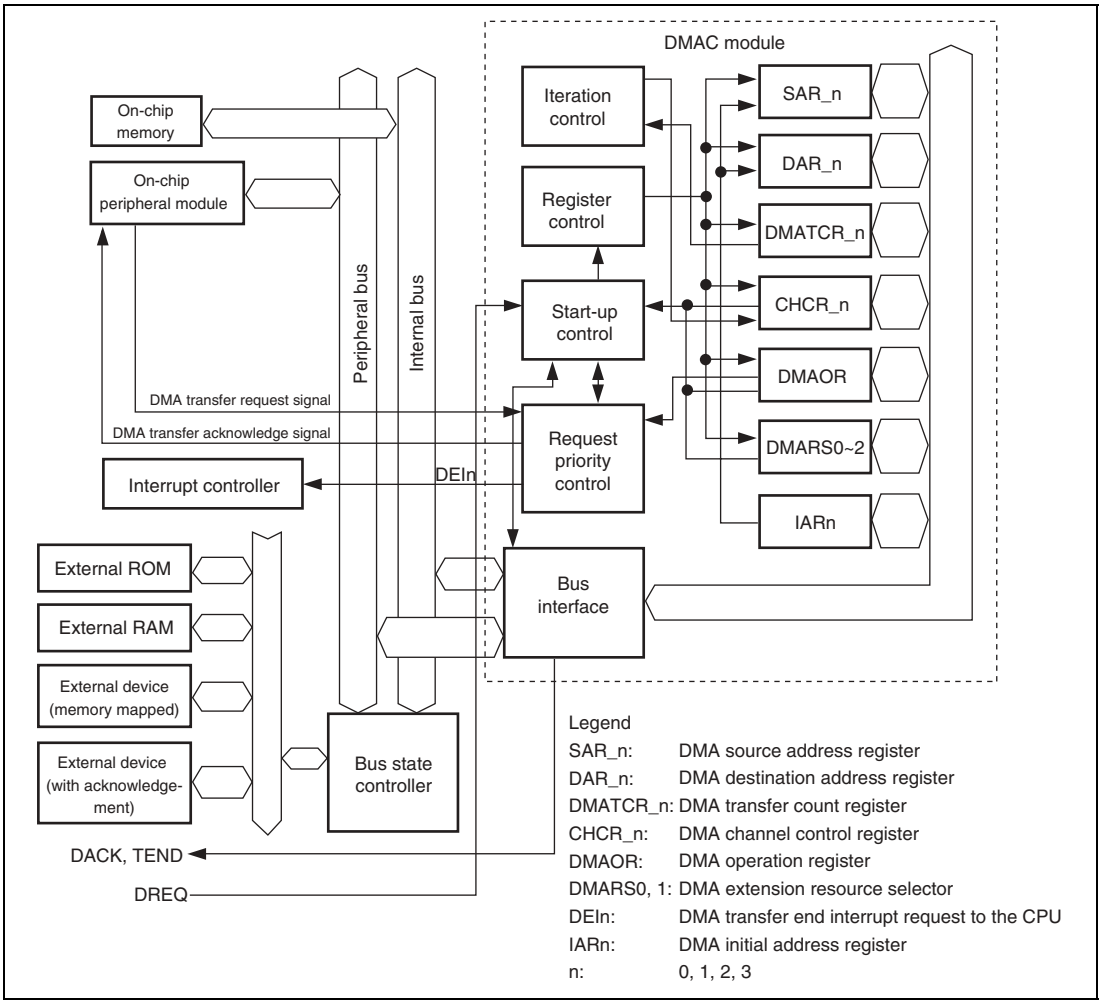


Figure 10.1 Block Diagram of the DMAC

10.2 Input/Output Pins

Table 10.1 lists the pin configuration of DMAC.

Table 10.1 Pin Configuration

Channel	Pin Name	I/O	Function
0	DREQ	Input	DMA transfer request
			DMA transfer request input from external device to channel 0
	DACK	Output	DMA transfer request acknowledge
			Strobe output to an external I/O at DMA transfer request from external device to channel 0
	TEND	Output	DMA transfer end
			DMA transfer end output for channel 0

10.3 Register Descriptions

Register configuration of DMAC is described below. See section 27, List of Registers, for the addresses of these registers and the state of them in each processing status. In the following description of registers for each channel, for example, SAR for channel 0 is described as SAR_0.

Channel 0:

- DMA source address register_0 (SAR_0)
- DMA destination address register_0 (DAR_0)
- DMA transfer count register_0 (DMATCR_0)
- DMA channel control register_0 (CHCR_0)
- DMA initial address register_0 (IAR_0)

Channel 1:

- DMA source address register_1 (SAR_1)
- DMA destination address register_1 (DAR_1)
- DMA transfer count register_1 (DMATCR_1)
- DMA channel control register _1 (CHCR_1)
- DMA initial address register_1 (IAR_1)

Channel 2:

- DMA source address register_2 (SAR_2)
- DMA destination address register_2 (DAR_2)
- DMA transfer count register_2 (DMATCR_2)
- DMA channel control register_2 (CHCR_2)
- DMA initial address register_2 (IAR_2)

Channel 3:

- DMA source address register_3 (SAR_3)
- DMA destination address register_3 (DAR_3)
- DMA transfer count register_3 (DMATCR_3)
- DMA channel control register_3 (CHCR_3)
- DMA initial address register_3 (IAR_3)

Common:

- DMA operation register (DMAOR)
- DMA extension resource selector 0 (DMARS0)
- DMA extension resource selector 1 (DMARS1)

10.3.1 DMA Source Address Register (SAR)

SAR is a 32-bit readable/writable register that specifies the source address of a DMA transfer. During a DMA transfer, this register indicates the next source address. When the data of an external device with DACK is transferred in the single address mode, the SAR is ignored.

To transfer data in 16 bits or in 32 bits, specify the address with 16-bit or 32-bit address boundary. When transferring data in 16-byte units, a 16-byte boundary (address 16n) must be set for the source address value. The SAR is undefined at reset and retains the current value in standby or module standby mode.

10.3.2 DMA Destination Address Register (DAR)

DAR is a 32-bit readable/writable register that specifies the destination address of a DMA transfer. This register includes count functions, and during a DMA transfer, this register indicates the next destination address. When the data of an external device with DACK is transferred in the single address mode, the DAR is ignored.

To transfer data in 16 bits or in 32 bits, specify the address with 16-bit or 32-bit address boundary. When transferring data in 16-byte units, a 16-byte boundary (address 16n) must be set for the source address value. The DAR is undefined at reset and retains the current value in standby or module standby mode.

10.3.3 DMA Transfer Count Register (DMATCR)

DMATCR is a 32-bit readable/writable register that specifies the DMA transfer count (bytes, words, or longwords). The number of transfers is 1 when the setting is H'00000001, 16777215 when H'00FFFFFF is set, and 16777216 (the maximum) when H'00000000 is set. During a DMA transfer, this register indicates the remaining transfer count.

The upper eight bits of DMATCR will return 0 if read, and should only be written with 0. To transfer data in 16 bytes, one 16-byte transfer (128 bits) counts one. The DMATCR is undefined at reset and retains the current value in standby or module standby mode.

10.3.4 DMA Channel Control Register (CHCR)

CHCR is 32-bit readable/writable register that controls the DMA transfer mode. The CHCR is initialized at reset and retains the current value in the standby or module standby mode.

Bit	Bit Name	Initial Value	R/W	Descriptions
31 to 27	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
26	RPT	0	R	Repeat Mode This bit specifies whether the DMA transfer is carried out in repeat mode or normal mode. 0: DMA transfer is carried out in normal mode 1: DMA transfer is carried out in repeat mode
25	RAS	0	R	Repeat Space Set This bit sets repeat space as transfer source or transfer destination 0: Repeat area is transfer destination 1: Repeat area is transfer source
24	DA	0	R	DMA Asynchronous This bit specifies whether the DREQ signal is synchronous signal or asynchronous signal This bit is valid only in CHCR_0. In CHCR_1 to CHCR_3, this bit is reserved. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed. 0: DREQ is an asynchronous signal 1: DREQ is a synchronous signal
23	DO	0	R/W	DMA Overrun This bit selects whether DREQ is detected by overrun 0 or by overrun 1. This bit is valid only in CHCR_0. This bit is reserved in CHCR_1 to CHCR_3. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed. 0: Detects DREQ by overrun 0 1: Detects DREQ by overrun 1

Bit	Bit Name	Initial Value	R/W	Descriptions
22	TL	0	R/W	Transfer End Level This bit specifies the TEND signal output is high active or low active. This bit is valid only in CHCR_0. This bit is reserved in CHCR_1 to CHCR_3. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed. 0: Low-active output of TEND 1: High-active output of TEND
21 to 18	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
17	AM	0	R/W	Acknowledge Mode AM specifies whether DACK is output in data read cycle or in data write cycle in dual address mode. In single address mode, DACK is always output regardless of the specification by this bit. This bit is valid only in CHCR_0. This bit is reserved in CHCR_1 to CHCR_3. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed. 0: DACK output in read cycle (Dual address mode) 1: DACK output in write cycle (Dual address mode)
16	AL	0	R/W	Acknowledge Level AL specifies the DACK (acknowledge) signal output is high active or low active. This bit is valid only in CHCR_0. This bit is reserved in CHCR_1 to CHCR_3. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed. 0: Low-active output of DACK 1: High-active output of DACK

Bit	Bit Name	Initial Value	R/W	Descriptions
15	DM1	0	R/W	Destination Address Mode
14	DM0	0	R/W	DM1 and DM0 select whether the DMA destination address is incremented, decremented, or left fixed. (In single address mode, DM1 and DM0 bits are ignored when data is transferred to an external device with DACK.) 00: Fixed destination address (setting prohibited in 16-byte transfer) 01: Destination address is incremented (+1 in 8-bit transfer, +2 in 16-bit transfer, +4 in 32-bit transfer, +16 in 16-byte transfer) 10: Destination address is decremented (–1 in 8-bit transfer, –2 in 16-bit transfer, –4 in 32-bit transfer; illegal setting in 16-byte transfer) 11: Reserved (setting prohibited)
13	SM1	0	R/W	Source Address Mode
12	SM0	0	R/W	SM1 and SM0 select whether the DMA source address is incremented, decremented, or left fixed. (In single address mode, SM1 and SM0 bits are ignored when data is transferred from an external device with DACK.) 00: Fixed source address (setting prohibited in 16-byte transfer) 01: Source address is incremented (+1 in 8-bit transfer, +2 in 16-bit transfer, +4 in 32-bit transfer, +16 in 16-byte transfer) 10: Source address is decremented (–1 in 8-bit transfer, –2 in 16-bit transfer, –4 in 32-bit transfer; illegal setting in 16-byte transfer) 11: Reserved (setting prohibited)

Bit	Bit Name	Initial Value	R/W	Descriptions
11	RS3	0	R/W	Resource Select
10	RS2	0	R/W	RS3 to RS0 specify which transfer requests will be sent to the DMAC. The changing of transfer request source should be done in the state that DMA enable bit (DE) is set to 0.
9	RS1	0	R/W	
8	RS0	0	R/W	

0	0	0	External request, dual address mode
0	0	1	Setting prohibited
0	1	0	External request / Single address mode External address space → external device with DACK
0	1	1	External request / Single address mode External device with DACK → external address space
1	0	0	Auto request
1	0	1	Setting prohibited
1	1	0	Setting prohibited
1	1	1	Setting prohibited
0	0	0	DMA expansion request module selection specification
0	0	1	Setting prohibited
0	1	0	Setting Prohibited
0	1	1	Setting Prohibited
1	0	0	Setting Prohibited
1	0	1	Setting Prohibited
1	1	0	Setting Prohibited
1	1	1	Setting Prohibited

Note: External request specification is valid only in CHCR_0; not valid in CHCR_1 to CHCR_3.

7	DL	0	R/W	DREQ Level (DL) and DREQ Edge Select (DS)
6	DS	0	R/W	These bits specify the sampling method of the DREQ pin input and the sampling level. These bits are valid only in CHCR_0. These bits are reserved in CHCR_1 to CHCR_3. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed. In channel 0, also, if the transfer request source is specified as an on-chip peripheral module or if an auto-request is specified, the specification by this bit is ignored. 00: DREQ detected in low level 01: DREQ detected at falling edge 10: DREQ detected in high level 11: DREQ detected at rising edge

Bit	Bit Name	Initial Value	R/W	Descriptions
5	TB	0	R/W	Transfer Bus Mode This bit specifies the bus mode when DMA transfers data. 0: Cycle steal mode (Initial Value) 1: Burst mode
4	TS1	0	R/W	Transfer Size
3	TS0	0	R/W	TS1 and TS0 specify the size of data to be transferred. Select the size of data to be transferred when the source or destination is an on-chip peripheral module register of which transfer size is specified. 00: Byte size 01: Word size (two bytes) 10: Longword size (four bytes) 11: 16-byte unit (four longword transfers)
2	IE	0	R/W	Interrupt Enable This bit specifies whether or not an interrupt request is generated to the CPU at the end of the DMA transfer. Setting this bit to 1 generates an interrupt request (DEI) to the CPU when TE bit is set to 1. 0: Interrupt request is not generated 1: Interrupt request is generated
1	TE	0	R/W*	Transfer End Flag This bit shows that DMA transfer ends. TE is set to 1 when data transfer ends when DMATCR becomes to 0. TE bit is not set to 1 in the following cases. • DMA transfer ends due to an NMI interrupt or DMA address error before DMATCR becomes to 0. • DMA transfer is ended by clearing the DE bit and DME bit in DMA operation register (DMAOR). This bit can only be cleared by writing 0 after reading 1. Even if the DE bit is set to 1 while this bit is set to 1, transfer is not enabled. 0: During the DMA transfer or DMA transfer has been suspended 1: Data transfer ends by the specified count (DMACTR = 0) Clearing condition: Writing 0 after TE = 1 read

Bit	Bit Name	Initial Value	R/W	Descriptions
0	DE	0	R/W	DMA Enable This bit enabler or disables the DMA transfer. In an auto request mode, DMA transfer starts by setting the DE bit and DME bit in DMAOR to 1. In this time, all of the bits TE, NMIF in DMAOR, and AE must be 0's. In an external request or peripheral module request, DMA transfer starts if DMA transfer request is generated by the devices or peripheral modules after setting the bits DE and DME to 1. In this case, however, all of the bits TE, NMIF, and AE must be 0's an in the case of auto request mode. Clearing the DE bit to 0 can terminate the DMA transfer. 0: DMA transfer disabled 1: DMA transfer enabled

Note: * Writing 0 is possible to clear the flag.

10.3.5 DMA Initial Address Register (IAR)

IAR is a 32-bit readable/writable register that specifies the initial address of SAR or DAR for repeating the DMA transfer.

To transfer data in 16 bits or in 32 bits, specify the address with 16-bit or 32-bit address boundary. When transferring data in 16-byte units, a 16-byte boundary must be set for the source address value. IAR is undefined at reset and retains the current value in standby or module standby mode.

10.3.6 DMA Operation Register (DMAOR)

DMAOR is a 32-bit readable/writable register that specifies the priority level of channels at the DMA transfer. This register shows the DMA transfer status. DMAOR is undefined at reset and retains the current value in the standby or module standby mode.

Bit	Bit Name	Initial Value	R/W	Description
31	—	All 0	R	Reserved
30				These bits are always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
29	CMS1	0	R/W	Cycle Steal Mode Select 1, 0
28	CMS0	0	R/W	These bits select either normal mode or intermittent mode in cycle steal mode. It is necessary that the bus modes of all channels be set to cycle steal mode to make the intermittent mode valid. 00: Normal mode 01: Reserved (setting prohibited) 10: Intermittent mode 16 Executes one DMA transfer in each of 16 clocks of an external bus clock. 11: Intermittent mode 64 Executes one DMA transfer in each of 64 clocks of an external bus clock.
27	—	All 0	R	Reserved
26				These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
25	PR1	0	R/W	Priority Mode
24	PR0	0	R/W	PR1 and PR0 select the priority level between channels when there are transfer requests for multiple channels simultaneously. 00: Fixed mode 1: CH0 > CH1 > CH2 > CH3 01: Fixed mode 2: CH0 > CH2 > CH3 > CH1 10: The status of the channel select round-robin mode: RCn bit is reflected to the priority. 11: All channel round-robin mode
23 to 19	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
18	AE	0	R/(W)*	Address Error Flag AE indicates that an address error occurred during DMA transfer. When the AE bit is set, the DMA transfer is not enabled even if the DE bit of CHCR and the DME bit of DMAOR are set to 1. This bit can only be cleared by writing 0 after reading 1. 0: No DMAC address error Clear conditions: Writing AE = 0 after AE = 1 read 1: DMAC address error
17	NMIF	0	R/(W)*	NMI Flag NMIF indicates that an NMI interrupt occurred. When the NMIF bit is set, the DMA transfer is not enabled even if the DE bit of CHCR and the DME bit of DMAOR are set to 1. Only 0 can be written to clear this bit after 1 is read. When the NMI is input, the DMA transfer in progress can be done in one transfer unit. When the DMAC is not in operational, the NMIF bit is set to 1 even if the NMI interrupt was input. 0: No NMI input Clearing conditions: Writing NMIF = 0 after NMIF = 1 read 1: NMI interrupt occurs

Bit	Bit Name	Initial Value	R/W	Description
16	DME	0	R/W	DMA Master Enable DME enables or disables DMA transfers on all channels. If the DME bit and the DE bit corresponding to each channel in CHCR are set to 1, transfer is enabled in the corresponding channel. If this bit is cleared during transfer, transfers in all the channels can be suspended. 0: Disable DMA transfers on all channels 1: Enable DMA transfers on all channels
15 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

Note: * Writing 0 is possible to clear the flag.

10.3.7 DMA Extension Resource Selector 0 and 1 (DMARS0 and DMARS1)

DMARS is a 16-bits readable/writable register that specifies the DMA transfer sources from peripheral modules in each channel. DMARS0 specifies for channels 0 and 1, DMARS1 specifies for channels 2 and 3.

The DMARS0 and DMARS1 are initialized at reset and retain the current values in standby or module standby mode.

Transfer requests from the various modules are specified by the MID and RID as shown in table 10.2. When MID/RID other than the values listed in table 10.2 is set, the operation of this LSI is not guaranteed. The transfer request from the DMARS register is valid only when the resource select bits (RS3 [3:0]) has been set to B'1000 for CHCR0 to CHCR3 registers. Otherwise, even if the DMARS has been set, transfer request source is not accepted.

- DMARS0

Bit	Bit Name	Initial Value	R/W	Description
15	C1MID5	0	R/W	Transfer request modules ID5 to ID0 for DMA channel 1 (MID) See table 10.2.
14	C1MID4	0	R/W	
13	C1MID3	0	R/W	
12	C1MID2	0	R/W	
11	C1MID1	0	R/W	
10	C1MID0	0	R/W	
9	C1RID1	0	R/W	Transfer request registers ID1 and ID0 for DMA channel 1 (RID) See table 10.2.
8	C1RID0	0	R/W	
7	C0MID5	0	R/W	Transfer request modules ID5 to ID0 for DMA channel 0 (MID) See table 10.2
6	C0MID4	0	R/W	
5	C0MID3	0	R/W	
4	C0MID2	0	R/W	
3	C0MID1	0	R/W	
2	C0MID0	0	R/W	
1	C0RID1	0	R/W	Transfer request registers ID1 and ID0 for DMA channel 0 (RID) See table 10.2.
0	C0RID0	0	R/W	

- DMARS1

Bit	Bit Name	Initial Value	R/W	Description
15	C3MID5	0	R/W	Transfer request modules ID5 to ID0 for DMA channel 3 (MID) See table 10.2.
14	C3MID4	0	R/W	
13	C3MID3	0	R/W	
12	C3MID2	0	R/W	
11	C3MID1	0	R/W	
10	C3MID0	0	R/W	
9	C3RID1	0	R/W	Transfer request registers ID1 and ID0 for DMA channel 3 (RID) See table 10.2.
8	C3RID0	0	R/W	
7	C2MID5	0	R/W	Transfer request modules ID5 to ID0 for DMA channel 2 (MID) See table 10.2.
6	C2MID4	0	R/W	
5	C2MID3	0	R/W	
4	C2MID2	0	R/W	
3	C2MID1	0	R/W	
2	C2MID0	0	R/W	
1	C2RID1	0	R/W	Transfer request modules ID1 and ID0 for DMA channel 2 (RID) See table 10.2.
0	C2RID0	0	R/W	

Table 10.2 DMARS Settings

Peripheral Module	Setting Value for One Channel (MID + RID)	MID	RID	Function
SCIF0	H'21	B'001000	B'01	Transmit
	H'22		B'10	Receive
SCIF1	H'29	B'001010	B'01	Transmit
	H'2A		B'10	Receive
SIOF	H'55	B'010101	B'01	Transmit
	H'56		B'10	Receive
USBF	H'73	B'011100	B'11	Transmit and receive 0
	H'70		B'00	Transmit and receive 1

10.4 Operation

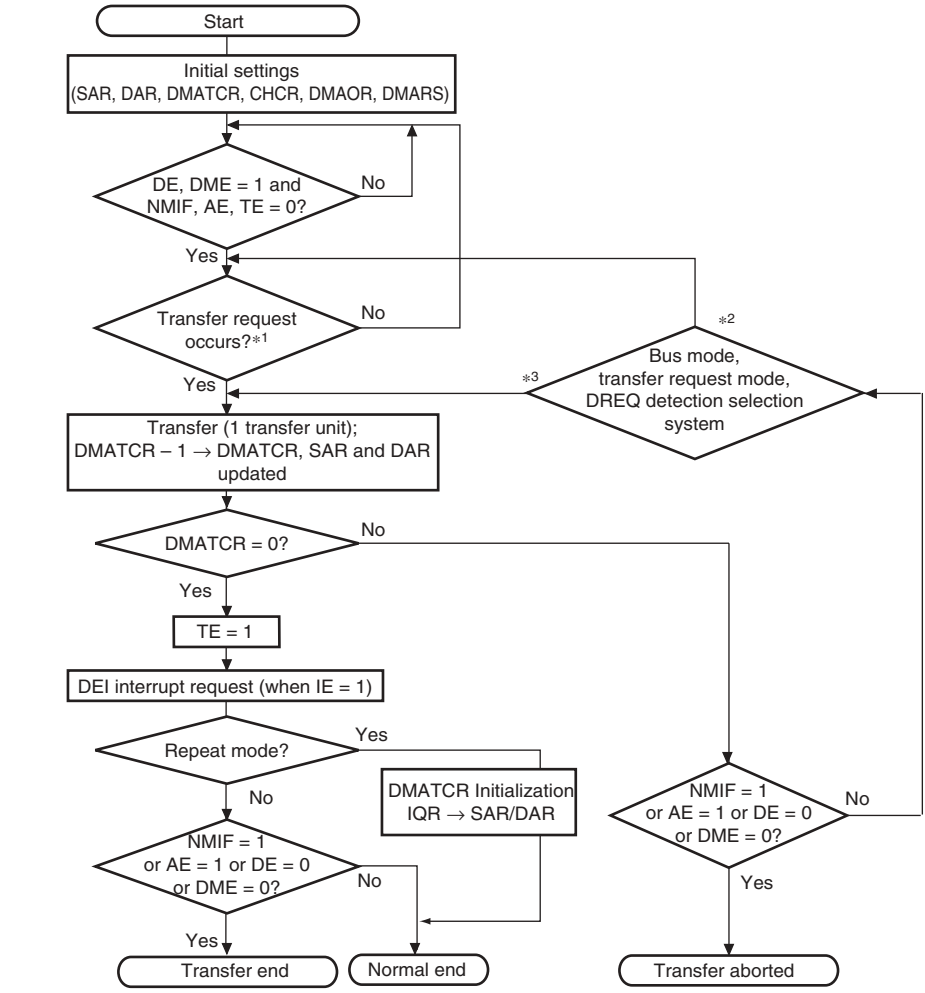
When there is a DMA transfer request, the DMAC starts the transfer according to the predetermined channel priority order; when the transfer end conditions are satisfied, it ends the transfer. Transfers can be requested in three modes: auto request, external request, and on-chip module request. The dual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

10.4.1 DMA Transfer Flow

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

1. Checks to see if transfer is enabled (DE = 1, DME = 1, TE = 0, AE = 0, NMIF = 0)
2. When a transfer request comes and transfer is enabled, the DMAC transfers 1 transfer unit of data (depending on the TS0 and TS1 settings). For an auto request, the transfer begins automatically when the DE bit and DME bit are set to 1. The DMATCR value will be decremented for each transfer. The actual transfer flows vary by address mode and bus mode.
3. When the specified number of transfer have been completed (when DMATCR reaches 0), the transfer ends normally. If the IE bit of the CHCR is set to 1 at this time, a DEI interrupt is sent to the CPU. If the RPT bit is set to 1, the IAR value is set in SAR or DAR and DMATCR is initialized to the written value. In this case, transfer can be continued only by clearing the TE bit.
4. When an NMI interrupt is generated, the transfer is aborted. Transfers are also aborted when the DE bit of the CHCR or the DME bit of the DMAOR are changed to 0.

Figure 10.2 is a flowchart of this procedure.



Notes: 1 . In auto-request mode, transfer begins when the bits are set as follows:
When NMIF and AE are cleared to 0, and TE is cleared to 0, and DE and DME are set to 1
2. DREQ = level detection in burst mode (external request) or cycle-steal mode.
3. DREQ = edge detection in burst mode (external request), or auto-request mode in burst mode.

Figure 10.2 DMA Transfer Flowchart

10.4.2 Repeat Mode Transfer

The repeat mode of DMAC can be used by setting 1 to the RPT bit of the CHCR_n register. When the repeat mode is set and data transfer ends (DMATCR_n becomes 0), the DEI interrupt is generated if the IE bit of CHCR_n is set, and then the value of IAR_n is copied onto SAR_n or DAR_n specified by RAS bit of CHCR_n, and DMATCR_n is restored to an initial value. DMA transfer can be executed continuously when the TE bit of the CHCR_n register is cleared to 0.

10.4.3 DMA Transfer Requests

DMA transfer requests are basically generated in either the data transfer source or destination, but they can also be generated by devices and on-chip peripheral modules that are neither the source nor the destination. Transfers can be requested in three modes: auto request, external request, and on-chip module request. The request mode is selected in the RS3 to RS0 bits of the DMA channel control registers 0 to 3 (CHCR_0 to CHCR_3), and the DMA extension resource selectors 0 and 1 (DMARS0 and DMARS1).

Auto-Request Mode: When there is no transfer request signal from an external source, as in a memory-to-memory transfer or a transfer between memory and an on-chip peripheral module unable to request a transfer, the auto-request mode allows the DMAC to automatically generate a transfer request signal internally. When the DE bits of CHCR_0 to CHCR_3 and the DME bit of the DMAOR are set to 1, the transfer begins so long as the TE bits of CHCR_0 to CHCR_3 and the NMIF bit of DMAOR are all 0.

External Request Mode: In this mode a transfer is performed at the request signal (DREQ) of an external device. This is valid only for DMA channel 0. Choose one of the modes shown in table 10.3 according to the application system. When this mode is selected, if the DMA transfer is enabled (DE = 1, DME = 1, TE = 0, AE = 0, NMIF = 0), a transfer is performed upon a request at the DREQ input.

Table 10.3 Selecting External Request Modes with the RS Bits

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

Choose to detect DREQ by either the edge or level of the signal input with the DREQ level (DL) bit and DS bit of CHCR_0 as shown in table 10.4. The source of the transfer request does not have to be the data transfer source or destination.

Table 10.4 Selecting External Request Detection with DL, DS Bits

CHCR		
DL	**DS**	**Detection of External Request**
0	0	Low level detection
	1	Falling edge detection
1	0	High level detection
	1	Rising edge detection

When DREQ is accepted, the DREQ pin becomes request accept disabled state (non-sensitive period). After issuing acknowledge signal DACK for the accepted DREQ, the DREQ pin again becomes request accept enabled state.

When DREQ is used by level detection, there are following two cases by the timing to detect the next DREQ after outputting DACK.

Overrun0: Transfer is aborted after the same number of transfer has been performed as requests.

Overrun1: Transfer is aborted after transfers have been performed for (the number of requests plus 1) times.

The DO bit in CHCR selects this overrun 0 or overrun 1. (Table 10.5)

Table 10.5 Selecting External Request Detection with DO Bit

CHCR	
DO	**External Request**
0	Overrun 0
1	Overrun 1

On-Chip Peripheral Module Request: In this mode (table 10.6), the transfer is performed in response to the transfer request signal (interrupt request signal) of an on-chip peripheral module. Transfer request signals comprise the transmit data empty transfer request and receive data full transfer request from the SCIF0, SCIF1, SIOF, and USBF.

When this mode is selected, if the DMA transfer is enabled (DE = 1, DME = 1, TE = 0, AE = 0, NMIF = 0), a transfer is performed upon the input of a transfer request signal.

When a transmit data empty transfer request of SCIF is set as the transfer request, the transfer destination must be the SCIF's transmit data register. Likewise, when the SCIF receive data full transfer request is set as the transfer request, the transfer source must be the SCIF's receive data register. These conditions also apply to the SIOF and USBF.

Table 10.6 Selecting On-Chip Peripheral Module Request Modes with the RS3 to RS0 Bits

CHCR	DMARS		DMA Transfer Request Source	DMA Transfer Request Signal	Source	Destination	Bus Mode
RS[3:0]	MID	RID					
1000	001000	01	SCIF0 transmitter	TDFE(transmit data FIFO empty interrupt)	Any	SCFTDR_0	Cycle steal
		10	SCIF0 receiver	RDF (receive data FIFO full interrupt)	SCFRDR_0	Any	Cycle steal
	001010	01	SCIF1 transmitter	TDFE (transmit data FIFO empty interrupt)	Any	SCFTDR_1	Cycle steal
		10	SCIF1 receiver	RDF (receive data FIFO full interrupt)	SCFRDR_1	Any	Cycle steal
	010101	01	SIOF transmitter	TDFE (transmit data FIFO empty interrupt)	Any	SIOF/SITDR_0	Cycle steal
		10	SIOF receiver	RDF (receive data FIFO full interrupt)	SIOF/SIRDR_0	Any	Cycle steal
	011100	11	USBF transmitter 0	Transmit data empty request 0	Any	EPDR2i, 5	Cycle steal
			USBF receiver 0	Receive data full request 0	EPDR2o, 6	Any	Cycle steal
		00	USBF transmitter 1	Transmit data empty request 1	Any	EPDR2i, 5	Cycle steal
			USBF receiver 1	Receive data full request 1	EPDR2o, 6	Any	Cycle steal

10.4.4 Channel Priority

When the DMAC receives simultaneous transfer requests on two or more channels, it selects a channel according to a predetermined priority order. The four modes (fixed mode 1, fixed mode 2, channel selective round-robin mode, and all-channel round-robin mode) are selected using the bits PR1 and PR0 in DMAOR.

Fixed Mode: In these modes, the priority levels among the channels remain fixed. There are two kinds of fixed modes as follows:

Fixed mode 1: CH0 > CH1 > CH2 > CH3

Fixed mode 2: CH0 > CH2 > CH3 > CH1

These are selected by the bits PR1 and the PR0 in DMAOR.

Round-Robin Mode: Each time one word, byte, longword, or 16-byte is transferred on one channel, the priority order is rotated. The channel on which the transfer was just finished rotates to the bottom of the priority order. The round-robin mode operation is shown in figure 10.3. The priority of the round-robin mode is CH0 > CH1 > CH2 > CH3 immediately after reset. When round-robin mode is selected, cycle steal mode and burst mode should not be mixed as the bus mode of two or more channels.

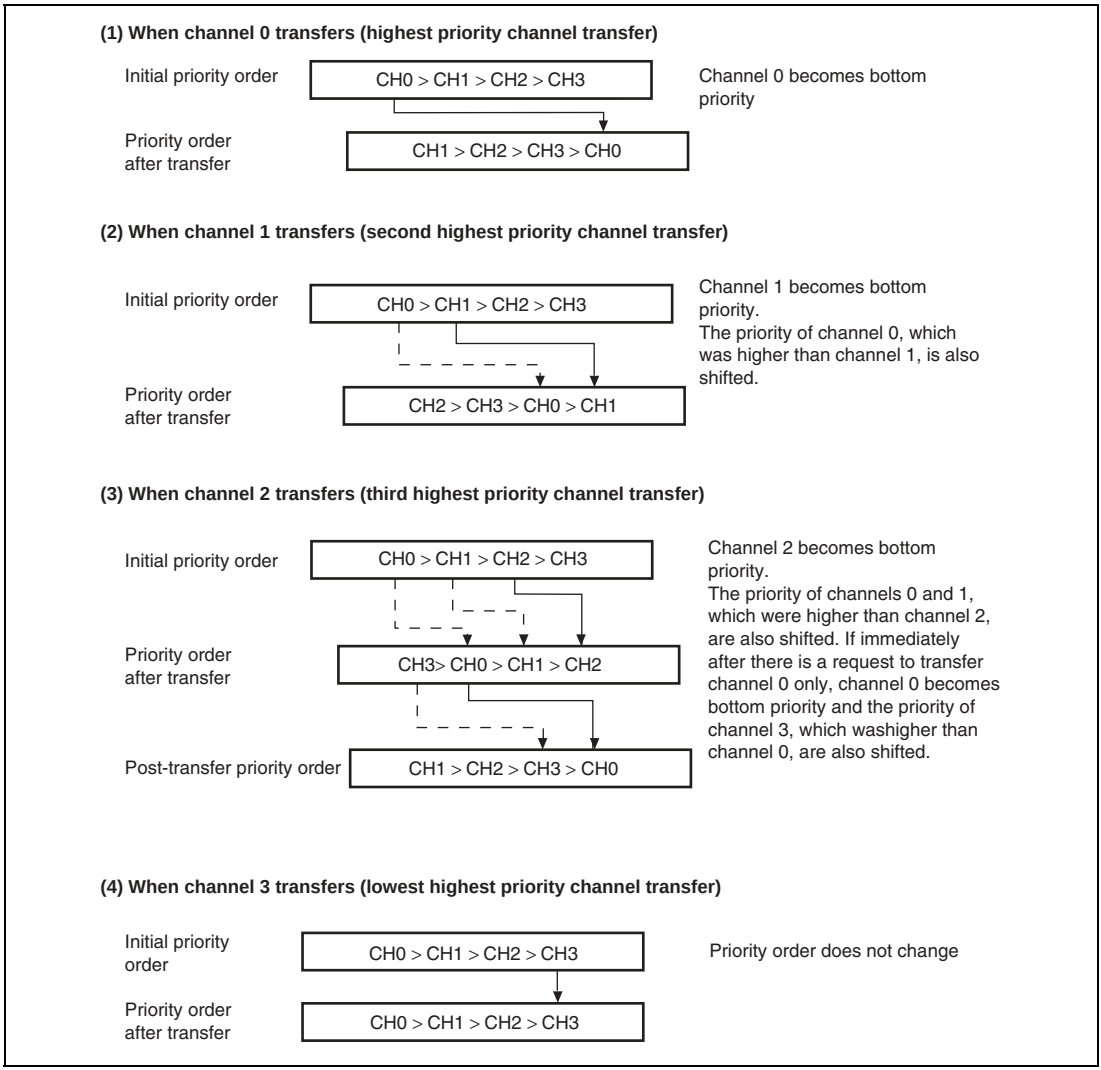


Figure 10.3 Round-Robin Mode

Figure 10.4 shows how the priority order changes when channel 0 and channel 3 transfers are requested simultaneously and a channel 1 transfer is requested during the channel 0 transfer. The DMAC operates as follows:

1. Transfer requests are generated simultaneously to channels 0 and 3.
2. Channel 0 has a higher priority, so the channel 0 transfer begins first (channel 3 waits for transfer).
3. A channel 1 transfer request occurs during the channel 0 transfer (channels 1 and 3 are both waiting)
4. When the channel 0 transfer ends, channel 0 becomes lowest priority.

5. At this point, channel 1 has a higher priority than channel 3, so the channel 1 transfer begins (channel 3 waits for transfer).
6. When the channel 1 transfer ends, channel 1 becomes lowest priority.
7. The channel 3 transfer begins.
8. When the channel 3 transfer ends, channels 3 and 2 shift downward in priority so that channel 3 becomes the lowest priority.

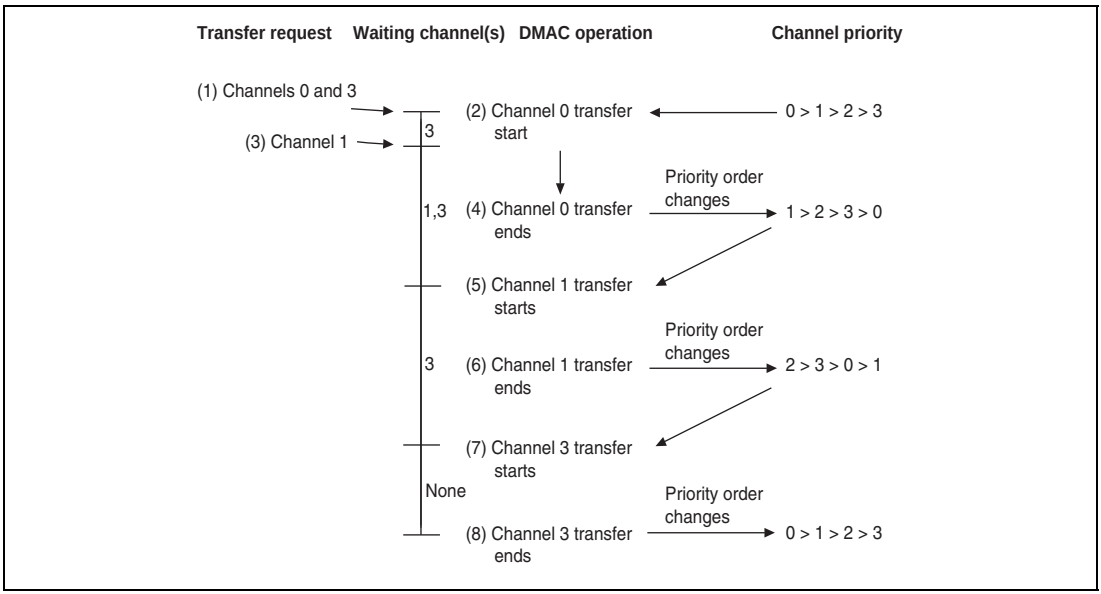


Figure 10.4 Changes in Channel Priority in Round-Robin Mode

10.4.5 DMA Transfer Types

DMA transfer has two types; single address mode transfer and dual address mode transfer, they depend on the number of bus cycles of access to source and destination. A data transfer timing depends on the bus mode, which has cycle steal mode and burst mode. The DMAC supports the transfers shown in table 10.7.

Table 10.7 Supported DMA Transfers

	Destination				
Source	**External Device with DACK**	**External Memory**	**Memory-Mapped External Device**	**On-Chip Peripheral Module**	**X/Y Memory U Memory**
External device with DACK	Not available	Dual, single	Dual, single	Not available	Not available
External memory	Dual, single	Dual	Dual	Dual	Dual
Memory-mapped external device	Dual, single	Dual	Dual	Dual	Dual
On-chip peripheral module	Not available	Dual	Dual	Dual	Dual
X/Y memory, U memory	Not available	Dual	Dual	Dual	Dual

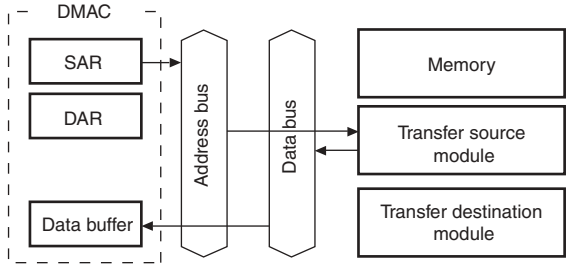
Notes: 1. Dual: Dual address mode
2. Single: Single address mode
3. 16-byte transfer is not available for on-chip peripheral modules.

Address Modes:

1. Dual Address Mode

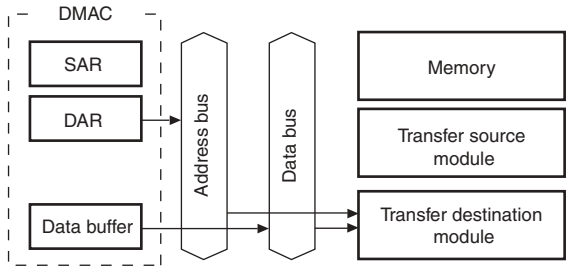
   In the dual address mode, both the transfer source and destination are accessed (selected) by an address. The source and destination can be located externally or internally.

   DMA transfer requires two bus cycles because data is read from the transfer source in a data read cycle and written to the transfer destination in a data write cycle. At this time, transfer data is temporarily stored in the DMAC. In the transfer between external memories as shown in figure 10.5, data is read to the DMAC from one external memory in a data read cycle, and then that data is written to the other external memory in a write cycle.



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

First bus cycle



The DAR value is an address and the value stored in the data buffer in the DMAC is written to the transfer destination module.

Second bus cycle

Figure 10.5 Data Flow of Dual Address Mode

Auto request, external request, and on-chip peripheral module request are available for the transfer request. DACK can be output in read cycle or write cycle in dual address mode. The AM bit of the channel control register (CHCR) can specify whether the DACK is output in read cycle or write cycle.

Figure 10.6 shows an example of DMA transfer timing in dual address mode.

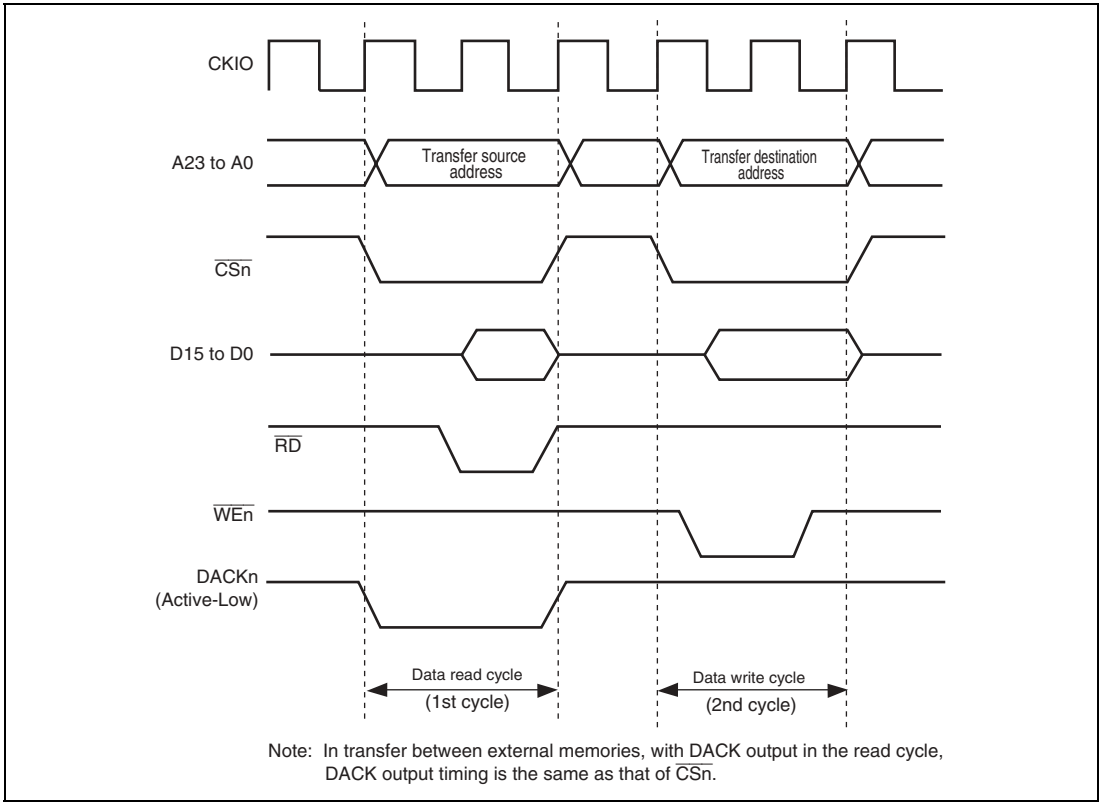


Figure 10.6 Example of DMA Transfer Timing in Dual Mode (Source: Ordinary memory, Destination: Ordinary memory)

2. Single Address Mode

In single address mode, either the transfer source or transfer destination peripheral device is accessed (selected) by means of the DACK signal, and the other device is accessed by address. In this mode, the DMAC performs one DMA transfer in one bus cycle, accessing one of the external devices by outputting the DACK transfer request acknowledge signal to it, and at the same time outputting an address to the other device involved in the transfer. For example, in the case of transfer between external memory and an external device with DACK shown in figure 10.7, when the external device outputs data to the data bus, that data is written to the external memory in the same bus cycle.

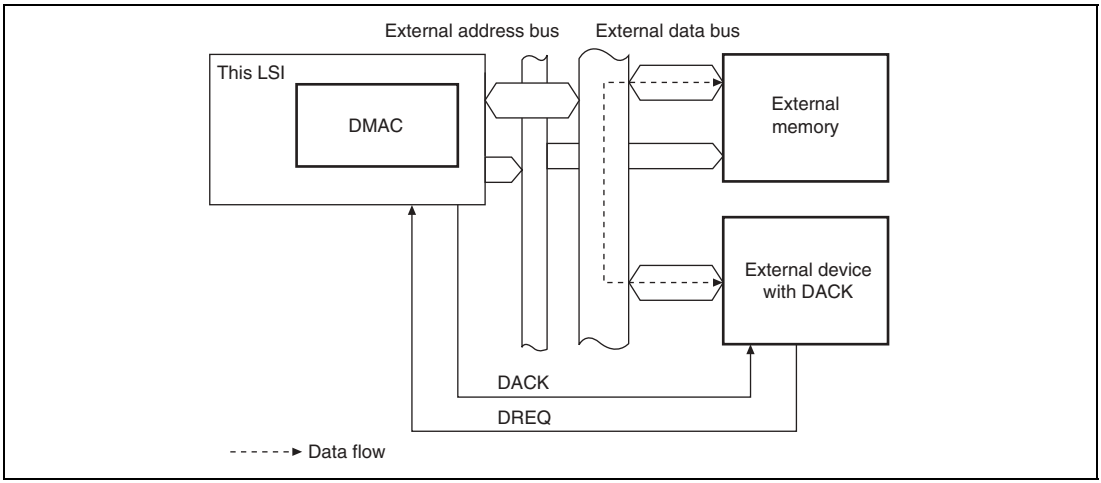


Figure 10.7 Data Flow in Single Address Mode

Two kinds of transfer are possible in single address mode: (1) transfer between an external device with DACK and a memory-mapped external device, and (2) transfer between an external device with DACK and external memory. In both cases, only the external request signal (DREQ) is used for transfer requests.

Figures 10.8 shows an example of DMA transfer timing in single address mode.

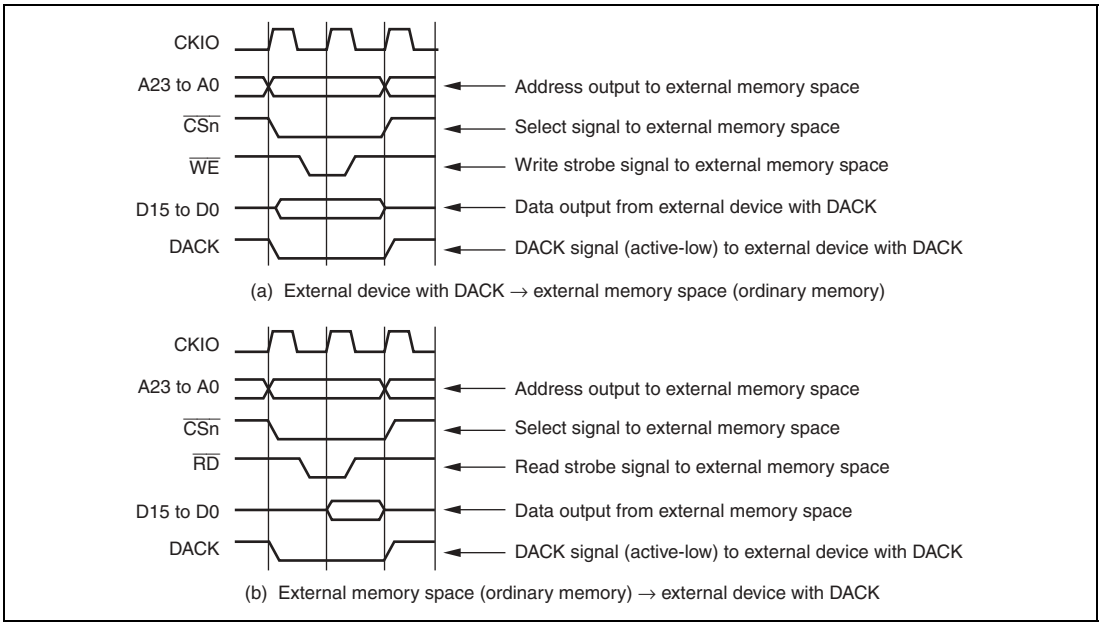


Figure 10.8 Example of DMA Transfer Timing in Single Address Mode

Bus Modes: There are two bus modes: cycle steal and burst. Select the mode in the TB bits of the channel control register (CHCR).

1. Cycle-Steal Mode

- Normal mode

  In the normal mode of cycle-steal, the bus mastership is given to another bus master after a one-transfer-unit (byte, word, long-word, or 16 bytes unit) DMA transfer. When another transfer request occurs, the bus masterships are obtained from the other bus master and a transfer is performed for one transfer unit. When that transfer ends, the bus mastership is passed to the other bus master. This is repeated until the transfer end conditions are satisfied.

  In the cycle-steal mode, transfer areas are not affected regardless of settings of the transfer request source, transfer source, and transfer destination.

Figure 10.9 shows an example of DMA transfer timing in the cycle steal mode. Transfer conditions shown in the figure are:

1. Dual address mode
2. DREQ low level detection

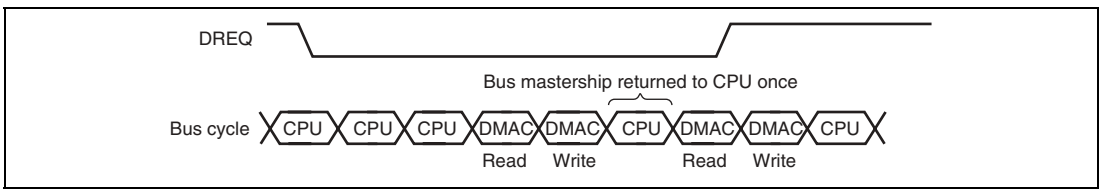


Figure 10.9 DMA Transfer Example in the Cycle-Steal Normal Mode (Dual Address, DREQ Low Level Detection)

- Intermittent Mode 16 and Intermittent Mode 64

  In the intermittent mode of cycle steal, DMAC returns the bus mastership to other bus master whenever a unit of transfer (byte, word, longword, or 16 bytes) is complete. If the next transfer request occurs after that, DMAC gets the bus mastership from other bus master after waiting for 16 or 64 clocks in Bϕ count. DMAC then transfers data of one unit and returns the bus mastership to other bus master. These operations are repeated until the transfer end condition is satisfied. It is thus possible to make lower the ratio of bus occupation by DMA transfer than the normal mode of cycle steal.

  When DMAC gets again the bus mastership, DMA transfer can be postponed in case of entry updating due to cache miss.

  This intermittent mode can be used for all transfer section; transfer requester, source, and destination. The bus modes, however, must be cycle steal mode in all channels.

Figure 10.10 shows an example of DMA transfer timing in cycle steal intermittent mode. Transfer conditions shown in the figure are:

1. Dual address mode
2. DREQ low level detection

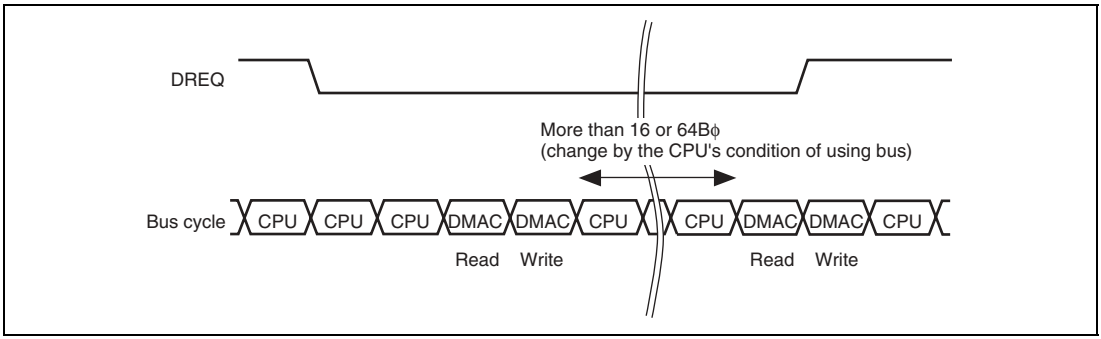


Figure 10.10 Example of DMA Transfer in Cycle Steal Intermittent Mode (Dual address, DREQ low level detection)

2. Burst Mode

   Once the bus mastership is obtained, the transfer is performed continuously until the transfer end condition is satisfied. In the external request mode with low level detection of the DREQ pin, however, when the DREQ pin is driven high, the bus passes to the other bus master after the DMAC transfer request that has already been accepted ends, even if the transfer end conditions have not been satisfied.

   The burst mode cannot be used for other than CMT00, CMT01, CMT02, and CMT03 when the on-chip peripheral module is the transfer request source. Figure 10.11 shows DMA transfer timing in the burst mode.

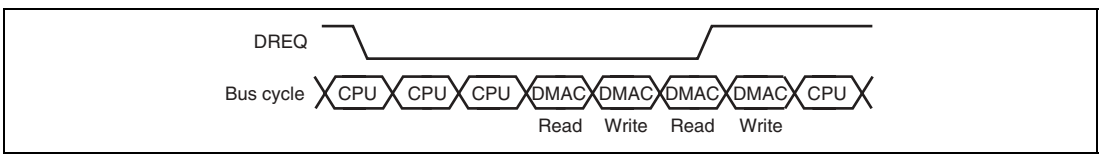


Figure 10.11 DMA Transfer Example in the Burst Mode (Dual Address, DREQ low level detection)

Relationship between Request Modes and Bus Modes by DMA Transfer Category: Table 10.9 shows the relationship between request modes and bus modes by DMA transfer category.

Table 10.8 Relationship of Request Modes and Bus Modes by DMA Transfer Category

Address Mode	Transfer Category	Request Mode	Bus Mode	Transfer Size (bits)	Usable Channels
Dual	External device with DACK and external memory	External	B/C	8/16/32/128	0
	External device with DACK and memory-mapped external device	External	B/C	8/16/32/128	0
	External memory and external memory	External /auto	B/C	8/16/32/128	0 to 5*3
	External memory and memory-mapped external device	External /auto	B/C	8/16/32/128	0 to 5*3
	Memory-mapped external device and memory-mapped external device	External /auto	B/C	8/16/32/128	0 to 5*3
	External memory and on-chip peripheral module	All*1	C	8/16/32/128*2	0 to 5*3
	Memory-mapped external device and on-chip peripheral module	All*1	C	8/16/32/128*2	0 to 5*3
	On-chip peripheral module and on-chip peripheral module	All*1	C	8/16/32/128*2	0 to 5*3
	X/Y memory, U memory and X/Y memory, U memory	External /auto	B/C	8/16/32/128	0 to 5*3
	X/Y memory, U memory and memory-mapped external device	External /auto	B/C	8/16/32/128	0 to 5*3
	X/Y memory, U memory and on-chip peripheral module	All*1	C	8/16/32/128*2	0 to 5*3
	X/Y memory, U memory and external memory	External /auto	B/C	8/16/32/128	0 to 5*3
Single	External device with DACK and external memory	External	B/C	8/16/32/128	0
	External device with DACK and memory-mapped external device	External	B/C	8/16/32/128	0

B: Burst, C: Cycle steal

Notes: 1. External requests, auto requests, and on-chip peripheral module requests are all available. In the case of on-chip peripheral module requests, transfer source register should be the transfer source or transfer.

2. Access size permitted for the on-chip peripheral module register functioning as the transfer source or transfer destination.

3. If the transfer request is an external request, only channel 0 is available.

Bus Mode and Channel Priority Order: When a given channel 1 is transferring in burst mode and there is a transfer request to a channel 0 with a higher priority in fixed mode (CH0 > CH1), the transfer of channel 0 will begin immediately.

At this time, the channel 1 transfer will continue when the channel 0 transfer has completely finished, even if channel 0 is operating in burst mode.

When channel 0 is operating in cycle steal mode, 1 will begin operating again without releasing bus mastership after channel 0 with a higher priority completes the transfer of one transfer unit. Then transfer is carried out between the two channels in the order channel 0, channel 1, channel 0, channel 1. In other words, the CPU cycle after transfer in cycle steal mode is switched with transfer in burst mode (hereafter, this is called preferential execution of burst mode). This example is shown in figure 10.12. When more than two channels in burst mode are conflicted, transfer on a channel with highest priority among those channels is executed.

When transfers of more than two channels are performed, the bus is not given to the bus master until conflicted transfers in burst mode have been completed.

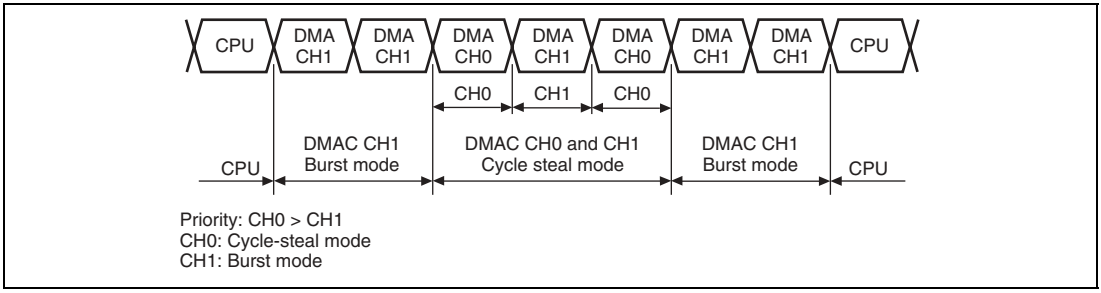


Figure 10.12 Bus State when Multiple Channels Are Operating

The priority order in round-robin mode is changed as is shown in figure 10.3. Note that in the burst mode, channel for cycle steal and channel for burst mode should not be mixed.

10.4.6 Number of Bus Cycle States and DREQ Pin Sampling Timing

Number of Bus Cycle States: When the DMAC is the bus master, the number of bus cycle states is controlled by the bus state controller (BSC) in the same way as when the CPU is the bus master. For details, see section 9, Bus State Controller (BSC).

DREQ Pin Sampling Timing: Figures 10.13 to 10.16 show the DREQ input sampling timings in each bus mode.

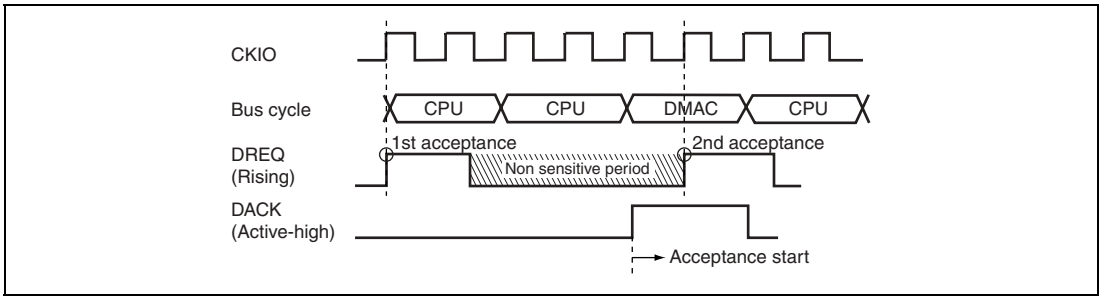


Figure 10.13 Example of DREQ Input Detection in Cycle Steal Mode Edge Detection

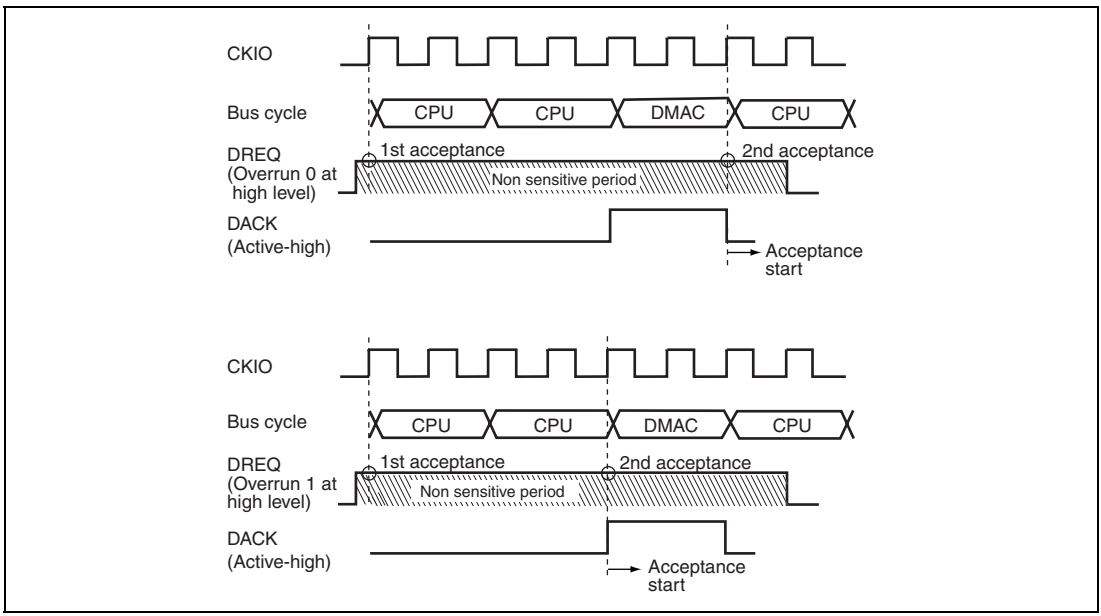


Figure 10.14 Example of DREQ Input Detection in Cycle Steal Mode Level Detection

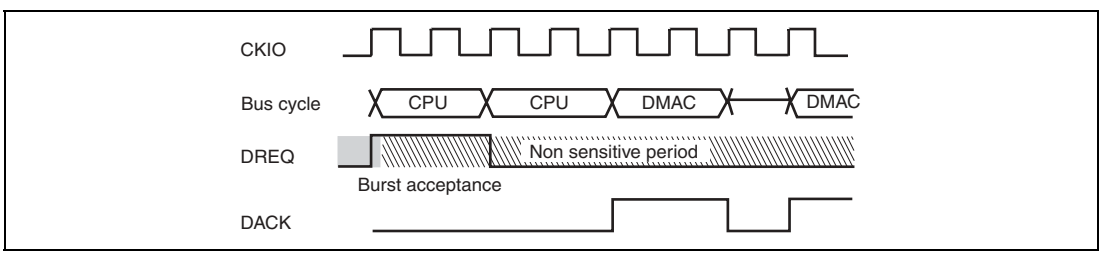


Figure 10.15 Example of DREQ Input Detection in Burst Mode Edge Detection

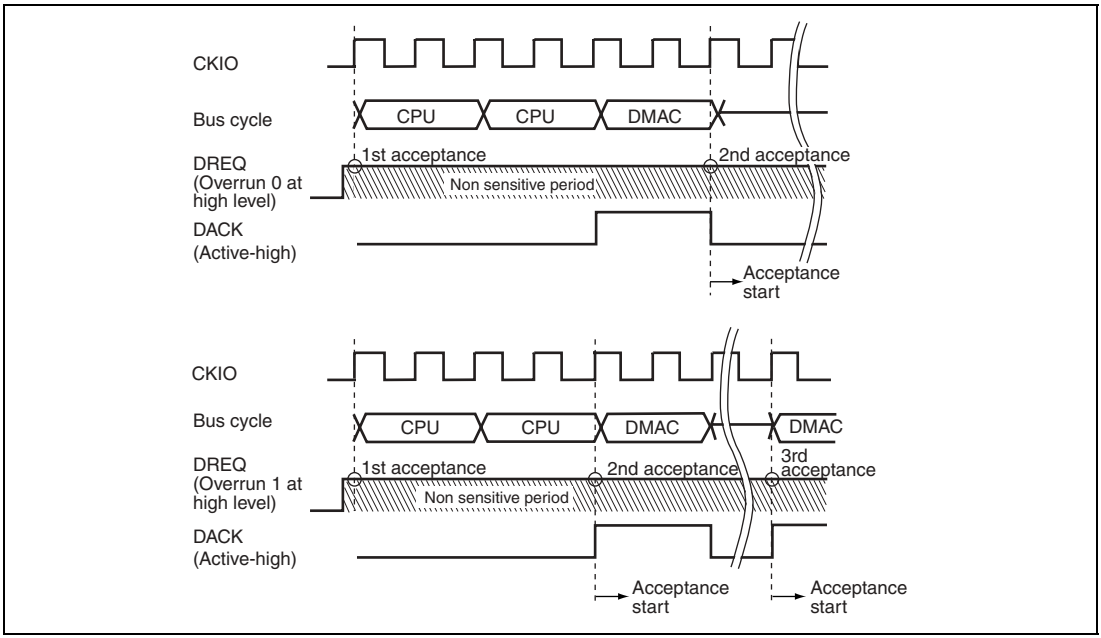


Figure 10.16 Example of DREQ Input Detection in Burst Mode Level Detection

Figure 10.17 shows the TEND output timing.

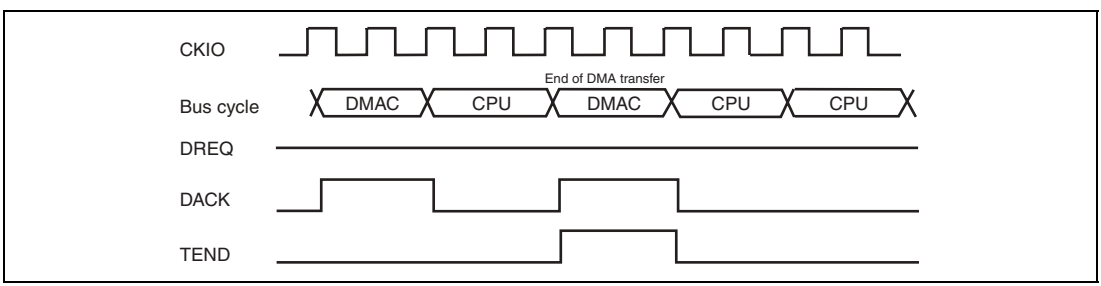
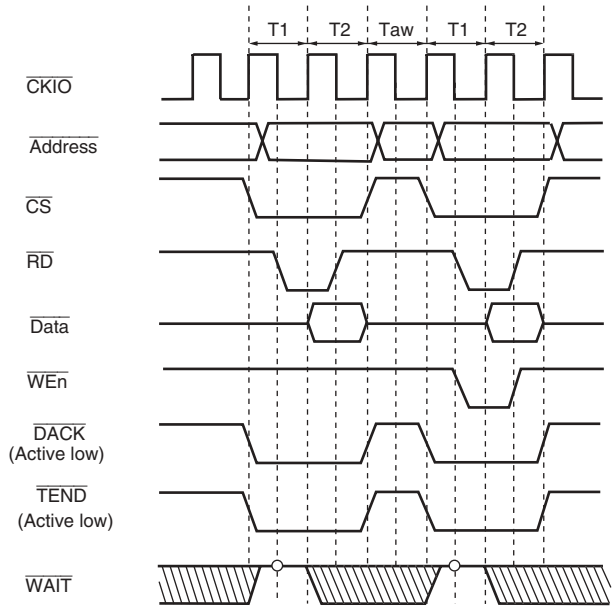


Figure 10.17 Example of DMA Transfer End Signal Timing in Cycle Steel Mode Level Detection

To execute a long word access to an 8-bit or 16-bit external device or to execute a word access to an 8-bit external device, the DACK and TEND outputs are divided for data alignment as shown in figure 10.18.



Note: TEND is asserted for the last transfer unit of DMA transfers.
If a transfer unit is divided into multiple bus cycles and
if CS is negated during the bus cycle, TEND is also divided.

Figure 10.18 BSC Ordinary Memory Access (No wait, Idle Cycle 1, Longword Access to 16-bit Device)

10.5 Usage Notes

Rewriting Channel Control Register (CHCR_n): When rewriting a channel control register (CHCR_n) of each DMAC channel, the DE bit of corresponding channel should be cleared to 0 precedently.

Shift to Power-Down Mode: Note that shifting to standby mode or module standby state by setting the module standby bit in DMAC should not be performed during DMA transfer. When shifting to software standby mode or module standby state, the DE bit of all channels should be cleared to 0 precedently.

Section 11 Clock Pulse Generator (CPG)

This LSI has a clock pulse generator (CPG) that generates an internal clock (Iϕ), a peripheral clock (Pϕ), and a bus clock (Bϕ). The CPG consists of an oscillator, PLL circuit, and divider circuit.

11.1 Features

- Three clock modes

  Selection of three clock modes by frequency range, turning on and off the PLL, a direct connection to a crystal unit, and external clock input.
- Three clocks generated independently

  An internal clock for the CPU and cache (Iϕ); a peripheral clock (Pϕ) for the on-chip circuits; a bus clock (Bϕ = CKIO) for the external bus interface.
- Frequency change function

  Internal and peripheral clock frequencies can be changed independently using the PLL (phase locked loop) circuit and divider circuit within the CPG. Frequencies are changed by software using frequency control register (FRQCR) settings.
- Power-down mode control

  The clock can be stopped for sleep mode and software standby mode and specific modules can be stopped using the module standby function.

A block diagram of the CPG is shown in figure 11.1.

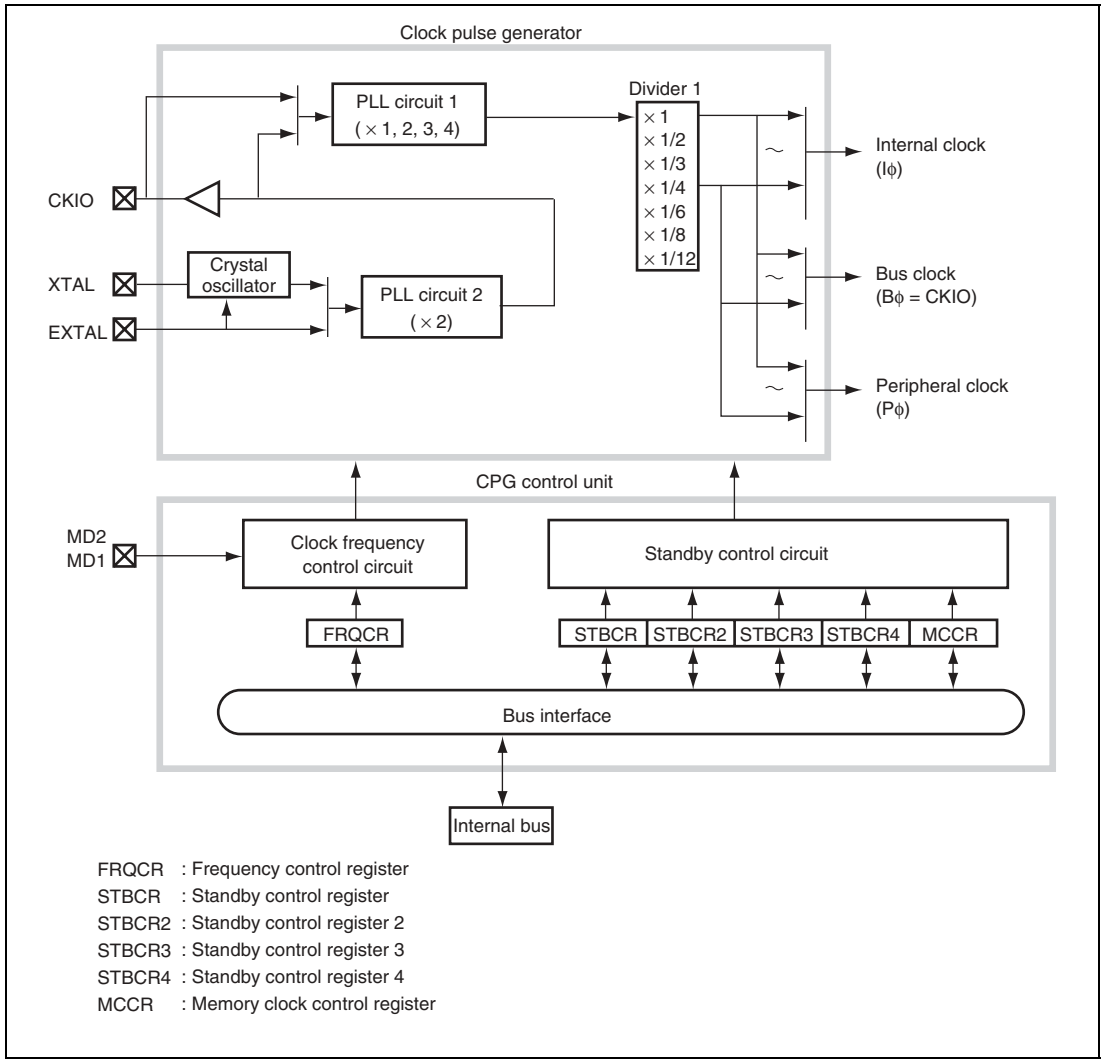


Figure 11.1 Block Diagram of Clock Pulse Generator

The clock pulse generator blocks function as follows:

PLL Circuit 1: PLL circuit 1 multiplies the input clock frequency by 1, 2, 3 or 4 from the CKIO pin. The multiplication rate is set using the frequency control register (FRQCR). When this is done, the phases of the leading edge of the internal clock, bus clock, and peripheral clock are controlled so that those will agree with the phase of the leading edge of the CKIO pin.

PLL Circuit 2: PLL circuit 2 doubles the input clock frequency from the crystal oscillator or EXTAL pin. The multiplication rate is fixed according to the clock operating mode. The clock operating mode is specified by the MD1 and MD2 pins. For details on clock operating mode, see table 11.2.

Crystal Oscillator: The crystal oscillator is an oscillator circuit in which a crystal resonator is connected to the XTAL pin or EXTAL pin. This can be used according to the clock operating mode 6.

Divider 1: Divider 1 generates a clock at the operating frequency used by the internal, bus, or peripheral clock. The operating frequency can be 1, 1/2, 1/3, 1/4, 1/6, 1/8, or 1/12 times the output frequency of PLL circuit 1, as long as it stays at or above the clock frequency of the CKIO pin. The division ratio is set in the frequency control register (FRQCR). *

Note: * Internal and peripheral clocks can be selected independently using the frequency control register (FRQCR). However, even though it is the selectable combination, if the number of operation frequencies which surpasses upper limit value of each operating frequency is set, operation of this LSI can not be guaranteed. Since operation frequency of bus clock should be the same as that of the CKIO, the division ratio is automatically set using the multiplication rate of PLL circuit 1.

Clock Frequency Control Circuit: The clock frequency control circuit controls the clock frequency using the MD1 and MD2 pins and the frequency control register (FRQCR).

Standby Control Circuit: The standby control circuit controls the states of the clock pulse generator and other modules during clock switching or sleep, or standby modes.

Frequency Control Register: The frequency control register (FRQCR) has control bits assigned for the following functions: clock output/non-output from the CKIO pin, the frequency multiplication ratio of PLL circuit 1, and the frequency division ratio of the internal clock and the peripheral clock.

Standby Control Register: The standby control registers (STBCR, STBCR2 to STBCR4) have bits for controlling the power-down modes. See section 13, Power-Down Modes, for more information.

Memory Clock Control Register: The memory clock control register (MCCR) has the memory clock in sleep and the control bits of 32-k clock oscillator for BT. For more information on MCCR, see section 13, Power-Down Mode.

11.2 Input/Output Pins

Table 11.1 lists the CPG pins and their functions.

Table 11.1 Clock Pulse Generator Pins Configuration

Pin Name	I/O	Function
MD1	Input	Mode control pin Set the clock operating mode
MD2	Input	Mode control pin Set the clock operating mode
XTAL	Output	Crystal output pin Connected to the crystal resonator
EXTAL	Input	Crystal input pin (clock input pin) Connected to the crystal resonator or used to input an external clock
CKIO	Input/ Output	Clock input/output pin Used as the external clock input or output pin

11.3 Clock Operating Modes

Table 11.2 shows the relationship between the mode control pins (MD2 and MD1) combinations and the clock operating modes. Table 11.3 shows the usable frequency ranges in the clock operating.

Table 11.2 Clock Operating Modes

	Pin Values		Clock I/O				
Mode	MD2	MD1	Source	Output	PLL2 On/Off	PLL1 On/Off	CKIO Frequency
5	1	0	EXTAL	CKIO	ON (×2)	ON (×1, 2, 3, 4)	(EXTAL) ×2
6	0	0	Crystal resonator	CKIO	ON (×2)	ON (×1, 2, 3, 4)	(Crystal resonator) ×2
7	1	1	CKIO	—	OFF	ON (×1, 2, 3, 4)	(CKIO)

Mode 5: This LSI is supplied with a clock that is doubled by the PLL circuit 2 after receiving an external clock from the EXTAL pin. This reduces the frequency of the externally generated clock. Input clock frequency and CKIO frequency range from 10 MHz to 30 MHz and from 20 MHz to 60 MHz, respectively.

Mode 6: This LSI is supplied with a clock that is doubled by the PLL circuit 2 after operating the on-chip crystal oscillator. This reduces the frequency of the externally generated clock. Input clock frequency and CKIO frequency range from 10 MHz to 30 MHz and from 20 MHz to 60 MHz, respectively.

Mode 7: In this mode, the CKIO pin functions as an input. This LSI is supplied with a clock that is wave-formed and multiplied by PLL circuit 1 and multiplied after receiving an external clock. This mode is effective for connecting synchronous DRAM because the PLL circuit 1. Input clock frequency (CKIO frequency) range from 20 MHz to 60 MHz.

Table 11.3 Possible Combination of Clock Modes and FRQCR Values

Mode	FRQCR Register Value	PLL Circuit 1	PLL Circuit 2	Clock Ratio (Iϕ:Bϕ:Pϕ)	Input Clock Frequency Range	CKIO Pin Frequency Range
5, 6	H'1000	ON (×1)	ON (×2)	2:2:2	10MHz to 15MHz	20MHz to 30MHz
	H'1001*[2]	ON (×1)	ON (×2)	2:2:1	10MHz to 30MHz	20MHz to 60MHz
	H'1002	ON (×1)	ON (×2)	2:2:2/3	10MHz to 30MHz	20MHz to 60MHz
	H'1003*[1]	ON (×1)	ON (×2)	2:2:1/2	10MHz to 30MHz	20MHz to 60MHz
	H'1004	ON (×1)	ON (×2)	2:2:1/3	10MHz to 30MHz	20MHz to 60MHz
	H'1005	ON (×1)	ON (×2)	2:2:1/4	10MHz to 30MHz	20MHz to 60MHz
	H'1006	ON (×1)	ON (×2)	2:2:1/6	10MHz to 30MHz	20MHz to 60MHz
	H'1101	ON (×2)	ON (×2)	4:2:2	10MHz to 15MHz	20MHz to 30MHz
	H'1103	ON (×2)	ON (×2)	4:2:1	10MHz to 30MHz	20MHz to 60MHz
	H'1104	ON (×2)	ON (×2)	4:2:2/3	10MHz to 30MHz	20MHz to 60MHz
	H'1105	ON (×2)	ON (×2)	4:2:1/2	10MHz to 30MHz	20MHz to 60MHz
	H'1106	ON (×2)	ON (×2)	4:2:1/3	10MHz to 30MHz	20MHz to 60MHz
	H'1111	ON (×2)	ON (×2)	2:2:2	10MHz to 15MHz	20MHz to 30MHz
	H'1113	ON (×2)	ON (×2)	2:2:1	10MHz to 30MHz	20MHz to 60MHz
	H'1202	ON (×3)	ON (×2)	6:2:2	10MHz to 15MHz	20MHz to 30MHz
	H'1204	ON (×3)	ON (×2)	6:2:1	10MHz to 20MHz	20MHz to 40MHz
	H'1206	ON (×3)	ON (×2)	6:2:1/2	10MHz to 20MHz	20MHz to 40MHz
	H'1303	ON (×4)	ON (×2)	8:2:2	10MHz to 15MHz	20MHz to 30MHz
	H'1305	ON (×4)	ON (×2)	8:2:1	10MHz to 15MHz	20MHz to 30MHz
	H'1306	ON (×4)	ON (×2)	8:2:2/3	10MHz to 15MHz	20MHz to 30MHz
	H'1313	ON (×4)	ON (×2)	4:2:2	10MHz to 15MHz	20MHz to 30MHz
7	H'1000	ON (×1)	OFF	1:1:1	20MHz to 30MHz	20MHz to 30MHz
	H'1001*[2]	ON (×1)	OFF	1:1:1/2	20MHz to 60MHz	20MHz to 60MHz
	H'1002	ON (×1)	OFF	1:1:1/3	20MHz to 60MHz	20MHz to 60MHz
	H'1003*[1]	ON (×1)	OFF	1:1:1/4	20MHz to 60MHz	20MHz to 60MHz
	H'1004	ON (×1)	OFF	1:1:1/6	20MHz to 60MHz	20MHz to 60MHz
	H'1005	ON (×1)	OFF	1:1:1/8	20MHz to 60MHz	20MHz to 60MHz
	H'1006	ON (×1)	OFF	1:1:1/12	20MHz to 60MHz	20MHz to 60MHz
	H'1101	ON (×2)	OFF	2:1:1	20MHz to 30MHz	20MHz to 30MHz
	H'1103	ON (×2)	OFF	2:1:1/2	20MHz to 60MHz	20MHz to 60MHz
	H'1104	ON (×2)	OFF	2:1:1/3	20MHz to 60MHz	20MHz to 60MHz

Mode	FRQCR Register Value	PLL Circuit 1	PLL Circuit 2	Clock Ratio (Iφ:Bφ:Pφ)	Input Clock Frequency Range	CKIO Pin Frequency Range
7	H'1105	ON (×2)	OFF	2:1:1/4	20MHz to 60MHz	20MHz to 60MHz
	H'1106	ON (×2)	OFF	2:1:1/6	20MHz to 60MHz	20MHz to 60MHz
	H'1111	ON (×2)	OFF	1:1:1	20MHz to 30MHz	20MHz to 30MHz
	H'1202	ON (×3)	OFF	3:1:1	26.7MHz to 30MHz	26.7MHz to 30MHz
	H'1204	ON (×3)	OFF	3:1:1/2	26.7MHz to 40MHz	26.7MHz to 40MHz
	H'1206	ON (×3)	OFF	3:1:1/4	26.7MHz to 40MHz	26.7MHz to 40MHz
	H'1303	ON (×4)	OFF	4:1:1	20MHz to 30MHz	20MHz to 30MHz
	H'1305	ON (×4)	OFF	4:1:1/2	20MHz to 30MHz	20MHz to 30MHz
	H'1306	ON (×4)	OFF	4:1:1/3	20MHz to 30MHz	20MHz to 30MHz
	H'1313	ON (×4)	OFF	2:1:1	20MHz to 30MHz	20MHz to 30MHz
	H'1333	ON (×4)	OFF	1:1:1	20MHz to 30MHz	20MHz to 30MHz

Notes: 1. Initial value immediately after reset in each mode.

2. The value is settable when the boot function is used.
The boot function should not be used with values other than these values.
There is a limitation in the SCIF transfer rate which can be set according to the input clock frequency. Refer to section 17.5.2, Clock Frequency and Data Transfer Rate when Using Boot Function in section 17, Boot Function (BOOT) for detail.

Cautions:

1. Input of divider circuit 1 becomes PLL circuit 1 output
2. Input of PLL circuit 1 becomes:
CKIO pin input in mode 7
PLL circuit2 output in modes 5 and 6
3. The frequency of the internal clock becomes:
The product of the frequency of the CKIO pin, the frequency multiplication ratio of PLL circuit 1, and the division ratio of divider 1
Set the internal clock frequency not more 120 MHz (T.B.D). Do not set the internal clock frequency lower than the frequency of the CKIO pin.
4. The frequency of the peripheral clock becomes:
The product of the frequency of the CKIO pin, the frequency multiplication ratio of PLL circuit 1, and the division ratio of divider 1. Set the peripheral clock frequency lower than or equal to 30 MHz. Note that the frequency should not be higher than the CKIO pin frequency.
5. ×1, ×2, ×3, or ×4 can be used as the multiplication ratio of PLL circuit 1. ×1, ×1/2, ×1/3, ×1/4, ×1/6, ×1/8 and ×1/12 can be selected as the division ratios of divider 1. Set the rate using frequency control register (FRQCR).
6. The output frequency of PLL circuit 1 is the product of the CKIO frequency and the multiplication ratio of PLL circuit 1. Use the output frequency not more than 120 MHz (T.B.D).
7. Bus clock frequency is always set to be equal to the frequency of the CKIO pin.
8. Clock ratios in table 11.3 are ratios of each frequency to one input clock frequency.

9. Operation of this LSI cannot be guaranteed using combinations other than those in table 11.3.

11.4 Register Descriptions

The CPG has the following registers. Refer the section 27, List of Registers, for the addresses of the registers and the register states in each operating mode.

- Frequency control register (FRQCR)
- Memory clock control register (MCCR)*
- Standby control register (STBCR, STBCR2 to STBCR4)*

Note: * For further information on MCCR, STBCR, and STBCR2 to STBCR4, see section 13 Power-Down Mode

11.4.1 Frequency Control Register (FRQCR)

FRQCR is a 16-bit readable/writable register used to specify whether a clock is output from the CKIO pin at standby mode, the frequency multiplication ratio of PLL circuit 1, and the frequency division ratio of the internal clock and the peripheral clock.

Only word access can be used on the FRQCR register.

This register is initialized by power-on reset from the $\overline{\text{RESETP}}$ pin and retains the previous value after manual reset and standby mode.

Bit	Bit Name	Initial Value	R/W	Description
15 to 13	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
12	CKOEN	1	R/W	Clock Output Enable When the HIZCNT bit in CMNCR is set to 1, CKOEN specifies whether a clock is output from the CKIO pin, or whether the CKIO pin is placed in the level-fixed state during release of the standby mode. If CKOEN is cleared to 0 and HIZCNT in CMNCR is set to 1, the CKIO pin is fixed at low. Therefore, the malfunction of an external circuit because of an unstable CKIO clock during release of the standby mode can be prevented. In clock operating mode 7, the CKIO pin functions as an input regardless of this bit value. 0: CKIO pin goes low during release of the standby mode. 1: Clock is output from CKIO pin
11	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
10	STC2	0	R/W	Frequency Multiplication Ratio
9	STC1	0	R/W	000: × 1 time
8	STC0	0	R/W	001: × 2 times 010: × 3 times 011: × 4 times 100: Setting prohibited 101: Setting prohibited 110: Setting prohibited 111: Setting prohibited
7	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
6	IFC2	0	R/W	Internal Clock Frequency Division Ratio
5	IFC1	0	R/W	These bits specify the frequency division ratio of the internal clock with respect to the output frequency of PLL circuit 1.
4	IFC0	0	R/W	
				000: × 1 time
				001: × 1/2 time
				010: Setting prohibited
				011: × 1/4 time
				100: Setting prohibited
				101: Setting prohibited
				110: Setting prohibited
				111: Setting prohibited
3	—	0	R	Reserved
				This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
2	PFC2	0	R/W	Peripheral Clock Frequency Division Ratio
1	PFC1	1	R/W	These bits specify the division ratio of the peripheral clock frequency with respect to the output frequency of PLL circuit 1.
0	PFC0	1	R/W	
				000: × 1 time
				001: × 1/2 time
				010: × 1/3 time
				011: × 1/4 time
				100: × 1/6 time
				101: × 1/8 time
				110: × 1/12 time
				110: Setting prohibited
				111: Setting prohibited

Note: The operations in the prohibited setting are not guaranteed.

11.5 Changing the Frequency

The frequency of the internal clock and peripheral clock can be changed either by changing the multiplication rate of PLL circuit 1 or by changing the division rates of divider 1. All of these are controlled by software through the frequency control register. The methods are described below.

11.5.1 Changing the Multiplication Rate

A PLL settling time is required when the multiplication rate of PLL circuit 1 is changed. The on-chip WDT counts the settling time.

1. In the initial state, the multiplication rate of PLL circuit 1 is 1.
2. Set a value that will become the specified oscillation settling time in the WDT and stop the WDT. The following must be set:

   WTCSR register TME bit = 0: WDT stops

   WTCSR register CKS2 to CKS0 bits: Division ratio of WDT count clock

   WTCNT counter: Initial counter value
3. Set the desired value in the STC[2:0] bits. The division ratio can also be set in the IFC[2:0] and PFC[2:0] bits.
4. The processor pauses internally and the WDT starts incrementing. The internal, bus, and peripheral clocks stop and the WDT is supplied with the clock. The clock will continue to be output at the CKIO pin.
5. Supply of the clock that has been set begins at WDT count overflow, and the processor begins operating again. The WDT stops after it overflows.

Note: When changing the clock frequency multiplication ratio, clear both UMCLKC bit and XYMCLKC bit of memory clock control register (MCCR) to 0.

11.5.2 Changing the Division Ratio

The WDT will not count unless the multiplication rate is changed simultaneously.

1. In the initial state, IFC[2:0] = B'000 and PFC[2:0] = B'011
2. Set the IFC[2:0] and PFC[2:0] bits to the new division ratio. The values that can be set are limited by the clock mode and the multiplication rate of PLL circuit 1. Note that if the wrong value is set, the processor will malfunction.
3. The clock is immediately supplied at the new division ratio.

11.6 Notes on Board Design

Note on Using a Crystal Resonator: When designing the board, place the crystal resonator, capacitors CL1 and CL2, and damping register R as close as possible to the XTAL and EXTAL pins. To prevent inductive interference in oscillation, ensure that the connection points of the capacitor to be connected to the crystal resonator are common and no other signal lines cross the signal lines for these pins.

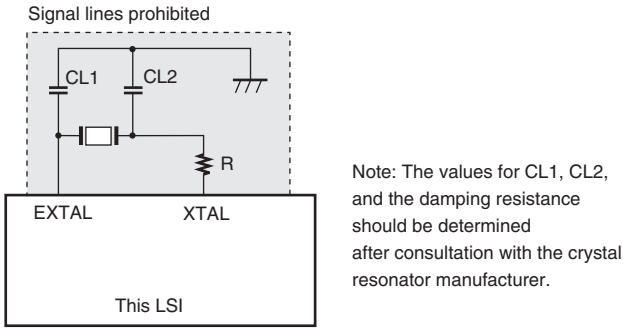


Figure 11.2 Note on Using a Crystal Resonator

Notes on Bypass Capacitor: A multilayer ceramic capacitor must be inserted for each pair of Vss and Vcc as a bypass capacitor. The bypass capacitor must be inserted as close as possible to the power supply pins of the LSI. Note that an appropriate bypass capacitor must be selected according to the capacitance and frequency of the LSI to be used.

Notes on Using a PLL Oscillator Circuit: Circuitry as shown in figure 11.3 is recommended around the PLL. In the Vcc and Vss connection pattern for the PLL, signal lines from the board power supply pins must be as short as possible and pattern width must be as wide as possible to reduce inductive interference.

In clock operating mode 7, the EXTAL pin is pulled up and the XTAL pin is left open.

Since the analog power supply pins of the PLL are sensitive to the noise, the system may malfunction due to inductive interference at the power supply pins. To prevent such malfunction, the analog power supply pin Vcc and digital power supply pin Vcc should not supply the same resources on the board if at all possible.

Please read “Vcc for PLL” above-mentioned as “VDD for PLL” when you do not select a built-in regulator (i.e. VDD external power supply is used).

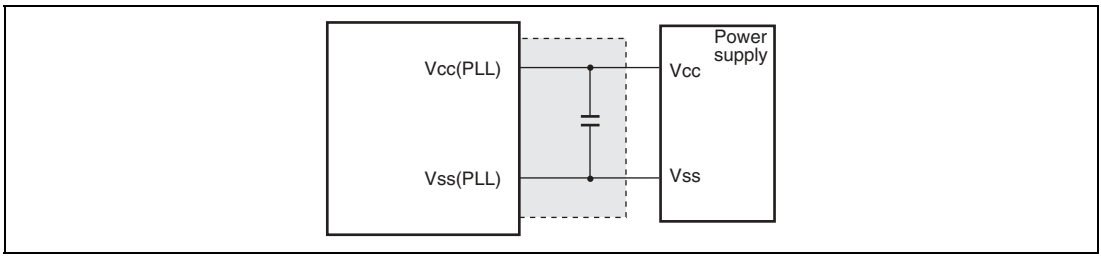


Figure 11.3 Note on Using a PLL Oscillator Circuit

Section 12 Watchdog Timer (WDT)

This LSI includes the watchdog timer (WDT), which enables reset on overflow of the counter when the value of the counter has not been updated because of a system malfunction.

The watchdog timer (WDT) is a single-channel timer that uses a peripheral clock as an input and counts the clock settling time and is used to clear standby mode and temporary standbys, such as frequency changes. It can also be used as an interval timer.

12.1 Features

The WDT has the following features:

- Can be used to ensure the clock settling time: Use the WDT to cancel a standby mode and the temporary standbys which occur when the clock frequency is changed.
- Can switch between watchdog timer mode and interval timer mode.
- Generates internal resets in watchdog timer mode: Internal resets occur after counter overflow.
- Interrupt generation in interval timer mode
  An interval timer interrupt is generated when the counter overflows.
- Choice of eight counter input clocks
  Eight clocks (×1 to ×1/4096) that are obtained by dividing the peripheral clock can be selected.
- Power-on reset or manual reset can be selected as a reset

Figure 12.1 shows a block diagram of the WDT.

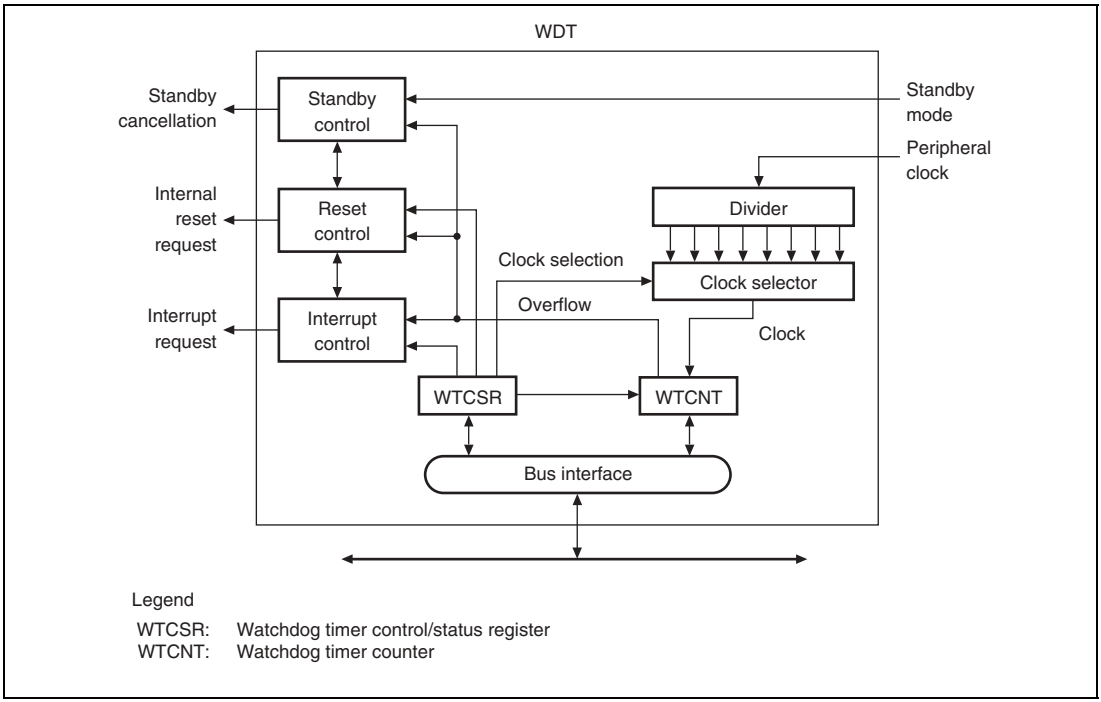


Figure 12.1 Block Diagram of the WDT

12.2 Register Descriptions

The WDT has the following two registers. Refer to the section 27, List of Registers, for the details of the addresses of these registers and the state of registers in each operating mode.

- Watchdog timer counter (WTCNT)
- Watchdog timer control/status register (WTCSR)

12.2.1 Watchdog Timer Counter (WTCNT)

The watchdog timer counter (WTCNT) is an 8-bit readable/writable register that increments on the selected clock. When an overflow occurs, it generates a reset in watchdog timer mode and an interrupt in interval timer mode. The WTCNT counter is initialized to H'00 only by a power-on reset caused by the $\overline{\text{RESETP}}$ pin. Use a word access to write to the WTCNT counter, writing H′5A in the upper byte. Use a byte access to read the WTCNT.

Note: The WTCNT differs from other registers in the prevention of erroneous writes. See section 12.2.3, Notes on Register Access, for details.

12.2.2 Watchdog Timer Control/Status Register (WTCSR)

The watchdog timer control/status register (WTCSR) is an 8-bit readable/writable register composed of bits to select the clock used for the count, bits to select the timer mode, overflow flags and enable bits. The WTCSR register holds its value in an internal reset due to WDT overflow. The WTCSR register is initialized only by a power-on reset via the $\overline{\text{RESETP}}$ pin.

When used to count the clock settling time for canceling a standby, it retains its value after counter overflow. Use a word access to write to the WTCSR counter, writing H'A5 in the upper byte. Use a byte access to read the WTCSR.

Note: The WTCNT differs from other registers in the prevention of erroneous writes. See section 12.2.3, Notes on Register Access, for details.

Bit	Bit Name	Initial Value	R/W	Description
7	TME	0	R/W	Timer Enable Starts and stops timer operation. Clear this bit to 0 when using the WDT in standby mode or when changing the clock frequency. 0: Timer disabled Count-up stops and WTCNT value is retained 1: Timer enabled
6	WT/$\overline{IT}$	0	R/W	Timer Mode Select Selects whether to use the WDT as a watchdog timer or an interval timer. 0: Use as interval timer 1: Use as watchdog timer Note: If WT/IT is modified when the WDT is running, the up-count may not be performed correctly.
5	RSTS	0	R/W	Reset Select Selects the type of reset when the WTCNT overflows in watchdog timer mode. In interval timer mode, this setting is ignored. 0: Power-on reset 1: Manual reset
4	WOVF	0	R/W	Watchdog Timer Overflow Indicates that the WTCNT has overflowed in watchdog timer mode. This bit is not set in interval timer mode. 0: No overflow 1: WTCNT has overflowed in watchdog timer mode
3	IOVF	0	R/W	Interval Timer Overflow Indicates that the WTCNT has overflowed in interval timer mode. This bit is not set in watchdog timer mode. 0: No overflow 1: WTCNT has overflowed in interval timer mode

Bit	Bit Name	Initial Value	R/W	Description
2	CKS2	0	R/W	Clock Select
1	CKS1	0	R/W	These bits select the clock to be used for the WTCNT count from the eight types obtainable by dividing the peripheral clock. The overflow period that is shown inside the parenthesis in the table is the value when the peripheral clock (Pϕ) is 15 MHz.
0	CKS0	0	R/W	

Bits 2 to 0	Clock Ratio	Overflow Cycle
000	1	(17 us)
001	1/4	(68 us)
010	1/16	(273 us)
011	1/32	(546 us)
100	1/64	(1.09 ms)
101	1/256	(4.36 ms)
110	1/1024	(17.48 ms)
111	1/4096	(69.91 ms)

Note: If bits CKS2 to CKS0 are modified when the WDT is running, the up-count may not be performed correctly. Ensure that these bits are modified only when the WDT is not running.

12.2.3 Notes on Register Access

The watchdog timer counter (WTCNT) and watchdog timer control/status register (WTCSR) are more difficult to write to than other registers. The procedures for reading writing to these registers are given below.

Writing to WTCNT and WTCSR: These registers must be written by a word transfer instruction. They cannot be written by a byte or longword transfer instruction. When writing to WTCNT, set the upper byte to H'5A and transfer the lower byte as the write data, as shown in figure 12.2. When writing to WTCSR, set the upper byte to H'A5 and transfer the lower byte as the write data. This transfer procedure writes the lower byte data to WTCNT or WTCSR.

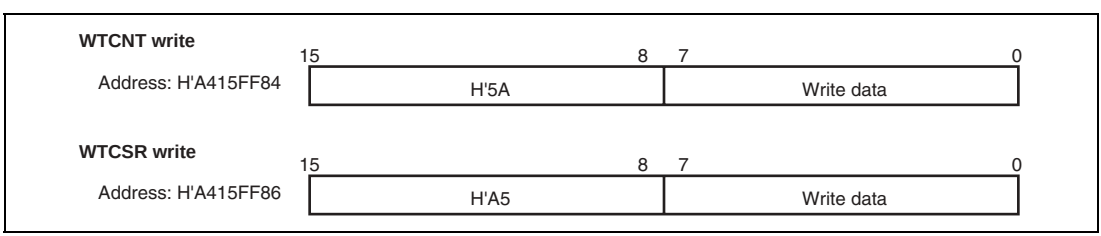


Figure 12.2 Writing to WTCNT and WTCSR

12.3 Operation

12.3.1 Canceling Software Standby

The WDT can be used to cancel software standby mode with an interrupt such as an NMI interrupt. The procedure is described below. (The WDT does not operate when resets are used for canceling, so keep the $\overline{\text{RESETP}}$ pin low until the clock stabilizes.)

1. Before transitioning to software standby mode, always clear the TME bit in WTCSR to 0. When the TME bit is 1, an erroneous reset or interval timer interrupt may be generated when the count overflows.
2. Set the type of count clock used in the CKS2 to CKS0 bits in WTCSR and the initial values for the counter in the WTCNT. These values should ensure that the time till count overflow is longer than the clock oscillation settling time. This setting does not depend on a value of the WT/$\overline{\text{IT}}$ bit.
3. Move to standby mode by executing a SLEEP instruction to stop the clock.
4. The WDT starts counting by detecting the edge change of the NMI signal.
5. When the WDT count overflows, the CPG starts supplying the clock and the processor resumes operation. The WOVF and IOVF flags in WTCSR are not set when this happens.
6. Since the WDT continues counting from H'00, set the STBY bit in the STBCR register to 0 in the interrupt processing program and this will stop the WDT. When the STBY bit remains 1, the LSI again enters the standby mode when the WDT has counted up to H'80. This standby mode can be canceled by power-on resets.

12.3.2 Changing the Frequency

To change the frequency used by the PLL, use the WDT. When changing the frequency only by switching the divider, do not use the WDT.

1. Before changing the frequency, always clear the TME bit in WTCSR to 0. When the TME bit is 1, an erroneous reset or interval timer interrupt may be generated when the count overflows.
2. Set the type of count clock used in the CKS2 to CKS0 bits of WTCSR and the initial values for the counter in the WTCNT counter. These values should ensure that the time till count overflow is longer than the clock oscillation settling time. This setting does not depend on a value of the WT/$\overline{\text{IT}}$ bit.
3. When the frequency control register (FRQCR) is written, the processor stops temporarily. The WDT starts counting.
4. When the WDT count overflows, the CPG resumes supplying the clock and the processor resumes operation. The WOVF and IOVF flags in WTCSR are not set when this happens.
5. The counter stops at the values H'00.

6. Before changing the WTCNT after the execution of the frequency change instruction, always confirm that the value of the WTCNT is H'00 by reading the WTCNT.

12.3.3 Using Watchdog Timer Mode

1. Clear the TME bit of WTCSR to 0. If the TME bit is set to 1, the reset or the interval timer interrupt might be generated by error operation at the counter overflow.
2. Set the WT/$\overline{\text{IT}}$ bit in the WTCSR to 1, set the reset type in the RSTS bit, set the type of count clock in the CKS2–CKS0 bits, and set the initial value of the counter in the WTCNT.
3. Set the TME bit in WTCSR to 1 to start the count in watchdog timer mode.
4. While operating in watchdog timer mode, rewrite the counter periodically to H'00 to prevent the counter from overflowing.
5. When the counter overflows, the WDT sets the WOVF flag in WTCSR to 1 and generates the type of reset specified by the RSTS bit. The counter then resumes counting.

12.3.4 Using Interval Timer Mode

When operating in interval timer mode, interval timer interrupts are generated at every overflow of the counter. This enables interrupts to be generated at set periods.

1. Clear the TME bit of WTCSR to 0. If the TME bit is set to 1, the reset or the interval timer interrupt might be generated by error operation at the counter overflow.
2. Clear the WT/$\overline{\text{IT}}$ bit in the WTCSR to 0, set the type of count clock in the CKS2–CKS0 bits, and set the initial value of the counter in the WTCNT.
3. Set the TME bit in WTCSR to 1 to start the count in interval timer mode.
4. When the counter overflows, the WDT sets the IOVF in WTCSR to 1 and an interval timer interrupt request is sent to INTC. The counter then resumes counting.
5. To clear the interval timer interrupt request in the exception handling routine, clear the flag which is set in the IOVF bit in WTCSR. For operation of interrupt exception handling, see section 4, Exception Handling and section 8, Interrupt Controller.

Section 13 Power-Down Modes

This LSI has three types of the power-down modes: sleep mode, software standby mode, and module standby function. In power-down mode, the CPU and some on-chip peripheral modules are halted. The power-down mode is cancelled by reset or interrupt sources.

The power dissipation is decreased in addition by stopping a built-in regulator if internal circuit power V_{DD}(= 1.5 V) is supplied from the external power supply.

13.1 Features

1. Sleep mode
2. Software standby mode
3. Module standby function
4. V_{DD} (= 1.5 V) can be supplied from the outside with turning off a built-in regulator.

13.1.1 Power-Down Modes

This LSI supports the following power-down modes and function:

1. Sleep mode
2. Software standby mode
3. Module standby function (cache, X/Y memory, U memory, DMAC, UBC, H-UDI and on-chip peripheral module)*

Note: * Also the USBH and USBF have the module standby function. However, the USBH and USBF modules have control functions for their own since their clock sources are different from those for other modules. For details, refer to section 18, USB Pin Multiplex Controller (USBPM).

Table 13.1 shows the transition conditions for entering the modes from the program execution state, as well as the CPU and peripheral module states in each mode and the procedures for canceling each mode.

On/off of a built-in regulator does not influence the operation of each mode.

Table 13.1 States of Power-Down Modes

Mode	Transition Conditions	State							Canceling Procedure
		CPG	CPU	CPU Register	On-Chip Memory	On-Chip Peripheral Modules	Pins	External Memory	
Sleep mode	Execute SLEEP instruction with STBY bit cleared to 0 in STBCR	Runs	Halts	Held	Halts (Contents held)	Run	Held	Refreshed automatically	1. Interrupt 2. Reset
Software Standby mode	Execute SLEEP instruction with STBY bit set to 1 in STBCR	Halts	Halts	Held	Halt (Contents held)	Halt	*	Self-refreshed	1. NMI, IRQ 2. Reset by $\overline{\text{RESETP}}$
Module standby function	Set MSTP bit to 1 in STBCR	Runs	Runs	Held	Specified module halts (Contents held)	Specified module halts	Held	Refreshed automatically	1. Clear MSTP bit to 0 2. Power-on reset

Notes: * For details, refer to Appendix, A. Pin States.

This state can be cleared by an emulator using the H-UDI command in ASE mode. For details, refer to the manual on emulator.

13.1.2 Reset

A reset is used at power-on or to re-execute from the initial state. This LSI supports two types of reset: power-on reset and manual reset. In power-on reset, any processing to be currently executed is terminated and any events not executed are canceled to execute reset processing immediately. In manual reset, processing required to maintain external memory contents is continued. The following shows the conditions in which power-on reset or manual reset occurs.

- Power-on reset
  A. A low level signal is input to the $\overline{\text{RESETP}}$ pin.
  B. The WDT counter overflows if WDT starts counting while the WT/$\overline{\text{IT}}$ and RSTS bits of the WTCSR are set to 1 and cleared to 0, respectively.
  C. An H-UDI reset occurs. (For details on H-UDI reset, refer to section 26, User Debugging Interface (H-UDI).)
- Manual reset
  The WDT counter overflows if WDT starts counting while the WT/$\overline{\text{IT}}$ and RSTS bits of the WTCSR are set to 1.

13.1.3 Input/Output Pins

Table 13.2 shows the pins for the on/off control of a built-in regulator.

Table 13.2 Pin Configuration

Pin Name	Input/Output	Description
$\overline{\text{RESETP}}$	Input	Power-on Reset Inputting low level signal to this pin cause a transition to power-on reset processing.
TEST_REG	Input	Built-in regulator selection pin. The built-in power supply regulator is used when this pin is high, external power input is used when this pin is low for the internal power (V_{DD} = 1.5 V). The level of this pin should not be changed during operation
REG_PD_MAIN	Input/NC	Built-in regulator control. This pin should be fixed to high when the external power supply is used for the internal power (V_{DD} = 1.5 V). This pin should be open when the built-in power supply regulator is used.
REG_PD_BGR	Input/NC	Built-in regulator control. This pin should be fixed to high when the external power supply is used for the internal power (V_{DD} = 1.5 V). This pin should be open when the built-in power supply regulator is used.
REG_PD_VREF	Input/NC	Built-in regulator control. This pin should be fixed to high when the external power supply is used for the internal power (V_{DD} = 1.5 V). This pin should be open when the built-in power supply regulator is used.

13.2 Register Descriptions

The registers listed below are used in power-down mode. Refer to section 27, List of Registers for the register addresses and register states in each operating mode.

- Standby control register (STBCR)
- Standby control register 2 (STBCR2)
- Standby control register 3 (STBCR3)
- Standby control register 4 (STBCR4)
- Memory clock control register (MCCR)

13.2.1 Standby Control Register (STBCR)

STBCR is an 8-bit readable/writable register that specifies the state of the power-down mode. This register is initialized at power-on reset but retains the previous value after manual reset.

Bit	Bit Name	Initial Value	R/W	Description
7	STBY	0	R/W	Software Standby Specifies transition to software standby mode. 0: Executing SLEEP instruction puts chip into sleep mode 1: Executing SLEEP instruction puts chip into software standby mode
6, 5	—	All 0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
4	STBXTL	0	R/W	Standby Crystal Specifies whether the crystal oscillator stops or continues in standby mode. 0: Stops crystal oscillator in standby mode. 1: Continues crystal oscillator in standby mode.
3	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
2	MSTP2	0	R/W	Module Stop 2 Halts the clock supply to the timer unit TMU when the MSTP2 bit has been set to 1. 0: TMU runs 1: Clock supply to TMU halted
1, 0	—	All 0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.

13.2.2 Standby Control Register 2 (STBCR2)

STBCR2 is a readable/writable 8-bit register that controls the operation of modules in the power-down mode. The STBCR2 register is initialized at power-on reset but retains the previous value after manual reset.

Bit	Bit Name	Initial Value	R/W	Description
7	MSTP10	0	R/W	Module Stop 10 When the MSTP10 bit is set to 1, the supply of the clock to the H-UDI is halted. 0: H-UDI runs 1: Clock supply to H-UDI halted
6	MSTP9	0	R/W	Module Stop 9 When the MSTP9 bit is set to 1, the supply of the clock to the UBC is halted. 0: UBC runs 1: Clock supply to UBC halted
5	MSTP8	0	R/W	Module Stop 8 When the MSTP8 bit is set to 1, the supply of the clock to the DMAC is halted. 0: DMAC runs 1: Clock supply to DMAC halted
4	MSTP7	0	R/W	Module Stop 7 When the MSTP7 bit is set to 1, the supply of the clock to the DSP is halted. 0: DSP runs 1: Clock supply to DSP halted
3	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
2	MSTP5	0	R/W	Module Stop 5 When the MSTP5 bit is set to 1, the supply of the clock to the cache memory is halted. 0: The cache memory runs 1: Clock supply to the cache memory halted

Bit	Bit Name	Initial Value	R/W	Description
1	MSTP4	0	R/W	Module Stop 4 When the MSTP4 bit is set to 1, the supply of the clock to the U memory is halted. 0: The U memory runs 1: Clock supply to the U memory halted
0	MSTP3	0	R/W	Module Stop 3 When the MSTP3 bit is set to 1, the supply of the clock to the X/Y memory is halted. 0: The X/Y memory runs 1: Clock supply to the X/Y memory halted

13.2.3 Standby Control Register 3 (STBCR3)

STBCR3 is an 8-bit readable/writable register that controls the operation of modules in the power-down mode. STBCR3 is initialized at power-on reset but retains the previous value after manual reset.

Bit	Bit Name	Initial Value	R/W	Description
7	MSTP37	0	R/W	Module Stop 37 When the MSTP37 bit is set to 1, the supply of the clock to the SIOF is halted. 0: SIOF runs 1: Clock supply to SIOF halted
6 to 0	—	All 0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

13.2.4 Standby Control Register 4 (STBCR4)

STBCR4 is an 8-bit readable/writable register that controls the operation of modules in the power-down mode. STBCR4 is initialized at power-on reset but retains the previous value after manual reset.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
6	MSTP46	0	R/W	Module Stop 46 When the MSTP46 bit is set to 1, the supply of the clock to the USB is halted. 0: USB runs 1: Clock supply to USB halted Note: If this bit is set to 1, it can only be cleared by a power-on reset. Even if 0 is written to this bit, it cannot be cleared to 0.
5, 4	—	All 0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
3	MSTP43	0	R/W	Module Stop 43 When the MSTP43 bit is set to 1, the supply of the clock to the BOOT is halted. 0: BOOT runs 1: Clock supply to BOOT halted Note: When this bit is changed from 1 to 0 (when the supply to the BOOT was in a halted state and then changed to a run state), it takes time until access to the BOOT module is possible. When this bit is changed from 1 to 0, accessing the BOOT module should be made after the STBCR4 register is read to secure a time for warm up.
2	MSTP42	0	R/W	Module Stop 42 When the MSTP42 bit is set to 1, the supply of the clock to the DAC is halted. 0: DAC runs 1: Clock supply to DAC halted

Bit	Bit Name	Initial Value	R/W	Description
1	MSTP41	0	R/W	Module Stop 41 When the MSTP41 bit is set to 1, the supply of the clock to the SCIF1 is halted. 0: The SCIF1 runs 1: Clock supply to the SCIF1 halted
0	MSTP40	0	R/W	Module Stop 40 When the MSTP40 bit is set to 1, the supply of the clock to the SCIF0 is halted. 0: The SCIF0 runs 1: Clock supply to the SCIR0 halted

13.2.5 Memory Clock Control Register (MCCR)

MCCR is an 8-bit readable/writable register that controls the operation of the memory clock on/off in sleep mode. MCCR is initialized at power-on reset but retains the previous value after manual reset.

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
2	UMCLKC	0	R/W	U Memory Clock Control Controls the U memory on/off in sleep mode 0: U memory clock off in sleep mode 1: U memory clock on in sleep mode When this bit is set to 1, it is possible to access the U memory with DMAC and USBH in sleep mode. Note: When changing the clock frequency multiplication ratio, clear UMCLKC bit to 0. There is the restriction on the interrupt processing when this function is used. Refer to section 7, U Memory for details.
1	XYMCLKC	0	R/W	X/Y Memory Clock Control Controls the X/Y memory on/off in sleep mode 0: X/Y memory clock off in sleep mode 1: X/Y memory clock on in sleep mode When this bit is set to 1, it is possible to access the X/Y memory with DMAC and USBH in sleep mode. Note: When changing the clock frequency multiplication ratio, clear XYMCLKC bit to 0. There is the restriction on the interrupt processing when this function is used. Refer to section 6, X/Y Memory for details.
0	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.

13.3 Operation

13.3.1 Sleep Mode

1. Transition to Sleep Mode

   Executing the SLEEP instruction when the STBY bit in STBCR is 0 causes a transition from the program execution state to sleep mode. Although the CPU halts immediately after executing the SLEEP instruction, the contents of its internal registers remain unchanged. The on-chip peripheral modules continue to run in sleep mode and the clock continues to be output to the CKIO pin.

2. Canceling Sleep Mode

   Sleep mode is canceled by interrupts (NMI, IRQ, and on-chip peripheral module) or reset. Interrupts are accepted in sleep mode even when the BL bit in the SR register is 1. If necessary, save SPC and SSR to the stack before executing the SLEEP instruction.

   — Canceling with an Interrupt

   When an NMI, IRQ or on-chip peripheral module interrupt occurs, sleep mode is canceled and interrupt exception handling is executed. A code indicating the interrupt source is set in the INTEVT2 registers.

   — Canceling with a Reset

   Sleep mode is canceled by a power-on reset or a manual reset.

13.3.2 Software Standby Mode

1. Transition to Software Standby Mode

   The LSI switches from a program execution state to a software standby mode by executing the SLEEP instruction when the STBY bit in STBCR register is 1. In software standby mode, not only the CPU but also the clock and on-chip peripheral modules halt. The clock output state from the CKIO pin can be selected by combining the CKOEN bit in the FRQCR register and the HIZCNT bit in the CMNCR register.

   The contents of the CPU and cache registers remain unchanged. Some registers of on-chip peripheral modules are, however, initialized. Table 13.3 lists the states of on-chip peripheral module registers in a software standby mode.

Table 13.3 Register States in Software Standby Mode

Module	Registers Initialized	Registers Retaining Data
Interrupt controller (INTC)	—	All registers
Bus state controller (BSC)	—	All registers
DMAC	—	All registers
Clock pulse generator (CPG)	—	All registers
Timer Unit (TMU)	TSTR register	Except TSTR register
SIOF, SCIF0, SCIF1	—	All registers
USBPM, USBH, USBF	—	All registers
D/A converter (DAC)	—	All registers
I/O port	—	All registers
User break controller (UBC)	—	All registers
H-UDI	—	All registers

The procedure for switching to software standby mode is as follows:

1. Clear the TME bit in the WDT's timer control register (WTCSR) to 0 to stop the WDT.
2. Set the WDT's timer counter (WTCNT) to 0 and the CKS2 to CKS0 bits in the WTCSR register to appropriate values to secure the specified oscillation settling time.
3. After the STBY bit in the STBCR register is set to 1, a SLEEP instruction is executed.

2. Canceling Software Standby Mode

Software standby mode is canceled by interrupts (NMI, IRQ) or a reset.

— Canceling with an Interrupt

The on-chip WDT can be used for hot starts. When the chip detects an NMI or IRQ interrupt, the clock will be supplied to the entire chip and software standby mode canceled after the time set in the WDT's timer control/status register has elapsed. Interrupt handling then begins and a code indicating the interrupt source is set in the INTEVT2 registers. After the branch to the interrupt handling routine, clear the STBY bit in the STBCR register. WTD stops automatically. If the STBY bit is not cleared, WTD continues operation and a transition is made to software standby mode* when the WTCNT reaches H'80. Interrupts are accepted in software standby mode even when the BL bit in the SR register is 1. If necessary, save SPC and SSR to the stack before executing the SLEEP instruction.

Immediately after an interrupt is detected, the phase of the CKIO pin clock output may be unstable, until the software standby mode is canceled when the CKOEN bit in the FRQCR register and the HIZCNT bit in the CMNCR register are set to 1.

Note: * This standby mode can be canceled by the reset from the $\overline{\text{RESETP}}$ pin.

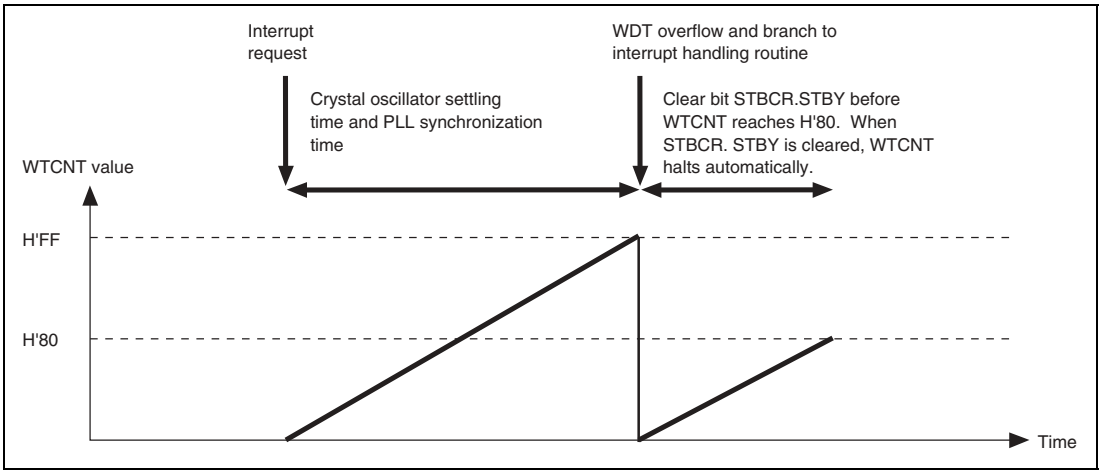


Figure 13.1 Canceling Standby Mode with STBCR.STBY

— Canceling with a Reset

Software standby mode is canceled by a reset using the $\overline{\text{RESETP}}$ pin. Keep the $\overline{\text{RESETP}}$ pin low until the clock oscillation settles. The internal clock will continue to be output to the CKIO pin except clock operating mode 7.

13.3.3 Module Standby Function

1. Transition to Module Standby Function

   Setting the standby control register MSTP bits to 1 halts the supply of clocks to the corresponding on-chip peripheral modules. This function can be used to reduce the power consumption in normal mode as well as sleep mode. The module should be disabled before a transition to the module standby function is made.

   In module standby state, the functions of the external pins of the on-chip peripheral modules change depending on the on-chip peripheral module and port settings. Almost all of the register states are the same as those in standby mode. For details, refer to table 13.3.

2. Canceling Module Standby Function

   The module standby function can be canceled by clearing the MSTP bits to 0, or by a power-on reset. To cancel the module standby function by clearing the corresponding MSTP bit to 0, read the MSTP bit to check the MSTP bit was cleared correctly.

13.3.4 State Transitions between Modes

Figure 13.2 shows the transition between modes.

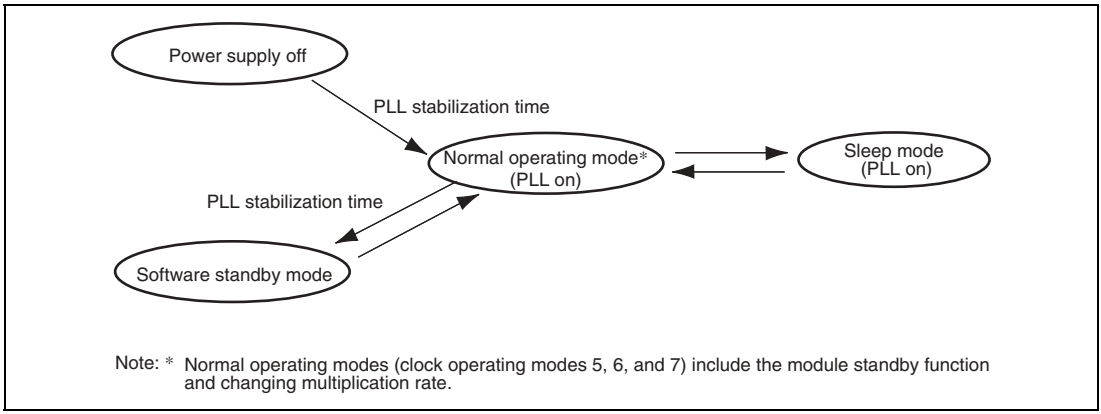


Figure 13.2 Transitions between Modes

13.3.5 Method of turning off the built-in regulator (external V_{DD} used)

The power dissipation of this LSI can be decreased by turning off the built-in regulator if internal power V_{DD} (= 1.5 V) is supplied from the outside.

When TEST_REG pin is low, REG_PD_MAIN pin, REG_PD_BGR pin, and REG_PD_VREF pin are high, the built-in regulator is turned off, and then internal power V_{DD} (= 1.5 V) can be supplied from the outside.

As a result, an internal power loss in a built-in regulator can be decreased.

13.4 Usage note

13.4.1 Reset by the $\overline{\text{RESETP}}$ pin

The RDI_REFCLK_IN pin should be supplied with some clock input although Renesas RF-IC is not used. The $\overline{\text{RESETP}}$ pin has the function to maintain the reset signal internally during specified clock cycles from RDI_REFCLK_IN.

Section 14 Timer Unit (TMU)

This LSI includes a three-channel 32-bit timer unit (TMU).

14.1 Features

- Each channel is provided with an auto-reload 32-bit down counter
- All the channels are provided with 32-bit constant registers and 32-bit down counters for an auto-reload that can be read or written to at any time
- All the channels generate interrupt requests when the 32-bit down counter underflows (H'00000000 → H'FFFFFFFF)
- Allows selection among 4 counter input clocks: Pϕ/4, Pϕ/16, Pϕ/64, and Pϕ/256

Figure 14.1 shows a block diagram of the TMU.

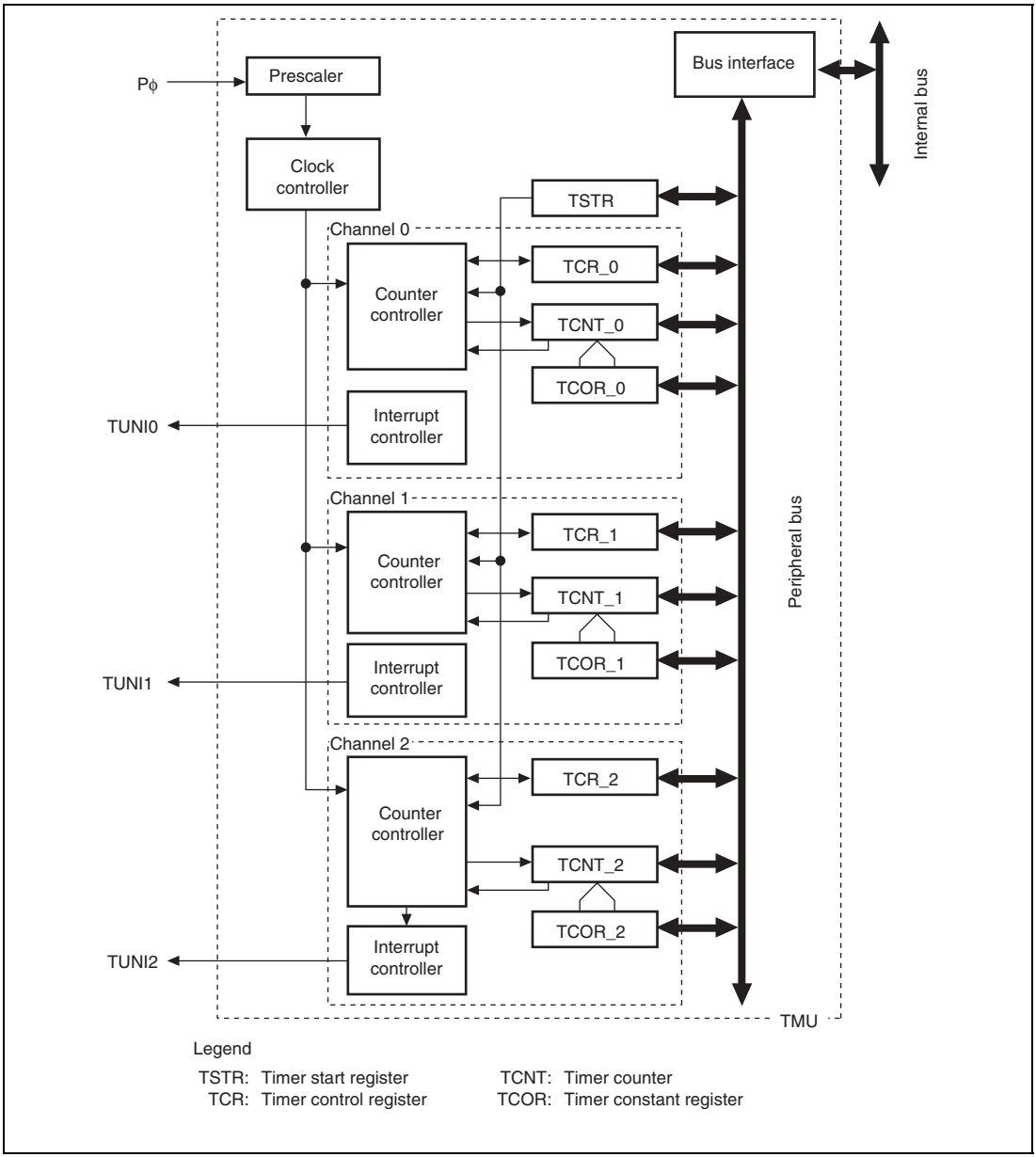


Figure 14.1 TMU Block Diagram

14.2 Register Descriptions

The TMU has the following registers. Refer to section 27, List of Registers, for more details of the addresses of these registers and state of these registers in each processor mode. The TMU register names are expressed in the XXX_N form, with XXX indicating the register name and N indicating the channel number. For example, TCOR_0 represents the TCOR for channel 0.

1. Common
   - Timer start register (TSTR)
2. Channel 0
   - Timer constant register_0 (TCOR_0)
   - Timer counter_0 (TCNT_0)
   - Timer control register_0 (TCR_0)
3. Channel 1
   - Timer constant register_1 (TCOR_1)
   - Timer counter_1 (TCNT_1)
   - Timer control register_1 (TCR_1)
4. Channel 2
   - Timer constant register_2 (TCOR_2)
   - Timer counter_2 (TCNT_2)
   - Timer control register_2 (TCR_2)

14.2.1 Timer Start Register (TSTR)

TSTR is an 8-bit readable/writable register that selects whether to run or halt the timer counters (TCNT).

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
2	STR2	0	R/W	Counter Start 2 Selects whether to run or halt timer counter 2 (TCNT_2). 0: TCNT_2 count halted 1: TCNT_2 counts
1	STR1	0	R/W	Counter Start 1 Selects whether to run or halt timer counter 1 (TCNT_1). 0: TCNT_1 count halted 1: TCNT_1 counts
0	STR0	0	R/W	Counter Start 0 Selects whether to run or halt timer counter 0 (TCNT_0). 0: TCNT_0 count halted 1: TCNT_0 counts

14.2.2 Timer Control Registers (TCR)

The TCR registers are 16-bit readable/writable registers that control the timer counters (TCNT) and interrupts.

The TCR registers control the issuance of interrupts when the flag indicating timer counter (TCNT) underflow has been set to 1, and also carry out counter clock selection.

Bit	Bit Name	Initial Value	R/W	Description
15 to 9	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
8	UNF	0	R/(W)*	Underflow Flag Status flag that indicates occurrence of a TCNT underflow. 0: TCNT has not underflowed [Clearing condition] 0 is written to UNF 1: TCNT has underflowed [Setting condition] TCNT underflows
7, 6	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
5	UNIE	0	R/W	Underflow Interrupt Control Controls enabling of interrupt generation when the status flag (UNF) indicating TCNT underflow has been set to 1. 0: Interrupt due to UNF (TUNI) is disabled 1: Interrupt due to UNF (TUNI) is enabled
4 to 2	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
1	TPSC1	0	R/W	Timer Prescaler 1 and 0
0	TPSC0	0	R/W	Select the TCNT count clock. 00: Count on Pϕ/4 01: Count on Pϕ/16 10: Count on Pϕ/64 11: Count on Pϕ/256

Note: * Only 0 can be written for flag clearing.

14.2.3 Timer Constant Registers (TCOR)

The TCOR registers set the value to be set in TCNT when TCNT underflows.

The TCOR registers are 32-bit readable/writable registers. Their initial value is H'FFFFFFFF.

14.2.4 Timer Counters (TCNT)

TCNT counts down upon input of a clock. The clock input is selected using the TPSC1 and TPSC0 bits in the timer control register (TCR).

When a TCNT count-down results in an underflow (H'00000000 → H'FFFFFFFF), the underflow flag (UNF) in the timer control register (TCR) of the relevant channel is set. The TCOR value is simultaneously set in TCNT itself and the count-down continues from that value.

Initial value of the TCNT is H'FFFFFFFF.

14.3 Operation

Each of the three channels has a 32-bit timer counter (TCNT) and a 32-bit timer constant register (TCOR). The TCNT counts down. The auto-reload function enables synchronized counting.

14.3.1 Counter Operation

When the STR0–STR2 bits in the timer start register (TSTR) are set to 1, the corresponding timer counter (TCNT) starts counting. When a TCNT underflows, the UNF flag of the corresponding timer control register (TCR) is set. At this time, if the UNIE bit in TCR is 1, an interrupt request is sent to the CPU. Also at this time, the value is copied from TCOR to TCNT and the down-count operation is continued.

1. Count Operation Setting Procedure

   An example of the procedure for setting the count operation is shown in figure 14.2.

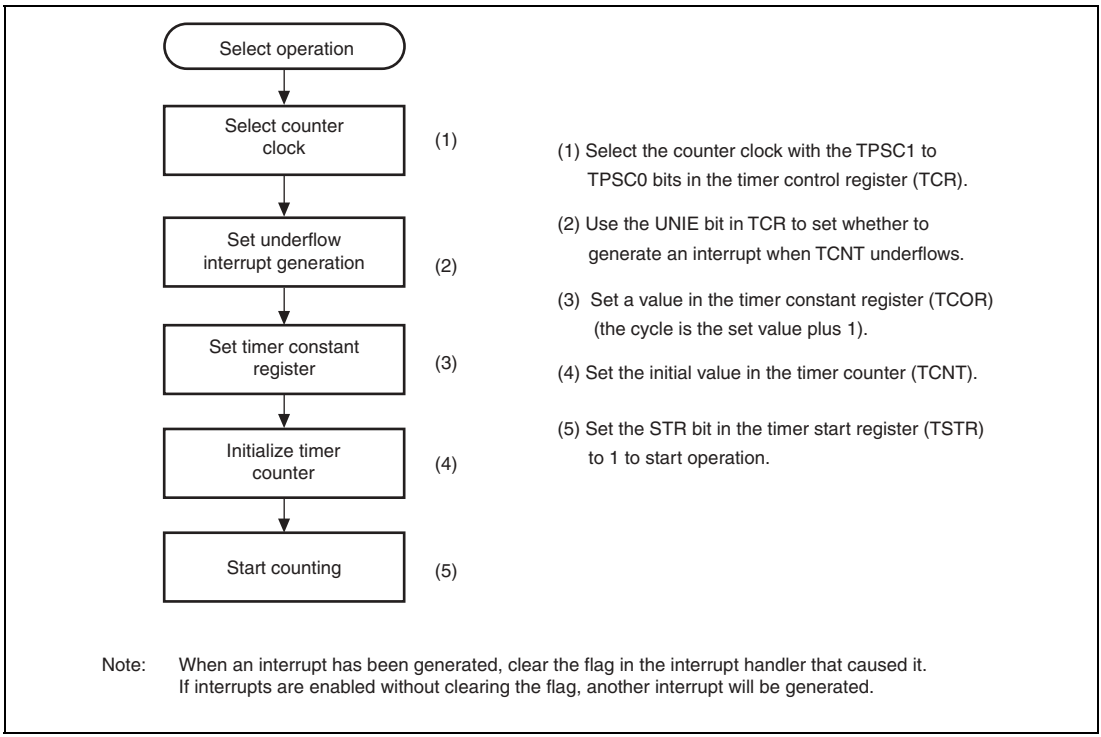


Figure 14.2 Setting Count Operation

2. Auto-Reload Count Operation

Figure 14.3 shows the TCNT auto-reload operation.

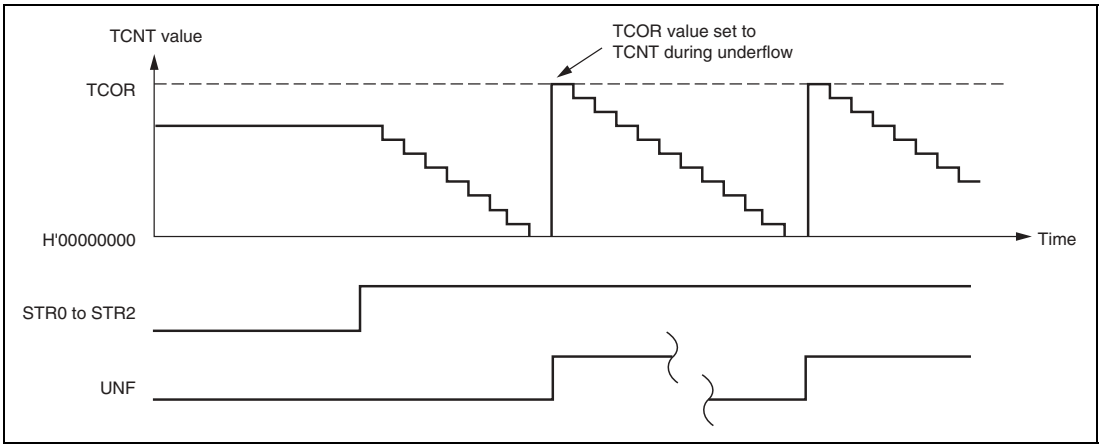


Figure 14.3 Auto-Reload Count Operation

3. TCNT Count Timing

Set the TPSC1 and TPSC0 bits in TCR to select one of the four internal clocks (Pϕ/4, Pϕ/16, Pϕ/64, Pϕ/256) that is created by dividing the peripheral module clock. Figure 14.4 shows the timing.

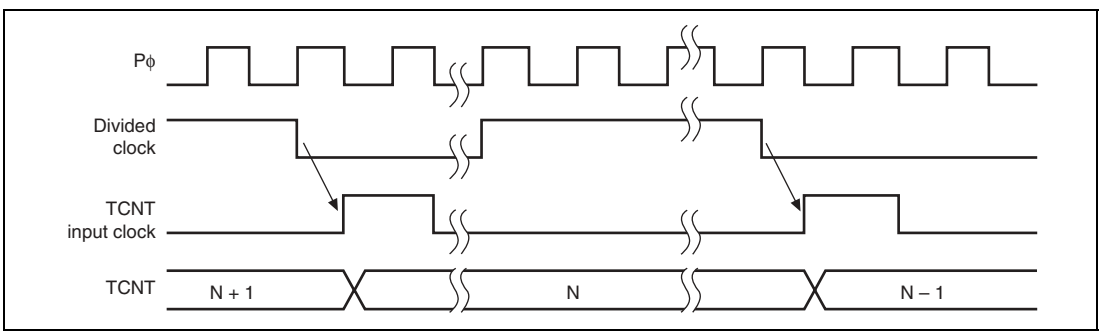


Figure 14.4 Count Timing when Internal Clock Is Operating

14.4 Interrupts

There is one source of TMU interrupts: underflow interrupts (TUNI).

14.4.1 Status Flag Set Timing

The UNF bit is set to 1 when the TCNT underflows. Figure 14.5 shows the timing.

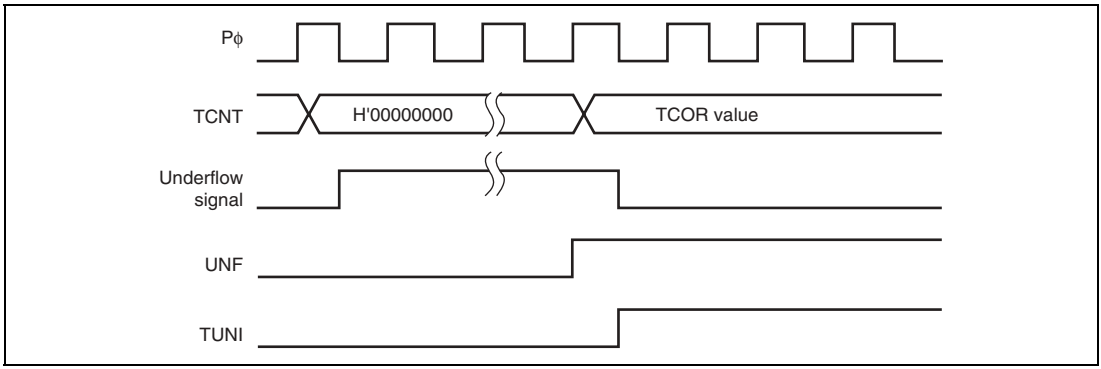


Figure 14.5 UNF Set Timing

14.4.2 Status Flag Clear Timing

The status flag can be cleared by writing 0 from the CPU. Figure 14.6 shows the timing.

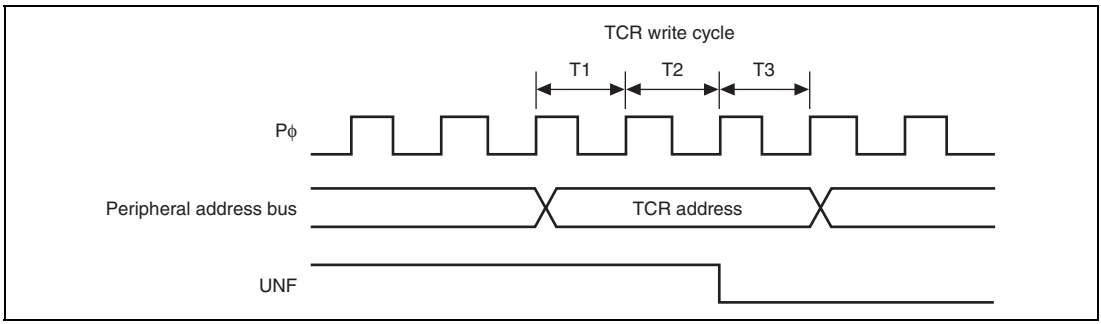


Figure 14.6 Status Flag Clear Timing

14.4.3 Interrupt Sources and Priorities

The TMU produces underflow interrupts for each channel. When the interrupt request flag and interrupt enable bit are both set to 1, the interrupt is requested. Codes are set in the interrupt event register 2 (INTEVT2) for these interrupts and interrupt processing occurs according to the codes.

The relative priorities of channels can be changed using the interrupt controller (see section 4, Exception Handling, and section 8, Interrupt Controller (INTC)). Table 14.1 lists TMU interrupt sources.

Table 14.1 TMU Interrupt Sources

Channel	Interrupt Source	Description	Priority
0	TUNI0	Underflow interrupt 0	High
1	TUNI1	Underflow interrupt 1	↕
2	TUNI2	Underflow interrupt 2	Low

14.5 Usage Notes

Writing to Registers: Synchronization processing is not performed for timer counting during register writes. When writing to registers, always clear the appropriate start bits for the channel (STR2 to STR0) in the timer start register (TSTR) to halt timer counting.

Reading Registers: Synchronization processing is performed for timer counting during register reads. When timer counting and register read processing are performed simultaneously, the register value before TCNT counting down (with synchronization processing) is read.

Section 15 Serial I/O with FIFO (SIOF)

This LSI includes a clock-synchronized serial I/O module with FIFO (SIOF)(Serial Input/Output with FIFO). The SIOF can perform serial communication with a serial peripheral interface bus (SPI).

15.1 Features

- Serial transfer
  16-stage 32-bit FIFOs (transmission is independent)
  Supports 8-bit data/16-bit data/16-bit stereo audio output
  MSB first for data transmission
  Supports a maximum of 48-kHz sampling rate
  Synchronization by either frame synchronization pulse or left/right channel switch
  Supports CODEC control data interface
  Connectable to linear, audio, or A-Law or µ-Law CODEC chip
  Supports both master and slave modes
- Serial clock
  An external pin input or internal clock (Pϕ) can be selected as the clock source.
- Interrupts: One type
- DMA transfer
  Supports DMA transfer by a transfer request for transmission
- SPI mode
  Fixed master mode can perform the full-duplex communication with the SPI slave devices continuously.
  Selects the falling/rising edge of the SCK as data sampling.
  Selects the clock phase of the SCK as a transmit timing.
  Selects three slave devices.
  The length of transmit/receive data is fixed to 8 bits.

Figure 15.1 shows a block diagram of the SIOF.

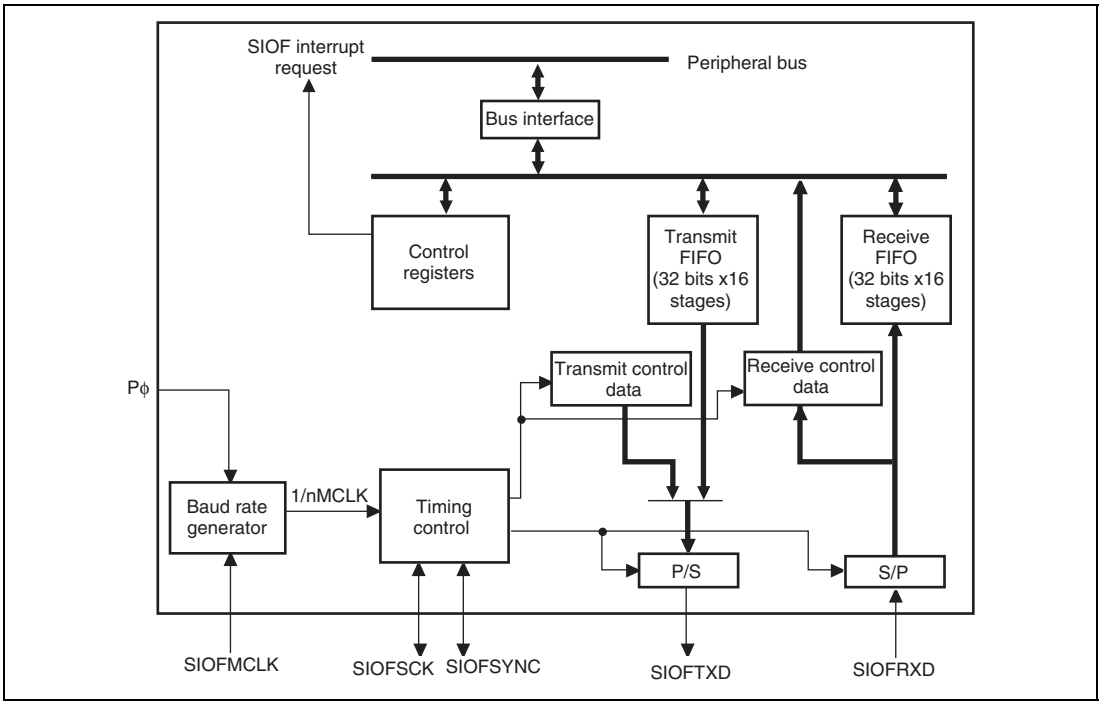


Figure 15.1 Block Diagram of SIOF

15.2 Input/Output Pins

The pin configuration in this module is shown in table 15.1.

Table 15.1 Pin Configuration

Pin Name	Abbreviation*	I/O	Function
SIOF_MCLK	SIOFMCLK	Input	Master clock input
SIOF_SCK (SCK)	SIOFSCK (SCK)	I/O	Serial clock (common to transmission/reception) In SPI mode, fixed to output.
SIOF_SYNC (SIOF_$\overline{SS0}$)	SIOFSYNC ($\overline{SS0}$)	I/O	Frame synchronous signal (common to transmission/reception) In SPI mode, fixed to output, and selects slave device 0.
SIOF_$\overline{SS1}$	($\overline{SS1}$)	Output	In SPI mode, selects slave device 1.
SIOF_$\overline{SS2}$	($\overline{SS2}$)	Output	In SPI mode, selects slave device 2.
SIOF_TXD (MOSI)	SIOFTXD (MOSI)	Output	Transmit data
SIOF_RXD (MISO)	SIOFRXD (MISO)	Input	Receive data

Note: * The pins are collectively called SIOFMCLK, SIOFSCK, SIOFSYNC, SIOFTXD, and SIOFRXD in the following descriptions. In SPI mode, the pins are called SCK, $\overline{SS0}$, $\overline{SS1}$, $\overline{SS2}$, MOSI, and MISO.

15.3 Register Descriptions

The SIOF has the following registers. For the addresses of these registers and the register states in each operating state, refer to section 27, List of Registers.

- Mode register (SIMDR)
- Control register (SICTR)
- Transmit data register (SITDR)
- Receive data register (SIRDR)
- Transmit control data register (SITCR)
- Receive control data register (SIRCR)
- Status register (SISTR)
- Interrupt enable register (SIIER)
- FIFO control register (SIFCTR)
- Clock select register (SISCR)
- Transmit data assign register (SITDAR)
- Receive data assign register (SIRDAR)
- Control data assign register (SICDAR)
- SPI control register (SPICR)

15.3.1 Mode Register (SIMDR)

SIMDR is a 16-bit readable/writable register that sets the SIOF operating mode.

Bit	Bit Name	Initial Value	R/W	Description
15	TRMD1	1	R/W	Transfer Mode
14	TRMD0	0	R/W	Select transfer mode as shown in table 15.2. 00: Slave mode 1 01: Slave mode 2 10: Master mode 1 11: Master mode 2
13	SYNCAT	0	R/W	SIOFSYNC Pin Valid Timing Indicates the position of the SIOFSYNC signal to be output as a synchronization pulse. 0: At the start-bit data of frame 1: At the last-bit data of slot

Bit	Bit Name	Initial Value	R/W	Description
12	REDG	0	R/W	Receive Data Sampling Edge 0: The SIOFRXD signal is sampled at the falling edge of SIOFSCK. (The SIOFTXD signal is output at the rising edge of SIOFSCK). 1: The SIOFRXD signal is sampled at the rising edge of SIOFSCK. (The SIOFTXD signal is output at the falling edge of SIOFSCK). Note: This bit is valid only in master mode.
11	FL3	0	R/W	Frame Length
10	FL2	0	R/W	00xx: Data length is 8 bits and frame length is 8 bits.
9	FL1	0	R/W	0100: Data length is 8 bits and frame length is 16 bits.
8	FL0	0	R/W	0101: Data length is 8 bits and frame length is 32 bits. 0110: Data length is 8 bits and frame length is 64 bits. 0111: Data length is 8 bits and frame length is 128 bits. 10xx: Data length is 16 bits and frame length is 16 bits. 1100: Data length is 16 bits and frame length is 32 bits. 1101: Data length is 16 bits and frame length is 64 bits. 1110: Data length is 16 bits and frame length is 128 bits. 1111: Data length is 16 bits and frame length is 256 bits. Note: When data length is specified as 8 bits, control data cannot be transmitted or received. x: Don't care
7	TXDIZ	0	R/W	SIOFTXD Pin Output when Transmission is Invalid* 0: High output (1 output) when invalid 1: High-impedance state when invalid Note: Invalid means when disabled, and when a slot that is not assigned as transmit data or control data is being transmitted.
6	RCIM	0	R/W	Receive Control Data Interrupt Mode 0: Sets the RCRDY bit in SISTR when the contents of SIRCR change. 1: Sets the RCRDY bit in SISTR each time when the SIRCR receives the control data.

Bit	Bit Name	Initial Value	R/W	Description
5	SYNCAC	0	R/W	SIOFSYNC Pin Polarity Valid when the SIOFSYNC signal is output as synchronous pulse in master mode. 0: Active-high 1: Active-low
4	SYNCDL	0	R/W	Data Pin Bit Delay for SIOFSYNC Pin Valid when the SIOFSYNC signal is output as synchronous pulse. Only one-bit delay is valid for transmission in slave mode. 0: No bit delay 1: 1-bit delay
3 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation cannot be guaranteed.

Table 15.2 shows the operation in each transfer mode.

Table 15.2 Operation in Each Transfer Mode

Transfer Mode	Master/Slave	SIOFSYNC	Bit Delay	Control Data Method*
Slave mode 1	Slave	Synchronous pulse	SYNCDL bit	Slot position
Slave mode 2	Slave	Synchronous pulse		Secondary FS
Master mode 1	Master	Synchronous pulse		Slot position
Master mode 2	Master	L/R	No	Not supported

Note: * The control data method is valid only when the FL bit is specified as 1xxx. (x: Don't care)

15.3.2 Control Register (SICTR)

SICTR is a 16-bit readable/writable register that sets the SIOF operating state.

Bit	Bit Name	Initial Value	R/W	Description
15	SCKE	0	R/W	Serial Clock Output Enable This bit is valid in master mode. 0: Disables the SIOFSCK output (outputs 0) 1: Enables the SIOFSCK output • If this bit is set to 1, the SIOF initializes the baud rate generator and initiates the operation. At the same time, the SIOF outputs the clock generated by the baud rate generator to the SIOFSCK pin. This bit is initialized in module stop mode.
14	FSE	0	R/W	Frame Synchronous Signal Output Enable This bit is valid in master mode. 0: Disables the SIOFSYNC output (outputs 0) 1: Enables the SIOFSYNC output • If this bit is set to 1, the SIOF initializes the frame counter and initiates the operation. This bit is initialized in module stop mode.
13 to 10	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
9	TXE	0	R/W	Transmit Enable 0: Disables data transmission from the SIOFTXD pin 1: Enables data transmission from the SIOFTXD pin • This bit setting becomes valid at the start of the next frame (at the rising edge of the SIOFSYNC signal). • When the 1 setting for this bit becomes valid, the SIOF issues a transmit transfer request according to the setting of the TFWM bit in SIFCTR. When transmit data is stored in the transmit FIFO, transmission of data from the SIOFTXD pin begins. • This bit is initialized upon a transmit reset. This bit is initialized in module stop mode.
8	RXE	0	R/W	Receive Enable 0: Disables data reception from SIOFRXD 1: Enables data reception from SIOFRXD • This bit setting becomes valid at the start of the next frame (at the rising edge of the SIOFSYNC signal). • When the 1 setting for this bit becomes valid, the SIOF begins the reception of data from the SIOFRXD pin. When receive data is stored in the receive FIFO, the SIOF issues a reception transfer request according to the setting of the RFWM bit in SIFCTR. • This bit is initialized upon receive reset. This bit is initialized in module stop mode.
7 to 2	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
1	TXRST	0	R/W	Transmit Reset 0: Does not reset transmit operation 1: Resets transmit operation • This bit setting becomes valid immediately. This bit should be cleared to 0 before setting the register to be initialized. • When the 1 setting for this bit becomes valid, the SIOF immediately sets transmit data from the SIOFTXD pin to 1, and initializes the transmit data register and transmit-related status. The following are initialized. — SITDR — SITCR — Transmit FIFO write pointer — TCRDY, TFEMP, and TDREQ bits in SISTR This bit is automatically cleared by SIOF after completes a reset, to be always read as 0.
0	RXRST	0	R/W	Receive Reset 0: Does not reset receive operation 1: Resets receive operation • This bit setting becomes valid immediately. This bit should be cleared to 0 before setting the register to be initialized. • When the 1 setting for this bit becomes valid, the SIOF immediately disables reception from the SIOFRXD pin, and initializes the receive data register and receive-related status. The following are initialized. — SIRDR — SIRCR — Receive FIFO read pointer — RCRDY, RFFUL, and RDREQ bits in SISTR This bit is automatically cleared by SIOF after completes a reset, to be always read as 0.

15.3.3 Transmit Data Register (SITDR)

SITDR is a 32-bit write-only register that specifies the SIOF transmit data.

SITDR is initialized by the conditions specified in section 27, List of Registers, or by a transmit reset caused by the TXRST bit in SICTR.

SITDR is initialized in module stop mode.

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	SITDL15 to SITDL0	All 0	W	Left-Channel Transmit Data Specify data to be output from the SIOFTXD pin as left-channel data. The position of the left-channel data in the transmit frame is specified by the TDLA bit in SITDAR. • These bits are valid only when the TDLE bit in SITDAR is set to 1.
15 to 0	SITDR15 to SITDR0	All 0	W	Right-Channel Transmit Data Specify data to be output from the SIOFTXD pin as right-channel data. The position of the right-channel data in the transmit frame is specified by the TDRA bit in SITDAR. • These bits are valid only when the TDRE bit and TLREP bit in SITDAR are set to 1 and cleared to 0, respectively.

15.3.4 Receive Data Register (SIRDR)

SIRDR is a 32-bit read-only register that reads receive data of the SIOF. SIRDR stores data in the receive FIFO and is initialized by the conditions specified in section 27, List of Registers, or by a receive reset caused by the RXRST bit in SICTR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	SIRDL15 to SIRDL 0	All 0	R	Left-Channel Receive Data Store data received from the SIOFRXD pin as left-channel data. The position of the left-channel data in the receive frame is specified by the RDLA bit in SIRDAR. • These bits are valid only when the RDLE bit in SIRDAR is set to 1.
15 to 0	SIRDR15 to SIRDR 0	All 0	R	Right-Channel Receive Data Store data received from the SIOFRXD pin as right-channel data. The position of the right-channel data in the receive frame is specified by the RDRA bit in SIRDAR. • These bits are valid only when the RDRE bit in SIRDAR is set to 1.

15.3.5 Transmit Control Data Register (SITCR)

SITCR is a 32-bit readable/writable register that specifies transmit control data of the SIOF. SITCR can be specified only when the FL bit in SIMDR is specified as 1xxx (x: Don't care).

SITCR is initialized by the conditions specified in section 27, List of Registers, or by a transmit reset caused by the TXRST bit in SICTR.

SITCR is initialized in module stop mode.

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	SITC015 to SITC00	All 0	R/W	Control Channel 0 Transmit Data Specify data to be output from the SIOFTXD pin as control channel 0 transmit data. The position of the control channel 0 data in the transmit or receive frame is specified by the CD0A bit in SICDAR. • These bits are valid only when the CD0E bit in SICDAR is set to 1.
15 to 0	SITC115 to SITC10	All 0	R/W	Control Channel 1 Transmit Data Specify data to be output from the SIOFTXD pin as control channel 1 transmit data. The position of the control channel 1 data in the transmit or receive frame is specified by the CD1A bit in SICDAR. • These bits are valid only when the CD1E bit in SICDAR is set to 1.

15.3.6 Receive Control Data Register (SIRCR)

SIRCR is a 32-bit readable/writable register that stores receive control data of the SIOF. SIRCR can be specified only when the FL bit in SIMDR is specified as 1xxx (x: Don't care).

SIRCR is initialized by the conditions specified in section 27, List of Registers, or by a receive reset caused by the RXRST bit in SICTR.

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	SIRC015 to SIRC00	All 0	R/W	Control Channel 0 Receive Data Store data received from the SIOFRXD pin as control channel 0 receive data. The position of the control channel 0 data in the transmit or receive frame is specified by the CD0A bit in SICDAR. • These bits are valid only when the CD0E bit in SICDAR is set to 1.
15 to 0	SIRC115 to SIRC10	All 0	R/W	Control Channel 1 Receive Data Store data received from the SIOFRXD pin as control channel 1 receive data. The position of the control channel 1 data in the transmit or receive frame is specified by the CD1A bit in SICDAR. • These bits are valid only when the CD1E bit in SICDAR is set to 1.

15.3.7 Status Register (SISTR)

SISTR is a 16-bit read-only register that shows the SIOF state. Each bit in this register becomes an SIOF interrupt source when the corresponding bit in SIIER is set to 1.

SISTR is initialized in module stop mode.

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to this bit, the operation is not guaranteed.
14	TCRDY	0	R	Transmit Control Data Ready 0: Indicates that a write to SITCR is disabled 1: Indicates that a write to SITCR is enabled • If SITCR is written when this bit is cleared to 0, SITCR is over-written and the previous contents of SITCR are not output from the SIOFTXD pin. • This bit is valid when the TXE bit in SITCR is set to 1. • This bit indicates a state of the SIOF. If SITCR is written, the SIOF clears this bit. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.
13	TFEMP	0	R	Transmit FIFO Empty 0: Indicates that transmit FIFO is not empty 1: Indicates that transmit FIFO is empty • This bit is valid when the TXE bit in SICTR is 1. • This bit indicates a state; if SITDR is written, the SIOF clears this bit. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.

Bit	Bit Name	Initial Value	R/W	Description
12	TDREQ	0	R	Transmit Data Transfer Request 0: Indicates that the size of empty space in the transmit FIFO does not exceed the size specified by the TFWM bit in SIFCTR. 1: Indicates that the size of empty space in the transmit FIFO exceeds the size specified by the TFWM bit in SIFCTR. A transmit data transfer request is issued when the empty space in the transmit FIFO exceeds the size specified by the TFWM bit in SIFCTR. When using transmit data transfer through the DMAC, this bit is always cleared by one DMAC access. After DMAC access, when conditions for setting this bit are satisfied, the SIOF again indicates 1 for this bit. • This bit is valid when the TXE bit in SICTR is 1. • This bit indicates a state; if the size of empty space in the transmit FIFO is less than the size specified by the TFWM bit in SIFCTR, the SIOF clears this bit. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.
11	—	0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to this bit, the operation is not guaranteed.
10	RCRDY	0	R	Receive Control Data Ready 0: Indicates that the SIRCR stores no valid data. 1: Indicates that the SIRCR stores valid data. • If SIRCR is written when this bit is set to 1, SIRCR is modified by the latest data. • This bit is valid when the RXE bit in SICTR is set to 1. • This bit indicates a state of the SIOF. If SIRCR is read, the SIOF clears this bit. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.

Bit	Bit Name	Initial Value	R/W	Description
9	RFFUL	0	R	Receive FIFO Full 0: Receive FIFO not full 1: Receive FIFO full • This bit is valid when the RXE bit in SICTR is 1. • This bit indicates a state; if SIRDR is read, the SIOF clears this bit. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.
8	RDREQ	0	R	Receive Data Transfer Request 0: Indicates that the size of valid space in the receive FIFO does not exceed the size specified by the RFWM bit in SIFCTR. 1: Indicates that the size of valid space in the receive FIFO exceeds the size specified by the RFWM bit in SIFCTR. A receive data transfer request is issued when the valid space in the receive FIFO exceeds the size specified by the RFWM bit in SIFCTR. When using receive data transfer through the DMAC, this bit is always cleared by one DMAC access. After DMAC access, when conditions for setting this bit are satisfied, the SIOF again indicates 1 for this bit. • This bit is valid when the RXE bit in SICTR is 1. • This bit indicates a state; if the size of valid space in the receive FIFO is less than the size specified by the RFWM bit in SIFCTR, the SIOF clears this bit. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.
7, 6	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
5	SAERR	0	R/W	Slot Assign Error 0: Indicates that no slot assign error occurs 1: Indicates that a slot assign error occurs A slot assign error occurs when the specifications in SITDAR, SIRDAR, and SICDAR overlap. If a slot assign error occurs, the SIOF does not transmit data to the SIOFTXD pin and does not receive data from the SIOFRXD pin. Note that the SIOF does not clear the TXE bit or RXE bit in SICTR at a slot assign error. • This bit is valid when the TXE bit or RXE bit in SICTR is 1. • When 1 is written to this bit, the contents are cleared. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.
4	FSERR	0	R/W	Frame Synchronization Error 0: Indicates that no frame synchronization error occurs 1: Indicates that a frame synchronization error occurs A frame synchronization error occurs when the next frame synchronization timing appears before the previous data or control data transfers have been completed. If a frame synchronization error occurs, the SIOF performs transmission or reception for slots that can be transferred. • This bit is valid when the TXE or RXE bit in SICTR is 1. • When 1 is written to this bit, the contents are cleared. Writing 0 to this bit is invalid. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.

Bit	Bit Name	Initial Value	R/W	Description
3	TFOVF	0	R/W	Transmit FIFO Overflow 0: No transmit FIFO overflow 1: Transmit FIFO overflow A transmit FIFO overflow means that there has been an attempt to write to SITDR when the transmit FIFO is full. When a transmit FIFO overflow occurs, the SIOF indicates overflow, and writing is invalid. • This bit is valid when the TXE bit in SICTR is 1. • When 1 is written to this bit, the contents are cleared. Writing 0 to this bit is invalid. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.
2	TFUDF	0	R/W	Transmit FIFO Underflow 0: No transmit FIFO underflow 1: Transmit FIFO underflow A transmit FIFO underflow means that loading for transmission has occurred when the transmit FIFO is empty. When a transmit FIFO underflow occurs, the SIOF repeatedly sends the previous transmit data. • This bit is valid when the TXE bit in SICTR is 1. • When 1 is written to this bit, the contents are cleared. Writing 0 to this bit is invalid. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.
1	RFUDF	0	R/W	Receive FIFO Underflow 0: No receive FIFO underflow 1: Receive FIFO underflow A receive FIFO underflow means that reading of SIRDR has occurred when the receive FIFO is empty. When a receive FIFO underflow occurs, the value of data read from SIRDR is not guaranteed. • This bit is valid when the RXE bit in SICTR is 1. • When 1 is written to this bit, the contents are cleared. Writing 0 to this bit is invalid. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.

Bit	Bit Name	Initial Value	R/W	Description
0	RFOVF	0	R/W	Receive FIFO Overflow 0: No receive FIFO overflow 1: Receive FIFO overflow A receive FIFO overflow means that writing has occurred when the receive FIFO is full. When a receive FIFO overflow occurs, the SIOF indicates overflow, and receive data is lost. • This bit is valid when the RXE bit in SICTR is 1. • When 1 is written to this bit, the contents are cleared. Writing 0 to this bit is invalid. • If the issue of interrupts by this bit is enabled, an SIOF interrupt is issued.

15.3.8 Interrupt Enable Register (SIIER)

SIIER is a 16-bit readable/writable register that enables the issue of SIOF interrupts. When each bit in this register is set to 1 and the corresponding bit in SISTR is set to 1, the SIOF issues an interrupt.

Bit	Bit Name	Initial Value	R/W	Description
15	TDMAE	0	R/W	Transmit Data DMA Transfer Request Enable Transmits an interrupt as an interrupt to the CPU/DMA transfer request. The TDREQE bit can be set as transmit interrupts. 0: Used as a CPU interrupt 1: Used as a DMA transfer request to the DMAC
14	TCRDYE	0	R/W	Transmit Control Data Ready Enable 0: Disables interrupts due to transmit control data ready 1: Enables interrupts due to transmit control data ready
13	TFEMPE	0	R/W	Transmit FIFO Empty Enable 0: Disables interrupts due to transmit FIFO empty 1: Enables interrupts due to transmit FIFO empty

Bit	Bit Name	Initial Value	R/W	Description
12	TDREQE	0	R/W	Transmit Data Transfer Request Enable 0: Disables interrupts due to transmit data transfer requests 1: Enables interrupts due to transmit data transfer requests
11	RDMAE	0	R/W	Receive Data DMA Transfer Request Enable Transmits an interrupt as an interrupt to the CPU/DMA transfer request. The RDREQE bit can be set as receive interrupts. 0: Used as a CPU interrupt 1: Used as a DMA transfer request to the DMAC
10	RCRDYE	0	R/W	Receive Control Data Ready Enable 0: Disables interrupts due to receive control data ready 1: Enables interrupts due to receive control data ready
9	RFFULE	0	R/W	Receive FIFO Full Enable 0: Disables interrupts due to receive FIFO full 1: Enables interrupts due to receive FIFO full
8	RDREQE	0	R/W	Receive Data Transfer Request Enable 0: Disables interrupts due to receive data transfer requests 1: Enables interrupts due to receive data transfer requests
7, 6	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.
5	SAERRE	0	R/W	Slot Assign Error Enable 0: Disables interrupts due to slot assign error 1: Enables interrupts due to slot assign error
4	FSERRE	0	R/W	Frame Synchronization Error Enable 0: Disables interrupts due to frame synchronization error 1: Enables interrupts due to frame synchronization error
3	TFOVFE	0	R/W	Transmit FIFO Overflow Enable 0: Disables interrupts due to transmit FIFO overflow 1: Enables interrupts due to transmit FIFO overflow

Bit	Bit Name	Initial Value	R/W	Description
2	TFUDFE	0	R/W	Transmit FIFO Underflow Enable 0: Disables interrupts due to transmit FIFO underflow 1: Enables interrupts due to transmit FIFO underflow
1	RFUDFE	0	R/W	Receive FIFO Underflow Enable 0: Disables interrupts due to receive FIFO underflow 1: Enables interrupts due to receive FIFO underflow
0	RFOVFE	0	R/W	Receive FIFO Overflow Enable 0: Disables interrupts due to receive FIFO overflow 1: Enables interrupts due to receive FIFO overflow

15.3.9 FIFO Control Register (SIFCTR)

SIFCTR is a 16-bit readable/writable register that indicates the area available for the transmit/receive FIFO transfer.

Bit	Bit Name	Initial Value	R/W	Description
15	TFWM2	0	R/W	Transmit FIFO Watermark
14	TFWM1	0	R/W	000: Issue a transfer request when 16 stages of the transmit FIFO are empty.
13	TFWM0	0	R/W	001: Setting prohibited 010: Setting prohibited 011: Setting prohibited 100: Issue a transfer request when 12 or more stages of the transmit FIFO are empty. 101: Issue a transfer request when 8 or more stages of the transmit FIFO are empty. 110: Issue a transfer request when 4 or more stages of the transmit FIFO are empty. 111: Issue a transfer request when 1 or more stages of transmit FIFO are empty. • A transfer request to the transmit FIFO is issued by the TDREQE bit in SISTR. • The transmit FIFO is always used as 16 stages of the FIFO regardless of these bit settings.

Bit	Bit Name	Initial Value	R/W	Description
12	TFUA4	1	R	Transmit FIFO Usable Area
11	TFUA3	0	R	Indicate the number of words that can be transferred by the CPU or DMAC as B'00000 (full) to B'10000 (empty).
10	TFUA2	0	R	
9	TFUA1	0	R	
8	TFUA0	0	R	
7	RFWM2	0	R/W	Receive FIFO Watermark
6	RFWM1	0	R/W	000: Issue a transfer request when 1 stage or more of the receive FIFO are valid.
5	RFWM0	0	R/W	001: Setting prohibited 010: Setting prohibited 011: Setting prohibited 100: Issue a transfer request when 4 or more stages of the receive FIFO are valid. 101: Issue a transfer request when 8 or more stages of the receive FIFO are valid. 110: Issue a transfer request when 12 or more stages of the receive FIFO are valid. 111: Issue a transfer request when 16 stages of the receive FIFO are valid. • A transfer request to the receive FIFO is issued by the RDREQE bit in SISTR. • The receive FIFO is always used as 16 stages of the FIFO regardless of these bit settings.
4	RFUA4	0	R	Receive FIFO Usable Area
3	RFUA3	0	R	Indicate the number of words that can be transferred by the CPU or DMAC as B'00000 (empty) to B'10000 (full).
2	RFUA2	0	R	
1	RFUA1	0	R	
0	RFUA0	0	R	

15.3.10 Clock Select Register (SISCR)

SISCR is a 16-bit readable/writable register that sets the serial clock generation conditions for the master clock. SISCR can be specified when the TRMD1 and TRMD0 bits in SIMDR are specified as B'10 or B'11.

Bit	Bit Name	Initial Value	R/W	Description
15	MSSEL	1	R/W	Master Clock Source Selection 0: Uses the input signal of the SIOFMCLK pin as the master clock 1: Uses Pϕ as the master clock The master clock is the clock input to the baud rate generator.
14	MSIMM	1	R/W	Master Clock Direct Selection 0: Uses the output clock of the baud rate generator as the serial clock 1: Uses the master clock itself as the serial clock
13	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, the operation is not guaranteed.
12	BRPS4	0	R/W	Prescaler Setting Set the master clock division ratio according to the count value of the prescaler of the baud rate generator. The range of settings is from B'00000 (× 1/1) to B'11111 (× 1/32).
11	BRPS3	0	R/W	
10	BRPS2	0	R/W	
9	BRPS1	0	R/W	
8	BRPS0	0	R/W	
7 to 3	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
2	BRDV2	0	R/W	Baud Rate Generator's Division Ratio Setting
1	BRDV1	0	R/W	Set the frequency division ratio for the output stage of the baud rate generator.
0	BRDV0	0	R/W	000: Prescaler output × 1/2
				001: Prescaler output × 1/4
				010: Prescaler output × 1/8
				011: Prescaler output × 1/16
				100: Prescaler output × 1/32
				101: Setting prohibited
				110: Setting prohibited
				111: Prescaler output × 1/1*
				The final frequency division ratio of the baud rate generator is determined by BRPS × BRDV (maximum 1/1024).
				Note: * This setting is valid only when the BRPS4 to BRPS0 bits are set to B'00000.

15.3.11 Transmit Data Assign Register (SITDAR)

SITDAR is a 16-bit readable/writable register that specifies the position of the transmit data in a frame (slot number).

Bit	Bit Name	Initial Value	R/W	Description
15	TDLE	0	R/W	Transmit Left-Channel Data Enable
				0: Disables left-channel data transmission
				1: Enables left-channel data transmission
14 to 12	—	All 0	R	Reserved
				These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.
11	TDLA3	0	R/W	Transmit Left-Channel Data Assigns 3 to 0
10	TDLA2	0	R/W	Specify the position of left-channel data in a transmit frame as B'0000 (0) to B'1110 (14).
9	TDLA1	0	R/W	1111: Setting prohibited
8	TDLA0	0	R/W	• Transmit data for the left channel is specified in the SITDL bit in SITDR.

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	0	R/W	Transmit Right-Channel Data Enable 0: Disables right-channel data transmission 1: Enables right-channel data transmission
6	TLREP	0	R/W	Transmit Left-Channel Repeat 0: Transmits data specified in the SITDR bit in SITDR as right-channel data 1: Repeatedly transmits data specified in the SITDL bit in SITDR as right-channel data • This bit setting is valid when the TDRE bit is set to 1. • When this bit is set to 1, the SITDR bit settings are ignored.
5, 4	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.
3	TDRA3	0	R/W	Transmit Right-Channel Data Assign Specify the position of right-channel data in a transmit frame as B'0000 (0) to B'1110 (14). 1111: Setting prohibited • Transmit data for the right channel is specified in the SITDR bit in SITDR.
2	TDRA2	0	R/W	
1	TDRA1	0	R/W	
0	TDRA0	0	R/W	

15.3.12 Receive Data Assign Register (SIRDAR)

SIRDAR is a 16-bit readable/writable register that specifies the position of the receive data in a frame (slot number).

Bit	Bit Name	Initial Value	R/W	Description
15	RDLE	0	R/W	Receive Left-Channel Data Enable 0: Disables left-channel data reception 1: Enables left-channel data reception
14 to 12	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
11	RDLA3	0	R/W	Receive Left-Channel Data Assign
10	RDLA2	0	R/W	Specify the position of left-channel data in a receive frame as B'0000 (0) to B'1110 (14).
9	RDLA1	0	R/W	
8	RDLA0	0	R/W	1111: Setting prohibited • Receive data for the left channel is stored in the SIRDL bit in SIRDR.
7	RDRE	0	R/W	Receive Right-Channel Data Enable 0: Disables right-channel data reception 1: Enables right-channel data reception
6 to 4	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.
3	RDRA3	0	R/W	Receive Right-Channel Data Assign
2	RDRA2	0	R/W	Specify the position of right-channel data in a receive frame as B'0000 (0) to B'1110 (14).
1	RDRA1	0	R/W	
0	RDRA0	0	R/W	1111: Setting prohibited • Receive data for the right channel is stored in the SIRDR bit in SIRDR.

15.3.13 Control Data Assign Register (SICDAR)

SICDAR is a 16-bit readable/writable register that specifies the position of the control data in a frame (slot number). SICDAR can be specified only when the FL bit in SIMDR is specified as 1xxx (x: Don't care).

Bit	Bit Name	Initial Value	R/W	Description
15	CD0E	0	R/W	Control Channel 0 Data Enable 0: Disables transmission and reception of control channel 0 data 1: Enables transmission and reception of control channel 0 data
14 to 12	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
11	CD0A3	0	R/W	Control Channel 0 Data Assign
10	CD0A2	0	R/W	Specify the position of control channel 0 data in a receive or transmit frame as B'0000 (0) to B'1110 (14).
9	CD0A1	0	R/W	
8	CD0A0	0	R/W	1111: Setting prohibited • Transmit data for the control channel 0 data is specified in the SITD0 bit in SITCR. • Receive data for the control channel 0 data is stored in the SIRD0 bit in SIRCR.
7	CD1E	0	R/W	Control Channel 1 Data Enable 0: Disables transmission and reception of control channel 1 data 1: Enables transmission and reception of control channel 1 data
6 to 4	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.
3	CD1A3	0	R/W	Control Channel 1 Data Assign
2	CD1A2	0	R/W	Specify the position of control channel 1 data in a receive or transmit frame as B'0000 (0) to B'1110 (14).
1	CD1A1	0	R/W	
0	CD1A0	0	R/W	1111: Setting prohibited • Transmit data for the control channel 1 data is specified in the SITD1 bit in SITCR. • Receive data for the control channel 1 data is stored in the SIRD1 bit in SIRCR.

15.3.14 SPI Control Register (SPICR)

SPICR is a 16-bit readable/writable register that specifies the operating mode of the SPI.

Bit	Bit Name	Initial Value	R/W	Description
15	SPIM	0	R/W	SPI Mode Selects the SIOF operating mode. 0: Operates as the SIOF. 1: The SIOF operates in master mode of the SPI.
14	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, the operation is not guaranteed.
13	CPHA	0	R/W	SPI Clock Phase Selects the SPI clock phase. 0: Samples data at the first edge of the SCK. 1: Samples data at the second edge of the SCK.
12	CPOL	0	R/W	SPI Clock Polarity Selects the SPI clock polarity. 0: The SCK is high-active, and goes low in the idle state. 1: The SCK is low-active, and goes high in the idle state.
11	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, the operation is not guaranteed.
10	SS2E*	0	R/W	Slave Device 2 ($\overline{SS2}$) Enable 0: Not select slave device 2. 1: Selects slave device 2.
9	SS1E*	0	R/W	Slave Device 1 ($\overline{SS1}$) Enable 0: Not select slave device 1. 1: Selects slave device 1.
8	SS0E*	0	R/W	Slave Device 0 ($\overline{SS0}$) Enable 0: Not select slave device 0. 1: Selects slave device 0.
7, 6	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, the operation is not guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
5	SSAST1	0	R/W	Setting of SS Assert
4	SSAST0	0	R/W	Set the setup timing of the SS for the SCK.

- CPHA = 0 (Unit: SCK clock)

SSAST[1:0]	SS Setup	SS Hold
00	0.5 clock	0 clock
01	1 clock	0.5 clock
10	1.5 clock	1 clock
11	2 clock	1.5 clock

- CPHA = 1 (Unit: SCK clock)

SSAST[1:0]	SS Setup	SS Hold
00	0 clock	0.5 clock
01	0.5 clock	1 clock
10	1 clock	1.5 clock
11	1.5 clock	2 clock

3, 2	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
1	FLD1	0	R/W	Frame Delay
0	FLD0	0	R/W	Specify the minimum time of the idle state between frames in terms of the clock number of the SCK. 00: No delay. The continuous communication of the SPI is performed when the $\overline{SS}$ pin asserts low continuously. 01: Delay for 1 clock of the SCK. 10: Delay for 2 clocks of the SCK. 11: Delay for 3 clocks of the SCK.

Note: * The SS0E, SS1E, and SS2E bits must not be simultaneously set to 1.

15.4 Operation

15.4.1 Serial Clocks

Master/Slave Modes: The following 2 modes are available as the SIOF clock mode.

- Slave mode: SIOFSCK, SIOFSYNC input
- Master mode: SIOFSCK, SIOFSYNC output

Baud Rate Generator: In SIOF master mode, the baud rate generator (BRG) is used to generate the serial clock. The division ratio is from 1/1 to 1/1024.

Figure 15.2 shows connections for supply of the serial clock.

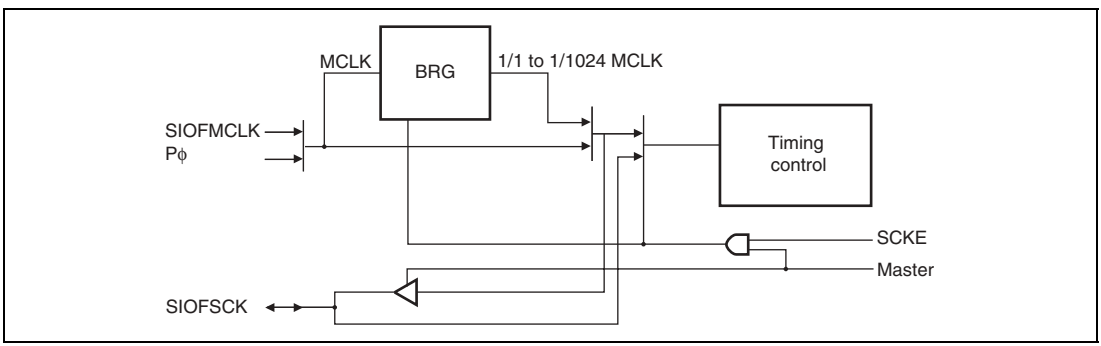


Figure 15.2 Serial Clock Supply

Table 15.3 shows an example of serial clock frequency.

Table 15.3 SIOF Serial Clock Frequency

	Sampling Rate		
Frame Length	**8 kHz**	**44.1 kHz**	**48 kHz**
32 bits	256 kHz	1.4112 MHz	1.536 MHz
64 bits	512 kHz	2.8224 MHz	3.072 MHz
128 bits	1.024 MHz	5.6448 MHz	6.144 MHz
256 bits	2.048 MHz	11.289 MHz	12.289 MHz

15.4.2 Serial Timing

SIOFSYNC: The SIOFSYNC is a frame synchronous (FS) signal. Depending on the transfer mode, it has the following two functions.

- Synchronous pulse: 1-bit-width pulse indicating the start of the frame
- L/R: 1/2-frame-width pulse indicating the left-channel stereo data (L) in high level and the right-channel stereo data (R) in low level

Figure 15.3 shows the SIOFSYNC synchronization timing.

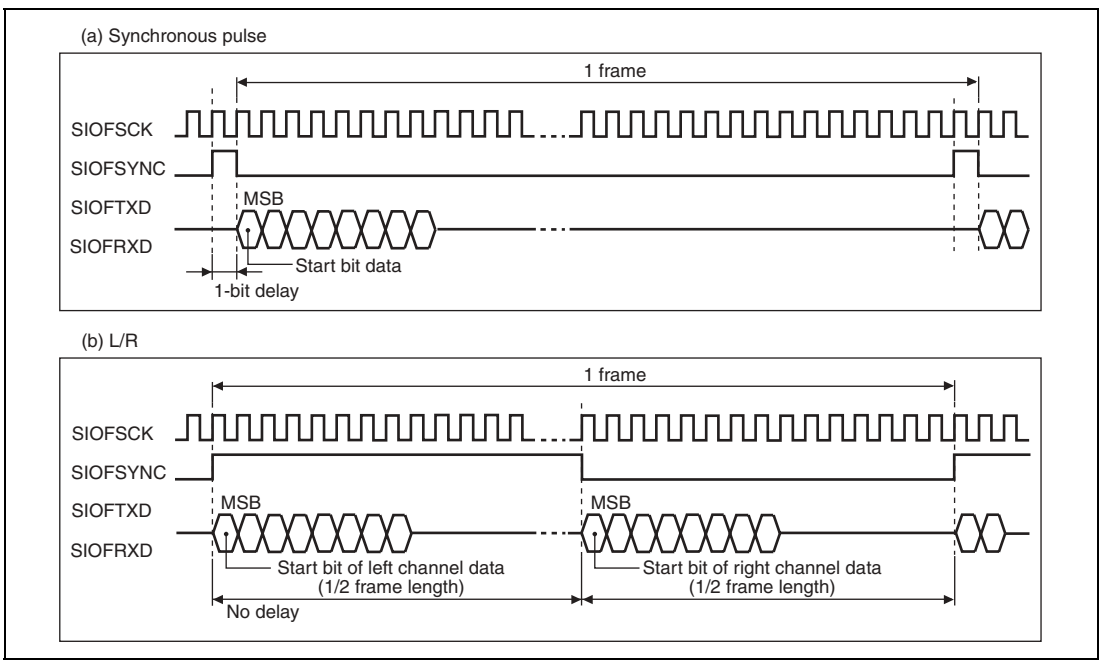


Figure 15.3 Serial Data Synchronization Timing

Transmit/Receive Timing: The SIOFTXD transmit timing and SIOFRXD receive timing relative to the SIOFSCK can be set as the sampling timing in the following two ways. The transmit/receive timing is set using the REDG bit in SIMDR.

- Falling-edge sampling
- Rising-edge sampling

Figure 15.4 shows the transmit/receive timing.

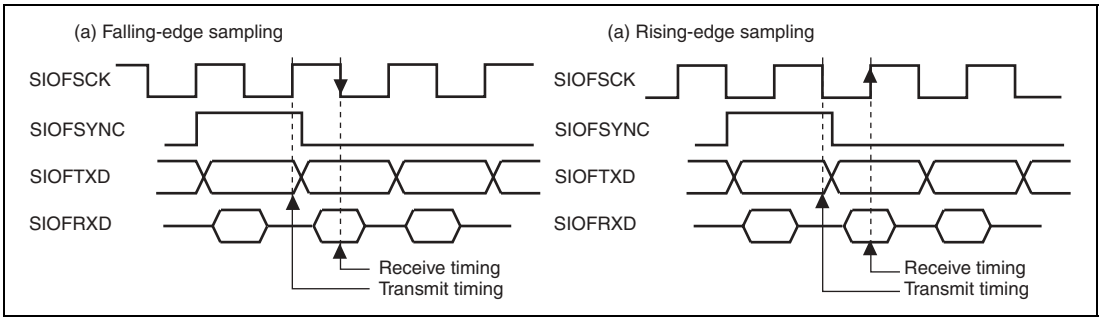


Figure 15.4 SIOF Transmit/Receive Timing

15.4.3 Transfer Data Format

The SIOF performs the following transfer.

- Transmit/receive data: Transfer of 8-bit data/16-bit data/16-bit stereo data
- Control data: Transfer of 16-bit data (uses the specific register as interface)

Transfer Mode: The SIOF supports the following four transfer modes as listed in table 15.4. The transfer mode can be specified by the TRMD1 and TRMD0 bits in SIMDR.

Table 15.4 Serial Transfer Modes

Transfer Mode	SIOFSYNC	Bit Delay	Control Data
Slave mode 1	Synchronous pulse	SYNCDL bit	Slot position
Slave mode 2	Synchronous pulse		Secondary FS
Master mode 1	Synchronous pulse		Slot position
Master mode 2	L/R	No	Not supported

Frame Length: The length of the frame to be transferred by the SIOF is specified by the FL3 to FL0 bits in SIMDR. Table 15.5 shows the relationship between the FL3 to FL0 bit settings and frame length.

Table 15.5 Frame Length

FL3 to FL0	Slot Length	Number of Bits in a Frame	Transfer Data
00xx	8	8	8-bit monaural data
0100	8	16	8-bit monaural data
0101	8	32	8-bit monaural data
0110	8	64	8-bit monaural data
0111	8	128	8-bit monaural data
10xx	16	16	16-bit monaural data
1100	16	32	16-bit monaural/stereo data
1101	16	64	16-bit monaural/stereo data
1110	16	128	16-bit monaural/stereo data
1111	16	256	16-bit monaural/stereo data

Note: x: Don't care

Slot Position: The SIOF can specify the position of transmit data, receive data, and control data in a frame (common to transmission and reception) by slot numbers. The slot number of each data is specified by the following registers.

- Transmit data: SITDAR
- Receive data: SIRDAR
- Control data: SICDAR

Only 16-bit data is valid for control data. In addition, control data is always assigned to the same slot number both in transmission and reception.

15.4.4 Register Allocation of Transfer Data

Transmit/Receive Data: Writing and reading of transmit/receive data is performed for the following registers.

- Transmit data writing: SITDR (32-bit access)
- Receive data reading: SIRDR (32-bit access)

Figure 15.5 shows the transmit/receive data and the SITDR and SIRDR bit alignment.

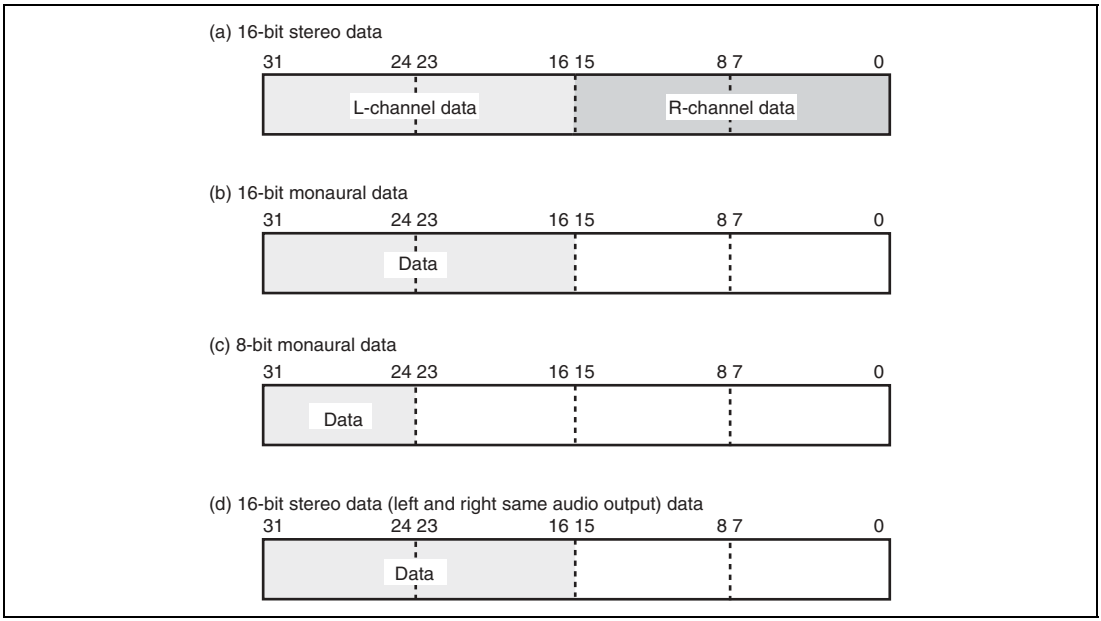


Figure 15.5 Transmit/Receive Data Bit Alignment

Note: In the figure, only the shaded areas are transmitted or received as valid data. Therefore, access must be made in byte units for 8-bit data, and in word units for 16-bit data. Data in unshaded areas is not transmitted or received.

Monaural or stereo can be specified for transmit data by the TDLE bit and TDRE bit in SITDAR. Monaural or stereo can be specified for receive data by the RDLE bit and RDRE bit in SIRDAR. To achieve left and right same audio output while stereo is specified for transmit data, specify the TLREP bit in SITDAR. Tables 15.6 and 15.7 show the audio mode specification for transmit data and that for receive data, respectively.

Table 15.6 Audio Mode Specification for Transmit Data

	Bit		
Mode	**TDLE**	**TDRE**	**TLREP**
Monaural	1	0	x
Stereo	1	1	0
Left and right same audio output	1	1	1

Note: x: Don't care

Table 15.7 Audio Mode Specification for Receive Data

	Bit	
Mode	**RDLE**	**RDRE**
Monaural	1	0
Stereo	1	1

Note: Left and right same audio mode is not supported in receive data.
To execute 8-bit monaural transmission or reception, use the left channel.

Control Data: Control data is written to or read from by the following registers.

- Transmit control data write: SITCR (32-bit access)
- Receive control data read: SIRCR (32-bit access)

Figure 15.6 shows the control data and bit alignment in SITCR and SIRCR.

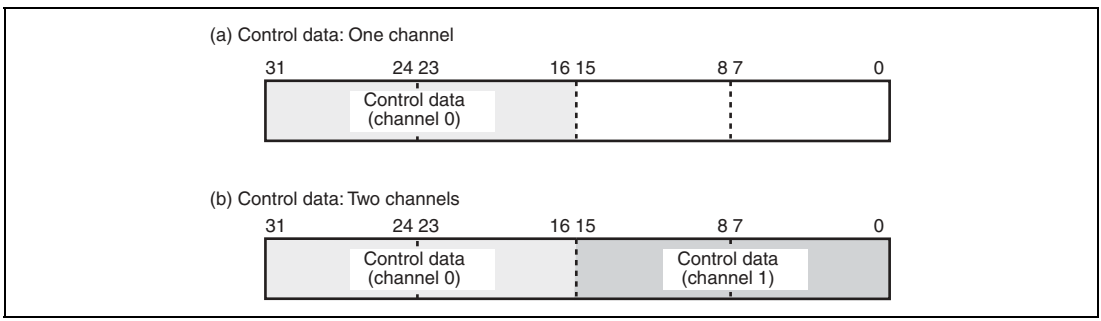


Figure 15.6 Control Data Bit Alignment

The number of channels in control data is specified by the CD0E and CD1E bits in SICDAR. Table 15.8 shows the relationship between the number of channels in control data and bit settings.

Table 15.8 The Number of Channels in Control Data Setting

	Bit	
Number of Channels	**CD0E**	**CD1E**
1	1	0
2	1	1

Note: To use only one channel in control data, use channel 0.

15.4.5 Control Data Interface

Control data performs control command output to the CODEC and status input from the CODEC. The SIOF supports the following two control data interface methods.

- Control by slot position
- Control by secondary FS

Control data is valid only when data length is specified as 16 bits.

Control by Slot Position (Master Mode 1, Slave Mode 1): Control data is transferred for all frames transmitted or received by the SIOF by specifying the slot position of control data. This method can be used in both SIOF master and slave modes. Figure 15.7 shows an example of the control data interface timing by slot position control.

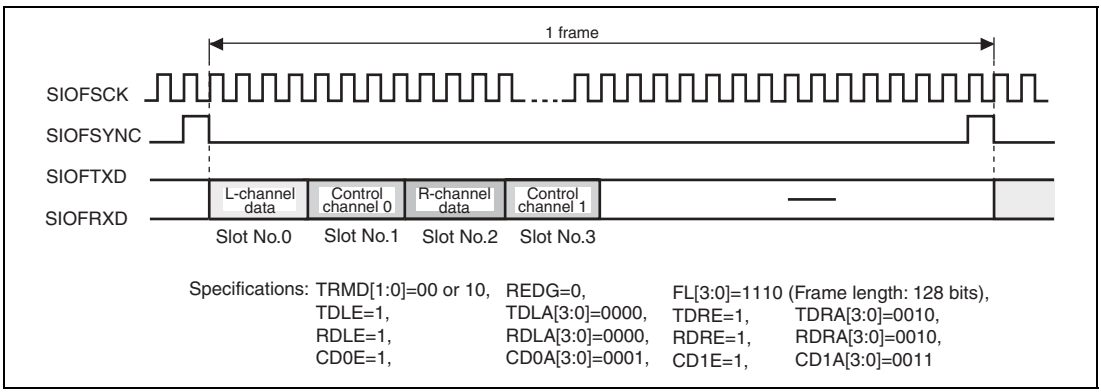


Figure 15.7 Control Data Interface (Slot Position)

Control by Secondary FS (Slave Mode 2): The CODEC normally outputs the SIOFSYNC signal as synchronization pulse (FS). In this method, the CODEC outputs the secondary FS specific to the control data transfer after 1/2 frame time has been passed (not the normal FS output timing) to transmit or receive control data. This method is valid for SIOF slave mode. The following summarizes the control data interface procedure by the secondary FS.

- Transmit normal transmit data of LSB = 0 (the SIOF forcibly clears 0).
- To execute control data transmission, send transmit data of LSB = 1 (the SIOF forcibly set to 1 by writing SITCDR).
- The CODEC outputs the secondary FS.
- The SIOF transmits or receives (stores in SIRCDR) control data (data specified by SITCDR) synchronously with the secondary FS.

Figure 15.8 shows an example of the control data interface timing by the secondary FS.

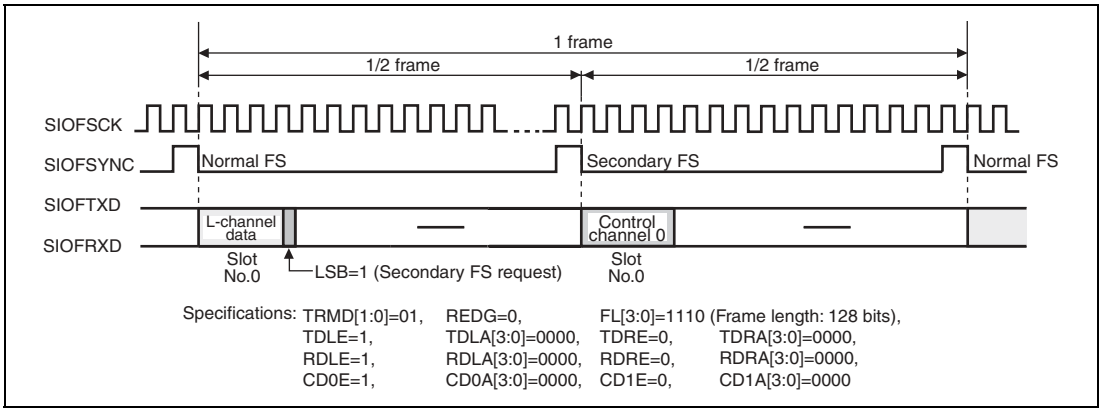


Figure 15.8 Control Data Interface (Secondary FS)

15.4.6 FIFO

Overview: The transmit and receive FIFOs of the SIOF have the following features.

- 16-stage 32-bit FIFOs for transmission and reception
- The FIFO pointer can be updated in one read or write cycle regardless of access size of the CPU and DMAC. (One-stage 32-bit FIFO access cannot be divided into multiple accesses.)

Transfer Request: The transfer request of the FIFO can be issued to the CPU or DMAC as the following interrupt sources.

- FIFO transmit request: TDREQ (transmit interrupt source)
- FIFO receive request: RDREQ (receive interrupt source)

The request conditions for FIFO transmit or receive can be specified individually. The request conditions for the FIFO transmit and receive are specified by the TFWM2 to TFWM0 bits and RFWM2 to RFWM0 bits in SIFCTR, respectively. Tables 15.9 and 15.10 summarize the conditions specified by SIFCTR.

Table 15.9 Conditions to Issue Transmit Request

TFWM2 to TFWM0	Number of Requested Stages	Transmit Request	Used Areas
000	1	Empty area is 16 stages	Smallest
100	4	Empty area is 12 stages or more	
101	8	Empty area is 8 stages or more	
110	12	Empty area is 4 stages or more	
111	16	Empty area is 1 stage or more	Largest

Table 15.10 Conditions to Issue Receive Request

RFWM2 to RFWM0	Number of Requested Stages	Receive Request	Used Areas
000	1	Valid data is 1 stage or more	Smallest
100	4	Valid data is 4 stages or more	
101	8	Valid data is 8 stages or more	
110	12	Valid data is 12 stages or more	
111	16	Valid data is 16 stages	Largest

The number of stages of the FIFO is always sixteen even if the data area or empty area exceeds the FIFO size (the number of FIFOs). Accordingly, an overflow error or underflow error occurs if data area or empty area exceeds sixteen FIFO stages. The FIFO transmit or receive request is canceled when the above condition is not satisfied even if the FIFO is not empty or full.

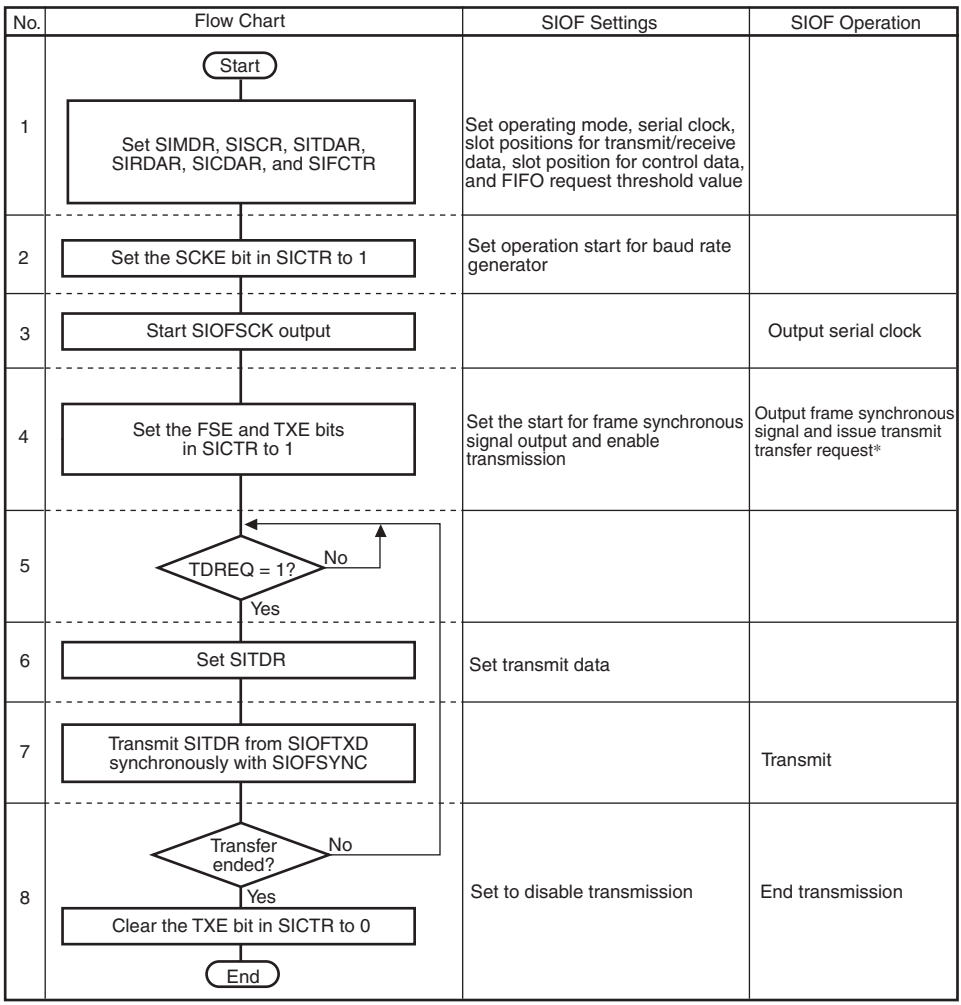
Number of FIFOs: The number of FIFO stages used in transmission and reception is indicated by the following register.

- Transmit FIFO: The number of empty FIFO stages is indicated by the TFUA4 to TFUA0 bits in SIFCTR.
- Receive FIFO: The number of valid data stages is indicated by the RFUA4 to RFUA0 bits in SIFCTR.

  The above indicate possible data numbers that can be transferred by the CPU or DMAC.

15.4.7 Transmit and Receive Procedures

Transmission in Master Mode: Figure 15.9 shows an example of settings and operation for master mode transmission.



Note: * When interrupts due to transmit data underflow are enabled, after setting the no. 6 transmit data, the TXE bit should be set to 1.

Figure 15.9 Example of Transmit Operation in Master Mode

Reception in Master Mode: Figure 15.10 shows an example of settings and operation for master mode reception.

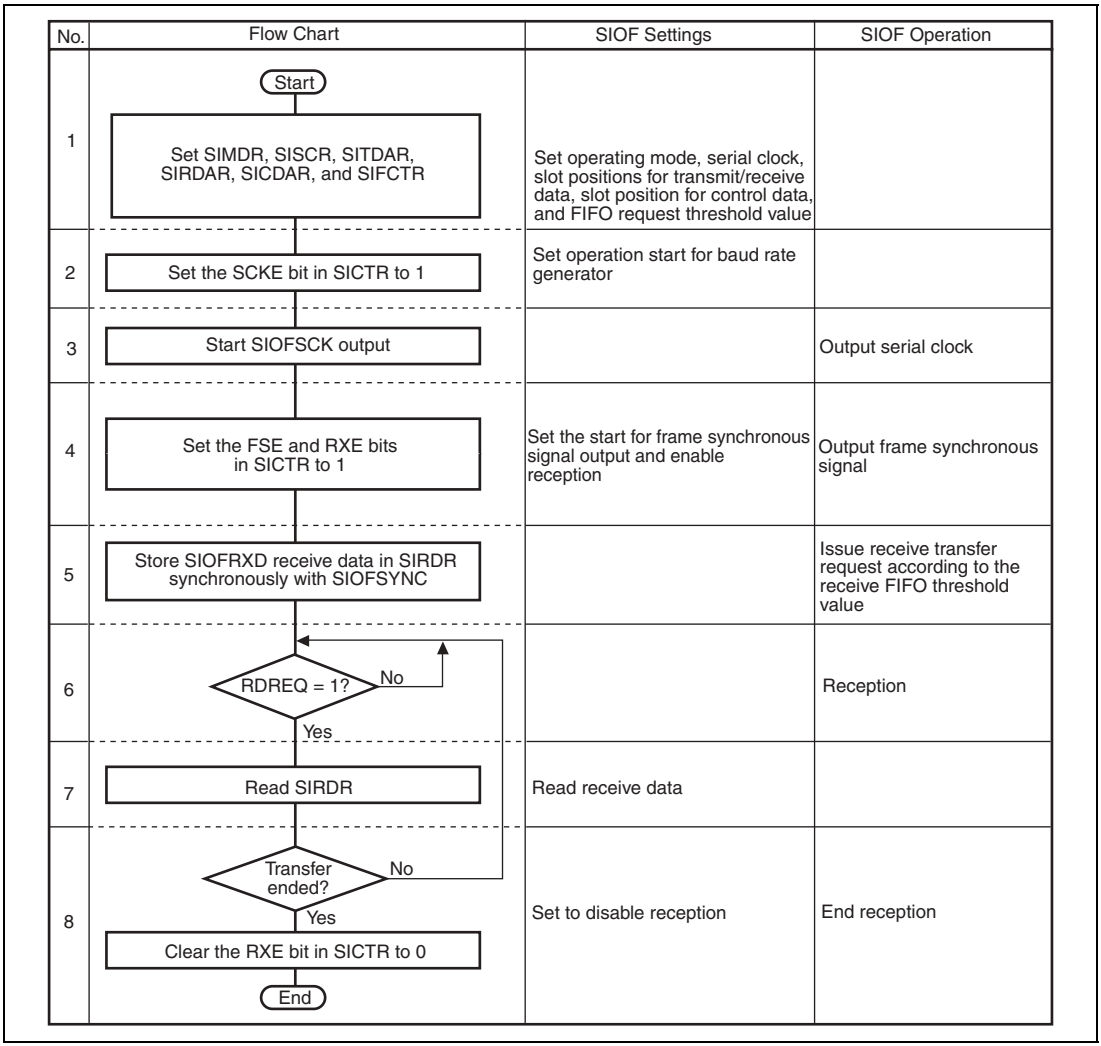


Figure 15.10 Example of Receive Operation in Master Mode

Transmission in Slave Mode: Figure 15.11 shows an example of settings and operation for slave mode transmission.

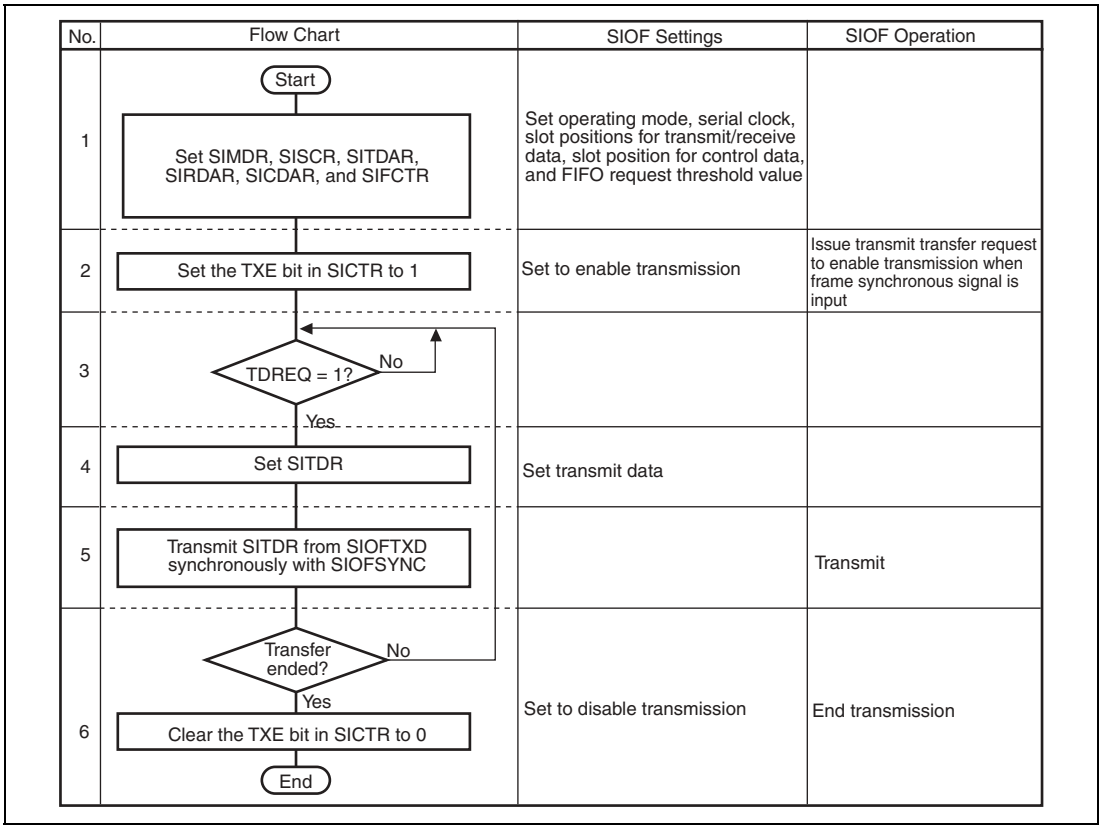


Figure 15.11 Example of Transmit Operation in Slave Mode

Reception in Slave Mode: Figure 15.12 shows an example of settings and operation for slave mode reception.

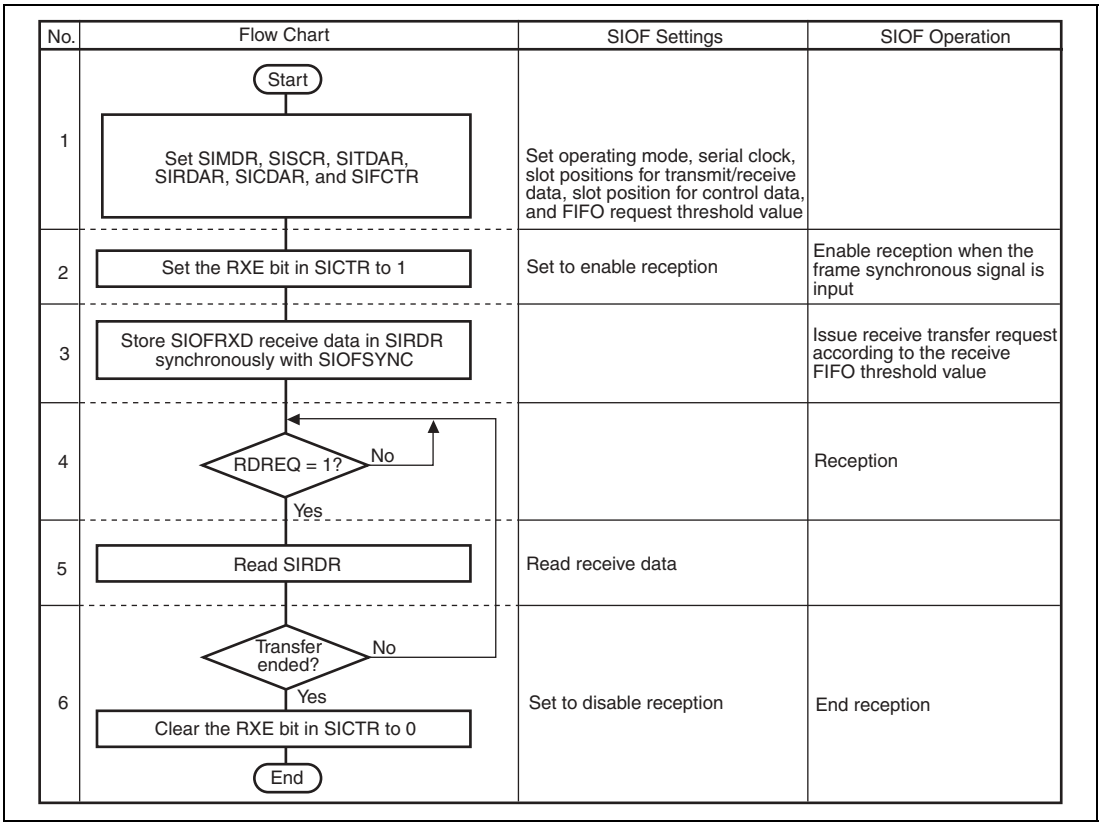


Figure 15.12 Example of Receive Operation in Slave Mode

Transmit/Receive Reset: The SIOF can separately reset the transmit and receive units by setting the following bits to 1.

- Transmit reset: TXRST bit in SICTR
- Receive reset: RXRST bit in SICTR

Table 15.11 shows the details of initialization upon transmit or receive reset.

Table 15.11 Transmit and Receive Reset

Type	Objects Initialized
Transmit reset	SITDR
	Transmit FIFO write pointer
	TCRDY, TFEMP, and TDREQ bits in SISTR
	TXE bit in SICTR
Receive reset	SIRDR
	Receive FIFO read pointer
	RCRDY, RFFUL, and RDREQ bits in SISTR
	RXE bit in SICTR

Module Stop Mode: The SIOF stops the transmit/receive operation in module stop mode. Then the following contents are initialized.

- SITDR
- SITCR
- Read pointer of receive FIFO
- Write pointer of transmit FIFO
- SISTR
- SICTR

15.4.8 Interrupts

The SIOF has one type of interrupt.

Interrupt Sources: Interrupts can be issued by several sources. Each source is shown as an SIOF status in SISTR. Table 15.12 lists the SIOF interrupt sources.

Table 15.12 SIOF Interrupt Sources

No.	Classification	Bit Name	Function Name	Description
1	Transmission	TDREQ	Transmit FIFO transfer request	The transmit FIFO stores data of specified size or more.
2		TFEMP	Transmit FIFO empty	The transmit FIFO is empty.
3	Reception	RDREQ	Receive FIFO transfer request	The receive FIFO stores data of specified size or more.
4		RFFUL	Receive FIFO full	The receive FIFO is full.
5	Control	TCRDY	Transmit control data ready	The transmit control register is ready to be written.
6		RCRDY	Receive control data ready	The receive control data register stores valid data.
7	Error	TFUDF	Transmit FIFO underflow	Serial data transmit timing has arrived while the transmit FIFO is empty.
8		TFOVF	Transmit FIFO overflow	Write to the transmit FIFO is performed while the transmit FIFO is full.
9		RFOVF	Receive FIFO overflow	Serial data is received while the receive FIFO is full.
10		RFUDF	Receive FIFO underflow	The receive FIFO is read while the receive FIFO is empty.
11		FSERR	FS error	A synchronous signal is input before the specified bit number has been passed (in slave mode).
12		SAERR	Slot assign error	The same slot is specified in both serial data and control data.

Whether an interrupt is issued or not as the result of an interrupt source is determined by the SIIER settings. If an interrupt source is set to 1 and the corresponding bit in SIIER is set to 1, an SIOF interrupt issued.

Regarding Transmit and Receive Classification: The transmit sources and receive sources are signals indicating the state; after being set, if the state changes, they are automatically cleared by the SIOF.

When the DMA transfer is used, a DMA transfer request is pulled low (0 level) for one cycle at the end of DMA transfer.

Processing when Errors Occur: On occurrence of each of the errors indicated as a status in SISTR, the SIOF performs the following operations.

- Transmit FIFO underflow (TFUDF)

  The immediately preceding transmit data is again transmitted.
- Transmit FIFO overflow (TFOVF)

  The contents of the transmit FIFO are protected, and the write operation causing the overflow is ignored.
- Receive FIFO overflow (RFOVF)

  Data causing the overflow is discarded and lost.
- Receive FIFO underflow (RFUDF)

  An undefined value is output on the bus.
- FS error (FSERR)

  The internal counter is reset according to the FSYN signal in which an error occurs.
- Slot assign error (SAERR)
  - If the same slot is assigned to both serial data and control data, the slot is assigned to serial data.
  - If the same slot is assigned to two control data items, data cannot be transferred correctly.

15.4.9 Transmit and Receive Timing

Examples of the SIOF serial transmission and reception are shown in figure 15.13 through figure 15.20.

8-bit Monaural Data (1): Synchronous pulse method, falling edge sampling, slot No.0 used for transmit and receive data, a frame length = 8 bits

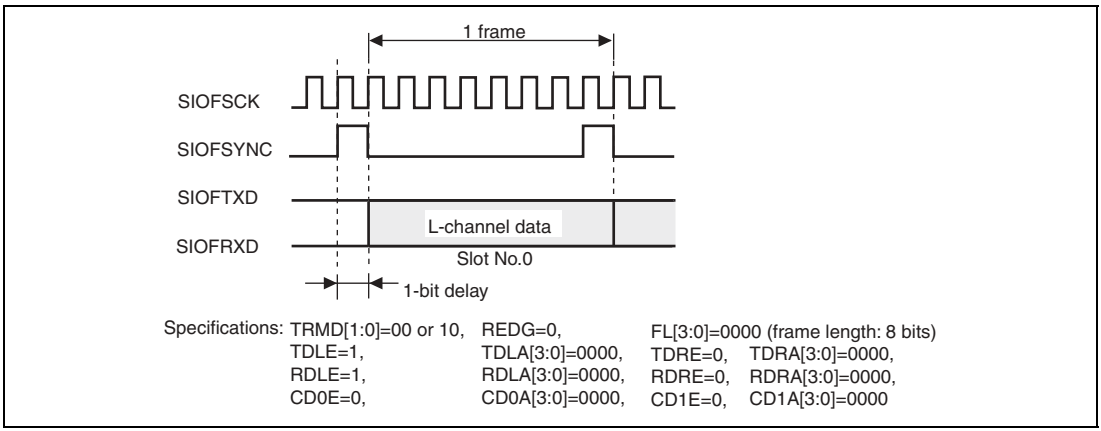


Figure 15.13 Transmit and Receive Timing (8-Bit Monaural Data (1))

8-bit Monaural Data (2): Synchronous pulse method, falling edge sampling, slot No.0 used for transmit and receive data, and frame length = 16 bits

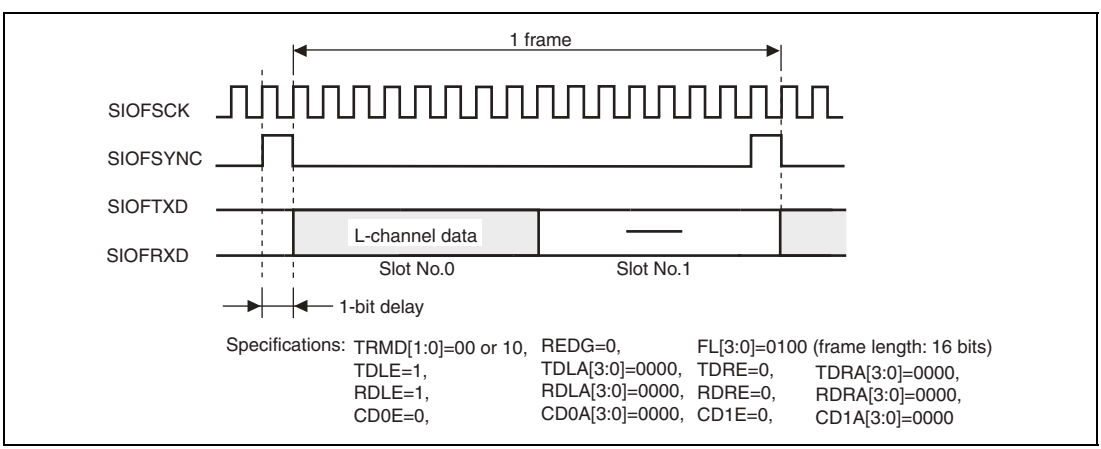


Figure 15.14 Transmit and Receive Timing (8-Bit Monaural Data (2))

16-bit Monaural Data (1): Synchronous pulse method, falling edge sampling, slot No.0 used for transmit and receive data, and frame length = 64 bits

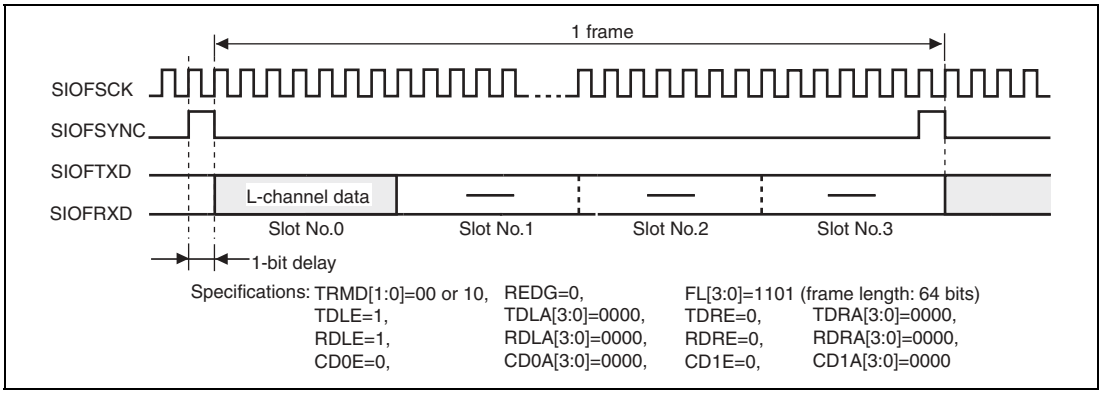


Figure 15.15 Transmit and Receive Timing (16-Bit Monaural Data (1))

16-bit Stereo Data (1): L/R method, rising edge sampling, slot No.0 used for left channel data, slot No.1 used for right channel data, and frame length = 32 bits

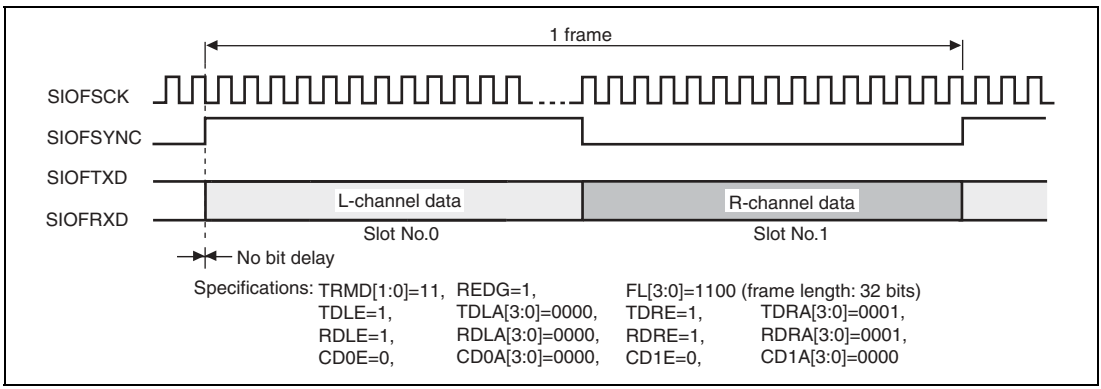


Figure 15.16 Transmit and Receive Timing (16-Bit Stereo Data (1))

16-bit Stereo Data (2): L/R method, rising edge sampling, slot No.0 used for left-channel transmit data, slot No.1 used for left-channel receive data, slot No.2 used for right-channel transmit data, slot No.3 used for right-channel receive data, and frame length = 64 bits

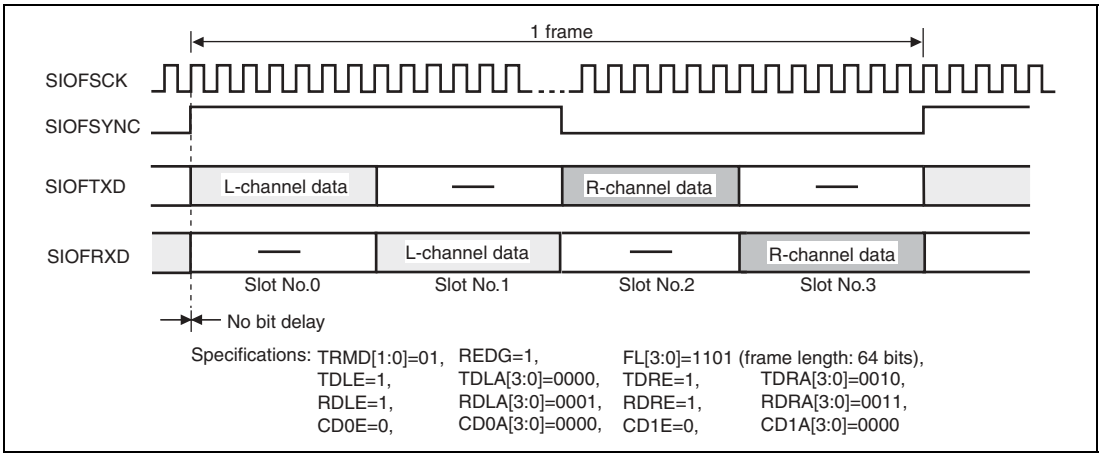


Figure 15.17 Transmit and Receive Timing (16-Bit Stereo Data (2))

16-bit Stereo Data (3): Synchronous pulse method, falling edge sampling, slot No.0 used for left-channel data, slot No.1 used for right-channel data, slot No.2 used for control channel 0 data, slot No.3 used for control channel 1 data, and frame length = 128 bits

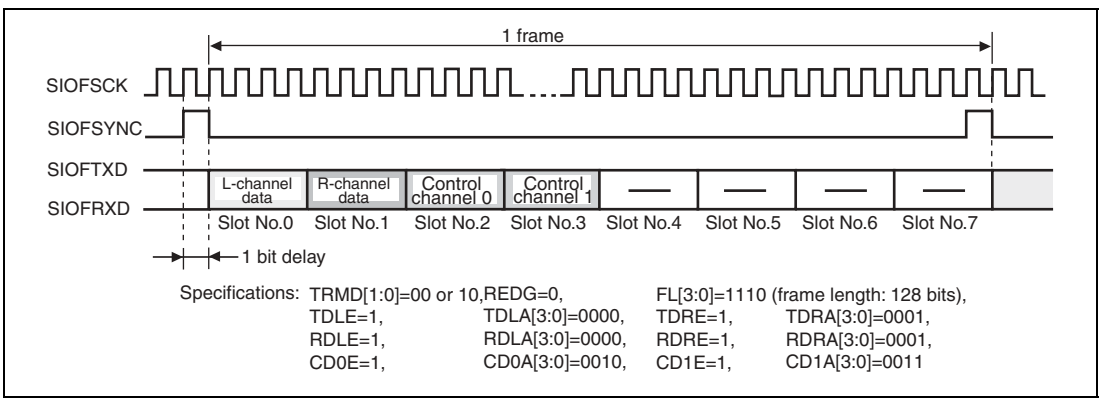


Figure 15.18 Transmit and Receive Timing (16-Bit Stereo Data (3))

16-bit Stereo Data (4): Synchronous pulse method, falling edge sampling, slot No.0 used for left-channel data, slot No.2 used for right-channel data, slot No.1 used for control channel 0 data, slot No.3 used for control channel 1 data, and frame length = 128 bits

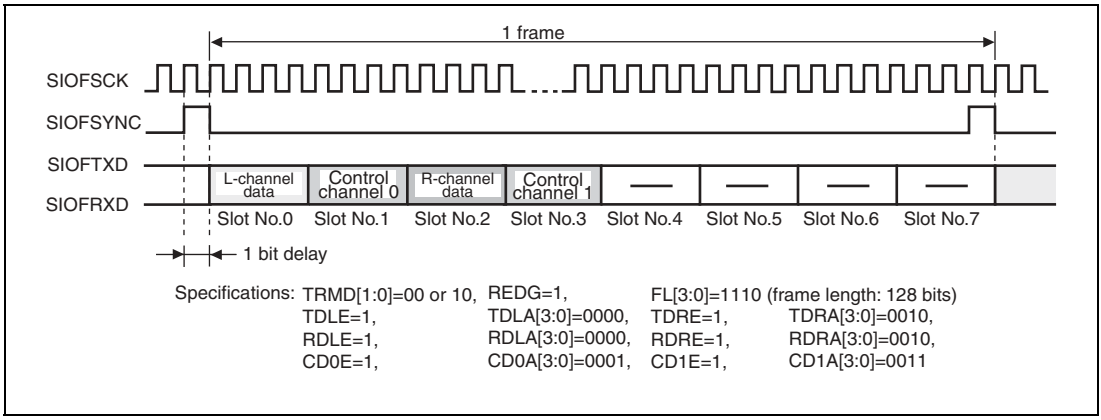


Figure 15.19 Transmit and Receive Timing (16-Bit Stereo Data (4))

Synchronization-Pulse Output Mode at End of Each Slot (SYNCAT Bit = 1): Synchronous pulse method, falling edge sampling, slot No.0 used for left-channel data, slot No.1 used for right-channel data, slot No.2 used for control channel 0 data, slot No.3 used for control channel 1 data, and frame length = 128 bits

In this mode, valid data must be set to slot No. 0.

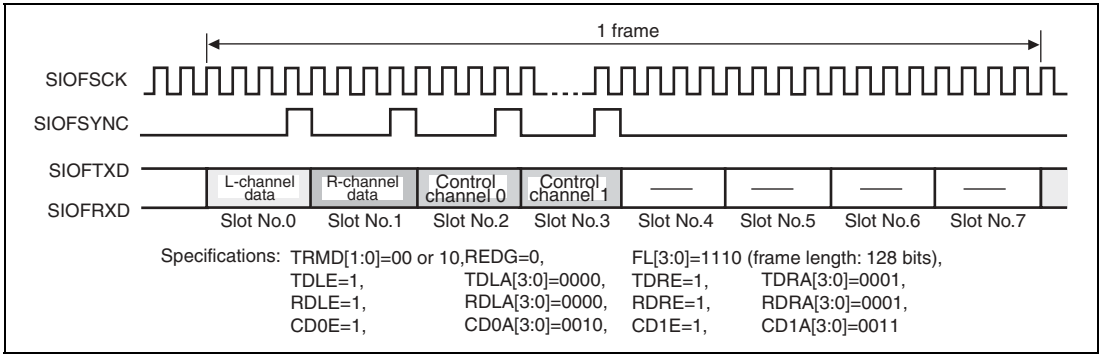


Figure 15.20 Transmit and Receive Timing (16-Bit Stereo Data)

15.4.10 SPI Mode

SPI-mode operation is selected for the SIOF by the setting in SPICR.

Example of Configuration: Figure 15.21 shows an example of the configuration for SPI-mode communications.

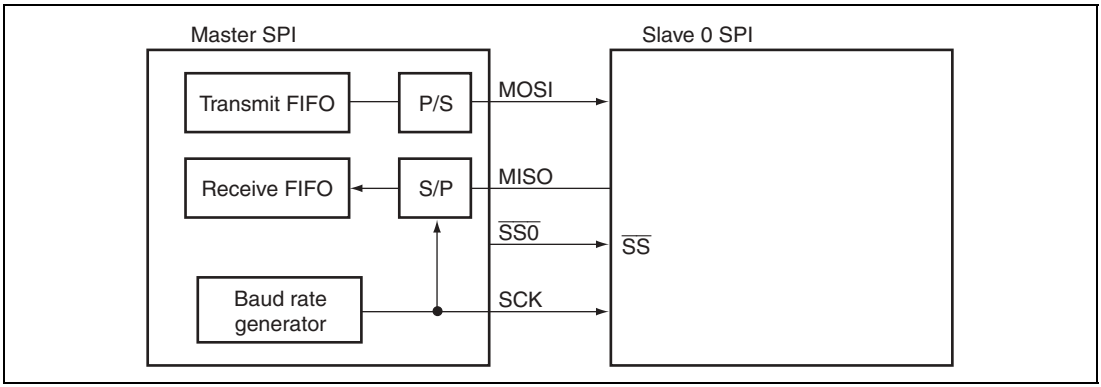
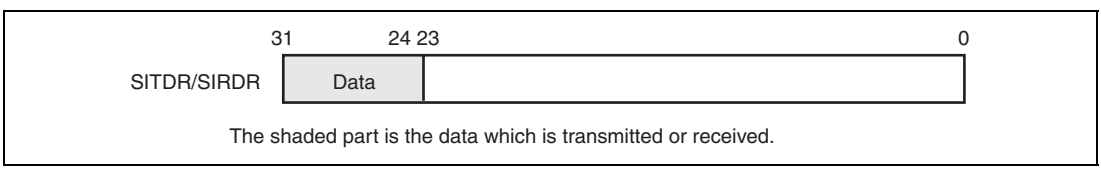


Figure 15.21 Example of Configuration in SPI Mode

SPI Operation: The states of operation in SPI mode are described in terms of transmission and reception in table 15.13. In SPI mode, the data length is fixed to 8 bits and the values of the upper 8 bits of SITDR and SIRDR are the valid data for transmission and reception, respectively.



The only sources of interrupts that should be enabled in SPI-mode transfer are transmit data transfer request (TDREQ), transmit FIFO empty (TFEMP), receive-data transfer request (RDREQ), receive-FIFO full (RFFUL), and receive-FIFO overflow (RFOVF). Enabled or disabled states are selectable by the interrupt enable register (SIIER). Interrupts from other sources must be disabled at all times.

For the DMA transfer requests, the enabled sources are transmit-data DMA transfer request (TDMA) and receive-data DMA transfer request (RDMA). Enabled or disabled states are selectable by the interrupt enable register (SIIER).

In SPI mode, the baud rate is set by SISCR.

Table 15.13 States of Transmit and Receive Operations in SPI Mode

TXE	RXE	TDMAE	RDMAE	SPI Transmit/Receive Operation
0	0	Don't care	Don't care	Transmission/reception is disabled
0	1	0	1	Half-Duplex Reception The transmit FIFO does not operate and dummy data is transmitted from the MOSI. Data received at the MISO is stored in the receive FIFO and is transferred by using the DMA. Receive operation continues as long as RE bit = 1; the receive-FIFO overflow (RFOVF) status is set after the receive FIFO has become full and further receive data is ignored.
1	0	0	0	Half-Duplex Transmission The data in the transmit FIFO is transmitted from the MOSI. The receive FIFO does not operate, and data on the MISO is ignored. When the transmit FIFO becomes empty, the transmit operation is completed.
		1	0	Half-Duplex Transmission The data which has been transferred by using the DMA to the transmit FIFO is transmitted from the MOSI. The receive FIFO does not operate and data on the MISO is ignored. When the transmit FIFO becomes empty, the transmit operation is completed.
1	1	0	0	Full-Duplex Communication The transmit and receive FIFOs operate at the same time. Data in the transmit FIFO are transmitted or received. When there is no data left in the transmit FIFO, the transmit and receive operations are completed.

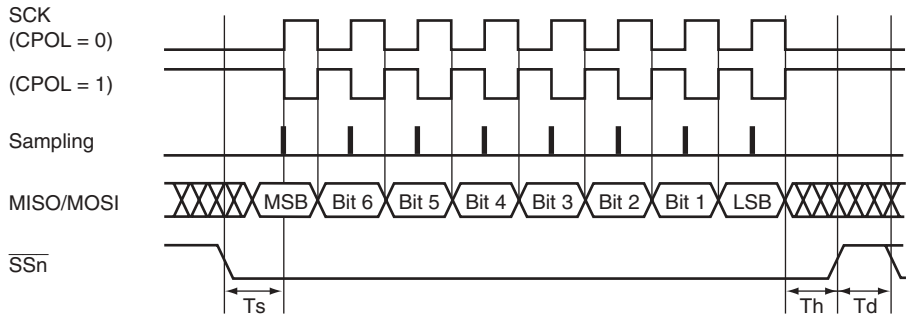
Note: In SPI mode, settings other than the above are prohibited.

In half-duplex reception (transmission is disabled), the value output from the MOSI can be controlled by the TXDIZ bit in SIMDR as follows.

TXDIZ = 0: Transmission is disabled, 1 is output on the MOSI.

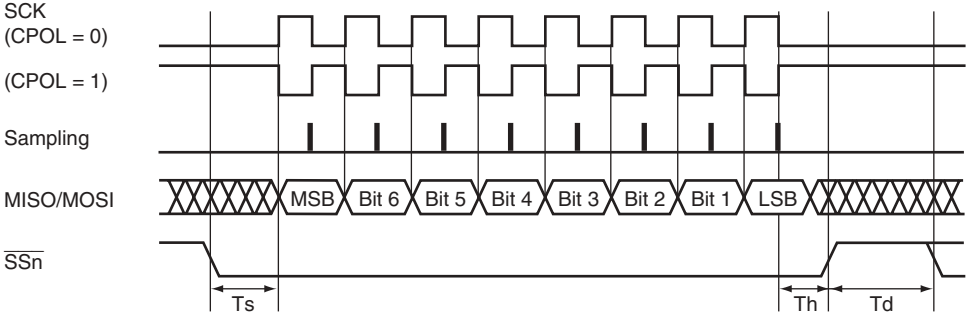
TXDIZ = 1: Transmission is disabled, the MOSI is in the high-impedance state.

Serial Clock Timing: Timing on the data and clock lines in SPI mode is shown in figures 15.22 and 15.23. The user can select from four serial transfer formats, which differ according to the phase and polarity of the serial clock.



Ts: The setup time for the SCK edge. The minimum value is 1/2 of the period of the SCK. The setting is made by the SSAST1 and SSAST0 bits in SPICR.
Th: The hold time for the SCK edge. The minimum value is 0.
Td: The idle time. A number of SCK-clock cycles from 0 to 3 is set by the FLD1 and FLD0 bits in SPICR.

Figure 15.22 SPI Data/Clock Timing 1 (CPHA = 0)



Ts: The setup time for the SCK edge. The minimum value is 0. The setting is made by the SSAST1 and SSAST0 bits in SPICR.
Th: The hold time for the SCK edge. The minimum value is 1/2 of the period of the SCK.
Td: The idle time. A number of SCK-clock cycles from 0 to 3 is set by the FLD1 and FLD0 bits in SPICR.

Figure 15.23 SPI Data/Clock Timing 2 (CPHA = 1)

15.5 Usage Notes

1. The transitions to the software standby mode and module standby mode should not be made by setting the module standby bit of the SIOF during data reception and transmission. The transitions to software standby mode and module standby mode should be made after both TXE and RXE bits in the SICTR register have been cleared (transmission/reception disabled).
2. The SIOFTXD pin in SIOF is output pin even when the TXE bit is cleared. The SIOFRXD pin is in the input enabled state even when the RXE bit is cleared. In software standby mode, the SIOFTXD pin is high-impedance and the SIOFRXD pin is fixed to input disable state. Note that the SIOFTXD pin is multiplexed with the port PTA1 and its initial settings after a power-on reset are an input pin and the pull-up MOS is on. The SIOFRXD pin is multiplexed with the port PTA2, and the initial settings after a power-on reset are an input pin and the pull-up MOS is on. Care is required on dealing the signal lines that are connected to the SIOFTXD and SIOFRXD pins.
3. The SIOF has some 32-bit registers that handle two pieces of independent 16-bit data. Note that the order of upper word and lower word on the memory may be switched within the register in a data transfer with above registers when switching the big endian and little endian by using the MD5 pin.
4. When the SIOF is set as a transfer destination or transfer source, a transfer is carried out only in longword units.

Section 16 Serial Communication Interface with FIFO (SCIF)

This LSI has a two-channel serial communication interface with on-chip FIFO buffers (Serial Communication Interface with FIFO: SCIF). The SCIF can perform asynchronous and clock synchronous serial communication.

64-stage FIFO registers are provided for both transmission and reception, enabling fast, efficient, and continuous communication.

16.1 Features

- Asynchronous mode

  Serial data communication is executed using an asynchronous system in which synchronization is achieved character by character. Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA).

  There is a choice of eight serial data communication formats.

  — Data length: 7 or 8 bits
  — Stop bit length: 1 or 2 bits
  — Parity: Even/odd/none
  — LSB-first transfer
  — Receive error detection: Parity, framing, and overrun errors
  — Break detection: If a framing error is followed by at least one frame at the space “0” (low) level, a break is detected.
- Clock synchronous mode

  Serial data communication is synchronized with a clock. Serial data communication can be carried out with other chips that have a synchronous communication function.

  — Data length: 8 bits
  — LSB-first transfer
- Full-duplex communication capability

  The transmitter and receiver are independent units, enabling transmission and reception to be performed simultaneously.

  The transmitter and receiver both have a 64-stage FIFO buffer structure, enabling fast and continuous serial data transmission and reception.
- On-chip baud rate generator allows any bit rate to be selected.
- Choice of serial clock source: internal clock from baud rate generator or external clock from SCK pin.

- Six interrupt sources in asynchronous mode

  There are six interrupt sources—transmit-data-stop, transmit-FIFO-data-empty, receive-FIFO-data-full, receive-error (framing/parity error), break-receive, and receive-data-ready interrupts. The vectors of each interrupt are the same.
- Two interrupt sources in clock synchronous mode

  There are two interrupt sources—transmit-FIFO-data-empty and receive-FIFO-data-full interrupts. The vectors of each interrupt are the same.
- The DMA controller (DMAC) can be activated to execute a data transfer in the event of a transmit-FIFO-data-empty, transmit-data-stop, or receive-FIFO-data-full interrupt. The DMAC requests of transmit-FIFO-data-empty and transmit-data-stop interrupts are the same.
- On-chip modem control functions ($\overline{\text{CTS}}$ and $\overline{\text{RTS}}$)
- On-chip transmit-data-stop functions (only in asynchronous mode)
- When not in use, the SCIF can be stopped by halting its clock supply to reduce power consumption.
- The amount of data in the transmit/receive FIFO registers and the number of receive errors in the receive data in the receive FIFO register can be ascertained.

Figure 16.1 shows a block diagram of the SCIF.

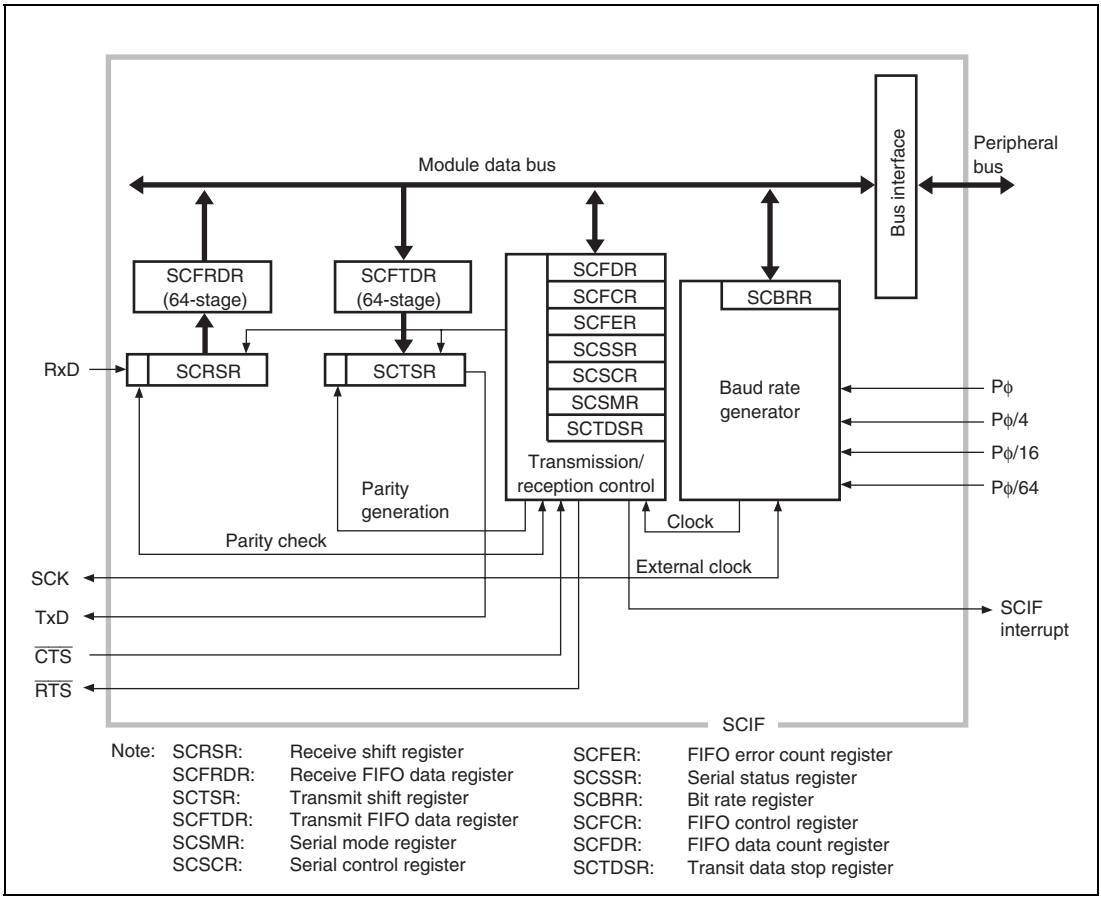


Figure 16.1 Block Diagram of SCIF

16.2 Input/Output Pins

Table 16.1 shows the SCIF pin configuration.

Table 16.1 Pin Configuration

Channel	Pin Name	Abbreviation*[1]	I/O	Function
0	SCIF0_SCK	SCK	Input/output	Serial clock pin
				Clock input/output
	SCIF0_RxD	RxD*[2]	Input	Receive data pin
				Receive data input
	SCIF0_TxD	TxD*[2]	Output	Transmit data pin
				Transmit data output
	SCIF0_$\overline{\text{CTS}}$	$\overline{\text{CTS}}$	Input	Modem control pin
				Transmission possible
	SCIF0_$\overline{\text{RTS}}$	$\overline{\text{RTS}}$	Output	Modem control pin
				Transmit request
1	SCIF1_SCK	SCK	Input/output	Serial clock pin
				Clock input/output
	SCIF1_RxD	RxD*[2]	Input	Receive data pin
				Receive data input
	SCIF1_TxD	TxD*[2]	Output	Transmit data pin
				Transmit data output
	SCIF1_$\overline{\text{CTS}}$	$\overline{\text{CTS}}$	Input	Modem control pin
				Transmission possible
	SCIF1_$\overline{\text{RTS}}$	$\overline{\text{RTS}}$	Output	Modem control pin
				Transmit request

Notes: 1. The pins are collectively called SCK, RxD, TxD, $\overline{\text{CTS}}$, and $\overline{\text{RTS}}$ without channel number in the following descriptions.

2. These pins are made to function as serial pins by setting SCIF operation with the TE and RE bits in SCSCR.

16.3 Register Descriptions

The SCIF has the following internal registers. For details on register addresses and register states in each processing state, refer to section 27, List of Registers.

Channel 0:

- Serial mode register_0 (SCSMR_0)
- Bit rate register_0 (SCBRR_0)
- Serial control register_0 (SCSCR_0)
- Transmit data stop register_0 (SCTDSR_0)
- FIFO error count register_0 (SCFER_0)
- Serial status register_0 (SCSSR_0)
- FIFO control register_0 (SCFCR_0)
- FIFO data count register_0 (SCFDR_0)
- Transmit FIFO data register_0 (SCFTDR_0)
- Receive FIFO data register_0 (SCFRDR_0)

Channel 1:

- Serial mode register_1 (SCSMR_1)
- Bit rate register_1 (SCBRR_1)
- Serial control register_1 (SCSCR_1)
- Transmit data stop register_1 (SCTDSR_1)
- FIFO error count register_1 (SCFER_1)
- Serial status register_1 (SCSSR_1)
- FIFO control register _1(SCFCR_1)
- FIFO data count register_1 (SCFDR_1)
- Transmit FIFO data register_1 (SCFTDR_1)
- Receive FIFO data register_1 (SCFRDR_1)

16.3.1 Receive Shift Register (SCRSR)

SCRSR is the register used to receive serial data.

The SCIF sets serial data input from the RxD pin in SCRSR in the order received, starting with the LSB (bit 0), and converts it to parallel data. When one byte of data has been received, it is transferred to the receive FIFO data register, SCFRDR, automatically.

SCRSR cannot be directly read or written to by the CPU.

16.3.2 Receive FIFO Data Register (SCFRDR)

SCFRDR is a 64-stage 8-bit FIFO register that stores received serial data (receive FIFO).

When the SCIF has received one byte of serial data, it transfers the received data from SCRSR to SCFRDR where it is stored, and completes the receive operation. SCRSR is then enabled for reception, and consecutive receive operations can be performed until the receive FIFO data register is full (64 data bytes).

SCFRDR is a read-only register, and cannot be written to by the CPU.

If a read is performed when there is no receive data in the receive FIFO data register, an undefined value will be returned. When the receive FIFO data register is full of receive data, subsequent serial data is lost.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	SCFRD7 to SCFRD0	Undefined	R	Serial Receive Data FIFO

16.3.3 Transmit Shift Register (SCTSR)

SCTSR is the register used to transmit serial data.

To perform serial data transmission, the SCIF first transfers transmit data from SCFTDR to SCTSR, then sends the data sequentially to the TxD pin starting with the LSB (bit 0).

When transmission of one byte is completed, the next transmit data is transferred from SCFTDR to SCTSR, and transmission is started automatically.

SCTSR cannot be directly read or written to by the CPU.

16.3.4 Transmit FIFO Data Register (SCFTDR)

SCFTDR is an 8-bit 64-stage FIFO data register that stores data for serial transmission (transmit FIFO).

If SCTSR is empty when transmit data is written to SCFTDR, the SCIF transfers the transmit data written in SCFTDR to SCTSR and starts serial transmission.

SCFTDR is a write-only register, and cannot be read by the CPU.

The next data cannot be written when SCFTDR is filled with 64 bytes of transmit data. Data written in this case is ignored.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	SCFTD7 to SCFTD0	Undefined	W	Serial Transmit Data FIFO

16.3.5 Serial Mode Register (SCSMR)

SCSMR is a 16-bit readable/writable register used to set the SCIF's serial transfer format and select the baud rate generator clock source and the sampling rate.

Bit	Bit Name	Initial Value	R/W	Description
15 to 11	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
10	SRC2	0	R/W	Sampling Control
9	SRC1	0	R/W	Select the sampling rate in asynchronous mode. This setting is valid only in asynchronous mode.
8	SRC0	0	R/W	000: Sampling rate 1/16 001: Sampling rate 1/5 010: Sampling rate 1/7 011: Sampling rate 1/11 100: Sampling rate 1/13 101: Sampling rate 1/17 110: Sampling rate 1/19 111: Sampling rate 1/27
7	C/A	0	R/W	Communication Mode Selects whether the SCI operates in the asynchronous or clock synchronous mode. 0: Asynchronous mode 1: Clock synchronous mode
6	CHR	0	R/W	Character Length Selects seven or eight bits as the data length. This setting is only valid in asynchronous mode. In clock synchronous mode, the data length is always eight bits, regardless of the CHR setting. 0: 8-bit data 1: 7-bit data* Note: * When the 7-bit data is selected, the MSB bit (bit 7) in the transmit FIFO data register (SCFTDR) is not transmitted.

Bit	Bit Name	Initial Value	R/W	Description
5	PE	0	R/W	Parity Enable Selects whether or not parity bit addition is performed in transmission, and parity bit checking in reception. This setting is only valid in asynchronous mode. In synchronous mode, parity bit addition and checking is not performed, regardless of the PE setting. 0: Parity bit addition and checking disabled 1: Parity bit addition and checking enabled* Note: * When the PE bit is set to 1, the parity (even or odd) specified by the O/E bit is added to transmit data before transmission. In reception, the parity bit is checked for the parity (even or odd) specified by the O/E bit.
4	O/E	0	R/W	Parity Mode Selects either even or odd parity for use in parity addition and checking. The O/E bit setting is only valid when the PE bit is set to 1, enabling parity bit addition and checking. The O/E bit setting is invalid when parity addition and checking is disabled in asynchronous and clock synchronous mode. 0: Even parity*[1] 1: Odd parity*[2] Notes: 1. When even parity is set, parity bit addition is performed in transmission so that the total number of 1-bits in the transmit character plus the parity bit is even. In reception, a check is performed to see if the total number of 1-bits in the receive character plus the parity bit is even. 2. When odd parity is set, parity bit addition is performed in transmission so that the total number of 1-bits in the transmit character plus the parity bit is odd. In reception, a check is performed to see if the total number of 1-bits in the receive character plus the parity bit is odd.

Bit	Bit Name	Initial Value	R/W	Description
3	STOP	0	R/W	Stop Bit Length Selects one or two bits as the stop bit length. In reception, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, it is treated as a stop bit; if it is 0, it is treated as the start bit of the next transmit character. This setting is only valid in asynchronous mode. In clock synchronous mode, this setting is invalid since stop bits are not added. 0: One stop bit*[1] 1: Two stop bits*[2] Notes: 1. In transmission, a single 1-bit (stop bit) is added to the end of a transmit character before it is sent. 2. In transmission, two 1-bits (stop bits) are added to the end of a transmit character before it is sent.
2	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
1	CKS1	0	R/W	Clock Select
0	CKS0	0	R/W	Select the clock source for the on-chip baud rate generator. 00: Pϕ 01: Pϕ/4 10: Pϕ/16 11: Pϕ/64

16.3.6 Serial Control Register (SCSCR)

SCSCR is a 16-bit readable/writable register that enables or disables the SCIF transfer operations and interrupt requests, and selects the serial clock source.

Bit	Bit Name	Initial Value	R/W	Description
15	TDRQE	0	R/W	Transmit Data Transmission Request Enable When transmitting data while TIE = 1 and transmit FIFO data empty has been generated, transmit FIFO data empty interrupt or DMA transfer request is selected. 0: Interrupt request to CPU is performed 1: Transmit data transmission request to the DMA is performed
14	RDRQE	0	R/W	Receive Data Transmission Request Enable When receiving data while RIE = 1 and received FIFO data full is generated, received FIFO data full interrupt or DMA transfer request is selected. 0: Interrupt request to CPU is performed 1: Transmit data transmission request to the DMA is performed
13, 12	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
11	TSIE	0	R/W	Transmit Data Stop Interrupt Enable Enables or disables generation of a transmit-data-stop interrupt when the TSE bit in SCFCR is enabled and the TSF flag in SCSSR is set to 1. 0: Transmit-data-stop interrupt disabled* 1: Transmit-data-stop interrupt enabled Note: * The interrupt request is cleared by clearing the TSF flag to 0 after reading 1 from it or clearing the TSIE bit to 0.

Bit	Bit Name	Initial Value	R/W	Description
10	ERIE	0	R/W	Receive Error Interrupt Enable Enables or disables generation of a receive-error (framing or parity error) interrupt when the ER flag in SCSSR is set to 1. 0: Receive-error interrupt disabled* 1: Receive-error interrupt enabled Note: * The interrupt request is cleared by clearing the ER flag to 0 after reading 1 from it or clearing the ERIE bit to 0.
9	BRIE	0	R/W	Break Interrupt Enable Enables or disables generation of a break-receive interrupt when the BRK flag in SCSSR is set to 1. 0: Break-receive interrupt disabled* 1: Break-receive interrupt enabled Note: * The interrupt request is cleared by clearing the BRK flag to 0 after reading 1 from it or clearing the BRIE bit to 0.
8	DRIE	0	R/W	Receive Data Ready Interrupt Enable Enables or disables generation of a receive-data-ready interrupt when the DR flag in SCSSR is set to 1. 0: Receive-data-ready interrupt disabled* 1: Receive-data-ready interrupt enabled Note: * The interrupt request is cleared by clearing the DR flag to 0 after reading 1 from it or clearing the DRIE bit to 0.
7	TIE	0	R/W	Transmit Interrupt Enable Enables or disables generation of a transmit-FIFO-data-empty interrupt request when the TDFE flag in SCSSR is set to 1. 0: Transmit-FIFO-data-empty interrupt request disabled* 1: Transmit-FIFO-data-empty interrupt request enabled Note: * The interrupt request is cleared by writing transmit data exceeding the transmit trigger set number to SCFTDR, reading 1 from the TDFE flag, then clearing it to 0, or clearing the TIE bit to 0.

Bit	Bit Name	Initial Value	R/W	Description
6	RIE	0	R/W	Receive Interrupt Enable Enables or disables generation of a receive-FIFO-data-full interrupt request when the RDF flag in SCSSR is set to 1. 0: Receive-FIFO-data-full interrupt request disabled* 1: Receive-FIFO-data-full interrupt request enabled Note: * The interrupt requests is cleared by reading 1 from the RDF flag, then clearing the flag to 0, or clearing the RIE bit to 0.
5	TE	0	R/W	Transmit Enable Enables or disables the start of serial transmission by the SCIF. 0: Transmission disabled 1: Transmission enabled* Note: * The serial mode register (SCSMR) and FIFO control register (SCFCR) settings must be made, the transmit format decided, and the transmit FIFO reset, before the TE bit is set to 1. If settings of above registers are changed with the TE bit enabled, the operation cannot be guaranteed.
4	RE	0	R/W	Receive Enable Enables or disables the start of serial reception by the SCIF. 0: Reception disabled*[1] 1: Reception enabled*[2] Notes: 1. Clearing the RE bit to 0 does not affect the DR, ER, BRK, RDF, FER, PER, and ORER flags, which retain their state. 2. The serial mode register (SCSMR) and FIFO control register (SCFCR) settings must be made, the receive format decided, and the receive FIFO reset, before the RE bit is set to 1. If settings of above registers are changed with the RE bit enabled, the operation cannot be guaranteed.
3, 2	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
1	CKE1	0	R/W	Clock Enable
0	CKE0	0	R/W	Select the SCIF clock source. The CKE1 and CKE0 bits must be set before determining the SCIF operating mode with SCSMR. 00: Internal clock/SCK pin functions as input pin (input signal ignored) 01: Internal clock/SCK pin functions as serial clock output*[1] 10: External clock/SCK pin functions as clock input*[2] 11: External clock/SCK pin functions as clock input*[2] When data is sampled by the on-chip baud rate generator, set bits CKE1 and CKE0 to B'00 (internal clock/SCK pin functions as input pin (input signal ignored)). When using the SCK pin as a port, set bits CKE1 and CKE0 to B'00. Notes: 1. In synchronous mode, a clock with a frequency equal to the bit rate is output. 2. In asynchronous mode, a clock with a sampling rate should be input. For example, when the sampling rate is 1/16, a clock with a frequency of 8 times the bit rate should be input. When an external clock is not input, set bits CKE1 and CKE0 to B'00 or B'01.

16.3.7 FIFO Error Count Register (SCFER)

SCFER is a 16-bit read-only register that indicates the number of receive errors (framing or parity error).

Bit	Bit Name	Initial Value	R/W	Description
15, 14	—	All 0	R	Reserved These bits are always read as 0.
13	PER5	0	R	Parity Error Count Indicates the number of data, in which parity errors are generated, in receive data stored in the receive FIFO data register (SCFRDR) in asynchronous mode. After setting the ER bit in SCSSR, the value of bits 13 to 8 indicates the number of parity error generated data. When all 64 bytes of receive data in SCFRDR have parity errors, the PER5 to PER0 bits indicate 0.
12	PER4	0	R	
11	PER3	0	R	
10	PER2	0	R	
9	PER1	0	R	
8	PER0	0	R	
7, 6	—	All 0	R	Reserved These bits are always read as 0.
5	FER5	0	R	Framing Error Count Indicates the number of data, in which framing errors are generated, in receive data stored in the receive FIFO data register (SCFRDR) in asynchronous mode. After setting the ER bit in SCSSR, the value of bits 5 to 0 indicates the number of framing error generated data. When all 64 bytes of receive data in SCFRDR have framing errors, the FER5 to FER0 bits indicate 0.
4	FER4	0	R	
3	FER3	0	R	
2	FER2	0	R	
1	FER1	0	R	
0	FER0	0	R	

16.3.8 Serial Status Register (SCSSR)

SCSSR is a 16-bit readable/writable register that indicates the SCIF status.

However, 1 cannot be written to the ORER, TSF, ER, TDFE, BRK, RDF, and DR flags. Also note that in order to clear these flags to 0, they must be read as 1 beforehand. The TEND, FER, and PER flags are read-only flags and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
15 to 10	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
9	ORER	0	R/(W)*	Overrun Error Flag Indicates that an overrun error occurred during reception. This bit is only valid in asynchronous mode. 0: Reception in progress, or reception has ended successfully*[1] [Clearing conditions] • Power-on reset or manual reset • When 0 is written to ORER after reading ORER = 1 1: An overrun error occurred during reception*[2] [Setting condition] When serial reception is completed while receive FIFO is full Notes: 1. The ORER flag is not affected and retains its previous state when the RE bit in SCSCR is cleared to 0. 2. The receive data prior to the overrun error is retained in SCFRDR, and the data received subsequently is lost. Serial reception cannot be continued while the ORER flag is set to 1.

Bit	Bit Name	Initial Value	R/W	Description
8	TSF	0	R/(W)*	Transmit Data Stop Flag Indicates that the number of transmit data matches the value of SCTDSR. 0: Number of transmit data does not match the value of SCTDSR [Clearing conditions] • Power-on reset or manual reset • When 0 is written to TSF after reading TSF = 1 1: Number of transmit data matches the value of SCTDSR
7	ER	0	R/(W)*	Receive Error Indicates that a framing error or parity error occurred during reception in asynchronous mode.*[1] 0: No framing error or parity error occurred during reception [Clearing conditions] • Power-on reset or manual reset • When 0 is written to ER after reading ER = 1 1: A framing error or parity error occurred during reception [Setting conditions] • When the SCIF checks whether the stop bit at the end of the receive data is 1 when reception ends, and the stop bit is 0*[2] • When, in reception, the number of 1-bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/E bit in SCSMR Notes: 1. The ER flag is not affected and retains its previous state when the RE bit in SCSCR is cleared to 0. When a receive error occurs, the receive data is still transferred to SCFRDR, and reception continues. The FER and PER bits in SCSSR can be used to determine whether there is a receive error in the data read from SCFRDR. 2. When the stop length is two bits, only the first stop bit is checked for a value of 1; the second stop bit is not checked.

Bit	Bit Name	Initial Value	R/W	Description
6	TEND	1	R	Transmit End Indicates that there is no valid data in SCFTDR when the last bit of the transmit character is sent, and transmission has been ended. 0: Transmission is in progress [Clearing condition] When data is written to SCFTDR 1: Transmission has been ended [Setting condition] When there is no transmit data in SCFTDR on transmission of a 1-byte serial transmit character
5	TDFE	1	R/(W)*	Transmit FIFO Data Empty Indicates that data has been transferred from SCFTDR to SCTSR, the number of data bytes in SCFTDR has fallen to or below the transmit trigger data number set by bits TTRG1 and TTRG0 in the FIFO control register (SCFCR), and new transmit data can be written to SCFTDR. 0: A number of transmit data bytes exceeding the transmit trigger set number have been written to SCFTDR [Clearing condition] When transmit data exceeding the transmit trigger set number is written to SCFTDR, and 0 is written to TDFE after reading TDFE = 1 1: The number of transmit data bytes in SCFTDR does not exceed the transmit trigger set number [Setting conditions] • Power-on reset or manual reset • When the number of SCFTDR transmit data bytes falls to or below the transmit trigger set number as the result of a transmit operation*[1] Note: 1. As SCFTDR is a 64-byte FIFO register, the maximum number of bytes that can be written when TDFE = 1 is 64 – (transmit trigger set number). Data written in excess of this will be ignored. The number of data bytes in SCFTDR is indicated by SCFDR.

Bit	Bit Name	Initial Value	R/W	Description
4	BRK	0	R/(W)*	Break Detect Indicates that a receive data break signal has been detected in asynchronous mode. 0: A break signal has not been received [Clearing conditions] • Power-on reset or manual reset • When 0 is written to BRK after reading BRK = 1 1: A break signal has been received*[1] [Setting condition] When data with a framing error is received, followed by the space 0 level (low level) for at least one frame length Note: 1. When a break is detected, the receive data (H'00) following detection is not transferred to SCFRDR. When the break ends and the receive signal returns to mark 1, receive data transfer is resumed.
3	FER	0	R	Framing Error Indicates a framing error in the data read from SCFRDR in asynchronous mode. 0: There is no framing error in the receive data read from SCFRDR [Clearing conditions] • Power-on reset or manual reset • When there is no framing error in SCFRDR read data 1: There is a framing error in the receive data read from SCFRDR [Setting condition] When there is a framing error in SCFRDR read data

Bit	Bit Name	Initial Value	R/W	Description
2	PER	0	R	Parity Error Indicates a parity error in the data read from SCFRDR in asynchronous mode. 0: There is no parity error in the receive data read from SCFRDR [Clearing conditions] • Power-on reset or manual reset • When there is no parity error in SCFRDR read data 1: There is a parity error in the receive data read from SCFRDR [Setting condition] When there is a parity error in SCFRDR read data
1	RDF	0	R/(W)*	Receive FIFO Data Full Indicates that the received data has been transferred from SCRSR to SCFRDR, and the number of receive data bytes in SCFRDR is equal to or greater than the receive trigger number set by bits RTRG1 and RTRG0 in the FIFO control register (SCFCR). 0: The number of receive data bytes in SCFRDR is less than the receive trigger set number [Clearing conditions] • Power-on reset or manual reset • When SCFRDR is read until the number of receive data bytes in SCFRDR falls below the receive trigger set number, and 0 is written to RDF after reading RDF = 1 1: The number of receive data bytes in SCFRDR is equal to or greater than the receive trigger set number [Setting condition] When SCFRDR contains at least the receive trigger set number of receive data bytes*[1] Note:1. SCFRDR is a 64-byte FIFO register. When RDF = 1, at least the receive trigger set number of data bytes can be read. If data is read when SCFRDR is empty, an undefined value will be returned. The number of receive data bytes in SCFRDR is indicated by the lower bits of SCFDR.

Bit	Bit Name	Initial Value	R/W	Description
0	DR	0	R/(W)*	Receive Data Ready Indicates that there are fewer than the receive trigger set number of data bytes in SCFRDR and no further data will arrive in asynchronous mode. 0: Reception is in progress or has ended successfully and there is no receive data left in SCFRDR [Clearing conditions] • Power-on reset or manual reset • When all the receive data in SCFRDR has been read, and 0 is written to DR after reading DR = 1 1: No further receive data has arrived [Setting condition] When SCFRDR contains fewer than the receive trigger set number of receive data bytes and no further data will arrive.*[1] Note: 1. The DR bit is set 15etu after the last data is received at a sampling rate of 1/16 regardless of the setting of the sampling control bits in SCSMR. etu: Elementary time unit (time for transfer of one bit)

Note: * Only 0 can be written for clearing the flags.

16.3.9 Bit Rate Register (SCBRR)

SCBRR is an 8-bit readable/writable register that sets the serial transfer bit rate in accordance with the baud rate generator operating clock selected by bits CKS1 and CKS0 in SCSMR.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	SCBRD7 to SCBRD0	H'FF	R/W	Bit Rate Setting

The SCBRR setting is found from the following equation.

Asynchronous Mode:

1. When sampling rate is 1/16

$$N = \frac{P\phi}{32 \times 2^{2n-1} \times B} \times 10^6 - 1$$

2. When sampling rate is 1/5

$$N = \frac{P\phi}{10 \times 2^{2n-1} \times B} \times 10^6 - 1$$

3. When sampling rate is 1/11

$$N = \frac{P\phi}{22 \times 2^{2n-1} \times B} \times 10^6 - 1$$

4. When sampling rate is 1/13

$$N = \frac{P\phi}{26 \times 2^{2n-1} \times B} \times 10^6 - 1$$

5. When sampling rate is 1/27

$$N = \frac{P\phi}{54 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Clock Synchronous Mode:

$$N = \frac{P\phi}{4 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Where B: Bit rate (bits/s)
N: SCBRR setting for baud rate generator
Asynchronous mode: $0 \leq N \leq 255$
Clock synchronous mode: $1 \leq N \leq 255$
Pϕ: Peripheral module operating frequency (MHz)
n: Baud rate generator input clock (n = 0 to 3)
(See the table below for the relation between n and the clock.)

		SCSMR Setting	
n	**Clock**	**CKS1**	**CKS0**
0	Pϕ	0	0
1	Pϕ/4	0	1
2	Pϕ/16	1	0
3	Pϕ/64	1	1

The bit rate error in asynchronous mode is found from the following equation:

1. When sampling rate is 1/16

$$\text{Error (\%)} = \left(\frac{P\phi \times 10^6}{(1+N) \times B \times 32 \times 2^{2n-1}} - 1 \right) \times 100$$

2. When sampling rate is 1/5

$$\text{Error (\%)} = \left(\frac{P\phi \times 10^6}{(1+N) \times B \times 10 \times 2^{2n-1}} - 1 \right) \times 100$$

3. When sampling rate is 1/11

$$\text{Error (\%)} = \left(\frac{P\phi \times 10^6}{(1+N) \times B \times 22 \times 2^{2n-1}} - 1 \right) \times 100$$

4. When sampling rate is 1/13

$$\text{Error (\%)} = \left(\frac{P\phi \times 10^6}{(1+N) \times B \times 26 \times 2^{2n-1}} - 1 \right) \times 100$$

5. When sampling rate is 1/27

$$\text{Error}\ (\%) = \left(\frac{P\phi \times 10^6}{(1+N) \times B \times 54 \times 2^{2n-1}} - 1 \right) \times 100$$

16.3.10 FIFO Control Register (SCFCR)

SCFCR is a 16-bit readable/writable register that resets the data count and sets the trigger data number for the transmit and receive FIFO registers, and also contains a loopback test enable bit.

Bit	Bit Name	Initial Value	R/W	Description
15	TSE	0	R/W	Transmit Data Stop Enable Enables or disables the transmit data stop function. This function is enabled only in asynchronous mode. Since this function is not supported in clock synchronous mode, clear this bit to 0 in clock synchronous mode. 0: Transmit data stop function disabled 1: Transmit data stop function enabled
14	TCRST	0	R/W	Transmit Count Reset Clears the transmit count to 0. This bit is valid only when the transmit data stop function is used. 0: Transmit count reset disabled* 1: Transmit count reset enabled (clearing to 0) Note:* The transmit count is reset (clearing to 0) is performed in power-on reset or manual reset.
13 to 11	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
10	RSTRG2	0	R/W	$\overline{\text{RTS}}$ Output Active Trigger
9	RSTRG1	0	R/W	The $\overline{\text{RTS}}$ signal goes high when the number of receive data bytes in SCFRDR is equal to or greater than the trigger set number shown in below.
8	RSTRG0	0	R/W	000: 63 001: 1 010: 8 011: 16 100: 32 101: 48 110: 54 111: 60
7	RTRG1	0	R/W	Receive FIFO Data Number Trigger
6	RTRG0	0	R/W	Set the number of receive data bytes that sets the receive data full (RDF) flag in the serial status register (SCSSR). The RDF flag is set when the number of receive data bytes in SCFRDR is equal to or greater than the trigger set number shown in below. 00: 1 01: 16 10: 32 11: 48
5	TTRG1	0	R/W	Transmit FIFO Data Number Trigger
4	TTRG0	0	R/W	Set the number of remaining transmit data bytes that sets the transmit FIFO data register empty (TDFE) flag in the serial status register (SCSSR). The TDFE flag is set when, as the result of a transmit operation, the number of transmit data bytes in the transmit FIFO data register (SCFTDR) falls to or below the trigger set number shown in below. 00: 32 (32) 01: 16 (48) 10: 2 (62) 11: 0 (64) Note: The values in parentheses are the number of empty bytes in SCFTDR when the flag is set.

Bit	Bit Name	Initial Value	R/W	Description
3	MCE	0	R/W	Modem Control Enable Enables modem control signals $\overline{\text{CTS}}$ and $\overline{\text{RTS}}$. This setting is only valid in asynchronous mode. 0: Modem signal disabled* 1: Modem signal enabled Note: * $\overline{\text{CTS}}$ is disabled regardless of the input value, and $\overline{\text{RTS}}$ is also fixed at 0.
2	TFRST	0	R/W	Transmit FIFO Data Register Reset Invalidates the transmit data in the transmit FIFO data register and resets it to the empty state. 0: Reset operation disabled* 1: Reset operation enabled Note: * A reset operation is performed in the event of a power-on reset or manual reset.
1	RFRST	0	R/W	Receive FIFO Data Register Reset Invalidates the receive data in the receive FIFO data register and resets it to the empty state. 0: Reset operation disabled* 1: Reset operation enabled Note: * A reset operation is performed in the event of a power-on reset or manual reset.
0	LOOP	0	R/W	Loopback Test Internally connects the transmit output pin (TxD) and receive input pin (RxD), and $\overline{\text{RTS}}$ pin and $\overline{\text{CTS}}$ pin, enabling loopback testing. 0: Loopback test disabled 1: Loopback test enabled

16.3.11 FIFO Data Count Register (SCFDR)

SCFDR is a 16-bit read-only register that indicates the number of data bytes stored in the transmit FIFO data register (SCFTDR) and receive FIFO data register (SCFRDR).

Bits 14 to 8 show the number of transmit data bytes in SCFTDR, and bits 6 to 0 show the number of receive data bytes in SCFRDR.

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved This bit is always read as 0.
14	T6	0	R	These bits show the number of untransmitted data bytes in SCFTDR. A value of H'00 means that there is no transmit data, and a value of H'40 means that SCFTDR is full of transmit data.
13	T5	0	R	
12	T4	0	R	
11	T3	0	R	
10	T2	0	R	
9	T1	0	R	
8	T0	0	R	
7	—	0	R	Reserved This bit is always read as 0.
6	R6	0	R	These bits show the number of receive data bytes in SCFRDR. A value of H'00 means that there is no receive data, and a value of H'40 means that SCFRDR is full of receive data.
5	R5	0	R	
4	R4	0	R	
3	R3	0	R	
2	R2	0	R	
1	R1	0	R	
0	R0	0	R	

16.3.12 Transmit Data Stop Register (SCTDSR)

SCTDSR is an 8-bit readable/writable register that sets the number of transmit data bytes. SCTDSR is valid only when the TSE bit in the FIFO control register (SCFCR) is enabled. Transmit operation is stopped when the number of data bytes set in SCTDSR is transmitted. The setting value should be H'00 (one byte) to H'FF (256 bytes). This function is only enabled in asynchronous mode.

SCTDSR is initialized to H'FF.

16.4 Operation

16.4.1 Overview

The SCIF can carry out serial communication in asynchronous mode, in which synchronization is achieved character by character, and in clock synchronous mode, in which synchronization is achieved with clock pulses.

64-stage FIFO buffers are provided for both transmission and reception, reducing the CPU overhead and enabling fast, continuous communication to be performed.

16.4.2 Asynchronous Mode

The asynchronous mode is described below.

The transfer format is selected using the serial mode register (SCSMR), as shown in table 16.2. The SCIF clock source is determined by the CKE1 and CKE0 bits in the serial control register (SCSCR).

- Data length: Choice of seven or eight bits
- Choice of parity addition and addition of one or two stop bits (the combination of these parameters determines the transfer format and character length)
- Detection of framing errors, parity errors, overrun errors, receive-FIFO-data-full state, receive-data-ready state, and breaks, during reception
- Indication of the number of data bytes stored in the transmit and receive FIFO registers
- Choice of internal or external clock as the SCIF clock source
  - When internal clock is selected: the SCIF operates on the baud rate generator clock.
  - When external clock is selected: A clock must be input according to the sampling rate. For example, when the sampling rate is 1/16, a clock with a frequency of 8 times the bit rate must be input (the on-chip baud rate generator is not used.)

Table 16.2 SCSMR Settings for Serial Transfer Format Selection

SCSMR Settings				SCIF Transfer Format		
Bit 6: CHR	**Bit 5: PE**	**Bit 3: STOP**	**Mode**	**Data Length**	**Parity Bit**	**Stop Bit Length**
0	0	0	Asynchronous mode	8-bit data	No	1 bit
		1				2 bits
	1	0			Yes	1 bit
		1				2 bits
1	0	0		7-bit data	No	1 bit
		1				2 bits
	1	0			Yes	1 bit
		1				2 bits

16.4.3 Serial Operation in Asynchronous Mode

Data Transfer Format: Table 16.3 shows the transfer formats that can be used in asynchronous mode. Any of eight transfer formats can be selected according to the SCSMR settings.

Table 16.3 Serial Transfer Formats

SCSMR 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		
		1	S	8-bit data								STOP	STOP	
	1	0	S	8-bit data								P	STOP	
		1	S	8-bit data								P	STOP	STOP
1	0	0	S	7-bit data							STOP			
		1	S	7-bit data							STOP	STOP		
	1	0	S	7-bit data							P	STOP		
		1	S	7-bit data							P	STOP	STOP	

S: Start bit
STOP: Stop bit
P: Parity bit

Clock: Either an internal clock generated by the on-chip baud rate generator or an external clock input at the SCK pin can be selected as the serial clock for the SCIF, according to the setting of the CKE1 and CKE0 bits in SCSCR.

When an external clock is input at the SCK pin, a clock must be input according to the sampling rate. For example, when the sampling rate is 1/16, a clock with a frequency of 8 times the bit rate must be input.

Data Transfer Operations:

1. SCIF Initialization

   Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR to 0, then initialize the SCIF as described below.

   When the transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the transmit shift register (SCTSR) is initialized. Note that clearing the TE and RE bits to 0 does not change the contents of SCSSR, SCFTDR, or SCFRDR. The TE bit should be cleared to 0 after all transmit data has been sent and the TEND bit in SCSSR has been set to 1. Clearing to 0 can also be performed during transmission, but the data being transmitted will go to the high-impedance state after the clearance. Before setting TE to 1 again to start transmission, the TFRST bit in SCFCR should first be set to 1 to reset SCFTDR.

   When an external clock is used, the clock should not be stopped during operation, including initialization, since operation will be unreliable in this case.

Figure 16.2 shows a sample SCIF initialization flowchart.

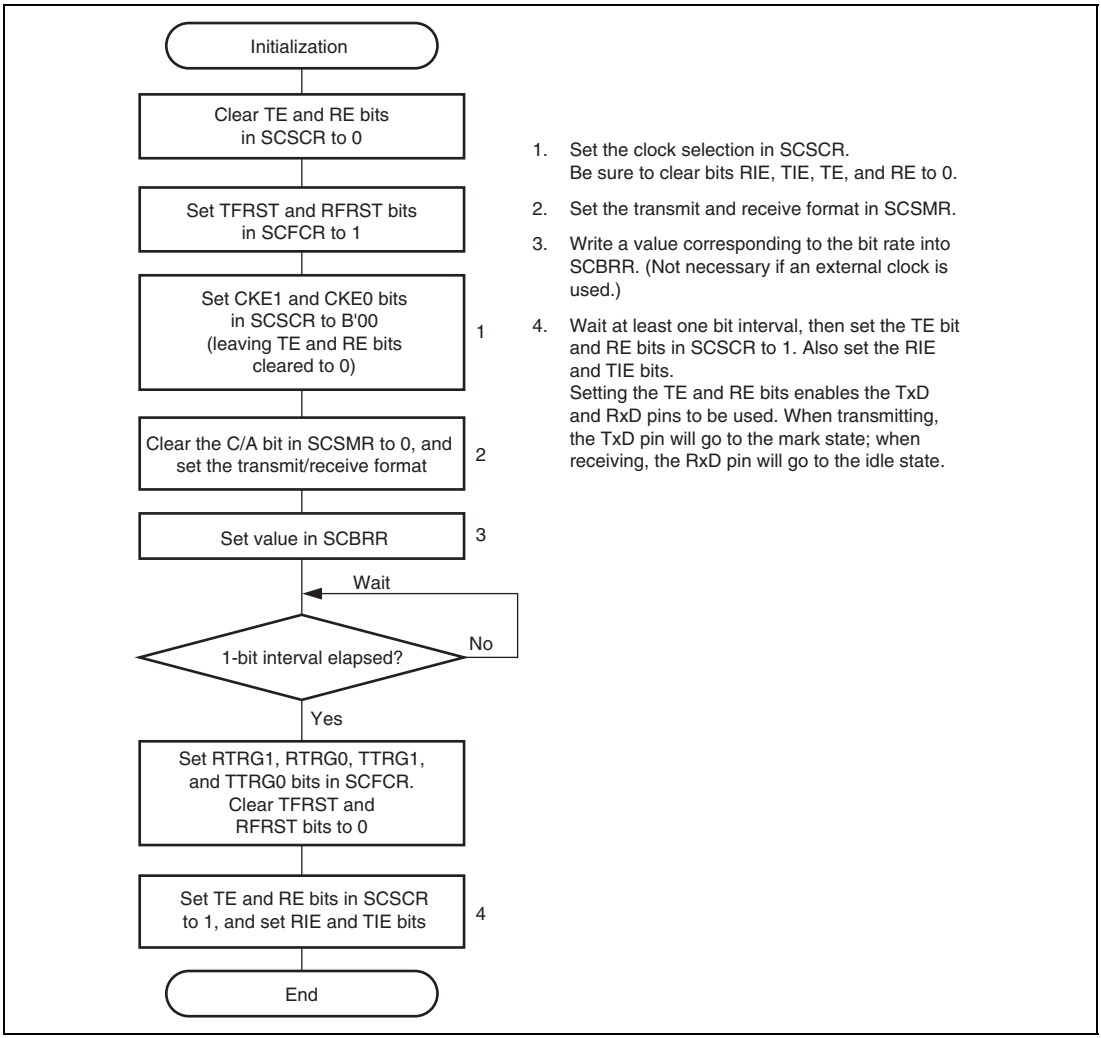
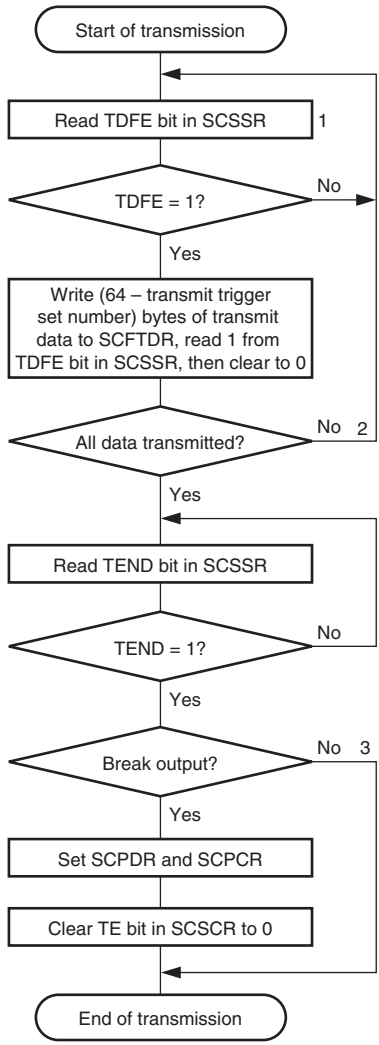


Figure 16.2 Sample SCIF Initialization Flowchart

2. Serial Data Transmission

Figure 16.3 shows a sample flowchart for serial transmission.

Use the following procedure for serial data transmission after enabling the SCIF for transmission.



1. SCIF status check and transmit data write:
   Read the serial status register (SCSSR) and check that the TDFE flag is set to 1, then write transmit data to SCFTDR, read 1 from the TDFE flag, then clear the flag to 0.
   The number of data bytes that can be written is 64 – (transmit trigger set number).
2. Serial transmission continuation procedure:
   To continue serial transmission, read 1 from the TDFE flag to confirm that writing is possible, then write data to SCFTDR, and then clear the TDFE bit to 0.
3. Break output at the end of serial transmission:
   To output a break in serial transmission, set the port SC data register (SCPDR) and port SC control register (SCPCR), then clear the TE bit in SCSCR to 0.
   In steps 1 and 2, it is possible to ascertain the number of data bytes that can be written from the number of transmit data bytes in SCFTDR indicated by the upper 8 bits of SCFDR.

Figure 16.3 Sample Serial Transmission Flowchart

In serial transmission, the SCIF operates as described below.

A. When data is written into SCFTDR, the SCIF transfers the data from SCFTDR to SCTSR and starts transmitting. Confirm that the TDFE flag in the serial status register (SCSSR) is set to 1 before writing transmit data to SCFTDR. The number of data bytes that can be written is at least 64 – (transmit trigger set number).

B. When data is transferred from SCFTDR to SCTSR and transmission is started, consecutive transmit operations are performed until there is no transmit data left in SCFTDR. When the number of transmit data bytes in SCFTDR falls to or below the transmit trigger number set in the FIFO control register (SCFCR), the TDFE flag is set. If the TIE bit in SCSCR is set to 1 at this time, a transmit-FIFO-data-empty interrupt (TXI) request is generated.

   When the transmit data stop function is used and the number of data bytes set in the transmit data stop register (SCTDSR) is matched, transmit operation is stopped, and the TSF flag in the serial status register (SCSSR) is set. If the TSIE bit in the serial control register (SCSCR) is set to 1, a transmit-data-stop-interrupt (TDI) request is generated. The vectors of transmit-FIFO-data-empty and transmit-data-stop interrupts are the same.

The serial transmit data is sent from the TxD pin in the following order.

   a. Start bit: One 0-bit is output.
   b. Transmit data: 8-bit or 7-bit data is output in LSB-first order.
   c. Parity bit: One parity bit (even or odd parity) is output.
      A format in which a parity bit is not output can also be selected.
   d. Stop bit(s): One or two 1-bits (stop bits) are output.
   e. Mark state: 1 is output continuously until the start bit that starts the next transmission is sent.

C. The SCIF checks the SCFTDR transmit data at the timing for sending the stop bit. If data is present, the data is transferred from SCFTDR to SCTSR, the stop bit is sent, and then serial transmission of the next frame is started.

   If there is no transmit data, the TEND flag in SCSSR is set to 1, the stop bit is sent, and then the line goes to the mark state in which 1 is output.

Figure 16.4 shows an example of the operation for transmission in asynchronous mode.

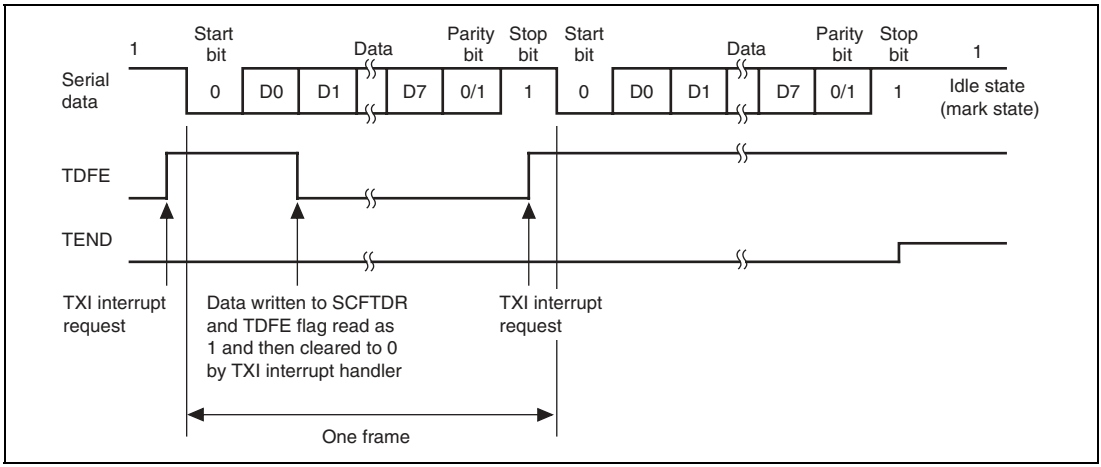


Figure 16.4 Example of Transmit Operation (Example with 8-Bit Data, Parity, One Stop Bit)

- Transmit Data Stop Function

  When a value in the SCTDSR register is matched with the number of transmit data bytes, this function stops the transmit operation. Interrupts can be generated and the DMAC can be activated by setting the TSIE bit (interrupt enable bit).

  Figure 16.5 shows an example of operation for the transmit data stop function.

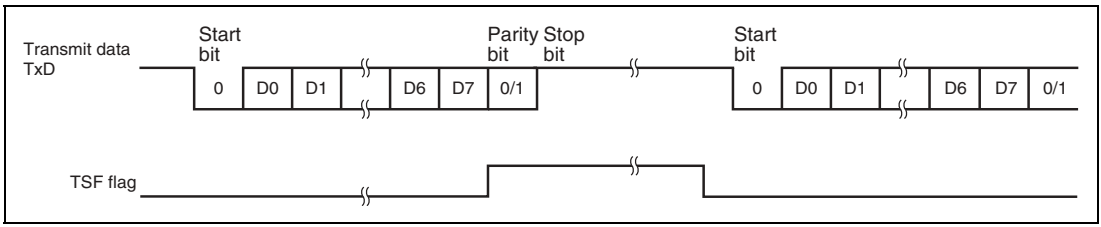


Figure 16.5 Example of Transmit Data Stop Function

Figure 16.6 shows a flowchart for the transmit data stop function.

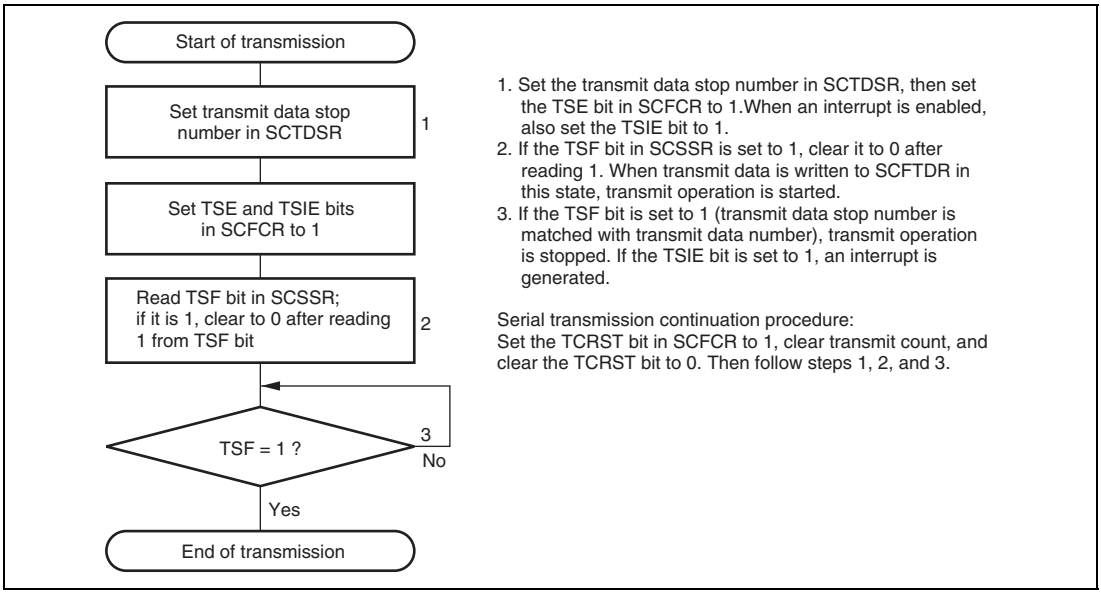
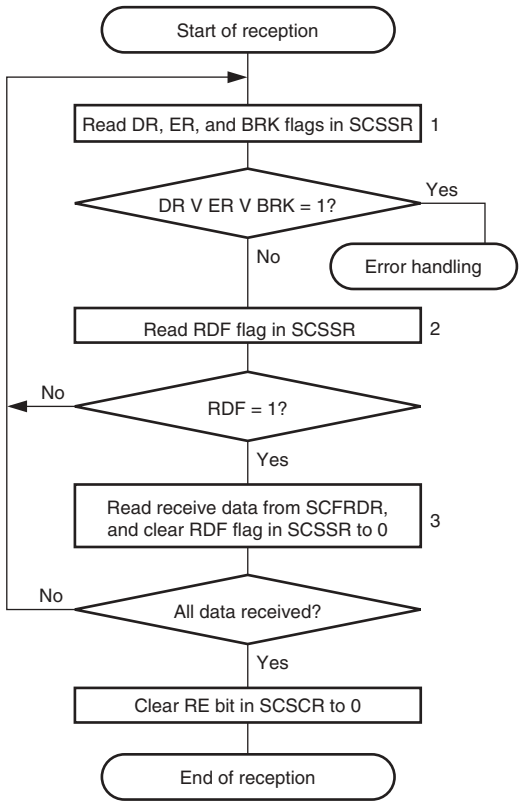


Figure 16.6 Transmit Data Stop Function Flowchart

3. Serial Data Reception

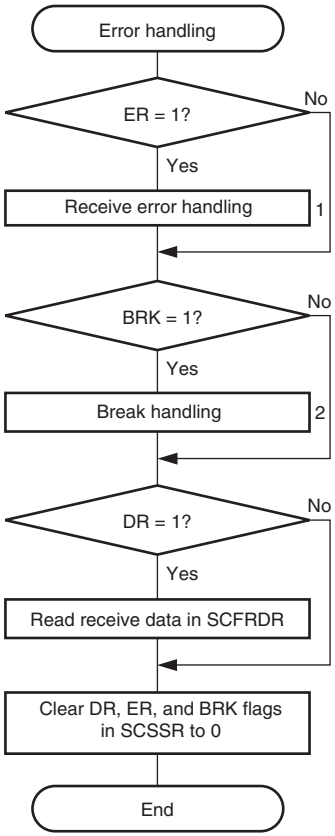
Figures 16.7 and 16.8 show sample flowcharts for serial reception.

Use the following procedure for serial data reception after enabling the SCIF for reception.



1. Receive error handling and break detection:
   Read the DR, ER, and BRK flags in SCSSR to identify any error, perform the appropriate error handling, then clear the DR, ER, and BRK flags to 0. In the case of a framing error, a break can also be detected by reading the value of the RxD pin.
2. SCIF status check and receive data read:
   Read SCSSR and check that RDF = 1, then read the receive data in SCFRDR, read 1 from the RDF flag, and then clear the RDF flag to 0. The transition of the RDF flag from 0 to 1 can also be identified by an RXI interrupt.
3. Serial reception continuation procedure:
   To continue serial reception, read at least the receive trigger set number of data bytes from SCFRDR, read 1 from the RDF flag, and then clear the RDF flag to 0. The number of receive data bytes in SCFRDR can be ascertained by reading the lower bits of SCFDR.

Figure 16.7 Sample Serial Reception Flowchart (1)



1 Whether a framing error or parity error has occurred in the receive data read from SCFRDR can be ascertained from the FER and PER bits in SCSSR.

2 When a break signal is received, receive data is not transferred to SCFRDR while the BRK flag is set. However, note that the last data in SCFRDR is H'00 and the break data in which a framing error occurred is stored.

Figure 16.8 Sample Serial Reception Flowchart (2)

In serial reception, the SCIF operates as described below.

A. The SCIF monitors the communication line, and if 0 of a start bit is detected, performs internal synchronization and starts reception.

B. The received data is stored in SCRSR in LSB-to-MSB order.

C. The parity bit and stop bit are received.

After receiving these bits, the SCIF carries out the following checks.

a. Stop bit check: the SCIF checks whether the stop bit is 1. If there are two stop bits, only the first is checked.

b. The SCIF checks whether receive data can be transferred from the receive shift register (SCRSR) to SCFRDR.

c. Break check: the SCIF checks that the BRK flag is 0, indicating that the break state is not set.

If all the above checks are passed, the receive data is stored in SCFRDR.

Note: Reception continues when a receive error (a framing error or parity error) occurs.

D. If the RIE bit in SCSCR is set to 1 when the RDF flag changes to 1, a receive-FIFO-data-full interrupt (RXI) request is generated.
If the ERIE bit in SCSCR is set to 1 when the ER flag changes to 1, a receive-error interrupt (ERI) request is generated.
If the BRIE bit in SCSCR is set to 1 when the BRK flag changes to 1, a break reception interrupt (BRI) request is generated.
If the DRIE bit in SCSCR is set to 1 when the DR flag changes to 1, a receive-data-ready interrupt (DRI) request is generated.
The vectors of interrupts generated by each factor are the same.

Figure 16.9 shows an example of the operation for reception in asynchronous mode.

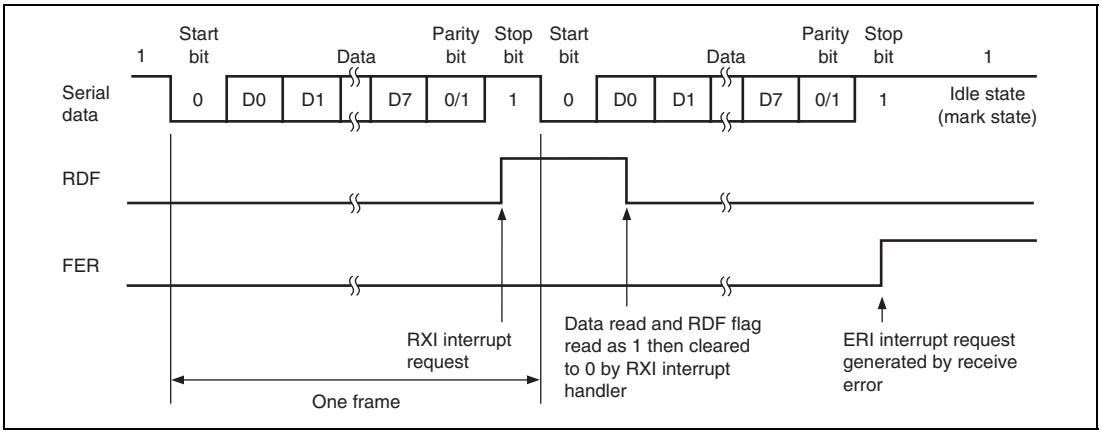


Figure 16.9 Example of SCIF Receive Operation (Example with 8-Bit Data, Parity, One Stop Bit)

4. Transmit/receive when using the Modem Function

   When using a modem function, transmission can be stopped and started again according to the $\overline{\text{CTS}}$ input value. When the $\overline{\text{CTS}}$ is set to 1 during transmission, the data enters a mark state after transmitting one frame. When $\overline{\text{CTS}}$ is set to 0, the next transmit data is output starting with a start bit.

   Figure 16.10 shows an example of operation for the $\overline{\text{CTS}}$ control.

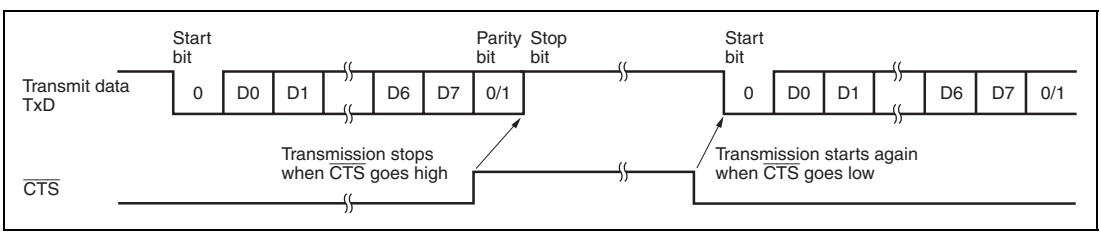


Figure 16.10 $\overline{\text{CTS}}$ Control Operation

When using a modem function and the receive FIFO (SCFRDR) is at least the number of the $\overline{\text{RTS}}$ output trigger, the $\overline{\text{RTS}}$ signal goes high.

Figure 16.11 shows an example of operation for the $\overline{\text{RTS}}$ control.

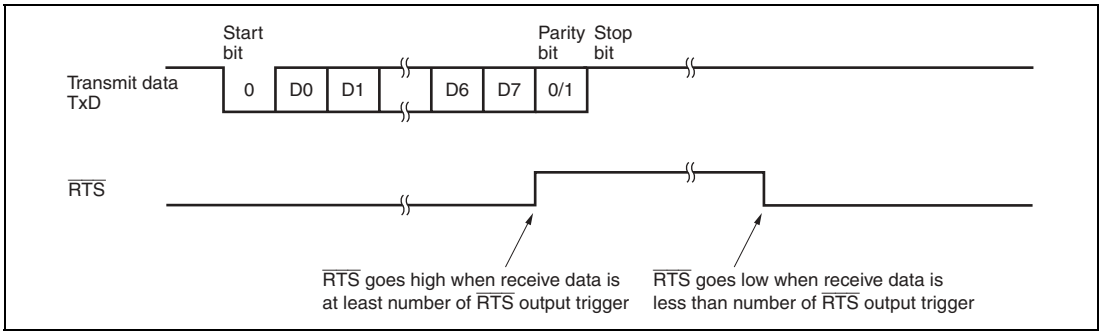


Figure 16.11 $\overline{\text{RTS}}$ Control Operation

16.4.4 Clock Synchronous Mode

The clock synchronous mode is described below.

64-stage FIFO buffers are provided for both transmission and reception, reducing the CPU overhead and enabling fast, continuous communication to be performed.

The operating clock source is selected using the serial mode register (SCSMR). The SCIF clock source is determined by the CKE1 and CKE0 bits in the serial control register (SCSCR).

- Transmit/receive format: Fixed 8-bit data
- Indication of the number of data bytes stored in the transmit and receive FIFO registers
- Internal clock or external clock used as the SCIF clock source

  When the internal clock is selected:
  The SCIF operates on the baud rate generator clock and outputs a serial clock from SCK pin.

  When the external clock is selected:
  The SCIF operates on the external clock input through the SCK pin.

16.4.5 Serial Operation in Clock Synchronous Mode

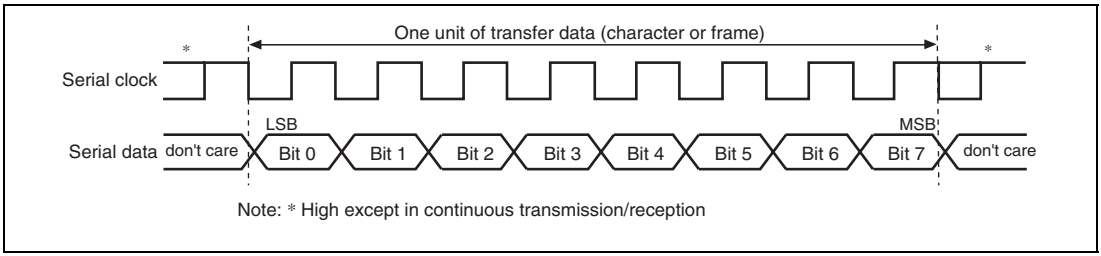


Figure 16.12 Data Format in Clock Synchronous Communication

In clock synchronous serial communication, data on the communication line is output from a falling edge of the serial clock to the next falling edge. Data is guaranteed valid at the rising edge of the serial clock.

In serial communication, each character is output starting with the LSB and ending with the MSB. After the MSB is output, the communication line remains in the state of the MSB.

In clock synchronous mode, the SCIF receives data in synchronization with the rising edge of the serial clock.

Data Transfer Format: A fixed 8-bit data format is used. No parity or multiprocessor bits are added.

Clock: An internal clock generated by the on-chip baud rate generator or an external clock input through the SCK pin can be selected as the serial clock for the SCIF, according to the setting of the CKE1 and CKE0 bits in SCSCR.

Eight serial clock pulses are output in the transfer of one character, and when no transmission/reception is performed, the clock is fixed high. However, when the operation mode is reception only, the synchronous clock output continues while the RE bit is set to 1. To fix the clock high every time one character is transferred, write to the transmit FIFO data register (SCFTDR) the same number of dummy data bytes as the data bytes to be received and set the TE and RE bits to 1 at the same time to transmit the dummy data. When the specified number of data bytes are transmitted, the clock is fixed high.

Data Transfer Operations:

1. SCIF Initialization

   Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR to 0, then initialize the SCIF as described below.

   When the clock source, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the transmit shift register (SCTSR) is initialized. Note that clearing the TE and RE bits to 0 does not change the contents of SCSSR, SCFTDR, or SCFRDR. The TE bit should be cleared to 0 after all transmit data has been sent and the TEND bit in SCSSR has been set to 1. The TE bit should not be cleared to 0 during transmission; if attempted, the TxD pin will go to the high-impedance state. Before setting TE to 1 again to start transmission, the TFRST bit in SCFCR should first be set to 1 to reset SCFTDR.

   Figure 16.13 shows sample SCIF initialization flowcharts.

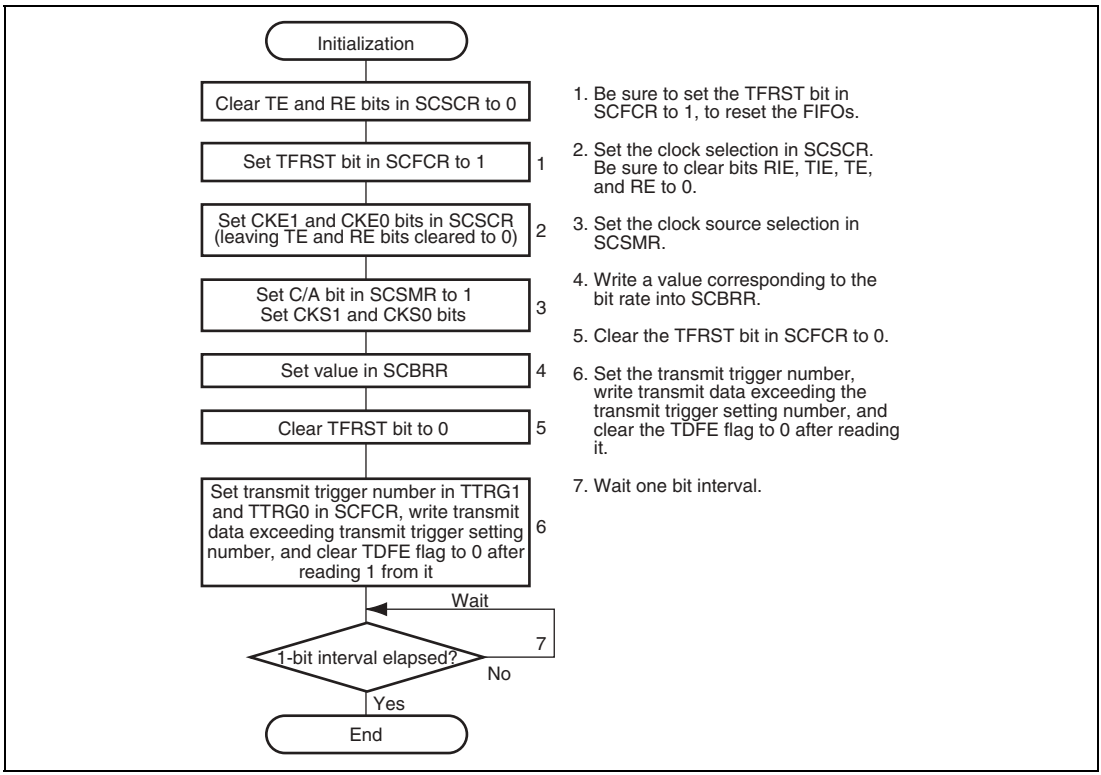


Figure 16.13 Sample SCIF Initialization Flowchart (1) (Transmission)

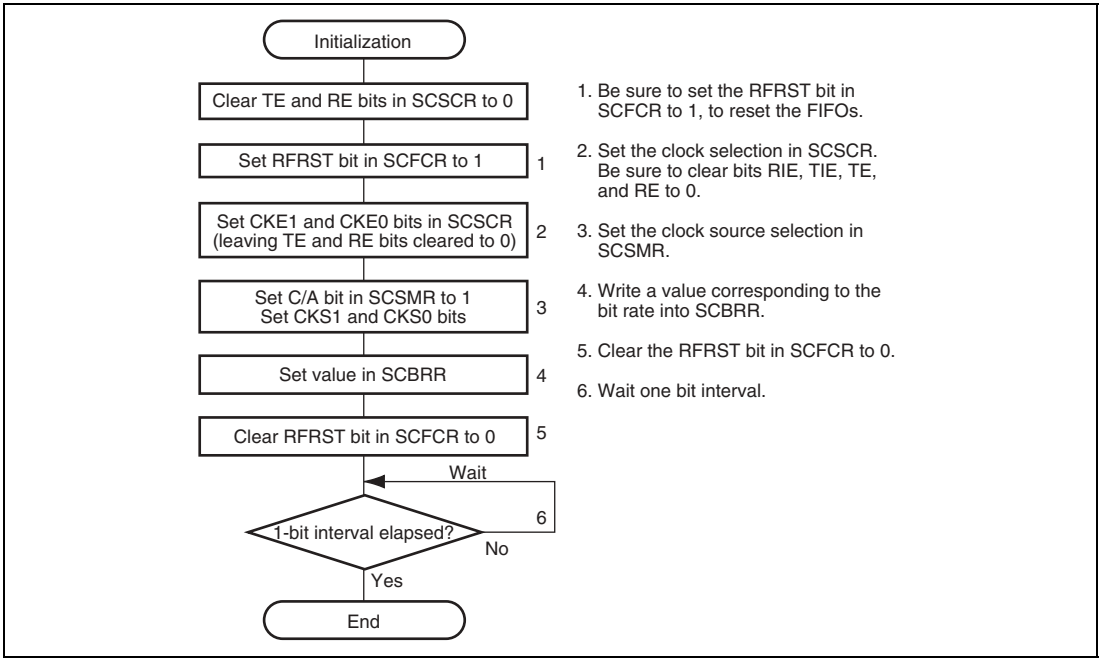


Figure 16.13 Sample SCIF Initialization Flowchart (2) (Reception)

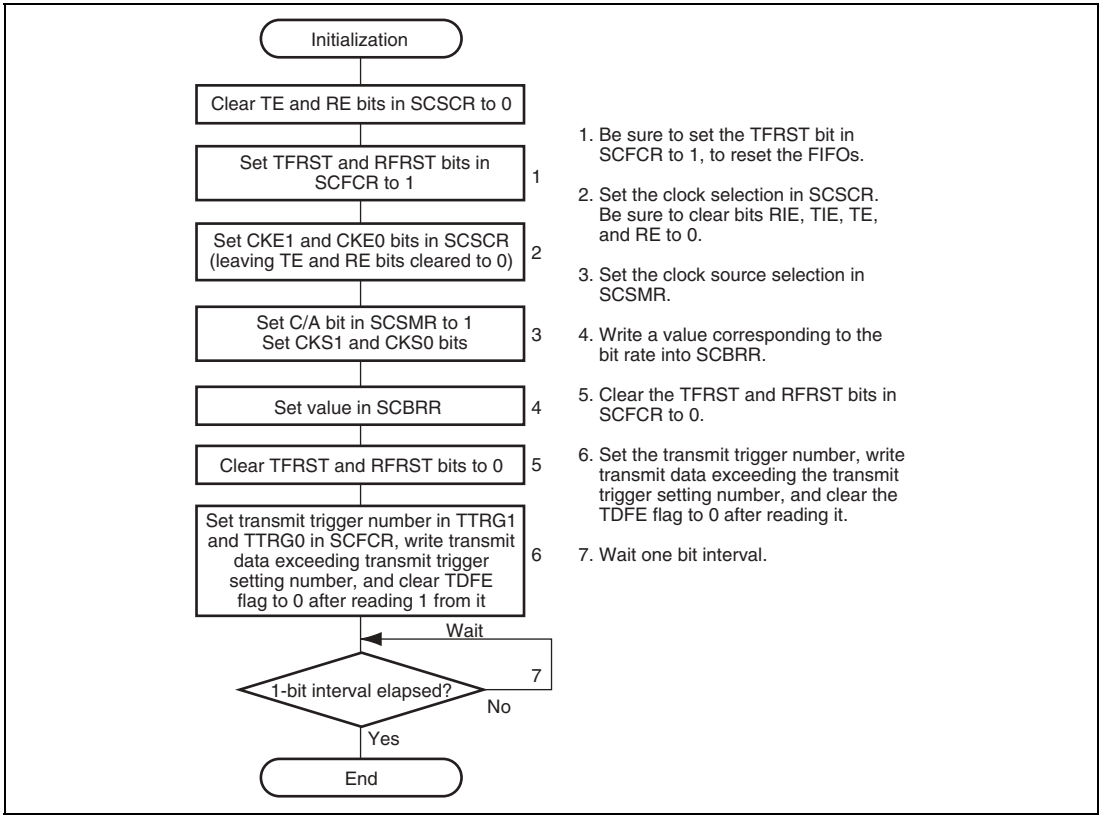


Figure 16.13 Sample SCIF Initialization Flowchart (3)
(Simultaneous Transmission and Reception)

2. Serial Data Transmission

Figure 16.14 shows sample flowcharts for serial transmission.

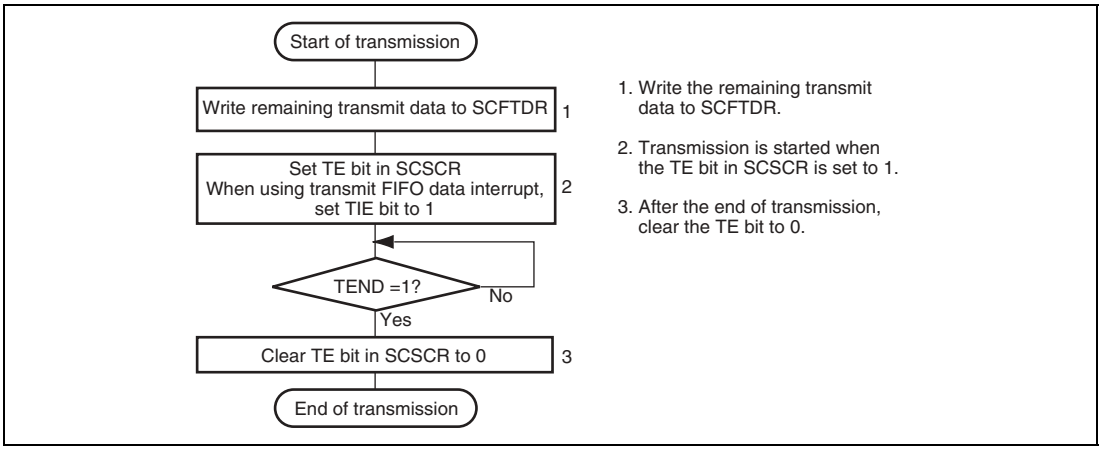


Figure 16.14 Sample Serial Transmission Flowchart (1)
(First Transmission after Initialization)

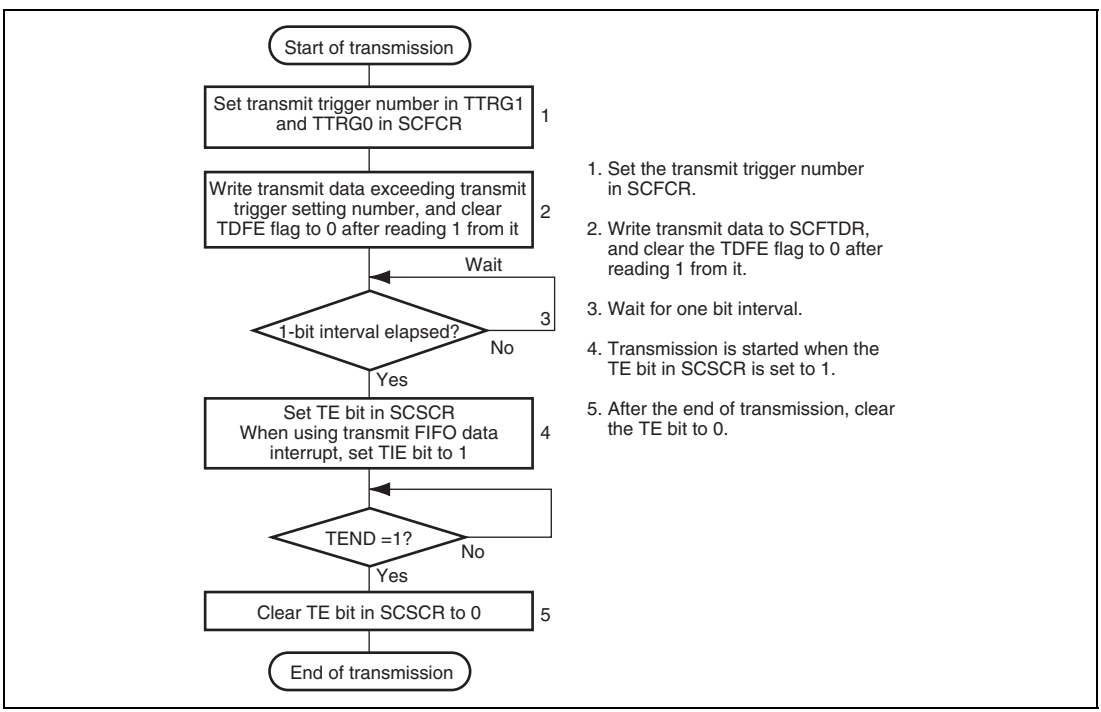


Figure 16.14 Sample Serial Transmission Flowchart (2)
(Second and Subsequent Transmission)

3. Serial Data Reception

Figure 16.15 shows sample flowcharts for serial reception.

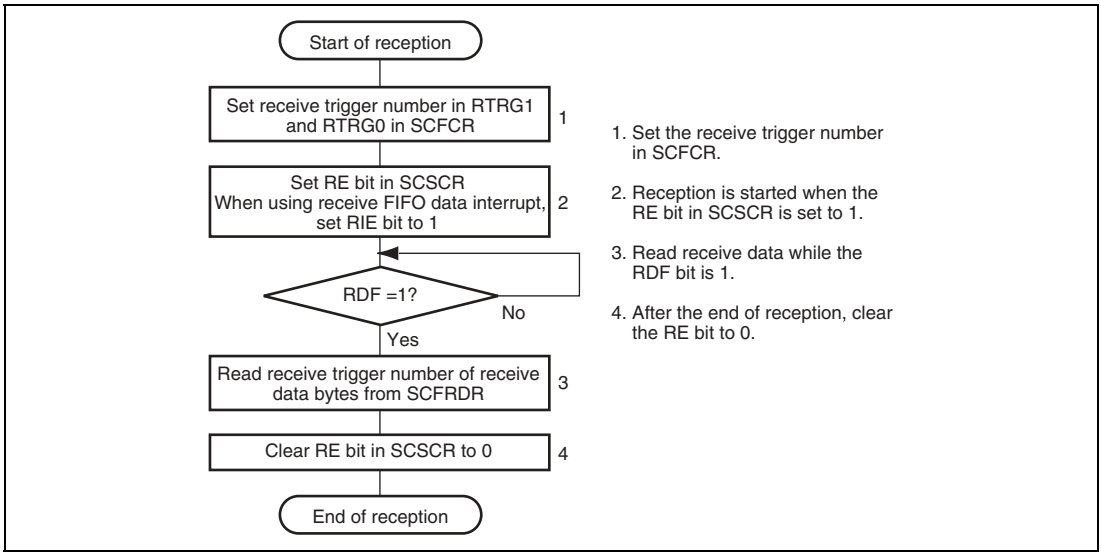


Figure 16.15 Sample Serial Reception Flowchart (1)
(First Reception after Initialization)

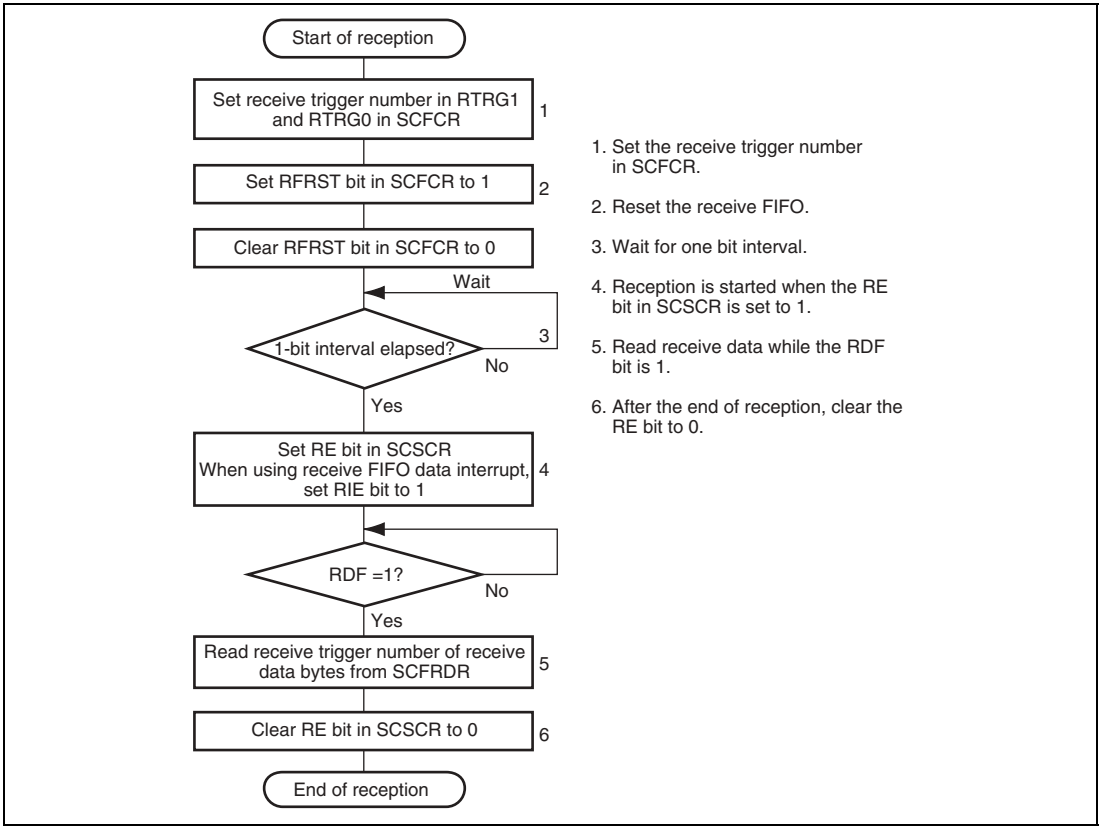


Figure 16.15 Sample Serial Reception Flowchart (2)
(Second and Subsequent Reception)

4. Simultaneous Serial Data Transmission and Reception

Figure 16.16 shows sample flowcharts for simultaneous serial transmission and reception.

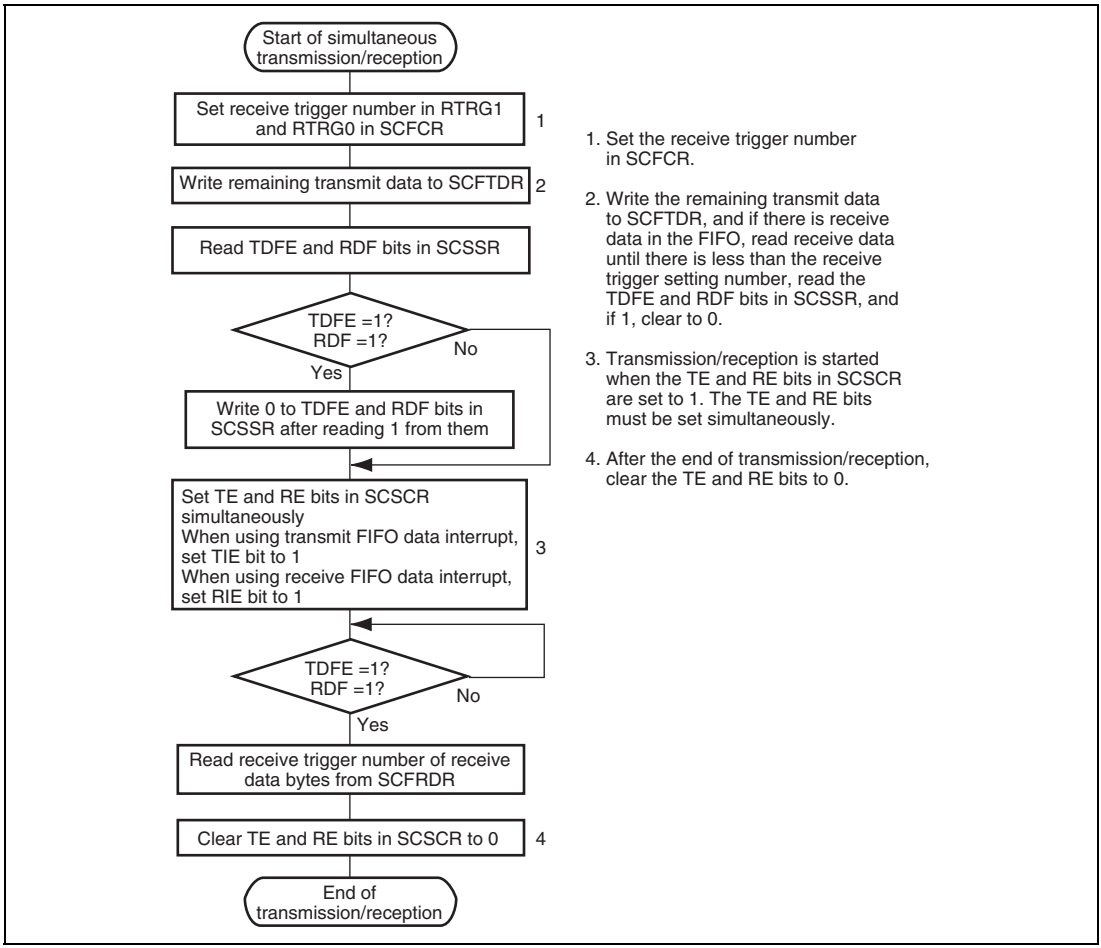


Figure 16.16 Sample Simultaneous Serial Transmission and Reception Flowchart (1) (First Transfer after Initialization)

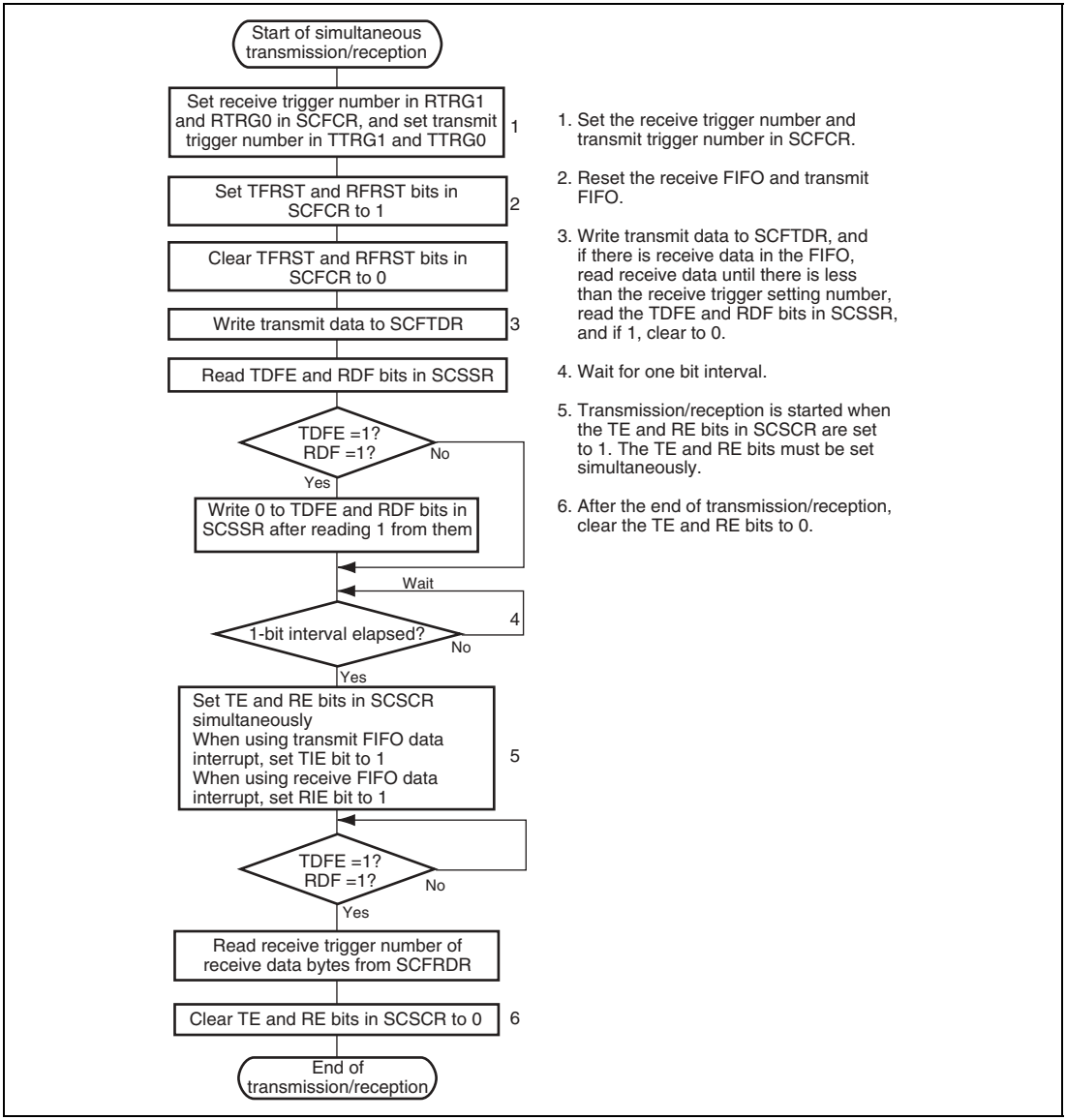


Figure 16.16 Sample Simultaneous Serial Transmission and Reception Flowchart (2) (Second and Subsequent Transfer)

16.5 SCIF Interrupt Sources and DMAC

The SCIF supports six interrupts in asynchronous mode—transmit-FIFO-data-empty (TXI), transmit-data-stop (TDI), receive-error (ERI), receive-FIFO-data-full (RXI), break-receive (BRI), and receive-data-ready (DRI). In clock synchronous mode, the SCIF supports two interrupts—transmit-FIFO-data-empty (TXI) and receive-FIFO-data-full (RXI). The vectors of each interrupt are the same.

Table 16.4 shows the interrupt sources. The interrupt sources can be enabled or disabled by means of the TIE, RIE, ERIE, BRIE, DRIE, and TSIE bits in SCSCR.

When the TDFE flag in SCSSR is set to 1, a TXI interrupt request is generated. When the TSF flag in SCSSR is set to 1, a TDI interrupt request is generated. The DMAC can be activated and data transfer performed on generation of TXI and TDI interrupt requests. The DMAC requests of TXI and TDI are assigned to the same vector.

When the RDF flag in SCSSR is set to 1, an RXI interrupt request is generated. The DMAC can be activated and data transfer performed on generation of an RXI interrupt request.

When using the DMAC for transmission/reception, set and enable the DMAC before making SCIF settings. See section 10, Direct Memory Access Controller (DMAC), for details of the DMAC setting procedure.

When the ER flag in SCSSR is set to 1, an ERI interrupt request is generated.

When the BRK flag in SCSSR is set to 1, a BRI interrupt request is generated.

When the DR flag in SCSSR is set to 1, a DRI interrupt request is generated.

When the TSF flag in SCSSR is set to 1, a TDI interrupt request is generated.

The vectors of TXI, TDI, ERI, BRI, RXI and DRI are the same.

The DMAC activation and interrupts cannot be generated simultaneously by the same source. The following procedure should be used for the DMAC activation.

1. Set the interrupt enable bits (TIE, RIE and TDIE) corresponding to the generated source to 1.
2. Mask the corresponding interrupt requests by using the interrupt mask register of the interrupt controller.

Table 16.4 SCIF Interrupt Sources

Description	DMAC Activation
Interrupt initiated by receive error flag (ER) or break flag (BRK)	Not possible
Interrupt initiated by receive FIFO data full flag (RDF) or receive data ready (DR)	Possible*1
Interrupt initiated by transmit FIFO data empty flag (TDFE) or transmit data stop flag (TSF)	Possible*2

Notes: 1. The DMAC can be activated only by a receive-FIFO-data-full interrupt request.
2. The DMAC can be activated by a transmit-FIFO-data-empty (TDFE) or transmit-data-stop (TSF) interrupt request. When the DMAC is activated by the TSF interrupt, it is cleared by either of two cases listed below.
(1) The TSF flag is read by the CPU.
(2) The transmit FIFO is full.

See section 4, Exception Handling, for priorities and the relationship with non-SCIF interrupts.

16.6 Notes on Usage

Note the following when using the SCIF.

SCFTDR Writing and the TDFE Flag: The TDFE flag in the serial status register (SCSSR) is set when the number of transmit data bytes written in the transmit FIFO data register (SCFTDR) has fallen to or below the transmit trigger number set by bits TTRG1 and TTRG0 in the FIFO control register (SCFCR). After TDFE is set, transmit data up to the number of empty bytes in SCFTDR can be written, allowing efficient continuous transmission.

However, if the number of data bytes written in SCFTDR is equal to or less than the transmit trigger number, the TDFE flag will be set to 1 again after being read as 1 and cleared to 0. TDFE clearing should therefore be carried out when SCFTDR contains more than the transmit trigger number of transmit data bytes.

The number of transmit data bytes in SCFTDR can be found from the 14 to 8 bits of the FIFO data count register (SCFDR).

SCFRDR Reading and the RDF Flag: The RDF flag in the serial status register (SCSSR) is set when the number of receive data bytes in the receive FIFO data register (SCFRDR) has become equal to or greater than the receive trigger number set by bits RTRG1 and RTRG0 in the FIFO control register (SCFCR). After RDF is set, receive data equivalent to the trigger number can be read from SCFRDR, allowing efficient continuous reception.

However, if the number of data bytes in SCFRDR is still equal to or greater than the trigger number after a read, the RDF flag will be set to 1 again if it is cleared to 0. RDF should therefore be cleared to 0 after being read as 1 after all receive data has been read.

The number of receive data bytes in SCFRDR can be found from the 6 to 0 bits of the FIFO data count register (SCFDR).

Break Detection and Processing: Break signals can be detected by reading the RxD pin directly when a framing error (FER) is detected. In the break state the input from the RxD pin consists of all 0s, so the FER flag is set and the parity error flag (PER) may also be set.

Although the SCIF stops transferring receive data to SCFRDR after receiving a break, the receive operation continues.

Receive Data Sampling Timing and Receive Margin: As an example, when the sampling rate is 1/16, the SCIF operates on a base clock with a frequency of 8 times the transfer rate.

In reception, the SCIF synchronizes internally with the fall of the start bit, which it samples on the base clock. Receive data is latched at the rising edge of the eighth base clock pulse.

Figure 16.17 shows an example of asynchronous mode receive data sampling timing in asynchronous mode.

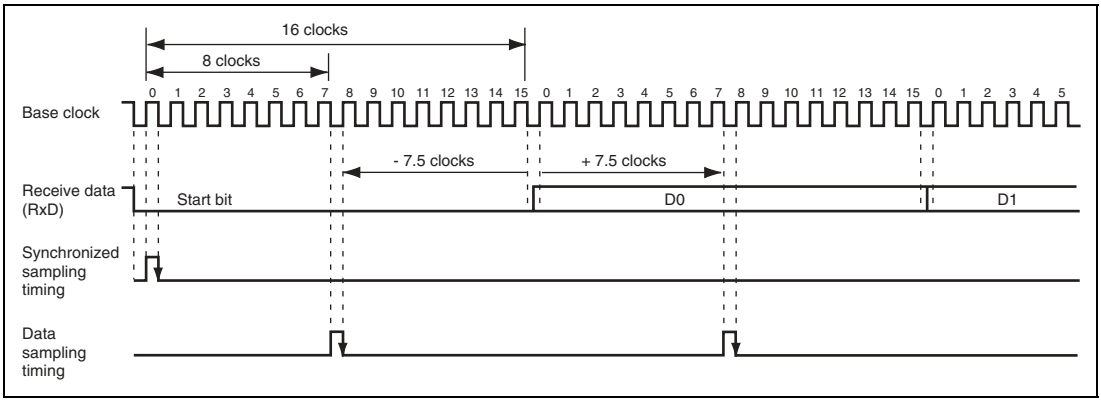


Figure 16.17 Receive Data Sampling Timing in Asynchronous Mode

The receive margin in asynchronous mode can therefore be expressed as shown in equation (1).

$$M = \left| \left(0.5 - \frac{1}{2N}\right) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100\% \quad \text{..... (1)}$$

M: Receive margin (%)
N: Ratio of clock frequency to bit rate (N = 16)
D: Clock duty cycle (D = 0 to 1.0)
L: Frame length (L = 9 to 12)
F: Absolute deviation of clock frequency

From equation (1), if F = 0 and D = 0.5, the receive margin is 46.875%, as given by equation (2).

When D = 0.5 and F = 0:

$$M = (0.5 - 1/(2 \times 16)) \times 100\% = 46.875\% \quad \text{..... (2)}$$

This is a theoretical value. A reasonable margin to allow in system designs is 20% to 30%.

Initialize SCIF: The communication format should be changed according to the following procedure after the TE and RE bits are cleared to 0. The transmit shift register (SCTSR) is initialized by clearing the TE bit to 0. The receive shift register (SCRSR) is initialized by clearing the RE bit to 0. The contents of the serial status register (SCSSR), transmit FIFO data register (SCFTDR), and receive FIFO data register (SCFRDR) is retained even when the TE and RE bits are cleared to 0. The TE bit should be cleared after all transmit data have been transmitted and the TEND bit in SCSSR has been set to 1. The TE bit can be cleared while the data transmission is in progress. However, the data go to high-impedance after the TE bit has been cleared to 0. To start transmission again by setting the TE bit to 1, the SCFTDR should be reset by setting the TFRST bit in SCFCR to 1.

The external clock should not be halted during the operation including an initialization since the operation becomes unstable.

Transition to Power-Down Mode: The transitions to software standby mode and module standby mode should not be made by setting the module standby bit of SCIF during serial transmission and reception. The transitions to software standby mode and module standby mode should be made after both TE and RE bits in the serial control register (SCSCR) have been cleared (transmission/reception disabled).

Transmit/Receive Pin in Transmission/Reception Disabled State: The TxD pin of the SCIF is in high-impedance state while the TE bit is cleared. The RxD pin is in input fixed state while the RE bit is cleared. Care is required on dealing a signal line that is connected to the TxD pin. When the connect destination pin is in input state, connect the pull-up register to a signal line.

Section 17 Boot Function (BOOT)

This LSI has a boot function which is used to download, to the on-chip memory, the firmware needed for the initial programming of the external flash memory. This function enables updating programs and dumping contents after the flash memory has been mounted on the board. This function eliminates the need to provide external address and data buses for the programming of the flash memory when this LSI and the flash memory, in bare-chip form, are sealed in a single package. This makes it possible to dramatically reduce the number of pins.

17.1 Features

- Automatic downloading of the initial programming program from SCIF0

  The operation of downloading, via the SCIF0, the initial programming program of the external flash memory to the on-chip memory (U memory), starts after a power-on reset. This transfer is synchronous and via the UART. This allows the downloading of any initial programming program according to the type of the external flash memory that is connected. If the specification defined in this section is conformed to, programs for purposes other than the initial programming of the external flash memory may be downloaded. The size of the program to be downloaded must be no more than 8,192 bytes and is otherwise freely determined.
- Automatic bit-rate adjustment

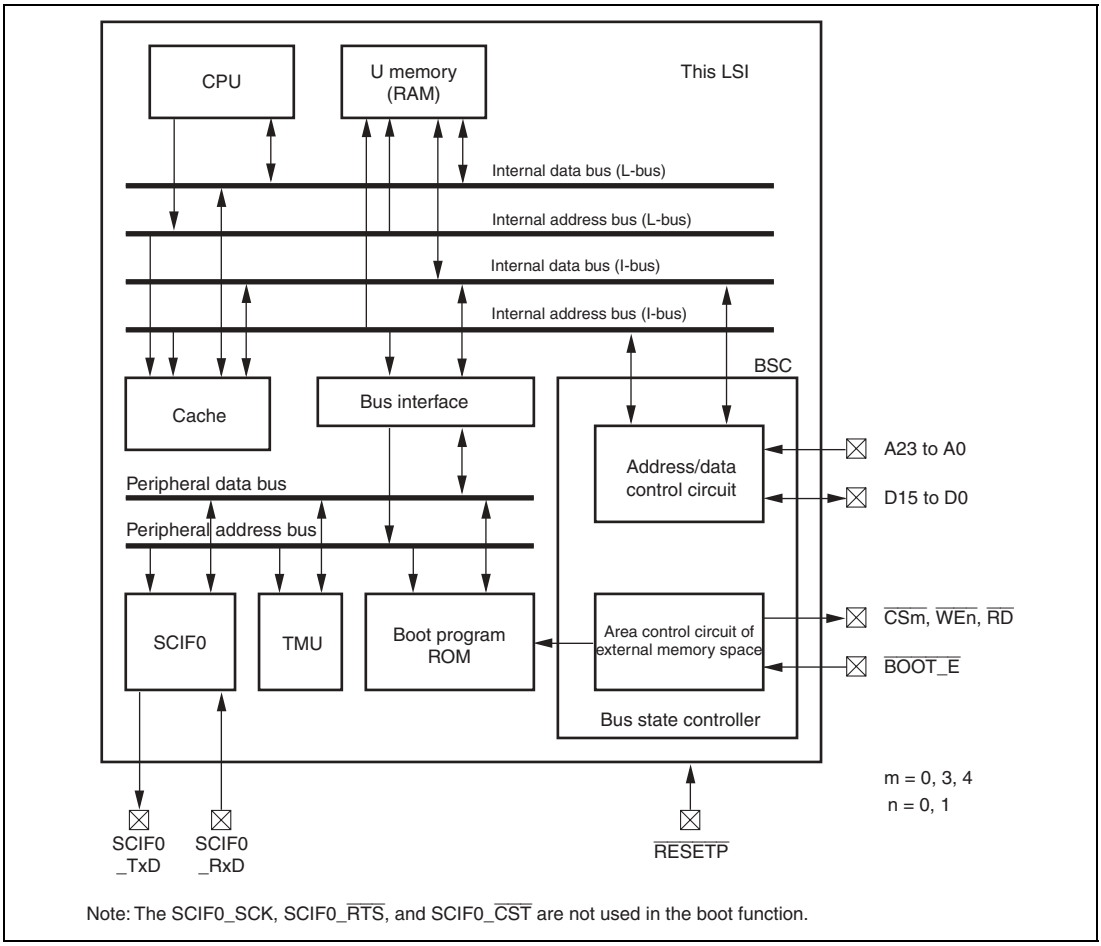
  Automatic adjustment of the bit-rate at which the program is downloaded via the SCIF0 is possible.
- High-speed execution of the downloaded program in on-chip memory

  Since the program downloaded from the SCIF0 is expanded in the on-chip RAM, it is executable at high speed.
- Execution of the downloaded program starts automatically on completion of downloading

  After the specific program has been downloaded, processing automatically branches to the start address (H'A55F0000) in the downloaded program. Execution thus automatically starts.

Note: Since the boot function is configured as a single circuit module, it is one of the target modules for power control by the module-standby function. The boot function is only necessary for boot processing. If, therefore, power consumption needs to be reduced in normal operation or in the sleep state, we recommend that the supply of the clock signal to the boot-function module be stopped by setting the MSTP43 bit in STBCR4 to 1. For details on the module-standby function, refer to section 13, Power-Down Modes.

Figure 17.1 shows the block diagram of the boot function (BOOT). The I/O pins for the bus state controller (BSC) (A23 to A0, D15 to D0, $\overline{WE1}$ to $\overline{WE0}$, $\overline{CS0}$, and $\overline{RD}$) are connected to the external flash memory and the I/O pins for the SCIF0 (SCIF0_RxD and SCIF0_TxD) are connected to the external host.



Note: The SCIF0_SCK, SCIF0_RTS, and SCIF0_CST are not used in the boot function.

Figure 17.1 Block Diagram of Boot Function

17.2 Input/Output Pin

Table 17.1 shows the pin only for the boot function. For details on the pins that are connected to the external flash memory, refer to section 9, Bus State Controller (BSC); for details on the pins that are related to the SCIF0, refer to section 16, Serial Communication Interface with FIFO (SCIF); and, for details on the power-on reset pin ($\overline{\text{RESETP}}$), refer to section 4, Exception Handling.

Table 17.1 Input/Output Pin

Pin Name	I/O	Function
$\overline{\text{BOOT_E}}$	Input	Boot Enable Input The state of this pin can only be switched during a reset.

17.3 Register Description

The BOOT has no registers which are accessible by the CPU.

17.4 Operation

17.4.1 Address Space in Boot Mode

When the $\overline{\text{BOOT_E}}$ pin is asserted to its low level by a power-on reset ($\overline{\text{RESETP}}$ = low), this LSI enters boot mode at the end of a reset. At this point, processing by this LSI would not normally branch to the reset vector. In boot mode, processing branches to the on-chip boot program instead, to initiate the boot function.

Because of the initial state by a reset, the CPU enters privileged mode. For details on privileged mode, refer to section 2.1, Processing States and Processing Modes. In boot mode, which differs from the normal privileged mode, area 0 in the external memory space is divided into the two areas shown in figure 17.2. The area from H'00000000 to H'01FFFFFF is exclusively used for the boot function. Access to locations in this area by the user is thus prohibited. The boot program is placed here. The area from H'02000000 to H'03FFFFFF is an area 0 which is the same as the area 0 of the normal privileged mode.

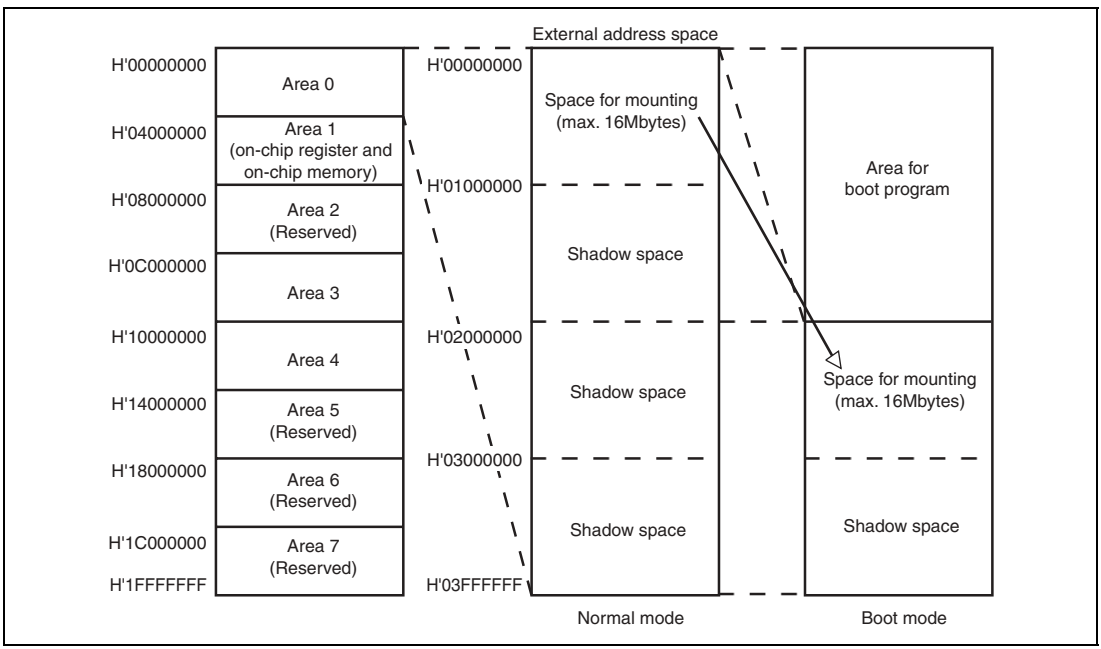


Figure 17.2 External Memory Space in Boot Mode

In the initial programming of the external flash memory, the reset vector always selects a branch to the address H'A0000000 (a virtual address as seen from the CPU) after the power-on reset signal has been cancelled and area 0 in the external address space is accessed. The programming program must thus be executed from a start address in area 0. In boot mode, however, addresses H'00000000 to H'01FFFFFF are exclusively used for the boot function, so programming is executed from the start address H'02000000. The maximum mounted area for this LSI is 16 Mbytes. Since the address range from H'02000000 to H'02FFFFFF is the shadow space for the range from H'00000000 to H'00FFFFFF, if programming starts at address H'02000000, access can be performed from address H'00000000 after boot mode has been cancelled.

Note: The state of the input on the $\overline{\text{BOOT_E}}$ pin input must not be changed except during a reset. If this state is changed, correct operation is not guaranteed.

17.4.2 Procedure for Execution of Boot Processing

The initial programming program of the external flash memory must be placed in the external host before the boot function is used. The initial programming program must be prepared in accordance with the programming algorithm for the external flash memory that is to be used.

The boot program starts preparing for the transfer of data to and from the external host over channel 0 of the SCIF. After the necessary initial settings for the SCIF0 have been made, the bit rate for use in transmission/reception is automatically adjusted to suit the external host. Figure 17.3 shows the procedure according to which boot processing is executed.

Automatic Adjustment Operation for SCIF0 Bit Transfer Rate: This LSI automatically adjusts the bit transfer rate for transmission/reception to a value in the range between 9.6 kbps and 38.4 kbps. When this LSI is activated in boot mode, it measures the period of the low levels in the synchronous data (H'00) which is continuously transmitted from the external host. Channel 0 in the timer unit (TMU) is used for this low-period measurement. Here, the external host must set the format for transfer between this LSI and the SCIF0 as 8-bit data, one stop bit, and no parity bit. After the boot program has calculated the bit rate at which bits are being transferred from the external host by using the measurement of the low period and the optimum setting has been made, a single byte with the value H'00 is transmitted to the external host as notification of the end of the bit-rate adjustment. After the external host has successfully received this notification of the end of bit-rate adjustment, a single byte with the value H'55 must be returned to this LSI. After this LSI has received this byte, it transmits a single byte with the value H'AA.

Note: If reception is not successful, execution enters an endless loop and the program will not react. In such a case, reinitiate boot mode after a reset, and repeat the above operations.

Transferring of Initial Programming Program: The boot program starts downloading the initial programming program after the automatic adjustment of the bit transfer rate has been completed. The maximum size of the initial programming program that is downloaded over the SCIF0 channel is 8,192 bytes (H'2000)*[1]. The downloaded initial programming program is stored in the address range from H'055F0000 to H'055F1FFF (physical addresses) in the on-chip memory (U memory). After the downloading has been completed, the boot program automatically branches to address H'055F0000*[2]. Therefore the start address of the program that is downloaded must be H'055F0000.

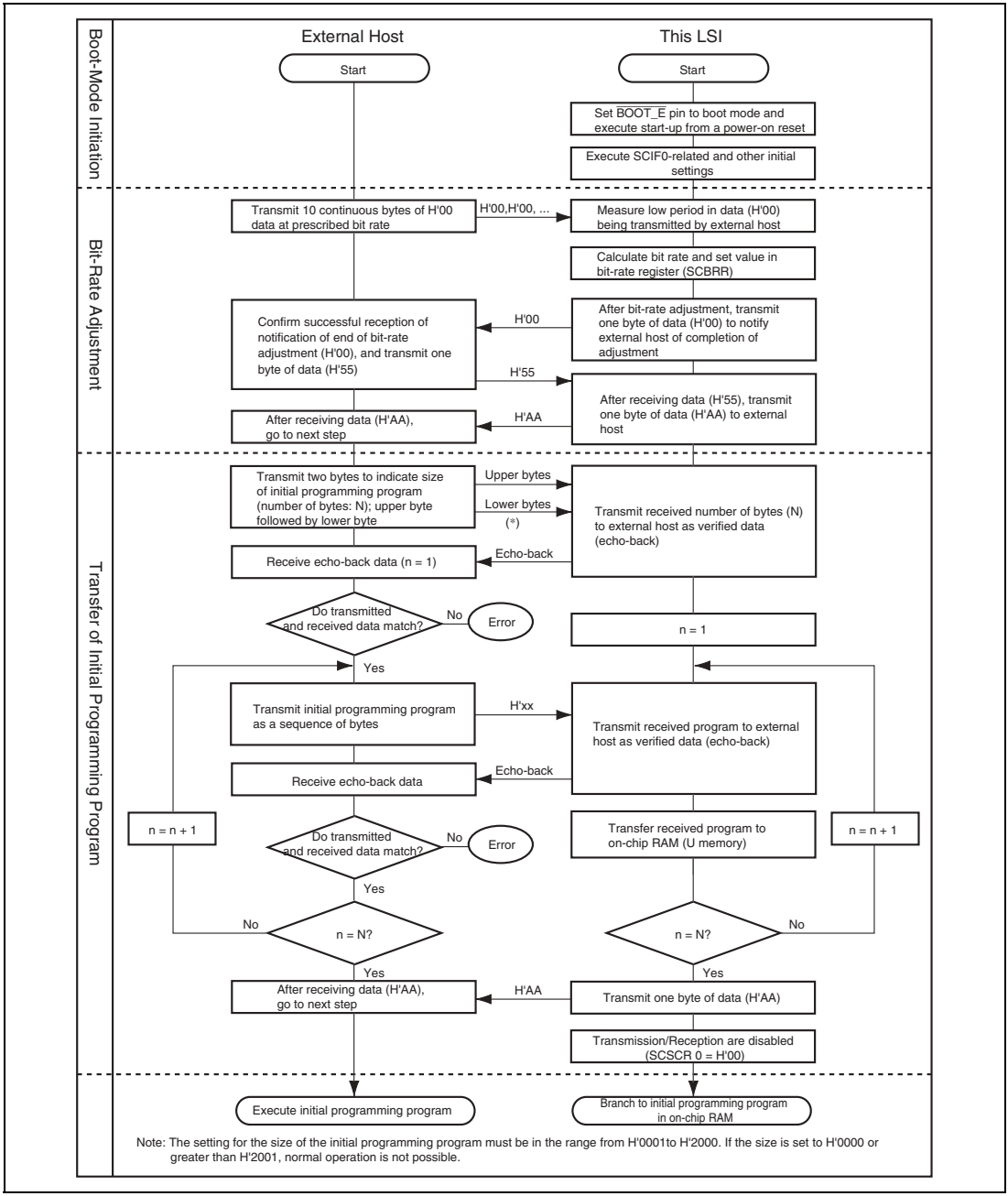
Although the boot program uses the address range from H'055FFFFC to H'055FFFFF (physical addresses) as a working area, the downloaded initial programming program is able to use this area after downloading has been completed.

Notes: 1. The size setting for the initial programming program must not be H'0000 or greater than H'2001, as normal operation is not possible in such a case. If such a program is used, correct operation is not guaranteed.

2. The initial setting routine of user program that is programmed by the initial programming program in boot mode should be stored from H'02000000 in area 0. The initial setting is enabled since the branch to the first address of area 0 (H'00000000) is made immediately after a power-on reset in normal mode.

A maximum of 16 Mbytes may be mounted in area 0, i.e., the address range from H'02000000 to H'02FFFFFF is available (H'00000000 to H'00FFFFFF in normal mode) as shown in figure 17.2.

In this LSI, external memory may be mounted in areas 3 and 4 as well as in area 0. The maximum size for each area is 16 Mbytes. Areas 3 and 4 may be used as expansion spaces for the external flash memory, when this is necessary, by preparing a corresponding initial programming program.



Note: The setting for the size of the initial programming program must be in the range from H'0001to H'2000. If the size is set to H'0000 or greater than H'2001, normal operation is not possible.

Figure 17.3 Procedure for Execution of Boot Processing

17.5 Usage Note

17.5.1 Endian when Using Boot Function

The boot function is operated by the firmware that is stored in the boot ROM and functions on big endian. Therefore, when the $\overline{\text{BOOT_E}}$ pin is asserted to its low level by a power-on reset ($\overline{\text{RESETP}}$ = low), the MD5 pin should be driven low (big endian mode). After the specific program has been downloaded, the boot function is branched to the start address in the initial programming program for the external flash memory downloaded in the on-chip memory (U-memory). The initial programming program needs to be specified to operate on big endian. To execute normal operation on little endian, pay attention to the code alignment when the flash memory is initially programmed.

17.5.2 Clock Frequency and Data Transfer Rate when Using Boot Function

The transfer rate of SCIF which communicates with an external host when the boot function is used has the limitation shown in table 17.2. Note that a proper value should be set for the transfer rate in the program for the external host, such as "exflash".

Table 17.2 Clock frequency and data transfer rate of SCIF when boot function is used

CPG setting	Input clock frequency	SCIF transfer rate
Mode 5	10MHz or more	9600bps
Clock ratio	13MHz or more	19200bps
Iϕ:Bϕ:Pϕ = 2:2:1	16MHz or more	38400bps
Mode 7	20MHz or more	9600bps
Clock ratio	26MHz or more	19200bps
Iϕ:Bϕ:Pϕ = 1:1:1/2	32MHz or more	38400bps

Section 18 USB Pin Multiplex Controller (USBPM)

18.1 Feature

The USB pin multiplex controller controls the data path to USB transceiver from USB host controller or USB function controller. This USB pin multiplex controller also incorporates a clock select control function that selects the clock signal to be provided to the USB host controller and USB function controller as either an internally generated clock or externally input clock.

Both the USB host controller and the USB function controller are connected to the USB transceiver via multiplexer that is controlled by the EXCPG control register (EXCPGCR). The USB transceiver can be connected to USB host controller or USB function controller.

This LSI generates a 48-MHz clock, which is specified by the USB1.1 specifications, by the on-chip multiplier circuit using a13-MHz clock input from the RDI_REFCLK_IN pin and provides the clock to the USB host controller and USB function controller. A 48-MHz clock can also be externally provided to the USB host controller and USB function controller. The on-chip multiplier circuit allows the user to receive a stable clock from the RF-IC to be connected to this LSI directly without connecting 48-MHz TCXO externally. Note, however, that a specific length of jitter may occur when the on-chip multiplier circuit is used. Before using a clock provided from the on-chip multiplier circuit, check the connectivity to other USB devices.

Figure 18.1 shows the USB pin multiplexer indicating the connection among the on-chip USB host controller, USB function controller, on-chip port 1 analog USB transceiver, and clock control circuit.

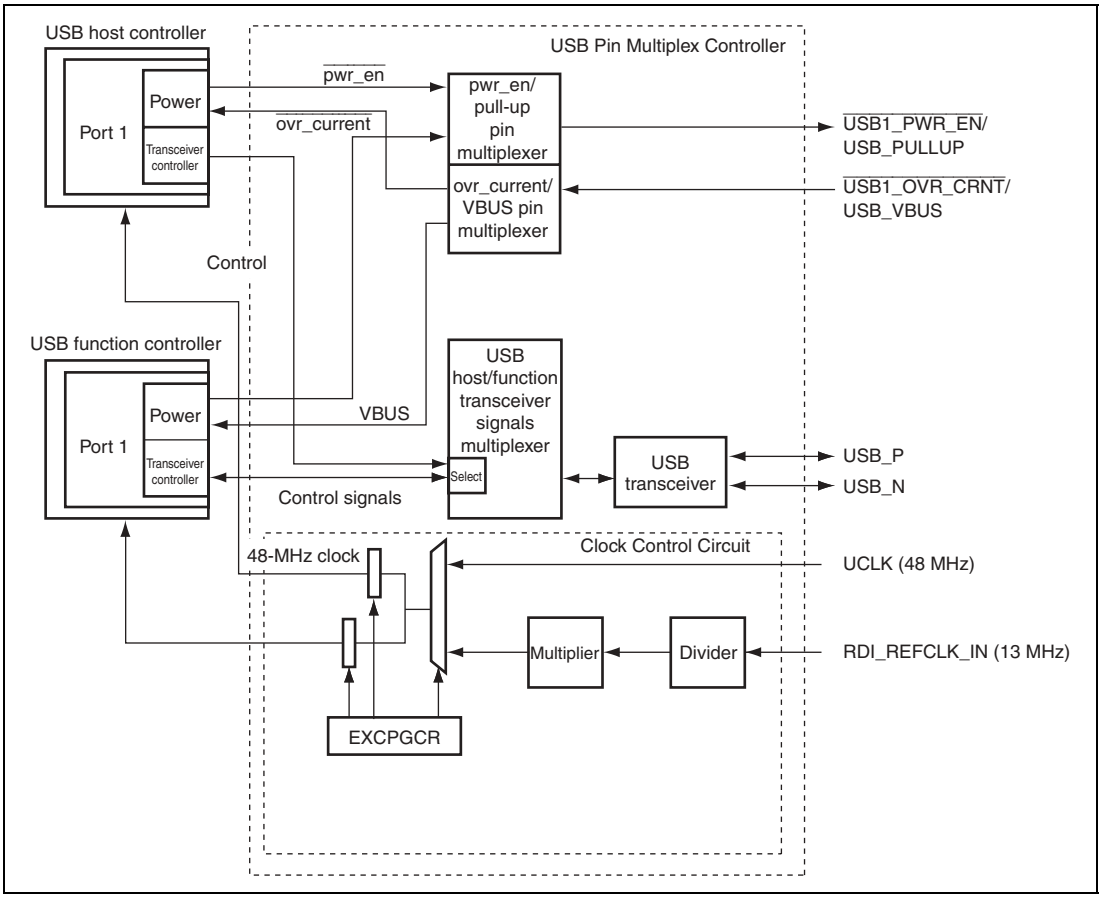


Figure 18.1 Block Diagram of this USB

18.2 Input/Output Pins

USB pin multiplexer controller has pins that are shown in tables 18.1, 18.2, and 18.3

Table 18.1 Pin Configuration (Analog Transceiver Signal)

Pin Name	I/O	Description
USB_P	Input/Output	P pin
		D+ port transceiver pin
USB_N	Input/Output	D pin
		D– port transceiver pin

Note: The pins shown in table 18.1 can be used as port 1 USB host controller pins, or port 1 USB function controller pins. Ensure these pins are open when not used.

Table 18.2 Pin Configuration (Power Control signal)

Pin Name	I/O	Description
$\overline{\text{USB_PWR_EN}}$/ USB_PULLUP	Output	Power enable pin/ USB pull-up control pin
		Enables power supply (USBH)/controls USB pull-up (USBF)*
$\overline{\text{USB_OVR_CRNT}}$/ USB_VBUS	Input	Over-current pin/VBUS pin
		Detects over-current (USBH)/monitors USB cable connection (USBF)*

Note: The pins shown in table 18.2 can be used for power control for USB. Pins indicated by * have multiplexed functions for the USB host controller (USBH) and the USB function controller (USBF).

Table 18.3 Pin Configuration (Clock Control signal)

Pin Name	I/O	Description
RDI_REFCLK_IN	Input	Bluetooth interface clock pin
		Pin for the Bluetooth interface clock (13MHz) input (generates a 48-MHz clock by the on-chip multiplier)
UCLK	Input	External clock pin
		Pin for the 48-MHz clock input from an external device

Note: A 48-MHz clock is provided to the USBH or USBF by using RDI_REFCLK_IN or UCLK.

18.3 Register Description

Register configuration of the USB pin multiplexer controller is described below. See section 27, List of Registers, for the addresses of this register and the state of it in each processing status.

- EXCPG control register (EXCPGCR)

18.3.1 EXCPG Control Register (EXCPGCR)

EXCPGCR controls providing a 48-MHz clock to the USB host controller and USB function controller and performs module reset.

Bit	Bit Name	Initial Value	R/W	Descriptions
7 to 5	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
4	USBVALID	0	R	USB Status The USB module status bit that Indicates whether the USB module (both the USB host and USB function controllers) can be used or not. This bit is a read-only bit. A write access to this bit is ignored. Immediately after power-on reset of the USB module, immediately returning from the software standby, or immediately after reset by USBRESET, the USB module cannot be used for a short time. Before using the USB module, check this bit status. Note: Even if the USBHSTP and USBFSTP bits are set, the on-chip multiplier circuit does not stop and this bit remains set to 1. 0: USB module cannot be used. 1: USB module can be used.

Bit	Bit Name	Initial Value	R/W	Descriptions
3	USBRESET	1	R/W	USB Reset The USB module reset bit to be used to reset the USB module (both the USB host and USB function controllers). If this bit is cleared to 0, the USBVALID bit is cleared to 0 after approximately 1 μs. After checking that the USBVALID bit is cleared to 0, set the USBRESET bit to 1. After this USBRESET bit is set to 1, the USB module starts the reset sequence. After the reset sequence, the USBVALID bit is set to 1 again. To ensure USB module initialization, follow the above sequence. To provide a clock again to the USB module after the clock has been stopped using the USBHSTP and USBFSTP bits, USB module operation may be unstable. Therefore, use the USB module after resetting by this bit. 0: USB module is reset. 1: USB module is in normal operation.
2	USBHSTP	0	R/W	Module Stop USBH The USB host controller stop bit to be used to provide or stop the USB host controller operating clock. Clearing this bit to 0 provides the clock to the USB host controller. Setting this bit to 1 stops the clock. If both the USBHSTP and USBFSTP bits are set to 1 or cleared to 0, the USB transceiver operation stops. The USB_P and the USB_N outputs pins enter the high-impedance state, and then $\overline{\text{USB_PWR_EN}}$/USB_PULLUP pin is brought low. Before using the USB host or USB function controller, clear one of USBHSTP or USBFSTP to 0. Note that the USB host controller and USB function controller cannot be used simultaneously. If the USB module control register is accessed while both the USBHSTP and USBFSTP bits are cleared to 0, correct operation cannot be guaranteed. Note that the USB host controller register access is ignored while the clock provided to the USB host controller is stopped. 0: USB host controller operation continues. 1: Stop providing clock to the USB host controller.

Bit	Bit Name	Initial Value	R/W	Descriptions
1	USBFSTP	0	R/W	Module Stop USBF The USB function controller stop bit to be used to provide or stop the USB function controller operating clock. Clearing this bit to 0 provides the clock to the USB function controller. Setting this bit to 1 stops the clock. If both the USBHSTP and USBFSTP bits are set to 1 or cleared to 0, the USB transceiver operation stops. The USB_P and the USB_N outputs pins enter the high-impedance state, and then the $\overline{\text{USB_PWR_EN}}$/USB_PULLUP pin is brought low. Before using the USB host or USB function controller, clear one of USBHSTP and USBFSTP to 0. Note that the USB host controller and USB function controller cannot be used simultaneously. If the USB module control register is accessed while both the USBHSTP and USBFSTP bits are cleared to 0, correct operation cannot be guaranteed. Note that the USB function controller register access is ignored while the clock provided to the USB function controller is stopped. 0: USB function controller operation continues. 1: Stop providing clock to the USB function controller.
0	USBCLKSEL	0	R/W	USB Clock Select The USB clock source select bit used to select whether the USB clock is internally generated or the USB clock is externally input. After this bit is modified, both the USB host and the USB function controllers must be initialized by the USBRESET bit. 0: The USB clock is generated internally. 1: The USB clock is input by the UCLK pin.

18.4 Examples of External Circuit

18.4.1 Example of the Connection between USB Function Controller and External Circuit

Figures 18.2 and 18.3 show example connections of USB function controller and external circuit. When the USB function controller is used, a signal must be provided to the cable connection monitor pin (USB_VSBUS). The USB_VBUS pin is multiplexed with the $\overline{\text{USB_OVR_CRNT}}$ pin, and writing 0 to USBFSTP and 1 to USBHSTP in EXCPGCR register selects the USB_VBUS pin functions. According to the status of the USB_VBUS pin, the USB function controller recognizes whether the cable is connected/disconnected. Also, pin D+ must be pulled up in order to notify the USB host/hub that the connection is established. The sample circuits in figures 18.2 and 18.3 use the USB_PULLUP pin for pull-up control.

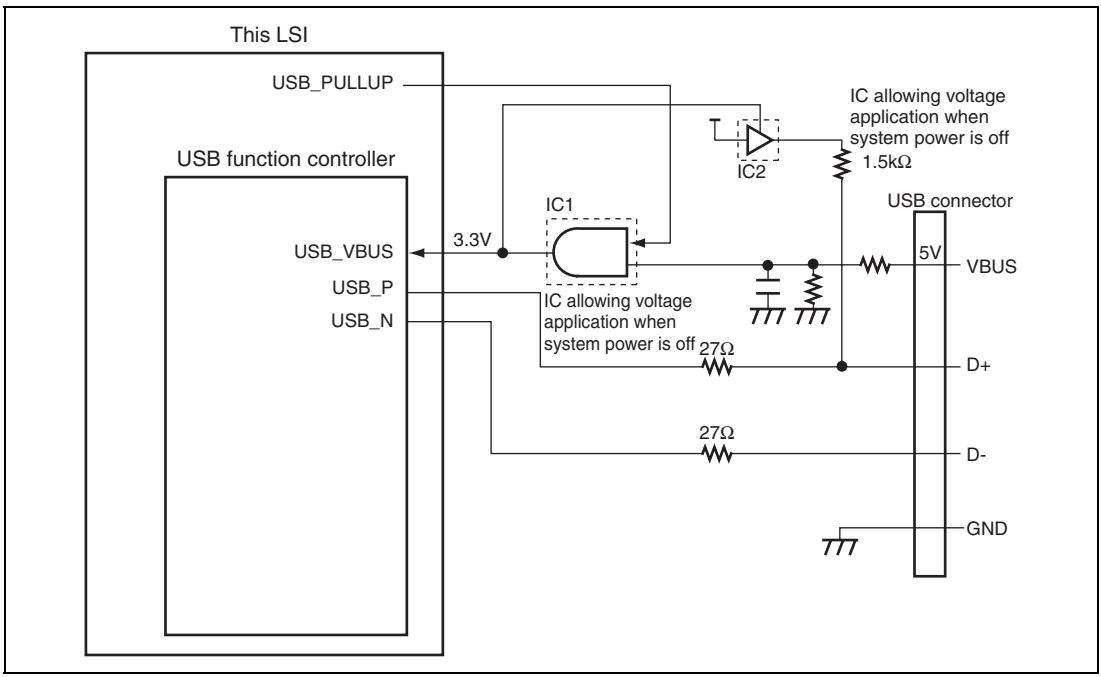


Figure 18.2 Example 1 of Connection between USB Function Controller and External Circuit

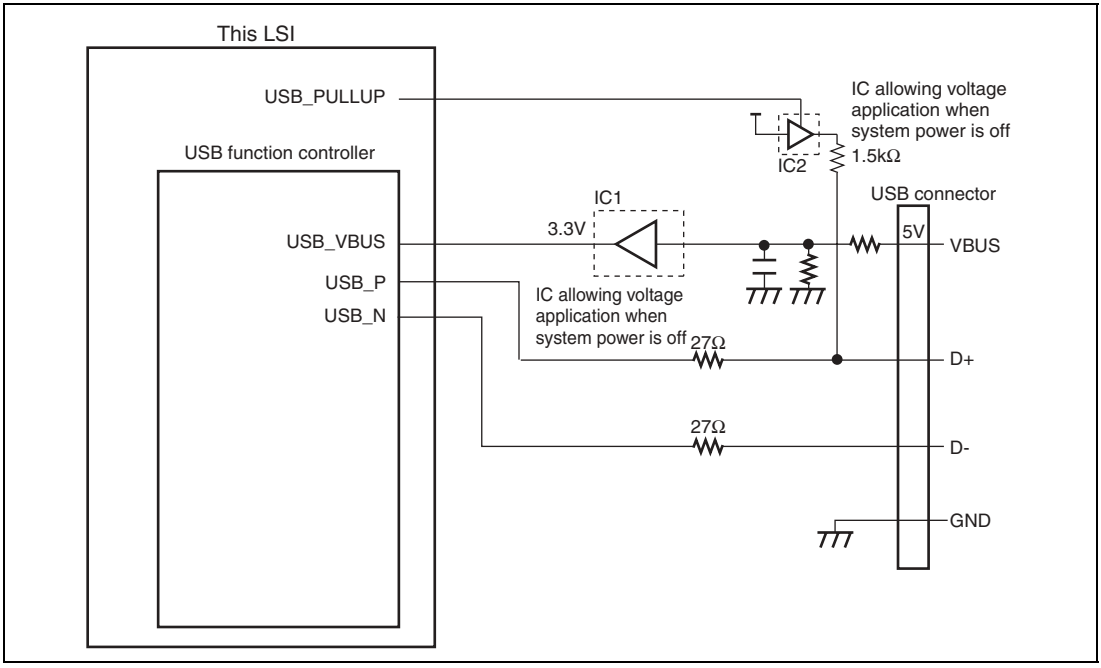


Figure 18.3 Example 2 of Connection between USB Function Controller and External Circuit

- D+ Pull-up Control

  Control D+ pull-up via the USB_PULLUP pin in the system in case that the connection-notification (D+ pull-up) of connection to the USB host controller or hub is to be inhibited (i.e., during high-priority processing or initialization processing).

  The D+ pull-up control signal and USB_VBUS pin input signal should be controlled via the USB_PULLUP pin and the USB cable VBUS (AND circuit) as shown in the examples in figure 18.2.

  D+ pull-up is inhibited when the USB_PULLUP pin is low in an example of figure 18.2. (The initial setting of the $\overline{\text{USB_PWR_EN}}$/USB_PULLUP pin is low.)

  Pull-up D+ after confirming that USB_VBUS pin became high, when the pull-up control is performed directly by USB_PULLUP pin as is shown in an example of figure 18.3.

  Use an IC such that allows voltage application when system power is off (for example, HD74LV1G126A) for the pull-up control IC (IC2 in figures 18.2 and 18.3).

  (The UDC core in this LSI holds the powered state when USB_VBUS pin is low, regardless of the D+/D− state.)

- Detection of USB Cable Connection/Disconnection

  As USB function controller in this LSI manages the state by hardware, USB_VBUS signal is necessary to recognize connection or disconnection of the USB cable. The power supply signal (VBUS) in the USB cable is used for USB_VBUS. However, if the cable is connected to the USB host or hub when the power of USB function controller (this LSI—installed system) is off, a voltage of 5 V will be applied from the USB host controller or hub.

  Therefore, use an IC such that allows voltage application when system power is off (for example, HD74LV1G08A) for the IC1 in figures 18.2 to 18.3.

18.4.2 Example of the Connection between USB Host Controller and External Circuit

Figure 18.4 shows example connections of the USB host controller and external circuit. When using the USB host controller, a separate LSI must be used for USB power bus control (equivalent to the USB power control LSI's in figure 18.4). Make sure the LSI has the power supply capacity to satisfy the USB standard, and select one that has an overcurrent protection function. Configure the system so that the input to the $\overline{\text{USB_OVR_CRNT}}$ pin is Low on detection of an overcurrent.

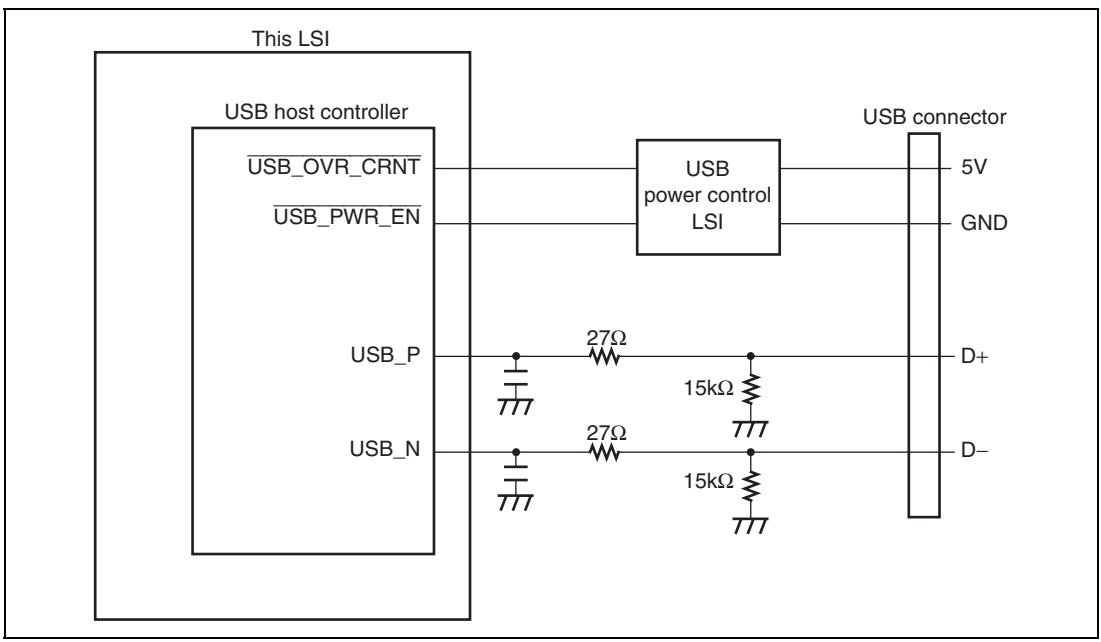


Figure 18.4 Example of Connection USB Host Controller and External Circuit

18.5 Usage Notes

About the Examples of External Circuit: These examples of connection with an external circuit in this chapter are for reference only, therefore proper operation is not guaranteed with these circuit examples. If system countermeasures are required for external surges and ESD noise, use a protective diode, etc.

About the Clock Input to the USB: When the USB clock is generated internally, the clock source is generated not by an input clock from the EXTAL pin specified as the CPU clock source but an input clock by the RDI_REFCLK_IN pin specified as clock used for data I/O with the RF-IC of the Bluetooth and the link controller. To use the USB function of this LSI without being connected to the RF-IC of the Bluetooth, the USB should be input the 13-MHz clock conforming to the USB standard by the RDI_REFCLK_IN pin. To use the USB function of this LSI without being connecting to the RF-IC of the Bluetooth actually, input of 48-MHz clock conforming to the USB standard by means of the UCLK pin is recommended.

Problems with Using Non-specified Commercial USB Cable: Some USB cables on the market do not meet the specification, and there are confirmed cases of non-specified USB cable to have problems with connectivity.

Use only specified USB cable with this product.

Section 19 USB Host Controller (USBH)

The USB host controller module (hereafter referred to as the host controller) incorporated in this LSI supports the Open Host Controller (OpenHCI) Specification for the Universal Serial Bus (USB) as well as the universal serial bus specification ver.1.1.

The OpenHCI specification for the USB is a register-level description of the host controller. It is necessary to refer the OpenHCI specification to develop drivers for this host controller and hardware.

19.1 Features

The host controller in this LSI has the following features:

- Conforms to the OpenHCI specification ver.1.0 register set
- Corresponding to the universal serial bus specification ver.1.1
- Supports root hub function
- Supports low-speed (1.5 Mbps) and full-speed (12 Mbps) modes
- Supports overcurrent detection
- Supports 127-endpoint control in maximum

19.2 Input/Output Pins

Table 19.1 shows the pin configuration of the USB host controller.

For the detailed method for setting each pin, refer to section 18, USB Pin Multiplex Controller(USBPM).

Table 19.1 Pin Configuration

Pin Name	I/O	Function
UCLK	Input	External Clock Pin For direct input to 48MHz clock from an external device
RDI_REFCLK_IN	Input	External clock input The 13 MHz external clock is input. This signal is multiplied internally, and the clock of 48 MHz is generated.
$\overline{\text{USB_PWR_EN}}$/ USB_PULLUP	Output	Power Enable Pin/Pull-up Control Pin Enables power supply (USBH)/controls USB pull-up (USBF)
$\overline{\text{USB_OVR_CRNT}}$/ USB_VBUS	Input	Overcurrent Pin/ VBUS Pin Detects over-current (USBH)/monitors USB cable connection (USBF)
USB_P	Input/ Output	P Pin D+ port transceiver pin
USB_N	Input/ Output	N Pin D– port transceiver pin

Note: Either of RDI_REFCLK_IN or UCLK is used for 48MHz clock of the USB.

19.3 Register Descriptions

The USB host controller (USBH) has the following registers. For details on register addresses and register states during each processing, refer to section 27, List of Registers.

- HcRevision register (HREVR)
- HcControl register (HCTLR)
- HcCommandStatus register (HCSR)
- HcInterruptStatus register (HISR)
- HcInterruptEnable register (HIER)
- HcInterruptDisable register (HIDR)
- HcHCCA register (HHCCAR)
- HcPeriodCurrentED register (HPCEDR)
- HcControlHeadED register (HCHEDR)
- HcControlCurrentED register (HCCEDR)
- HcBulkHeadED register (HBHEDR)
- HcBulkCurrentED register (HBCEDR)
- HcDoneHeadED register (HDHEDR)
- HcFmInterval register (HFIR)
- HcFmRemaining register (HFRR)
- HcFmNumber register (HFNR)
- HcPeriodicStart register (HPSR)
- HcLSThreshold register (HLSTR)
- HcRhDescriptorA register (HRDRA)
- HcRhDescriptorB register (HRDRB)
- HcRhStatus register (HRSR)
- HcRhPortStatus1 register (HRPSR)

19.3.1 HcRevision Register (HREVR)

HREVR includes the BCD expression of the HCI specification version implemented for the host controller. The value H'10 corresponds to the version 1.0. All HCI implementation conforming to this specification have the value of H'10.

Bit	Bit Name	Initial Value	R/W	Description
31 to 8	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
7	REV7	0	R	Revision
6	REV6	0	R	Version Number (BCD Expression)
5	REV5	0	R	H'10: Version 1.0
4	REV4	1	R	
3	REV3	0	R	
2	REV2	0	R	
1	REV1	0	R	
0	REV0	0	R	

19.3.2 HcControl Register (HCTLR)

HCTLR specifies the operation mode for the host controller.

Bit	Bit Name	Initial Value	R/W	Description
31 to 9	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
8	IR	0	R/W	Interrupt Routine Determines the issue of interrupts generated by the event registered in the HcInterruptStatus register. The host controller driver clears this bit at the same time as the hardware reset, however, does not clear at the same time as the software reset. The host controller driver uses this bit as a tag to indicate the ownership of the host controller. 0: All interrupts are issued to the interrupt mechanism of the normal host bus. 1: Interrupts are issued to SMI.
7	HCFS1	0	R/W	Host Controller Function Status
6	HCFS0	0	R/W	The host controller driver detects whether the host controller has started to transmit SOF after having read the SF bit in the HcInterruptStatus register. This bit can be changed only in the USB Suspend state by the host controller. The host controller can move from the USB Suspend state to the USB Resume state after having detected the resume signal from the downstream port. In the host controller, USB Suspend is entered after the software reset so that USB Reset is entered after the hardware reset. The former resets the route hub. 00: USB Reset 01: USB Resume 10: USB Operational 11: USB Suspend

Bit	Bit Name	Initial Value	R/W	Description
5	BLE	0	R/W	Bulk List Enable This bit is set to enable the processing of the bulk list in the next frame. The host controller checks this bit when the processing of the list has been determined. When disabling, the host controller driver can correct the list. When the HcBulkCurrentED register indicates ED* to be deleted, the host controller driver should hasten the pointer by updating the HcBulkCurrentED register before re-enabling the list processing. 0: Bulk list processing is not carried out. 1: Bulk list processing is carried out.
4	CLE	0	R/W	Control List Enable This bit is set to enable the processing of the control list in the next frame. If cleared by the host controller driver, the processing of the control list is not carried out after next SOF. The host controller must check this bit whenever the list will be processed. When disabling, the host controller driver can correct the list. When the HcControlCurrentED register indicates ED to be deleted, the host controller driver should hasten the pointer by updating the HcControlCurrentED register before re-enabling the list processing. 0: Control list processing is not carried out. 1: Control list processing is carried out.
3	IE	0	R/W	Isochronous Enable This bit is used by the host controller driver to enable/disable the processing of isochronous ED. While processing the periodic list in the frame, the host controller will check the status of this bit when it finds an isochronous ED (F = 1). If set (enabled), the host controller continues to process ED. If cleared (disabled), the host controller stops the processing of the periodic list (currently includes only isochronous ED) and starts to process the bulk/control list. Setting this bit is guaranteed to be valid in the next frame (not in the current frame). 0: Isochronous ED processing is not carried out. 1: Isochronous ED processing is carried out.

Bit	Bit Name	Initial Value	R/W	Description
2	PLE	0	R/W	Periodic List Enable This bit is set to enable the processing of the periodic list in the next frame. If cleared by the host controller driver, no periodic list processing is carried out after next SOF. The host controller must check this bit before the host controller starts to process the list. 0: The periodic list processing is not carried out after next SOF. 1: The periodic list processing is carried out after next SOF.
1	CBSR1	0	R/W	Control Bulk Service Ratio
0	CBSR0	0	R/W	This bit specifies the service ratio of the control and bulk ED. The host controller must compare the ratio specified by the internal calculation whether it has processed control ED in determining whether another control ED is continued to be supplied or switched to bulk ED before any periodic list is processed. The internal calculation is retained when the frame boundary is exceeded. In case of reset, the host controller driver is responsible for restoring this value. 00: 1:1 01: 2:1 10: 3:1 11: 4:1

Note: * ED is Endpoint Descriptor.

19.3.3 HcCommandStatus Register (HCSR)

HCSR is used by the host controller not only for reflecting the current status of the host controller, but also for receiving a command issued by the host controller driver. A write is for setting the host controller driver. The host controller must guarantee that the bit to which 1 is written to is set and the bit to which 0 is written to is unchanged. The host controller driver can distribute multiple clear commands to the host controller by a previously issued command. The host controller driver can read all bits normally.

The SOC0 and SOC1 bits indicate the number of the frame that has detected the SchedulingOverrun error by the host controller. This occurs when the periodic list has not completed before EOF. When the SchedulingOverrun error is detected, the host controller increments the counter and sets the SO field in the HcInterruptStatus register.

Bit	Bit Name	Initial Value	R/W	Description
31 to 18	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
17	SOC1	0	R/W	Scheduling Overrun Count
16	SOC0	0	R/W	These bits are incremented when each SchedulingOverrun error occurs. These bits are initially set to B'00 and returned to B'11. These bits are incremented when a SchedulingOverrun error is detected even though the SO bit in the HcInterruptStatus register is set. These bits are used by the host controller driver to monitor any continuous scheduling problem.
15 to 4	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
3	OCR	0	R/W	Ownership Change Request This bit is set by the OS host controller driver to request the change of the control of the host controller. When this bit is set, the host controller sets the OC bit in the HcInterruptStatus register. After a change, this bit is cleared and remains until the next request from the OS host controller driver. 0: After a change, this bit is cleared and remains until the next request from the OS host controller driver. 1: Set the OC bit in the HcInterruptStatus register.

Bit	Bit Name	Initial Value	R/W	Description
2	BLF	0	R/W	Bulk List Filled This bit is used to indicate that there are some TDs in the list. This bit is set by the host controller driver to the list when TD* is added to ED. When the host controller starts to process the head of the list, it checks this bit. As long as this bit is 0, the host controller does not start to process the list. When this bit is 1, the host controller starts to process the list to set this bit to 0. When the host controller detects TD in the list, the host controller sets this bit to 1. When TD is never found in the list and the host controller driver does not set this bit, the host controller completes the processing of the list. This bit is still 0 when the bulk list processing is stopped. 0: The list is not processed. 1: The list is processed.
1	CLF	0	R/W	Control List Filled This bit is used to indicate that there are some TDs in the control list. This bit is set by the host controller driver when TD is added to ED in the control list. When the host controller starts to process the head of the control list, it checks this bit. As long as this bit is 0, the host controller does not start to process the control list. If this bit is 1, the host controller starts to process the control list and this bit is set to 0. When the host controller detects TD in the list, the host controller sets this bit to 1. When TD is never detected in the control list and the host controller driver does not set this bit, the host controller completes the processing of the control list. This bit is still 0 when the control list processing is stopped. 0: The list is not processed. 1: The list is processed.

Bit	Bit Name	Initial Value	R/W	Description
0	HCR	0	R/W	Host Controller Request This bit is set by the host controller driver to initiate the software reset of the host controller. The system is moved to the USB Suspend state in which most of the operational registers are reset except for the next state regardless of the functional state of the host controller. For example, an access without the IR field in the HcControl register and host bus are allowed. This bit is cleared by the host controller upon completion of the reset operation. The reset operation must be completed within 10 μs. When this bit is set, it does not cause any reset to the route hub and the next reset signal is not output to the downstream port. 0: Cleared by the host controller at the completion of the reset operation 1: USB Suspend state

Note: * TD is Transfer Descriptor.

19.3.4 HcInterruptStatus Register (HISR)

HISR indicates the status in various events that cause hardware interrupts. When an event occurs, the host controller sets the corresponding bit in this register. When the bit is set to 1, a hardware interrupt is generated while an interrupt is enabled and the MIE bit is set in the HcInterruptEnable register (see section 19.3.5, HcInterruptEnable Register (HIER)). When an interrupt occurs, the exception code (H'A00) of the host controller interrupt (USBHI) is set in the interrupt event register 2 (INTEVT2) and the exception handling is started. For details on the interrupt event register 2 (INTEVT2), see section 4, Exception Handling, and for details on the exception code and interrupt processing of the host controller interrupt (USBHI), see section 8, Interrupt Controller (INTC).

The host controller driver clears a specified bit in this register by writing 1 in the bit position to be cleared. The host controller driver cannot set any of these bits. The host controller never clears bits.

Bit	Bit Name	Initial Value	R/W	Description
31	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to this bit.
30	OC	0	R/W	Ownership Change This bit is set when the OCR bit in the HcCommandStatus register is set by the host controller driver. This event generates a system management interrupt (SMI) at once when not masked. When there is no SMI pin, this bit is cleared to 0. 0: The OCR bit in the HcCommandStatus register is not set. 1: The OCR bit in the HcCommandStatus register is set.
29 to 7	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
6	RHSC	0	R/W	Root Hub Status Change This bit is set when the content of the HcRhStatus register or the content of any HcRhPortStatus register [Number of Downstream Port] has changed. 0: The content of the HcRhStatus register or HcRhPortStatus register is not changed. 1: The content of the HcRhStatus register or HcRhPortStatus register is changed.
5	FNO	0	R/W	Frame Number Overflow This bit is set when MSB (bit 15) in the HcFmNumber register changes value from 0 to 1 or from 1 to 0 or after HccaFrameNumber is updated. 0: MSB in the HcFmNumber register or HccaFrameNumber is not updated. 1: MSB in the HcFmNumber register or HccaFrameNumber is updated.
4	UE	0	R/W	Unrecoverable Error This bit is set when the host controller detects a system error that is not related to USB. The host controller driver clears this bit after the host controller is reset. 0: System error has not generated yet. 1: System error is detected.

Bit	Bit Name	Initial Value	R/W	Description
3	RD	0	R/W	Resume Detected This bit is set when the host controller detects that the USB device outputs a resume signal. This bit is not set when the host controller driver sets the USB Resume state. 0: The resume signal is not detected. 1: The resume signal is detected.
2	SF	0	R/W	Start of Frame This bit is set by the host controller when each frame starts and after HccaFrameNumber is updated. The host controller simultaneously generates the SOF token. 0: Each frame has not initiated or HccaFrameNumber is not updated. 1: Initiation of each frame and updating of HccaFrameNumber
1	WDH	0	R/W	Write back Done Head This bit is set immediately after the host controller has written HcDoneHead to HccaDoneHead. HccaDoneHead is not updated until this bit is cleared. The host controller driver should clear this bit only after the content of HccaDoneHead has been stored. 0: When cleared after set to 1. 1: When HcDoneHead is written to HccaDoneHead.
0	SO	0	R/W	Scheduling Overrun This bit is set when the USB schedule has overrun after HccaFrameNumber has updated in the current frame. SchedulingOverrun also increments the SOC0 and SOC1 bits in the HcCommandStatus register. 0: The USB schedule has not overrun. 1: The USB schedule has overrun.

19.3.5 HcInterruptEnable Register (HIER)

Each enable bit in HIER controls whether to generate an interrupt by an event related to the HcInterruptStatus register. A hardware interrupt is requested to the CPU when an event bit in the HcInterruptStatus register is set, a corresponding bit in the HcInterruptEnable register is set, and the MasterInterruptEnable (MIE) bit is set. As a result, the exception code (H'A00) of the host controller interrupt (USBHI) is set in the interrupt event register 2 (INTEVT2) and the exception handling is started (the USBHI exception code is used in common regardless of the content of the interrupt generation event). Therefore, the USBHI exception code can be used when an interrupt generation is detected by the host controller driver. For details on the interrupt event register 2 (INTEVT2), see section 4, Exception Handling, and for details on the exception code and interrupt processing of the host controller interrupt (USBHI), see section 8, Interrupt Controller (INTC).

Writing 1 in this register sets the corresponding bit, while writing 0 does not change it. When the specified bit needs to be set, writing data in which only the corresponding bit is 1 enables the bit to be set independently without checking the states of other bits.

HIER is in conjunction with the HcInterruptDisable register which is described later. Therefore when a specific bit needs to be cleared to 0, writing 1 to the corresponding bit in the HcInterruptDisable register automatically clears the corresponding bit in this register.

Bit	Bit Name	Initial Value	R/W	Description
31	MIE	0	R/W	Master Interrupt Enable When this bit is set to 1, an interrupt generation by the event specified in another bit in this register is enabled. This is used by the host controller driver so that the master interrupt is enabled. When 1 is written to the MIE bit in the HcInterruptDisable register, this bit is cleared to 0. 0: This bit is ignored. 1: Enables interrupt generation due to the event specified by other bit.
30	OC	0	R/W	Ownership Change Interrupt Enable When this bit is set to 1, an interrupt generation by the Ownership Change event is enabled. When 1 is written to the OC bit in the HcInterruptDisable register, this bit is cleared to 0. 0: This bit is ignored. 1: Enables interrupt generation due to the Ownership Change event.

Bit	Bit Name	Initial Value	R/W	Description
29 to 7	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
6	RHSC	0	R/W	Root Hub Status Change Interrupt Enable When this bit is set to 1, an interrupt generation by the Root Hub Status Change event is enabled. When 1 is written to the RHSC bit in the HcInterruptDisable register, this bit is cleared to 0. 0: This bit is ignored. 1: Enables interrupt generation due to the Root Hub Status Change event.
5	FNO	0	R/W	Frame Number Overflow Interrupt Enable When this bit is set to 1, an interrupt generation by the Frame Number Overflow event is enabled. When 1 is written to the FNO bit in the HcInterruptDisable register, this bit is cleared to 0. 0: This bit is ignored. 1: Enables interrupt generation due to the Frame Number Overflow event.
4	UE	0	R/W	Unrecoverable Error Interrupt Enable When this bit is set to 1, an interrupt generation by the Unrecoverable Error event is enabled. When 1 is written to the UE bit in the HcInterruptDisable register, this bit is cleared to 0. 0: This bit is ignored. 1: Enables interrupt generation due to the Unrecoverable Error event.
3	RD	0	R/W	Resume Detected Interrupt Enable When this bit is set to 1, an interrupt generation by the Resume Detected event is enabled. When 1 is written to the RD bit in the HcInterruptDisable register, this bit is cleared to 0. 0: This bit is ignored. 1: Enables interrupt generation due to the Resume Detected event.

Bit	Bit Name	Initial Value	R/W	Description
2	SF	0	R/W	Start of Frame Interrupt Enable When this bit is set to 1, an interrupt generation by the Start of Frame event is enabled. When 1 is written to the SF bit in the HcInterruptDisable register, this bit is cleared to 0. 0: This bit is ignored. 1: Enables interrupt generation due to the Start of Frame event.
1	WDH	0	R/W	WriteBack Done Head Interrupt Enable When this bit is set to 1, an interrupt generation by the WriteBack Done Head event is enabled. When 1 is written to the WDH bit in the HcInterruptDisable register, this bit is cleared to 0. 0: This bit is ignored. 1: Enables interrupt generation due to the WriteBack Done Head event.
0	SO	0	R/W	Scheduling Overrun Interrupt Enable When this bit is set to 1, an interrupt generation by the SchedulingOverrun event is enabled. When 1 is written to the SO bit in the HcInterruptDisable register, this bit is cleared to 0. 0: This bit is ignored. 1: Enables interrupt generation due to the SchedulingOverrun event.

19.3.6 HcInterruptDisable Register (HIDR)

Each disable bit in HIDR controls whether to mask an interrupt by an event related to the HcInterruptStatus register. A hardware interrupt is not requested to the CPU when an event bit in the HcInterruptStatus register is set and a corresponding bit in the HcInterruptDisable register is set or the MIE bit in the HcInterruptDisable register is set.

Writing 1 in this register clears the corresponding bit in the HcInterruptEnable register (disable state), while writing 0 does not change it. When the specified bit is disabled, writing data in which only the corresponding bit is 1 enables the bit to be disabled independently without checking the states of other bits.

HIDR is in conjunction with the HcInterruptEnable register which is described before. Therefore when the specified bit needs to be cleared to 0 (enable state), writing 1 in the corresponding bit in the HcInterruptEnable register enables the corresponding bit in this register to be automatically cleared to 0. When this register is read, the current value of the HcInterruptEnable register is returned.

Bit	Bit Name	Initial Value	R/W	Description
31	MIE	0	R/W	Master Interrupt Disable When this bit is set to 1, an interrupt generation by the event specified in another bit in this register is masked. When 1 is written to the MIE bit in the HcInterruptEnable register, this bit is cleared to 0. 0: This bit is ignored. 1: Disables interrupt generation due to the event specified by other bit.
30	OC	0	R/W	Ownership Change Interrupt Disable When this bit is set to 1, an interrupt generation by the Ownership Change event is masked. When 1 is written to the OC bit in the HcInterruptEnable register, this bit is cleared to 0. 0: This bit is ignored. 1: Disables interrupt generation due to the Ownership Change event.
29 to 7	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.

Bit	Bit Name	Initial Value	R/W	Description
6	RHSC	0	R/W	Root Hub Status Change Interrupt Disable When this bit is set to 1, an interrupt generation by the Root Hub Status Change event is masked. When 1 is written to the RHSC bit in the HcInterruptEnable register, this bit is cleared to 0. 0: This bit is ignored. 1: Disables interrupt generation due to the Root Hub Status Change event.
5	FNO	0	R/W	Frame Number Overflow Interrupt Disable When this bit is set to 1, an interrupt generation by the Frame Number Overflow event is masked. When 1 is written to the FNO bit in the HcInterruptEnable register, this bit is cleared to 0. 0: This bit is ignored. 1: Disables interrupt generation due to the Frame Number Overflow event.
4	UE	0	R/W	Unrecoverable Error Interrupt Disable When this bit is set to 1, an interrupt generation by the Unrecoverable Error event is masked. When 1 is written to the UE bit in the HcInterruptEnable register, this bit is cleared to 0. 0: This bit is ignored. 1: Disables interrupt generation due to the Unrecoverable Error event.
3	RD	0	R/W	Resume Detected Interrupt Disable When this bit is set to 1, an interrupt generation by the Resume Detected event is masked. When 1 is written to the RD bit in the HcInterruptEnable register, this bit is cleared to 0. 0: This bit is ignored. 1: Disables interrupt generation due to the Resume Detected event.
2	SF	0	R/W	Start of Frame Interrupt Disable When this bit is set to 1, an interrupt generation by the Start of Frame event is masked. When 1 is written to the SF bit in the HcInterruptEnable register, this bit is cleared to 0. 0: This bit is ignored. 1: Disables interrupt generation due to the Start of Frame event.

Bit	Bit Name	Initial Value	R/W	Description
1	WDH	0	R/W	WriteBack Done Head Interrupt Disable When this bit is set to 1, an interrupt generation by the WriteBack Done Head event is masked. When 1 is written to the WDH bit in the HcInterruptEnable register, this bit is cleared to 0. 0: This bit is ignored. 1: Disables interrupt generation due to the WriteBack Done Head event.
0	SO	0	R/W	Scheduling Overrun Interrupt Disable When this bit is set to 1, an interrupt generation by the SchedulingOverrun event is masked. When 1 is written to the SO bit in the HcInterruptEnable register, this bit is cleared to 0. 0: This bit is ignored. 1: Disables interrupt generation due to the SchedulingOverrun event.

19.3.7 HcHCCA Register (HHCCAR)

HHCCAR includes physical addresses in the host controller communication area. The host controller driver determines the alignment limitation by writing 1 to all bits in the HcHCCA register and by reading the content of the HcHCCA register. Alignment is determined by checking the number of 0 in the lower bits. The minimum alignment is 256 bytes. Consequently, bits 0 to 7 must always be 0 when they are read. This area is used to retain the control structure and interrupt table that are accessed by the host controller and host controller driver.

Bit	Bit Name	Initial Value	R/W	Description
31 to 8	HCCA	All 0	R/W	These bits are physical addresses in the Host Controller Communication Area.
7 to 0	—	All 0	R	These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.

19.3.8 HcPeriodCurrentED Register (HPCEDR)

HPCEDR includes a physical address of the current isochronous ED or interrupt ED.

Bit	Bit Name	Initial Value	R/W	Description
31 to 4	PCED	All 0	R	Period Current ED These bits are physical addresses of the current isochronous ED or interrupt ED.
3 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.

19.3.9 HcControlHeadED Register (HCHEDR)

HCHEDR includes a physical address of the first ED in the control list.

Bit	Bit Name	Initial Value	R/W	Description
31 to 4	CHED	All 0	R/W	Control Head ED These bits are physical addresses of the first ED in the control list.
3 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.

19.3.10 HcControlCurrentED Register (HCCEDR)

HCCEDR includes a physical address of the current ED in the control list.

Bit	Bit Name	Initial Value	R/W	Description
31 to 4	CCED	All 0	R/W	Control Current ED These bits are physical addresses of the current ED in the control list.
3 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.

19.3.11 HcBulkHeadED Register (HBHEDR)

HBHEDR includes a physical address of the first ED in the bulk list.

Bit	Bit Name	Initial Value	R/W	Description
31 to 4	BHED	All 0	R/W	Bulk Head ED These bits are physical addresses of the first ED in the bulk list.
3 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.

19.3.12 HcBulkCurrentED Register (HBCEDR)

HBCEDR includes a physical address of the current ED in the bulk list. When the bulk list is supplied by the round robin method, endpoints are ordered to the list according to these insertions.

Bit	Bit Name	Initial Value	R/W	Description
31 to 4	BCED	All 0	R/W	Bulk Current ED These bits are physical addresses of the current ED in the bulk list.
3 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.

19.3.13 HcDoneHeadED Register (HDHEDR)

HDHEDR includes a physical address of the finally completed TD added to the done queue. The host controller driver need not read this register so that the content is written to HCCA periodically in normal operation.

Bit	Bit Name	Initial Value	R/W	Description
31 to 4	DH	All 0	R	Done Head These bits are physical addresses of the finally completed TD added to the done queue.

Bit	Bit Name	Initial Value	R/W	Description
3 to 0	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.

19.3.14 HcFmInterval Register (HFIR)

HFIR includes a 14-bit value indicating the bit time interval of the frame (i.e., between two serial SOFs) and a 15-bit value indicating the maximum packet size at a full speed that is transmitted and received by the host controller without causing SchedulingOverrun. The host controller driver adjusts the frame interval minutely by writing a new value over the current value in each SOF.

Bit	Bit Name	Initial Value	R/W	Description
31	FIT	0	R/W	Frame Interval Toggle This bit is toggled by the host controller driver whenever it loads a new value into the Frame Interval bits.
30	FSMPS14	0	R/W	FSLargest Data Packet These bits specify a value which is loaded into the Largest Data Packet Counter at the beginning of each frame. The counter value expresses the largest data amount of the bit that can be transmitted and received in one transaction by the host controller at any given time without causing SchedulingOverrun. The field value is calculated by the host controller driver.
29	FSMPS13	0	R/W	
28	FSMPS12	0	R/W	
27	FSMPS11	0	R/W	
26	FSMPS10	0	R/W	
25	FSMPS9	0	R/W	
24	FSMPS8	0	R/W	
23	FSMPS7	0	R/W	
22	FSMPS6	0	R/W	
21	FSMPS5	0	R/W	
20	FSMPS4	0	R/W	
19	FSMPS3	0	R/W	
18	FSMPS2	0	R/W	
17	FSMPS1	0	R/W	
16	FSMPS0	0	R/W	
15, 14	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.

Bit	Bit Name	Initial Value	R/W	Description
13	FI13	1	R/W	Frame Interval
12	FI12	0	R/W	These bits specify the interval between two serial SOFs with bit times. The nominal value is set to H'2EDF (11999).
11	FI11	1	R/W	
10	FI10	1	R/W	
9	FI9	1	R/W	The host controller driver must store the current value of this field before resetting the host controller. With this procedure, this bit is reset to its nominal value by the host controller by setting the HCR bit in the HcCommandStatus register. The host controller driver can select to restore the stored value at the completion of the reset sequence.
8	FI8	0	R/W	
7	FI7	1	R/W	
6	FI6	1	R/W	
5	FI5	0	R/W	
4	FI4	1	R/W	
3	FI3	1	R/W	
2	FI2	1	R/W	
1	FI1	1	R/W	
0	FI0	1	R/W	

19.3.15 HcFmRemaining Register (HFRR)

HFRR is a 14-bit down counter indicating the bit time remaining in the current frame.

Bit	Bit Name	Initial Value	R/W	Description
31	FRT	0	R	Frame Remaining Toggle This bit is always loaded from the FIT bit in the HcFmInterval register when the FR bits reach 0. This bit is used by the host controller driver for synchronization between the FrameInterval and FrameRemaining registers.
30 to 14	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
13	FR13	0	R	Frame Remaining These counters are decremented at each bit time. When these counters reach 0, these counters are reset by loading the value of the Frame Interval bit specified in the HcFmInterval register at the next bit time boundary. When the host controller transits to the USB Operational state, it reads the FI bits in the HcFmInterval register again and uses the updated value from the next SOF.
12	FR12	0	R	
11	FR11	0	R	
10	FR10	0	R	
9	FR9	0	R	
8	FR8	0	R	
7	FR7	0	R	
6	FR6	0	R	
5	FR5	0	R	
4	FR4	0	R	
3	FR3	0	R	
2	FR2	0	R	
1	FR1	0	R	
0	FR0	0	R	

19.3.16 HcFmNumber Register (HFNR)

HFNR is a 16-bit counter. It indicates the reference of timing between events occurring in the host controller and host controller driver. The host controller driver uses a 16-bit value specified in this register and generates a 32-bit frame number without necessity for a frequent access to the register.

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
15	FN15	0	R	Frame Number These bits are incremented when the HcFmRemaining register is reloaded. The count will return to H'0 after reaching H'FFFF. When the host controller transits to the USB Operational state, these bits are automatically incremented. After the host controller increments the FN15 to FN0 bits and sends SOF in each frame boundary, the content is written to HCCA before the host controller reads the first ED in the frame. After writing to HCCA, the host controller sets the SF bit in the HcInterruptStatus register.
14	FN14	0	R	
13	FN13	0	R	
12	FN12	0	R	
11	FN11	0	R	
10	FN10	0	R	
9	FN9	0	R	
8	FN8	0	R	
7	FN7	0	R	
6	FN6	0	R	
5	FN5	0	R	
4	FN4	0	R	
3	FN3	0	R	
2	FN2	0	R	
1	FN1	0	R	
0	FN0	0	R	

19.3.17 HcPeriodicStart Register (HPSR)

HPSR has a 14-bit programmable value, which determines the earliest time when the host controller should start to process the periodic list.

Bit	Bit Name	Initial Value	R/W	Description
31 to 14	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
13	PS13	0	R/W	Periodic Start This field is cleared after the hardware reset. Then this field is set by the host controller driver while the host controller performs initial settings. This value is roughly calculated as 10% off from the HcFmInterval register. The normal value is H '2A2F. When the HcFmRemaining register reaches the specified value, the processing of the periodic list has a higher priority than the control/bulk processing. Consequently, the host controller starts to process the periodic list after the completion of the current control/bulk transaction.
12	PS12	0	R/W	
11	PS11	0	R/W	
10	PS10	0	R/W	
9	PS9	0	R/W	
8	PS8	0	R/W	
7	PS7	0	R/W	
6	PS6	0	R/W	
5	PS5	0	R/W	
4	PS4	0	R/W	
3	PS3	0	R/W	
2	PS2	0	R/W	
1	PS1	0	R/W	
0	PS0	0	R/W	

19.3.18 HcLSThreshold Register (HLSTR)

HLSTR includes an 12-bit value that is used by the host controller to determine whether or not to authorize the transfer of the LS packed 8 bytes in maximum before EOF. The host controller and host controller driver cannot change this value.

Bit	Bit Name	Initial Value	R/W	Description
31 to 12	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
11	LST11	0	R/W	LSThreshold This field contains a value to be compared with the FR bits prior to the beginning of low-speed transaction. The transaction is started only when the FR bit value is beyond the value of this bit. The value is calculated by the host controller driver considering the transmission and set-up overhead.
10	LST10	1	R/W	
9	LST9	1	R/W	
8	LST8	0	R/W	
7	LST7	0	R/W	
6	LST6	0	R/W	
5	LST5	1	R/W	
4	LST4	0	R/W	
3	LST3	1	R/W	
2	LST2	0	R/W	
1	LST1	0	R/W	
0	LST0	0	R/W	

19.3.19 HcRhDescriptorA Register (HRDRA)

HRDRA is the first register of two registers describing the features of the root hub. The descriptor length (11), descriptor type (T.B.D), and the hub controller current bit (0) of the class descriptor of the hub are emulated by the host controller driver. All other bits are placed in the HcRhDescriptorA register and HcRhDescriptorB register.

Bit	Bit Name	Initial Value	R/W	Description
31	POTPGT7	0	R/W	Power On To Power Good Time
30	POTPGT6	0	R/W	These bits specify the waiting time until a host controller driver access the power-on port of the root hub. These bits are implementation specific. The unit of time is 2 ms. The time is calculated as POTPGT × 2 ms.
29	POTPGT5	0	R/W	
28	POTPGT4	0	R/W	
27	POTPGT3	0	R/W	
26	POTPGT2	0	R/W	
25	POTPGT1	1	R/W	
24	POTPGT0	0	R/W	
23 to 13	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
12	NOCP	1	R/W	No Over Current Protection This bit selects how the overcurrent status of the root hub port is reported. When this bit is cleared, the OCPM bit specifies global report or report at each port. 0: Overcurrent status is collectively reported for all downstream ports. 1: Overcurrent protection is not supported.
11	OCPM	0	R/W	Over Current Protection Mode This bit selects how the overcurrent status in the root-hub port is reported. At reset, this bit reflects the same mode as the PSM bit. When the NOCP bit is cleared, this bit is valid. 0: Overcurrent status is collectively reported for all downstream ports. 1: Overcurrent status is reported for each port.
10	DT	0	R	DeviceType This bit indicates that the route hub is not a compound device. This bit should always be set to 0.

Bit	Bit Name	Initial Value	R/W	Description
9	NPS	1	R/W	No Power Switching This bit selects whether the power switching is supported or ports are always power-supplied. This bit is implementation specific. When this bit is cleared, the PSM bit specifies the global/port switching. Note: Because the initial value is 1, first clear this bit (write 0 with the host controller driver) to enable power switching of the port. 0: Ports can be power-switched. 1: Ports are always powered on when the host controller is powered on.
8	PSM	0	R/W	Power Switching Mode This bit specifies how the power switching of the root-hub port is controlled. This bit is implementation specific. This bit is valid only when the NPS bit is cleared. 0: All ports are simultaneously power-supplied. 1: Each port is power-supplied individually. In this mode, the port power is controlled with either of global/port switching. When the PPCM bit in the HcRhDescriptorB register is set, the port is reacted only to the port-power command (set/clear port power). When the port mask is cleared, the port is controlled only by the global power-switch (set/clear global power).
7	NDP7	0	R	Number Downstream Ports These bits specify the number of downstream ports supported by the root hub. These bits are implementation specific. The minimum number of ports is 1. The maximum number of ports which are supported by the OpenHCI is 15.
6	NDP6	0	R	
5	NDP5	0	R	
4	NDP4	0	R	
3	NDP3	0	R	
2	NDP2	0	R	
1	NDP1	1	R	
0	NDP0	0	R	

19.3.20 HcRhDescriptorB Register (HRDRB)

HRDRB is the second register of two registers describing the features of the root hub. These bits must be set during the initial setting so as to correspond to the system implementation.

Bit	Bit Name	Initial Value	R/W	Description
31 to 18	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
17	PPCM	0	R/W	Port Power Control Mask This bit indicates whether the port is influenced by the global power-control command when the PSM bit in the HcRhDescriptorA register is set. When this bit is set, the power state of the port is affected by the power control at each port (set/clear port power). When this bit is cleared, the port is controlled by the global power switch (set/clear global power). If the device is placed in global switching mode (PSM bit = 0), this bit is not valid. Note: Clear the NPS bit in the HcRhDescriptorA register so that the power to all ports is off (PortPowerStatus = 0), then set this bit. 0: Ports are controlled by the global power switch. 1: Port#1 is affected by the power control at each port.
16 to 2	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
1	DR	0	R/W	Device Removable This bit is dedicated to the ports of the root hub. When this bit is cleared, the set device becomes removable. When this bit is set, do not remove the set device. 0: Device attached to port#1 is removable. 1: Device attached to port#1 is not removable.
0	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to this bit.

19.3.21 HcRhStatus Register (HRSR)

HRSR is divided into two parts. The lower word of a longword indicates the hub status bits and the upper word indicates the hub status change bit. The reserved bits should be set to 0.

Bit	Bit Name	Initial Value	R/W	Description
31	CRWE	0	W	Clear Remote Wakeup Enable Writing 1 clears this bit. When this bit is set to 0, this bit is not cleared.
30 to 18	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
17	OCIC	0	R/W	Over Current Indicator Change When the OCI field in this register is changed, this bit is set by the host controller. Writing 1 by the host controller driver clears this bit. When this bit is set to 0, this bit is not cleared.
16	LPSC	0	R/W	(Read) Local Power Status Change The root hub does not support the local power status function. Therefore, this bit is always read as 0. (Write) Set Global Power This bit is written to 1 to power on (sets the PPS bit in the HcRhPortStatus register) all ports in global power mode (PSM bit in the HcRhDescriptorA register = 0). This bit sets the PPS bit only to the port in which the PPCM bit in the HcRhDescriptorB register is not set in power mode at each port. When 0 is written to, this bit is not cleared.
15	DRWE	0	R/W	(Read) Device Remote Wakeup Enable This bit enables the CSC bit in the HcRhPortStatus register as a resume event and generates the state transition from USB Suspend to USB Resume and the Resume Detected interrupt. 0: Connect Status Change is not the Remote Wakeup event. 1: Connect Status Change is the Remote Wakeup event. (Write) Set Remote Wakeup Enable Writing 1 sets this bit. When 0 is written to, this bit is not set.

Bit	Bit Name	Initial Value	R/W	Description
14 to 2	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
1	OCI	0	R/W	Over Current Indicator This bit reports the overcurrent condition when the global report is performed. When this bit is set, an overcurrent condition exists. When this bit is cleared, all power operations are normal. This bit is always 0 when the overcurrent protection at each port is carried out. 0: All power operations are normal. 1: An overcurrent condition exists.
0	LPS	0	R/W	(Read) Local Power Status The root hub does not support the local power status function. Therefore, this bit is always read as 0. (Write) Clear Global Power This bit is written to 1 to power off (clears the PPS bit in the HcRhPortStatus register) all ports in global power mode (PSM bit in the HcRhDescriptorA register = 0). In power mode at each port, the PPS bit is cleared to the port in which the PPCM bit in the HcRhDescriptorB register is not set. When 0 is written to, this bit is invalid.

19.3.22 HcRhPortStatus Register (HRPSR)

HRPSR is used for base-controlling each port and to report the port event. The lower word is used to reflect the port status while the upper word reflects the status change. Some status bits have special writing (see below). If the transaction (token that passes the hand shake) is in progress while a write to ChangePortStatus occurs, the resultant port status change must be postponed until the completion of the transaction. Always write reserved bits to 0.

Bit	Bit Name	Initial Value	R/W	Description
31 to 21	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
20	PRSC	0	R/W	Port Reset Status Change This bit is set when the 10 ms port reset signal has completed. Writing 1 by the host controller driver clears this bit. When this bit is written to 0, this bit is not cleared. 0: Port reset is not completed. 1: Port reset is completed.
19	OCIC	0	R/W	Port Over Current Indicator Change This bit is valid when an overcurrent condition is reported on the base of each port. This bit is set when the root hub changes the POCI bit. Writing 1 by the host controller driver clears this bit. When this bit is written to 0, this bit is not cleared. 0: Port Over Current Indicator has not changed. 1: Port Over Current Indicator has changed.
18	PSSC	0	R/W	Port Suspend Status Change This bit is set when all resume sequences have completed. These sequences include 20 ms resume pulse, LS EOP, and 3 ms resychronization delay. Writing 1 by the host controller driver clears this bit. When this bit is written to 0, this bit is not cleared. When the PRSC bit is set, this bit is cleared. 0: Resume sequence has not completed. 1: Resume sequence has completed.

Bit	Bit Name	Initial Value	R/W	Description
17	PESC	0	R/W	Port Enable Status Change This bit is set when the PES bit is cleared due to a hardware event. This bit is not set by the change of writing of the host controller driver. Writing 1 by the host controller driver clears this bit. When this bit is written to 0, this bit is not cleared. 0: Port Enable Status has not changed. 1: Port Enable Status has changed.
16	CSC	0	R/W	Connect Status Change This bit is set whenever the connection or disconnection event occurs. Writing 1 by the host controller driver clears this bit. When this bit is written to 0, this bit is not cleared. If the CCS bit is cleared when Set Port Reset, Set Port Enable, or Set Port Suspend is written to, writing when the power supply of the port is disconnected does not occur, so this bit is set to report the driver to re-evaluate the connection status. Note: If the DR bit in the HcRhDescriptorB register is set, this bit is set only after the root hub reset to report the system that a device can be attached. 0: Current Connect Status has not changed. 1: Current Connect Status has changed.
15 to 10	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.
9	LSDA	0	R/W	(Read) Low Speed Device Attached This bit indicates the speed of the device attached to this port. When this bit is set, a low-speed device is attached to this port. When this bit is cleared, a full-speed device is attached to this port. This bit is valid only when the CCS bit is set. 0: A full-speed device is attached. 1: A low-speed device is attached. (Write) Clear Port Power Writing 1 by the host controller driver clears the PPS bit. When this bit is written to 0, this bit is not cleared.

Bit	Bit Name	Initial Value	R/W	Description
8	PPS	1	R/W	(Read) Port Power Status This bit reflects the power status of the port regardless of power switching mode to be executed. However, because the initial value of the No Power Switching bit in the HcRhDescriptorA register is 1, this bit is first fixed to 1. The No Power Switching bit must first be cleared when power is switched, as shown below. When an overcurrent condition is detected, this bit is cleared. Writing Set Port Power or Set Global Power by the host controller driver sets this bit. Also, writing Clear Port Power or Clear Global Power by the host controller driver clears this bit. The PSM bit in the HcRhDescriptorA register and the PPCM bit in the HcRhDescriptorB register determine which power control switch can be used. Only Set/Clear Global Power controls this bit in global switching mode (PSM bit = 0). If the PPCM bit of that port is set in power switching mode (PSM bit = 1), only the Set/Clear Port Power command is enabled. If the mask is not set, only the Set/Clear Global Power command is enabled. When the port power is disabled, the CCS, PES, PSS, and PRS bits are reset. Note: If power switching mode is not supported, this bit is always read as 1. 0: Port power is off. 1: Port power is on. (Write) Set Port Power Writing 1 by the host controller driver sets the PPS bit. When this bit is written to 0, this bit is not set.
7 to 5	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. The operation is not guaranteed if 1 is written to these bits.

Bit	Bit Name	Initial Value	R/W	Description
4	PRS	0	R/W	(Read) Port Reset Status When this bit is set by writing to Set Port Reset, the port reset signal is output. This bit is cleared when the PRSC bit is set upon completion of a reset. When the CCS bit is cleared, this bit is not set. 0: Port reset signal is not active. 1: Port reset signal is active. (Write) Set Port Reset Writing 1 by the host controller driver sets the port reset signal. When this bit is written to 0, this bit is not set. When the CCS bit is cleared, this write does not set the PRS bit, instead, sets the CSC bit. This reports a reset of the disconnection port to the driver.
3	POCI	0	R/W	(Read) Port Over Current Indicator This bit is valid only when a root hub is placed in such a way that an overcurrent condition is reported on the base of each port. If the overcurrent report at each port is not supported, this bit is cleared to 0. If this bit is cleared, all power controls are normal in this port. If this bit is set, an overcurrent state exists in this port. This bit always reflects an overcurrent input signal. 0: No overcurrent state. 1: Overcurrent state is detected. (Write) Clear Suspend Status Writing 1 by the host controller driver initiates a resume. When this bit is written to 0, this bit is invalid. If the PSS bit is set, a resume is initiated.

Bit	Bit Name	Initial Value	R/W	Description
2	PSS	0	R/W	(Read) Port Suspend Status This bit indicates that the port is under the process to suspend or during the resume sequence. Writing Set Suspend State sets this bit and setting the PSSC bit clears this bit at the end of the resume interval. If the CCS bit is cleared, this bit cannot be set. When the PRSC bit is set upon completion of the port reset or the host controller is placed in the USB resume state, this bit is cleared. If an upstream resume is in progress, it is transmitted to the host controller. 0: Port is not suspended. 1: Port is suspended. (Write) Set Port Suspend Writing 1 by the host controller driver sets the PSS bit. When this bit is written to 0, this bit is not set. In addition, when the CCS bit is cleared, the PSS bit is not set by this writing. Instead, the CSC bit is set. This reports the suspended state of the disconnection port to the driver.
1	PES	0	R/W	(Read) Port Enable Status This bit indicates whether the port is enabled or disabled. The root hub clears this bit when the overcurrent condition and an operational bus error such as disconnect event, power-off switch, or babble are detected. The PESC bit is set by this change. This bit is set by writing Set Port Enable and cleared by writing Clear Port Enable. This bit cannot be set when the CCS bit is cleared. In addition, this bit is set upon completion of the port reset by which the PRSC bit is set, or upon completion of the port suspend by which the PSSC bit is set. 0: Port disabled. 1: Port enabled. (Write) Set Port Enable Writing 1 by the host controller driver sets the PES bit. When this bit is written to 0, this bit is not set. If the CCS bit is cleared, this writing does not set the PES bit, instead, sets the CSC bit. This reports the driver that the disconnection port has been tried to be enabled.

Bit	Bit Name	Initial Value	R/W	Description
0	CCS	0	R/W	(Read) Current Connect Status This bit indicates the current status of the downstream port. Note: If the set device is not removable (DeviceRemovable), this bit is always read as 1. 0: No device connected. 1: Device connected. (Write) Clear Port Enable Writing 1 by the host controller driver clears the PES bit. When this bit is written to 0, this bit is not cleared. The CCS bit is not affected by any writing.

19.4 Data Storage Format of USB Host Controller

19.4.1 Storage Format of Transferred Data

The data storage format (endian) of the USB host controller is in conjunction with the CPU endian specification by the MD5 pin. Therefore, when the CPU is operated as big endian, transfer data of the USB host controller must be treated in big-endian order. When the CPU is operated as little endian, transfer data of the USB host controller must be treated in little-endian order.

19.4.2 Storage Format of Descriptor

ED (endpoint descriptor) and TD (transfer descriptor) that define each transfer transaction of the USB host controller must be aligned so that each Dword corresponds to the longword boundary (addresses 4n to 4n + 3) of the memory. In this case, the difference of endian has no influence.

19.5 Usage Restrictions of USB Host Controller

19.5.1 Restriction of Reset Control

Following procedure should be used when the software reset is issued (the HCR bit of the HCSR register is set to 1) while USB state is Operational, or when the USB state is shifted from the Operational state to the Reset state (the HCFS1/0 bit of the HCTLR register is cleared to 00).

1. Each list processing is discontinued. (The BLE bit, the CLE bit, the IE bit, and the PLE bit of the HCTLR register are cleared to 0.)
2. Wait until the SOF interrupt from the USB host controller is detected twice. (SF bit interrupt of HISR register)
3. Put the USB into the Suspend state by setting the HCFS1/0 bits of the HCTLR register to 11.
4. The reset processing is executed.

Section 20 USB Function Controller (USBF)

This LSI incorporates an USB function controller (USBF).

20.1 Features

- UDC (USB device controller) supporting USB 1.1 processes incorporated USB protocol automatically.

  Automatic processing of USB standard commands for endpoint 0 (some commands and class/vendor commands require decoding and processing by firmware)
- Transfer speed: Full-speed

An arbitrary endpoint configuration can be set by specifying the endpoint number used by the USB host and the endpoint FIFO number (transfer method and transfer direction are fixed) provided by this USB function controller via the EPIRn0 to EPIRn5 registers.

Endpoint Name	Abbreviation	Endpoint FIFO Number	Transfer Type	Maximum Packet Size	FIFO Buffer Capacity (Byte)	DMA Transfer	Application
Endpoint 0	EP0s	H'00	Setup	8	8	—	Emulation
	EP0i	H'00	Control-in	64	64	—	
	EP0o	H'00	Control-out	64	64	—	
Endpoint 1	EP1	H'01	Interrupt-in	16	16	—	Modem, packet
Endpoint 2i	EP2i	H'02	Bulk-in	64	128	Possible	
Endpoint 2o	EP2o	H'03	Bulk-out	64	128	Possible	
Endpoint 3i	EP3i	H'05	Isochronous-in	64	128	—	AT command control, audio
Endpoint 3o	EP3o	H'06	Isochronous-out	60	120	—	
Endpoint 4	EP4	H'07	Interrupt-in	16	16	—	SD/MMC memory card
Endpoint 5	EP5	H'08	Bulk-in	32	64	Possible	
Endpoint 6	EP6	H'09	Bulk-out	32	64	Possible	

- Interrupt requests: generates various interrupt signals necessary for USB transmission/reception
- Clock: Selection of internal system clock/external input (48 MHz) by means of EXCPG
- Power-down mode

  Power consumption can be reduced by stopping UDC internal clock when USB cable is disconnected

  Automatic transition to/recovery from suspend state
- Power mode: Self-powered, bus-powered

- Endpoint configuration: An arbitrary endpoint configuration can be set. (Endpoint configuration based on the Bluetooth specification standard is also supported.)

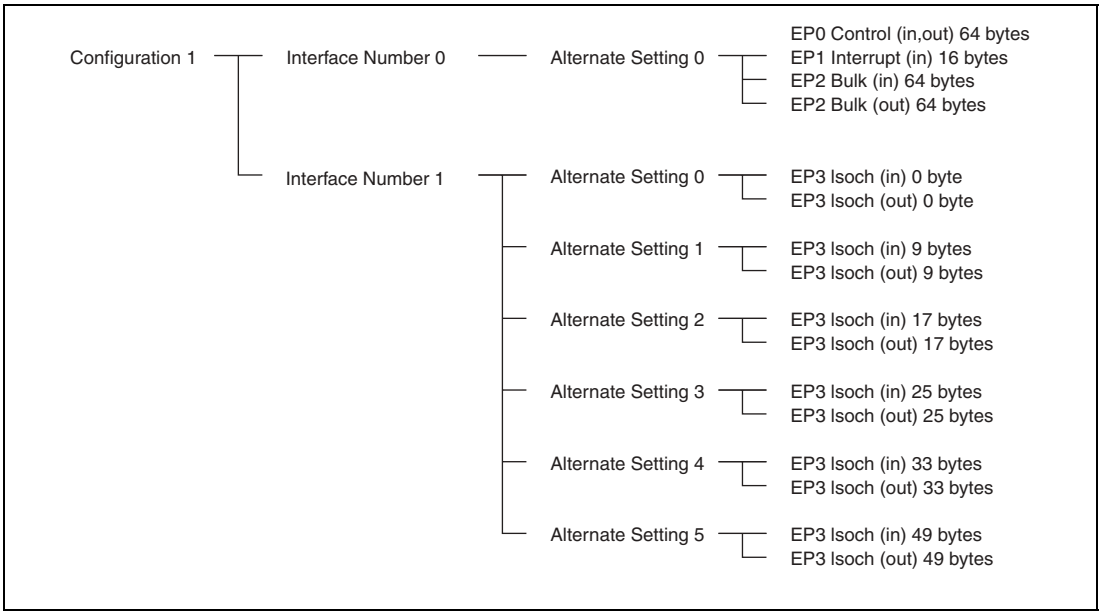


Figure 20.1 shows the block diagram of USBF.

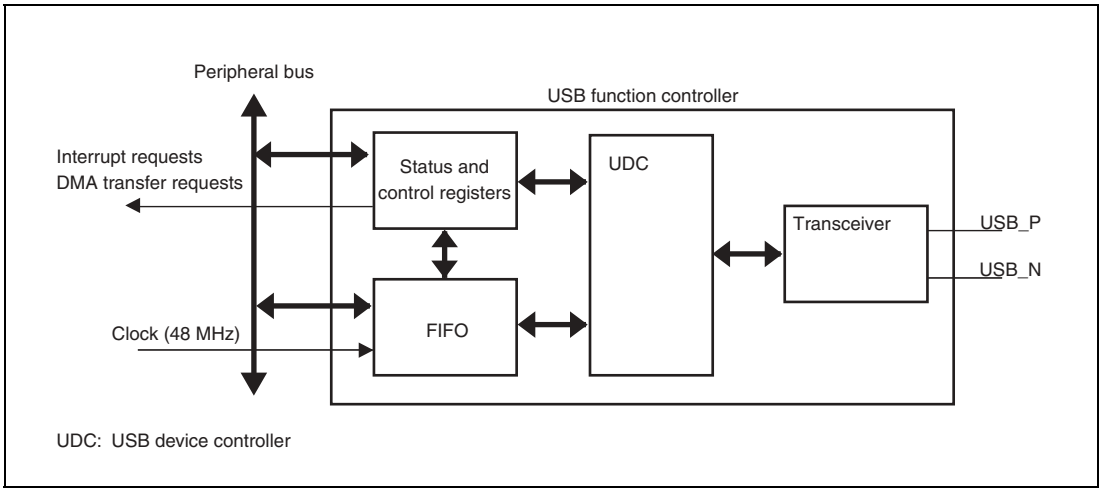


Figure 20.1 Block Diagram of USB Function Controller

20.2 Input/Output Pins

Table 20.1 lists the pin configuration of USBF. For further information on pin settings, see section 18, USB Pin Multiplex Controller (USBPM).

Table 20.1 Pin Configuration

Pin Name	I/O	Function
UCLK	Input	External clock pin Pin for directly inputting an external clock of 48 MHz
$\overline{\text{USB_PWR_EN}}$/ USB_PULLUP	Output	Power enable pin/pull-up control pin Power-supply application enable control for USBH/pull-up control pin for USBF
$\overline{\text{USB_OVR_CRNT}}$/ USB_VBUS	Input	Over current pin/VBUS pin Over current detection for USBH/USB-cable connection monitor pin for USBF
USB_P	I/O	P pin Port transceiver D + pin
USB_N	I/O	N pin Port transceiver D – pin

20.3 Register Descriptions

USB has following registers. For the information on the addresses of these registers and the state of the register in each processing condition, see section 27, List of Registers.

- Interrupt flag register 0 (IFR0)
- Interrupt select register 0 (ISR0)
- Interrupt enable register 0 (IER0)
- EP0i data register (EPDR0i)
- EP0o data register (EPDR0o)
- EP0s data register (EPDR0s)
- EP1 data register (EPDR1)
- EP2i data register (EPDR2i)
- EP2o data register (EPDR2o)
- EP3i data register (EPDR3i)
- EP3o data register (EPDR3o)
- EP4 data register (EPDR4)
- EP5 data register (EPDR5)
- EP6 data register (EPDR6)
- EP0o receive data size register (EPSZ0o)
- EP2o receive data size register (EPSZ2o)
- EP3o receive data size register (EPSZ3o)
- EP6 receive data size register (EPSZ6)
- Trigger register (TRG)
- Data status register (DASTS)
- FIFO clear register (FCLR)
- DMA transfer setting register (DMA)
- Endpoint stall register (EPSTL)
- Configuration value register (CVR)
- Time stamp register (TSR)
- Control register (CTLR)
- Endpoint information register (EPIRn0 to EPIRn5)

20.3.1 Interrupt Flag Register 0 (IFR0)

IFR0 is an interrupt flag register for EP0i, EP0o, EP1, EP2i, EP2o, EP3i, EP3o, and EP4 to EP6, bus reset, setup command reception, VBUS, SUS/RES, SOF, SETC, and SETI. When each flag is set to 1 and an interrupt is enabled in the corresponding bit of the interrupt enable register 0 (IER0), an interrupt occurs from the INT pin specified by the corresponding bit in the interrupt select register (ISR0). Interrupt requests and interrupt request signals can be specified by ISR0. Clearing is performed by writing 0 to the bit to be cleared. Writing 1 is not valid and nothing is changed.

Bit	Bit Name	Initial Value	R/W	Description
31	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
30	BRST	0	R/W	Bus reset [Setting condition] • When a bus reset signal is detected on the USB bus. [Clearing conditions] • At a reset • When 0 is written to by CPU
29	SETUP TS	0	R/W	Setup command receive complete [Setting condition] • When 8-byte data that decodes the command by the function is normally received and an ACK handshake is returned to the host from the function. [Clearing conditions] • At a reset • When 0 is written to by CPU
28	VBUS MN	0	R*[2]	USB Connection Status Status bit to monitor the USB_VBUS pin state. Reflects the state of the USB_VBUS pin.

Bit	Bit Name	Initial Value	R/W	Description
27	VBUSF	0	R/W	USB Disconnection Detection [Setting condition] • When the function is connected to the USB bus or disconnected from it. Note: The USB_VBUS pin of this module is used for detecting connection/disconnection. [Clearing conditions] • At a reset • When 0 is written to by CPU.
26	SURSS	0	R*[1]	Suspend/Resume Status Status bit indicating the state of the bus 1: Suspend state 0: Normal state
25	SURES	0	R/W	Suspend /Resume Detection [Setting condition] • When the bus transits from the normal state to the suspend state or from the suspend state to the normal state. [Clearing conditions] • At a reset • When 0 is written to by CPU
24	CFDN	0	R/W	Endpoint Information Load Complete [Setting condition] • When setting the endpoint information written in the EPIR register is completed in this module. [Clearing conditions] • At a reset • When 0 is written to by CPU Note: This module operates normally as USB function module after the setting of the endpoint information is completed.

Bit	Bit Name	Initial Value	R/W	Description
23	SOF	0	R/W	SOF Packet [Setting condition] • When the SOF packet is detected. [Clearing conditions] • At a reset • When 0 is written to by CPU
22	SETC	0	R/W	Set Configuration Command Detection [Setting condition] • When the Set Configuration command is detected. [Clearing conditions] • At a reset • When 0 is written to by CPU
21	SETI	0	R/W	Set Interface Command Detection [Setting condition] • When the Set Interface command is detected. [Clearing conditions] • At a reset • When 0 is written to by CPU
20	EP6 FULL	0	R*[1]	EP6 (Bulk out) FIFO Full [Setting condition] • The FIFO buffer of EP6 has a dual-buffer configuration, and this bit is set when at least one of the FIFO buffer is full. [Setting conditions] • At a reset • When both FIFO buffers are empty. Note: EP6 FULL is a status bit, and cannot be cleared.

Bit	Bit Name	Initial Value	R/W	Description
19	EP5 EMPTY	1	R*[1]	EP5 (Bulk in) FIFO Empty [Setting conditions] • At a reset • The FIFO buffer of EP5 has a dual-buffer configuration, and this bit is set when at least one of the FIFO buffer is empty. [Clearing condition] • When both of FIFO buffers are not empty. Note: EP5 EMPTY is a status bit, and cannot be cleared.
18	EP5 TR	0	R/W	EP5 (Bulk in) Transfer Request [Setting condition] • When an IN token is received from the host to EP5 and both of FIFO buffers are empty. [Clearing conditions] • At a reset • When 0 is written to by CPU
17	EP4 TR	0	R/W	EP4 (Int) Transfer Request [Setting condition] • When IN token is issued from the host to EP4 and the FIFO buffer is empty. [Clearing conditions] • At a reset • When 0 is written to by CPU
16	EP4 TS	0	R/W	EP4 (Int) Transmit Complete [Setting condition] • When data is normally transferred from the function to the host and an ACK handshake is returned after the data to be transmitted to the host is written in EP4. [Clearing conditions] • At a reset • When 0 is written to by CPU

Bit	Bit Name	Initial Value	R/W	Description
15, 14	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
13	EP3o TF	0	R/W	EP3o (Iso out) Abnormal Reception Flag indicating the FIFO state of EP3o. Indicates the state of the FIFO buffer that was readable after the data reception is completed and the next SOF packet is received. [Setting condition] • When the transfer data from the host is abnormally received (packet error) by EP3o. [Clearing conditions] • At a reset • When 0 is written to by CPU
12	EP3o TS	0	R/W	EP3o (Iso out) Normal Reception Flag indicating the FIFO state of EP3o. Indicates the state of the FIFO buffer that was readable after the data reception is completed and the next SOF packet is received. [Setting condition] • When the transfer data from the host is normally received by EP3o. [Clearing conditions] • At a reset • When 0 is written to by CPU
11	EP3i TR	0	R/W	EP3i (Iso in) Transmit Request [Setting condition] • When the FIFO buffer to be transmitted when an IN token is issued from the host to EP3i is empty. [Clearing conditions] • At a reset • When 0 is written to by CPU

Bit	Bit Name	Initial Value	R/W	Description
10	EP3i TS	0	R/W	EP3i (Iso in) Normal Transmission Flag indicating the FIFO state of EP3i. After the SOF packet is received, the FIFO buffer is switched automatically. The FIFO buffer which has transmitted the data to the host in the previous frame (before SOF reception) can be written to from the CPU. This bit shows the transmitting state in front of this one. [Setting condition] When a transmission was carried out normally in the previous frame. [Clearing conditions] • At a reset • When 0 is written to by CPU
9, 8	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
7	EP2o FULL	0	R*[1]	EP2o (Bulk out) FIFO Full [Setting condition] The FIFO buffer of EP2o has a dual-buffer configuration, and this bit is set when at least one of the FIFO buffer is full. [Clearing conditions] • At a reset • When both FIFO buffers are empty. Note: EP2o FULL is a status bit, and cannot be cleared.
6	EP2i EMPTY	1	R*[1]	EP2i (Bulk in) FIFO Empty [Setting condition] • At a reset • The FIFO buffer of EP2i has a dual-buffer configuration, and this bit is set when at least one of the FIFO buffer is empty. [Clearing condition] When both FIFO buffers are not empty. Note: EP2i EMPTY is a status bit, and cannot be cleared.

Bit	Bit Name	Initial Value	R/W	Description
5	EP2i TR	0	R/W	EP2i (Bulk in) Transfer Request [Setting condition] • When an IN token is issued from the host to EP2i and both FIFO buffers are empty. [Clearing conditions] • At a reset • When 0 is written to by CPU
4	EP1 TR	0	R/W	EP1 (Int) Transfer Request [Setting condition] • When IN token is issued from the host to EP1 and the FIFO buffer is empty. [Clearing conditions] • At a reset • When 0 is written to by CPU
3	EP1 TS	0	R/W	EP1 (Int) Transmit Complete [Setting condition] • When data to be transmitted to the host is written to EP1, then data is normally transferred from the function to the host, and an ACK handshake is returned. [Clearing conditions] • At a reset • When 0 is written to by CPU
2	EP0o TS	0	R/W	EP0o Receive Complete [Setting condition] • When data is normally received from the host to EP0o and an ACK handshake is returned from the function to the host. [Clearing conditions] • At a reset • When 0 is written to by CPU

Bit	Bit Name	Initial Value	R/W	Description
1	EP0i TR	0	R/W	EP0i Transfer Request [Setting condition] • When IN token is issued from the host to EP0i and the FIFO buffer is empty. [Clearing conditions] • At a reset • When 0 is written to by CPU
0	EP0i TS	0	R/W	EP0i Transmit Complete [Setting condition] • When data to be transmitted to the host is written to EP0i, then data is normally transferred from the function to the host, and an ACK handshake is returned. [Clearing conditions] • At a reset • When 0 is written to by CPU

Notes: 1. FIFO full bits of EP2o and EP6 and FIFO empty bits of EP2i and EP5 are status bits indicating the FIFO states, so they cannot be cleared.
2. The USB connection status (bit 28) is a bit indicating the state of the USB_VBUS pin, so it cannot be cleared. Output of an interrupt is not carried out either.

20.3.2 Interrupt Select Register 0 (ISR0)

ISR selects the interrupt request pin to output the IFR0 interrupt. The USB function module has two interrupt request signals (INT0N and INT1N) to output an interrupt request to the interrupt controller (INTC). If the corresponding bit in ISR0 is cleared to 0, an INT0N interrupt request will be issued and an interrupt request is issued from the INT1N pin when set to 1. INT0N and INT1N interrupt request signals correspond to USBFI0 and USBFI1 of the interrupt controller (INTC), respectively.

In the initial value, each of interrupt source of the interrupt flag register is issued from the INT0N interrupt request signal.

Bit	Bit Name	Initial Value	R/W	Description
31	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
30	BRST IS	0	R/W	BRST Interrupt Select 0: Outputs a BRST (bus reset) interrupt request from INT0N 1: Outputs a BRST (bus reset) interrupt request from INT1N
29	SETUP TS IS	0	R/W	SETUP Interrupt Select 0: Outputs a SETUP(setup data receive complete) interrupt request from INT0N 1: Outputs a SETUP(setup data receive complete) interrupt request from INT1N
28	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
27	VBUSF IS	0	R/W	VBUSF Interrupt Select 0: Outputs a VBUSF interrupt request from INT0N 1: Outputs a VBUSF interrupt request from INT1N
26	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
25	SURES IS	0	R/W	SURES Interrupt Select 0: Outputs a SURES interrupt request from INT0N 1: Outputs a SURES interrupt request from INT1N
24	CFDN IS	0	R/W	CFDN Interrupt Select 0: Outputs a CFDN interrupt request from INT0N 1: Outputs a CFDN interrupt request from INT1N
23	SOF IS	0	R/W	SOF Interrupt Select 0: Outputs a SOF interrupt request from INT0N 1: Outputs a SOF interrupt request from INT1N

Bit	Bit Name	Initial Value	R/W	Description
22	SETC IS	0	R/W	SETC Interrupt Select 0: Outputs a SETC interrupt request from INT0N 1: Outputs a SETC interrupt request from INT1N
21	SETI IS	0	R/W	SETI Interrupt Select 0: Outputs a SETI interrupt request from INT0N 1: Outputs a SETI interrupt request from INT1N
20	EP6 FULL IS	0	R/W	EP6 FULL Interrupt Select 0: Outputs an EP6 FULL interrupt request from INT0N 1: Outputs an EP6 FULL interrupt request from INT1N
19	EP5 EMPTY IS	0	R/W	EP5 EMPTY Interrupt Select 0: Outputs an EP5 EMPTY interrupt request from INT0N 1: Outputs an EP5 EMPTY interrupt request from INT1N
18	EP5 TR IS	0	R/W	EP5 TR Interrupt Select 0: Outputs an EP5 TR (transfer request) interrupt request from INT0N 1: Outputs an EP5 TR transfer request) interrupt request from INT1N
17	EP4 TR IS	0	R/W	EP4 TR Interrupt Select 0: Outputs an EP4 TR (transfer request) interrupt request from INT0N 1: Outputs an EP4 TR (transfer request) interrupt request from INT1N
16	EP4 TS IS	0	R/W	EP4 TS Interrupt Select 0: Outputs an EP4 TS (receive complete) interrupt request from INT0N 1: Outputs an EP4 TS (receive complete) interrupt request from INT1N
15, 14	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
13	EP3o TF IS	0	R/W	EP3o TF Interrupt Select 0: Outputs an EP3o TF interrupt request from INT0N 1: Outputs an EP3o TF interrupt request from INT1N

Bit	Bit Name	Initial Value	R/W	Description
12	EP3o TS IS	0	R/W	EP3o TS Interrupt Select 0: Outputs an EP3o TS (receive complete) interrupt request from INT0N 1: Outputs an EP3o TS (receive complete) interrupt request from INT1N
11	EP3i TR IS	0	R/W	EP3i TR Interrupt Select 0: Outputs an EP3i TR (transfer request) interrupt request from INT0N 1: Outputs an EP3i TR (transfer request) interrupt request from INT1N
10	EP3i TS IS	0	R/W	EP3i TS Interrupt Select 0: Outputs an EP3i TS (receive complete) interrupt request from INT0N 1: Outputs an EP3i TS (receive complete) interrupt request from INT1N
9, 8	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
7	EP2o FULL IS	0	R/W	EP2o FULL Interrupt Select 0: Outputs an EP2o FULL interrupt request from INT0N 1: Outputs an EP2o FULL interrupt request from INT1N
6	EP2i EMPTY IS	0	R/W	EP2i EMPTY Interrupt Select 0: Outputs an EP2i EMPTY interrupt request from INT0N 1: Outputs an EP2i EMPTY interrupt request from INT1N
5	EP2i TR IS	0	R/W	EP2i TR Interrupt Select 0: Outputs an EP2i TR (transfer request) interrupt request from INT0N 1: Outputs an EP2i TR (transfer request) interrupt request from INT1N
4	EP1 TR IS	0	R/W	EP1 TR Interrupt Select 0: Outputs an EP1 TR (transfer request) interrupt request from INT0N 1: Outputs an EP1 TR (transfer request) interrupt request from INT1N

Bit	Bit Name	Initial Value	R/W	Description
3	EP1 TS IS	0	R/W	EP1 TS Interrupt Select 0: Outputs an EP1 TS (receive complete) interrupt request from INT0N 1: Outputs an EP1 TS (receive complete) interrupt request from INT1N
2	EP0o TS IS	0	R/W	EP0o TS Interrupt Select 0: Outputs an EP0oTS (receive complete) interrupt request from INT0N 1: Outputs an EP0o TS (receive complete) interrupt request from INT1N
1	EP0i TR IS	0	R/W	EP0i TR Interrupt Select 0: Outputs an EP0i TR (transfer request) interrupt request from INT0N 1: Outputs an EP0i TR (transfer request) interrupt request from INT1N
0	EP0i TS IS	0	R/W	EP0i TS Interrupt Select 0: Outputs an EP0i TS (receive complete) interrupt request from INT0N 1: Outputs an EP0i TS (receive complete) interrupt request from INT1N

20.3.3 Interrupt Enable Register 0 (IER0)

IER0 enables the interrupt requests of IFR0. When an interrupt flag is set to 1 while the corresponding bit of each interrupt enable bit of IFR0 is set to 1, the INT0N or INT1N pin set in the interrupt select register (ISR0) is asserted low and an interrupt request is issued.

Bit	Bit Name	Initial Value	R/W	Description
31	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
30	BRST IE	0	R/W	BRST Interrupt Enable 0: Disables a BRST (bus reset) interrupt request 1: Enables a BRST (bus reset) interrupt request
29	SETUP TS IE	0	R/W	SETUP TS Interrupt Enable 0: Disables a SETUP TS (setup data receive complete) interrupt request 1: Enables a SETUP TS (setup data receive complete) interrupt request
28	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
27	VBUSF IE	0	R/W	VBUSF Interrupt Enable 0: Disables a VBUSF interrupt request 1: Enables a VBUSF interrupt request
26	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
25	SURES IE	0	R/W	SURES Interrupt Enable 0: Disables a SURES interrupt request 1: Enables a SURES interrupt request
24	CFDN IE	0	R/W	CFDN Interrupt Enable 0: Disables a CFDN interrupt request 1: Enables a CFDN interrupt request

Bit	Bit Name	Initial Value	R/W	Description
23	SOF IE	0	R/W	SOF Interrupt Enable 0: Disables a SOF interrupt request 1: Enables a SOF interrupt request
22	SETC IE	0	R/W	SETC Interrupt Enable 0: Disables a SETC interrupt request 1: Enables a SETC interrupt request
21	SETI IE	0	R/W	SETI Interrupt Enable 0: Disables a SETI interrupt request 1: Enables a SETI interrupt request
20	EP6 FULL IE	0	R/W	EP6 FULL Interrupt Enable 0: Disables an EP6 FULL interrupt request 1: Enables an EP6 FULL interrupt request
19	EP5 EMPTY IE	0	R/W	EP5 EMPTY Interrupt Enable 0: Disables an EP5 EMPTY interrupt request 1: Enables an EP5 EMPTY interrupt request
18	EP5 TR IE	0	R/W	EP5 TR Interrupt Enable 0: Disables an EP5 TR (transfer request) interrupt request 1: Enables an EP5 TR (transfer request) interrupt request
17	EP4 TR IE	0	R/W	EP4 TR Interrupt Enable 0: Disables an EP4 TR (transfer request) interrupt request 1: Enables an EP4 TR (transfer request) interrupt request
16	EP4 TS IE	0	R/W	EP4 TS Interrupt Enable 0: Disables an EP4 TS (receive complete) interrupt request 1: Enables an EP4 TS (receive complete) interrupt request
15, 14	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

Bit	Bit Name	Initial Value	R/W	Description
13	EP3o TF IE	0	R	EP3o TF Interrupt Enable 0: Disables an EP3o TF interrupt request 1: Enables an EP3o TF interrupt request
12	EP3o TS IE	0	R	EP3o TS Interrupt Enable 0: Disables an EP3o TS (receive complete) interrupt request 1: Enables an EP3o TS (receive complete) interrupt request
11	EP3i TR IE	0	R/W	EP3i TR Interrupt Enable 0: Disables an EP3i TR (transfer request) interrupt request 1: Enables an EP3i TR (transfer request) interrupt request
10	EP3i TS IE	0	R/W	EP3i TS Interrupt Enable 0: Disables an EP3i TS (receive complete) interrupt request 1: Enables an EP3i TS (receive complete) interrupt request
9, 8	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
7	EP2o FULL IE	0	R/W	EP2o FULL Interrupt Enable 0: Disables an EP2o FULL interrupt request 1: Enables an EP2o FULL interrupt request
6	EP2i EMPTY IE	0	R/W	EP2i EMPTY Interrupt Enable 0: Disables an EP2i EMPTY interrupt request 1: Enables an EP2i EMPTY interrupt request
5	EP2i TR IE	0	R/W	EP2i TR Interrupt Enable 0: Disables an EP2i TR (transfer request) interrupt request 1: Enables an EP2i TR (transfer request) interrupt request

Bit	Bit Name	Initial Value	R/W	Description
4	EP1 TR IE	0	R/W	EP1 TR Interrupt Enable 0: Disables an EP1 TR (transfer request) interrupt request 1: Enables an EP1 TR (transfer request) interrupt request
3	EP1 TS IE	0	R/W	EP1 TS Interrupt Enable 0: Disables an EP1 TS (receive complete) interrupt request 1: Enables an EP1 TS (receive complete) interrupt request
2	EP0o TS IE	0	R/W	EP0o TS Interrupt Enable 0: Disables an EP0o TS (receive complete) interrupt request 1: Enables an EP0o TS (receive complete) interrupt request
1	EP0i TR IE	0	R/W	EP0i TR Interrupt Enable 0: Disables an EP0i TR (transfer request) interrupt request 1: Enables an EP0i TR (transfer request) interrupt request
0	EP0i TS IE	0	R/W	EP0i TS Interrupt Enable 0: Disables an EP0i TS (receive complete) interrupt request 1: Enables an EP0i TS (receive complete) interrupt request

20.3.4 EP0i Data Register (EPDR0i)

EPDR0i is a 64-byte 8-bit transmit FIFO buffer for endpoint 0. EPDR0i holds one packet of transmit data for control-in. Transmit data is fixed by writing one packet of data and setting EP0iPKTE in TRG. When an ACK handshake is returned from the host after the data has been transmitted, EP0iTS in IFR0 is set. This FIFO buffer can be initialized by means of EP0iCLR in FCLR.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Data register for control-in transfer

20.3.5 EP0o Data Register (EPDR0o)

EPDR0o is a 64-byte 8-bit receive FIFO buffer for endpoint 0. EPDR0o holds endpoint 0 receive data other than setup commands. When data is received normally, EP0oTS in IFR0is set, and the number of receive bytes is indicated in the EP0o receive data size register. After the data has been read, setting EP0oRDFN in TRG enables the next packet to be received. This FIFO buffer can be initialized by means of EP0oCLR in the FIFO clear register (FCLR).

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	R	Data register for control-out transfer

20.3.6 EP0s Data Register (EPDR0s)

EPDR0s is a data register specifically for endpoint 0 setup command. EPDR0s holds 8-byte command data sent in the setup stage. However, only the command to be processed by a microprocessor (firmware) is received. The command data to be processed automatically by this module is not stored.

When the reception of data in the setup stage starts during read, the previous data is overwritten unconditionally. When the reception of the next command occurs during command read, the command reception starts and data read during command read is invalid.

Note: EPDR0s must be read in 8-byte units. If reading is suspended while it is in progress, data received in the next setup cannot be read successfully.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	R	Data register for storing the setup command at the control-out transfer

20.3.7 EP1 Data Register (EPDR1)

EPDR1 is a 16-byte 8-bit transmit FIFO buffer for endpoint 1. EPDR1 holds one packet of transmit data for the interrupt transfer of endpoint 1. Transmit data is fixed by writing one packet of data and setting EP1PKTE in TRG. When an ACK handshake is returned from the host after the data has been transmitted, EP1TS in IFR0 is set. This FIFO buffer can be initialized by means of EP1CLR in FCLR.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Data register for endpoint 1 transfer

20.3.8 EP2i Data Register (EPDR2i)

EPDR2i is a 128-byte 8-bit transmit FIFO buffer for endpoint 2i. EPDR2i has a dual-buffer configuration, and has a capacity of twice the maximum packet size. When transmit data is written to this FIFO buffer and EP2iPKTE in TRG is set, one packet of transmit data is fixed, and the dual-FIFO buffer is switched over. Transmit data for this FIFO buffer can be transferred by DMA. This FIFO buffer can be initialized by means of EP2iCLR in FCLR.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Data register for endpoint 2i transfer

20.3.9 EP2o Data Register (EPDR2o)

EPDR2o is a 128-byte 8-bit receive FIFO buffer for endpoint 2o. EPDR2o has a dual-buffer configuration, and has a capacity of twice the maximum packet size. The number of receive byte is displayed in the EPSZ2o register. The buffer on read side can be received again by writing EP2oRDFN in TRG to 1 after data is read. The receive data of this FIFO buffer can be transferred by DMA. This FIFO buffer can be initialized by means of EP2oCLR in FCLR.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	R	Data register for endpoint 2o transfer

20.3.10 EP3i Data Register (EPDR3i)

EPDR3i is a 128-byte 8-bit transmit FIFO buffer for endpoint 3i. EPDR3i has a dual-buffer configuration, and has a capacity of twice the maximum packet size. When transmit data is written to this FIFO buffer and an SOF packet is received, one packet of transmit data is fixed, and the dual-FIFO buffer is switched over. This FIFO buffer can be initialized by means of EP3iCLR in FCLR.

Note: When an SOF packet cannot be received, an SOF interrupt is generated by an SOF marker function. However, data in a frame which could not receive an SOF is invalid.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Data register for endpoint 3i transfer

20.3.11 EP3o Data Register (EPDR3o)

EPDR3o is a 120-byte 8-bit receive FIFO buffer for endpoint 3o. EPDR3o has a dual-buffer configuration, and has a capacity of twice the maximum packet size. The number of receive byte is displayed in the EPSZ3o register. The receive data is fixed when an SOF packet is received. Accordingly, all receive data must be read until the next SOF packet is received. When the next SOF packet is received, the FIFO side is automatically switched over, and the previous data will not be possible to be read. This FIFO buffer can be initialized by means of EP3o CLR in FCLR.

Note: When an SOF packet cannot be received, an SOF interrupt is generated by an SOF marker function. However, data in a frame which could not receive an SOF is invalid.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	R	Data register for endpoint 3o transfer

20.3.12 EP4 Data Register (EPDR4)

EPDR4 is a 16-byte 8-bit transmit FIFO buffer for endpoint 4. EPDR4 holds one packet of transmit data for the interrupt transfer of endpoint 4. Transmit data is fixed by writing one packet of data and setting EP4PKTE in TRG. When an ACK handshake is returned from the host after the data has been transmitted, EP4TS in IFR0 is set. This FIFO buffer can be initialized by means of EP4CLR in FCLR.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Data register for endpoint 4 transfer

20.3.13 EP5 Data Register (EPDR5)

EPDR5 is a 64-byte 8-bit transmit FIFO buffer for endpoint 5. EPDR5 has a dual-buffer configuration, and has a capacity of twice the maximum packet size. When transmit data is written to this FIFO buffer and setting EP5PKTE in the trigger register, one packet of transmit data is fixed, and the dual-FIFO buffer is switched over. Data can be transferred to this FIFO buffer by DMA. This FIFO buffer can be initialized by means of EP5CLR in the FCLR register.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Data register for endpoint 5 transfer

20.3.14 EP6 Data Register (EPDR6)

EPDR6 is a 64-byte 8-bit receive FIFO buffer for endpoint 6. EPDR6 has a dual-buffer configuration, and has a capacity of twice the maximum packet size. The number of receive byte is displayed in the EPSZ6 receive data size register. The buffer on read side can be received again by writing EP6RDFN in TRG to 1 after data is read. The receive data of this FIFO buffer can be transferred by DMA. This FIFO buffer can be initialized by means of EP6CLR in FCLR.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	D7 to D0	Undefined	R	Data register for endpoint 6 transfer

20.3.15 EP0o Receive Data Size Register (EPSZ0o)

EPSZ0o is a receive data size resister for endpoint 0o. EPSZ0o indicates the number of bytes received from the host.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	—	All 0	R	Number of receive data for endpoint 0o

20.3.16 EP2o Receive Data Size Register (EPSZ2o)

EPSZ2o is a receive data size resister for endpoint 2o. EPSZ2o indicates the number of bytes received from the host. FIFO of endpoint 2o has a dual-buffer configuration. The size of the received data indicated by this register is the size of the currently selected side (can be read by CPU).

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	—	All 0	R	Number of received bytes for endpoint 2o

20.3.17 EP3o Receive Data Size Register (EPSZ3o)

EPSZ3o is a receive data size resister for endpoint 3. EPSZ3o indicates the number of bytes received from the host. FIFO of endpoint 3o has a dual-buffer configuration. The size of the received data indicated by this register is the size of the currently selected side (can be read by CPU).

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	—	All 0	R	Number of received bytes for endpoint 3o

20.3.18 EP6 Receive Data Size Register (EPSZ6)

EPSZ6 is a receive data size resister for endpoint 6. EPSZ6 indicates the number of bytes received from the host. FIFO of endpoint 6 has a dual-buffer configuration. The size of the received data indicated by this register is the size of the currently selected side (can be read by CPU).

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	—	All 0	R	Number of received bytes for endpoint 6

20.3.19 Trigger Register (TRG)

TRG generates one-shot triggers FIFO for each endpoint of EP0s, EP0i, EP0o, EP1, EP2i, EP2o, EP4, EP5, and EP6. The packet enable trigger for the IN FIFO register and read complete trigger for the OUT FIFO register are triggers to be given. If a bit corresponding to each endpoint is set to 1, a trigger is given. Each bit in this register is automatically cleared to 0 after it is set to 1.

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	—	Reserved The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
14	EP6 RDFN	0	W	EP6 Read Complete
13	EP5 PKTE	0	W	EP5 Packet Enable
12	EP4 PKTE	0	W	EP4 Packet Enable
11 to 7	—	All 0	—	Reserved The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
6	EP2o RDFN	0	W	EP2o Read Complete
5	EP2i PKTE	0	W	EP2i Packet Enable
4	EP1 PKTE	0	W	EP1 Packet Enable
3	—	0	—	Reserved The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
2	EP0s RDFN	1	W	EP0s Read Complete
1	EP0o RDFN	0	W	EP0o Read Complete
0	EP0i PKTE	0	W	EP0i Packet Enable

20.3.20 Data Status Register (DASTS)

DASTS indicates whether the transmit FIFO data register contains valid data. DASTS is set to 1 when data written to the transmit FIFO is enabled by writing PKTE in TRG to 1, and cleared to 0 when all data has been transmitted to the host. In case of a dual-configuration FIFO for endpoints 2i and 5, this bit is cleared to 0 when both sides are empty.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0.
6	EP5DE	0	R	EP5 Data Enable
5	EP4DE	0	R	EP4 Data Enable
4	—	0	R	Reserved This bit is always read as 0.
3	EP2iDE	0	R	EP2i Data Enable
2	EP1DE	0	R	EP1 Data Enable
1	—	0	R	Reserved This bit is always read as 0.
0	EP0iDE	0	R	EP0i Data Enable

20.3.21 FIFO Clear Register (FCLR)

FCLR is a one shot register to clear the FIFO buffers for each endpoint. Writing 1 to a bit clears the data in the corresponding FIFO buffer. Each bit in this register is automatically cleared to 0 after it is set to 1.

In case of the transmit FIFO, by writing data in the FIFO buffer, the data by which PKTE in TRG is not written to 1 and the data enabled by writing 1 can be cleared. In case of the receive FIFO, the unfixed data during reception and the data of which reception has not been completed can be cleared.

Both sides of the dual-configuration FIFO buffers for EP2i, EP2o, EP3i, EP3o EP5, and EP6 can be cleared.

The corresponding interrupt flag is not cleared by this clear instruction. Do not clear a FIFO buffer during transmission.

Bit	Bit Name	Initial Value	R/W	Description
15	—	—	—	Reserved The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
14	EP6 CLR	—	W	EP6 Clear
13	EP5 CLR	—	W	EP5 Clear
12	EP4 CLR	—	W	EP4 Clear
11	—	—	—	Reserved The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
10	EP3o CLR	—	W	EP3o Clear
9	EP3i CLR	—	W	EP3i Clear
8, 7	—	—	—	Reserved The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
6	EP2o CLR	—	W	EP2o Clear
5	EP2i CLR	—	W	EP2i Clear
4	EP1 CLR	—	W	EP1 Clear
3, 2	—	—	—	Reserved The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
1	EP0o CLR	—	W	EP0o Clear
0	EP0i CLR	—	W	EP0i Clear

20.3.22 DMA Transfer Setting Register (DMA)

DMA is set when the dual address transfer of the DMAC is used to the data register for endpoints EP2i, EP2o, EP5, and EP6 to which transfer is possible by DMA.

Bit	Bit Name	Initial Value	R/W	Description
7	EP6 DMAE	0	R/W	EP6DMA Enable Enables DMA transfer for EP6.
6	EP6 DMAS	0	R/W	EP6DMA Request Select Selects DMA transfer request output for EP6.
5	EP5 DMAE	0	R/W	EP5DMA Enable Enables DMA transfer for EP5.

Bit	Bit Name	Initial Value	R/W	Description
4	EP5 DMAS	0	R/W	EP5DMA Request Select Selects DMA transfer request output for EP5.
3	EP2o DMAE	0	R/W	EP2oDMA Enable Enables DMA transfer for EP2o.
2	EP2o DMAS	0	R/W	EP2oDMA Request Select Selects DMA transfer request output for EP2o.
1	EP2i DMAE	0	R/W	EP2iDMA Enable Enables DMA transfer for EP2i.
0	EP2i DMAS	0	R/W	EP2iDMA Request Select Selects DMA transfer request output for EP2i.

20.3.23 Endpoint Stall Register (EPSTL)

EPSTL stalls each endpoint. The endpoint in which the stall bit is set to 1 returns a stall handshake to the host from the next transfer when 1 is written to. The stall bit for endpoint 0 is cleared automatically on reception of 8 byte command data for which decoding is performed by the function and the EP0 STL bit is cleared. When the SETUPTS bit in IFR0 is set to 1, a write of the EP0 STL bit to 1 is ignored. For detailed operation, see section 20.6, Stall Operations.

Bit	Bit Name	Initial Value	R/W	Description
15 to 11	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
10	EP6 STL	0	R/W	EP6 Stall Sets EP6 stall
9	EP5 STL	0	R/W	EP5 Stall Sets EP5 stall
8	EP4 STL	0	R/W	EP4 Stall Sets EP4 stall
7	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
6	EP3o STL	0	R/W	EP3o Stall Sets EP3o stall

Bit	Bit Name	Initial Value	R/W	Description
5	EP3i STL	0	R/W	EP3i Stall Sets EP3i stall
4	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
3	EP2o STL	0	R/W	EP2o Stall Sets EP2o stall
2	EP2i STL	0	R/W	EP2i Stall Sets EP2i stall
1	EP1 STL	0	R/W	EP1 Stall Sets EP1 stall
0	EP0 STL	0	R/W	EP0 Stall Sets EP0 stall

20.3.24 Configuration Value Register (CVR)

CVR is a register to store the Configuration/Interface/Alternate value to be set when the Set Configuration/Set Interface command is normally received from the host.

Bit	Bit Name	Initial Value	R/W	Description
7	CNFV	0	R	Configuration Value The configuration setting value is stored when the Set Configuration command has been received. CNFV is updated when SETC in the interrupt flag register, IFR0, is set to 1.
6	INTV3	0	R	Interface Value
5	INTV2	0	R	The interface setting value is stored when the Set Interface command has been received. INTV is updated when SETI in the interrupt flag register, IFR0, is set to 1.
4	INTV1	0	R	
3	INTV0	0	R	
2	ALTV2	0	R	Alternate Value
1	ALTV1	0	R	The alternate setting value is stored when the Set interface command has been received. ALTV is updated when SETI in the interrupt flag register, IFR0, is set to 1.
0	ALTV0	0	R	

20.3.25 Time Stamp Register (TSR)

TSR is a register to store the current time stamp value. The time stamp is updated when the SOF bit in IFR0 is set to1. The value of the time stamp when the SOF marker function is enabled and the SOF packet is broken remains as previous one.

Bit	Bit Name	Initial Value	R/W	Description
15 to 11	—	All 0	R	Reserved. These bits are always read as 0.
10	D10	0	R	Time stamp data
9	D9	0	R	
8	D8	0	R	
7	D7	0	R	
6	D6	0	R	
5	D5	0	R	
4	D4	0	R	
3	D3	0	R	
2	D2	0	R	
1	D1	0	R	
0	D0	0	R	

20.3.26 Control Register (CLTR)

CLTR is a register to make various settings of the USBF.

When the isochronous transfer is used, the SOF marker function must be enabled.

Bit	Bit name	Initial value	R/W	Description
7, 6	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
5	PULLUPE	0	R/W	Pull-up Enable Controls the output value of the USB_PULLUP pin.

Bit	Bit name	Initial value	R/W	Description
4	RWUPS	0	R	Remote Wakeup Status Status bit to indicate that the remote wakeup from the host is enabled/disabled. Indicates 0 when the remote wakeup is disabled with Device Remote Wakeup by the Set Feature/Clear Feature request and indicates 1 when it is enabled.
3	RSME	0	R/W	Resume Enable Bit to clear the suspend state (performs the remote wakeup) When this bit is written to 1, a resume register is set. When this bit will be used, be sure to hold to 1 for one clock or more at 12 MHz in minimum and then clear to 0 again.
2	PWMD	0	R/W	Power Mode Select 0: USBF is operated in self-powered mode 1: USBF is operated in bus-powered mode
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 the stall setting bit (EPSTLR/EPxSTL) of the returned endpoint is automatically cleared. Control in a unit of endpoint is disabled as this bit is common for all endpoints. When this bit is set to 0, be sure to clear the stall setting bit of each endpoint by using software. Enable this bit before setting EPSTL to 1.
0	SOFME	0	R/W	SOF Marker Function Enable Controls SOF marker function. When this bit is set to 1, the SOF interrupt flag is set to 1 every 1 msec even if an SOF packet is broken. Before setting this bit to 1, check if the SOF interrupt flag of IFR0 register is detected correctly. If a suspension is detected, this bit must be cleared to 0. Before setting this bit to 1 again, check the SOF detection.

20.3.27 Endpoint Information Register (EPIRn0 to EPIRn5)

EPIR is a register to set the configuration information for each endpoint. 6 bytes of the information are required for one endpoint. Write the data in order from endpoint 0. Do not write more than 6 (bytes) × 19 (endpoints) = 114 bytes. Write this information once at power-on reset. Do not write it again afterwards.

Write data of one endpoint is described below. EPIR writes data in the same address in order. Therefore though there is only one EPIR register, write data for registration number n (n is from 0 to 18) is listed as EPIRn0 to EPIRn5 (EPIR [registration number] [write order]) for the purpose of explaining. Write data in order from EPIR00.

- EPIRn0

Bit	Bit Name	Initial value	R/W	Description
7 to 4	D7 to D4	Undefined	W	Endpoint Number Settable range: B'0000 to B′0110 Note: B'0111 to B'1111 = Setting prohibited
3 to 0	D3 to D0	Undefined	W	Configuration number to which endpoint belongs Settable range: B'0000 to B′0001 Note: B'0010 to B'1111 = Setting prohibited

- EPIRn1

Bit	Bit Name	Initial value	R/W	Description
7 to 4	D7 to D4	Undefined	W	Interface number to which endpoint belongs Settable range: B'0000 to B'0010 Note: B'0011 to B'1111 = Setting prohibited
3 to 0	D3 to D0	Undefined	W	Alternate number to which endpoint belongs Settable range: Differs based on the interface number (D7 to D4) D7 to D4 = B'0000: D3 to D0 = B'0000 D7 to D4 = B'0001: D3 to D0 = B'0000 to B'0101 D7 to D4 = B'0010: D3 to D0 = B'0000 Note: Other than above = Setting prohibited

- EPIRn2

Bit	Bit Name	Initial value	R/W	Description
7, 6	D7, D6	Undefined	W	Reserved The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
5, 4	D5, D4	Undefined	W	Transfer Method of Endpoint [Settable range] B'00: Control B'01: Isochronous B'10: Bulk B'11: Interrupt
3	D3	Undefined	W	Transfer Direction of Endpoint [Settable range] B'0: Out B'1: In
2 to 0	D2 to D0	Undefined	W	Reserved The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.

- EPIRn3

Bit	Bit Name	Initial value	R/W	Description
7 to 1	D7 to D1	Undefined	W	Maximum Packet Size of Endpoint Settable range: B'0000000 to B'1000000 Note: B'1000001 to B'1111111= Setting prohibited
0	D0	Undefined	W	Reserved The write value should always be 0. If 1 is written in this bit, correct operation cannot be guaranteed.

- EPIRn4

Bit	Bit Name	Initial value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Reserved The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.

- EPIRn5

Bit	Bit Name	Initial value	R/W	Description
7 to 0	D7 to D0	Undefined	W	Endpoint FIFO Number Settable range: H'00 to H'03, H'05 to H'09 Note: H'04, H'0A to H'FF = Setting prohibited

An endpoint number specified by the EPIRn0 register is an endpoint number used by the USB host. The endpoint FIFO number specified by the EPIRn5 register corresponds to the FIFO number which is attached to each endpoint physically supported by this USB function module. For details on correspondence between the endpoint functions and FIFO numbers in this USB function module, refer to section 20.1, Features. Note that the setting values for EPIRn0 to EPIRn5 are limited as described below.

1. Since each endpoint FIFO is optimized by a dedicated hardware corresponding to each transfer method, transfer direction, and maximum packet size, set the endpoint FIFO with a transfer method, transfer direction, and maximum packet size shown in the table 20.2. Example: Endpoint FIFO number 1 cannot be set as other than interrupt transfer and maximum packet size (16 bytes). Although endpoint FIFO number 5 cannot be set as other than isochronous transfer and IN, maximum packet size can be set in the range of 0 to 64 bytes.
2. Endpoint number 0 and endpoint FIFO number 0 must correspond.
3. The maximum packet size of endpoint FIFO number 0 can be set to 64 bytes only.
4. The setting value of endpoint FIFO number 0 can be set to the maximum packet size only and the rest data is all 0.
5. The maximum packet size of endpoint FIFO numbers 1 and 7 can be set to 16 only.
6. The maximum packet size of endpoint FIFO numbers 2 and 3 can be set to 64 only.
7. The maximum packet size of endpoint FIFO numbers 8 and 9 can be set to 32only.
8. The maximum packet size of endpoint FIFO number 5 can be set in the range of 0 to 64.
9. The maximum packet size of endpoint FIFO number 6 can be set in the range of 0 to 60.
10. When the isochronous transfer is set, Alternate can be used in the range of 0 to 5 for the same endpoint. Be sure to allocate the Alternate to the same endpoint FIFO number.
11. Endpoint information can be set up to 19 in maximum.
12. Endpoint information of 19 pieces must be written.
13. All information of endpoints which are not used must be written as 0.

A list of restrictions of settable transfer method, transfer direction, and maximum packet size is described in table 20.2.

Table 20.2 Restrictions of Settable Values

Endpoint FIFO No.	Maximum Packet Size	Transfer Method	Transfer Direction
0	64 bytes	Control	—
1	16 bytes	Interrupt	IN
2	64 bytes	Bulk	IN
3	64 bytes	Bulk	OUT
5	0 to 64 bytes	Isochronous	IN
6	0 to 60 bytes	Isochronous	OUT
7	16 bytes	Interrupt	IN
8	32 bytes	Bulk	IN
9	32 bytes	Bulk	OUT

Examples of settings are shown below.

Example of Setting 1: This is an example which is recommended by HCI USB TRANSPORT LAYER. Data for definition numbers 16 to 18 must be all 0s.

Table 20.3 Example of Endpoint Configuration (1)

EP No.	Conf.	Int.	Alt.	Transfer Method	Transfer Direction	Maximum Packet Size	EP FIFO No.
0	—	—	—	Control	IN/OUT	64 bytes	0
1	1	0	0	Interrupt	IN	16 bytes	1
2	1	0	0	Bulk	IN	64 bytes	2
2	1	0	0	Bulk	OUT	64 bytes	3
3	1	1	0	Isochronous	IN	0 byte	5
3	1	1	0	Isochronous	OUT	0 byte	6
3	1	1	1	Isochronous	IN	9 bytes	5
3	1	1	1	Isochronous	OUT	9 bytes	6
3	1	1	2	Isochronous	IN	17 bytes	5
3	1	1	2	Isochronous	OUT	17 bytes	6
3	1	1	3	Isochronous	IN	25 bytes	5
3	1	1	3	Isochronous	OUT	25 bytes	6
3	1	1	4	Isochronous	IN	33 byte	5
3	1	1	4	Isochronous	OUT	33 bytes	6

EP No.	Conf.	Int.	Alt.	Transfer Method	Transfer Direction	Maximum Packet Size	EP FIFO No.
3	1	1	5	Isochronous	IN	49 byte	5
3	1	1	5	Isochronous	OUT	49 bytes	6
—	—	—	—	—	—	—	—
—	—	—	—	—	—	—	—
—	—	—	—	—	—	—	—

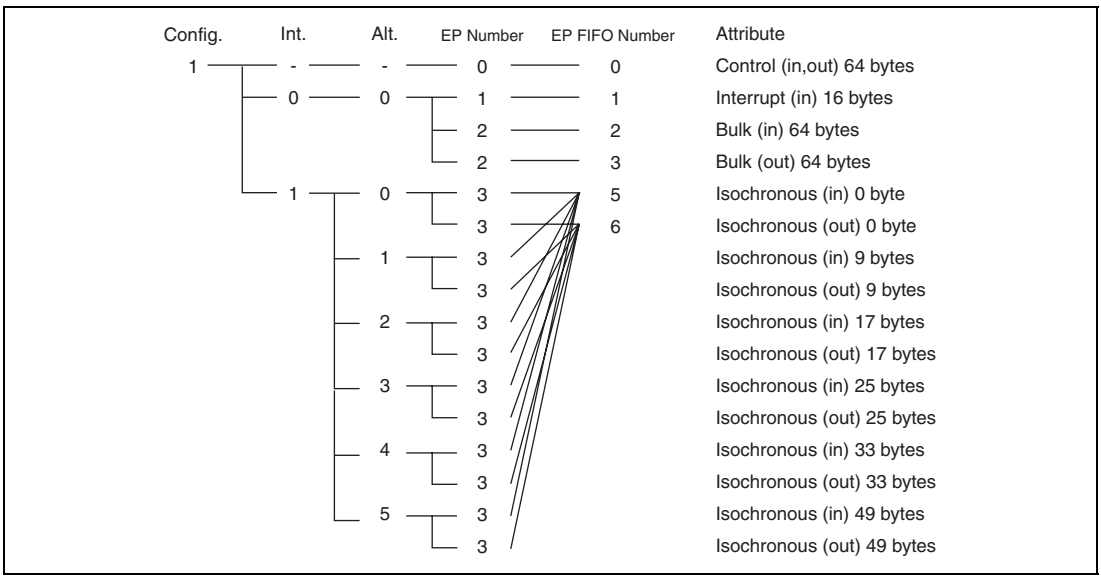


Figure 20.2 Example of Endpoint Configuration (1)

Table 20.4 Example of Setting of Endpoint Configuration Information (1)

N	EPIR[N]0	EPIR[N]1	EPIR[N]2	EPIR[N]3	EPIR[N]4	EPIR[N]5
0	00	00	00	80	00	00
1	11	00	38	20	00	01
2	21	00	28	80	00	02
3	21	00	20	80	00	03
4	31	10	18	00	00	05
5	31	10	10	00	00	06
6	31	11	18	12	00	05
7	31	11	10	12	00	06
8	31	12	18	22	00	05
9	31	12	10	22	00	06
10	31	13	18	32	00	05
11	31	13	10	32	00	06
12	31	14	18	42	00	05
13	31	14	10	42	00	06
14	31	15	18	62	00	05
15	31	15	10	62	00	06
16	00	00	00	00	00	00
17	00	00	00	00	00	00
18	00	00	00	00	00	00

Example of Setting 2: This is an example when there is endpoint information which is not used. All data of registration numbers 13 to 18 must be written as 0.

Table 20.5 Example of Endpoint Configuration (2)

EP No.	Conf.	Int.	Alt.	Transfer Method	Transfer Direction	Maximum Packet Size	EP FIFO No.
0	—	—	—	Control	IN/OUT	64 bytes	0
1	1	0	0	Interrupt	IN	16 bytes	1
2	1	0	0	Bulk	IN	64 bytes	2
2	1	0	0	Bulk	OUT	64 bytes	3
3	1	1	0	Isochronous	IN	0 byte	5
3	1	1	0	Isochronous	OUT	0 byte	6
3	1	1	1	Isochronous	IN	32 bytes	5
3	1	1	1	Isochronous	OUT	32 bytes	6
3	1	1	2	Isochronous	IN	64 bytes	5
3	1	1	2	Isochronous	OUT	60 bytes	6
4	1	2	0	Interrupt	IN	16 bytes	7
5	1	2	0	Bulk	IN	32 bytes	8
6	1	2	0	Bulk	OUT	32 bytes	9
—	—	—	—	—	—	—	—
—	—	—	—	—	—	—	—
—	—	—	—	—	—	—	—
—	—	—	—	—	—	—	—
—	—	—	—	—	—	—	—
—	—	—	—	—	—	—	—

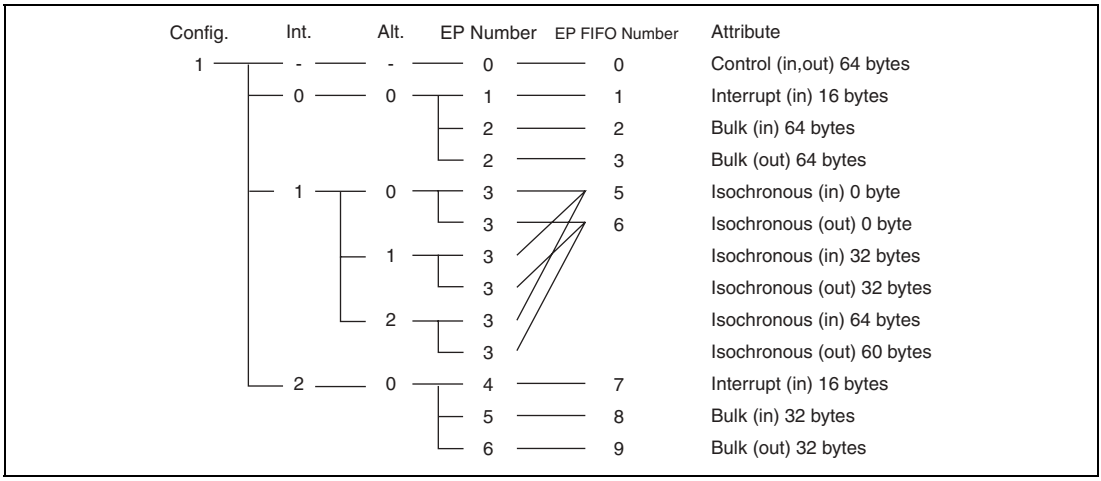


Figure 20.3 Example of Endpoint Configuration (2)

Table 20.6 Example of Setting of Endpoint Configuration Information (2)

n	EPIR[n]0	EPIR[n]1	EPIR[n]2	EPIR[n]3	EPIR[n]4	EPIR[n]5
0	00	00	00	80	00	00
1	11	00	38	20	00	01
2	21	00	28	80	00	02
3	21	00	20	80	00	03
4	31	10	18	00	00	05
5	31	10	10	00	00	06
6	31	11	18	40	00	05
7	31	11	10	40	00	06
8	31	12	18	80	00	05
9	31	12	10	78	00	06
10	41	20	38	20	00	07
11	51	20	28	40	00	08
12	61	20	20	40	00	09
13	00	00	00	00	00	00
14	00	00	00	00	00	00
15	00	00	00	00	00	00
16	00	00	00	00	00	00
17	00	00	00	00	00	00
18	00	00	00	00	00	00

20.4 Operation

20.4.1 Cable Connection

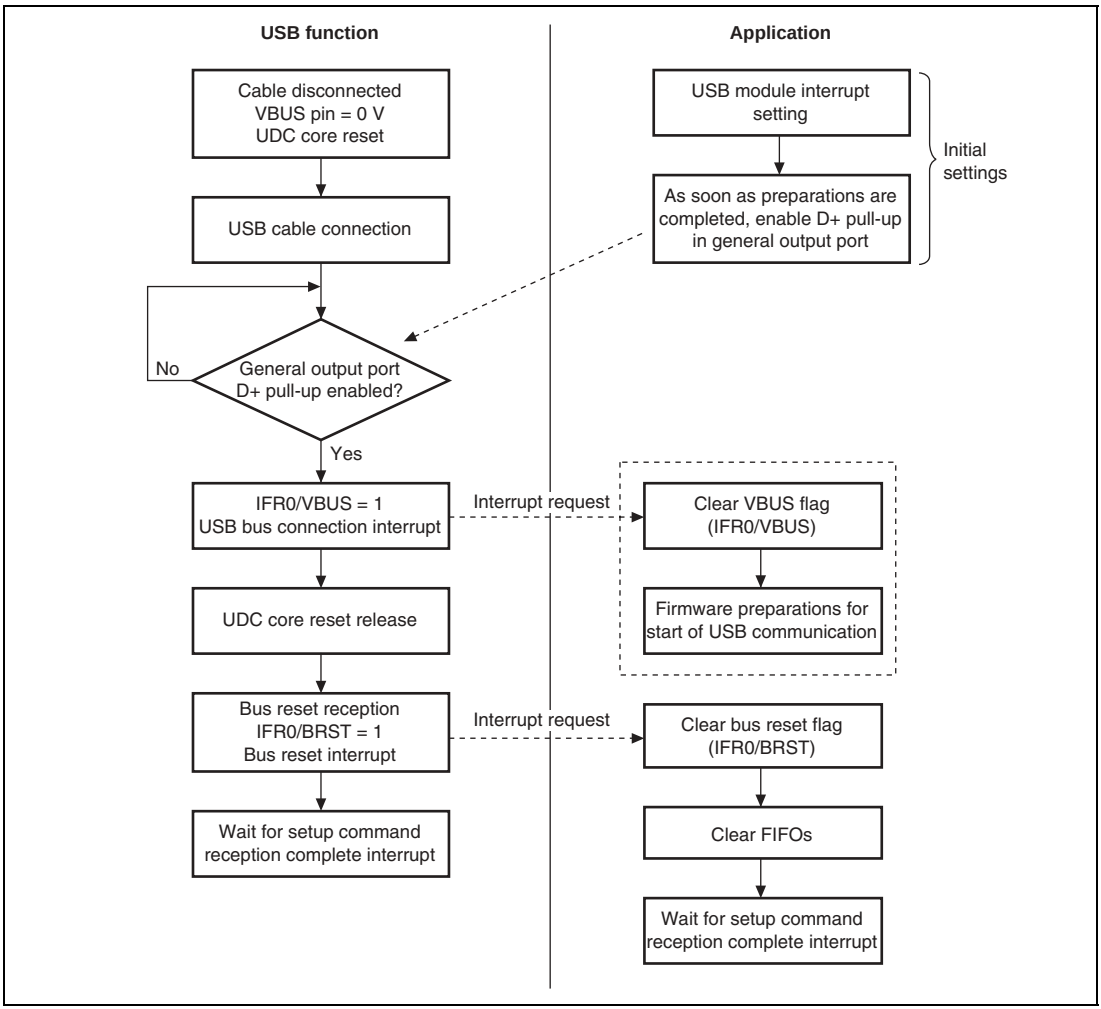


Figure 20.4 Cable Connection Operation

The above flowchart shows the operation in the case of in section 20.7, Example of External Circuitry for USB Function Controller.

In applications that do not require USB cable connection to be detected, processing by the USB bus connection interrupt is not necessary. Preparations should be made with the bus reset interrupt.

For details, see section 20.7, Example of External Circuitry for USB Function Controller.

20.4.2 Cable Disconnection

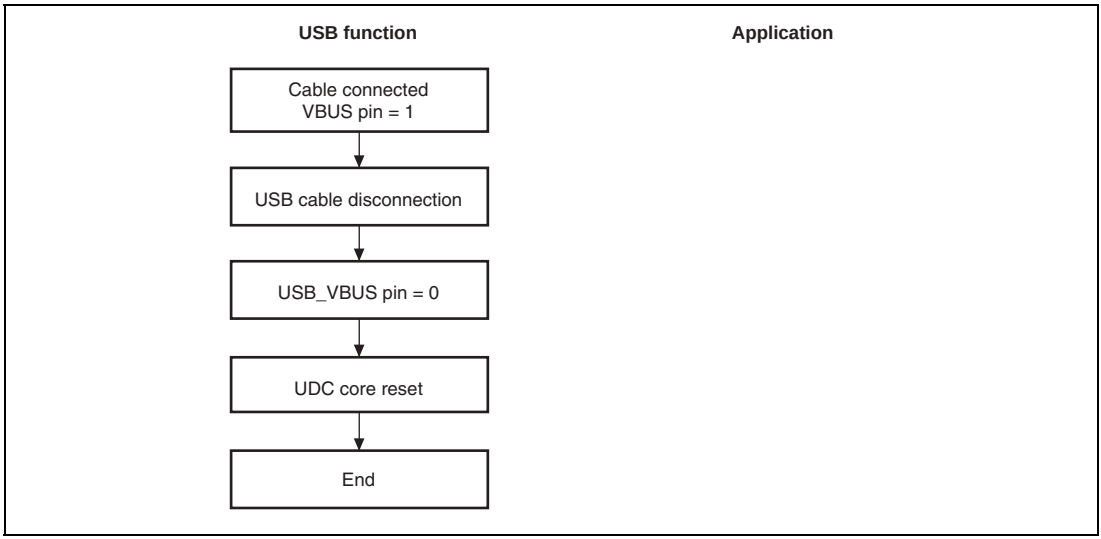


Figure 20.5 Cable Disconnection Operation

The above flowchart shows the operation in section 20.7, Example of External Circuitry for USB Function Controller.

For details, see section 20.7, Example of External Circuitry for USB Function Controller.

20.4.3 Control Transfer

Control transfer consists of three stages: setup, data (not always included), and status (figure 20.6). The data stage comprises a number of bus transactions. Operation flowcharts for each stage are shown below.

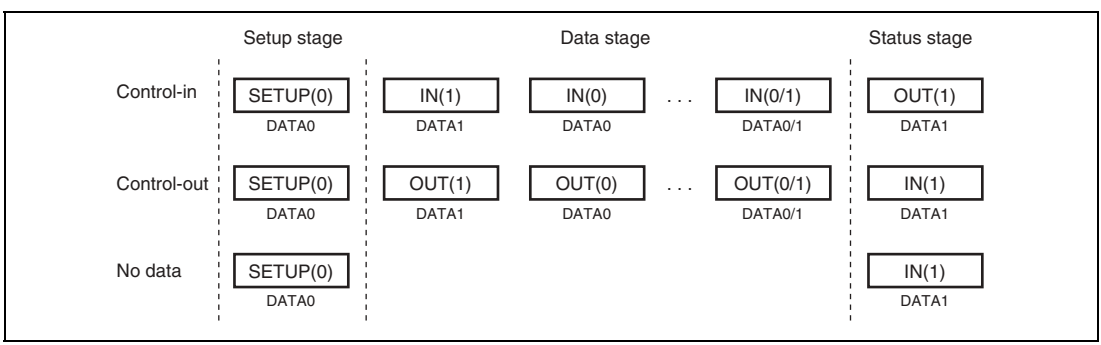
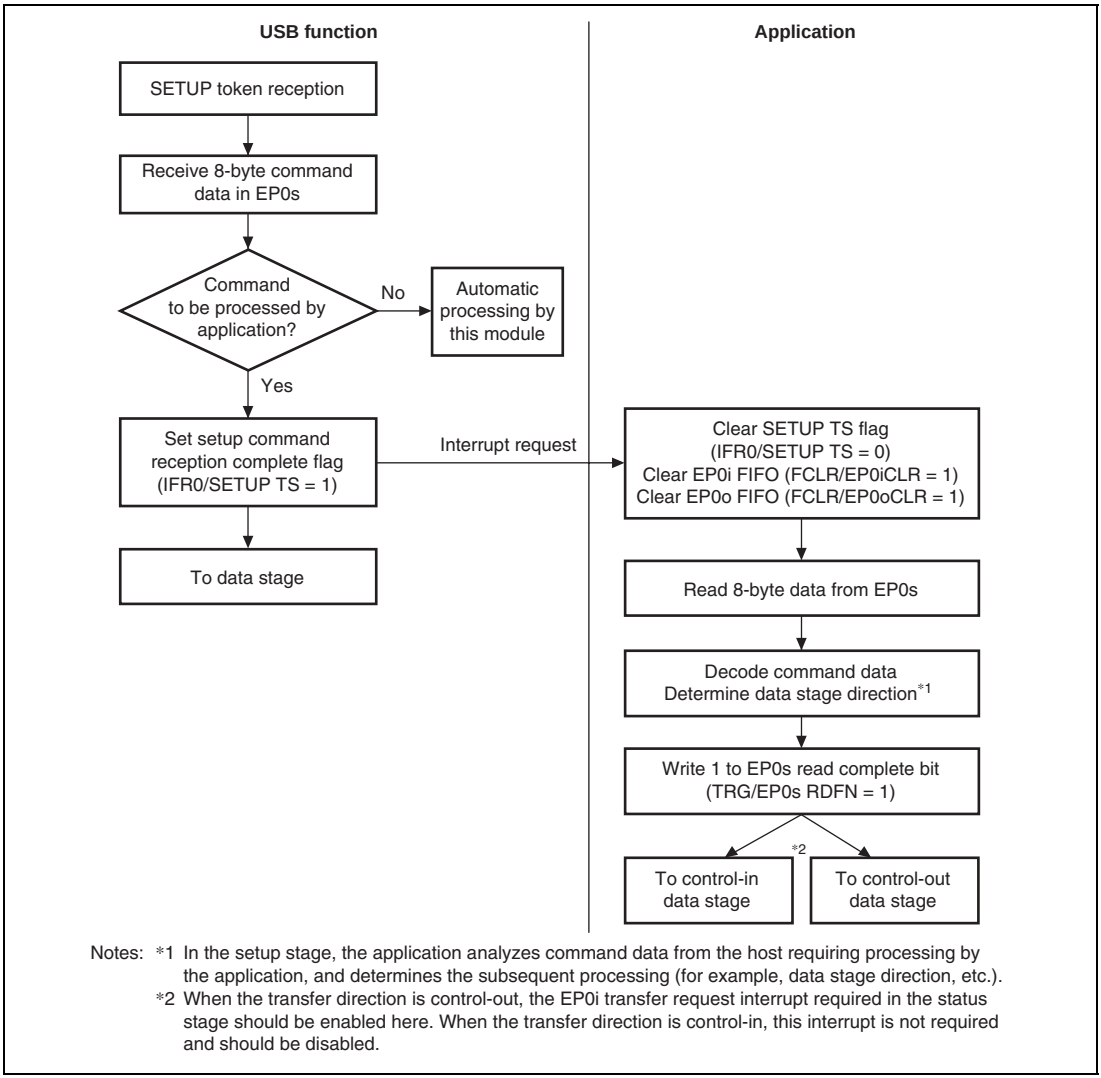


Figure 20.6 Transfer Stages in Control Transfer

Setup Stage:



Notes: *1 In the setup stage, the application analyzes command data from the host requiring processing by the application, and determines the subsequent processing (for example, data stage direction, etc.).
*2 When the transfer direction is control-out, the EP0i transfer request interrupt required in the status stage should be enabled here. When the transfer direction is control-in, this interrupt is not required and should be disabled.

Figure 20.7 Setup Stage Operation

Data Stage (Control-In):

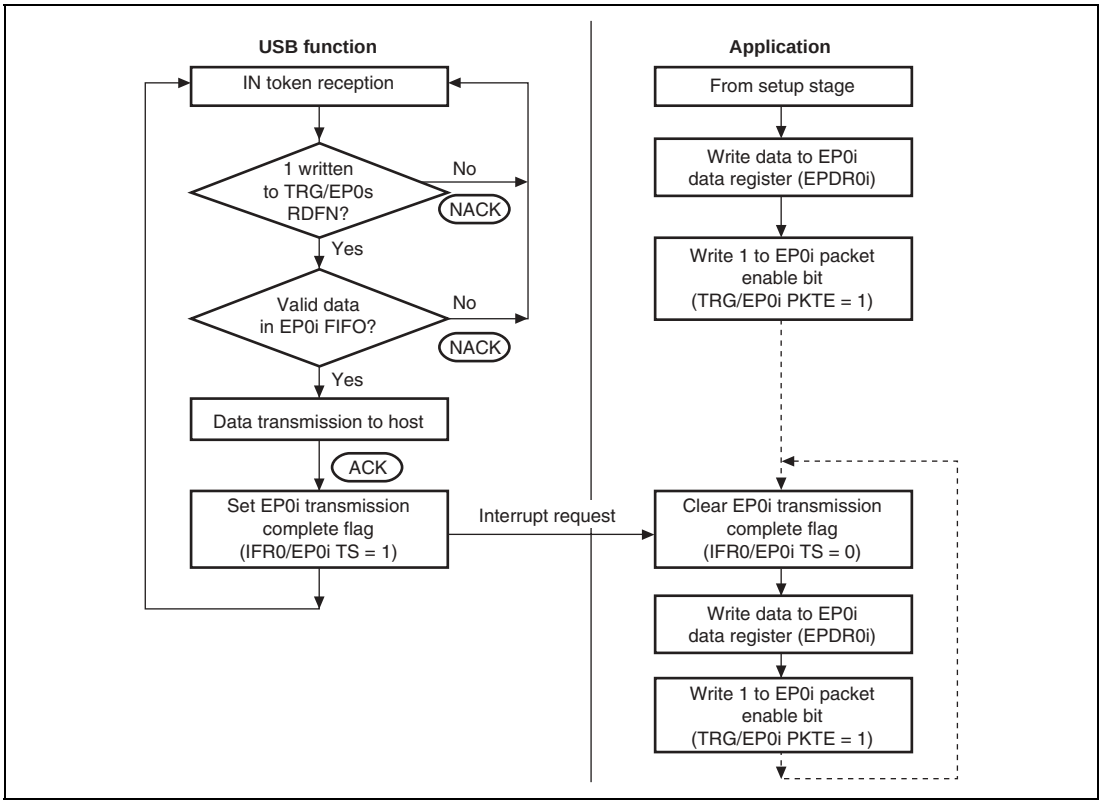


Figure 20.8 Data Stage (Control-In) Operation

The application first analyzes command data from the host in the setup stage, and determines the subsequent data stage direction. If the result of command 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 (IFR0/EP0i TS = 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.

Data Stage (Control-Out):

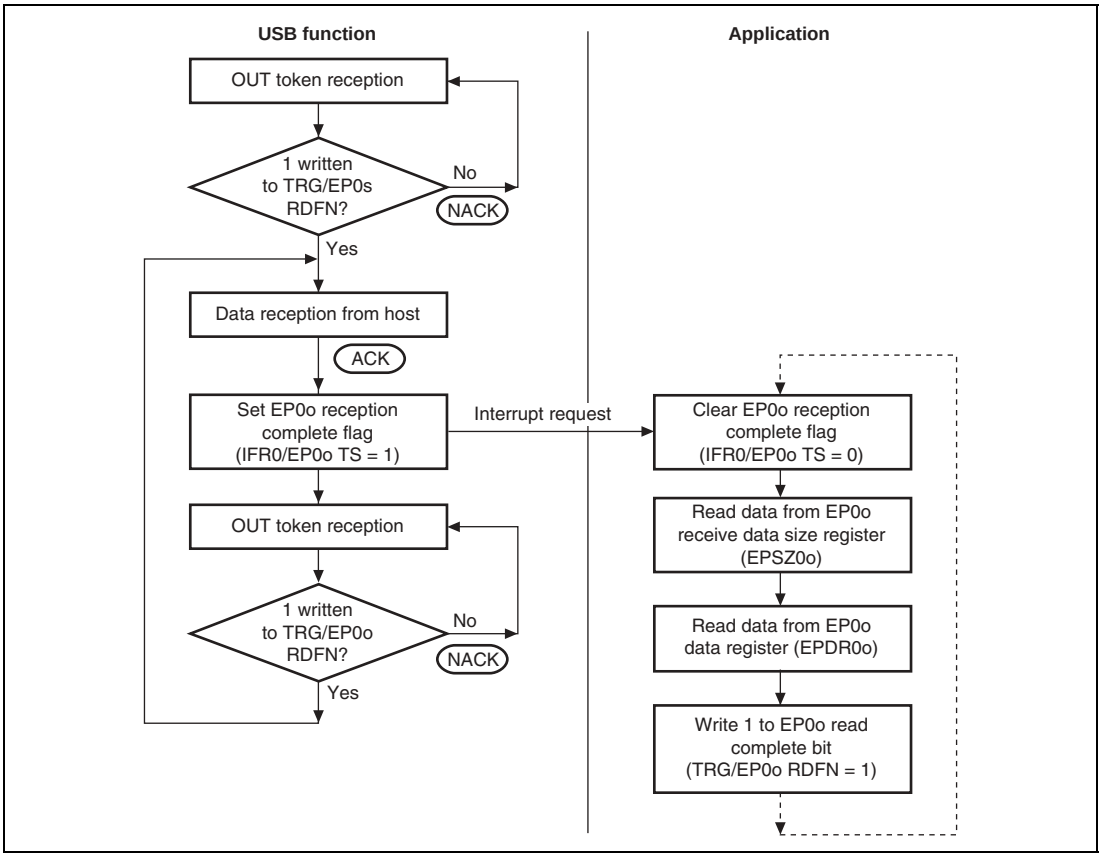


Figure 20.9 Data Stage (Control-Out) Operation

The application first analyzes command data from the host in the setup stage, and determines the subsequent data stage direction. If the result of command data analysis is that the data stage is out-transfer, the application waits for data from the host, and after data is received (IFR0/EP0o TS = 1), reads data from the FIFO. Next, the application writes 1 to the EP0o read complete bit, empties the receive FIFO, and waits for reception of the next data.

The end of the data stage is identified when the host transmits an IN token and the status stage is entered.

Status Stage (Control-In):

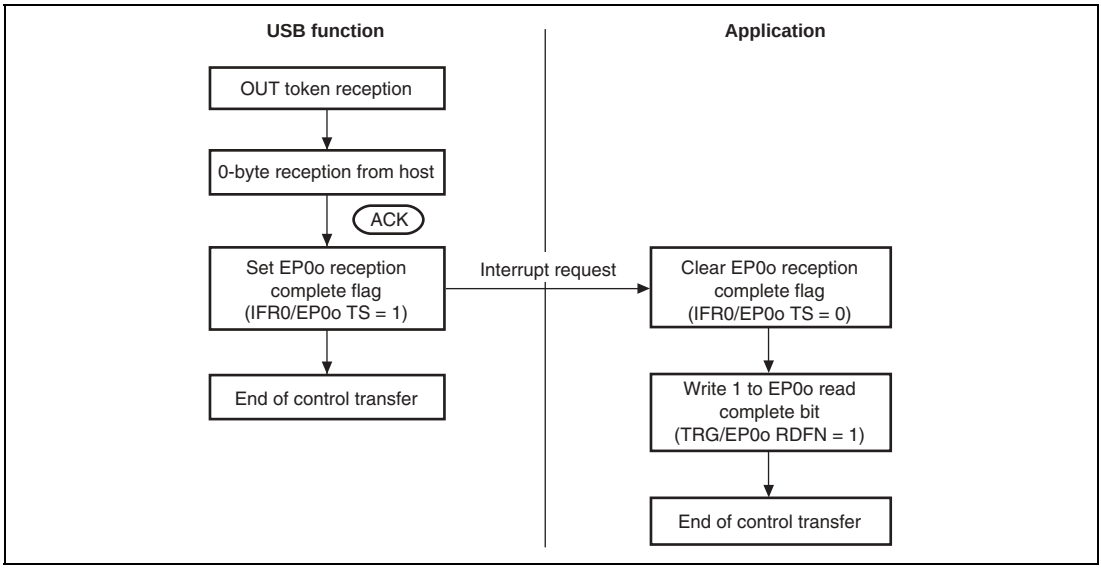


Figure 20.10 Status Stage (Control-In) Operation

The control-in status stage starts with an OUT token from the host. The application receives 0-byte data from the host, and ends control transfer.

Status Stage (Control-Out):

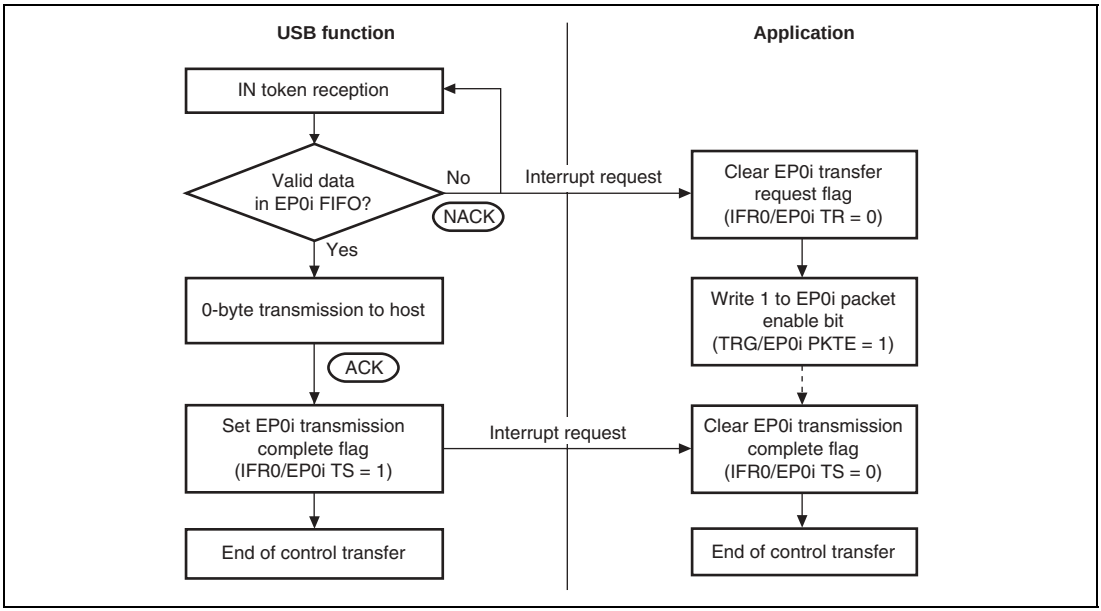
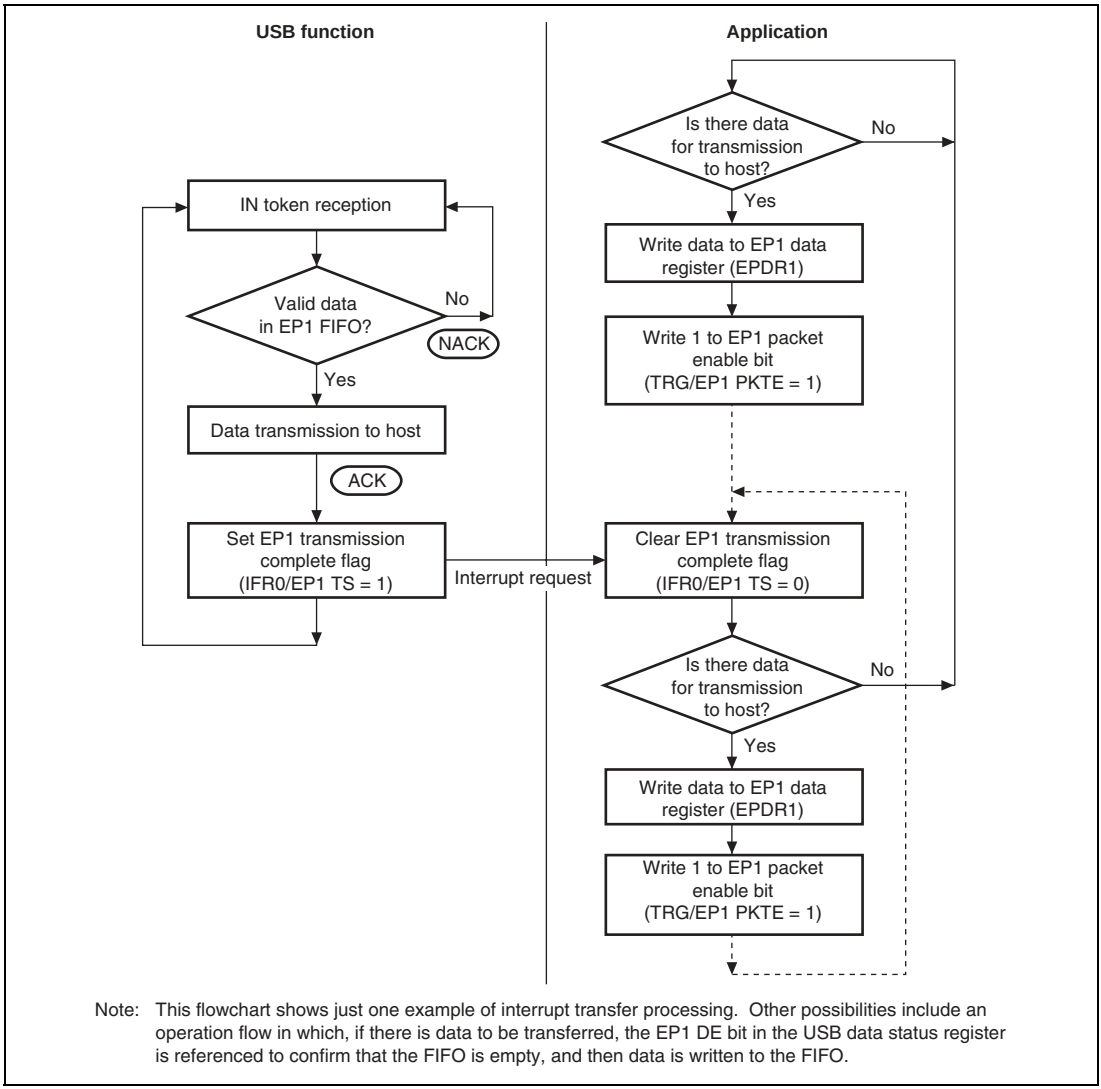


Figure 20.11 Status Stage (Control-Out) Operation

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 application recognizes from this interrupt that the status stage has started. Next, in order to transmit 0-byte data to the host, 1 is written to the EP0i packet enable bit but no data is written to the EP0i FIFO. As a result, the next IN token causes 0-byte data to be transmitted to the host, and control transfer ends.

After the application has finished all processing relating to the data stage, 1 should be written to the EP0i packet enable bit.

20.4.4 EP1, 4 Interrupt-In Transfer



Note: This flowchart shows just one example of interrupt transfer processing. Other possibilities include an operation flow in which, if there is data to be transferred, the EP1 DE bit in the USB data status register is referenced to confirm that the FIFO is empty, and then data is written to the FIFO.

Figure 20.12 EP1 Interrupt-In Transfer Operation

20.4.5 EP2i, 5 Bulk-In Transfer (Dual FIFOs)

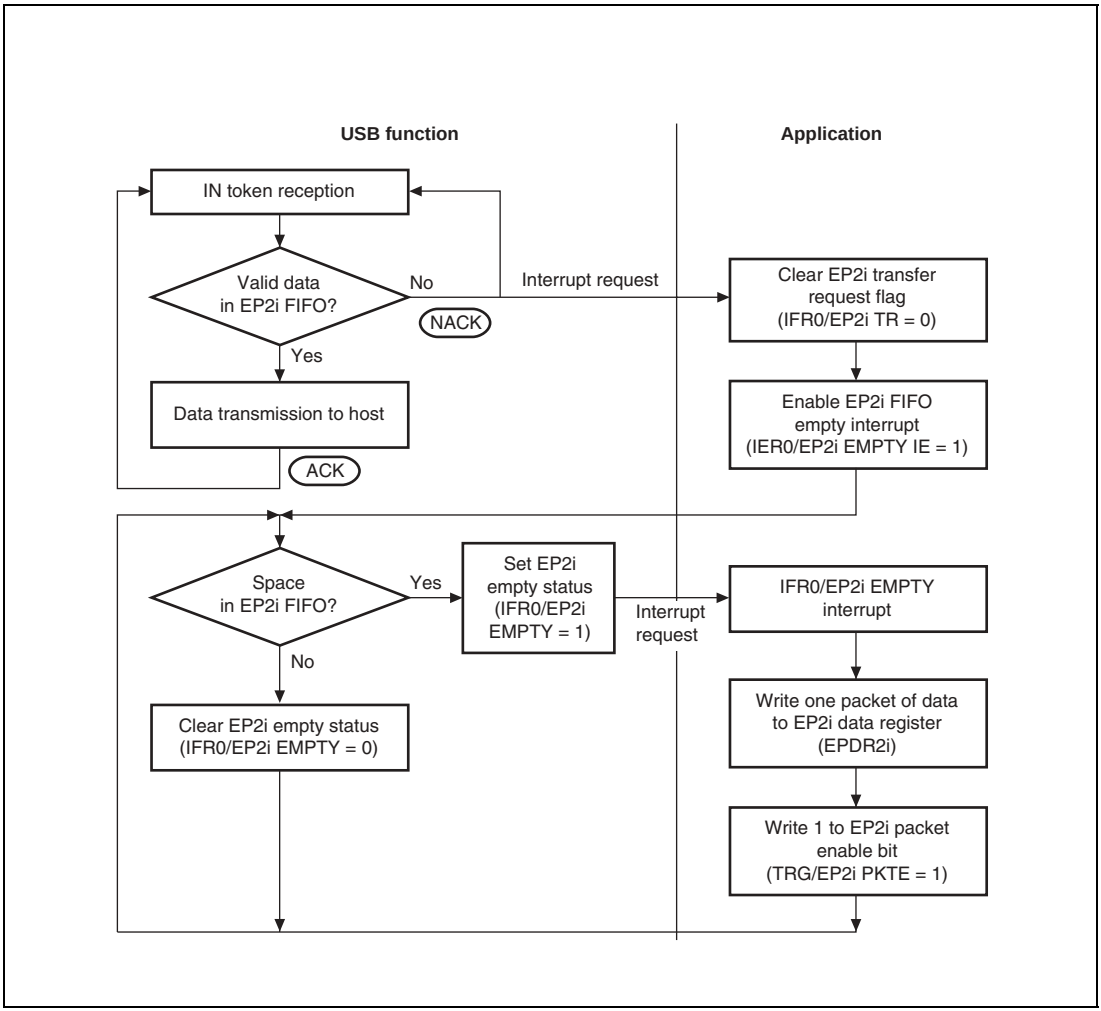


Figure 20.13 EP2 Bulk-In Transfer Operation

EP2i has two 64-byte FIFOs, but the user can perform data transmission and transmit data writes without being aware of this dual-FIFO configuration. However, one data write is performed for one FIFO. For example, even if both FIFOs are empty, it is not possible to perform EP2i/PKTE at one time after consecutively writing 128 bytes of data. EP2i/PKTE must be performed for each 64-byte write.

When performing bulk-in transfer, as there is no valid data in the FIFOs on reception of the first IN token, an IFR0/EP2i TR interrupt is requested. With this interrupt, 1 is written to the IER0/EP2i EMPTY IE bit, and the EP2i FIFO empty interrupt is enabled. At first, both EP2i FIFOs are empty, and so an EP2i 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, IFR0/EP2i EMPTY is cleared to 0. If at least one FIFO is empty, IFR0/EP2i EMPTY 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 IER0/EP2i EMPTY IE and disable interrupt requests.

20.4.6 EP2o, 6 Bulk-Out Transfer (Dual FIFOs)

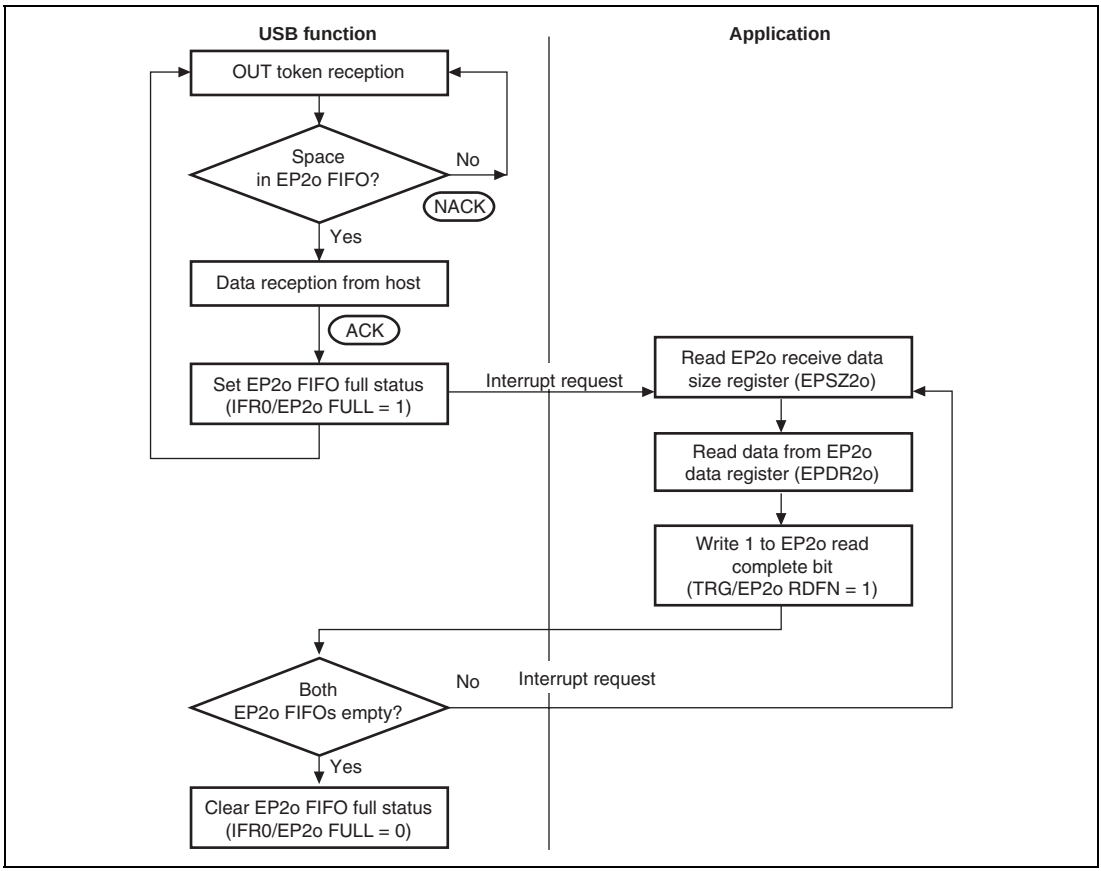


Figure 20.14 EP2o Bulk-Out Transfer Operation

EP2o has two 64-byte FIFOs, but the user can perform data reception and receive data reads without being aware of this dual-FIFO configuration.

When one FIFO is full after reception is completed, the IFR0/EP2o FULL bit 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, NACK is returned to the host automatically. When reading of the receive data is completed following data reception, 1 is written to the TRG/EP2o RDFN bit. This operation empties the FIFO that has just been read, and makes it ready to receive the next packet.

20.4.7 EP3i Isochronous-In Transfer

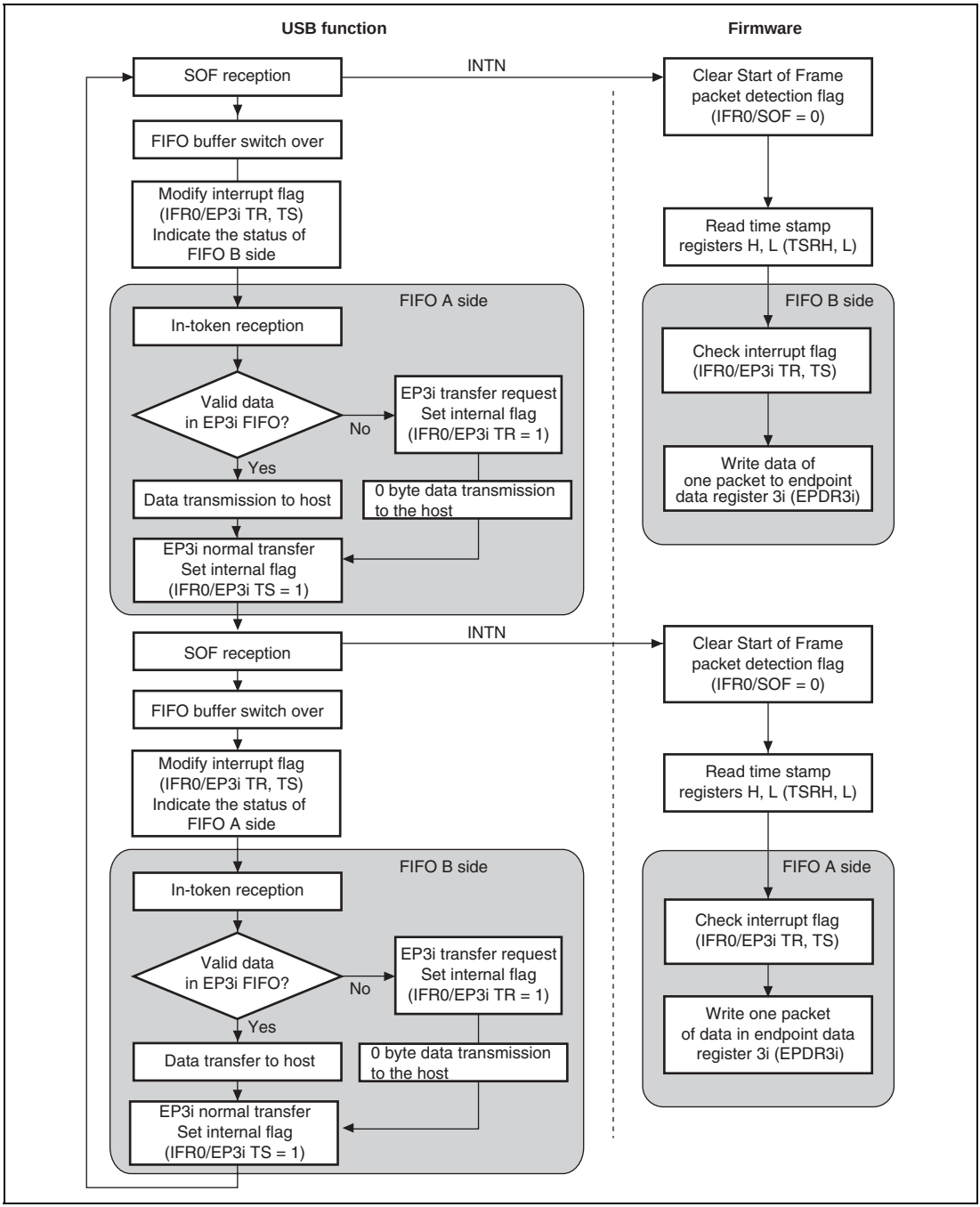


Figure 20.15 Operation of Isochronous-In Transfer

EP3i has two 64-byte FIFOs in maximum, but the user can perform data transmission and write transmit data without being aware of this dual-FIFO configuration. (In figure 20.15, FIFO sides A and B are used for description.)

In isochronous transfer, transfer is occurred only once per one frame (1 ms). So, when SOF is received, the FIFO buffer is switched automatically with hardware (the FIFO buffer is switched automatically at a cycle of 1 ms by hardware if the SOF marker function is enabled even in the case that SOF cannot be received by error).

FIFO buffers are switched over by the SOF reception. Therefore, the FIFO buffer in which the USB function module transmits the data to the host and the FIFO buffer in which the firmware writes the transmit data have different buffers, and a read and write of FIFO buffer are not competed. Accordingly, the data written by the firmware is the data to be transmitted in one frame after. The buffers of FIFOs are switched over automatically by the SOF reception, so a write of data must be completed within the frame.

The USB function module transmits data to the host and set the TS internal flag to 1 when data to be transmitted to the host exists in FIFO after an in-token is received. If there is no data in the FIFO buffer, set the TR internal flag to 1 and transmit 0-byte data to the host. The internal flag (TR, TS) information is automatically modified as IFR0 flag information (TR, TS) that can be read by the SOF reception from the firmware.

In firmware, first, the processing routine of the isochronous transfer is called by SOF interrupt to check the time stamp. Then one packet data is written to FIFO. This written data is transmitted to the host in the next frame. Whether the previous frame has been transmitted normally or not can be determined by reading the IFR0 flag information (TR, TS).

20.4.8 EP3o Isochronous Out Transfer

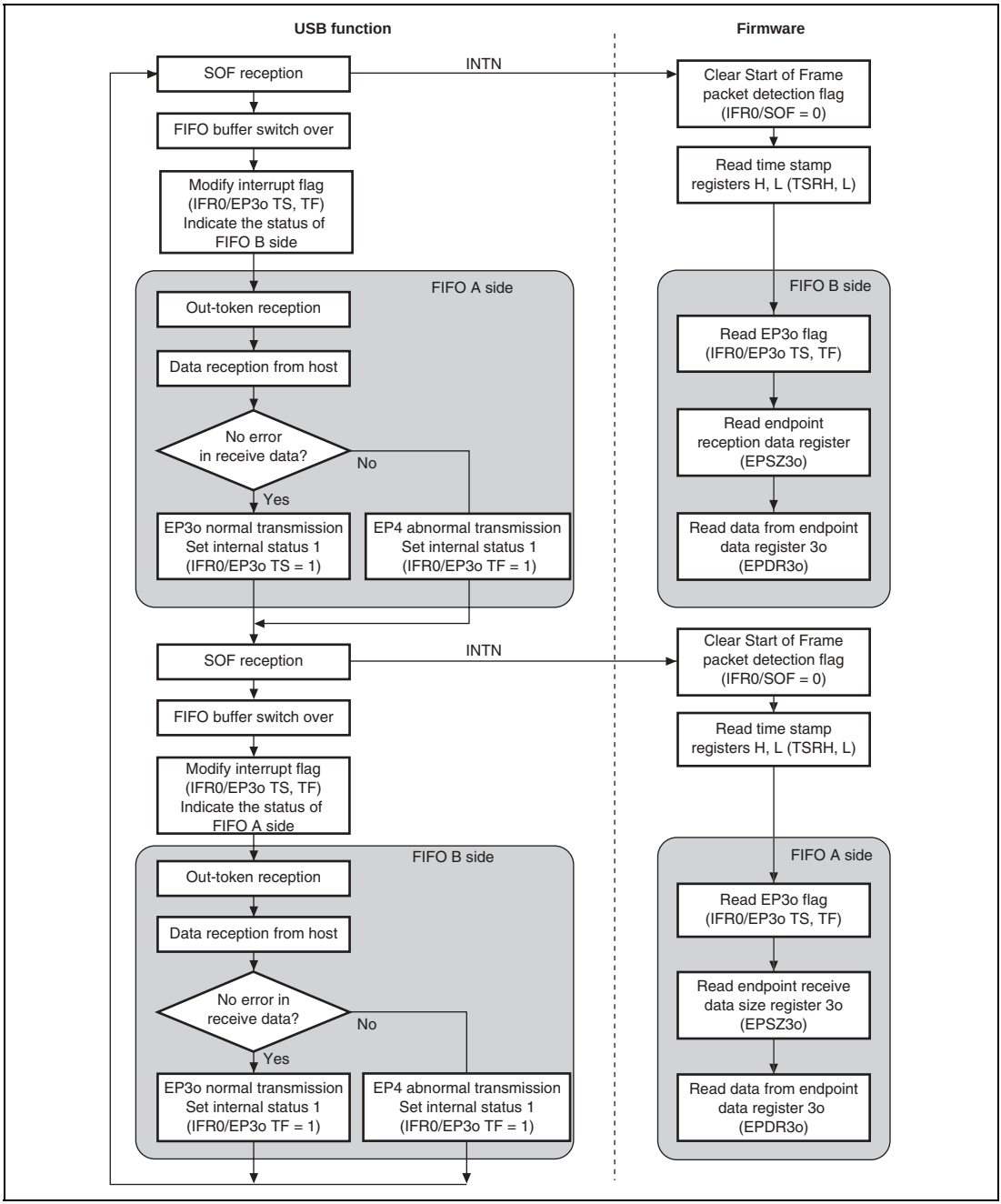


Figure 20.16 Operation of EP3o Isochronous-Out Transfer

EP3o has two 60-byte FIFOs in maximum, but the user can perform data transmission and read transmit data without being aware of this dual-FIFO configuration. (In figure 20.16, FIFO sides A and B are used for description.)

In isochronous transfer, transfer is occurred only once per one frame (1 ms). So, when SOF is received the FIFO buffer is switched automatically with hardware (the FIFO buffer is switched automatically at a cycle of 1 ms by hardware if the SOF marker function is enabled even in the case that SOF cannot be received by error).

FIFO buffers are switched over by the SOF reception. Therefore, the FIFO buffer in which the USB function module receives the data from the host and the FIFO buffer in which the firmware reads the receive data have different buffers, and a read and write of FIFO buffer are not competed. Accordingly, the data written by the firmware is the data received in one frame before. The buffers of FIFOs are switched over automatically by the SOF reception, so a read of data must be completed within the frame.

The USB function module receives data from the host after an out-token is received. If there is an error in the data, set the TF internal flag to 1. If there is no error in the data, set the TS internal flag to 1. The internal flag (TF, TS) information is automatically modified as IFR0 flag information (TF, TS) that can be read by the SOF reception from the firmware.

In firmware, first, the processing routine of the isochronous transfer is called by SOF interrupt to check the time stamp. Then data is read from the FIFO buffer. IFR0 flag information (TS, TF) is read and decided if the data has an error. The IFR0 flag information at this time represents the status of the currently readable FIFO buffer.

20.5 Processing of USB Standard Commands and Class/Vendor Commands

20.5.1 Processing of Commands Transmitted by Control Transfer

A command transmitted from the host by control transfer may require decoding and execution of command processing on the application side. Whether command decoding is required on the application side is indicated in table 20.7 below.

Table 20.7 Command Decoding on Application Side

Decoding not Necessary on Application Side	Decoding Necessary on Application Side
Clear Feature	Get Descriptor
Get Configuration	Class/Vendor command
Get Interface	Synch Frame
Get Status	Set Descriptor
Set Address	
Set Configuration	
Set Feature	
Set Interface	

If decoding is not necessary on the application side, command decoding, 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 on the application side, this module stores the command in the EP0s FIFO. After normal reception is completed, the IFR0/SETUP TS flag is set and an interrupt request is generated. In the interrupt routine, 8 bytes of data must be read from the EP0s data register (EPDR0s) and decoded by firmware. The necessary data stage and status stage processing should then be carried out according to the result of the decoding operation.

20.6 Stall Operations

20.6.1 Overview

This section describes stall operations in this module. There are two cases in which the USB function module stall function is used:

- When the application forcibly stalls an endpoint for some reason
- When a stall is performed automatically within the USB function module due to a USB specification violation

The USB function 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 references these internal status bits and determines whether to return a stall to the host. These bits cannot be cleared by the application; they must be cleared with a Clear Feature command from the host.

However, the internal status bit to EP0 is automatically cleared only when the setup command is received.

20.6.2 Forcible Stall by Application

The application uses the EPSTL register to issue a stall request for the USB function module. When the application wishes to stall a specific endpoint, it sets the corresponding bit in EPSTL (1-1 in figure 20.17). The internal status bits are not changed at this time. When a transaction is sent from the host for the endpoint for which the EPSTL bit was set, the USB function module references the internal status bit, and if this is not set, references the corresponding bit in EPSTL (1-2 in figure 20.17). If the corresponding bit in EPSTL is set, the USB function module sets the internal status bit and returns a stall handshake to the host (1-3 in figure 20.17). If the corresponding bit in EPSTL 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 command from the host, without regard to the EPSTL register. Even after a bit is cleared by the Clear Feature command (3-1 in figure 20.17), the USB function module continues to return a stall handshake while the bit in EPSTL is set, since the internal status bit is set each time a transaction is executed for the corresponding endpoint (1-2 in figure 20.17). To clear a stall, therefore, it is necessary for the corresponding bit in EPSTL to be cleared by the application, and also for the internal status bit to be cleared with a Clear Feature command (2-1, 2-2, and 2-3 in figure 20.17).

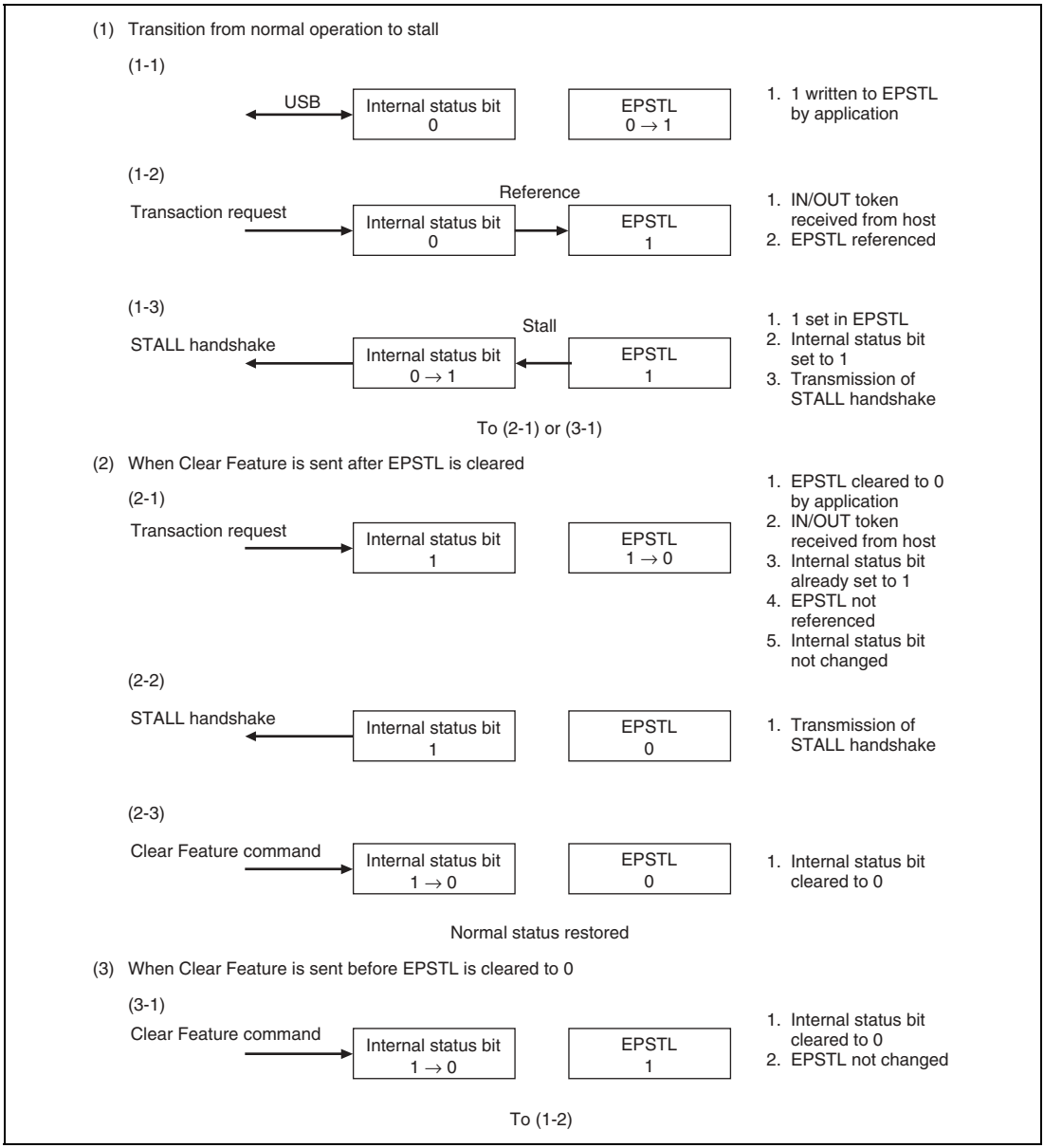


Figure 20.17 Forcible Stall by Application

20.6.3 Automatic Stall by USB Function Module

When a stall setting is made with the Set Feature command, or in the event of a USB specification violation, the USB function module automatically sets the internal status bit for the relevant endpoint without regard to the EPSTL register, and returns a stall handshake (1-1 in figure 20.18).

Once an internal status bit is set, it remains set until cleared by a Clear Feature command from the host, without regard to the EPSTL register. After a bit is cleared by the Clear Feature command, EPSTL is referenced (3-1 in figure 20.18). The USB function 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 20.18). To clear a stall, therefore, the internal status bit must be cleared with a Clear Feature command (3-1 in figure 20.18). If set by the application, EPSTL should also be cleared (2-1 in figure 20.18).

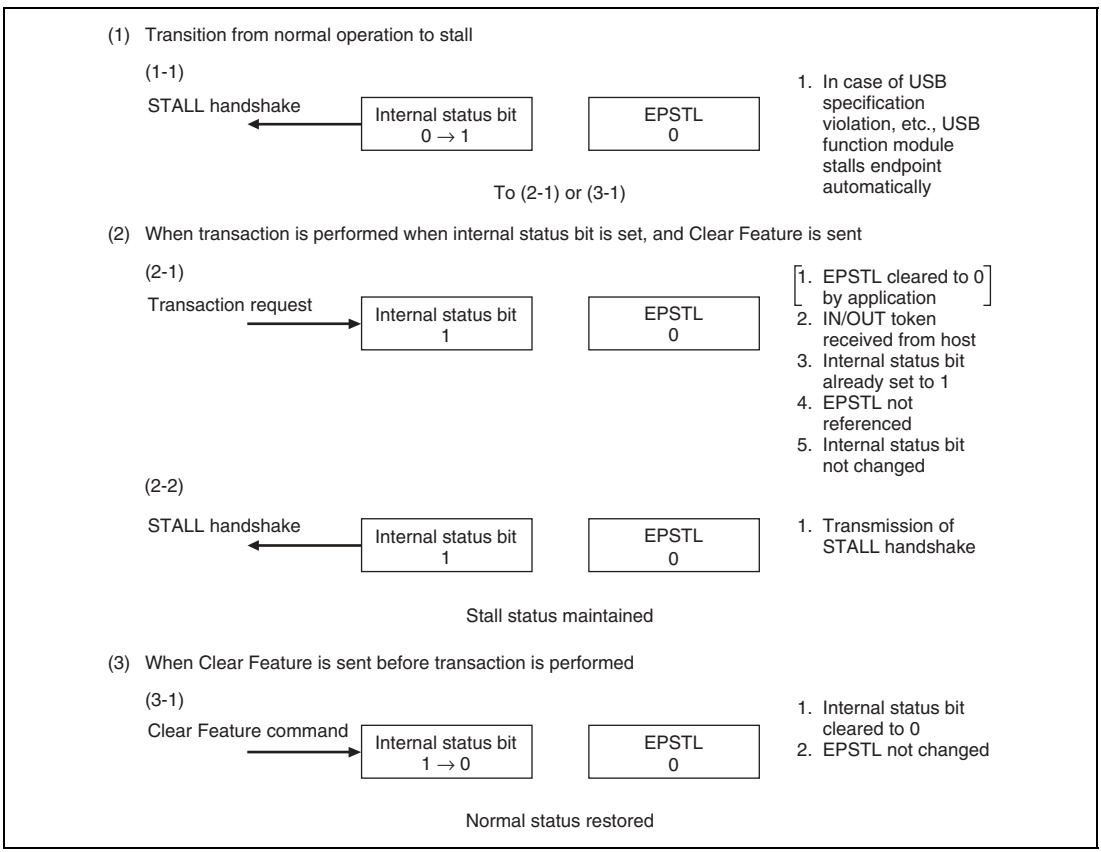
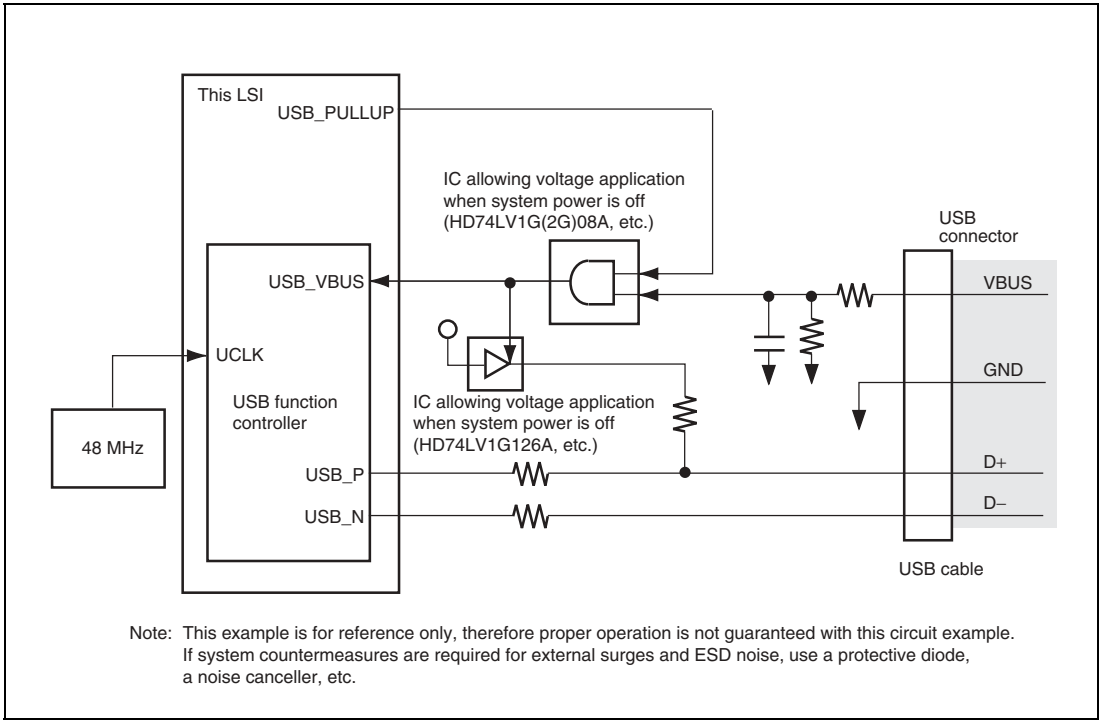


Figure 20.18 Automatic Stall by USB Function Module

20.7 Example of External Circuitry for USB Function Controller

D+ Pull-Up Control: In a system where it is wished to disable USB host/hub connection notification (D+ pull-up) (during high-priority processing or initialization processing, for example), D+ pull-up should be controlled using a general output port. However, if a USB cable is already connected to the host/hub and D+ pull-up is prohibited, D+ and D– will both go low (both of D+ and D– pulled down on the host/hub side) and the USB function controller will mistakenly identify this as reception of a USB bus reset from the host. Therefore, the D+ pull-up control signal and USB_VBUS pin input signal should be controlled using an UPLUP output port and the USB cable VBUS (AND circuit) as shown in the circuit example below (The UDC core in this LSI maintains the powered state when the USB_VBUS pin is low, regardless of the D+/D– state.)

Detection of USB Cable Connection/Disconnection: As USB states, etc., are managed by hardware in this module, a VBUS signal that recognizes connection/disconnection is necessary. The power supply signal (VBUS) in the USB cable is used for this purpose. However, if the cable is connected to the USB host/hub when the function (system installed in this LSI) power is off, a voltage (5 V) will be applied from the USB host/hub. Therefore, an IC (such as an HD74LV1G08A or 2G08A) that allows voltage application when the system power is off should be connected externally.



Note: This example is for reference only, therefore proper operation is not guaranteed with this circuit example. If system countermeasures are required for external surges and ESD noise, use a protective diode, a noise canceller, etc.

Figure 20.19 Example of USB Function Module External Circuitry

20.8 Usage Notes

20.8.1 Setup Data Reception

The following points should be noted on EPDR0s in which reception of 8-byte setup data is performed.

1. Since the setup command must be received in the USB, writing from the USB bus side is prior to reading from the CPU side. While the CPU reads data after completion of reception and reception of the next setup command is started, reading from the CPU side is forcibly invalid in order to give priority to writing. Therefore a value to be read after starting reception is undefined.
2. EPDR0s must be read in 8-byte units. If reading is suspended while it is in progress, data received in the next setup cannot be read successfully.

20.8.2 FIFO Clear

When the USB cable is disconnected during communication, data which is receiving or transmitting may remain in the FIFO. Therefore the FIFO must be cleared immediately after connecting the USB cable again.

Note that the FIFO in which data is receiving from the host or transmitting to the host must not be cleared.

20.8.3 Overreading/Overwriting of Data Register

The following points should be noted when the data register of the USB function controller is read from or written to.

Receive Data Register: The receive data register must not read data which is more than valid receive data bytes. That is, data which is more than bytes indicated in the receive data size register must not be read. In case of the receive data register which has the dual FIFO buffer, the maximum number of data which can be read in a single time is maximum packet size. Write 1 to TRG/RDFN after data in the current valid buffer is read. This writing switches the FIFO buffer. Then, the new number of bytes is reflected in the receive data size and the next data can be read.

Transmit Data Register: The transmit data register must not write data which is more than maximum packet size. In case of the transmit data register which has the dual FIFO buffer, the maximum number of data which can be written in a single time is maximum packet size. Write 1 to TRG/PKTE after data is written. This writing switches the FIFO buffer. Then, the next data can be written to another buffer. Therefore data must not be written in both buffers in a single time.

20.8.4 Assigning EP0 Interrupt Sources

The EP0 interrupt sources assigned to IFR0 (bits 0, 1, 2, and 29) must be assigned to the same interrupt pins by ISR0. The other interrupt sources have no restrictions.

20.8.5 FIFO Clear when DMA Transfer is Set

When the DMA transfer is enabled in endpoints 2o and 6, the data register cannot be cleared. Cancel the DMA transfer before clearing the data register.

20.8.6 Note on Using TR Interrupt

The bulk-in transfer has a transfer request interrupt (TR interrupt). The following points should be noted when using a TR interrupt.

When the IN token is sent from the USB host and there is no data in the corresponding EP FIFO, the TR interrupt flag is set. However, the TR interrupt is generated continuously at the timing as shown in figure 20.20. In this case, note that erroneous operation should not occur.

Note: When the IN token is received and there is no data in the corresponding EP FIFO, an NACK is determined. However, the TR interrupt flag is set after an NACK handshake is transmitted. Therefore when the next IN token is received before TRG/PKTE is written, the TR interrupt flag is set again.

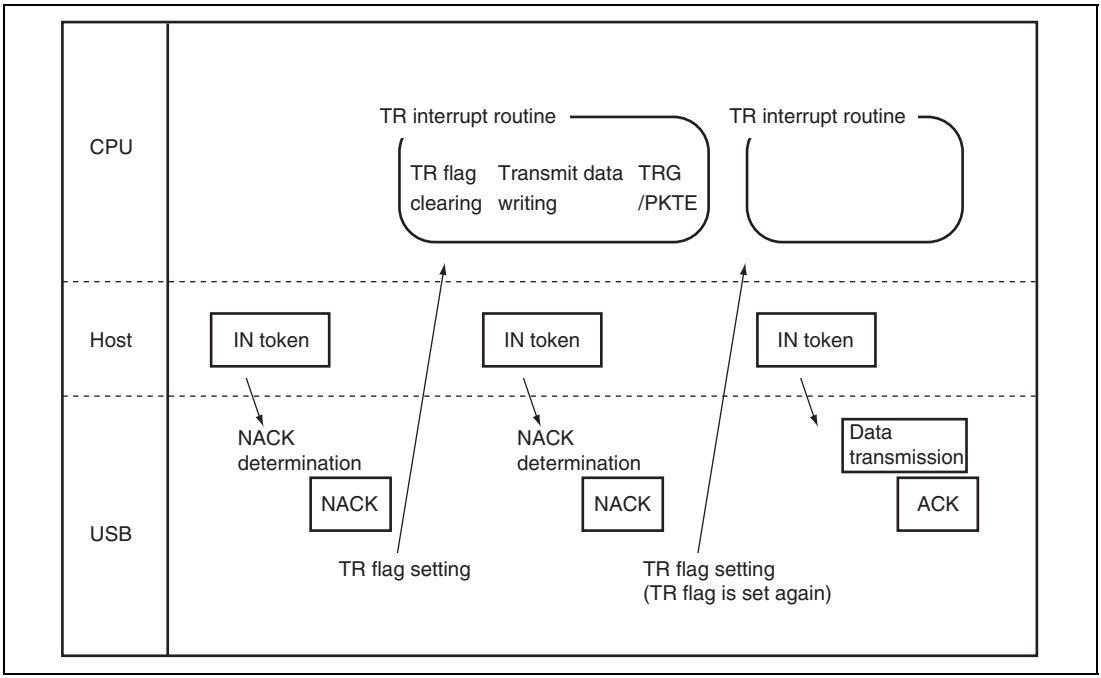


Figure 20.20 Set Timing of TR Interrupt Flag

20.8.7 Note on Clearing and Transition to Software Standby Mode

Figure 20.21 shows an example flow of clearing and transition to software standby mode.

When accessing the USBF again after software standby mode has been cleared, wait for operation stabilization time of the USB clock (48 MHz).

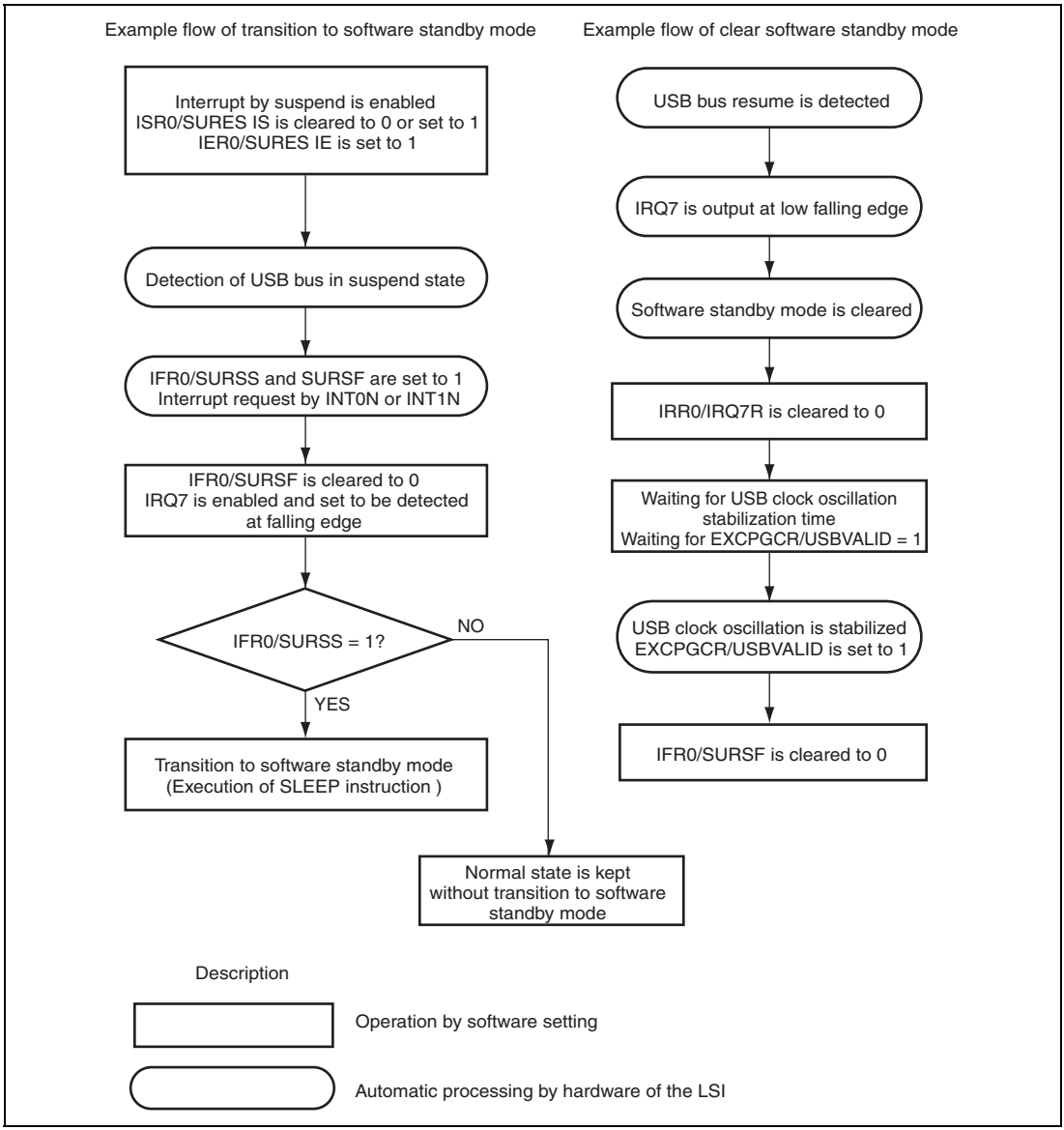


Figure 20.21 Example Flow of Clearing and Transition to Software Standby Mode

20.8.8 Main Clock Usage when Using Bus Password Function

When the main clock of this LSI (usually input from EXTAL) is input using the RF chip for the bluetooth as is the clock for the bluetooth interface (usually input from RDI_REFCLK_IN), clock cannot be provided again on the recovery from the suspend state. Therefore, when using the bus-powered function, a clock in active state should be used as the main clock. To hold down the power consumption in suspend state, the crystal oscillator should be connected to the EXTAL and XTAL pins, in order to provide main clock to the CPU.

20.8.9 Peripheral clock for USB

When using USB Function Controller, the frequency of the peripheral clock (Pϕ) should be 12MHz or larger. With the clock frequency less than 12MHz, the USB Function Controller does not operate normally. Refer to section 11, Clock Pulse Generator (CPG) for a detailed setting of the clock.

Section 21 Bluetooth Interface (BT)

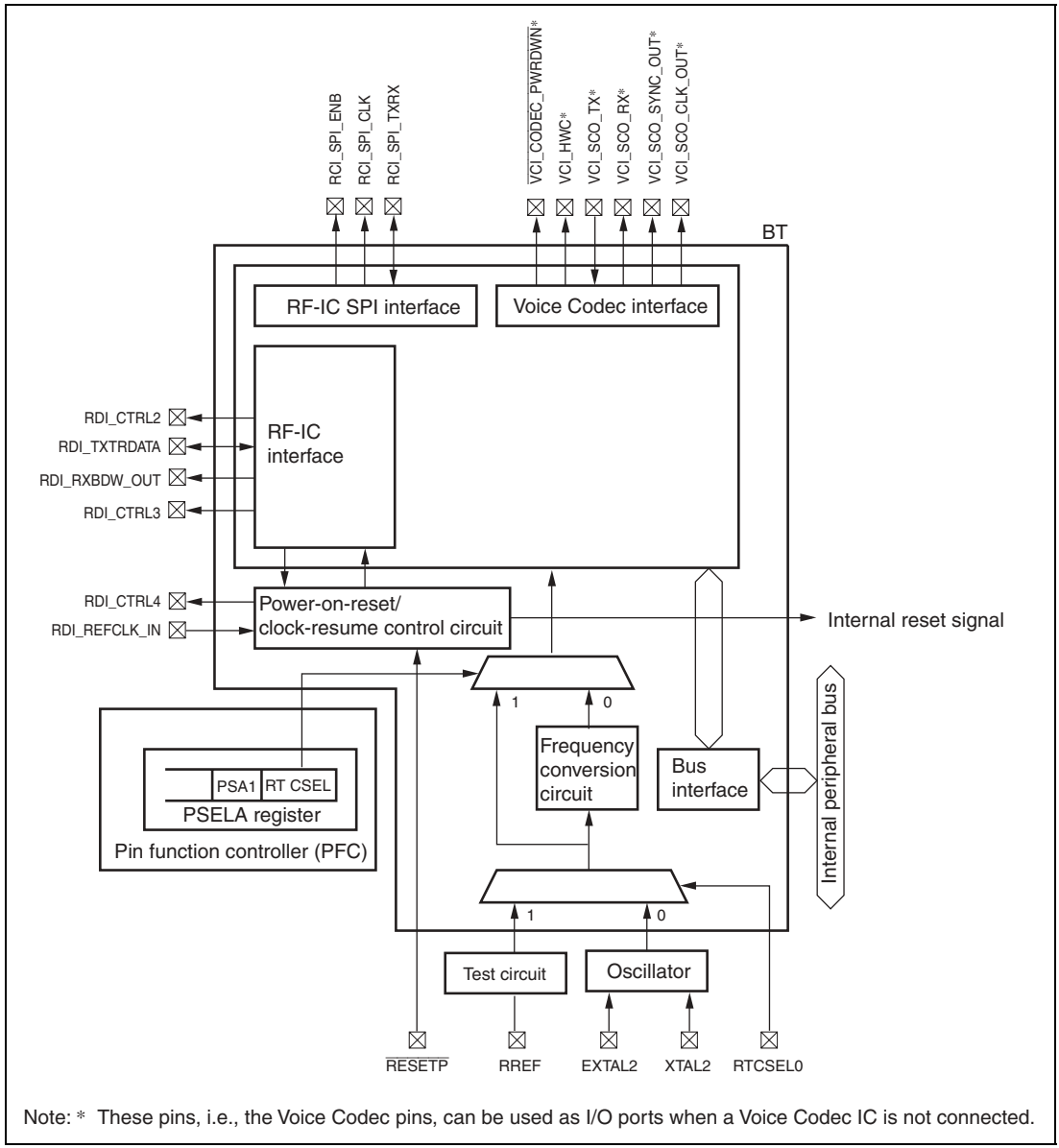
This LSI incorporates a Bluetooth interface (BT) as hardware for the baseband processing for the Bluetooth. This Bluetooth interface conforms to version 1.1 of the Bluetooth Specification, is configured by combination of hardware and firmware, and provides a lower protocol stack function to the host controller interface (HCI) layer. With this function, Bluetooth logo certification is acquired for a baseband layer (BB) and link management protocol layer (LMP).

To configure the upper protocol stack, HCI commands which are defined by version 1.1 of the Bluetooth Specification and 53 types of test command interface (TCI) commands are supported. For communication between upper and lower protocol stacks, nine types of functions are prepared as an application interface (API).

21.1 Features

- Conforms to version 1.1 of the Bluetooth Specification
- Frequency hopping within 79 channels as a form of spread-spectrum modulation
- Supports ACL (Asynchronous Connection-Less) and SCO (Synchronous Connection-Oriented) links
- Controls the ACL and SCO from upper layer above HCI
- Supports all types of packets: i.e., control packets for establishing links and connections, and packets for transmission and received packets
- Supports point-to-point and point-to-multipoint connections (piconet)
- Mutual conversion in Voice Codec between A-law and μ-law, CVSD and linear PCM, linear PCM and A/μ-law, or A/μ-law and CVSD
- Encryption/decryption
- Controls three power-down modes: the Hold, Sniff and Park modes
- Supports a direct interface with the Renesas RF ICs, such as the HD157100NP and HD157102NP
- Supports a direct interface with the STLC7550 (ST Microelectronics) or MC145483 (Motorola) Voice Codec IC without involving the CPU
- Includes a frequency conversion function that allows the use of a 32.768-kHz crystal resonator to provide the low-frequency clock for power-down modes
- Supports unique TCI commands (53 types) and API functions (9 types) in addition to the HCI commands

Figure 21.1 shows a block diagram of the Bluetooth interface (BT).



Note: * These pins, i.e., the Voice Codec pins, can be used as I/O ports when a Voice Codec IC is not connected.

Figure 21.1 Block Diagram of Bluetooth Interface (BT)

21.2 Input/Output Pins

Table 21.1 summarizes the input/output pins used by the BT.

Table 21.1 Input/Output Pins

Classification	Name	I/O	Function/Signal
RF-IC interface	RDI_CTRL2	Output	External power amplifier control output for supporting class 1
	RDI_TXTRDATA	I/O	Transmit/receive data I/O for the RF IC
	RDI_RXBDW_OUT	Output	Packet-control output for the RF IC
	RDI_REFCLK_IN	Input	Data I/O clock for the RF IC or main clock input for the BT
	RDI_CTRL3	Output	Strobe output to enable the oscillator in the RF IC
	RDI_CTRL4	Output	Reset-control signal for the RF IC, power supply for the interface, or power-down control output signal
SPI interface with RF IC	RCI_SPI_TXRX	I/O	SPI serial data I/O
	RCI_SPI_CLK	Output	Clock output for the SPI interface with the RF IC
	RCI_SPI_ENB	Output	Enable output for the SPI interface with the RF IC
Voice Codec interface	VCI_SCO_CLK_OUT	Output	Clock output for a Voice Codec IC*[1]
	VCI_SCO_SYNC_OUT	Output	Frame-synchronization signal output for a Voice Codec IC*[1]*[3]
	VCI_SCO_RX	Output	Receive data output to a Voice Codec IC*[1]
	VCI_SCO_TX	Input	Transmit data input from a Voice Codec IC*[1]
	VCI_HWC	Output	Operation mode select output for a Voice Codec IC*[1]
	$\overline{\text{VCI_CODEC_PWRDWN}}$	Output	Power-down control output for a Voice Codec IC*[1]
Clock select	RTCSEL0	Input	Test Pin*[2]
	RREF	Input	Test Pin*[2]
	EXTAL2	Input	For connection to a crystal resonator
	XTAL2	Input	For connection to a crystal resonator

Notes: 1. The Voice-Codec interface pins are available for use as I/O ports (port E) when a Voice Codec IC is not connected. For details, refer to section 23, I/O Ports.
2. The RTCSEL0 pin should be fixed to low, and RREF pin should be pulled-up to Vcc.
3. As the VCI_SCO_SYNC_OUT pin is multiplexed with the $\overline{\text{BREQ}}$ pin, it should be pulled-up for the normal power-up sequence.

21.3 Register Description

The BT registers are accessed via firmware that is provided rather than directly by the user. Therefore, no description is given here. For the functions of the firmware, refer to the separate manual.

All registers in BT module are initialized only by the power-on reset input pin. They are not initialized by the power-on reset and manual reset generated from WDT.

21.4 RF-IC Interface

The Bluetooth interface (BT) supports the direct connection with the Renesas RF ICs, such as HD157100NP and HD157102NP.

21.4.1 Connection with RF ICs

Figure 21.2 shows an example of the connection of this LSI and a Renesas RF IC, such as the HD157100NP and HD157102NP. When a power-on reset signal is applied, the reset signal is sent to the BT power-on reset/clock-resume control circuit and is also sent to the RF IC via the RDI_CTRL4 pin. After the RDI_CTRL4 pin has fulfilled its function as a reset pin, the pin will function as a power-supply pin for the interface of the RF IC. Even after the external power-on reset signal has been negated, the reset state is maintained within this LSI.

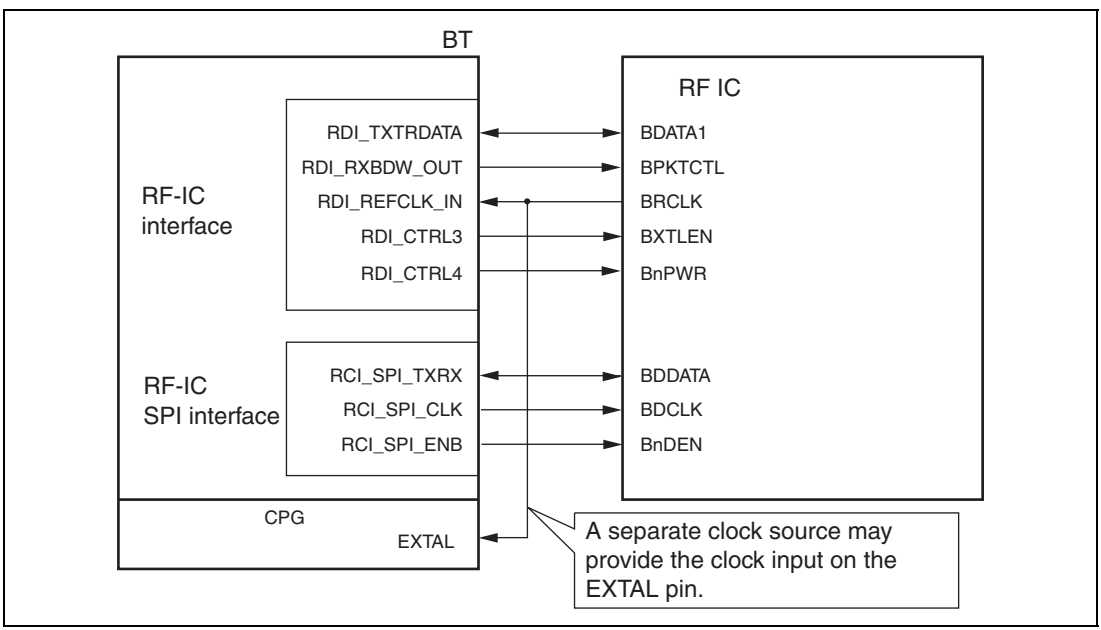


Figure 21.2 Example of Connection to RF IC

After it has been initialized by a reset, the RF IC starts outputting a clock signal (RDI_REFCLK_IN). A counter in the BT power-on-reset/clock-resume control circuit starts counting the cycles of a clock signal from the RF IC. When the value in the counter matches the required value, the RF IC is released from the reset state.

The clock signal (RDI_REFCLK_IN) received from the RF IC may be connected to the clock-pulse generator's EXTAL pin for use as the LSI's main clock signal. This may reduce the number of crystal oscillators required to configure the overall Bluetooth system. In this case, 13 MHz is the only available main-clock frequency for this LSI, since the frequency on the RDI_REFCLK_IN pin will be 13 MHz.

An independent clock can be input without connecting the output of the RDI_REFCLK_IN signal to the EXTAL pin*. In this case, the main clock may run at any frequency within the range guaranteed for use with this LSI. For details on this range, refer to section 28, Electrical Characteristics. For details on the EXTAL pin, refer to section 11, Clock Pulse Generator (CPG).

Note: * When the on-chip USBF bus-powered function is used, make sure to input an independent clock from the RDI_REFCLK_IN input clock and set it to an active state.

21.5 Voice Codec IC Interface

This BT employs a function of the direct transfer of data with the RF IC over an SCO link without intervention of the CPU. As an external Voice Codec IC, this BT supports a direct interface with the STLC7550 (STMicroelectronics) or MC145483 (Motorola) Voice Codec IC.

21.5.1 Voice Codec (STLC7550) Interface

Figure 21.3 shows an example of the connection of this BT and the Voice Codec (STLC7550). The SCLK pin of the STLC7550 is not used.

The VCI_SCO_SYNC_OUT pin is multiplexed with the $\overline{\text{BREQ}}$ pin. This pin should be pulled-up for the normal power-up sequence.

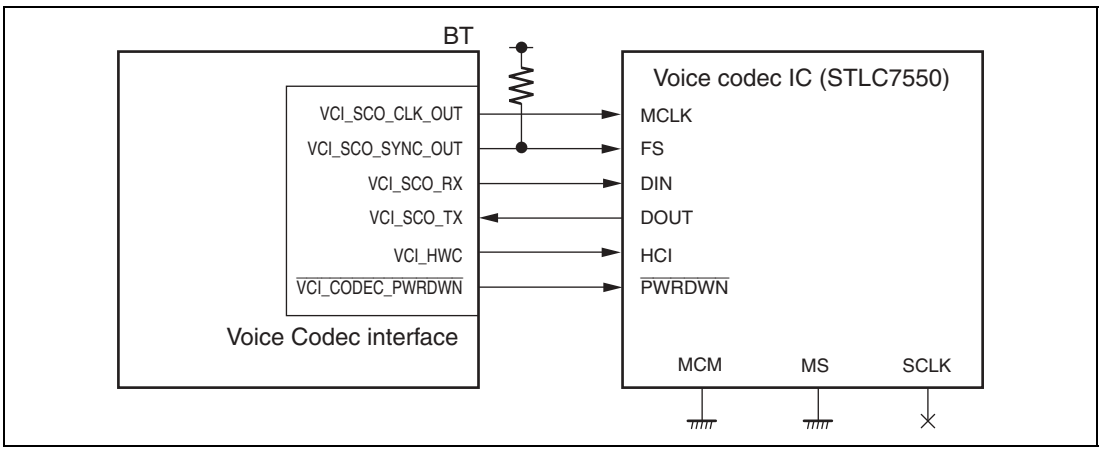


Figure 21.3 Example of Connection to Voice Codec (STLC7550)

21.5.2 Voice Codec (MC145483) Interface

Figure 21.4 (1) shows an example of the connection of this BT and the Voice Codec (MC145483).

The VCI_SCO_SYNC_OUT pin is multiplexed with the $\overline{\text{BREQ}}$ pin. This pin should be pulled-up for the normal power-up sequence.

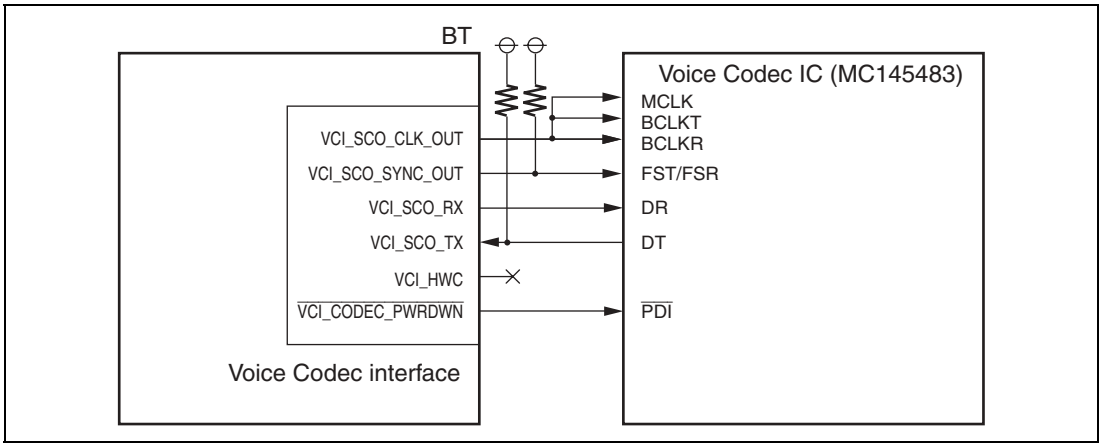


Figure 21.4 (1) Example of Connection to Voice Codec (MC145483)

Figure 21.4 (2) shows the timing chart.

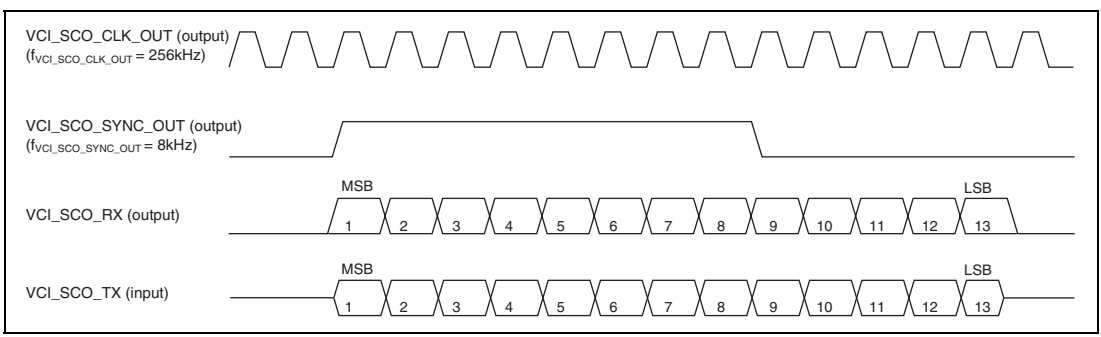


Figure 21.4 (2) Timing chart for connection to Voice Codec (MC145483)

21.6 Low-Frequency Clock Oscillator Interface

This BT has an interface to receive an external clock running at 32 kHz or 32.768 kHz to support Bluetooth's power-down modes (Sniff, Park, and Hold). There are four ways to supply this low-frequency clock (described below). The method is specified by the value of the RTCSEL1 bit in the PSELA. For details on the PSELA, refer to section 24, Pin Function Controller (PFC).

Here are the five possible ways to obtain the low-frequency clock signal;

- connect the 32-kHz crystal resonator to the EXTAL2 and XTAL2 pins to make the on-chip oscillator circuit provide the clock;
- connect the 32.768-kHz crystal resonator to the EXTAL2 and XTAL2 pins to make the on-chip oscillation circuit provide the clock of 32.768kHz, and generate the pseudo 32-kHz clock in the frequency-conversion circuit;
- directly input 32.0-kHz from the EXTAL2 pin; and
- directly input 32.768-kHz from the EXTAL2 pin, and generate the pseudo 32-kHz clock in the frequency-conversion circuit.

Table 21.2 is a list of the settings.

Table 21.2 Settings and Functions

RTCSEL1 Bit	Function
1	The 32-kHz crystal resonator is connected to the EXTAL2 and XTAL2 pins to make the on-chip oscillation circuit provide a signal. If not, 32.0-kHz clock should be supplied at the EXTAL2 pin.
0	The 32.768-kHz crystal resonator is connected to the EXTAL2 and XTAL2 pins to make the on-chip oscillation circuit provide a signal. If not, 32.768-kHz clock should be supplied at the EXTAL2 pin.

21.6.1 Using RTCSEL1 Bit to Select Clock Function

The RTCSEL1 bit in the PSELA selects whether or not the frequency-conversion circuit is activated. The frequency-conversion circuit generates a pseudo-32-kHz clock signal from a 32.768-kHz signal. Activating the frequency-conversion circuit allows the use of the 32.768-kHz crystal resonator, as used with the RTC (Real Time Clock), instead of the connection of a 32.0-kHz crystal resonator to the EXTAL2 and XTAL2 pins.

Clearing the RTCSEL1 bit to 0 activates the frequency-conversion circuit. Setting the RTCSEL1 bit makes the circuit inactive and the 32.768-kHz frequency is passed through the circuit. When the RTCSEL0 pin is set to 1, the RTCSEL1 bit should always be set to 1. The initial value of the RTCSEL1 bit is 0. For details on the PSELA, refer to section 24, Pin Function Controller (PFC).

21.6.2 Frequency-Conversion Circuit Operations

128 cycles of a 32.768-kHz signal are equal in duration to 125 cycles of a 32-kHz signal. This is used to convert the 32.768-kHz clock into a 32-kHz clock in the following way. Firstly, the 128 cycles are divided into three states by the counting of cycles, as shown in figure 21.5. State 0 is made up of the first 43 cycles, state 1 is made up of the next 42 cycles, and state 2 is made up of the remaining 43 cycles. Then, removing a single falling edge from each of these states leaves 125 cycles. This method realizes 125 cycles from a 32.768-kHz clock within the same period as 125 cycles of the 32-kHz clock. The resulting frequency is 32000 cycles per second. A 32-kHz clock signal has thus been generated from the converted clock.

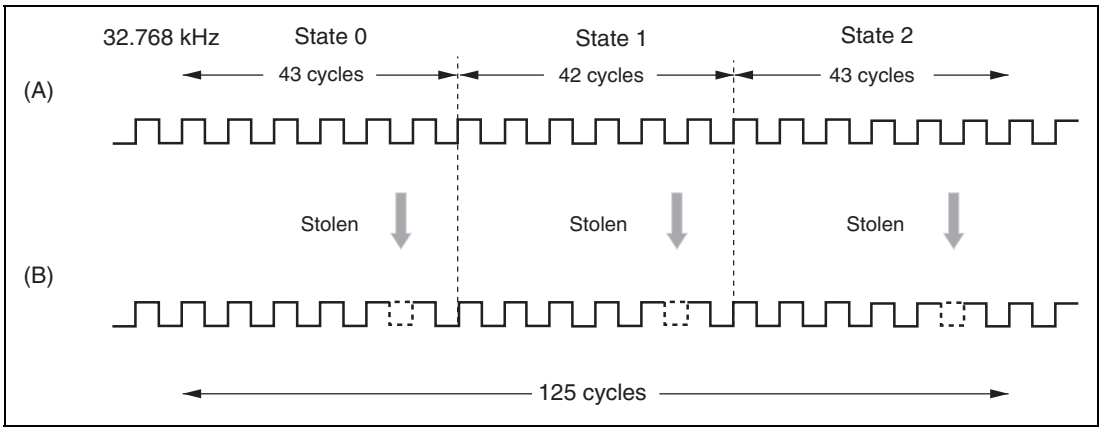


Figure 21.5 Function of Frequency-Conversion Circuit

The cumulative time differences between each cycle of a 32-kHz signal and the corresponding cycle of a pseudo-32-kHz signal are shown in figure 21.6. Since the total time for given numbers of cycles for the 32-kHz clock matches that of the pseudo 32-kHz clock every 125 cycles (see (A) and (B)), the maximum error is approximately 30 µsec.

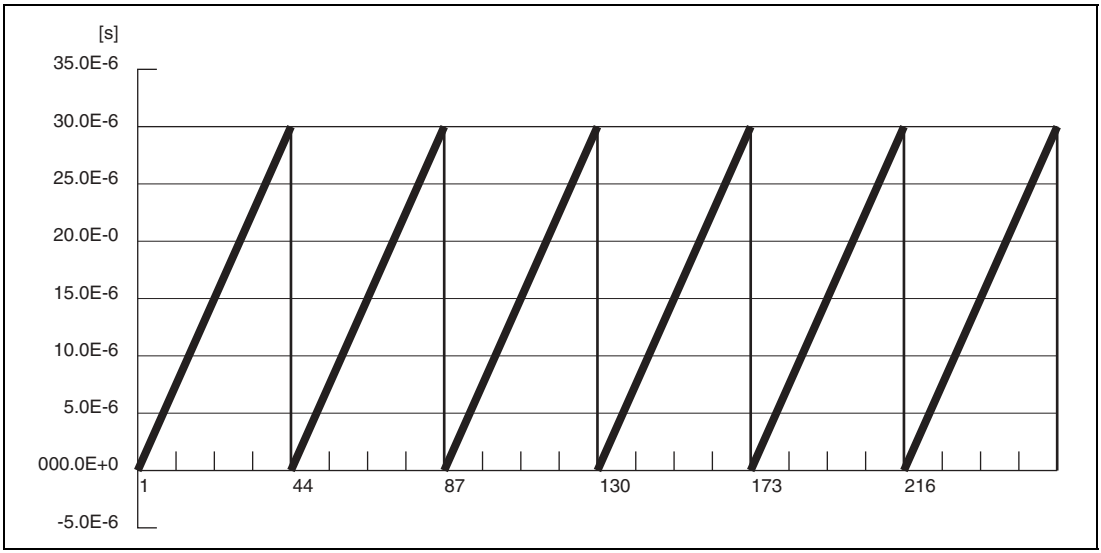
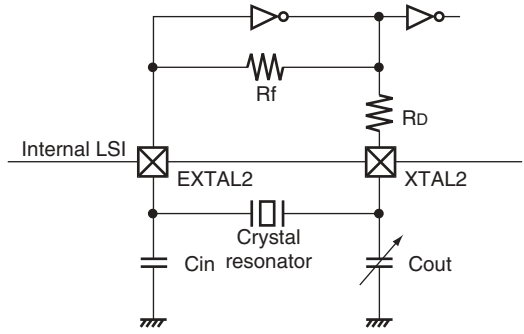


Figure 21.6 Errors in Pseudo-32-kHz Clock Obtained by Conversion from 32.768-kHz Clock

21.6.3 Note on Connecting External Crystal Resonator

Figure 21.7 shows a note when connecting an external crystal resonator to the EXTAL2 and XTAL2 pins.



Notes: 1. A variable capacitor for frequency adjustment may be placed in the Cin or Cout side according to the adjustment range, stability, and so on.
2. The value of the on-chip resistor is Rf = 10 MΩ and RD = 400 kΩ under the typical condition.
3. The values of the Cin and Cout include floating capacitance due to wiring.
4. Since the oscillation stabilization time of the crystal oscillation differs according to the constant of the mounting circuit, floating capacitance, and so on, it should be determined in consultation with the crystal resonator manufacturer.
5. Place the crystal resonator and load capacitance (Cin and Cout) near this LSI (in order to prevent operation failure from occurring after a noise is externally inducted to the EXTAL2 and XTAL2 pins).

Figure 21.7 Note on Using Crystal Resonator

21.7 Power-On Reset/Clock-Resume Control Circuit

The BT of this LSI allows direct connection to the RF IC and thus minimizes the need for externally connected components.

21.7.1 Power-On Reset

The clock, which is input from the HD157100NP RF IC or HD157102NP (Renesas) and is used for the interface with the RF IC, may be used as the main clock of this LSI (see figure 21.2). The RF IC (Renesas) starts outputting a clock signal after a certain period of time following the negation of a power-on reset. However, the input of a clock signal to this LSI is required to settle the oscillation of the on-chip oscillator during reset. For this reason, the LSI's internal reset signal is kept at a low level after negation of the external power-on reset signal. The internal reset signal is only negated after the required period from the application of an external clock signal has elapsed. The timing chart for this is shown in figure 21.8.

There are two types of the internal reset signals for the USB and the other modules. The required period of the internal reset signal for the USB being asserted is different from that of the others. The timing chart is shown in figure 21.8.

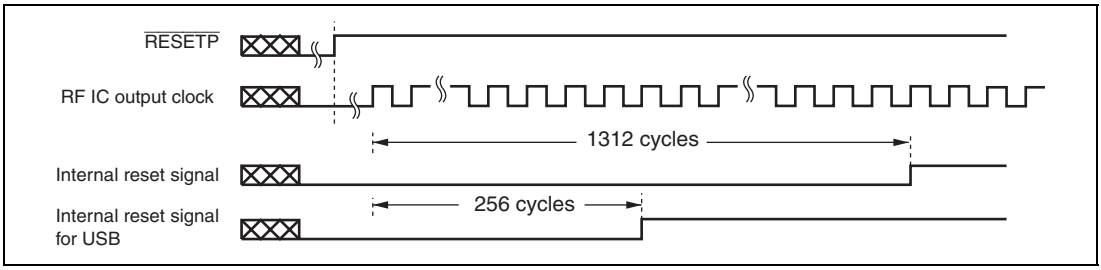


Figure 21.8 Timing Chart of Reset Signal

21.7.2 Clock-Resume Control

In the power-down modes, a command that stops the output of the clock from the RF IC may be sent to the RF IC. When the clock input from the RF IC is used as the main clock of this LSI, this LSI is also stopped and the BT is operated by a separately input clock running at 32-kHz or 32.768-kHz. The LSI is equipped with a clock-resume control function to allow it to leave this mode. The sequence of this clock-resume control is given below.

1. The BT outputs an interrupt signal.
2. The BT issues a wake-up command to the RF-IC.
3. The INTC receives an interrupt.
4. The CPU is activated.

21.8 Bluetooth HCI/TCI Commands and API

To configure the upper protocol stack, this LSI supports host controller interface (HCI) and test control interface (TCI) commands which are defined by version 1.1 of the Bluetooth Specification, and 53 types of unique TCI commands. For details on the HCI and TCI commands, which correspond to version 1.1 of the Bluetooth Specification, refer to version 1.1 of the Bluetooth Specification. When detailed information on unique TCI commands is required, contact our sales department.

The protocol stack firmware lower than the HCI is provided as a library in this LSI, and this enables the Bluetooth interface easily to be added to a developed program. In addition, as the application interface (API) of this library bases on H4 of the Bluetooth Specification, it is easy to be applied to this LSI. As the API function of this LSI, 9 types of functions are prepared. For details, contact our sales department.

22.1 Features

This LSI incorporates a two-channel D/A converter (DAC) with the following features.

- 8-bit resolution
- Two output channels
- Conversion time: Min. 10 µs (when load capacitance is 20 pF)
- Output voltage: 0 V to AVcc (analog power supply)

Figure 22.1 shows the block diagram of the DAC.

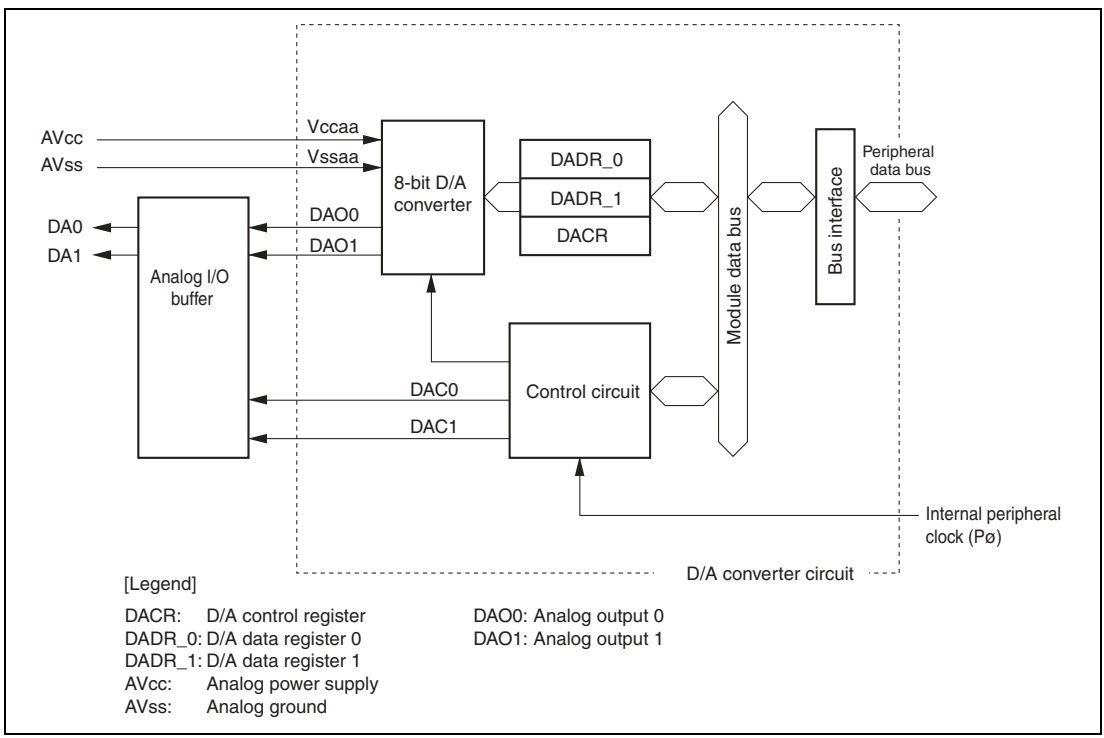


Figure 22.1 Block Diagram of DAC

22.2 Input/Output Pins

Table 22.1 summarizes the input/output pins used by the DAC.

Table 22.1 Pin Configuration

Pin Name	I/O	Function
Avcc	—	Analog block power supply and D/A conversion reference voltage
Avss	—	Analog block ground
DA0	Output	Channel 0 analog output
DA1	Output	Channel 1 analog output

22.3 Register Descriptions

The DAC has the following registers. Refer to section 27, List of Registers, for more details of the addresses of these registers and the state of these registers in each processor mode.

- D/A data register 0 (DADR_0)
- D/A data register 1 (DADR_1)
- D/A control register (DACR)

22.3.1 D/A Data Registers 0 and 1 (DADR_0, DADR_1)

DADR_0 and DADR_1 are 8-bit readable/writable registers that store data for D/A conversion. When the D/A output enable bits (DAOE1, DAOE0) of the DA control register (DACR) are set to 1, the contents of the D/A data register (DADR_0, DADR_1) are converted and output to analog output pins (DA0, DA1). The D/A data register (DADR_0, DADR_1) is initialized to H'00 at a reset. Note that DADR_0 and DADR_1 are not initialized in software standby or module standby.

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	—	H'00	R/W	8-bit registers that store data for D/A conversion.

22.3.2 D/A Control Register (DACR)

DACR is an 8-bit readable/writable register that controls the DAC operation. DACR is initialized to H'3F at a reset. Note that DACR is not initialized in software standby or module standby.

Bit	Bit Name	Initial Value	R/W	Description
7	DAOE1	0	R/W	Controls D/A conversion for channel 1 and analog output. 0: D/A conversion for channel 1 and analog output (DA1) are disabled 1: D/A conversion for channel 1 and analog output (DA1) are enabled
6	DAOE0	0	R/W	Controls D/A conversion for channel 0 and analog output. 0: D/A conversion for channel 0 and analog output (DA0) are disabled 1: D/A conversion for channel 0 and analog output (DA0) are enabled
5 to 0	—	All 1	R	Reserved These bits are always read as 1. The write value should always be 1. If 0 is written to these bits, correct operation cannot be guaranteed.

22.4 Operation

The DAC incorporates two D/A channels that can operate individually.

The DAC executes D/A conversion while analog output is enabled by the D/A control register (DACR). If the D/A data registers (DADR_0 and DADR_1) are modified, the D/A converter immediately initiates the new data conversion. When the DAOE1 and DAOE0 bits in the DACR register are set to 1, D/A conversion results are output.

An example of D/A conversion for channel 0 is shown below. The operation timing is shown in figure 22.2.

1. Write conversion data to DADR_0.
2. When the DAOE0 bit in DACR is set to 1, D/A conversion starts. The results are output after the conversion has ended. The output value will be (DADR_0 contents/256) × AVcc.

   The conversion results are output continuously until DADR_0 is modified or the DAOE0 bit is cleared to 0.

3. When D/A data register 0 (DADR_0) is modified, the conversion starts again.
   The results are output after the conversion has ended.
4. When the DAOE0 in the DACR bit is cleared to 0, analog output is disabled (high-impedance state).

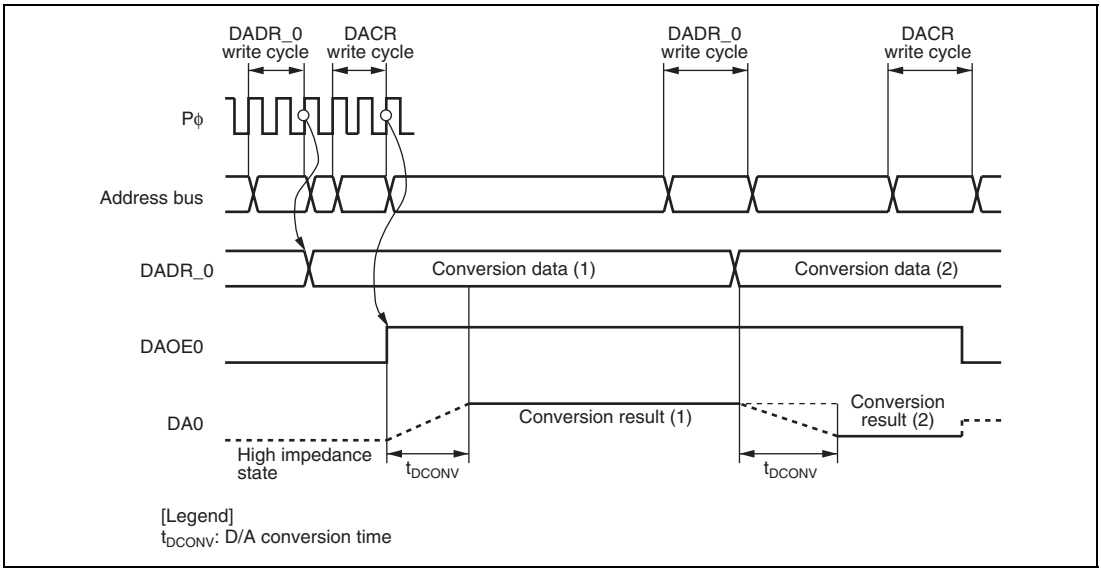


Figure 22.2 D/A Converter Operation Example

Section 23 I/O Ports

This LSI has six I/O ports (ports A, B, D, E, G and H). All port pins are multiplexed with other pin functions (the pin function controller (PFC) handles the selection of pin functions and pull-up MOS control). Each port has a data register which stores data for the pins.

23.1 Port A

Port A is a 7-bit input/output port with the pin configuration shown in figure 23.1. Each pin has an input pull-up MOS, which is controlled by the port A control register (PACR) in the PFC.

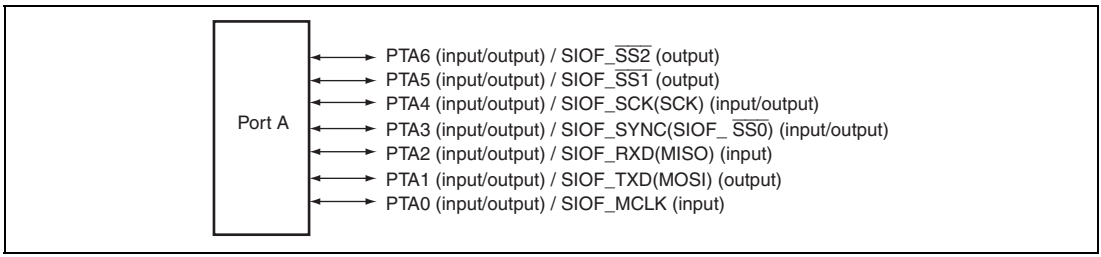


Figure 23.1 Port A

23.1.1 Register Description

Port A has the following register. Refer to section 27, List of Registers, for the address of this register and state of this register in each processing mode.

- Port A data register (PADR)

23.1.2 Port A Data Register (PADR)

PADR is a 7-bit readable/writable register that stores data for pins PTA6 to PTA0. Bits PA6DT to PA0DT correspond to pins PTA6 to PTA0. When the pin function is general output port, if the port is read, the value of the corresponding PADR bit is returned directly. When the function is general input port, if the port is read, the corresponding pin level is read.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
6	PA6DT	0	R/W	Table 23.1 shows the function of PADR.
5	PA5DT	0	R/W	
4	PA4DT	0	R/W	
3	PA3DT	0	R/W	
2	PA2DT	0	R/W	
1	PA1DT	0	R/W	
0	PA0DT	0	R/W	

Table 23.1 Port A Data Register (PADR) Read/Write Operations

PACR State*1				
PAnMD1*2	PAnMD0*2	Pin State	Read	Write
0	0	Other function	PADR value	Value is written to PADR, but does not affect pin state.
	1	Output	PADR value	Write value is output from pin.
1	0	Input (Pull-up MOS on)	Pin state	Value is written to PADR, but does not affect pin state.
	1	Input (Pull-up MOS off)	Pin state	Value is written to PADR, but does not affect pin state.

Notes: 1. For details on PACR, refer to section 24, Pin Function Controller (PFC).
2. n = 0 to 6

23.2 Port B

Port B is an 8-bit input/output port with the pin configuration shown in figure 23.2. Each pin has an input pull-up MOS, which is controlled by the port B control register (PBCR) in the PFC.

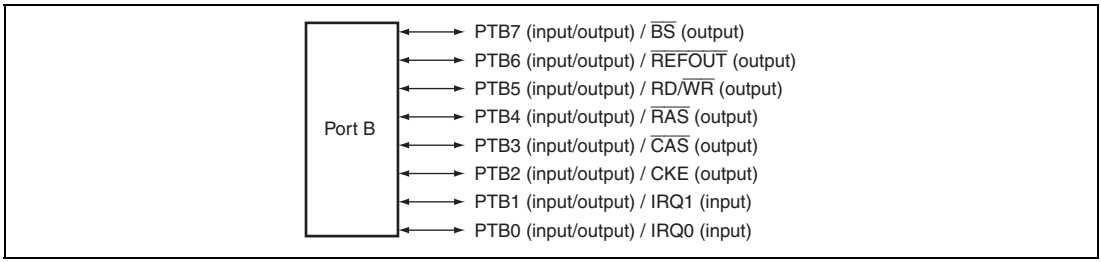


Figure 23.2 Port B

23.2.1 Register Description

Port B has the following register. Refer to section 27, List of Registers, for the address of this register and state of this register in each processing mode.

- Port B data register (PBDR)

23.2.2 Port B Data Register (PBDR)

PBDR is an 8-bit readable/writable register that stores data for pin PTB7 to PTB0. Bits PB7DT to PB0DT correspond to pins PTB7 to PTB0. When the pin function is general output port, if the port is read, the value of the corresponding PBDR bit is returned directly. When the function is general input port, if the port is read, the corresponding pin level is read.

Bit	Bit Name	Initial Value	R/W	Description
7	PB7DT	0	R/W	Table 23.2 shows the function of PBDR.
6	PB6DT	0	R/W	
5	PB5DT	0	R/W	
4	PB4DT	0	R/W	
3	PB3DT	0	R/W	
2	PB2DT	0	R/W	
1	PB1DT	0	R/W	
0	PB0DT	0	R/W	

Table 23.2 Port B Data Register (PBDR) Read/Write Operations

PBCR State*[1]				
PBnMD1*[2]	**PBnMD0*[2]**	**Pin State**	**Read**	**Write**
0	0	Other function	PBDR value	Value is written to PBDR, but does not affect pin state.
	1	Output	PBDR value	Write value is output from pin.
1	0	Input (Pull-up MOS on)	Pin state	Value is written to PBDR, but does not affect pin state.
	1	Input (Pull-up MOS off)	Pin state	Value is written to PBDR, but does not affect pin state.

Notes: 1. For details on PBCR, refer to section 24, Pin Function Controller (PFC).
2. n = 0 to 7

23.3 Port D

Port D is a 6-bit input/output port with the pin configuration shown in figure 23.3. Each pin has an input pull-up MOS, which is controlled by the port D control register (PDCR) in the PFC.

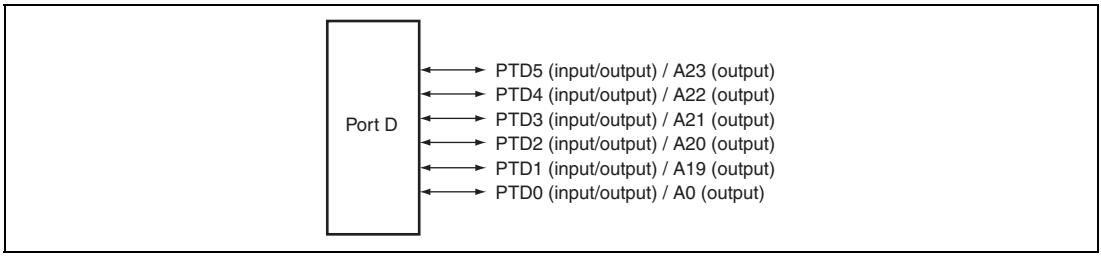


Figure 23.3 Port D

23.3.1 Register Description

Port D has the following register. Refer to section 27, List of Registers, for the address of this register and state of this register in each processing mode.

- Port D data register (PDDR)

23.3.2 Port D Data Register (PDDR)

PDDR is a 6-bit readable/writable register that stores data for pins PTD5 to PTD0. Bits PD5DT to PD0DT correspond to pins PTD5 to PTD0. When the pin function is general output port, if the port is read, the value of the corresponding PDDR bit is returned directly. When the function is general input port, if the port is read, the corresponding pin level is read.

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
5	PD5DT	0	R/W	Table 23.3 shows the function of PDDR.
4	PD4DT	0	R/W	
3	PD3DT	0	R/W	
2	PD2DT	0	R/W	
1	PD1DT	0	R/W	
0	PD0DT	0	R/W	

Table 23.3 Port D Data Register (PDDR) Read/Write Operations

PDCR State*1				
PDnMD1*2	**PDnMD0*2**	**Pin State**	**Read**	**Write**
0	0	Other function	PDDR value	Value is written to PDDR, but does not affect pin state.
	1	Output	PDDR value	Write value is output from pin.
1	0	Input (Pull-up MOS on)	Pin state	Value is written to PDDR, but does not affect pin state.
	1	Input (Pull-up MOS off)	Pin state	Value is written to PDDR, but does not affect pin state.

Notes: 1. For details on PDCR, refer to section 24, Pin Function Controller (PFC).
2. n = 0 to 5

23.4 Port E

Port E is a 7-bit input/output port with the pin configuration shown in figure 23.4. Each pin has an input pull-up MOS, which is controlled by the port E control register (PECR) in the PFC.

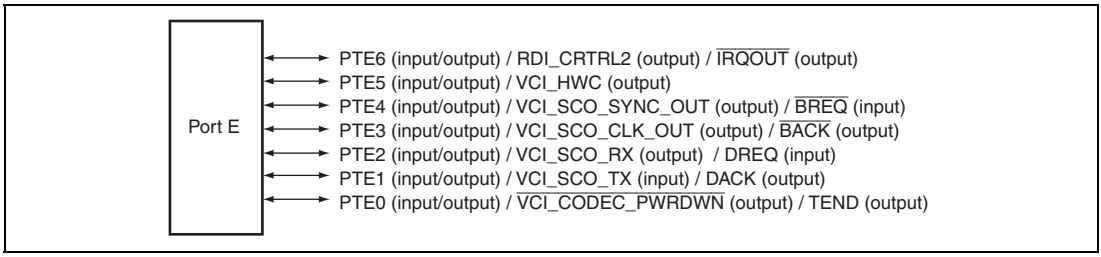


Figure 23.4 Port E

23.4.1 Register Description

Port E has the following register. Refer to section 27, List of Registers, for the address of this register and state of this register in each processing mode.

- Port E data register (PEDR)

23.4.2 Port E Data Register (PEDR)

PEDR is a 7-bit readable/writable register that stores data for pins PTE6 to PTE0. Bits PE6DT to PE0DT correspond to pins PTE6 to PTE0. When the pin function is general output port, if the port is read, the value of the corresponding PEDR bit is returned directly. When the function is general input port, if the port is read, the corresponding pin level is read.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R	Reserved This bit is always read as 0. The write value should always be 0. If 1 is written to this bit, correct operation cannot be guaranteed.
6	PE6DT	0	R/W	Table 23.4 shows the function of PEDR.
5	PE5DT	0	R/W	
4	PE4DT	0	R/W	
3	PE3DT	0	R/W	
2	PE2DT	0	R/W	
1	PE1DT	0	R/W	
0	PE0DT	0	R/W	

Table 23.4 Port E Data Register (PEDR) Read/Write Operations

PECR State*1				
PEnMD1*2	**PEnMD0***2	**Pin State**	**Read**	**Write**
0	0	Other function	PEDR value	Value is written to PEDR, but does not affect pin state.
	1	Output	PEDR value	Write value is output from pin.
1	0	Input (Pull-up MOS on)	Pin state	Value is written to PEDR, but does not affect pin state.
	1	Input (Pull-up MOS off)	Pin state	Value is written to PEDR, but does not affect pin state.

Notes: 1. For details on PECR, refer to section 24, Pin Function Controller (PFC).
2. n = 0 to 6

23.5 Port G

Port G is a 5-bit input/output port with the pin configuration shown in figure 23.5. Each pin has an input pull-up MOS, which is controlled by the port G control register (PGCR) in the PFC.

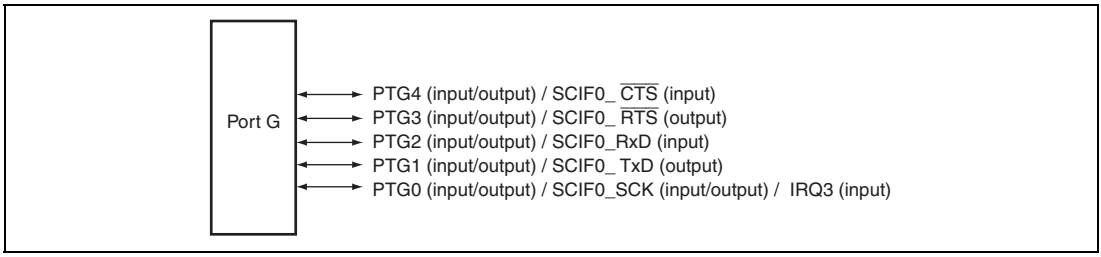


Figure 23.5 Port G

23.5.1 Register Description

Port G has the following register. Refer to section 27, List of Registers, for the address of this register and state of this register in each processing mode.

- Port G data register (PGDR)

23.5.2 Port G Data Register (PGDR)

PGDR is a 5-bit readable/writable register that stores data for pins PTG4 to PTG0. Bits PG4DT to PG0DT correspond to pins PTG4 to PTG0. When the pin function is general output port, if the port is read, the value of the corresponding PGDR bit is returned directly. When the function is general input port, if the port is read the corresponding pin level is read.

If the TE or RE bit is set to 1 in the serial control register _0 (SCSCR_0) of the serial communication interface with FIFO (SCIF), the SCIF0_TxD or SCIF0_RxD is selected regardless of the PGCR register setting.

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
4	PG4DT	0	R/W	Table 23.5 shows the function of PGDR.
3	PG3DT	0	R/W	
2	PG2DT	0	R/W	
1	PG1DT	0	R/W	
0	PG0DT	0	R/W	

Table 23.5 Port G Data Register (PGDR) Read/Write Operations

PGCR State*1				
PGnMD1*2	PGnMD0*2	Pin State	Read	Write
0	0	Other function	PGDR value	Value is written to PGDR, but does not affect pin state.
	1	Output	PGDR value	Write value is output from pin.
1	0	Input (Pull-up MOS on)*3	Pin state	Value is written to PGDR, but does not affect pin state.
	1	Input (Pull-up MOS off)*3	Pin state	Value is written to PGDR, but does not affect pin state.

Notes: 1. For details on PGCR, refer to section 24, Pin Function Controller (PFC).
2. n = 0 to 4
3. When n = 1, pin state is a high impedance (output).

23.6 Port H

Port H is a 5-bit input/output port with the pin configuration shown in figure 23.6. Each pin has an input pull-up MOS, which is controlled by the port H control register (PHCR) in the PFC.

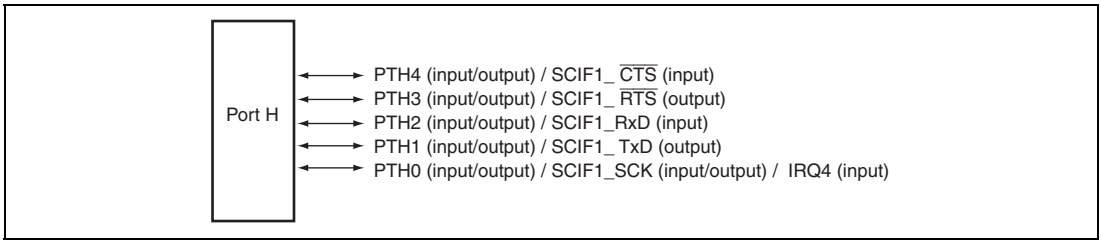


Figure 23.6 Port H

23.6.1 Register Description

Port H has the following register. Refer to section 27, List of Registers, for the address of this register and state of this register in each processing mode.

- Port H data register (PHDR)

23.6.2 Port H Data Register (PHDR)

PHDR is a 5-bit readable/writable register that stores data for pins PTH4 to PTH0. Bits PH4DT to PH0DT correspond to pins PTH4 to PTH0. When the function is general output port, if the port is read, the value of the corresponding PHDR bit is returned directly. When the function is general input port, if the port is read, the corresponding pin level is read.

If the TE or RE bit is set to 1 in the serial control register _1 (SCSCR_1) of the serial communication interface with FIFO (SCIF), the SCIF1_TxD or SCIF1_RxD is selected regardless of the PHCR register setting.

Bit	Bit Name	Initial Value	R/W	Description
7 to 5	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
4	PH4DT	0	R/W	Table 23.6 shows the function of PHDR.
3	PH3DT	0	R/W	
2	PH2DT	0	R/W	
1	PH1DT	0	R/W	
0	PH0DT	0	R/W	

Table 23.6 Port H Data Register (PHDR) Read/Write Operations

PHCR State*1				
PHnMD1*2	**PHnMD0***2	**Pin State**	**Read**	**Write**
0	0	Other function	PHDR value	Value is written to PHDR, but does not affect pin state.
	1	Output	PHDR value	Write value is output from pin.
1	0	Input (Pull-up MOS on)*3	Pin state	Value is written to PHDR, but does not affect pin state.
	1	Input (Pull-up MOS off)*3	Pin state	Value is written to PHDR, but does not affect pin state.

Notes: 1. For details on PHCR, refer to section 24, Pin Function Controller (PFC).
2. n = 0 to 4
3. When n = 1, pin state is a high impedance (output).

Section 24 Pin Function Controller (PFC)

24.1 Overview

Some pins in this LSI are multiplexed with I/O port functions (for details on this I/O port refer to section 23, I/O Ports). The pin function controller (PFC) consists of registers to select the pin functions and I/O directions of multiplex pins. Pin functions and I/O directions can be individually selected for every pin regardless of the LSI operating mode. Table 24.1 lists the multiplex pins of this LSI. In the table, pin functions in the shaded column can be used immediately after resetting.

Table 24.1 Multiplex Pins

Port	Function 1 (Related Module)	Function 2 (Related Module)	Function 3 (Related Module)
A	PTA6 input/output (port)	SIOF_$\overline{SS2}$ output (SIOF)	—
A	PTA5 input/output (port)	SIOF_$\overline{SS1}$ output (SIOF)	—
A	PTA4 input/output (port)	SIOF_SCK (SCK) input/output (SIOF)	—
A	PTA3 input/output (port)	SIOF_SYNC (SIOF_$\overline{SS0}$) input/output (SIOF)	—
A	PTA2 input/output (port)	SIOF_RXD (MISO) input (SIOF)	—
A	PTA1 input/output (port)	SIOF_TXD (MOSI) output (SIOF)	—
A	PTA0 input/output (port)	SIOF_MCLK input (SIOF)	—
B	PTB7 input/output (port)	$\overline{BS}$ output (BSC)	—
B	PTB6 input/output (port)	$\overline{REFOUT}$ output (BSC)	—
B	PTB5 input/output (port)	RD/$\overline{WR}$ output (BSC)	—
B	PTB4 input/output (port)	$\overline{RAS}$ output (BSC)	—
B	PTB3 input/output (port)	$\overline{CAS}$ output (BSC)	—
B	PTB2 input/output (port)	CKE output (BSC)	—
B	PTB1 input/output (port)	IRQ2 input (INTC)	—
B	PTB0 input/output (port)	IRQ0 input (INTC)	—
D	PTD5 input/output (port)	A23 output (BSC)	—
D	PTD4 input/output (port)	A22 output (BSC)	—
D	PTD3 input/output (port)	A21 output (BSC)	—
D	PTD2 input/output (port)	A20 output (BSC)	—
D	PTD1 input/output (port)	A19 output (BSC)	—
D	PTD0 input/output (port)	A0 output (BSC)	—

Port	Function 1 (Related Module)	Function 2 (Related Module)	Function 3 (Related Module)
E	PTE6 input/output (port)	RDI_CTRL2 output (BT)	$\overline{\text{IRQOUT}}$ output (INTC)
E	PTE5 input/output (port)	VCI_HWC output (BT)	—
E	PTE4 input/output (port)	VCI_SCO_SYNC_OUT output (BT)	$\overline{\text{BREQ}}$ input (BSC)
E	PTE3 input/output (port)	VCI_SCO_CLK_OUT output (BT)	$\overline{\text{BACK}}$ output (BSC)
E	PTE2 input/output (port)	VCI_SCO_RX output (BT)	DREQ input (DMAC)
E	PTE1 input/output (port)	VCI_SCO_TX input (BT)	DACK output (DMAC)
E	PTE0 input/output (port)	$\overline{\text{VCI_CODEC_PWRDWN}}$ output (BT)	TEND output (DMAC)
G	PTG4 input/output (port)	SCIF0_$\overline{\text{CTS}}$ input (SCIF0)	—
G	PTG3 input/output (port)	SCIF0_$\overline{\text{RTS}}$ output (SCIF0)	—
G	PTG2 input/output (port)	SCIF0_RxD input (SCIF0)	—
G	PTG1 input/output (port)	SCIF0_TxD output (SCIF0)	—
G	PTG0 input/output (port)	SCIF0_SCK input/output (SCIF0)	IRQ3 input (INTC)
H	PTH4 input/output (port)	SCIF1_$\overline{\text{CTS}}$ input (SCIF1)	—
H	PTH3 input/output (port)	SCIF1_$\overline{\text{RTS}}$ output (SCIF1)	—
H	PTH2 input/output (port)	SCIF1_RxD input (SCIF1)	—
H	PTH1 input/output (port)	SCIF1_TxD output (SCIF1)	—
H	PTH0 input/output (port)	SCIF1_SCK input/output (SCIF1)	IRQ4 input (INTC)

24.2 Register Descriptions

PFC has the following registers. For details on register addresses and register states in each operation mode, refer to section 27, List of Registers.

- Port A control register (PACR)
- Port B control register (PBCR)
- Port D control register (PDCR)
- Port E control register (PECR)
- Port G control register (PGCR)
- Port H control register (PHCR)
- Pin select register A (PSELA)
- I/O buffer Hi-Z control register A (HIZCRA)
- Noise canceller control register (NCCR)

Note: The PFC registers are not initialized by a manual reset and hold the contents.

24.2.1 Port A Control Register (PACR)

PACR is a 16-bit readable/writable register that selects the pin function and input pull-up MOS control.

Bit	Bit Name	Initial Value*	R/W	Description
15, 14	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
13	PA6MD1	1	R/W	PTA6 Mode
12	PA6MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)
11	PA5MD1	1	R/W	PTA5 Mode
10	PA0MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)

Bit	Bit Name	Initial Value*	R/W	Description
9	PA4MD1	1	R/W	PTA4 Mode
8	PA4MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
7	PA3MD1	1	R/W	PTA3 Mode
6	PA3MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
5	PA2MD1	1	R/W	PTA2 Mode
4	PA2MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
3	PA1MD1	1	R/W	PTA1 Mode
2	PA1MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
1	PA0MD1	1	R/W	PTA0 Mode
0	PA0MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)

Note: * This register is not initialized by a manual reset and holds the contents.

24.2.2 Port B Control Register (PBCR)

PBCR is a 16-bit readable/writable register that selects the pin function and input pull-up MOS control.

Bit	Bit Name	Initial Value*	R/W	Description
15	PB7MD1	0	R/W	PTB7 Mode
14	PB7MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
13	PB6MD1	0	R/W	PTB6 Mode
12	PB6MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
11	PB5MD1	0	R/W	PTB5 Mode
10	PB5MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
9	PB4MD1	0	R/W	PTB4 Mode
8	PB4MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
7	PB3MD1	0	R/W	PTB3 Mode
6	PB3MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
5	PB2MD1	0	R/W	PTB2 Mode
4	PB2MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)

Bit	Bit Name	Initial Value*	R/W	Description
3	PB1MD1	1	R/W	PTB1 Mode
2	PB1MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
1	PB0MD1	1	R/W	PTB0 Mode
0	PB0MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)

Note: * This register is not initialized by a manual reset and holds the contents.

24.2.3 Port D Control Register (PDCR)

PDCR is a 16-bit readable/writable register that selects the pin function and input pull-up MOS control.

Bit	Bit Name	Initial Value*	R/W	Description
15 to 12	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
11	PD5MD1	0	R/W	PTD5 Mode
10	PD5MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
9	PD4MD1	0	R/W	PTD4 Mode
8	PD4MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)

Bit	Bit Name	Initial Value*	R/W	Description
7	PD3MD1	0	R/W	PTD3 Mode
6	PD3MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
5	PD2MD1	0	R/W	PTD2 Mode
4	PD2MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
3	PD1MD1	0	R/W	PTD1 Mode
2	PD1MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
1	PD0MD1	0	R/W	PTD0 Mode
0	PD0MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)

Note: * This register is not initialized by a manual reset and holds the contents.

24.2.4 Port E Control Register (PECR)

PECR is a 16-bit readable/writable register that selects the pin function and input pull-up MOS control.

Bit	Bit Name	Initial Value*	R/W	Description
15, 14	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
13	PE6MD1	1	R/W	PTE6 Mode
12	PE6MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)
11	PE5MD1	1	R/W	PTE5 Mode
10	PE5MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)
9	PE4MD1	0	R/W	PTE4 Mode
8	PE4MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)
7	PE3MD1	0	R/W	PTE3 Mode
6	PE3MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)
5	PE2MD1	1	R/W	PTE2 Mode
4	PE2MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)

Bit	Bit Name	Initial Value*	R/W	Description
3	PE1MD1	1	R/W	PTE1 Mode
2	PE1MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
1	PE0MD1	1	R/W	PTE0 Mode
0	PE0MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)

Note: * This register is not initialized by a manual reset and holds the contents.

24.2.5 Port G Control Register (PGCR)

PGCR is a 16-bit readable/writable register that selects the pin function and input pull-up MOS control.

Bit	Bit Name	Initial Value*	R/W	Description
15 to 10	—	All 0	R	Reserved
				These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
9	PG4MD1	1	R/W	PTG4 Mode
8	PG4MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)
7	PG3MD1	1	R/W	PTG3 Mode
6	PG3MD0	0	R/W	00: Other functions (see table 24.1)
				01: Port output
				10: Port input (pull-up MOS: On)
				11: Port input (pull-up MOS: Off)

Bit	Bit Name	Initial Value*	R/W	Description
5	PG2MD1	1	R/W	PTG2 Mode
4	PG2MD0	0	R/W	When the RE bit in the SCSCR_0 register is 0: 00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off) When the RE bit in the SCSCR_0 register is 1: Regardless of the PG2MD1 or PG2MD0 bit setting, other functions (SCIF0_RxD) is selected.
3	PG1MD1	1	R/W	PTG1 Mode
2	PG1MD0	0	R/W	When the TE bit in the SCSCR_0 register is 0: 00: Other functions (see table 24.1) 01: Port output 10: Output high-impedance state 11: Output high-impedance state When the TE bit in the SCSCR_0 register is 1: Regardless of the PG1MD1 or PG1MD0 bit setting, other function (SCIF0_TxD) is selected.
1	PG0MD1	1	R/W	PTG0 Mode
0	PG0MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)

Note: * This register is not initialized by a manual reset and holds the contents.

24.2.6 Port H Control Register (PHCR)

PHCR is a 16-bit readable/writable register that selects the pin function and input pull-up MOS control.

Bit	Bit Name	Initial Value*	R/W	Description
15 to10	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
9	PH4MD1	1	R/W	PTH4 Mode
8	PH4MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)
7	PH3MD1	1	R/W	PTH3 Mode
6	PH3MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)
5	PH2MD1	1	R/W	PTH2 Mode
4	PH2MD0	0	R/W	When the RE bit in the SCSCR_1 register is 0: 00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off) When the RE bit in the SCSCR_1 is 1: Regardless of the PH2MD1 or PH2MD0 bit setting, other functions (SCIF1_RxD) is selected.

Bit	Bit Name	Initial Value*	R/W	Description
3	PH1MD1	1	R/W	PTH1 Mode
2	PH1MD0	0	R/W	When the TE bit in the SCSCR_1 register is 0: 00: Other functions (see table 24.1) 01: Port output 10: Output high-impedance state 11: Output high-impedance state When the TE bit in the SCSCR_1 register is 1: Regardless of the PH1MD1 or PH1MD0 bit setting, other function (SCIF1_TxD) is selected.
1	PH0MD1	1	R/W	PTH0 Mode
0	PH0MD0	0	R/W	00: Other functions (see table 24.1) 01: Port output 10: Port input (pull-up MOS: On) 11: Port input (pull-up MOS: Off)

Note: * This register is not initialized by a manual reset and holds the contents.

24.2.7 Pin Select Register A (PSELA)

PSELA is a 16-bit readable/writable register that has two purposes.

Bit 0 (RTCSEL1) selects a frequency for the low-frequency clock source that is used in the Bluetooth interface (BT). When a 32.000kHz clock is selected, the clock is input to the BT. When 32.768-kHz clock is selected, the clock is converted into a pseudo 32.000kHz clock by the frequency conversion circuit. For details, refer to section 21, Bluetooth Interface (BT).

Bits 1 to 8 select the function for a pin multiplexed with two or more other functions. When one of the other functions are used, the corresponding bit in the PSELA should be specified before the other function is specified by the port control register.

Setting example: To use $\overline{\text{IRQOUT}}$ function in the PTE6/RDI_CTRL2/$\overline{\text{IRQOUT}}$ pin

1. Write 1 to the PSA8 bit in the PSELA register.
2. Set the PE6MD1 and PE6MD0 bits in the port E control register (PECR) to (0.0) (other functions).

Bit	Bit Name	Initial Value*	R/W	Description
15 to 9	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
8	PSA8	0	R/W	RDI_CTRL2/$\overline{\text{IRQOUT}}$ Select 0 : Selects RDI_CTRL2 1 : Selects $\overline{\text{IRQOUT}}$
7	PSA7	1	R/W	VCI_SCO_SYNC_OUT/$\overline{\text{BREQ}}$ Select 0 : Selects VCI_SCO_SYNC_OUT 1 : Selects $\overline{\text{BREQ}}$
6	PSA6	1	R/W	VCI_SCO_CLK_OUT/$\overline{\text{BACK}}$ Select 0 : Selects VCI_SCO_CLK_OUT 1 : Selects $\overline{\text{BACK}}$
5	PSA5	0	R/W	VCI_SCO_RX/DREQ Select 0: Selects VCI_SCO_RX 1: Selects DREQ
4	PSA4	0	R/W	VCI_SCO_TX/DACK Select 0: Selects VCI_SCO_TX 1: Selects DACK
3	PSA3	0	R/W	$\overline{\text{VCI_CODEC_PWRDWN}}$/TEND Select 0 : Selects $\overline{\text{VCI_CODEC_PWRDWN}}$ 1 : Selects TEND
2	PSA2	0	R/W	SCIF0_SCK/IRQ3 Select 0 : Selects SCIF0_SCK 1 : Selects IRQ3
1	PSA1	0	R/W	SCIF1_SCK/IRQ4 Select 0 : Selects SCIF1_SCK 1 : Selects IRQ4
0	RTCSEL 1	0	R/W	Low-frequency clock frequency Select 0 : Selects 32.768-kHz 1 : Selects 32.000-kHz

Note: * This register is not initialized by a manual reset and holds the contents.

24.2.8 I/O Buffer Hi-Z Control Register A (HIZCRA)

HIZCRA is a 16-bit readable/writable register that controls Hi-Z and pull-up for pins for every function unit*.

Note: * Even if the output Hi-Z state is specified in the HIZCRA, the pull-up MOS remains turned on when port input (pull-up MOS: On) is selected in the PnCR.

Bit	Bit Name	Initial Value*	R/W	Description
15 to 9	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. When writing 1, the operation of this LSI is not guaranteed.
8	HIZA8	0	R/W	Controls high impedance for pins multiplexed with SIOF related signals (SIOF_RXD(MISO)/SIOF_TXD(MOSI)/SIOF_SYNC(SIOF_$\overline{SS0}$)/SIOF_$\overline{SS1}$/SIOF_$\overline{SS2}$). 0 : I/O buffer enters normal operation mode 1 : Input buffer is an OFF state, and I/O buffer enters output Hi-Z state.
7	HIZA7	0	R/W	Controls high impedance for pins multiplexed with SIOF related signals (SIOF_MCLK/SIOF_SCK(SCK)) 0 : I/O buffer enters normal operation mode 1 : Input buffer is an OFF state, and I/O buffer enters output Hi-Z state.
6	HIZA6	0	R/W	Controls high impedance for pins multiplexed with SCIF0 related signals(SCIF0_RxD/SCIF0_TxD/SCIF0_$\overline{RTS}$/SCIF0_$\overline{CTS}$) 0 : I/O buffer enters normal operation mode 1 : Input buffer is an OFF state, and I/O buffer enters output Hi-Z state.
5	HIZA5	0	R/W	Controls high impedance for pins multiplexed with SCIF0 related signals (SCIF0_SCK) 0 : I/O buffer enters normal operation mode 1 : Input buffer is an OFF state, and I/O buffer enters output Hi-Z state.

Bit	Bit Name	Initial Value*	R/W	Description
4	HIZA4	0	R/W	Controls high impedance for pins multiplexed with SCIF1 related signals (SCIF1_RxD/SCIF1_TxD/SCIF1_$\overline{\text{RTS}}$/SCIF1_$\overline{\text{CTS}}$) 0 : I/O buffer enters normal operation mode 1 : Input buffer is an OFF state, and I/O buffer enters output Hi-Z state.
3	HIZA3	0	R/W	Controls high impedance for pins multiplexed with SCIF1 related signals (SCIF1_SCK) 0 : I/O buffer enters normal operation mode 1 : Input buffer is an OFF state, and I/O buffer enters output Hi-Z state.
2	HIZA2	0	R/W	Controls high impedance for pins multiplexed with RD/$\overline{\text{WR}}$/$\overline{\text{RAS}}$/$\overline{\text{CAS}}$/CKE 0 : I/O buffer enters normal operation mode 1 : Input buffer is an OFF state, and I/O buffer enters output Hi-Z state.
1	HIZA1	0	R/W	Controls high impedance for IRQ2 pins 0 : I/O buffer enters normal operation mode 1 : Input buffer is an OFF state, and I/O buffer enters output Hi-Z state.*[1]
0	HIZA0	0	R/W	Controls high impedance for IRQ0 pins 0 : I/O buffer enters normal operation mode. 1 : Input buffer is an OFF state, and I/O buffer enters output Hi-Z state.*[2]

Notes: 1. This register is not initialized by a manual reset and holds the contents.
2. With this setting Shmidt input is selected.

24.2.9 Noise Canceller Control Register (NCCR)

NCCR is a readable/writable 16-bit register that controls the Noise Canceller On/Off state.

Bit	Bit Name	Initial Value*	R/W	Description
15 to 1	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
0	NCCR0	0	R/W	Controls the Noise Canceller in the TRST, NMI and IRQ registers. 0 : Noise Canceller off 1 : Noise Canceller on

Note: * This register is not initialized by a manual reset and holds the contents.

Section 25 User Break Controller

The user break controller (UBC) provides functions that simplify program debugging. These functions make it easy to design an effective self-monitoring debugger, enabling the chip to debug programs without using an in-circuit emulator. Break conditions that can be set in the UBC are instruction fetch or data read/write access, data size, data contents, address value, and stop timing in the case of instruction fetch.

25.1 Features

The UBC has the following features:

1. The following break comparison conditions can be set.

   Number of break channels: two channels (channels A and B)

   User break can be requested as either the independent or sequential condition on channels A and B (sequential break setting: channel A and then channel B match with break conditions, but not in the same bus cycle).

   — Address

   Comparison bits are maskable in 1-bit units; user can mask addresses at lower 12 bits (4-k page), lower 10 bits (1-k page), or any size of page, etc.

   One of the four address buses (L-bus address (LAB), I-bus address(IAB), X-memory address bus (XAB), and Y-memory address bus (YAB)) can be selected.

   — Data

   Only on channel B, 32-bit maskable.

   One of the four data buses (L-bus data (LDB), I-bus data (IDB), X-memory data bus (XDB) and Y-memory data bus (YDB)) can be selected.

   — Bus cycle

   Instruction fetch or data access

   — Read/write

   — Operand size

   Byte, word, and longword
2. A user-designed user-break condition exception processing routine can be run.
3. In an instruction fetch cycle, it can be selected that a break is set before or after an instruction is executed.
4. Maximum repeat times for the break condition (only for channel B): $2^{12} - 1$ times.
5. Eight pairs of branch source/destination buffers.

Figure 25.1 shows a block diagram of the UBC.

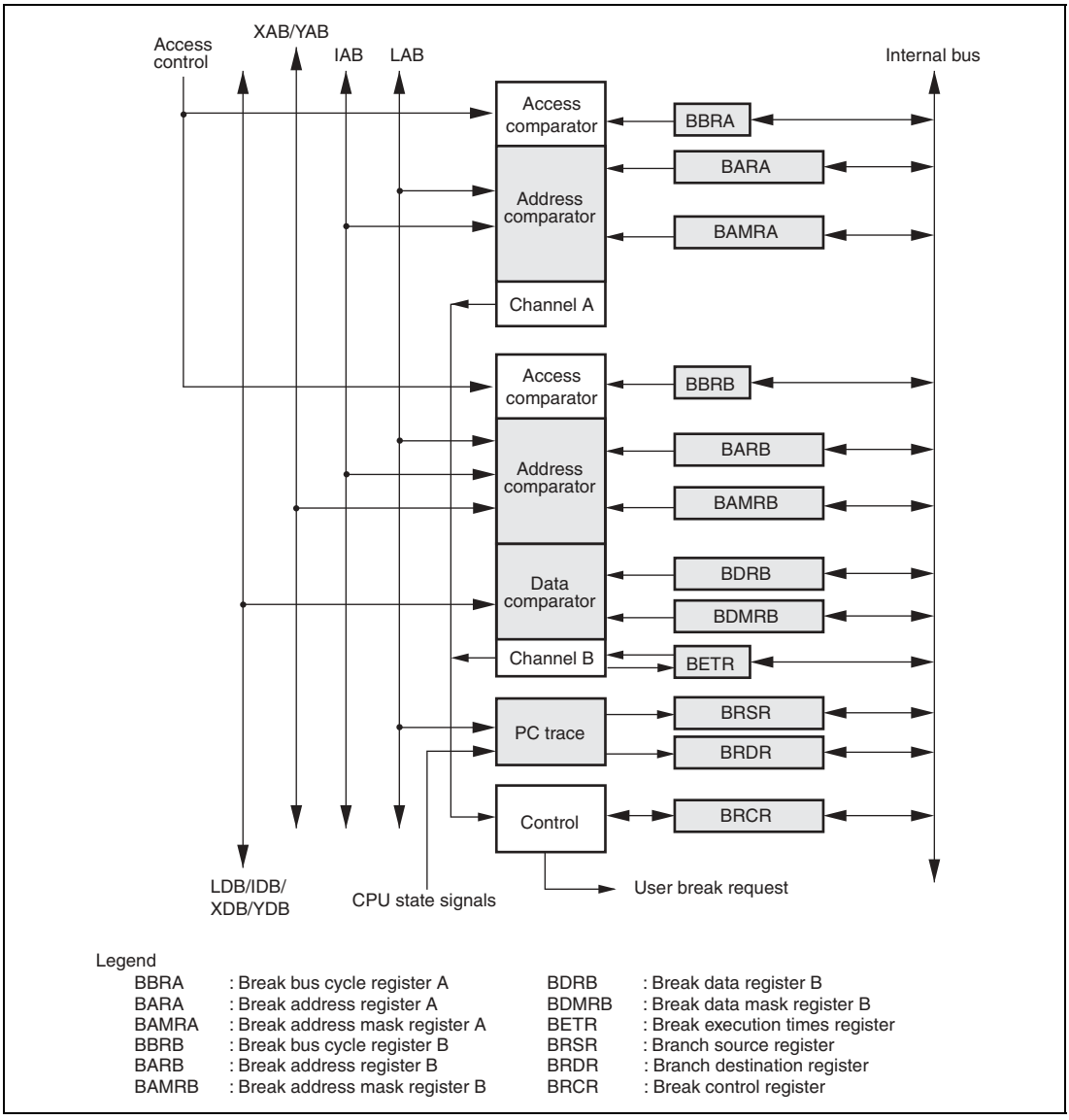


Figure 25.1 Block Diagram of User Break Controller

25.2 Register Descriptions

The UBC has the following registers. For details on register addresses and access sizes, refer to section 27, List of Registers.

- Break address register A (BARA)
- Break address mask register A (BAMRA)
- Break bus cycle register A (BBRA)
- Break address register B (BARB)
- Break address mask register B (BAMRB)
- Break bus cycle register B (BBRB)
- Break data register B (BDRB)
- Break data mask register B (BDMRB)
- Break control register (BRCR)
- Execution times break register (BETR)
- Branch source register (BRSR)
- Branch destination register (BRDR)

25.2.1 Break Address Register A (BARA)

BARA is a 32-bit readable/writable register. BARA specifies the address used as a break condition in channel A.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	BAA31 to BAA0	All 0	R/W	Break Address A Store the address on the LAB or IAB specifying break conditions of channel A.

25.2.2 Break Address Mask Register A (BAMRA)

BAMRA is a 32-bit readable/writable register. BAMRA specifies bits masked in the break address specified by BARA.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	BAMA31 to BAMA0	All 0	R/W	Break Address Mask A Specify bits masked in the channel A break address bits specified by BARA (BAA31 to BAA0). 0: Break address bit BAAn of channel A is included in the break condition 1: Break address bit BAAn of channel A is masked and is not included in the break condition Note: n = 31 to 0

25.2.3 Break Bus Cycle Register A (BBRA)

BBRA is a 16-bit readable/writable register, which specifies (1) L bus cycle or I bus cycle, (2) instruction fetch or data access, (3) read or write, and (4) operand size in the break conditions of channel A.

Bit	Bit Name	Initial Value	R/W	Description
15 to 8	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
7	CDA1	0	R/W	L Bus Cycle/I Bus Cycle Select A
6	CDA0	0	R/W	Select the L bus cycle or I bus cycle as the bus cycle of the channel A break condition. 00: Condition comparison is not performed 01: The break condition is the L bus cycle 10: The break condition is the I bus cycle 11: The break condition is the L bus cycle

Bit	Bit Name	Initial Value	R/W	Description
5	IDA1	0	R/W	Instruction Fetch/Data Access Select A
4	IDA0	0	R/W	Select the instruction fetch cycle or data access cycle as the bus cycle of the channel A break condition. 00: Condition comparison is not performed 01: The break condition is the instruction fetch cycle 10: The break condition is the data access cycle 11: The break condition is the instruction fetch cycle or data access cycle
3	RWA1	0	R/W	Read/Write Select A
2	RWA0	0	R/W	Select the read cycle or write cycle as the bus cycle of the channel A break condition. 00: Condition comparison is not performed 01: The break condition is the read cycle 10: The break condition is the write cycle 11: The break condition is the read cycle or write cycle
1	SZA1	0	R/W	Operand Size Select A
0	SZA0	0	R/W	Select the operand size of the bus cycle for the channel A break condition. 00: The break condition does not include operand size 01: The break condition is byte access 10: The break condition is word access 11: The break condition is longword access

25.2.4 Break Address Register B (BARB)

BARB is a 32-bit readable/writable register. BARB specifies the address used as a break condition in channel B. Control bits CDB1, CDB0, XYE, and XYS in BBRB select one of the four address buses for break condition B.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	BAB31 to BAB 0	All 0	R/W	Break Address B Stores an address which specifies a break condition in channel B. If the I bus or L bus is selected in BBRB, an IAB or LAB address is set in BAB31 to BAB0. If the X memory is selected in BBRB, the values in bits 15 to 1 in XAB are set in BAB31 to BAB17. In this case, the values in BAB16 to BAB0 are arbitrary. If the Y memory is selected in BBRB, the values in bits 15 to 1 in YAB are set in BAB15 to BAB1. In this case, the values in BAB31 to BAB16 are arbitrary.

Table 25.1 Specifying Break Address Register

Bus Selection in BBRB	BAB31 to BAB17	BAB16	BAB15 to BAB1	BAB0
L bus	LAB31 to LAB0			
I bus	IAB31 to IAB0			
X bus	XAB15 to XAB1	Don't care	Don't care	Don't care
Y bus	Don't care	Don't care	YAB15 to YAB1	Don't care

25.2.5 Break Address Mask Register B (BAMRB)

BAMRB is a 32-bit readable/writable register. BAMRB specifies bits masked in the break address specified by BARB.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	BAMB31 to BAMB0	All 0	R/W	Break Address Mask B Specifies bits masked in the break address of channel B specified by BARB (BAB31 to BAB0). 0: Break address BABn of channel B is included in the break condition 1: Break address BABn of channel B is masked and is not included in the break condition Note: n = 31 to 0

25.2.6 Break Data Register B (BDRB)

BDRB is a 32-bit readable/writable register. The control bits CDB1, CDB0, XYE, and XYS in BBRB select one of the four data buses for break condition B.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	BDB31 to BDB0	All 0	R/W	Break Data Bit B Stores data which specifies a break condition in channel B. If the I bus is selected in BBRB, the break data on IDB is set in BDB31 to BDB0. If the L bus is selected in BBRB, the break data on LDB is set in BDB31 to BDB0. If the X memory is selected in BBRB, the break data in bits 15 to 0 in XDB is set in BDB31 to BDB16. In this case, the values in BDB15 to BDB0 are arbitrary. If the Y memory is selected in BBRB, the break data in bits 15 to 0 in YDB are set in BDB15 to BDB0. In this case, the values in BDB31 to BDB16 are arbitrary.

Table 25.2 Specifying Break Data Register

Bus Selection in BBRB	BDB31 to BDB16	BDB15 to BDB0
L bus	LDB31 or LDB0	
I bus	IDB31 to IDB0	
X bus	XDB15 to XDB0	Don't care
Y bus	Don't care	YDB15 to YDB0

Notes: 1. Specify an operand size when including the value of the data bus in the break condition.
2. When the byte size is selected as a break condition, the same byte data must be set in bits 15 to 8 and 7 to 0 in BDRB as the break data.
3. Set the data in bits 31 to 16 when including the value of the data bus as an L-bus break condition for the MOVS.W @-As,Ds, MOVS.W @As,Ds, MOVS.W @As+,Ds, or MOVS.W @As+Ix,Ds instruction.

25.2.7 Break Data Mask Register B (BDMRB)

BDMRB is a 32-bit readable/writable register. BDMRB specifies bits masked in the break data specified by BDRB.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	BDMB31 to BDMB 0	All 0	R/W	Break Data Mask B Specifies bits masked in the break data of channel B specified by BDRB (BDB31 to BDB0). 0: Break data BDBn of channel B is included in the break condition 1: Break data BDBn of channel B is masked and is not included in the break condition Note: n = 31 to 0

Notes: 1. Specify an operand size when including the value of the data bus in the break condition.
2. When the byte size is selected as a break condition, the same byte data must be set in bits 15 to 8 and 7 to 0 in BDRB as the break mask data in BDMRB.
3. Set the mask data in bits 31 to 16 when including the value of the data bus as an L-bus break condition for the MOVS.W @-As,Ds, MOVS.W @As,Ds, MOVS.W @As+,Ds, or MOVS.W @As+Ix,Ds instruction.

25.2.8 Break Bus Cycle Register B (BBRB)

Break bus cycle register B (BBRB) is a 16-bit readable/writable register, which specifies (1) X bus or Y bus, (2) L bus cycle or I bus cycle, (3) instruction fetch or data access, (4) read or write, and (5) operand size in the break conditions of channel B.

Bit	Bit Name	Initial Value	R/W	Description
15 to 10	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
9	XYE	0	R/W	Selects the X memory bus or Y memory bus as the channel B break condition. Note that this bit setting is enabled only when the L bus is selected with the CDB1 and CDB0 bits. Selection between the X memory bus and Y memory bus is done by the XYS bit. 0: Selects L bus for the channel B break condition unconditionally 1: Selects X/Y memory bus for the channel B break condition
8	XYS	0	R/W	Selects the X bus or the Y bus as the bus of the channel B break condition. 0: Selects the X bus for the channel B break condition 1: Selects the Y bus for the channel B break condition
7	CDB1	0	R/W	L Bus Cycle/I Bus Cycle Select B
6	CDB0	0	R/W	Select the L bus cycle or I bus cycle as the bus cycle of the channel B break condition. 00: Condition comparison is not performed 01: The break condition is the L bus cycle 10: The break condition is the I bus cycle 11: The break condition is the L bus cycle
5	IDB1	0	R/W	Instruction Fetch/Data Access Select B
4	IDB0	0	R/W	Select the instruction fetch cycle or data access cycle as the bus cycle of the channel B break condition. 00: Condition comparison is not performed 01: The break condition is the instruction fetch cycle 10: The break condition is the data access cycle 11: The break condition is the instruction fetch cycle or data access cycle

Bit	Bit Name	Initial Value	R/W	Description
3	RWB1	0	R/W	Read/Write Select B
2	RWB0	0	R/W	Select the read cycle or write cycle as the bus cycle of the channel B break condition. 00: Condition comparison is not performed 01: The break condition is the read cycle 10: The break condition is the write cycle 11: The break condition is the read cycle or write cycle
1	SZB1	0	R/W	Operand Size Select B
0	SZB0	0	R/W	Select the operand size of the bus cycle for the channel B break condition. 00: The break condition does not include operand size 01: The break condition is byte access 10: The break condition is word access 11: The break condition is longword access

25.2.9 Break Control Register (BRCR)

BRCR sets the following conditions:

1. Channels A and B are used in two independent channel conditions or under the sequential condition.
2. A break is set before or after instruction execution.
3. Specify whether to include the number of execution times on channel B in comparison conditions.
4. Determine whether to include data bus on channel B in comparison conditions.
5. Enable PC trace.

BRCR is a 32-bit readable/writable register that has break conditions match flags and bits for setting a variety of break conditions.

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
15	SCMFCA	0	R/W	L Bus Cycle Condition Match Flag A When the L bus cycle condition in the break conditions set for channel A is satisfied, this flag is set to 1 (not cleared to 0). In order to clear this flag, write 0 into this bit. 0: The L bus cycle condition for channel A does not match 1: The L bus cycle condition for channel A matches
14	SCMFCB	0	R/W	L Bus Cycle Condition Match Flag B When the L bus cycle condition in the break conditions set for channel B is satisfied, this flag is set to 1 (not cleared to 0). In order to clear this flag, write 0 into this bit. 0: The L bus cycle condition for channel B does not match 1: The L bus cycle condition for channel B matches
13	SCMFDA	0	R/W	I Bus Cycle Condition Match Flag A When the I bus cycle condition in the break conditions set for channel A is satisfied, this flag is set to 1 (not cleared to 0). In order to clear this flag, write 0 into this bit. 0: The I bus cycle condition for channel A does not match 1: The I bus cycle condition for channel A matches
12	SCMFDB	0	R/W	I Bus Cycle Condition Match Flag B When the I bus cycle condition in the break conditions set for channel B is satisfied, this flag is set to 1 (not cleared to 0). In order to clear this flag, write 0 into this bit. 0: The I bus cycle condition for channel B does not match 1: The I bus cycle condition for channel B matches
11	PCTE	0	R/W	PC Trace Enable 0: Disables PC trace 1: Enables PC trace
10	PCBA	0	R/W	PC Break Select A Selects the break timing of the instruction fetch cycle for channel A as before or after instruction execution. 0: PC break of channel A is set before instruction execution 1: PC break of channel A is set after instruction execution

Bit	Bit Name	Initial Value	R/W	Description
9, 8	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
7	DBEB	0	R/W	Data Break Enable B Selects whether or not the data bus condition is included in the break condition of channel B. 0: No data bus condition is included in the condition of channel B 1: The data bus condition is included in the condition of channel B
6	PCBB	0	R/W	PC Break Select B Selects the break timing of the instruction fetch cycle for channel B as before or after instruction execution. 0: PC break of channel B is set before instruction execution 1: PC break of channel B is set after instruction execution
5, 4	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
3	SEQ	0	R/W	Sequence Condition Select Selects two conditions of channels A and B as independent or sequential conditions. 0: Channels A and B are compared under independent conditions 1: Channels A and B are compared under sequential conditions (channel A, then channel B)
2, 1	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
0	ETBE	0	R/W	Number of Execution Times Break Enable Enables the execution-times break condition only on channel B. If this bit is 1 (break enable), a user break is issued when the number of break conditions matches with the number of execution times that is specified by BETR. 0: The execution-times break condition is disabled on channel B 1: The execution-times break condition is enabled on channel B

25.2.10 Execution Times Break Register (BETR)

BETR is a 16-bit readable/writable register. When the execution-times break condition of channel B is enabled, this register specifies the number of execution times to make the break. The maximum number is $2^{12} - 1$ times. When a break condition is satisfied, it decreases BETR. A break is issued when the break condition is satisfied after BETR becomes H'0001.

Bit	Bit Name	Initial Value	R/W	Description
15 to 12	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0. If 1 is written to these bits, correct operation cannot be guaranteed.
11 to 0	BET11 to BET0	All 0	R/W	Number of Execution Times

Note: If the break condition of channel B is a break condition of instruction fetch cycle and the break instructions correspond to the following instructions, the BETR is not decremented by 1 every time when a break is occurred. Refer to the following for the decremented value.

Instruction	Countdown Value
RTE	4
DMULS.L Rm, Rn	2
DMULU.L Rm, Rn	2
MAC.L @Rm+,@Rn+	2
MAC.W @Rm+,@Rn+	2
MUL.L Rm, Rn	3
AND.B #imm,@(R0, GBR)	3
OR.B #imm,@(R0, GBR)	3
TAS.B @Rn	3
TST.B #imm,@(R0, GBR)	3
XOR.B #imm,@(R0, GBR)	3
LDC Rm,SR	4
LDC Rm,GBR	4

LDC Rm,VBR	4
LDC Rm,SSR	4
LDC Rm,SPC	4
LDC Rm,R0_BANK	4
LDC Rm,R1_BANK	4
LDC Rm,R2_BANK	4
LDC Rm,R3_BANK	4
LDC Rm,R4_BANK	4
LDC Rm,R5_BANK	4
LDC Rm,R6_BANK	4
LDC Rm,R7_BANK	4
LDC.L @Rm+,SR	6
LDC.L @Rm+,GBR	4
LDC.L @Rm+,VBR	4
LDC.L @Rm+,SSR	4
LDC.L @Rm+,SPC	4
LDC.L @Rm+,R0_BANK	4
LDC.L @Rm+,R1_BANK	4
LDC.L @Rm+,R2_BANK	4
LDC.L @Rm+,R3_BANK	4
LDC.L @Rm+,R4_BANK	4
LDC.L @Rm+,R5_BANK	4
LDC.L @Rm+,R6_BANK	4
LDC.L @Rm+,R7_BANK	4

LDC.L @Rn+,MOD	4
LDC.L @Rn+,RS	4
LDC.L @Rn+,RE	4
LDC Rn,MOD	4
LDC Rn,RS	4
LDC Rn,RE	4
BRS label	2
BSRF Rm	2
JSR @Rm	2

25.2.11 Branch Source Register (BRSR)

BRSR is a 32-bit read-only register. BRSR stores bits 27 to 0 in the address of the branch source instruction. BRSR has the flag bit that is set to 1 when a branch occurs. This flag bit is cleared to 0 when BRSR is read, the setting to enable PC trace is made, or BRSR is initialized by a power-on reset. Other bits are not initialized by a power-on reset. The eight BRSR registers have a queue structure and a stored register is shifted at every branch.

Bit	Bit Name	Initial Value	R/W	Description
31	SVF	0	R	BRSR Valid Flag Indicates whether the branch source address is stored. When a branch source address is fetched, this flag is set to 1. This flag is cleared to 0 by reading from BRSR. 0: The value of BRSR register is invalid 1: The value of BRSR register is valid
30 to 28	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
27 to 0	BSA27 to BSA0	—	R	Branch Source Address Store bits 27 to 0 of the branch source address.

25.2.12 Branch Destination Register (BRDR)

BRDR is a 32-bit read-only register. BRDR stores bits 27 to 0 in the address of the branch destination instruction. BRDR has the flag bit that is set to 1 when a branch occurs. This flag bit is cleared to 0 when BRDR is read, the setting to enable PC trace is made, or BRDR is initialized by a power-on reset. Other bits are not initialized by a power-on reset. The eight BRDR registers have a queue structure and a stored register is shifted at every branch.

Bit	Bit Name	Initial Value	R/W	Description
31	DVF	0	R	BRDR Valid Flag Indicates whether a branch destination address is stored. When a branch destination address is fetched, this flag is set to 1. This flag is cleared to 0 by reading BRDR. 0: The value of BRDR register is invalid 1: The value of BRDR register is valid
30 to 28	—	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
27 to 0	BDA27 to BDA0	—	R	Branch Destination Address Store bits 27 to 0 of the branch destination address.

25.3 Operation

25.3.1 Flow of the User Break Operation

The flow from setting of break conditions to user break exception processing is described below:

1. The break addresses are set in the break address registers (BARA or BARB). The masked addresses are set in the break address mask registers (BAMRA or BAMRB). The break data is set in the break data register (BDRB). The masked data is set in the break data mask register (BDMRB). The bus break conditions are set in the break bus cycle registers (BBRA or BBRB). Three groups of BBRA or BBRB (L bus cycle/I bus cycle select, instruction fetch/data access select, and read/write select) are each set. No user break will be generated if even one of these groups is set with 00. The respective conditions are set in the bits of the break control register (BRCR). Make sure to set all registers related to breaks before setting BBRA or BBRB.
2. When the break conditions are satisfied, the UBC sends a user break request to the CPU and sets the L bus condition match flag (SCMFCA or SCMFCB) and the I bus condition match flag (SCMFDA or SCMFDB) for the appropriate channel. When the X/Y memory bus is specified for channel B, SCMFCB is used for the condition match flag.
3. The appropriate condition match flags (SCMFCA, SCMFDA, SCMFCB, and SCMFDB) can be used to check if the set conditions match or not. The matching of the conditions sets flags, but they are not reset. 0 must first be written to them before they can be used again.
4. There is a chance that the break set in channel A and the break set in channel B occur around the same time. In this case, there will be only one break request to the CPU, but these two break channel match flags could be both set.
5. When selecting the I bus as the break condition, note the following:
   - Several bus masters, including the CPU and DMAC, are connected to the I bus. The UBC monitors bus cycles generated by all bus masters, and determines the condition match.
   - Physical addresses are used for the I bus. Set a physical address in break address registers (BARA and BARB). The bus cycles for logical addresses issued on the L bus by the CPU are converted to physical addresses before being output to the I bus. (If the address translation function is enabled, address translation by the MMU is carried out.)
   - For data access cycles issued on the L bus by the CPU, if their logical addresses are not to be cached, they are issued with the data size specified on the L bus and their addresses are not rounded.
   - For instruction fetch cycles issued on the L bus by the CPU, even though their logical addresses are not to be cached, they are issued in longwords and their addresses are rounded to match longword boundaries.

— If a logical address issued on the L bus by the CPU is an address to be cached and a cache miss occurs, its bus cycle is issued as a cache fill cycle on the I bus. In this case, it is issued in longwords and its address is rounded to match longword boundaries. However note that cache fill is not performed for a write miss in write through mode. In this case, the bus cycle is issued with the data size specified on the L bus and its address is not rounded. In write back mode, a write back cycle may be issued in addition to a read fill cycle. It is a longword bus cycle whose address is rounded to match longword boundaries.

— I bus cycles (including read fill cycles) resulting from instruction fetches on the L bus by the CPU are defined as instruction fetch cycles on the I bus, while other bus cycles are defined as data access cycles.

— The DMAC only issues data access cycles for I bus cycles.

— If a break condition is specified for the I bus, even when the condition matches in an I bus cycle resulting from an instruction executed by the CPU, at which instruction the break is to be accepted cannot be clearly defined.

6. While the block bit (BL) in the CPU status register (SR) is set to 1, no breaks can be accepted. However, condition determination will be carried out, and if the condition matches, the corresponding condition match flag is set to 1.

25.3.2 Break on Instruction Fetch Cycle

1. When L bus/instruction fetch/read/word or longword is set in the break bus cycle register (BBRA or BBRB), the break condition becomes the L bus instruction fetch cycle. Whether it breaks before or after the execution of the instruction can then be selected with the PCBA or PCBB bit of the break control register (BRCR) for the appropriate channel. If an instruction fetch cycle is set as a break condition, clear LSB in the break address register (BARA or BARB) to 0. A break cannot be generated as long as this bit is set to 1.
2. An instruction set for a break before execution breaks when it is confirmed that the instruction has been fetched and will be executed. This means this feature cannot be used on instructions fetched by overrun (instructions fetched at a branch or during an interrupt transition, but not to be executed). When this kind of break is set for the delay slot of a delayed branch instruction, the break is generated prior to execution of the delayed branch instruction.

   Note: If a branch does not occur at a delay condition branch instruction, the subsequent instruction is not recognized as a delay slot.

3. When the condition is specified to be occurred after execution, the instruction set with the break condition is executed and then the break is generated prior to the execution of the next instruction. As with pre-execution breaks, this cannot be used with overrun fetch instructions. When this kind of break is set for a delayed branch instruction and its delay slot, a break is not generated until the first instruction at the branch destination.
4. When an instruction fetch cycle is set for channel B, the break data register B (BDRB) is ignored. Therefore, break data cannot be set for the break of the instruction fetch cycle.

5. If the I bus is set for a break of an instruction fetch cycle, the condition is determined for the instruction fetch cycles on the I bus. For details, see 5 in section 25.3.1, Flow of the User Break Operation.

25.3.3 Break on Data Access Cycle

1. If the L bus is specified as a break condition for data access break, condition comparison is performed for the logical addresses (and data) accessed by the executed instructions, and a break occurs if the condition is satisfied. If the I bus is specified as a break condition, condition comparison is performed for the physical addresses (and data) of the data access cycles that are issued on the I bus by all bus masters including the CPU, and a break occurs if the condition is satisfied. For details on the CPU bus cycles issued on the I bus, see 5 in section 25.3.1, Flow of the User Break Operation.
2. The relationship between the data access cycle address and the comparison condition for each operand size is listed in table 25.3.

Table 25.3 Data Access Cycle Addresses and Operand Size Comparison Conditions

Access Size	Address Compared
Longword	Compares break address register bits 31 to 2 to address bus bits 31 to 2
Word	Compares break address register bits 31 to 1 to address bus bits 31 to 1
Byte	Compares break address register bits 31 to 0 to address bus bits 31 to 0

This means that when address H'00001003 is set in the break address register (BARA or BARB), for example, the bus cycle in which the break condition is satisfied is as follows (where other conditions are met).

Longword access at H'00001000

Word access at H'00001002

Byte access at H'00001003

3. When the data value is included in the break conditions on channel B:

   When the data value is included in the break conditions, either longword, word, or byte is specified as the operand size of the break bus cycle register B (BBRB). When data values are included in break conditions, a break is generated when the address conditions and data conditions both match. To specify byte data for this case, set the same data in two bytes at bits 15 to 8 and bits 7 to 0 of the break data register B (BDRB) and break data mask register B (BDMRB). When word or byte is set, bits 31 to 16 of BDRB and BDMRB are ignored. Set the word data in bits 31 to 16 in BDRB and BDMRB when including the value of the data bus as a break condition for the MOVS.W @-As,Ds, MOVS.W @As,Ds, MOVS.W @As+,Ds, or MOVS.W @As+Ix,Ds instruction (bits 15 to 0 are ignored).

4. Access by a PREF instruction is handled as read access in longword units without access data. Therefore, if including the value of the data bus when a PREF instruction is specified as a break condition, a break will not occur.
5. If the L bus is selected, a break occurs on ending execution of the instruction that matches the break condition, and immediately before the next instruction is executed. However, when data is also specified as the break condition, the break may occur on ending execution of the instruction following the instruction that matches the break condition. If the I bus is selected, the instruction at which the break will occur cannot be determined. When this kind of break occurs at a delayed branch instruction or its delay slot, the break may not actually take place until the first instruction at the branch destination.

25.3.4 Break on X/Y-Memory Bus Cycle

1. The break condition on an X/Y-memory bus cycle is specified only in channel B. If the XYE bit in BBRB is set to 1, the break address and break data on X/Y-memory bus are selected. At this time, select the X-memory bus or Y-memory bus by specifying the XYS bit in BBRB. The break condition cannot include both X-memory and Y-memory at the same time. The break condition is applied to an X/Y-memory bus cycle by specifying L bus/data access cycle/read or write/word or no specified operand size (in bits 7 to 0) in the break bus cycle register B (BBRB).
2. When an X-memory address is selected as the break condition, specify an X-memory address in the upper 16 bits in BARB and BAMRB. When a Y-memory address is selected, specify a Y-memory address in the lower 16 bits. Specification of X/Y-memory data is the same for BDRB and BDMRB.
3. The timing of a data access break for the X memory or Y memory bus to occur is the same as a data access break of the L bus. For details, see 5 in section 25.3.3, Break on Data Access Cycle.

25.3.5 Sequential Break

1. By setting the SEQ bit in BRCR to 1, the sequential break is issued when a channel B break condition matches after a channel A break condition matches. A user break is not generated even if a channel B break condition matches before a channel A break condition matches. When channels A and B conditions match at the same time, the sequential break is not issued. To clear the channel A condition match when a channel A condition match has occurred but a channel B condition match has not yet occurred in a sequential break specification, clear the SEQ bit in BRCR to 0.
2. In sequential break specification, the L/I/X/Y bus can be selected and the execution times break condition can be also specified. For example, when the execution times break condition is specified, the break condition is satisfied when a channel B condition matches with BETR = H'0001 after a channel A condition has matched.

25.3.6 Value of Saved Program Counter

When a break occurs, the address of the instruction from where execution is to be resumed is saved in the SPC, and the exception handling state is entered. If the L bus is specified as a break condition, the instruction at which the break should occur can be clearly determined (except for when data is included in the break condition). If the I bus is specified as a break condition, the instruction at which the break should occur cannot be clearly determined.

1. When instruction fetch (before instruction execution) is specified as a break condition:

   The address of the instruction that matched the break condition is saved in the SPC. The instruction that matched the condition is not executed, and the break occurs before it. However when a delay slot instruction matches the condition, the address of the delayed branch instruction is saved in the SPC.
2. When instruction fetch (after instruction execution) is specified as a break condition:

   The address of the instruction following the instruction that matched the break condition is saved in the SPC. The instruction that matches the condition is executed, and the break occurs before the next instruction is executed. However when a delayed branch instruction or delay slot matches the condition, these instructions are executed, and the branch destination address is saved in the SPC.
3. When data access (address only) is specified as a break condition:

   The address of the instruction immediately after the instruction that matched the break condition is saved in the SPC. The instruction that matches the condition is executed, and the break occurs before the next instruction is executed. However when a delay slot instruction matches the condition, the branch destination address is saved in the SPC.
4. When data access (address + data) is specified as a break condition:

   When a data value is added to the break conditions, the address of an instruction that is within two instructions of the instruction that matched the break condition is saved in the SPC. At which instruction the break occurs cannot be determined accurately.

   When a delay slot instruction matches the condition, the branch destination address is saved in the SPC. If the instruction following the instruction that matches the break condition is a branch instruction, the break may occur after the branch instruction or delay slot has finished. In this case, the branch destination address is saved in the SPC.

25.3.7 PC Trace

1. Setting PCTE in BRCR to 1 enables PC traces. When branch (branch instruction, and interrupt exception) is generated, the branch source address and branch destination address are stored in BRSR and BRDR, respectively.
2. The values stored in BRSR and BRDR are as given below due to the kind of branch.
   — If a branch occurs due to a branch instruction, the address of the branch instruction is saved in BRSR and the address of the branch destination instruction is saved in BRDR.
   — If a branch occurs due to an interrupt or exception, the value saved in SPC due to exception occurrence is saved in BRSR and the start address of the exception handling routine is saved in BRDR.

   When a repeat loop of the DSP extended function is used, control being transferred from the repeat end instruction to the repeat start instruction is not recognized as a branch, and the values are not stored in BRSR and BRDR.
3. BRSR and BRDR have eight pairs of queue structures. The top of queues is read first when the address stored in the PC trace register is read. BRSR and BRDR share the read pointer. Read BRSR and BRDR in order, the queue only shifts after BRDR is read. After switching the PCTE bit (in BRCR) off and on, the values in the queues are invalid.

25.3.8 Usage Examples

Break Condition Specified for L Bus Instruction Fetch Cycle:

(Example 1-1)

- Register specifications

  BARA = H'00000404, BAMRA = H'00000000, BBRA = H'0054, BARB = H'00008010, BAMRB = H'00000006, BBRB = H'0054, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00300400

  Specified conditions: Channel A/channel B independent mode

  — Channel A

    Address: H'00000404, Address mask: H'00000000

    Bus cycle: L bus/instruction fetch (after instruction execution)/read (operand size is not included in the condition)

  — Channel B

    Address: H'00008010, Address mask: H'00000006

    Data: H'00000000, Data mask: H'00000000

    Bus cycle: L bus/instruction fetch (before instruction execution)/read (operand size is not included in the condition)

A user break occurs after an instruction of address H'00000404 is executed or before instructions of addresses H'00008010 to H'00008016 are executed.

(Example 1-2)

- Register specifications

  BARA = H'00037226, BAMRA = H'00000000, BBRA = H'0056, BARB = H'0003722E, BAMRB = H'00000000, BBRB = H'0056, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00000008

  Specified conditions: Channel A/channel B sequential mode

  — Channel A

  Address: H'00037226, Address mask: H'00000000

  Bus cycle: L bus/instruction fetch (before instruction execution)/read/word

  — Channel B

  Address: H'0003722E, Address mask: H'00000000

  Data: H'00000000, Data mask: H'00000000

  Bus cycle: L bus/instruction fetch (before instruction execution)/read/word

  After an instruction with address H'00037226 is executed, a user break occurs before an instruction with address H'0003722E is executed.

(Example 1-3)

- Register specifications

  BARA = H'00027128, BAMRA = H'00000000, BBRA = H'005A, BARB = H'00031415, BAMRB = H'00000000, BBRB = H'0054, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00300000

  Specified conditions: Channel A/channel B independent mode

  — Channel A

  Address: H'00027128, Address mask: H'00000000

  Bus cycle: L bus/instruction fetch (before instruction execution)/write/word

  — Channel B

  Address: H'00031415, Address mask: H'00000000

  Data: H'00000000, Data mask: H'00000000

  Bus cycle: L bus/instruction fetch (before instruction execution)/read (operand size is not included in the condition)

  On channel A, no user break occurs since instruction fetch is not a write cycle. On channel B, no user break occurs since instruction fetch is performed for an even address.

(Example 1-4)

- Register specifications

  BARA = H'00037226, BAMRA = H'00000000, BBRA = H'005A, BARB = H'0003722E, BAMRB = H'00000000, BBRB = H'0056, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00000008

  Specified conditions: Channel A/channel B sequential mode

  — Channel A

    Address: H'00037226, Address mask: H'00000000

    Bus cycle: L bus/instruction fetch (before instruction execution)/write/word

  — Channel B

    Address: H'0003722E, Address mask: H'00000000

    Data: H'00000000, Data mask: H'00000000

    Bus cycle: L bus/instruction fetch (before instruction execution)/read/word

    Since instruction fetch is not a write cycle on channel A, a sequential condition does not match. Therefore, no user break occurs.

(Example 1-5)

- Register specifications

  BARA = H'00000500, BAMRA = H'00000000, BBRA = H'0057, BARB = H'00001000, BAMRB = H'00000000, BBRB = H'0057, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00300001, BETR = H'0005

  Specified conditions: Channel A/channel B independent mode

  — Channel A

    Address: H'00000500, Address mask: H'00000000

    Bus cycle: L bus/instruction fetch (before instruction execution)/read/longword

  — Channel B

    Address: H'00001000, Address mask: H'00000000

    Data: H'00000000, Data mask: H'00000000

    Bus cycle: L bus/instruction fetch (before instruction execution)/read/longword

    The number of execution-times break enable (5 times)

    On channel A, a user break occurs before an instruction of address H'00000500 is executed. On channel B, a user break occurs after the instruction of address H'00001000 are executed four times and before the fifth time.

(Example 1-6)

- Register specifications

  BARA = H'00008404, BAMRA = H'00000FFF, BBRA = H'0054, BARB = H'00008010, BAMRB = H'00000006, BBRB = H'0054, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00000400

  Specified conditions: Channel A/channel B independent mode

  — Channel A

    Address: H'00008404, Address mask: H'00000FFF

    Bus cycle: L bus/instruction fetch (after instruction execution)/read (operand size is not included in the condition)

  — Channel B

    Address: H'00008010, Address mask: H'00000006

    Data: H'00000000, Data mask: H'00000000

    Bus cycle: L bus/instruction fetch (before instruction execution)/read (operand size is not included in the condition)

    A user break occurs after an instruction with addresses H'00008000 to H'00008FFE is executed or before an instruction with addresses H'00008010 to H'00008016 are executed.

Break Condition Specified for L Bus Data Access Cycle:

(Example 2-1)

- Register specifications

  BARA = H'00123456, BAMRA = H'00000000, BBRA = H'0064, BARB = H'000ABCDE, BAMRB = H'000000FF, BBRB = H'006A, BDRB = H'0000A512, BDMRB = H'00000000, BRCR = H'00000080

  Specified conditions: Channel A/channel B independent mode

  — Channel A

    Address: H'00123456, Address mask: H'00000000

    Bus cycle: L bus/data access/read (operand size is not included in the condition)

  — Channel B

    Address: H'000ABCDE, Address mask: H'000000FF

    Data: H'0000A512, Data mask: H'00000000

    Bus cycle: L bus/data access/write/word

    On channel A, a user break occurs with longword read from address H'00123454, word read from address H'00123456, or byte read from address H'00123456. On channel B, a user break occurs when word H'A512 is written in addresses H'000ABC00 to H'000ABCFE.

(Example 2-2)

- Register specifications

  BARA = H'01000000, BAMRA = H'00000000, BBRA = H'0066, BARB = H'0000F000, BAMRB = H'FFFF0000, BBRB = H'036A, BDRB = H'00004567, BDMRB = H'00000000, BRCR = H'00300080

  Specified conditions: Channel A/channel B independent mode

  — Channel A

    Address: H'01000000, Address mask: H'00000000

    Bus cycle: L bus/data access/read/word

  — Channel B

    Y Address: H'0000F000, Address mask: H'FFFF0000

    Data: H'00004567, Data mask: H'00000000

    Bus cycle: Y bus/data access/write/word

    On channel A, a user break occurs during word read from address H'01000000 in the memory space. On channel B, a user break occurs when word data H'4567 is written in address H'0000F000 in the Y memory space. The X/Y-memory space is changed by a mode setting.

Break Condition Specified for I Bus Data Access Cycle:

(Example 3-1)

- Register specifications

  BARA = H'00314156, BAMRA = H'00000000, BBRA = H'0094, BARB = H'00055555, BAMRB = H'00000000, BBRB = H'00A9, BDRB = H'00007878, BDMRB = H'00000F0F, BRCR = H'00000080

  Specified conditions: Channel A/channel B independent mode

  — Channel A

    Address: H'00314156, Address mask: H'00000000

    Bus cycle: I bus/instruction fetch/read (operand size is not included in the condition)

  — Channel B

    Address: H'00055555, Address mask: H'00000000

    Data: H'00000078, Data mask: H'0000000F

    Bus cycle: I bus/data access/write/byte

    On channel A, a user break occurs when instruction fetch is performed for address H'00314156 in the memory space.

    On channel B, a user break occurs when the I bus writes byte data H'7 in address H'00055555.

25.4 Usage Notes

1. The CPU can read from or write to the UBC registers via the I bus. Accordingly, during the period from executing an instruction to rewrite the UBC register till the new value is actually rewritten, the desired break may not occur. In order to know the timing when the UBC register is changed, read from the last written register. Instructions after then are valid for the newly written register value.
2. UBC cannot monitor access to the L bus and I bus in the same channel.
3. Note on specification of sequential break:

   A condition match occurs when a B-channel match occurs in a bus cycle after an A-channel match occurs in another bus cycle in sequential break setting. Therefore, no break occurs even if a bus cycle, in which an A-channel match and a channel B match occur simultaneously, is set.
4. When a user break and another exception occur at the same instruction, which has higher priority is determined according to the priority levels defined in table 4.1 in section 4, Exception Handling. If an exception with higher priority occurs, the user break is not generated.
   - Pre-execution break has the highest priority.
   - When a post-execution break or data access break occurs simultaneously with a re-execution-type exception (including pre-execution break) that has higher priority, the re-execution-type exception is accepted, and the condition match flag is not set (see the exception in the following note). The break will occur and the condition match flag will be set only after the exception source of the re-execution-type exception has been cleared by the exception handling routine and re-execution of the same instruction has ended.
   - When a post-execution break or data access break occurs simultaneously with a completion-type exception (TRAPA) that has higher priority, though a break does not occur, the condition match flag is set.
5. Note the following exception for the above note.

   If a post-execution break or data access break is satisfied by an instruction that generates a CPU address error by data access, the CPU address error is given priority to the break. Note that the UBC condition match flag is set in this case.
6. Note the following when a break occurs in a delay slot.

   If a pre-execution break is set at the delay slot instruction of the RTE instruction, the break does not occur until the branch destination of the RTE instruction.
7. User breaks are disabled during UBC module standby mode. Do not read from or write to the UBC registers during UBC module standby mode; the values are not guaranteed.
8. When the repeat loop of the DSP extended function is used, even though a break condition is satisfied during execution of the entire repeat loop or several instructions in the repeat loop, the break may be held. For details, see section 4, Exception Handling.

This LSI incorporates a user debugging interface (H-UDI) and advanced user debugger (AUD) for a boundary scan function and emulator support.

This section summarizes the boundary scan function for the H-UDI. For details on the JTAG function incorporated in this LSI, refer to the material separately handed, BSDL (Boundary-Scan Description Language). For details on the emulator functions such as the AUD, refer to the user's manual of each emulator.

26.1 Features

The H-UDI is a serial I/O interface that conforms to JTAG (Joint Test Action Group, IEEE Standard 1149.1 and IEEE Standard Test Access Port and Boundary-Scan Architecture) specifications.

The H-UDI in this LSI supports a boundary scan mode, and is also used for emulator connection.

When using an emulator, H-UDI functions should not be used. When using the boundary scan function for the H-UDI, the $\overline{\text{ASEMD0}}$ pin should be fixed to high level. For the method of connecting the emulator, refer to the emulator's user's manual.

Figure 26.1 shows a block diagram of the H-UDI.

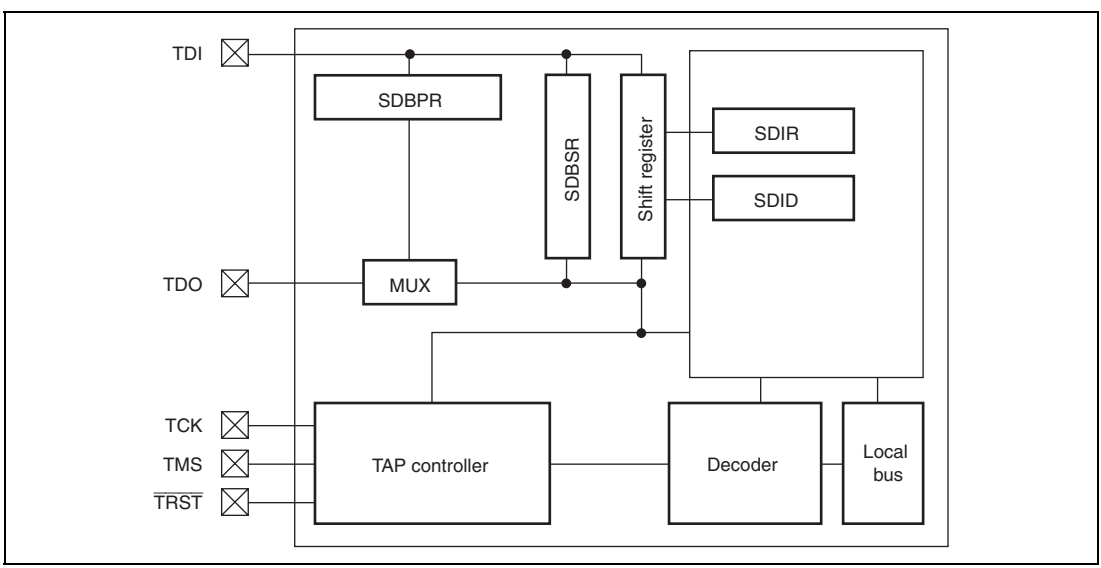


Figure 26.1 Block Diagram of H-UDI

26.2 Input/Output Pins

Table 26.1 shows the pin configuration of the H-UDI.

Table 26.1 Pin Configuration

Pin Name	I/O	Description
TCK	Input	Serial Data Input/Output Clock Pin Data is serially supplied to the H-UDI from the data input pin (TDI), and output from the data output pin (TDO), in synchronization with this clock.
TMS	Input	Mode Select Input Pin The state of the TAP control circuit is determined by changing this signal in synchronization with TCK. The protocol conforms to the JTAG standard (IEEE Std.1149.1).
$\overline{\text{TRST}}$	Input	Reset Input Pin Input is accepted asynchronously with respect to TCK, and when low, the H-UDI is reset. $\overline{\text{TRST}}$ must be low for a constant period when power is turned on regardless of using the H-UDI function. This is different from the JTAG standard. See section 26.4.2, Reset Configuration, for more information.
TDI	Input	Serial Data Input Pin Data transfer to the H-UDI is executed by changing this signal in synchronization with TCK.
TDO	Output	Serial Data Output Pin Data read from the H-UDI is executed by reading this pin in synchronization with TCK. The data output timing depends on the command type set in the SDIR. See section 26.3.2 Instruction Register (SDIR), for more information.
$\overline{\text{ASEMD0}}$	Input	ASE Mode Select Pin If a low level is input at the $\overline{\text{ASEMD0}}$ pin while the $\overline{\text{RESETP}}$ pin is asserted, ASE mode is entered; if a high level is input, normal mode is entered and the JTAG function can be used. In ASE mode, the dedicated emulator function can be used. The input to the $\overline{\text{ASEMD0}}$ pin should be fixed to either high or low level.
$\overline{\text{ASEBRKAK}}$, $\overline{\text{AUDSYNC}}$, AUDATA3 to AUDATA0, AUDCK	Output	Dedicated Emulator Pin

Note: The $\overline{\text{ASEMD0}}$ pin should be fixed to high level when using the boundary scan function.

26.3 Register Descriptions

The H-UDI has the following registers. For details on register addresses and register states in each processing state, see section 27, List of Registers*.

- Bypass register (SDBPR)
- Instruction register (SDIR)
- Boundary scan register (SDBSR)
- ID register (SDID)

Note: * The H-UDI registers can be accessed from the CPU only when the emulator function is enabled in ASE mode. These registers function as part of the dedicated circuit for the JTAG function in normal mode (mode in which boundary scan function is used). Therefore, these registers should not be accessed from the CPU in normal mode.

26.3.1 Bypass Register (SDBPR)

SDBPR is a 1-bit register that can be accessed only via the H-UDI pin. This register cannot be accessed from the CPU. When SDIR is set to bypass mode, SDBPR is connected between H-UDI pins TDI and TDO. The initial value is undefined.

26.3.2 Instruction Register (SDIR)

SDIR is a 3-bit register in normal mode. This register can be written only via the H-UDI pin. This register cannot be accessed from the CPU. Undefined value is read even when this register is accessed from the CPU using the address which is listed in section 27, List of Registers. JTAG IDCODE is set in SDIR as the initial value. Operation is not guaranteed if a reserved command is set in SDIR.

In ASE mode, SDIR is a 16-bit read-only register read from the CPU. The JTAG function is disabled. For details on the function of SDIR in ASE mode, refer to the emulator's user's manual.

Bit	Bit Name	Initial Value	R/W	Description
2	TI2	1	—	Test Instruction 2 to 0 The JTAG instruction is transferred to SDIR by a serial input from TDI. Cannot be accessed from the CPU for either read or write. For the commands, see table 26.2.
1	TI1	0	—	
0	TI0	0	—	

Table 26.2 JTAG Commands

Bits 2 to 0			
TI2	**TI1**	**TI0**	**Description**
0	0	0	JTAG EXTEST
0	0	1	JTAG SAMPLE/PRELOAD
0	1	0	Reserved
0	1	1	Reserved
1	0	0	JTAG IDCODE (Initial value)
1	0	1	Reserved
1	1	0	Reserved
1	1	1	JTAG BYPASS

26.3.3 Boundary Scan Register (SDBSR)

SDBSR is a shift register, located on the PAD, for controlling the input/output pins of this LSI. The initial value is undefined, and this register cannot be accessed from the CPU.

Using the EXTEST and SAMPLE/PRELOAD commands, a boundary scan test conforming to the JTAG standard can be carried out. Table 26.3 gives the correspondence between the pins of the SH7660 and boundary scan register.

Table 26.3 Correspondence between Pins of SH7660 and Boundary Scan Register

Bit	Pin Name	I/O	Bit	Pin Name	I/O
0	$\overline{\text{ASEBRKAK}}$	IN	33	PTG4/SCIF0_$\overline{\text{CTS}}$	IN
1		Control	34		Control
2		OUT	35		OUT
3	PTA0/SIOF_MCLK	IN	36	PTH1/SCIF1_TxD	IN
4		Control	37		Control
5		OUT	38		OUT
6	PTA1/SIOF_TXD	IN	39	PTH2/SCIF1_RxD	IN
7		Control	40		Control
8		OUT	41		OUT
9	PTA2/SIOF_RXD	IN	42	PTH3/SCIF1_$\overline{\text{RTS}}$	IN
10		Control	43		Control
11		OUT	44		OUT
12	PTA3/SIOF_SYNc (SIOF_SS0)	IN	45	PTH4/SCIF1_$\overline{\text{CTS}}$	IN
13		Control	46		Control
14		OUT	47		OUT
15	PTA4/SIOF_SCK	IN	48	NMI	IN
16		Control	49	PTH0/SCIF1_SCK/IRQ4	IN
17		OUT	50		Control
18	PTA5/SIOF_$\overline{\text{SS1}}$	IN	51		OUT
19		Control	52	PTG0/SCIF0_SCK/IRQ3	IN
20		OUT	53		Control
21	PTA6/SIOF_$\overline{\text{SS2}}$	IN	54		OUT
22		Control	55	PTB1/IRQ2	IN
23		OUT	56		Control
24	PTG1/SCIF0_TxD	IN	57		OUT
25		Control	58	PTB0/IRQ0	IN
26		OUT	59		Control
27	PTG2/SCIF0_RxD	IN	60		OUT
28		Control	61	RTCSEL0	IN
29		OUT	62	$\overline{\text{USB_PWR_EN}}$/USB_PULLUP	IN
30	PTG3/SCIF0_$\overline{\text{RTS}}$	IN	63		Control
31		Control	64		OUT
32		OUT	65	UCLK	IN

Bit	Pin Name	I/O	Bit	Pin Name	I/O
66	UCLK	Control	100	$\overline{\text{AUDSYNC}}$	OUT
67		OUT	101	AUDATA3	IN
68	$\overline{\text{USB_OVR_CRNT}}$/USB_VBUS	IN	102		Control
69		Control	103		OUT
70		OUT	104	AUDATA2	IN
71	CKIO	IN	105		Control
72		Control	106		OUT
73		OUT	107	AUDATA1	IN
74	MD5	IN	108		Control
75	MD2	IN	109		OUT
76	MD1	IN	110	AUDATA0	IN
77	PTE6/RDI_CTRL2/$\overline{\text{IRQOUT}}$	IN	111		Control
78		Control	112		OUT
79		OUT	113	AUDCK	IN
80	PTE0/VCI_CODEC_PWRDWN /TEND	IN	114		Control
81		Control	115		OUT
82		OUT	116	$\overline{\text{RD}}$	IN
83	PTE1/VCI_SCO_TX/DACK	IN	117		Control
84		Control	118		OUT
85		OUT	119	PTB7/$\overline{\text{BS}}$	IN
86	PTE2/VCI_SCO_RX/DREQ	IN	120		Control
87		Control	121		OUT
88		OUT	122	PTB6/$\overline{\text{REFOUT}}$	IN
89	PTE3/VCI_SCO_CLK_OUT /$\overline{\text{BACK}}$	IN	123		Control
90		Control	124		OUT
91		OUT	125	$\overline{\text{WAIT}}$	IN
92	PTE4/VCI_SCO_SYNC_OUT /$\overline{\text{BREQ}}$	IN	126	$\overline{\text{WE1}}$/DQM1	IN
93		Control	127		Control
94		OUT	128		OUT
95	PTE5/VCI_HWC	IN	129	$\overline{\text{WE0}}$/DQM0	IN
96		Control	130		Control
97		OUT	131		OUT
98	$\overline{\text{AUDSYNC}}$	IN	132	PTB2/CKE	IN
99		Control	133		Control

Bit	Pin Name	I/O	Bit	Pin Name	I/O
134	PTB2/CKE	OUT	168	A14	Control
135	PTB3/$\overline{CAS}$	IN	169	A13	OUT
136		Control	170		Control
137		OUT	171	A12	OUT
138	PTB4/$\overline{RAS}$	IN	172		Control
139		Control	173	A11	OUT
140		OUT	174		Control
141	PTB5/RD/$\overline{WR}$	IN	175	A10	OUT
142		Control	176		Control
143		OUT	177	A9	OUT
144	PTD5/A23	IN	178		Control
145		Control	179	A8	OUT
146		OUT	180		Control
147	PTD4/A22	IN	181	A7	OUT
148		Control	182		Control
149		OUT	183	A6	OUT
150	PTD3/A21	IN	184		Control
151		Control	185	A5	OUT
152		OUT	186		Control
153	PTD2/A20	IN	187	A4	OUT
154		Control	188		Control
155		OUT	189	A3	OUT
156	PTD1/A19	IN	190		Control
157		Control	191	A2	OUT
158		OUT	192		Control
159	A18	Control	193	A1	OUT
160		OUT	194		Control
161	A17	Control	195	PTD0/A0	IN
162		OUT	196		Control
163	A16	OUT	197		OUT
164		Control	198	D15	IN
165	A15	OUT	199		Control
166		Control	200		OUT
167	A14	OUT	201	D14	IN

Bit	Pin Name	I/O	Bit	Pin Name	I/O
202	D14	Control	229	D5	Control
203		OUT	230		OUT
204	D13	IN	231	D4	IN
205		Control	232		Control
206		OUT	233		OUT
207	D12	IN	234	D3	IN
208		Control	235		Control
209		OUT	236		OUT
210	D11	IN	237	D2	IN
211		Control	238		Control
212		OUT	239		OUT
213	D10	IN	240	D1	IN
214		Control	241		Control
215		OUT	242		OUT
216	D9	IN	243	D0	IN
217		Control	244		Control
218		OUT	245		OUT
219	D8	IN	246	$\overline{CS4}$	IN
220		Control	247		Control
221		OUT	248		OUT
222	D7	IN	249	$\overline{CS3}$	IN
223		Control	250		Control
224		OUT	251		OUT
225	D6	IN	252	$\overline{CS0}$	IN
226		Control	253		Control
227		OUT	254		OUT
228	D5	IN	255	$\overline{BOOT_E}$	IN

26.3.4 ID Register (SDID)

SDID is a 32-bit register that consists of the version, parts number, manufacturer, and fixed code. SDID can be read from the CPU even in normal mode as two 16-bit registers, SDIDH and SDIDL. However, SDID cannot be written to.

SDID cannot be accessed from the H-UDI pin.

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	DID31 to DID0	Refer to description	R	Device ID31 to ID0 Device ID register that is stipulated by JTAG. H′0028200F (initial value) for this LSI. The upper four bits may be changed by the chip version. SDIDH corresponds to bits 31 to 16. SDIDL corresponds to bits 15 to 0.

26.4 Operation

26.4.1 TAP Controller

Figure 26.2 shows the internal states of the TAP controller. State transitions basically conform to the JTAG standard.

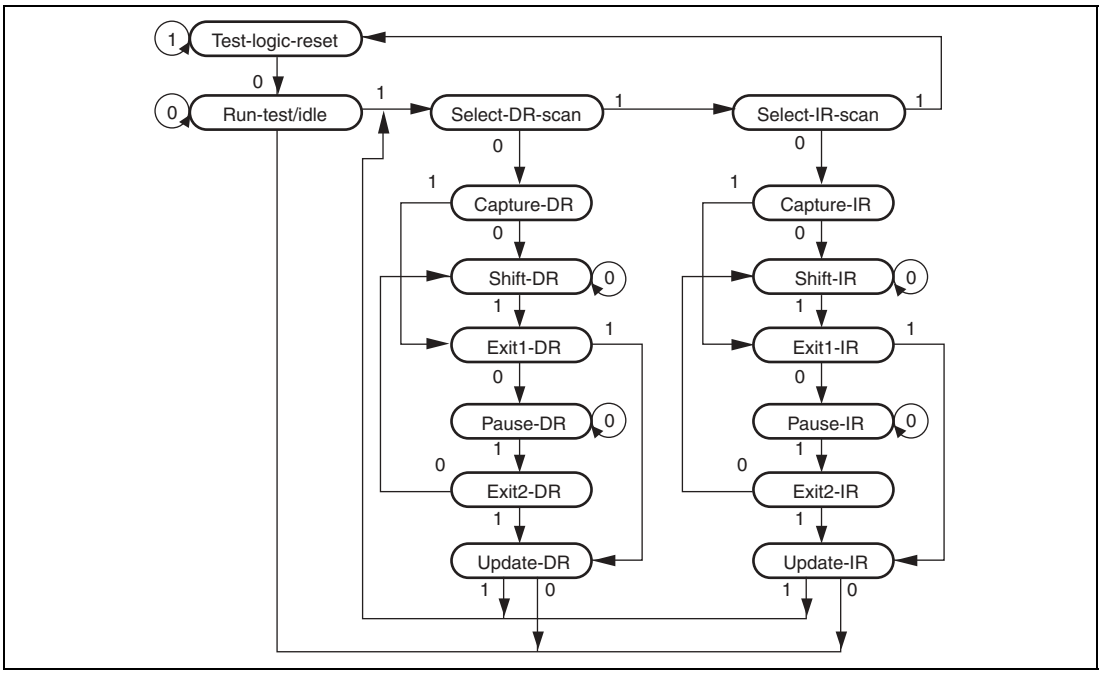


Figure 26.2 TAP Controller State Transitions

26.4.2 Reset Configuration

Table 26.4 lists the reset configuration of this LSI.

Table 26.4 Reset Configuration

$\overline{\text{ASEMD0}}$*[1]	$\overline{\text{RESETP}}$	$\overline{\text{TRST}}$	Chip State
H	L	L	Normal reset and H-UDI reset
		H	Normal reset
	H	L	H-UDI reset only
		H	Normal operation
L	L	L	Reset hold*[2]
		H	Normal reset
	H	L	H-UDI reset only
		H	Normal operation

Notes: 1. Performs normal mode or ASE mode settings
$\overline{\text{ASEMD0}}$ = H, normal mode
$\overline{\text{ASEMD0}}$ = L, ASE mode

2. In ASE mode, reset hold is enabled by driving the $\overline{\text{RESETP}}$ and $\overline{\text{TRST}}$ pins low for a constant cycle. In this state, the CPU does not start up, even if $\overline{\text{RESETP}}$ is driven high. When $\overline{\text{TRST}}$ is driven high, H-UDI operation is enabled, but the CPU does not start up. The reset hold state is cancelled by the following:
   - Another $\overline{\text{RESETP}}$ assert (power-on reset)
   - $\overline{\text{TRST}}$ reassert

26.4.3 TDO Output Timing

The timing for switching data output from the TDO in normal mode changes at the TCK falling edge. This is a timing defined by the JTAG standard. This timing for switching data output from the TDO varies if the emulator function is used in ASE mode. For details on the timing, refer to the emulator's user's manual.

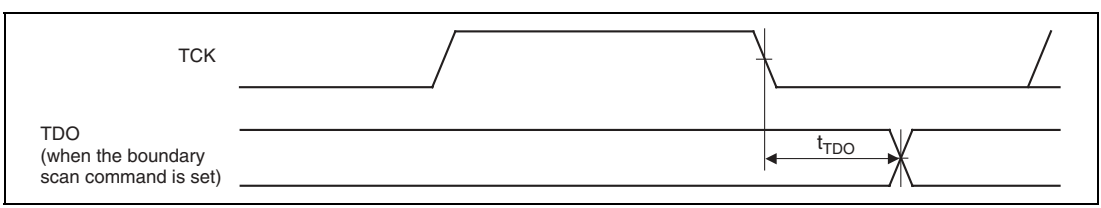


Figure 26.3 H-UDI Data Transfer Timing

26.5 Boundary Scan

When $\overline{\text{ASEMD0}}$ is driven high, the H-UDI pins are function as pins for the JTAG. All boundary scan cells on this LSI are connected as a single shift register with an input from the TDI pin and an output from the TDO pin to form the boundary scan register (SDBSR). The TDO and TDI pins of all LSIs that support the boundary scan function on the board are connected as a single shift register to form the boundary scan register. This allows wiring between the LSIs on the board and the mounted conditions of LSIs to be tested.

26.5.1 Supported Instructions

This LSI supports the three essential instructions defined in the JTAG standard (BYPASS, SAMPLE/PRELOAD, and EXTEST) and an optional instruction (IDCODE).

1. BYPASS

   The BYPASS instruction is an essential standard instruction that operates the bypass register. This instruction shortens the shift path to speed up serial data transfer involving other LSIs on the printed circuit board. This LSI's system circuits are not affected by execution of this instruction. The instruction code is B'111.

2. SAMPLE/PRELOAD

   The SAMPLE/PRELOAD instruction inputs values from this LSI's internal circuitry to the boundary scan register, outputs values from the scan path, and loads data onto the scan path. When this instruction is executing, this LSI's input pin signals are transmitted directly to the internal circuitry, and internal circuit values are directly output externally from the output pins. This LSI's system circuits are not affected by execution of this instruction. The instruction code is B'001.

   In a SAMPLE operation, a snapshot of a value to be transferred from an input pin to the internal circuitry, or a value to be transferred from the internal circuitry to an output pin, is latched into the boundary scan register and read from the scan path. Snapshot latching is performed in synchronization with the rising of TCK in the Capture-DR state. Snapshot latching does not affect normal operation of this LSI.

   In a PRELOAD operation, an initial value is set in the parallel output latch of the boundary scan register from the scan path prior to the EXTEST instruction. Without a PRELOAD operation, when the EXTEST instruction was executed an undefined value would be output from the output pin until completion of the initial scan sequence (transfer to the output latch) (with the EXTEST instruction, the parallel output latch value is constantly output to the output pin).

3. EXTEST

   This instruction is provided to test external circuitry when this LSI is mounted on a printed circuit board. When this instruction is executed, output pins are used to output test data (previously set by the SAMPLE/PRELOAD instruction) from the boundary scan register to the printed circuit board, and input pins are used to latch test results into the boundary scan register from the printed circuit board. If testing is carried out by using the EXTEST instruction N times, the Nth test data is scanned-in when test data (N-1) is scanned out.

   Data loaded into the output pin boundary scan register in the Capture-DR state is not used for external circuit testing (it is replaced by a shift operation).

   The instruction code is B'000.

4. IDCODE

   The H-UDI pins can be entered in IDCODE mode which is defined by the JTAG by setting the command to SDIR from the H-UDI pins. When the H-UDI is initialized (when the $\overline{\text{TRST}}$ pin is asserted or the TAP controller is transited to the Test-Logic-Reset state), IDCODE mode is entered.

   The instruction code is B'100.

26.5.2 Cautions

1. Boundary scan mode does not cover clock-related signals: EXTAL2, XTAL2, EXTAL, XTAL.
2. Boundary scan mode does not cover the reset-related signal: $\overline{\text{RESETP}}$.
3. Boundary scan mode does not cover H-UDI-related signals: TCK, TDI, TDO, TMS, $\overline{\text{TRST}}$, $\overline{\text{ASEMD0}}$.
4. Boundary scan mode does not cover some Bluetooth (BT) related signals: RDI_TXTRDATA, RDI_RXBDW_OUT, RDI_REFCLK_IN, RDI_CTRL3, RDI_CTRL4, RFSEL, RCI_SPI_TXRX, RCI_SPI_CLK, RCI_SPI_ENB, RREF.
5. Boundary scan mode does not cover the test-related signal: $\overline{\text{TEST_REG}}$, $\overline{\text{TEST}}$.
6. Boundary scan mode does not cover analog signals: USB_P, USB_N, DA0, DA1.
7. When the EXTEST command is set, fix the $\overline{\text{RESETP}}$ pin low.
8. While boundary scan is being executed, fix the $\overline{\text{ASEMD0}}$ pin high.

26.6 Usage Notes

1. An H-UDI command, once set, will not be modified as long as another command is not re-issued from the H-UDI. If the same command is given continuously, the command must be set after a command (BYPASS, etc.) that does not affect chip operations is once set.
2. Because chip operations are suspended in standby mode, H-UDI commands are not accepted. To maintain the TAP controller state before and after standby mode, TCK must be kept high during the transition to standby mode.
3. The H-UDI is used for emulator connection. Therefore, H-UDI functions cannot be used when using an emulator.
4. Usage of the $\overline{\text{TRST}}$ pin as a boundary scan function is optional. In this LSI, the TAP controller must be initialized simultaneously by the $\overline{\text{TRST}}$ pin at a power-on reset. Note however that in other LSIs, a power-on reset function may be available instead of the $\overline{\text{TRST}}$ pin or initialization of the TAP controller at a power-on reset may not be required. Take the following points in consideration when applying a power-on reset signal to the $\overline{\text{TRST}}$ pin.
   - A reset signal must always be applied when the power is turned on.
   - The power-on reset circuit must be separated from the LSI so that the $\overline{\text{TRST}}$ signal of the board tester does not affect the LSI operation. Resetting the TAP controller during board testing may generate a $\overline{\text{TRST}}$ signal asynchronously. If the power-on reset circuit is not separated in this case, the generated $\overline{\text{TRST}}$ signal will likely affect LSI operation during board testing (e.g. system reset occurs asynchronously).
   - Due to the opposite reason as described above, the power-on reset circuit must be separated from the LSI so that the LSI's system reset does not affect the $\overline{\text{TRST}}$ signal of the board tester. During board testing, the board tester controls the system reset signal via the boundary scans to examine the connection of the LSI system reset signal. If the power-on reset circuit is not separated in this case, the above operation resets the TAP controller, and testing may not be performed correctly.

Figure 26.4 shows an example of circuit connection.

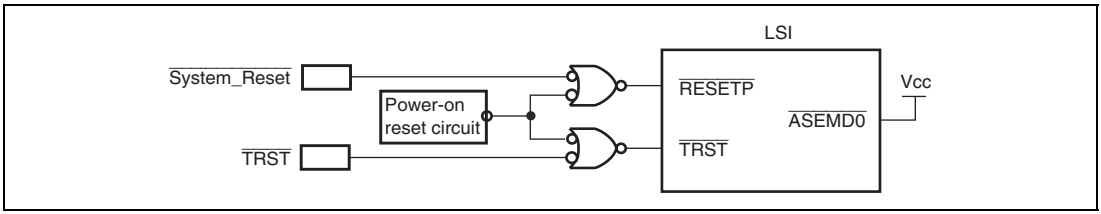


Figure 26.4 Example of Connecting Reset Signals without Mutual Interference

Section 27 List of Registers

The address map gives information on the on-chip I/O registers and is configured as described below.

1. Register Addresses (by functional module, in order of the corresponding section numbers)
   - Descriptions by functional module, in order of the corresponding section numbers
   - Access to reserved addresses which are not described in this list is disabled.
   - When registers consist of 16 or 32 bits, the addresses of the MSBs are given, on the presumption of a big-endian system.
2. Register Bits
   - Bit configurations of the registers are described in the same order as the Register Addresses (by functional module, in order of the corresponding section numbers).
   - Reserved bits are indicated by — in the bit name.
   - No entry in the bit-name column indicates that the whole register is allocated as a counter or for holding data.
   - When registers consist of 16 or 32 bits, bits are described from the MSB side. The order in which bytes are described is on the presumption of a big-endian system
3. Register States in Each Operating Mode
   - Register states are described in the same order as the Register Addresses (by functional module, in order of the corresponding section numbers).
   - For the initial state of each bit, refer to the description of the register in the corresponding section.
   - The register states described are for the basic operating modes. If there is a specific reset for an on-chip module, refer to the section on that on-chip module.

27.1 Register Addresses (by functional module, in order of the corresponding section numbers)

Entries under Access size indicates numbers of bits.

Note: Access to undefined or reserved addresses is prohibited. Since operation or continued operation is not guaranteed when these registers are accessed, do not attempt such access.

Register Name	Abbreviation	Number of Bits	Address	Module	Access Size
Interrupt event register 2	INTEVT2	32	H'A400 0000	Exception handling	32
TRAPA exception register	TRA	32	H'FFFF FFD0		32
Exception event register	EXPEVT	32	H'FFFF FFD4		32
Exception address register	TEA	32	H'FFFF FFFC		32
Cache control register 1	CCR1	32	H'FFFF FFEC	Cache	32
Cache control register 2	CCR2	32	H'A400 00B0		32
Interrupt priority register A	IPRA	16	H'A414 FEE2	INTC	16
Interrupt priority register B	IPRB	16	H'A414 FEE4		16
Interrupt priority register C	IPRC	16	H'A414 0016		16
Interrupt priority register D	IPRD	16	H'A414 0018		16
Interrupt priority register E	IPRE	16	H'A414 001A		16
Interrupt priority register F	IPRF	16	H'A408 0000		16
Interrupt priority register G	IPRG	16	H'A408 0002		16
Interrupt priority register H	IPRH	16	H'A408 0004		16
Interrupt mask register 0	IMR0	8	H'A408 0040		8
Interrupt mask register 1	IMR1	8	H'A408 0042		8
Interrupt mask register 4	IMR4	8	H'A408 0048		8
Interrupt mask register 5	IMR5	8	H'A408 004A		8
Interrupt mask register 6	IMR6	8	H'A408 004C		8
Interrupt mask register 9	IMR9	8	H'A408 0052		8
Interrupt mask clear register 0	IMCR0	8	H'A408 0060		8
Interrupt mask clear register 1	IMCR1	8	H'A408 0062		8
Interrupt mask clear register 4	IMCR4	8	H'A408 0068		8
Interrupt mask clear register 5	IMCR5	8	H'A408 006A		8
Interrupt mask clear register 6	IMCR6	8	H'A408 006C		8
Interrupt mask clear register 9	IMCR9	8	H'A408 0072		8

Register Name	Abbreviation	Number of Bits	Address	Module	Access Size
Interrupt control register 0	ICR0	16	H'A414 FEE0	INTC	16
Interrupt control register 1	ICR1	16	H'A414 0010		16
Interrupt control register 2	ICR2	16	H'A414 0012		16
Interrupt request register 0	IRR0	8	H'A414 0004		8
Common control register	CMNCR	32	H'A4FD 0000	BSC	32
Bus control register for CS0 space	CS0BCR	32	H'A4FD 0004		32
Bus control register for CS3 space	CS3BCR	32	H'A4FD 000C		32
Bus control register for CS4 space	CS4BCR	32	H'A4FD 0010		32
Wait control register for CS0 space	CS0WCR	32	H'A4FD 0024		32
Wait control register for CS3 space	CS3WCR	32	H'A4FD 002C		32
Wait control register for CS4 space	CS4WCR	32	H'A4FD 0030		32
SDRAM control register	SDCR	32	H'A4FD 0044		32
Refresh timer control/status register	RTCSR	32	H'A4FD 0048		32
Refresh timer counter	RTCNT	32	H'A4FD 004C		32
Refresh timer timer constant register	RTCOR	32	H'A4FD 0050		32
Reset wait counter	RWTCNT	32	H'A4FD 0054		32
SDRAM mode register for CS3 space	SDMR3	—	H'A4FD 5xxx*[1]		16
DMA source address register_0	SAR_0	32	H'A401 0020	DMAC	16/32
DMA destination address register_0	DAR_0	32	H'A401 0024		16/32
DMA transfer count register_0	DMATCR_0	32	H'A401 0028		16/32
DMA channel control register_0	CHCR_0	32	H'A401 002C		8/16/32
DMA initial address register_0	IAR_0	32	H'A401 0000		16/32
DMA source address register_1	SAR_1	32	H'A401 0030		16/32
DMA destination address register_1	DAR_1	32	H'A401 0034		16/32
DMA transfer count register_1	DMATCR_1	32	H'A401 0038		16/32
DMA channel control register_1	CHCR_1	32	H'A401 003C		8/16/32
DMA initial address register_1	IAR_1	32	H'A401 0004		16/32
DMA source address register_2	SAR_2	32	H'A401 0040		16/32
DMA destination address register_2	DAR_2	32	H'A401 0044		16/32
DMA transfer count register_2	DMATCR_2	32	H'A401 0048		16/32
DMA channel control register_2	CHCR_2	32	H'A401 004C		8/16/32
DMA initial address register_2	IAR_2	32	H'A401 0008		16/32
DMA source address register_3	SAR_3	32	H'A401 0050		16/32

Register Name	Abbreviation	Number of Bits	Address	Module	Access Size
DMA destination address register_3	DAR_3	32	H'A401 0054	DMAC	16/32
DMA transfer count register_3	DMATCR_3	32	H'A401 0058		16/32
DMA channel control register_3	CHCR_3	32	H'A401 005C		8/16/32
DMA initial address register_3	IAR_3	32	H'A401 000C		16/32
DMA operation register	DMAOR	32	H'A401 0060		8/16/32
DMA extended resource selector 0	DMARS0	16	H'A409 0000		16
DMA extended resource selector 1	DMARS1	16	H'A409 0004		16
Frequency control register*[3]	FRQCR	16	H'A415 FF80	CPG	16
Watchdog timer counter	WTCNT	8	H'A415 FF84	WDT	8/16*[2]
Watchdog timer control/status register	WTCSR	8	H'A415 FF86		8/16*[2]
Standby control register*[3]	STBCR	8	H'A415 FF82	Power-down modes	8
Standby control register 2	STBCR2	8	H'A415 FF88		8
Standby control register 3	STBCR3	8	H'A40A 0000		8
Standby control register 4	STBCR4	8	H'A40A 0004		8
Memory clock control register	MCCR	8	H'A40A0010		8
Timer start register	TSTR	8	H'A412 FE92	TMU	8
Timer constant register_0	TCOR_0	32	H'A412 FE94		32
Timer counter_0	TCNT_0	32	H'A412 FE98		32
Timer control register_0	TCR_0	16	H'A412 FE9C		16
Timer constant register_1	TCOR_1	32	H'A412 FEA0		32
Timer counter_1	TCNT_1	32	H'A412 FEA4		32
Timer control register_1	TCR_1	16	H'A412 FEA8		16
Timer constant register_2	TCOR_2	32	H'A412 FEAC		32
Timer counter_2	TCNT_2	32	H'A412 FEB0		32
Timer control register_2	TCR_2	16	H'A412 FEB4		16
Mode register	SIMDR	16	H'A442 0000	SIOF	16
Clock select register	SISCR	16	H'A442 0002		16
Transmit data assign register	SITDAR	16	H'A442 0004		16
Receive data assign register	SIRDAR	16	H'A442 0006		16
Control data assign register	SICDAR	16	H'A442 0008		16
Control register	SICTR	16	H'A442 000C		16
FIFO control register	SIFCTR	16	H'A442 0010		16
Status register	SISTR	16	H'A442 0014		16

Register Name	Abbreviation	Number of Bits	Address	Module	Access Size
Interrupt enable register	SIIER	16	H'A442 0016	SIOF	16
Transmit data register	SITDR	32	H'A442 0020		32
Receive data register	SIRDR	32	H'A442 0024		32
Transmit control data register	SITCR	32	H'A442 0028		32
Receive control data register	SIRCR	32	H'A442 002C		32
SPI control register	SPICR	16	H'A4420030		16
Serial mode register_0	SCSMR_0	16	H'A443 0000	SCIF_0	16
Bit rate register_0	SCBRR_0	8	H'A443 0004	(Channel 0)	8
Serial control register_0	SCSCR_0	16	H'A443 0008		16
Transmit data stop register_0	SCTDSR_0	8	H'A443 000C		8
FIFO error count register_0	SCFER_0	16	H'A443 0010		16
Serial status register_0	SCSSR_0	16	H'A443 0014		16
FIFO control register_0	SCFCR_0	16	H'A443 0018		16
FIFO data count register_0	SCFDR_0	16	H'A443 001C		16
Transmit FIFO data register_0	SCFTDR_0	8	H'A443 0020		8
Receive FIFO data register_0	SCFRDR_0	8	H'A443 0024		8
Serial mode register_1	SCSMR_1	16	H'A445 0000	SCIF_1	16
Bit rate register_1	SCBRR_1	8	H'A445 0004	(Channel 1)	8
Serial control register_1	SCSCR_1	16	H'A445 0008		16
Transmit data stop register_1	SCTDSR_1	8	H'A445 000C		8
FIFO error count register_1	SCFER_1	16	H'A445 0010		16
Serial status register_1	SCSSR_1	16	H'A445 0014		16
FIFO control register_1	SCFCR_1	16	H'A445 0018		16
FIFO data count register_1	SCFDR_1	16	H'A445 001C		16
Transmit FIFO data register_1	SCFTDR_1	8	H'A445 0020		8
Receive FIFO data register_1	SCFRDR_1	8	H'A445 0024		8
EXCPG control register	EXCPGCR	8	H'A447 0000	USBPM	8
HcRevision register	HREVR	32	H'A449 0000	USBH	32
HcContorol register	HCTLR	32	H'A449 0004		32
HcCommandStatus register	HCSR	32	H'A449 0008		32
HcInterruptStatus register	HISR	32	H'A449 000C		32
HcInterruptEnable register	HIER	32	H'A449 0010		32
HcInterruptDisable register	HIDR	32	H'A449 0014		32

Register Name	Abbreviation	Number of Bits	Address	Module	Access Size
HcHCCA register	HHCCAR	32	H'A449 0018	USBH	32
HcPeriodCurrentED register	HPCEDR	32	H'A449 001C		32
HcControlHeadED register	HCHEDR	32	H'A449 0020		32
HcControlCurrentED register	HCCEDR	32	H'A449 0024		32
HcBulkHeadED register	HBHEDR	32	H'A449 0028		32
HcBulkCurrentED register	HBCEDR	32	H'A449 002C		32
HcDoneHeadED register	HDHEDR	32	H'A449 0030		32
HcFmInterval register	HFIR	32	H'A449 0034		32
HcFmRemaining register	HFRR	32	H'A449 0038		32
HcFmNumber register	HFNR	32	H'A449 003C		32
HcPeriodicStart register	HPSR	32	H'A449 0040		32
HcLSThreshold register	HLSTR	32	H'A449 0044		32
HcRhDescriptor A register	HRDRA	32	H'A449 0048		32
HcRhDescriptor B register	HRDRB	32	H'A449 004C		32
HcRhStatus register	HRSR	32	H'A449 0050		32
HcRhPortStatus1 register	HRPSR	32	H'A449 0054		32
Interrupt flag register 0	IFR0	32	H'A448 0000	USBF	32
Interrupt select register 0	ISR0	32	H'A448 0010		32
Interrupt enable register 0	IER0	32	H'A448 0020		32
EP0i data register	EPDR0i	64B*4	H'A448 0030		8
EP0o data register	EPDR0o	64B*4	H'A448 0034		8
EP0s data register	EPDR0s	8B*4	H'A448 0038		8
EP1 data register	EPDR1	16B*4	H'A448 003C		8
EP2i data register	EPDR2i	128B*4	H'A448 0040		8
EP2o data register	EPDR2o	128B*4	H'A448 0044		8
EP3i data register	EPDR3i	128B*4	H'A448 004C		8
EP3o data register	EPDR3o	120B*4	H'A448 0050		8
EP4 data register	EPDR4	16B*4	H'A448 0054		8
EP5 data register	EPDR5	64B*4	H'A448 0058		8
EP6 data register	EPDR6	64B*4	H'A448 005C		8
EP0o receive data size register	EPSZ0o	8	H'A448 0080		8
EP2o receive data size register	EPSZ2o	8	H'A448 0084		8
EP3o receive data size register	EPSZ3o	8	H'A448 0088		8

Register Name	Abbreviation	Number of Bits	Address	Module	Access Size
EP6 receive data size register	EPSZ6	8	H'A448 008C	USBF	8
Trigger register	TRG	16	H'A448 00A0		16
Data status register	DASTS	8	H'A448 00A4		8
FIFO clear register	FCLR	16	H'A448 00A8		16
DMA transfer setting register	DMA	8	H'A448 00AC		8
Endpoint stall register	EPSTL	16	H'A448 00B0		32*[5]
Configuration value register	CVR	8	H'A448 00B4		8
Time stamp register	TSR	16	H'A448 00B8		16
Control register	CTLR	8	H'A448 00BC		8
Endpoint information register	EPIR	8	H'A448 00C0		8
D/A data register0	DADR_0	8	H'A440 0000	DAC	8
D/A data register1	DADR_1	8	H'A440 0001		8
D/A control register	DACR	8	H'A440 0002		8
Port A data register	PADR	8	H'A405 0120	I/O port	8
Port B data register	PBDR	8	H'A405 0122		8
Port D data register	PDDR	8	H'A405 0126		8
Port E data register	PEDR	8	H'A405 0128		8
Port G data register	PGDR	8	H'A405 012C		8
Port H data register	PHDR	8	H'A405 012E		8
Port A control register	PACR	16	H'A405 0100	PFC	16
Port B control register	PBCR	16	H'A405 0102		16
Port D control register	PDCR	16	H'A405 0106		16
Port E control register	PECR	16	H'A405 0108		16
Port G control register	PGCR	16	H'A405 010C		16
Port H control register	PHCR	16	H'A405 010E		16
Pin select register A	PSELA	16	H'A405 0140		16
I/O buffer Hi-Z control register A	HIZCRA	16	H'A405 0146		16
Noise Canceller control register	NCCR	16	H'A405 014C		16
Break data register B	BDRB	32	H'FFFF FF90	UBC	32
Break data mask register B	BDMRB	32	H'FFFF FF94		32
Break control register	BRCR	32	H'FFFF FF98		32
Execution count break register	BETR	16	H'FFFF FF9C		16
Break address register B	BARB	32	H'FFFF FFA0		32

Register Name	Abbreviation	Number of Bits	Address	Module	Access Size
Break address mask register B	BAMRB	32	H'FFFF FFA4	UBC	32
Break bus cycle register B	BBRB	16	H'FFFF FFA8		16
Branch source register	BRSR	32	H'FFFF FFAC		32
Break address register A	BARA	32	H'FFFF FFB0		32
Break address mask register A	BAMRA	32	H'FFFF FFB4		32
Break bus cycle register A	BBRA	16	H'FFFF FFB8		16
Branch destination register	BRDR	32	H'FFFF FFBC		32
Instruction register*[6]	SDIR	16	H'A410 0200	H-UDI	16
ID register	SDID/SDIDH	16	H'A410 0214		16
ID register	SDID/SDIDL	16	H'A410 0216		16

Notes: 1. Controls accessing of the SDRAM mode register. XXX depends on the setting value.
2. 8 bits when reading and 16 bits when writing.
3. If the SLEEP instruction is issued immediately after data is written to, the register value may not be reflected correctly. Before executing the SLEEP instruction, read the register.
4. The number of bits of each endpoint data register in USBF indicates the max value of FIFO buffer.
5. The endpoint stall register (EPSTL) must be accessed in 32 bits and the lower 2 bytes become valid.
6. 3-bit register when accessing from H-UDI interface.

27.2 Register Bits

Register addresses and 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 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
INTEVT2	—	—	—	—	—	—	—	—	Exception handling
	—	—	—	—	—	—	—	—	
	—	—	—	—	INTEVT2	INTEVT2	INTEVT2	INTEVT2	
	INTEVT2	INTEVT2	INTEVT2	INTEVT2	INTEVT2	INTEVT2	INTEVT2	INTEVT2	
TRA	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	TRA	TRA	
	TRA	TRA	TRA	TRA	TRA	TRA	—	—	
EXPEVT	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	EXPEVT	EXPEVT	EXPEVT	EXPEVT	
	EXPEVT	EXPEVT	EXPEVT	EXPEVT	EXPEVT	EXPEVT	EXPEVT	EXPEVT	
TEA	TEA	TEA	TEA	TEA	TEA	TEA	TEA	TEA	
	TEA	TEA	TEA	TEA	TEA	TEA	TEA	TEA	
	TEA	TEA	TEA	TEA	TEA	TEA	TEA	TEA	
	TEA	TEA	TEA	TEA	TEA	TEA	TEA	TEA	
CCR1	—	—	—	—	—	—	—	—	Cache
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	CF	CB	WT	CE	
CCR2	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	W3LOAD	W3LOCK	
	—	—	—	—	—	—	W2LOAD	W2LOCK	
IPRA	IPR15	IPR14	IPR13	IPR12	IPR11	IPR10	IPR9	IPR8	INTC
	IPR7	IPR6	IPR5	IPR4	—	—	—	—	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
IPRB	IPR15	IPR14	IPR13	IPR12	—	—	—	—	INTC
	—	—	—	—	—	—	—	—	
IPRC	IPR15	IPR14	IPR13	IPR12	IPR11	IPR10	IPR9	IPR8	
	IPR7	IPR6	IPR5	IPR4	IPR3	IPR2	IPR1	IPR0	
IPRD	IPR15	IPR14	IPR13	IPR12	—	—	—	—	
	—	—	—	—	IPR3	IPR2	IPR1	IPR0	
IPRE	IPR15	IPR14	IPR13	IPR12	—	—	—	—	
	—	—	—	—	—	—	—	—	
IPRF	—	—	—	—	—	—	—	—	
	IPR7	IPR6	IPR5	IPR4	IPR3	IPR2	IPR1	IPR0	
IPRG	IPR15	IPR14	IPR13	IPR12	IPR11	IPR10	IPR9	IPR8	
	—	—	—	—	—	—	—	—	
IPRH	—	—	—	—	IPR11	IPR10	IPR9	IPR8	
	—	—	—	—	—	—	—	—	
IMR0	IM7	—	—	IM4	IM3	IM2	IM1	IM0	
IMR1	—	—	—	—	IM3	IM2	IM1	IM0	
IMR4	—	IM6	IM5	IM4	IM3	—	—	—	
IMR5	—	—	—	—	—	—	IM1	IM0	
IMR6	—	—	—	—	—	—	IM1	—	
IMR9	—	—	—	—	—	IM2	IM1	IM0	
IMCR0	IMC7	—	—	IMC4	IMC3	IMC2	IMC1	IMC0	
IMCR1	—	—	—	—	IMC3	IMC2	IMC1	IMC0	
IMCR4	—	IMC6	IMC5	IMC4	IMC3	—	—	—	
IMCR5	—	—	—	—	—	—	IMC1	IMC0	
IMCR6	—	—	—	—	—	—	IMC1	—	
IMCR9	—	—	—	—	—	IMC2	IMC1	IMC0	
ICR0	NMIL	—	—	—	—	—	—	NMIE	
	—	—	—	—	—	—	—	—	
ICR1	—	IRQE	—	—	—	—	IRQ41S	IRQ40S	
	IRQ31S	IRQ30S	IRQ21S	IRQ20S	IRQ11S	IRQ10S	IRQ01S	IRQ00S	
ICR2	—	—	—	—	—	—	—	—	
	—	—	—	—	IRQ71S	IRQ70S	—	—	
IRR0	IRQ7R	—	—	IRQ4R	IRQ3R	IRQ2R	IRQ1R	IRQ0R	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
CMNCR	—	—	—	—	—	—	—	—	BSC
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	DMAIW1	DMAIW0	DMAIWA	—	ENDIAN	—	HIZMEM	HIZCNT	
CS0BCR	—	—	IWW1	IWW0	—	IWRWD1	IWRWD0	—	
	IWRWS1	IWRWS0	—	IWRRD1	IWRRD0	—	IWRRS1	IWRRS0	
	—	TYPE2	TYPE1	TYPE0	—	BSZ1	BSZ0	—	
	—	—	—	—	—	—	—	—	
CS3BCR	—	—	IWW1	IWW0	—	IWRWD1	IWRWD0	—	
	IWRWS1	IWRWS0	—	IWRRD1	IWRRD0	—	IWRRS1	IWRRS0	
	—	TYPE2	TYPE1	TYPE0	—	BSZ1	BSZ0	—	
	—	—	—	—	—	—	—	—	
CS4BCR	—	—	IWW1	IWW0	—	IWRWD1	IWRWD0	—	
	IWRWS1	IWRWS0	—	IWRRD1	IWRRD0	—	IWRRS1	IWRRS0	
	—	TYPE2	TYPE1	TYPE0	—	BSZ1	BSZ0	—	
	—	—	—	—	—	—	—	—	
CS0WCR	—	—	—	—	—	—	—	—	
(Other than	—	—	—	—	—	—	—	—	
burst ROM)	—	—	—	SW1	SW0	WR3	WR2	WR1	
	WR0	WM	—	—	—	—	HW1	HW0	
CS0WCR	—	—	—	—	—	—	—	—	
(Burst ROM)	—	—	—	—	—	—	BW1	BW0	
	—	—	—	—	—	W3	W2	W1	
	W0	WM	—	—	—	—	—	—	
CS3WCR	—	—	—	—	—	—	—	—	
(Other than	—	—	—	BAS	—	—	—	—	
SDRAM)	—	—	—	—	—	WR3	WR2	WR1	
	WR0	WM	—	—	—	—	—	—	
CS3WCR	—	—	—	—	—	—	—	—	
(SDRAM)	—	—	—	—	—	—	BW1	BW0	
	—	TRP1	TRP0	—	TRCD1	TRCD0	—	A3CL1	
	A3CL0	—	—	TRWL1	TRWL0	—	TRC1	TRC0	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
CS4WCR	—	—	—	—	—	—	—	—	BSC
(Other than burst ROM)	—	—	—	BAS	—	WW2	WW1	WW0	
	—	—	—	SW1	SW0	WR3	WR2	WR1	
	WR0	WM	—	—	—	—	HW1	HW0	
CS4WCR	—	—	—	—	—	—	—	—	
(Burst ROM)	—	—	—	—	—	—	BW1	BW0	
	—	—	—	SW1	SW0	W3	W2	W1	
	W0	WM	—	—	—	—	HW1	HW0	
SDCR	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	SLOW	RFSH	RMODE	PDOWN	BACTV	
	—	—	—	A3ROW1	A3ROW0	—	A3COL1	A3COL0	
RTCSR	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	CMF	—	CKS2	CKS1	CKS0	RRC2	RRC1	RRC0	
RTCNT	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
RTCOR	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
RWTCNT	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
SDMR3									

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
SAR0									DMAC
DAR0									
DMATCR0									
CHCR0	—	—	—	—	—	RPT	RAS	DA	
	DO	TL	—	—	—	—	AM	AL	
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
	DL	DS	TB	TS1	TS0	IE	TE	DE	
IAR_0									
SAR_1									
DAR_1									
DMATCR_1									

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
CHCR_1	—	—	—	—	—	RPT	RAS	DA	DMAC
	DO	TL	—	—	—	—	AM	AL	
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
	DL	DS	TB	TS1	TS0	IE	TE	DE	
IAR_1									
SAR_2									
DAR_2									
DMATCR_2									
CHCR_2	—	—	—	—	—	RPT	RAS	DA	
	DO	TL	—	—	—	—	AM	AL	
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
	DL	DS	TB	TS1	TS0	IE	TE	DE	
IAR_2									
SAR_3									

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
DAR_3									DMAC
DMATCR_3									
CHCR_3	—	—	—	—	—	RPT	RAS	DA	
	DO	TL	—	—	—	—	AM	AL	
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
	DL	DS	TB	TS1	TS0	IE	TE	DE	
IAR_3									
DMAOR	—	—	CMS1	CMS0	—	—	PR1	PR0	
	—	—	—	—	—	AE	NMIF	DME	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
DMARS0	C1MID5	C1MID4	C1MID3	C1MID2	C1MID1	C1MID0	C1RID1	C1RID0	
	C0MID5	C0MID4	C0MID3	C0MID2	C0MID1	C0MID0	C0RID1	C0RID0	
DMARS1	C3MID5	C3MID4	C3MID3	C3MID2	C3MID1	C3MID0	C3RID1	C3RID0	
	C2MID5	C2MID4	C2MID3	C2MID2	C2MID1	C2MID0	C2RID1	C2RID0	
FRQCR	—	—	—	CKOEN	—	STC2	STC1	STC0	CPG
	—	IFC2	IFC1	IFC0	—	PFC2	PFC1	PFC0	
WTCNT									WDT
WTCSR	TME	WT/$\overline{IT}$	RSTS	WOVF	IOVF	CKS2	CKS1	CKS0	
STBCR	STBY	—	—	STBXTL	—	MSTP2	—	—	Power-down mode
STBCR2	MSTP10	MSTP9	MSTP8	MSTP7	—	MSTP5	MSTP4	MSTP3	
STBCR3	MSTP37	—	—	—	—	—	—	—	
STBCR4	—	MSTP46	—	—	MSTP43	MSTP42	MSTP41	MSTP40	
MCCR	—	—	—	—	—	UMCLKC	XYMCLKC	—	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
TSTR	—	—	—	—	—	STR2	STR1	STR0	TMU
TCOR_0									
TCNT_0									
TCR_0	—	—	—	—	—	—	—	UNF	
	—	—	UNIE	—	—	—	TPSC1	TPSC0	
TCOR_1									
TCNT_1									
TCR_1	—	—	—	—	—	—	—	UNF	
	—	—	UNIE	—	—	—	TPSC1	TPSC0	
TCOR_2									
TCNT_2									
TCR_2	—	—	—	—	—	—	—	UNF	
	—	—	UNIE	—	—	—	TPSC1	TPSC0	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
SIMDR	TRMD1	TRMD0	SYNCAT	REDG	FL3	FL2	FL1	FL0	SIOF
	TXDIZ	RCIM	SYNCAC	SYNCDL	—	—	—	—	
SISCR	MSSEL	MSIMM	—	BRPS4	BRPS3	BRPS2	BRPS1	BRPS0	
	—	—	—	—	—	BRDV2	BRDV1	BRDV0	
SITDAR	TDLE	—	—	—	TDLA3	TDLA2	TDLA1	TDLA0	
	TDRE	TLREP	—	—	TDRA3	TDRA2	TDRA1	TDRA0	
SIRDAR	RDLE	—	—	—	RDLA3	RDLA2	RDLA1	RDLA0	
	RDRE	—	—	—	RDRA3	RDRA2	RDRA1	RDRA0	
SICDAR	CD0E	—	—	—	CD0A3	CD0A2	CD0A1	CD0A0	
	CD1E	—	—	—	CD1A3	CD1A2	CD1A1	CD1A0	
SICTR	SCKE	FSE	—	—	—	—	TXE	RXE	
	—	—	—	—	—	—	TXRST	RXRST	
SIFCTR	TFWM2	TFWM1	TFWM0	TFUA4	TFUA3	TFUA2	TFUA1	TFUA0	
	RFWM2	RFWM1	RFWM0	RFUA4	RFUA3	RFUA2	RFUA1	RFUA0	
SISTR	—	TCRDY	TFEMP	TDREQ	—	RCRDY	RFFUL	RDREQ	
	—	—	SAERR	FSERR	TFOVF	TFUDF	RFUDF	RFOVF	
SIIER	TDMAE	TCRDYE	TFEMPE	TDREQE	RDMAE	RCRDYE	RFFULE	RDREQE	
	—	—	SAERRE	FSERRE	TFOVFE	TFUDFE	RFUDFE	RFOVFE	
SITDR	SITDL15	SITDL14	SITDL13	SITDL12	SITDL11	SITDL10	SITDL9	SITDL8	
	SITDL7	SITDL6	SITDL5	SITDL4	SITDL3	SITDL2	SITDL1	SITDL0	
	SITDR15	SITDR14	SITDR13	SITDR12	SITDR11	SITDR10	SITDR9	SITDR8	
	SITDR7	SITDR6	SITDR5	SITDR4	SITDR3	SITDR2	SITDR1	SITDR0	
SIRDR	SIRDL15	SIRDL14	SIRDL13	SIRDL12	SIRDL11	SIRDL10	SIRDL9	SIRDL8	
	SIRDL7	SIRDL6	SIRDL5	SIRDL4	SIRDL3	SIRDL2	SIRDL1	SIRDL0	
	SIRDR15	SIRDR14	SIRDR13	SIRDR12	SIRDR11	SIRDR10	SIRDR9	SIRDR8	
	SIRDR7	SIRDR6	SIRDR5	SIRDR4	SIRDR3	SIRDR2	SIRDR1	SIRDR0	
SITCR	SITC015	SITC014	SITC013	SITC012	SITC011	SITC010	SITC09	SITC08	
	SITC07	SITC06	SITC05	SITC04	SITC03	SITC02	SITC01	SITC00	
	SITC115	SITC114	SITC113	SITC112	SITC111	SITC110	SITC19	SITC18	
	SITC17	SITC16	SITC15	SITC14	SITC13	SITC12	SITC11	SITC10	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
SIRCR	SIRC015	SIRC014	SIRC013	SIRC012	SIRC011	SIRC010	SIRC09	SIRC08	SIOF
	SIRC07	SIRC06	SIRC05	SIRC04	SIRC03	SIRC02	SIRC01	SIRC00	
	SIRC115	SIRC114	SIRC113	SIRC112	SIRC111	SIRC110	SIRC19	SIRC18	
	SIRC17	SIRC16	SIRC15	SIRC14	SIRC13	SIRC12	SIRC11	SIRC10	
SPICR	SPIM	—	CPHA	CPOL	—	SS2E	SS1E	SS0E	
	—	—	SSAST1	SSAST0	—	—	FLD1	FLD0	
SCSMR_0	—	—	—	—	—	SRC2	SRC1	SRC0	SCIF_0
	C/A	CHR	PE	O/E	STOP	—	CKS1	CKS0	
SCBRR_0	SCBRD7	SCBRD6	SCBRD5	SCBRD4	SCBRD3	SCBRD2	SCBRD1	SCBRD0	
SCSCR_0	TDRQE	RDRQE	—	—	TSIE	ERIE	BRIE	DRIE	
	TIE	RIE	TE	RE	—	—	CKE1	CKE0	
SCTDSR_0									
SCFER_0	—	—	PER5	PER4	PER3	PER2	PER1	PER0	
	—	—	FER5	FER4	FER3	FER2	FER1	FER0	
SCSSR_0	—	—	—	—	—	—	ORER	TSF	
	ER	TEND	TDFE	BRK	FER	PER	RDF	DR	
SCFCR_0	TSE	TCRST	—	—	—	RSTRG2	RSTRG1	RSTRG0	
	RTRG1	RTRG0	TTRG1	TTRG0	MCE	TFRST	RFRST	LOOP	
SCFDR_0	—	T6	T5	T4	T3	T2	T1	T0	
	—	R6	R5	R4	R3	R2	R1	R0	
SCFTDR_0	SCFTD7	SCFTD6	SCFTD5	SCFTD4	SCFTD3	SCFTD2	SCFTD1	SCFTD0	
SCFRDR_0	SCFRD7	SCFRD6	SCFRD5	SCFRD4	SCFRD3	SCFRD2	SCFRD1	SCFRD0	
SCSMR_1	—	—	—	—	—	SRC2	SRC1	SRC0	SCIF_1
	C/A	CHR	PE	O/E	STOP	—	CKS1	CKS0	
SCBRR_1	SCBRD7	SCBRD6	SCBRD5	SCBRD4	SCBRD3	SCBRD2	SCBRD1	SCBRD0	
SCSCR_1	TDRQE	RDRQE	—	—	TSIE	ERIE	BRIE	DRIE	
	TIE	RIE	TE	RE	—	—	CKE1	CKE0	
SCTDSR_1									
SCFER_1	—	—	PER5	PER4	PER3	PER2	PER1	PER0	
	—	—	FER5	FER4	FER3	FER2	FER1	FER0	
SCSSR_1	—	—	—	—	—	—	ORER	TSF	
	ER	TEND	TDFE	BRK	FER	PER	RDF	DR	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
SCFCR_1	TSE	TCRST	—	—	—	RSTRG2	RSTRG1	RSTRG0	SCIF_1
	RTRG1	RTRG0	TTRG1	TTRG0	MCE	TFRST	RFRST	LOOP	
SCFDR_1	—	T6	T5	T4	T3	T2	T1	T0	
	—	R6	R5	R4	R3	R2	R1	R0	
SCFTDR_1	SCFTD7	SCFTD6	SCFTD5	SCFTD4	SCFTD3	SCFTD2	SCFTD1	SCFTD0	
SCFRDR_1	SCFRD7	SCFRD6	SCFRD5	SCFRD4	SCFRD3	SCFRD2	SCFRD1	SCFRD0	
EXCPGCR	—	—	—	USBVALID	USBRESET	USBHSTP	USBFSTP	USBCLKSEL	USBPM
HREVR	—	—	—	—	—	—	—	—	USBH
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	REV7	REV6	REV5	REV4	REV3	REV2	REV1	REV0	
HCTLR	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	IR	
	HCFS1	HCFS0	BLE	CLE	IE	PLE	CBSR1	CBSR0	
HCSR	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	SOC1	SOC0	
	—	—	—	—	—	—	—	—	
	—	—	—	—	OCR	BLF	CLF	HCR	
HISR	—	OC	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	RHSC	FNO	UE	RD	SF	WDH	SO	
HIER	MIE	OC	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	RHSC	FNO	UE	RD	SF	WDH	SO	
HIDR	MIE	OC	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	RHSC	FNO	UE	RD	SF	WDH	SO	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
HHCCAR									USBH
	—	—	—	—	—	—	—	—	
HPCEDR									
					—	—	—	—	
HCHEDR									
					—	—	—	—	
HCCEDR									
					—	—	—	—	
HBHEDR									
					—	—	—	—	
HBCEDR									
					—	—	—	—	
HDHEDR									
					—	—	—	—	
HFIR	FIT	FSMPS14	FSMPS13	FSMPS12	FSMPS11	FSMPS10	FSMPS9	FSMPS8	
	FSMPS7	FSMPS6	FSMPS5	FSMPS4	FSMPS3	FSMPS2	FSMPS1	FSMPS0	
	—	—	FI13	FI12	FI11	FI10	FI9	FI8	
	FI7	FI6	FI5	FI4	FI3	FI2	FI1	FI0	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
HFRR	FRT	—	—	—	—	—	—	—	USBH
	—	—	—	—	—	—	—	—	
	—	—	FR13	FR12	FR11	FR10	FR9	FR8	
	FR7	FR6	FR5	FR4	FR3	FR2	FR1	FR0	
HFNR	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	FN15	FN14	FN13	FN12	FN11	FN10	FN9	FN8	
	FN7	FN6	FN5	FN4	FN3	FN2	FN1	FN0	
HPSR	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	PS13	PS12	PS11	PS10	PS9	PS8	
	PS7	PS6	PS5	PS4	PS3	PS2	PS1	PS0	
HLSTR	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	LST11	LST10	LST9	LST8	
	LST7	LST6	LST5	LST4	LST3	LST2	LST1	LST0	
HRDRA	POTPGT7	POTPGT6	POTPGT5	POTPGT4	POTPGT3	POTPGT2	POTPGT1	POTPGT0	
	—	—	—	—	—	—	—	—	
	—	—	—	NOCP	OCPM	DT	NPS	PSM	
	NDP7	NDP6	NDP5	NDP4	NDP3	NDP2	NDP1	NDP0	
HRDRB	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	PPCM	—	
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	DR	—	
HRSR	CRWE	—	—	—	—	—	—	—	
	—	—	—	—	—	—	OCIC	LPSC	
	DRWE	—	—	—	—	—	—	—	
	—	—	—	—	—	—	OCI	LPS	
HRPSR	—	—	—	—	—	—	—	—	
	—	—	—	PRSC	OCIC	PSSC	PESC	CSC	
	—	—	—	—	—	—	LSDA	PPS	
	—	—	—	PRS	POCI	PSS	PES	CCS	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
IFR0	—	BSRT	SETUP TS	VBUSMN	VBUSF	SURSS	SURES	CFDN	USBF
	SOF	SETC	SETI	EP6 FULL	EP5 EMPTY	EP5 TR	EP4 TR	EP4 TS	
	—	—	EP3o TF	EP3o TS	EP3I TR	EP3I TS	—	—	
	EP2o FULL	EP2i EMPTY	EP2I TR	EP1 TR	EP1 TS	EP0o TS	EP0I TR	EP0I TS	
ISR0	—	BSRT IS	SETUP TS IS	—	VBUSF IS	—	SURES IS	CFDN IS	
	SOF IS	SETC IS	SETI IS	EP6 FULL IS	EP5 EMPTY IS	EP5TR IS	EP4TR IS	EP4TS IS	
	—	—	EP3o TF IS	EP3o TS IS	EP3I TR IS	EP3I TS IS	—	—	
	EP2o FULL IS	EP2I EMPTY IS	EP2I TR IS	EP1 TR IS	EP1 TS IS	EP0o TS IS	EP0I TR IS	EP0I TS IS	
IER0	—	BSRT IE	SETUP TS IE	—	VBUSF IE	—	SURES IE	CFDN IE	
	SOF IE	SETC IE	SETI IE	EP6 FULL IE	EP5 EMPTY IE	EP5 TR IE	EP4 TR IE	EP4 TS IE	
	—	—	EP3o TF IE	EP3o TS IE	EP3I TR IE	EP3I TS IE	—	—	
	EP2o FULL IE	EP2I EMPTY IE	EP2I TR IE	EP1 TR IE	EP1 TS IE	EP0o TS IE	EP0i TR IE	EP0i TS IE	
EPDR0i	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR0o	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR0s	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR1	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR2i	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR2o	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR3i	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR3o	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR4	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR5	D7	D6	D5	D4	D3	D2	D1	D0	
EPDR6	D7	D6	D5	D4	D3	D2	D1	D0	
EPSZ0o	D7	D6	D5	D4	D3	D2	D1	D0	
EPSZ2o	D7	D6	D5	D4	D3	D2	D1	D0	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
EPSZ3o	D7	D6	D5	D4	D3	D2	D1	D0	USBF
EPSZ6	D7	D6	D5	D4	D3	D2	D1	D0	
TRG	—	EP6 RDFN	EP5 PKTE	EP4 PKTE	—	—	—	—	
	—	EP2o RDFN	EP2i PKTE	EP1 PKTE	—	EP0s RDFN	EP0o RDFN	EP0i PKTE	
DASTS	—	EP5 DE	EP4 DE	—	EP2i DE	EP1 DE	—	EP0i DE	
FCLR	—	EP6 CLR	EP5 CLR	EP4 CLR	—	EP3o CLR	EP3i CLR	—	
	—	EP2o CLR	EP2i CLR	EP1 CLR	—	—	EP0o CLR	EP0i CLR	
DMA	EP6 DMAE	EP6 DMAS	EP5 DMAE	EP5 DMAS	EP2o DMAE	EP2o DMAS	EP2i DMAE	EP2i DMAS	
EPSTL	—	—	—	—	—	EP6STL	EP5STL	EP4STL	
	—	EP3o STL	EP3i STL	—	EP2o STL	EP2i STL	EP1 STL	EP0 STL	
CVR	CNFV	INTV3	INTV2	INTV1	INTV0	ALTV2	ALTV1	ALTV0	
TSR	—	—	—	—	—	D10	D9	D8	
	D7	D6	D5	D4	D3	D2	D1	D0	
CTLR	—	—	PULLPE	RWUPS	RSME	PWMD	ASCE	SOFME	
EPIR	D7	D6	D5	D4	D3	D2	D1	D0	
DADR_0									DAC
DADR_1									
DACR	DAOE1	DAOE0	—	—	—	—	—	—	
PADR	—	PA6DT	PA5DT	PA4DT	PA3DT	PA2DT	PA1DT	PA0DT	I/O port
PBDR	PB7DT	PB6DT	PB5DT	PB4DT	PB3DT	PB2DT	PB1DT	PB0DT	
PDDR	—	—	PD5DT	PD4DT	PD3DT	PD2DT	PD1DT	PD0DT	
PEDR	—	PE6DT	PE5DT	PE4DT	PE3DT	PE2DT	PE1DT	PE0DT	
PGDR	—	—	—	PG4DT	PG3DT	PG2DT	PG1DT	PG0DT	
PHDR	—	—	—	PH4DT	PH3DT	PH2DT	PH1DT	PH0DT	
PACR	—	—	PA6MD1	PA6MD0	PA5MD1	PA5MD0	PA4MD1	PA4MD0	PFC
	PA3MD1	PA3MD0	PA2MD1	PA2MD0	PA1MD1	PA1MD0	PA0MD1	PA0MD0	
PBCR	PB7MD1	PB7MD0	PB6MD1	PB6MD0	PB5MD1	PB5MD0	PB4MD1	PB4MD0	
	PB3MD1	PB3MD0	PB2MD1	PB2MD0	PB1MD1	PB1MD0	PB0MD1	PB0MD0	
PDCR	—	—	—	—	PD5MD1	PD5MD0	PD4MD1	PD4MD0	
	PD3MD1	PD3MD0	PD2MD1	PD2MD0	PD1MD1	PD1MD0	PD0MD1	PD0MD0	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
PECR	—	—	PE6MD1	PE6MD0	PE5MD1	PE5MD0	PE4MD1	PE4MD0	PFC
	PE3MD1	PE3MD0	PE2MD1	PE2MD0	PE1MD1	PE1MD0	PE0MD1	PE0MD0	
PGCR	—	—	—	—	—	—	PG4MD1	PG4MD0	
	PG3MD1	PG3MD0	PG2MD1	PG2MD0	PG1MD1	PG1MD0	PG0MD1	PG0MD0	
PHCR	—	—	—	—	—	—	PH4MD1	PH4MD0	
	PH3MD1	PH3MD0	PH2MD1	PH2MD0	PH1MD1	PH1MD0	PH0MD1	PH0MD0	
PSELA	—	—	—	—	—	—	—	PSA8	
	PSA7	PSA6	PSA5	PSA4	PSA3	PSA2	PSA1	RTCSEL1	
HIZCRA	—	—	—	—	—	—	—	HIZA8	
	HIZA7	HIZA6	HIZA5	HIZA4	HIZA3	HIZA2	HIZA1	HIZA0	
NCCR	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	NCCR0	
BDRB	BDB31	BDB30	BDB29	BDB28	BDB27	BDB26	BDB25	BDB24	UBC
	BDB23	BDB22	BDB21	BDB20	BDB19	BDB18	BDB17	BDB16	
	BDB15	BDB14	BDB13	BDB12	BDB11	BDB10	BDB9	BDB8	
	BDB7	BDB6	BDB5	BDB4	BDB3	BDB2	BDB1	BDB0	
BDMRB	BDMB31	BDMB30	BDMB29	BDMB28	BDMB27	BDMB26	BDMB25	BDMB24	
	BDMB23	BDMB22	BDMB21	BDMB20	BDMB19	BDMB18	BDMB17	BDMB16	
	BDMB15	BDMB14	BDMB13	BDMB12	BDMB11	BDMB10	BDMB9	BDMB8	
	BDMB7	BDMB6	BDMB5	BDMB4	BDMB3	BDMB2	BDMB1	BDMB0	
BRCR	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	SCMFCA	SCMFCB	SCMFDA	SCMFDB	PCTE	PCBA	—	—	
	DBEB	PCBB	—	—	SEQ	—	—	ETBE	
BETR	—	—	—	—	BET11	BET10	BET9	BET8	
	BET7	BET6	BET5	BET4	BET3	BET2	BET1	BET0	
BARB	BAB31	BAB30	BAB29	BAB28	BAB27	BAB26	BAB25	BAB24	
	BAB23	BAB22	BAB21	BAB20	BAB19	BAB18	BAB17	BAB16	
	BAB15	BAB14	BAB13	BAB12	BAB11	BAB10	BAB9	BAB8	
	BAB7	BAB6	BAB5	BAB4	BAB3	BAB2	BAB1	BAB0	

Register Abbreviation	Bit 31/ 23/15/7	Bit 30/ 22/14/6	Bit 29/ 21/13/5	Bit 28/ 20/12/4	Bit 27/ 19/11/3	Bit 26/ 18/10/2	Bit 25/ 17/9/1	Bit 24/ 16/8/0	Module
BAMRB	BAMB31	BAMB30	BAMB29	BAMB28	BAMB27	BAMB26	BAMB25	BAMB24	UBC
	BAMB23	BAMB22	BAMB21	BAMB20	BAMB19	BAMB18	BAMB17	BAMB16	
	BAMB15	BAMB14	BAMB13	BAMB12	BAMB11	BAMB10	BAMB9	BAMB8	
	BAMB7	BAMB6	BAMB5	BAMB4	BAMB3	BAMB2	BAMB1	BAMB0	
BBRB	—	—	—	—	—	—	XYE	XYS	
	CDB1	CDB0	IDB1	IDB0	RWB1	RWB0	SZB1	SZB0	
BRSR	SVF	—	—	—	BSA27	BSA26	BSA25	BSA24	
	BSA23	BSA22	BSA21	BSA20	BSA19	BSA18	BSA17	BSA16	
	BSA15	BSA14	BSA13	BSA12	BSA11	BSA10	BSA9	BSA8	
	BSA7	BSA6	BSA5	BSA4	BSA3	BSA2	BSA1	BSA0	
BARA	BAA31	BAA30	BAA29	BAA28	BAA27	BAA26	BAA25	BAA24	
	BAA23	BAA22	BAA21	BAA20	BAA19	BAA18	BAA17	BAA16	
	BAA15	BAA14	BAA13	BAA12	BAA11	BAA10	BAA9	BAA8	
	BAA7	BAA6	BAA5	BAA4	BAA3	BAA2	BAA1	BAA0	
BAMRA	BAMA31	BAMA30	BAMA29	BAMA28	BAMA27	BAMA26	BAMA25	BAMA24	
	BAMA23	BAMA22	BAMA21	BAMA20	BAMA19	BAMA18	BAMA17	BAMA16	
	BAMA15	BAMA14	BAMA13	BAMA12	BAMA11	BAMA10	BAMA9	BAMA8	
	BAMA7	BAMA6	BAMA5	BAMA4	BAMA3	BAMA2	BAMA1	BAMA0	
BBRA	—	—	—	—	—	—	—	—	
	CDA1	CDA0	IDA1	IDA0	RWA1	RWA0	SZA1	SZA0	
BRDR	DVF	—	—	—	BDA27	BDA26	BDA25	BDA24	
	BDA23	BDA22	BDA21	BDA20	BDA19	BDA18	BDA17	BDA16	
	BDA15	BDA14	BDA13	BDA12	BDA11	BDA10	BDA9	BDA8	
	BDA7	BDA6	BDA5	BDA4	BDA3	BDA2	BDA1	BDA0	
SDIR*	—	—	—	—	—	TI2	TI1	TI0	H-UDI
	—	—	—	—	—	—	—	—	
SDID/SDIDH	DID31	DID30	DID29	DID28	DID27	DID26	DID25	DID24	
	DID23	DID22	DID21	DID20	DID19	DID18	DID17	DID16	
SDID/SDIDL	DID15	DID14	DID13	DID12	DID11	DID10	DID9	DID8	
	DID7	DID6	DID5	DID4	DID3	DID2	DID1	DID0	

Note: * H-UDI pin can write only.

27.3 Register States in Each Operating Mode

Register Abbreviation	Power-on Reset *[1]	Manual Reset*[1]	Software standby*[1]	Module Standby*[1]	Sleep	Module
INTEVT2	Initialized	Initialized	Retained	Retained	Retained	Exception handling
TRA	Initialized	Initialized	Retained	Retained	Retained	
EXPEVT	Initialized	Initialized	Retained	Retained	Retained	
TEA	Initialized	Initialized	Retained	Retained	Retained	
CCR1	Initialized	Initialized	Retained	Retained	Retained	Cache
CCR2	Initialized	Initialized	Retained	Retained	Retained	
IPRA	Initialized	Initialized	Retained	Retained	Retained	INTC
IPRB	Initialized	Initialized	Retained	Retained	Retained	
IPRC	Initialized	Initialized	Retained	Retained	Retained	
IPRD	Initialized	Initialized	Retained	Retained	Retained	
IPRE	Initialized	Initialized	Retained	Retained	Retained	
IPRF	Initialized	Initialized	Retained	Retained	Retained	
IPRG	Initialized	Initialized	Retained	Retained	Retained	
IPRH	Initialized	Initialized	Retained	Retained	Retained	
IMR0	Initialized	Initialized	Retained	Retained	Retained	
IMR1	Initialized	Initialized	Retained	Retained	Retained	
IMR4	Initialized	Initialized	Retained	Retained	Retained	
IMR5	Initialized	Initialized	Retained	Retained	Retained	
IMR6	Initialized	Initialized	Retained	Retained	Retained	
IMR9	Initialized	Initialized	Retained	Retained	Retained	
IMCR0	Initialized	Initialized	Retained	Retained	Retained	
IMCR1	Initialized	Initialized	Retained	Retained	Retained	
IMCR4	Initialized	Initialized	Retained	Retained	Retained	
IMCR5	Initialized	Initialized	Retained	Retained	Retained	
IMCR6	Initialized	Initialized	Retained	Retained	Retained	
IMCR9	Initialized	Initialized	Retained	Retained	Retained	
ICR0	Initialized	Initialized	Retained	Retained	Retained	
ICR1	Initialized	Initialized	Retained	Retained	Retained	
ICR2	Initialized	Initialized	Retained	Retained	Retained	
IRR0	Initialized	Initialized	Retained	Retained	Retained	

Register Abbreviation	Power-on Reset *[1]	Manual Reset*[1]	Software standby*[1]	Module Standby*[1]	Sleep	Module
CMNCR	Initialized	Retained	Retained	Retained	Retained	BSC
CS0BCR	Initialized	Retained	Retained	Retained	Retained	
CS3BCR	Initialized	Retained	Retained	Retained	Retained	
CS4BCR	Initialized	Retained	Retained	Retained	Retained	
CS0WCR	Initialized	Retained	Retained	Retained	Retained	
CS3WCR	Initialized	Retained	Retained	Retained	Retained	
CS4WCR	Initialized	Retained	Retained	Retained	Retained	
SDCR	Initialized	Retained	Retained	Retained	Retained	
RTCSR	Initialized	Retained	Retained	Retained	Retained	
RWTCNT	Initialized	Retained	Retained	Retained	Retained	
RTCNT	Initialized	Retained	Retained	Retained	Retained	
RTCOR	Initialized	Retained	Retained	Retained	Retained	
SDMR3	—	—	—	—	—	
SAR_0	Undefined	Undefined	Retained	Retained	Retained	DMAC
DAR_0	Undefined	Undefined	Retained	Retained	Retained	
DMATCR_0	Undefined	Undefined	Retained	Retained	Retained	
CHCR_0	Initialized	Initialized	Retained	Retained	Retained	
IAR_0	Undefined	Undefined	Retained	Retained	Retained	
SAR_1	Undefined	Undefined	Retained	Retained	Retained	
DAR_1	Undefined	Undefined	Retained	Retained	Retained	
DMATCR_1	Undefined	Undefined	Retained	Retained	Retained	
CHCR_1	Initialized	Initialized	Retained	Retained	Retained	
IAR_1	Undefined	Undefined	Retained	Retained	Retained	
SAR_2	Undefined	Undefined	Retained	Retained	Retained	
DAR_2	Undefined	Undefined	Retained	Retained	Retained	
DMATCR_2	Undefined	Undefined	Retained	Retained	Retained	
CHCR_2	Initialized	Initialized	Retained	Retained	Retained	
IAR_2	Undefined	Undefined	Retained	Retained	Retained	
SAR_3	Undefined	Undefined	Retained	Retained	Retained	
DAR_3	Undefined	Undefined	Retained	Retained	Retained	
DMATCR_3	Undefined	Undefined	Retained	Retained	Retained	
CHCR_3	Initialized	Initialized	Retained	Retained	Retained	
IAR_3	Undefined	Undefined	Retained	Retained	Retained	

Register Abbreviation	Power-on Reset *[1]	Manual Reset*[1]	Software standby*[1]	Module Standby*[1]	Sleep	Module
DMAOR	Initialized	Initialized	Retained	Retained	Retained	DMAC
DMARS0	Initialized	Initialized	Retained	Retained	Retained	
DMARS1	Initialized	Initialized	Retained	Retained	Retained	
DMARS2	Initialized	Initialized	Retained	Retained	Retained	
FRQCR	Initialized*[2]	Retained	Retained	Retained	Retained	CPG
WTCNT	Initialized*[2]	Retained	Retained	Retained	Retained	WDT
WTCSR	Initialized*[2]	Retained	Retained	Retained	Retained	
STBCR	Initialized	Retained	Retained	Retained	Retained	Power-down mode
STBCR2	Initialized	Retained	Retained	Retained	Retained	
STBCR3	Initialized	Retained	Retained	Retained	Retained	
STBCR4	Initialized	Retained	Retained	Retained	Retained	
MCCR	Initialized	Retained	Retained	Retained	Retained	
TSTR	Initialized	Initialized	Initialized	Initialized	Retained	TMU
TCOR_0	Initialized	Initialized	Retained	Retained	Retained	
TCNT_0	Initialized	Initialized	Retained	Retained	Retained	
TCR_0	Initialized	Initialized	Retained	Retained	Retained	
TCOR_1	Initialized	Initialized	Retained	Retained	Retained	
TCNT_1	Initialized	Initialized	Retained	Retained	Retained	
TCR_1	Initialized	Initialized	Retained	Retained	Retained	
TCOR_2	Initialized	Initialized	Retained	Retained	Retained	
TCNT_2	Initialized	Initialized	Retained	Retained	Retained	
TCR_2	Initialized	Initialized	Retained	Retained	Retained	
SIMDR	Initialized	Initialized	Retained	Retained	Retained	SIOF
SISCR	Initialized	Initialized	Retained	Retained	Retained	
SITDAR	Initialized	Initialized	Retained	Retained	Retained	
SIRDAR	Initialized	Initialized	Retained	Retained	Retained	
SICDAR	Initialized	Initialized	Retained	Retained	Retained	
SICTR	Initialized	Initialized	Retained	Retained*[3]	Retained	
SIFCTR	Initialized	Initialized	Retained	Retained	Retained	
SISTR	Initialized	Initialized	Retained	Initialized	Retained	
SIIER	Initialized	Initialized	Retained	Retained	Retained	
SITDR	Initialized	Initialized	Retained	Initialized	Retained	
SIRDR	Initialized	Initialized	Retained	Retained	Retained	

Register Abbreviation	Power-on Reset *[1]	Manual Reset*[1]	Software standby*[1]	Module Standby*[1]	Sleep	Module
SITCR	Initialized	Initialized	Retained	Initialized	Retained	SIOF
SIRCR	Initialized	Initialized	Retained	Retained	Retained	
SPICR	Initialized	Initialized	Retained	Retained	Retained	
SCSMR_0	Initialized	Initialized	Retained	Retained	Retained	SCIF_0
SCBRR_0	Initialized	Initialized	Retained	Retained	Retained	
SCSCR_0	Initialized	Initialized	Retained	Retained	Retained	
SCTDSR_0	Initialized	Initialized	Retained	Retained	Retained	
SCFER_0	Initialized	Initialized	Retained	Retained	Retained	
SCSSR_0	Initialized	Initialized	Retained	Retained	Retained	
SCFCR_0	Initialized	Initialized	Retained	Retained	Retained	
SCFDR_0	Initialized	Initialized	Retained	Retained	Retained	
SCFTDR_0	Initialized	Initialized	Retained	Retained	Retained	
SCFRDR_0	Initialized	Initialized	Retained	Retained	Retained	
SCSMR_1	Initialized	Initialized	Retained	Retained	Retained	SCIF_1
SCBRR_1	Initialized	Initialized	Retained	Retained	Retained	
SCSCR_1	Initialized	Initialized	Retained	Retained	Retained	
SCTDSR_1	Initialized	Initialized	Retained	Retained	Retained	
SCFER_1	Initialized	Initialized	Retained	Retained	Retained	
SCSSR_1	Initialized	Initialized	Retained	Retained	Retained	
SCFCR_1	Initialized	Initialized	Retained	Retained	Retained	
SCFDR_1	Initialized	Initialized	Retained	Retained	Retained	
SCFTDR_1	Initialized	Initialized	Retained	Retained	Retained	
SCFRDR_1	Initialized	Initialized	Retained	Retained	Retained	
EXCPGCR	Initialized	Initialized	Retained	Retained	Retained	USBPM
HREVR	Initialized	Initialized	Retained	Retained	Retained	USBH
HCTLR	Initialized	Initialized	Retained	Retained	Retained	
HCSR	Initialized	Initialized	Retained	Retained	Retained	
HISR	Initialized	Initialized	Retained	Retained	Retained	
HIER	Initialized	Initialized	Retained	Retained	Retained	
HIDR	Initialized	Initialized	Retained	Retained	Retained	
HHCCAR	Initialized	Initialized	Retained	Retained	Retained	
HPCEDR	Initialized	Initialized	Retained	Retained	Retained	
HCHEDR	Initialized	Initialized	Retained	Retained	Retained	

Register Abbreviation	Power-on Reset *[1]	Manual Reset*[1]	Software standby*[1]	Module Standby*[1]	Sleep	Module
HCCEDR	Initialized	Initialized	Retained	Retained	Retained	USBH
HBHEDR	Initialized	Initialized	Retained	Retained	Retained	
HBCEDR	Initialized	Initialized	Retained	Retained	Retained	
HDHEDR	Initialized	Initialized	Retained	Retained	Retained	
HFIR	Initialized	Initialized	Retained	Retained	Retained	
HFRR	Initialized	Initialized	Retained	Retained	Retained	
HFNR	Initialized	Initialized	Retained	Retained	Retained	
HPSR	Initialized	Initialized	Retained	Retained	Retained	
HLSTR	Initialized	Initialized	Retained	Retained	Retained	
HRDRA	Initialized	Initialized	Retained	Retained	Retained	
HRDRB	Initialized	Initialized	Retained	Retained	Retained	
HRSR	Initialized	Initialized	Retained	Retained	Retained	
HRPSR	Initialized	Initialized	Retained	Retained	Retained	
IFR0	Initialized	Initialized	Retained	Retained	Retained	USBF
ISR0	Initialized	Initialized	Retained	Retained	Retained	
IER0	Initialized	Initialized	Retained	Retained	Retained	
EPDR0i	Initialized	Initialized	Retained	Retained	Retained	
EPDR0o	Initialized	Initialized	Retained	Retained	Retained	
EPDR0s	Initialized	Initialized	Retained	Retained	Retained	
EPDR1	Initialized	Initialized	Retained	Retained	Retained	
EPDR2i	Initialized	Initialized	Retained	Retained	Retained	
EPDR2o	Initialized	Initialized	Retained	Retained	Retained	
EPDR3i	Initialized	Initialized	Retained	Retained	Retained	
EPDR3o	Initialized	Initialized	Retained	Retained	Retained	
EPDR4	Initialized	Initialized	Retained	Retained	Retained	
EPDR5	Initialized	Initialized	Retained	Retained	Retained	
EPDR6	Initialized	Initialized	Retained	Retained	Retained	
EPSZ0o	Initialized	Initialized	Retained	Retained	Retained	
EPSZ2o	Initialized	Initialized	Retained	Retained	Retained	
EPSZ3o	Initialized	Initialized	Retained	Retained	Retained	
EPSZ6	Initialized	Initialized	Retained	Retained	Retained	
TRG	Initialized	Initialized	Retained	Retained	Retained	
DASTS	Initialized	Initialized	Retained	Retained	Retained	

Register Abbreviation	Power-on Reset *[1]	Manual Reset*[1]	Software standby*[1]	Module Standby*[1]	Sleep	Module
FCLR	Initialized	Initialized	Retained	Retained	Retained	USBF
DMA	Initialized	Initialized	Retained	Retained	Retained	
EPSTL	Initialized	Initialized	Retained	Retained	Retained	
CVR	Initialized	Initialized	Retained	Retained	Retained	
TSR	Initialized	Initialized	Retained	Retained	Retained	
CTLR	Initialized	Initialized	Retained	Retained	Retained	
EPIR	Initialized	Initialized	Retained	Retained	Retained	
All registers	Initialized/ Retained*[4]	Retained	Retained	Retained	Retained	BT
DADR_0	Initialized	Retained	Retained	Retained	Retained	DAC
DADR_1	Initialized	Retained	Retained	Retained	Retained	
DACR	Initialized	Retained	Retained	Retained	Retained	
PADR	Initialized	Retained	Retained	Retained	Retained	I/O port
PBDR	Initialized	Retained	Retained	Retained	Retained	
PDDR	Initialized	Retained	Retained	Retained	Retained	
PEDR	Initialized	Retained	Retained	Retained	Retained	
PGDR	Initialized	Retained	Retained	Retained	Retained	
PHDR	Initialized	Retained	Retained	Retained	Retained	
PACR	Initialized	Retained	Retained	Retained	Retained	PFC
PBCR	Initialized	Retained	Retained	Retained	Retained	
PDCR	Initialized	Retained	Retained	Retained	Retained	
PECR	Initialized	Retained	Retained	Retained	Retained	
PGDR	Initialized	Retained	Retained	Retained	Retained	
PHDR	Initialized	Retained	Retained	Retained	Retained	
PSELA	Initialized	Retained	Retained	Retained	Retained	
HIZCRA	Initialized	Retained	Retained	Retained	Retained	
NCCR	Initialized	Retained	Retained	Retained	Retained	
BDRB	Initialized	Retained	Retained	Retained	Retained	UBC
BDMRB	Initialized	Retained	Retained	Retained	Retained	
BRCR	Initialized	Retained	Retained	Retained	Retained	
BETR	Initialized	Retained	Retained	Retained	Retained	
BARB	Initialized	Retained	Retained	Retained	Retained	
BAMRB	Initialized	Retained	Retained	Retained	Retained	

Register Abbreviation	Power-on Reset *[1]	Manual Reset*[1]	Software standby*[1]	Module Standby*[1]	Sleep	Module
BBRB	Initialized	Retained	Retained	Retained	Retained	UBC
BRSR	Initialized	Retained	Retained	Retained	Retained	
BARA	Initialized	Retained	Retained	Retained	Retained	
BAMRA	Initialized	Retained	Retained	Retained	Retained	
BBRA	Initialized	Retained	Retained	Retained	Retained	
BRDR	Initialized	Retained	Retained	Retained	Retained	
SDIR	Retained	Retained	Retained	Retained	Retained	H-UDI
SDID/SDIDH*[5]	—	—	—	—	—	
SDID/SDIDL*[5]	—	—	—	—	—	
U memory	Undefined	Undefined	Retained	Retained	Retained	U memory
X/Y memory	Undefined	Undefined	Retained	Retained	Retained	X/Y memory

Notes: 1. For details on initial values, please refer to the various sections related to the modules. In addition, registers whose initial values are undefined are also indicated as initialized.
2. Initialized by a reset signal from the $\overline{\text{RESETP}}$ pin, and retained by a reset signal from other pin.
3. Some bits are initialized.
4. All registers in BT module are initialized only by the power-on reset input from pin. They are not initialized by the power-on reset and manual reset generated from WDT.
5. Fixed value.

28.1 Absolute Maximum Ratings

Table 28.1 shows the absolute maximum ratings.

Table 28.1 Absolute Maximum Ratings

Item	Symbol	Rating	Unit
Power supply voltage (except V_{CC}_28)	V_{CC}(I/O) V_{CC}(internal) V_{CC}(PLL) V_{CC}(DLL) V_{CC}(SREG)	–0.3 to 4.2	V
Power supply voltage (V_{CC}_28)	V_{CC}_28	–0.3 to 4.2	V
Power supply voltage (V_{DD}): (using external power supply)	V_{DD}	–0.3 to 2.1	V
Input voltage (except V_{CC}_28)	V_{in}	–0.3 to V_{CC} + 0.3	V
Input voltage (V_{CC}_28)	V_{in}	–0.3 to V_{CC}_28 + 0.3	V
Analog power supply voltage (DAC)	AV_{CC} (DAC)	–0.3 to 4.2	V
Analog power supply voltage (USB)	AV_{CC} (USB)	–0.3 to 4.2	V
Analog input voltage (USB)	V_{IN}	–0.3 to AV_{CC} (USB) + 0.3	V
Operating temperature	T_{opr}	–40 to 85	°C
Storage temperature	T_{stg}	–55 to 125	°C

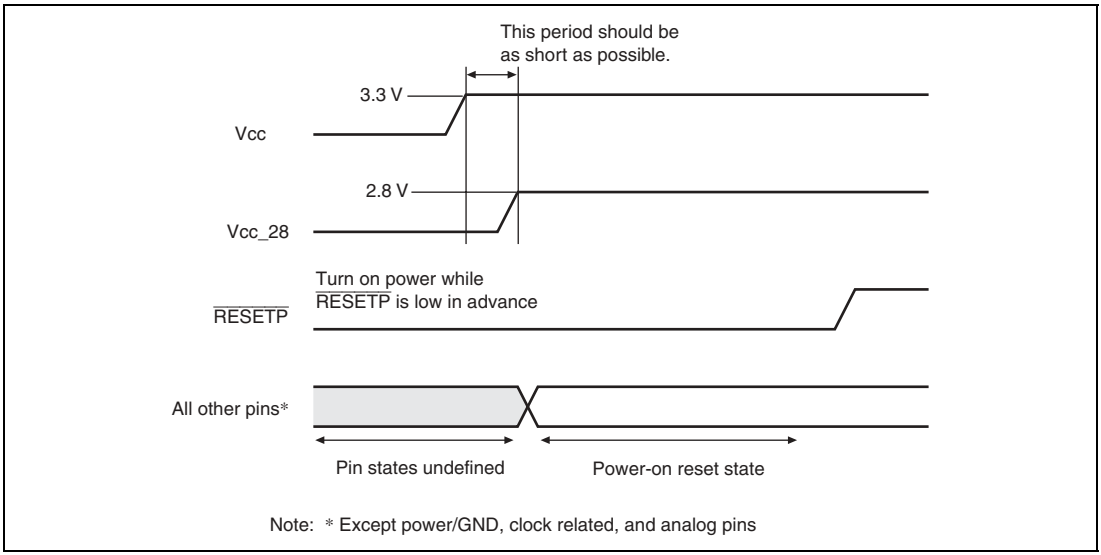
Usage Notes:

1. Operating the chip in excess of the absolute maximum rating may result in permanent damage.
2. Order of turning on V_{CC}_28 and other 3.3 V power supplies (V_{CC}, AV_{CC})
   Though there are no strict requirements, the following points are recommended since a cutoff of either power supply will cause unstable operation.
   — First turn on the 3.3 V power supplies except for the V_{CC}_28 power, then turn on the V_{CC}_28 power. This interval should be as short as possible.

— Until voltage is applied to all power supplies and a low level is input to the $\overline{\text{RESETP}}$ pin, internal circuits remain unsettled, and so pin states are also undefined. The system design must ensure that these undefined states do not cause erroneous system operation.

Waveforms at power-on are shown in the figure below.

— The power supply voltage of V_{CC}_28 should be within the range of the absolute maximum ratings above and also equal to the power supply voltage of the RF IC to be connected.



Power-On Sequence

3. Order of turning off power

   Though there are no strict requirements, the following points are recommended since a cutoff of either power supply will cause unstable operation.

   — In the reverse order of powering-on, first turn off the V_{CC}_28 power, then turn off the 3.3 V power supplies. This interval should be as short as possible.

   — Pin states are undefined while only the V_{CC}_28 power is off. The system design must ensure that these undefined states do not cause erroneous system operation.

4. When the built-in power supply regulator is used, each V_{DD} pin should be connected to the capacitor to reduce noise of power supply.
5. When the built-in power supply regulator is used, each V_{DD} pin should not be applied voltage from outside. Also, the V_{DD} pins cannot be used to supply current to the other devices.

28.2 DC Characteristics

Tables 28.2 and 28.3 list the DC characteristics.

Table 28.2 DC Characteristics (1) Common Items

Condition: Ta = –40 to 85°C

Item		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Current consumption	Normal operation	I_{CC}	—	—	350	mA	V_{CC} = 3.3 V Iϕ = 120 MHz Bϕ = 60 MHz
Input leak current	All input pins	I_{in}	—	—	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5 V
Three-state leak current	I/O, all output pins (off condition)	I_{STI}	—	—	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5 V
Pull-up resistance	Port pin	P_{pull}	30	60	120	kΩ	
Pin capacity	All digital pins*	C	—	—	10	pF	
Analog power supply voltage (DAC)		AV_{CC}(DAC)	2.7	3.3	3.6	V	
Analog power supply voltage (USB)		AV_{CC}(USB)	3.0	3.3	3.6	V	
Internal power supply voltage (V_{DD})		V_{DD}	1.4	1.5	1.6	V	When external power supply is used.

Note: * Power supply pins, analog pins, VBB, USB_N, USB_P, RDI_CTRL4, DA0, DA1, EXTAL, XTAL, EXTAL2, XTAL2, RREF, and NC are excluded.

Table 28.2 DC Characteristics (2-a) Except DAC and USB Related Pins

Condition: Ta = –40 to 85°C

Item		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Power supply voltage		V_{CC}	2.7	3.3	3.6	V	
		V_{CC}_28	2.7	2.8	3.0		*1
			2.7	3.3	3.6		*2
Input high voltage	NMI, MD1, MD2, MD5, $\overline{\text{ASEMD0}}$, IRQ0, IRQ2 to IRQ4	V_{IH}	$V_{CC} \times 0.9$	—	$V_{CC} + 0.3$	V	
	$\overline{\text{TRST}}$		$V_{CC} \times 0.75$	—	$V_{CC} + 0.3$		
	EXTAL, EXTAL2		$V_{CC} - 0.6$	—	$V_{CC} + 0.3$		
	$\overline{\text{RESETP}}$		V_{CC}_28 × 0.9	—	V_{CC}_28 + 0.3		
	RDI_TXTRDATA, RDI_REFCLK_IN, RCI_SPI_TXRX		2.0	—	V_{CC}_28 + 0.3		
	Other input pins		2.0	—	$V_{CC} + 0.3$		
Input low voltage	NMI, IRQ0, IRQ2 to IRQ4, MD1, MD2, MD5, $\overline{\text{ASEMD0}}$, $\overline{\text{TRST}}$	V_{IL}	–0.3	—	$V_{CC} \times 0.1$	V	
	EXTAL, EXTAL2		–0.3	—	$V_{CC} \times 0.2$		
	$\overline{\text{RESETP}}$, RDI_TXTRDATA, RDI_REFCLK_IN, RCI_SPI_TXRX		–0.3	—	V_{CC}_28 × 0.1		
	Other input pins		–0.3	—	$V_{CC} \times 0.2$		
Output high voltage	RDI_CTRL4	V_{OH}	2.25	—	—	V	V_{CC}_28, V_{CC} = 2.7 V, I_{OH} = –3 mA
	Other output pins		2.0	—	—		V_{CC}_28, V_{CC} = 2.7 V, I_{OH} = –2 mA

Item		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Output low voltage	RDI_TXTRDATA, RDI_RXBDW_OUT, RCI_SPI_CLK, RCI_SPI_ENB, RCI_SPI_TXRX	V_{OL}	—	—	0.55	V	V_{CC}_28 = 3.0 V, I_{OL} = 2.0 mA
	RDI_CTRL4	V_{OL}	—	—	0.25	V	V_{CC} = 3.6 V, I_{OL} = 0.5 mA
	Other output pins	V_{OL}	—	—	0.55	V	V_{CC} = 3.6 V, I_{OL} = 2.0 mA

Notes: *1 When Renesas Technology RF-IC (HD157100NP or HD157102NP) is connected.

*2 Connect these pins to Vcc when RF-IC is not used.

Notes: 1. AV_{CC} conditions must be: $V_{CC} - 0.2\ V \le AV_{CC}(DAC) \le V_{CC} + 0.2\ V$. Even if the D/A converter is not used, do not leave the AV_{CC} (DAC) and AV_{SS} (DAC) pins open. Connect AV_{CC} (DAC) to V_{CC}, and connect AV_{SS} (DAC) to V_{SS}.

2. Current consumption values shown are for V_{IH} (Min.) = V_{CC} – 0.5 V and V_{IL} (Max.) = 0.5 V with all output pins unloaded.

Table 28.2 DC Characteristics (2-b) USB Related Pins*

Condition: Ta = –40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Power supply voltage	V_{CC}	2.7	3.3	3.6	V	
Input high voltage	V_{IH}	2.0	—	V_{CC} + 0.3	V	
Input low voltage	V_{IL}	–0.3	—	$V_{CC} \times 0.2$	V	
Input high voltage (UCLK)	V_{IH}(UCLK)	V_{CC} – 0.3	—	V_{CC} + 0.3	V	
Input low voltage (UCLK)	V_{IL}(UCLK)	–0.3	—	$V_{CC} \times 0.2$	V	
Output high voltage	V_{OH}	2.4	—	—	V	V_{CC} = 2.7 V, I_{OH} = –200 µA
		2.0	—	—		V_{CC} = 2.7 V, I_{OH} = –2 mA
Output low voltage	V_{OL}	—	—	0.55	V	V_{CC} = 3.6 V, I_{OL} = 2.0 mA

Note: * The UCLK, $\overline{\text{USB_PWR_EN}}$/USB_PULLUP, and $\overline{\text{USB_OVR_CRNT}}$/USB_VBUS pins.

Table 28.2 DC Characteristics (2-c) USB Transceiver Related Pins*

Condition: Ta = –40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Power supply voltage	AV_{CC}(USB)	3.0	3.3	3.6	V	
Differential input sensitivity	V_{DI}	0.2	—	—	V	|(USB_P) – (USB_N)|
Differential common mode range	V_{CM}	0.8	—	2.5	V	
Single ended receiver threshold voltage	V_{SE}	0.8	—	2.0	V	
Output high voltage	V_{OH}	2.8	—	AV_{CC}(USB)	V	
Output low voltage	V_{OL}	—	—	0.3	V	
Tri-state leak current	I_{LO}	–10	—	10	µA	0 V < V_{IN} < 3.3 V

Note: * The USB_P and USB_N pins.

Table 28.3 Permissible Output Current Values

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}_28 = 2.7 to 3.0 V, AV_{CC} = 2.7 to 3.6 V, Ta = –40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit
Permissible output low current (per pin)	I_{OL}	—	—	2	mA
Permissible output low current (total)	I_{OL}	—	—	50	mA
Permissible output high current (per pin)	$-I_{OH}$	—	—	2	mA
Permissible output high current (total)	$(-I_{OH})$	—	—	50	mA

Note: To ensure chip reliability, do not exceed the output current values given in table 28.3.

28.3 AC Characteristics

The input of this LSI is basically a synchronous input. The setup and hold time of each input signal should be kept unless any notice.

Table 28.4 Operating Frequencies

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}–28 = 2.7 to 3.0 V, AV_{CC} = 2.7 to 3.6 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

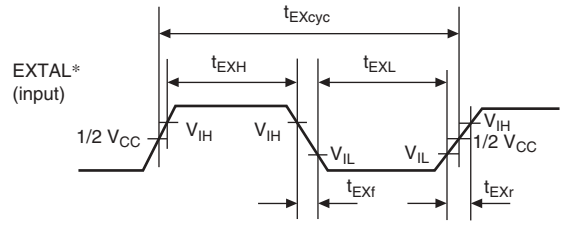
Item		Symbol	Min	Typ	Max	Unit	Remarks
Operating frequency	CPU, cache (Iϕ)	f	20	—	120	MHz	
	External bus (Bϕ)		20	—	60		
	Peripheral module (Pϕ)		4	—	30		

28.3.1 Clock Timing

Table 28.5 Clock Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}–28= 2.7 to 3.0 V, AV_{CC} = 2.7 to 3.6 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C, maximum external bus operating frequency: 60 MHz

Item	Symbol	Min	Max	Unit	Figure
EXTAL clock input frequency	f_{EX}	10	30	MHz	28.1
EXTAL clock input cycle time	t_{EXcyc}	33.3	100	ns	
EXTAL clock input low pulse width	t_{EXL}	15	—	ns	
EXTAL clock input high pulse width	t_{EXH}	15	—	ns	
EXTAL clock input rise time	t_{EXr}	—	4	ns	
EXTAL clock input fall time	t_{EXf}	—	4	ns	
CKIO clock input frequency	f_{CKI}	20	60	MHz	28.2
CKIO clock input cycle time	t_{CKICYC}	16.6	50	ns	
CKIO clock input low pulse width	t_{CKIL}	4	—	ns	
CKIO clock input high pulse width	t_{CKIH}	4	—	ns	
CKIO clock input rise time	t_{CKIR}	—	2	ns	
CKIO clock input fall time	t_{CKIF}	—	2	ns	
CKIO clock output frequency	f_{OP}	20	60	MHz	28.3
CKIO clock output cycle time	t_{cyc}	16.7	50	ns	
CKIO clock output low pulse width	t_{CKOL}	5	—	ns	
CKIO clock output high pulse width	t_{CKOH}	5	—	ns	
CKIO clock output rise time	t_{CKOr}	—	6	ns	
CKIO clock output fall time	t_{CKOf}	—	6	ns	
$\overline{\text{RESETP}}$ assert time (when Renesas RF-IC is connected)	t_{RESPW}	100	—	µs	28.4, 28.5
Power-On Oscillation Settling Time	t_{SOC1}	10	—	ms	28.4
Oscillation Settling Time on Return from Standby 1	t_{SOC2}	10	—	ms	28.5
Oscillation Settling Time on Return from Standby 2	t_{SOC3}	10	—	ms	28.6
PLL synchronization settling time	t_{PLL}	100	—	µs	28.7, 28.8



Note: * When clock is input from EXTAL pin

Figure 28.1 EXTAL Clock Input Timing

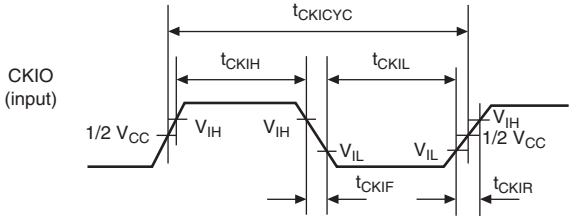


Figure 28.2 CKIO Clock Input Timing

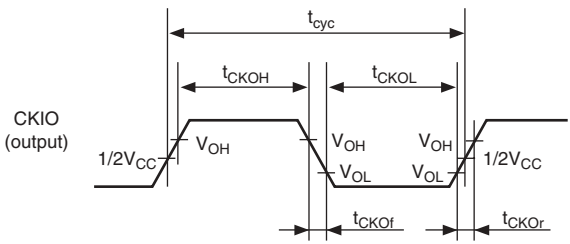
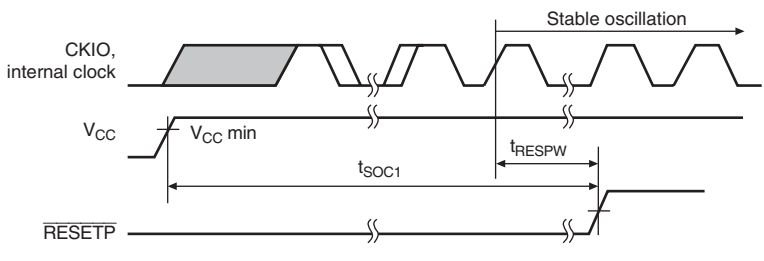


Figure 28.3 CKIO Clock Output Timing



Note: Oscillation settling time when on-chip oscillator is used

Figure 28.4 Power-On Oscillation Settling Time

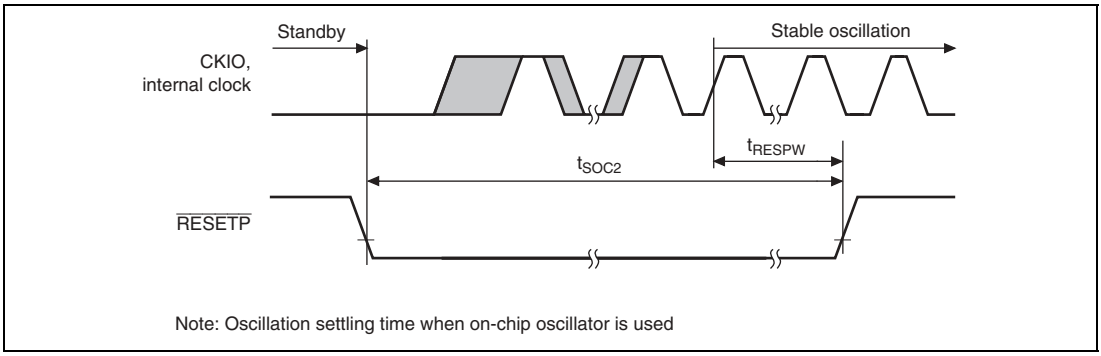


Figure 28.5 Oscillation Settling Time on Return from Standby (Return by Reset)

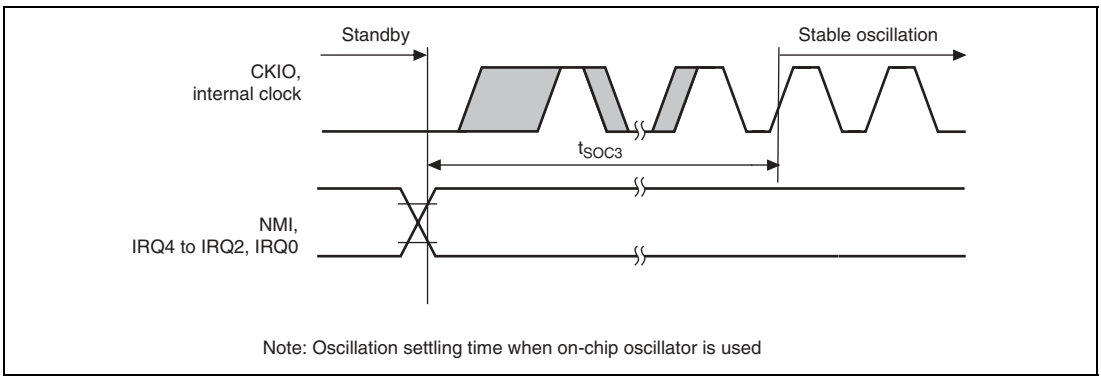


Figure 28.6 Oscillation Settling Time on Return from Standby (Return by NMI or IRQ)

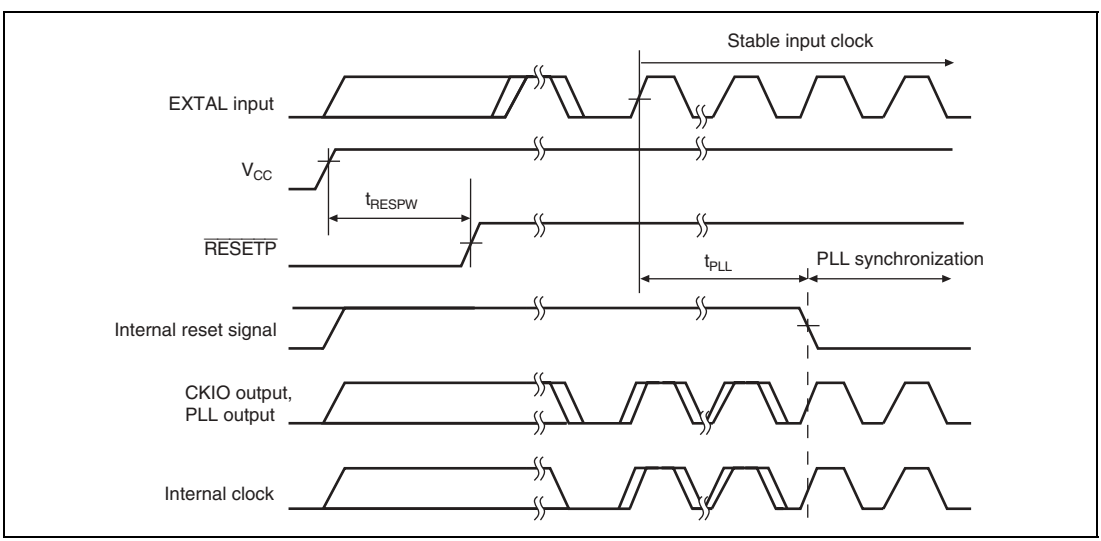


Figure 28.7 PLL Synchronization Settling Time at Power-On Reset

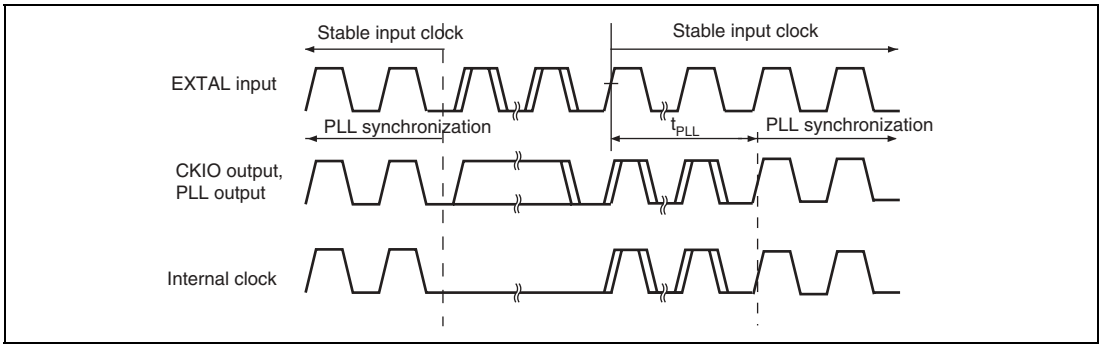


Figure 28.8 PLL Synchronization Settling Time in Case of Reset or NMI Interrupt

28.3.2 Control Signal Timing

Table 28.6 Control Signal Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}–28= 2.7 to 3.0 V, AV_{CC} = 2.7 to 3.6 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C*[2]

Item	Symbol	Min	Max	Unit	Figure
$\overline{\text{RESETP}}$ pulse width	t_{RESPW}	100	—	µs	28.9, 28.10
$\overline{\text{RESETP}}$ setup time*[1]	t_{RESPS}	30	—	ns	
$\overline{\text{RESETP}}$ hold time	t_{RESPH}	4	—	ns	
$\overline{\text{BOOT-E}}$ setup time*[4]	t_{BOOTS}	50	—	µs	28.9
$\overline{\text{BREQ}}$ setup time	t_{BREQS}	$1/2t_{cyc}$ *[3] + 7	—	ns	28.13
$\overline{\text{BREQ}}$ hold time	t_{BREQH}	$1/2t_{cyc}$ + 3	—	ns	
NMI setup time*[1]	t_{NMIS}	20	—	ns	28.10
NMI hold time	t_{NMIH}	4	—	ns	
IRQ4 to IRQ2, IRQ0 setup time*[1]	t_{IRQS}	20	—	ns	
IRQ4 to IRQ2, IRQ0 hold time	t_{IRQH}	4	—	ns	
$\overline{\text{IRQOUT}}$ delay time	t_{IRQOD}	—	13	ns	28.11
$\overline{\text{REFOUT}}$ delay time	t_{REFOD}	—	13	ns	28.12
$\overline{\text{BACK}}$ delay time	t_{BACKD}	$1/2t_{cyc}$	$1/2t_{cyc}$ + 13	ns	28.13, 28.14
Bus tri-state delay time 1	t_{BOFF1}	0	30	ns	
Bus tri-state delay time 2	t_{BOFF2}	0	30	ns	
Bus buffer-on time 1	t_{BON1}	0	30	ns	28.13, 28.14
Bus buffer-on time 2	t_{BON2}	0	30	ns	

Notes: 1. $\overline{\text{RESETP}}$, NMI, IRQ4 to IRQ2, and IRQ0 are asynchronous. Changes are detected at the clock rise when the setup time shown is used. If the setup time cannot be used, detection may be delayed until the next clock rises.
The pulse width of two cycles or more in peripheral module clock (Pϕ) is necessary for NMI and IRQ4 to IRQ2 and IRQ0 at the edge detection.

2. The upper limit of the external bus clock is 60 MHz.
3. tcyc means the external bus clock cycle (Bϕ clock cycle)
4. The $\overline{\text{BOOT-E}}$ signal should not be changed excluding for the period when $\overline{\text{RESETP}}$ asserts low (during the reset period). If the $\overline{\text{BOOT-E}}$ signal is changed excluding the reset period, operation cannot be guaranteed.

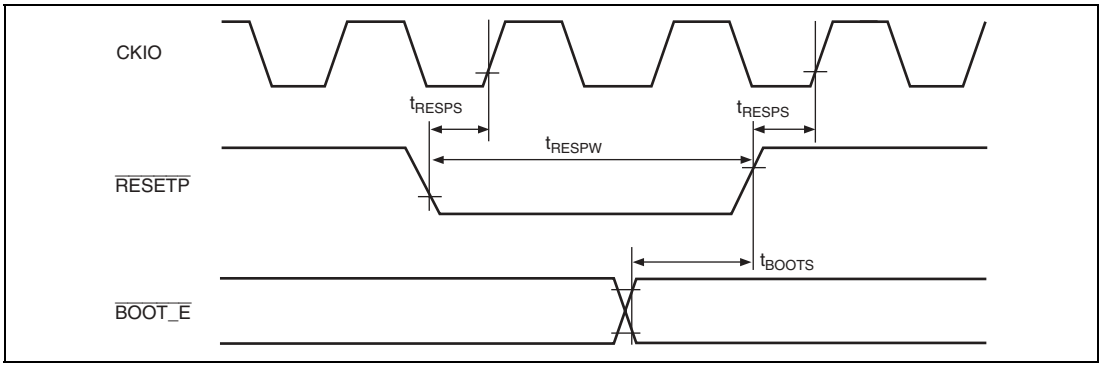


Figure 28.9 Reset and BOOT-E Input Timing

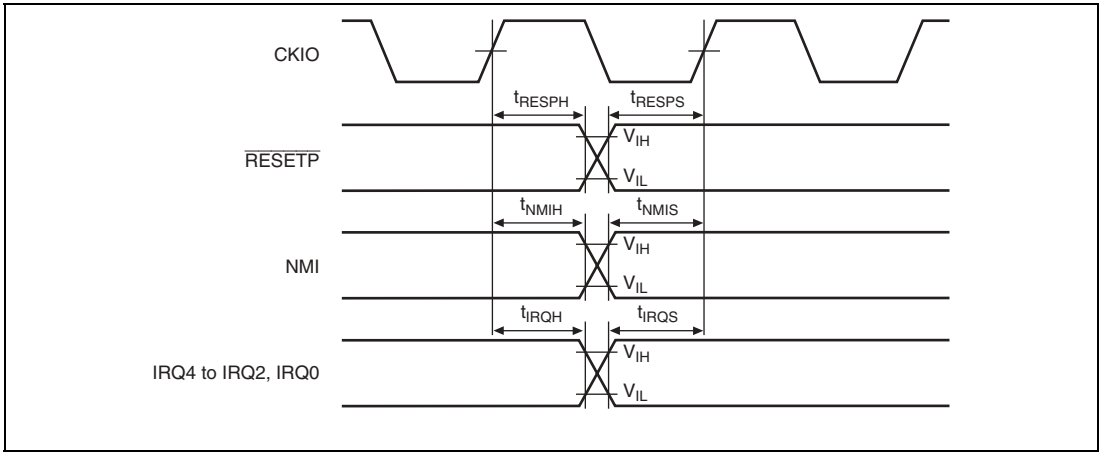


Figure 28.10 Interrupt Signal Input Timing

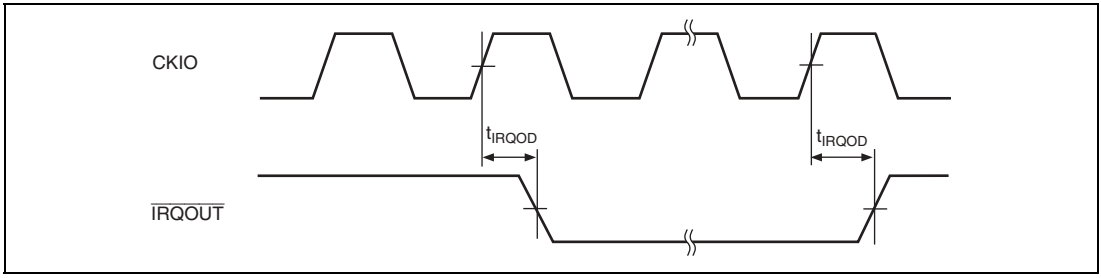


Figure 28.11 IRQOUT Timing

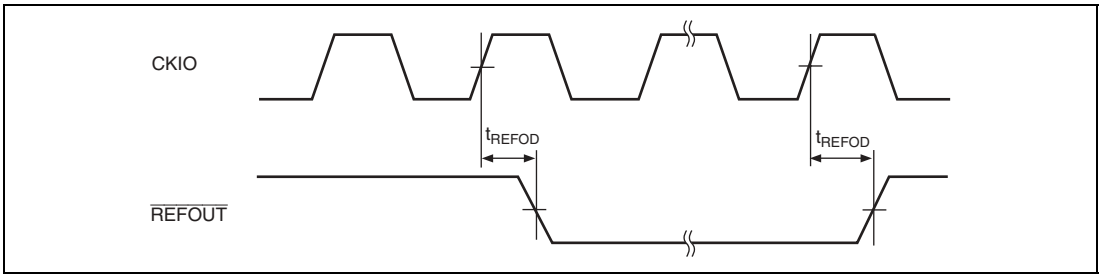


Figure 28.12 REFOUT Timing

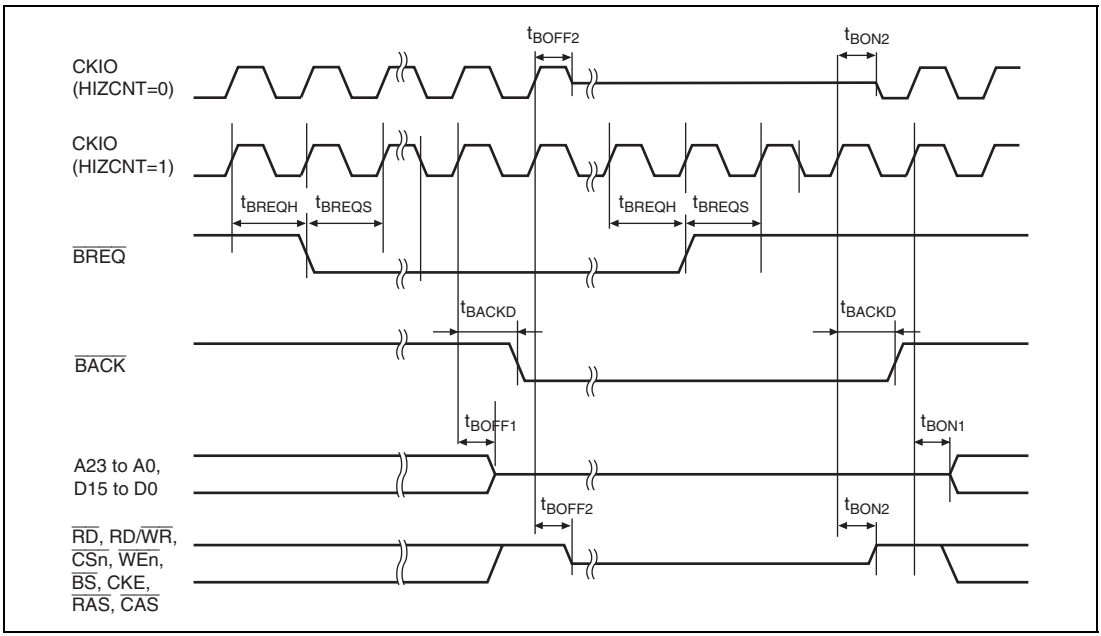


Figure 28.13 Bus Release Timing

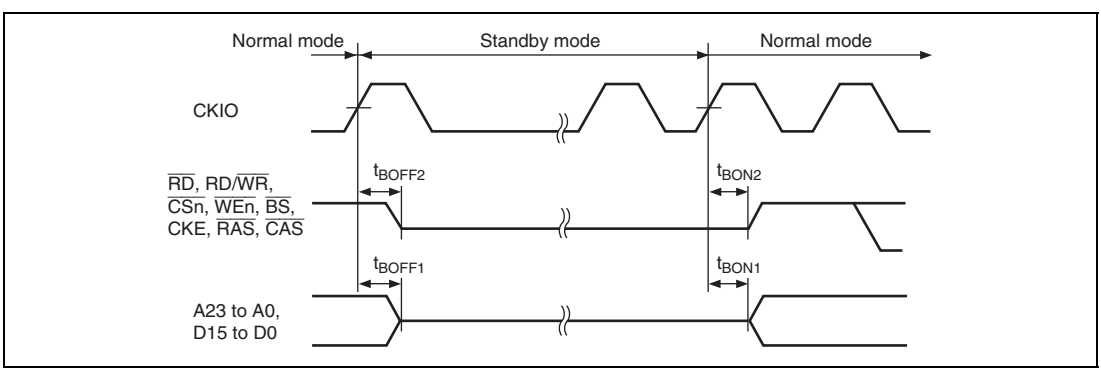


Figure 28.14 Pin Drive Timing at Standby

28.3.3 AC Bus Timing

Table 28.7 Bus Timing

Conditions: Clock modes 5, 6, 7*, V_{CC} = 2.7 to 3.6 V, V_{CC}_28= 2.7 to 3.0 V, AV_{CC} = 2.7 to 3.6 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

		60 MHz			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Figure**
Address delay time 1	t_{AD1}	1	13	ns	28.15 to 28.45
Address delay time 2	t_{AD2}	$1/2t_{cyc}$	$1/2t_{cyc}$ + 13	ns	28.22
Address delay time 3	t_{AD3}	$1/2t_{cyc}$	$1/2t_{cyc}$ + 13	ns	28.40 to 28.42
Address setup time	t_{AS}	0	—	ns	28.15 to 28.22
Address hold time	t_{AH}	0	—	ns	28.15 to 28.21
$\overline{BS}$ delay time	t_{BSD}	—	13	ns	28.15 to 28.39, 28.44, 28.45
$\overline{CS}$ delay time 1	t_{CSD1}	1	13	ns	28.15 to 28.39, 28.44, 28.45
$\overline{CS}$ delay time 2	t_{CSD2}	$1/2t_{cyc}$	$1/2t_{cyc}$ + 13	ns	28.40 to 28.42
Read/write delay time 1	t_{RWD1}	1	13	ns	28.15 to 28.39, 28.44, 28.45
Read/write delay time 2	t_{RWD2}	$1/2t_{cyc}$	$1/2t_{cyc}$ + 13	ns	28.40 to 28.42
Read strobe delay time	t_{RSD}	$1/2t_{cyc}$	$1/2t_{cyc}$ + 13	ns	28.15 to 28.22
Read data setup time 1	t_{RDS1}	$1/2t_{cyc}$ + 10	—	ns	28.15 to 28.21
Read data setup time 2	t_{RDS2}	7	—	ns	28.23 to 28.26, 28.31 to 28.33, 28.44, 28.45
Read data setup time 3	t_{RDS3}	$1/2t_{cyc}$ + 10	—	ns	28.22
Read data setup time 4	t_{RDS4}	$1/2t_{cyc}$ + 10	—	ns	28.40
Read data hold time 1	t_{RDH1}	0	—	ns	28.14 to 28.21
Read data hold time 2	t_{RDH2}	2	—	ns	28.23 to 28.26, 28.31 to 28.33, 28.44, 28.45
Read data hold time 3	t_{RDH3}	0	—	ns	28.22
Read data hold time 4	t_{RDH4}	0	—	ns	28.40
Write enable delay time 1	t_{WED1}	$1/2t_{cyc}$	$1/2t_{cyc}$ + 13	ns	28.15 to 28.21
Write enable delay time 2	t_{WED2}	—	13	ns	28.21
Write data delay time 1	t_{WDD1}	—	13	ns	28.15 to 28.21
Write data delay time 2	t_{WDD2}	—	13	ns	28.27 to 28.30, 28.34 to 28.36, 28.44, 28.45
Write data delay time 3	t_{WDD3}	—	$1/2t_{cyc}$ + 13	ns	28.40
Write data hold time 1	t_{WDH1}	1	—	ns	28.15 to 28.21
Write data hold time 2	t_{WDH2}	1	—	ns	28.27 to 28.30, 28.34 to 28.36, 28.44, 28.45

Item	Symbol	66 MHz Min	66 MHz Max	Unit	Figure
Write data hold time 3	t_{WDH3}	$1/2t_{cyc}$	—	ns	28.40
$\overline{WAIT}$ setup time	t_{WTS1}	$1/2t_{cyc} + 7$	—	ns	28.15 to 28.22
$\overline{WAIT}$ hold time	t_{WTH1}	$1/2t_{cyc} + 3$	—	ns	28.15 to 28.22
$\overline{RAS}$ delay time 1	t_{RASD1}	1	13	ns	28.23 to 28.39, 28.44, 28.45
$\overline{RAS}$ delay time 2	t_{RASD2}	$1/2t_{cyc}$	$1/2t_{cyc} + 13$	ns	28.40 to 28.43
$\overline{CAS}$ delay time 1	t_{CASD1}	1	13	ns	28.23 to 28.39, 28.44, 28.45
$\overline{CAS}$ delay time 2	t_{CASD2}	$1/2t_{cyc}$	$1/2t_{cyc} + 13$	ns	28.40 to 28.43
DQM delay time 1	t_{DQMD1}	1	13	ns	28.23 to 28.39, 28.44, 28.45
DQM delay time 2	t_{DQMD2}	$1/2t_{cyc}$	$1/2t_{cyc} + 13$	ns	28.40 to 28.43
CKE delay time 1	t_{CKED1}	1	13	ns	28.38, 28.39, 28.44, 28.45
CKE delay time 2	t_{CKED2}	$1/2t_{cyc}$	$1/2t_{cyc} + 13$	ns	28.42

Note: * The timing of SDRAM interface is in clock mode 7. It is recommended that the SDRAM interface be operated in clock 7.

28.3.4 Basic Timing

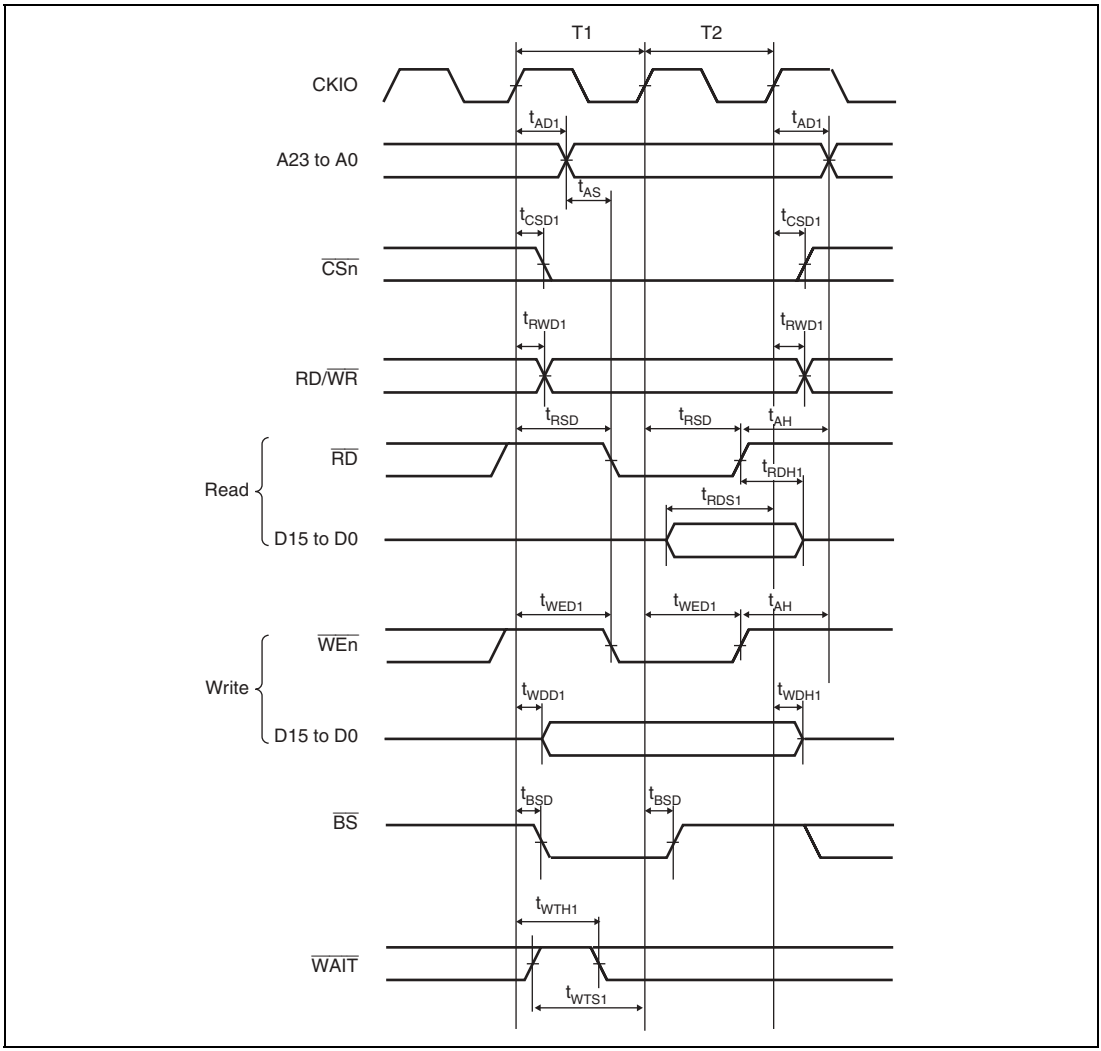


Figure 28.15 Basic Bus Cycle in Normal Space (No Wait)

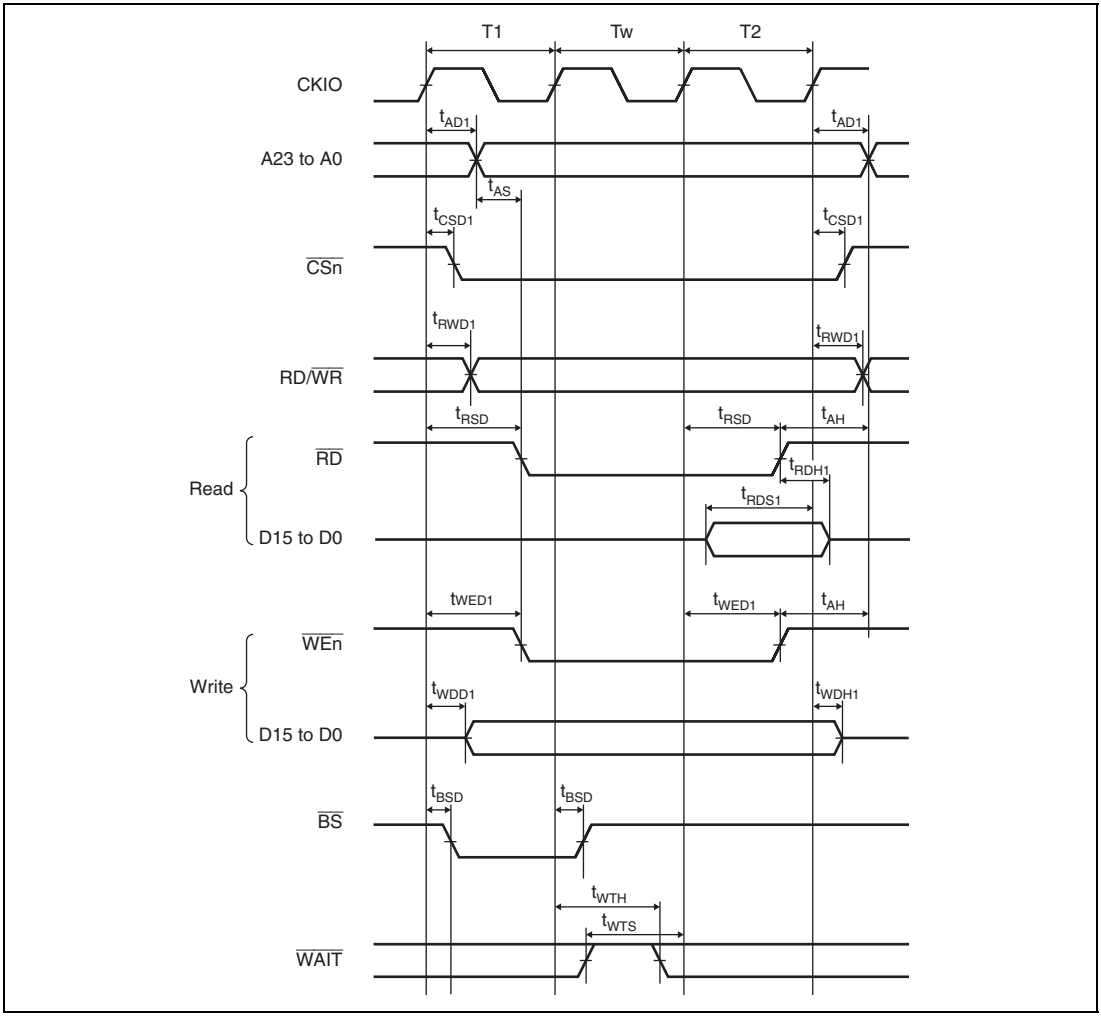


Figure 28.16 Basic Bus Cycle in Normal Space (Software Wait 1)

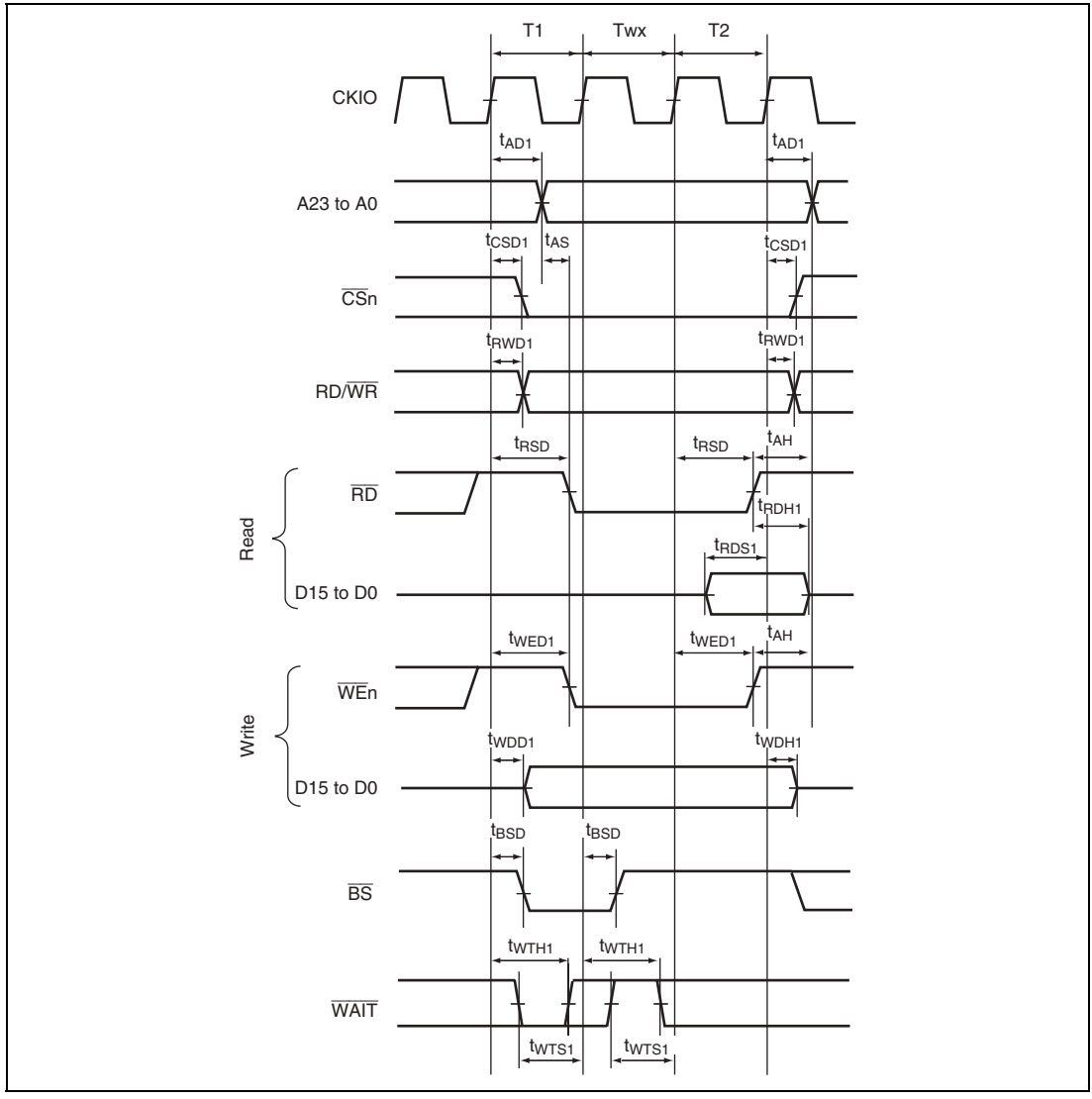


Figure 28.17 Basic Bus Cycle in Normal Space (Asynchronous External Wait 1 Input)

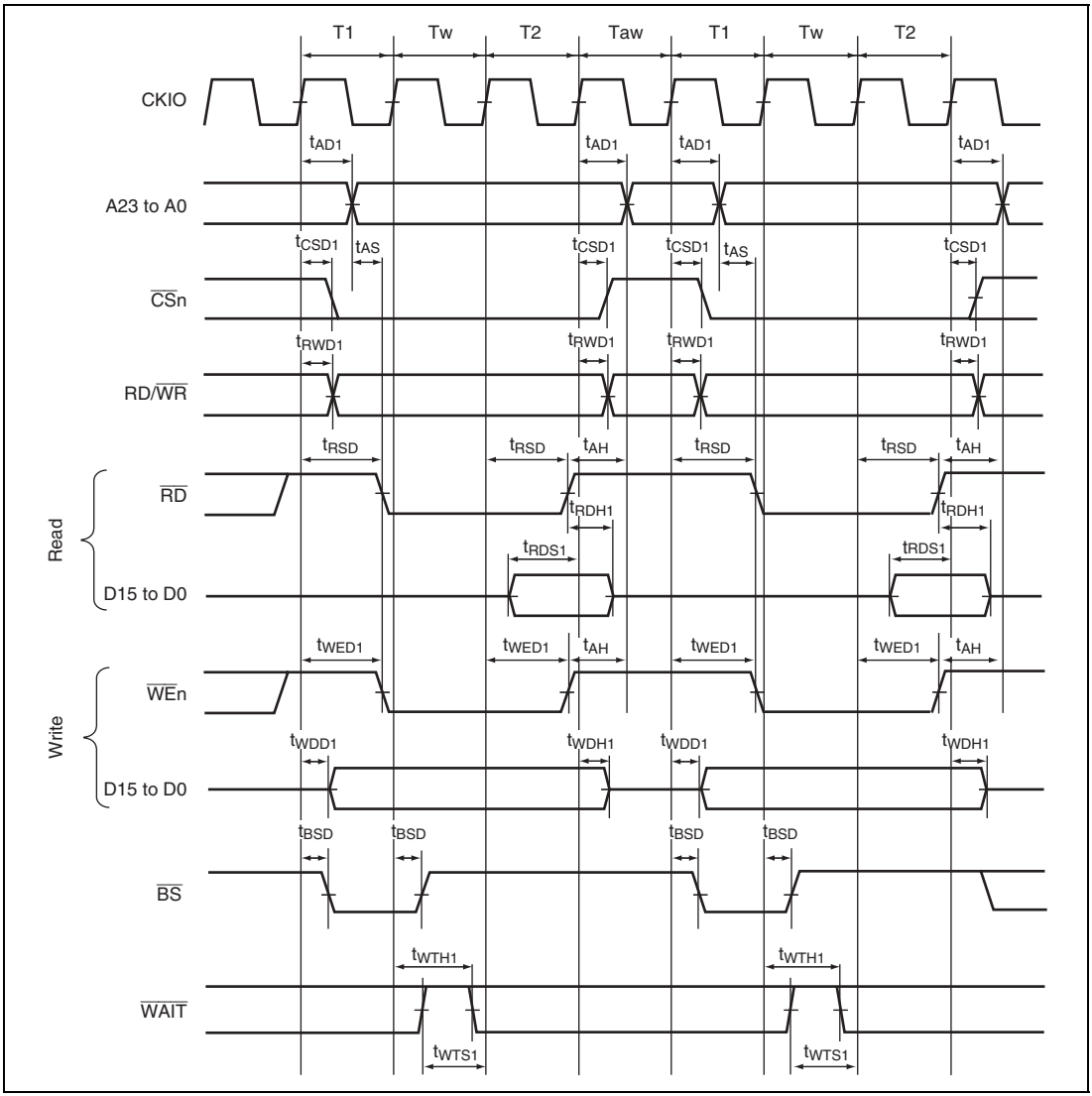


Figure 28.18 Basic Bus Cycle in Normal Space
(Software Wait 1, Asynchronous External Wait Valid (WM Bit = 0), No Idle Cycle)

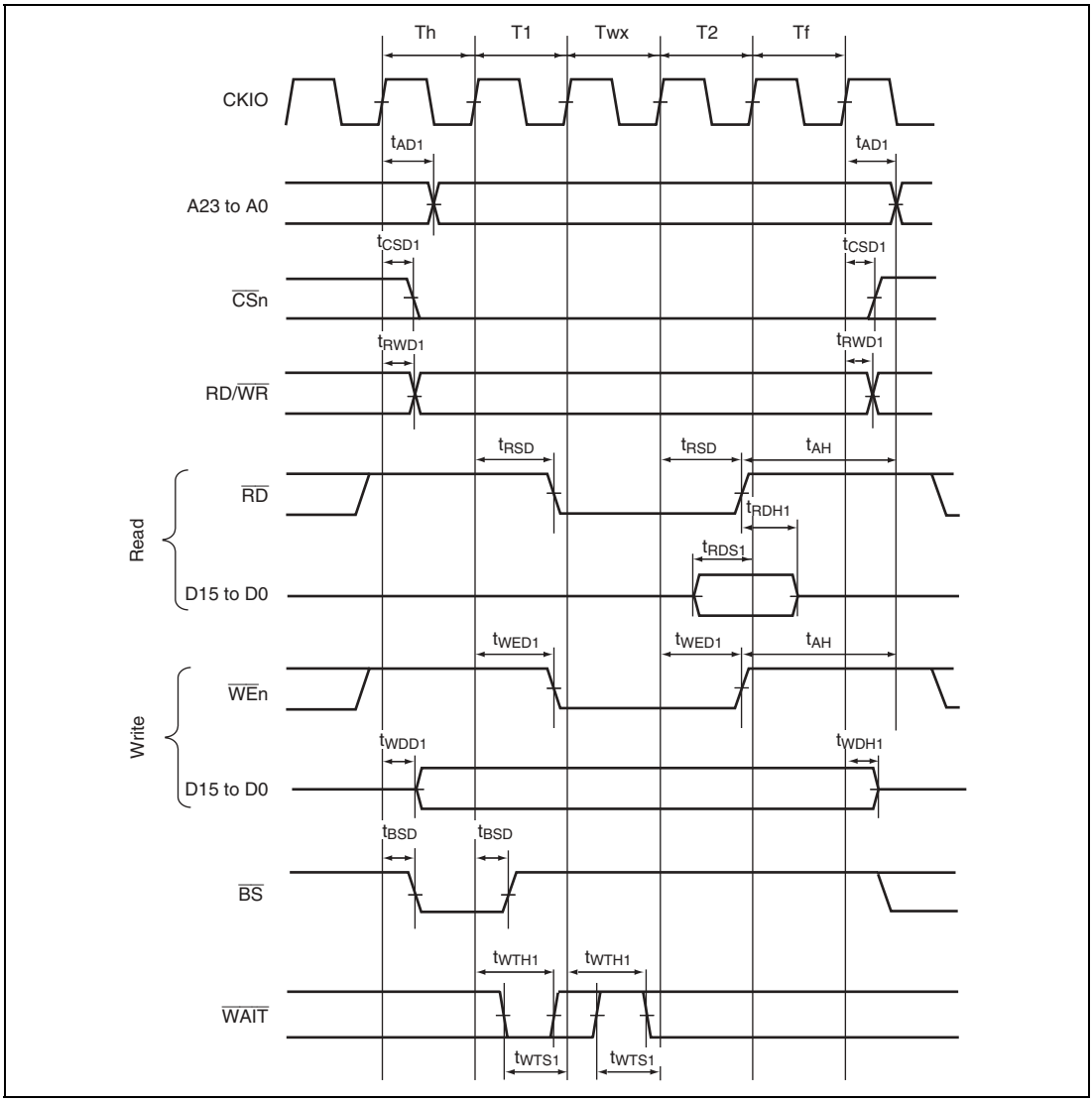


Figure 28.19 CS Extended Bus Cycle in Normal Space
(SW = 1 Cycle, HW = 1 Cycle, Asynchronous External Wait 1 Input)

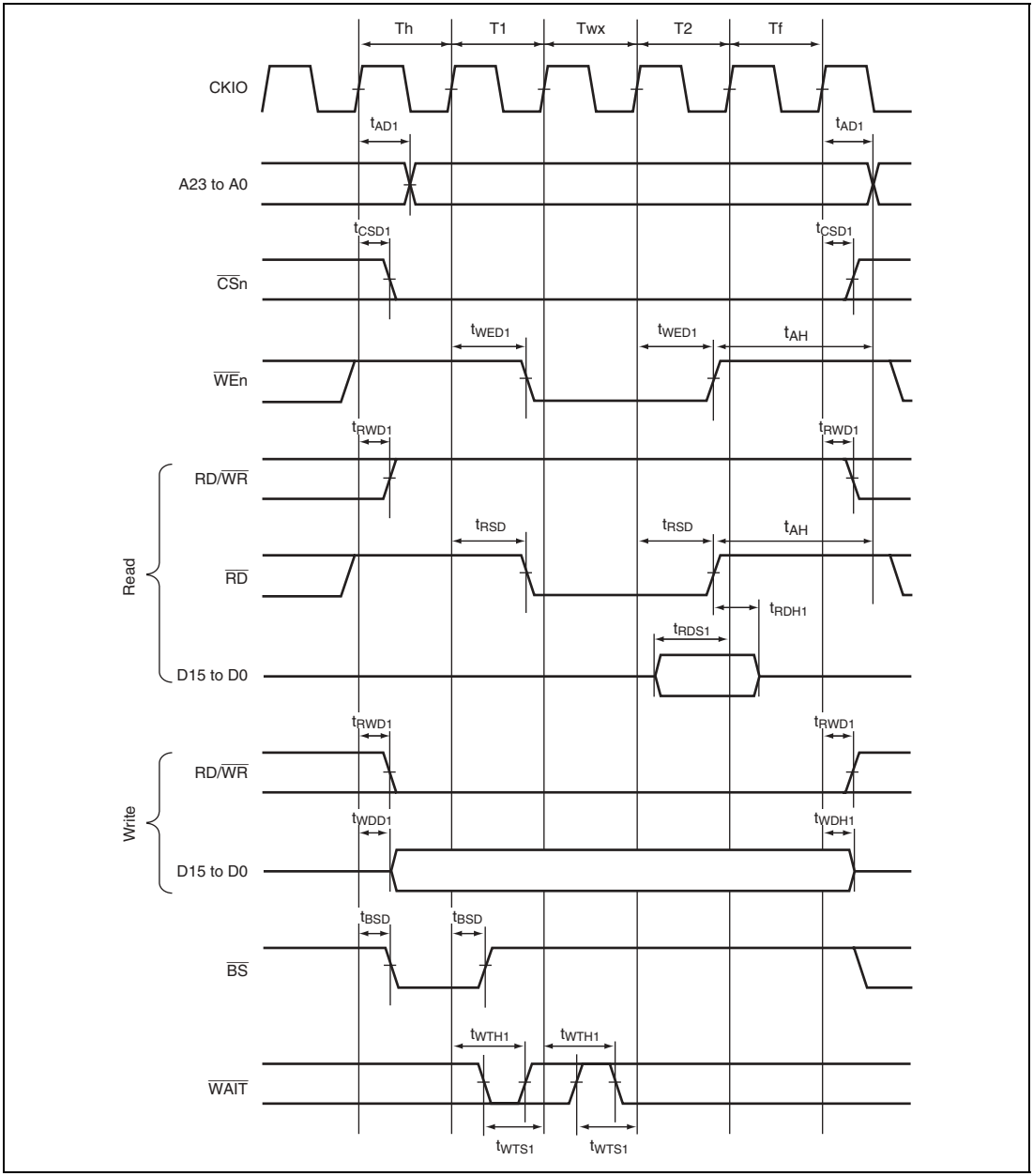


Figure 28.20 Bus Cycle of SRAM with Byte Selection (SW = 1 Cycle, HW = 1 Cycle, Asynchronous External Wait 1 Input, BAS = 0 (UB and LB in Write Cycle Controlled))

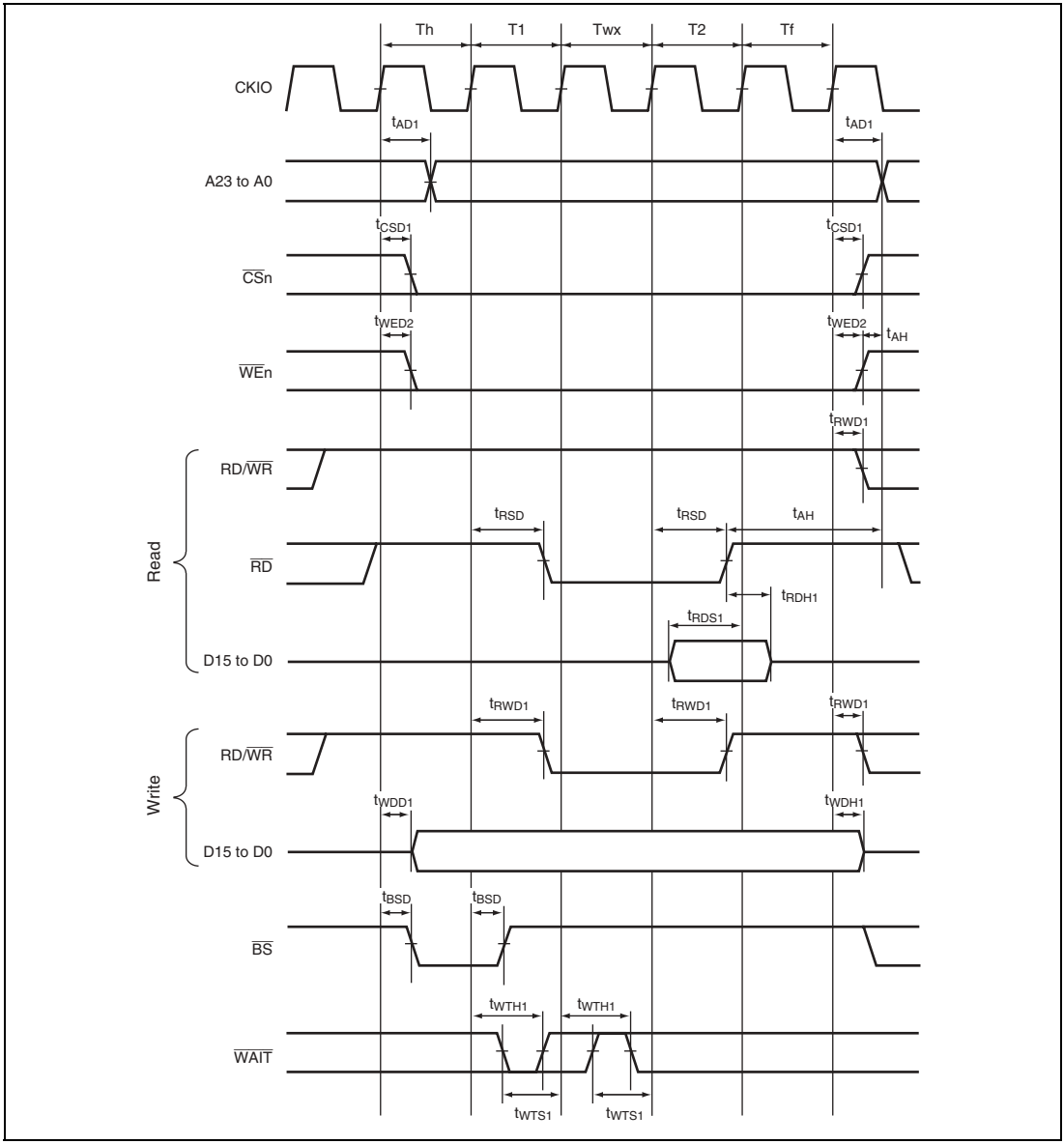


Figure 28.21 Bus Cycle of SRAM with Byte Selection (SW = 1 Cycle, HW = 1 Cycle, Asynchronous External Wait 1 Input, BAS = 1 (WE in Write Cycle Controlled))

28.3.5 Burst ROM Timing

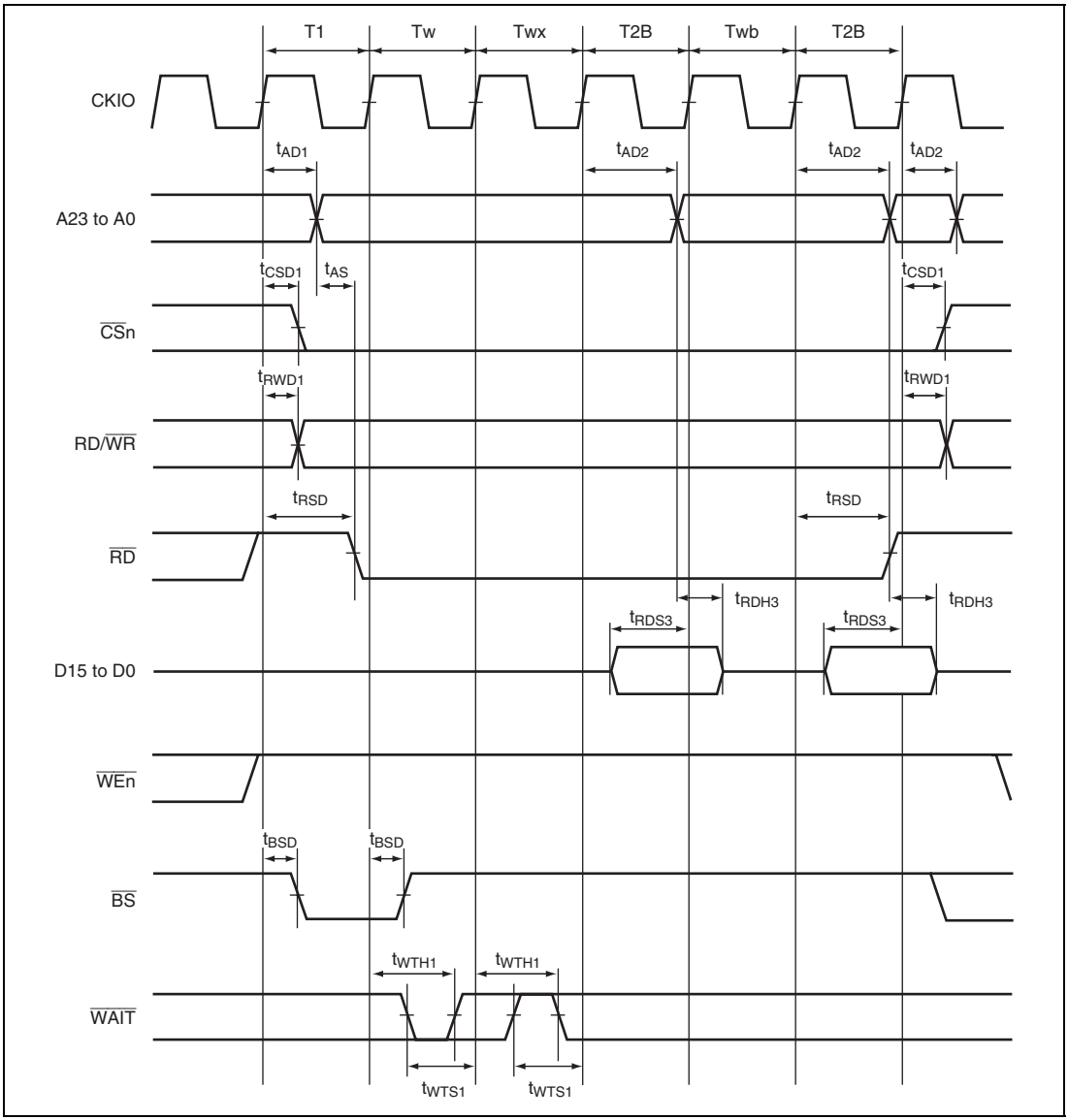
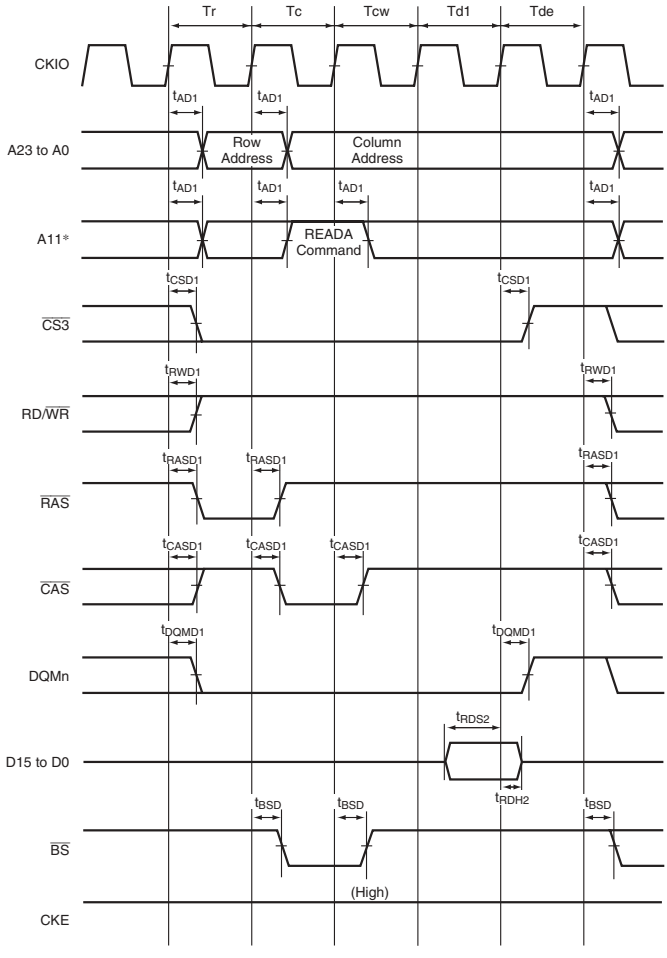


Figure 28.22 Read Bus Cycle of Burst ROM
(Software Wait 1, Asynchronous External Wait 1 Input, Burst Wait 1, Number of Burst 2)

28.3.6 Synchronous DRAM Timing



Note: * Address pin that is connected to A10 of SDRAM

Figure 28.23 Single Read Bus Cycle of Synchronous DRAM (Auto Precharge Mode, CAS Latency 2, TRCD = 1 Cycle, TRP = 1 Cycle)

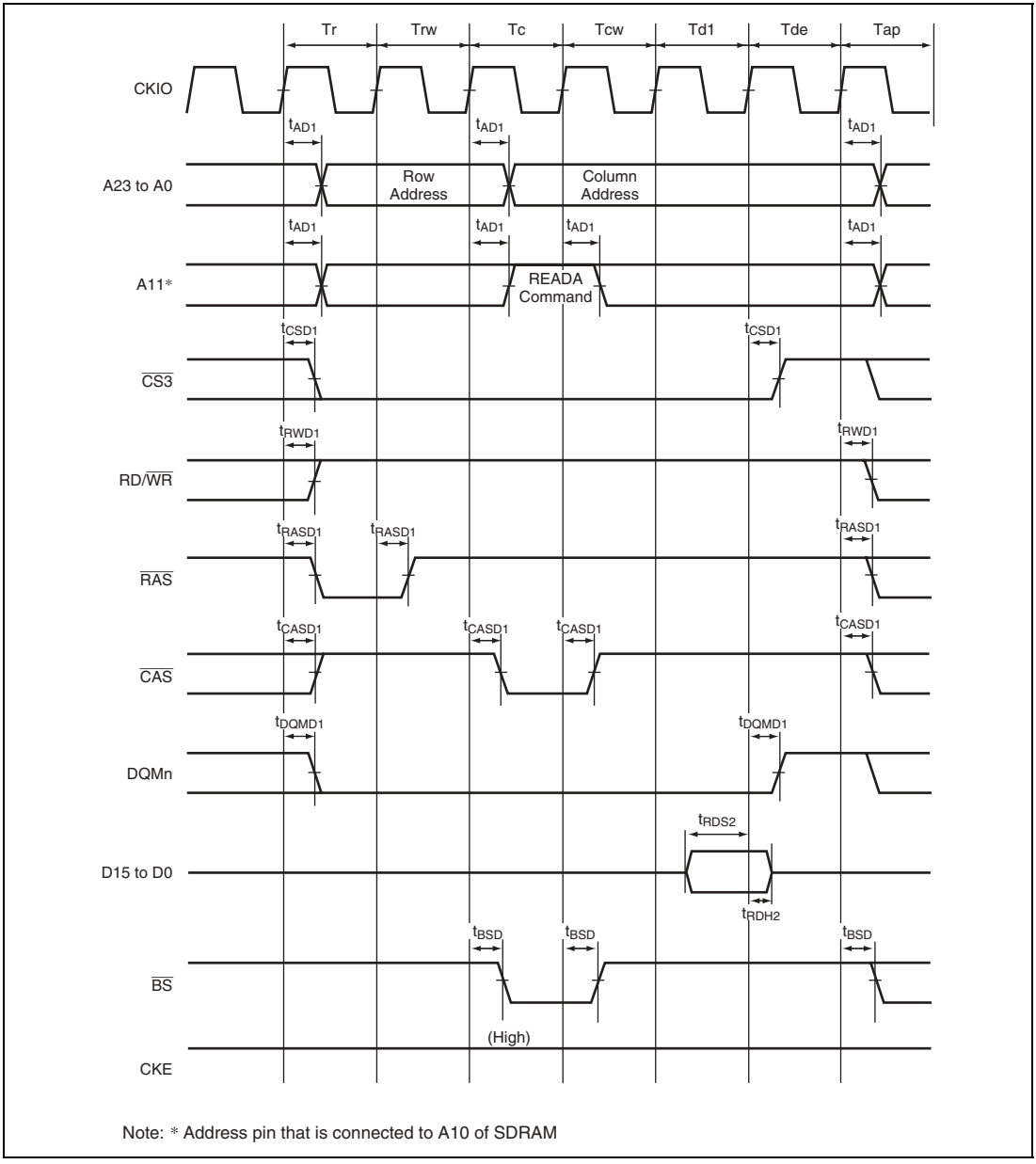
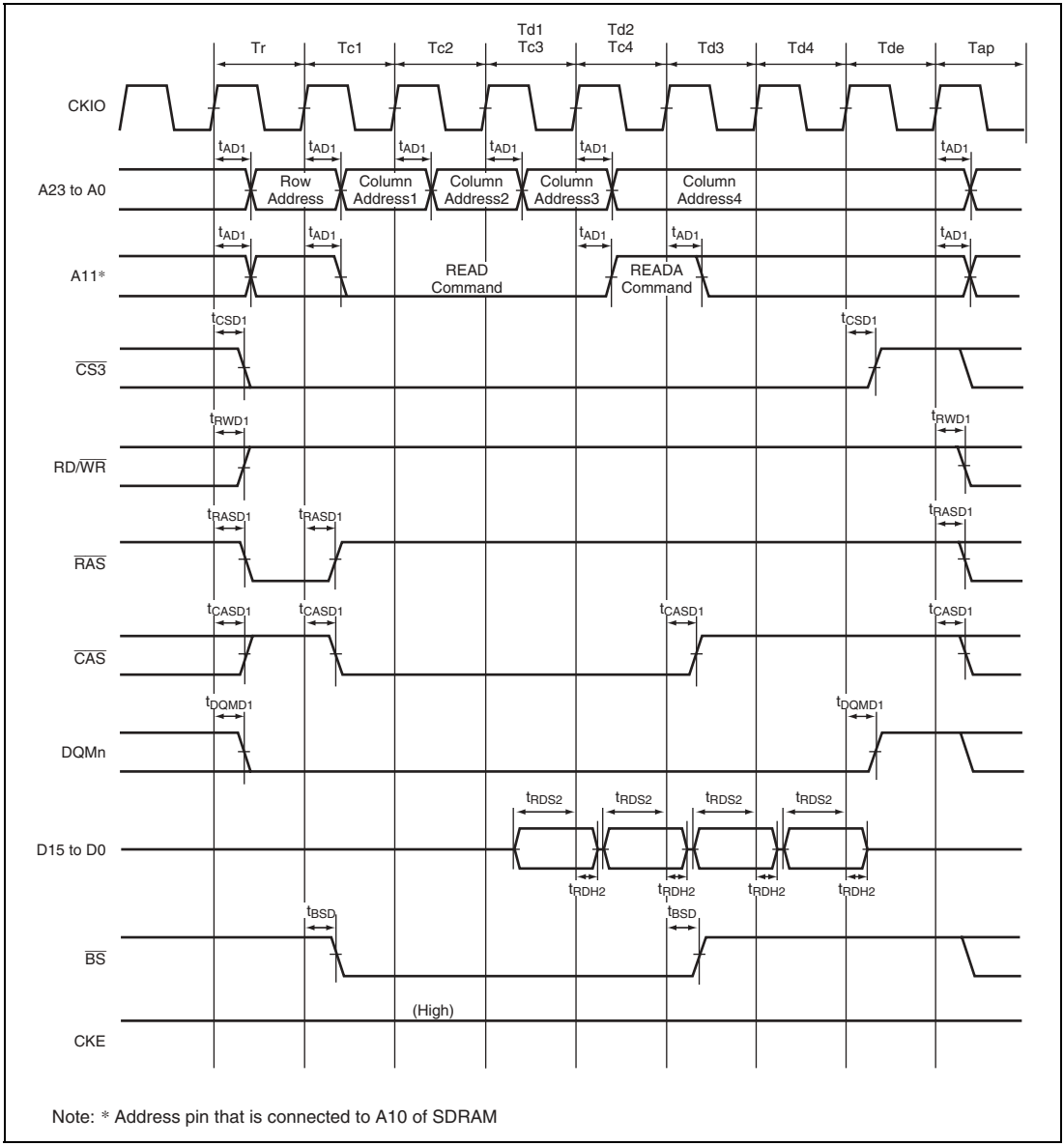


Figure 28.24 Single Read Bus Cycle of Synchronous DRAM (Auto Precharge Mode, CAS Latency 2, TRCD = 2 Cycle, TRP = 2 Cycle)



Note: * Address pin that is connected to A10 of SDRAM

Figure 28.25 Burst Read Bus Cycle of Synchronous DRAM (Single Read × 4)
(Auto Precharge Mode, CAS Latency 2, TRCD = 1 Cycle, TRP = 2 Cycle)

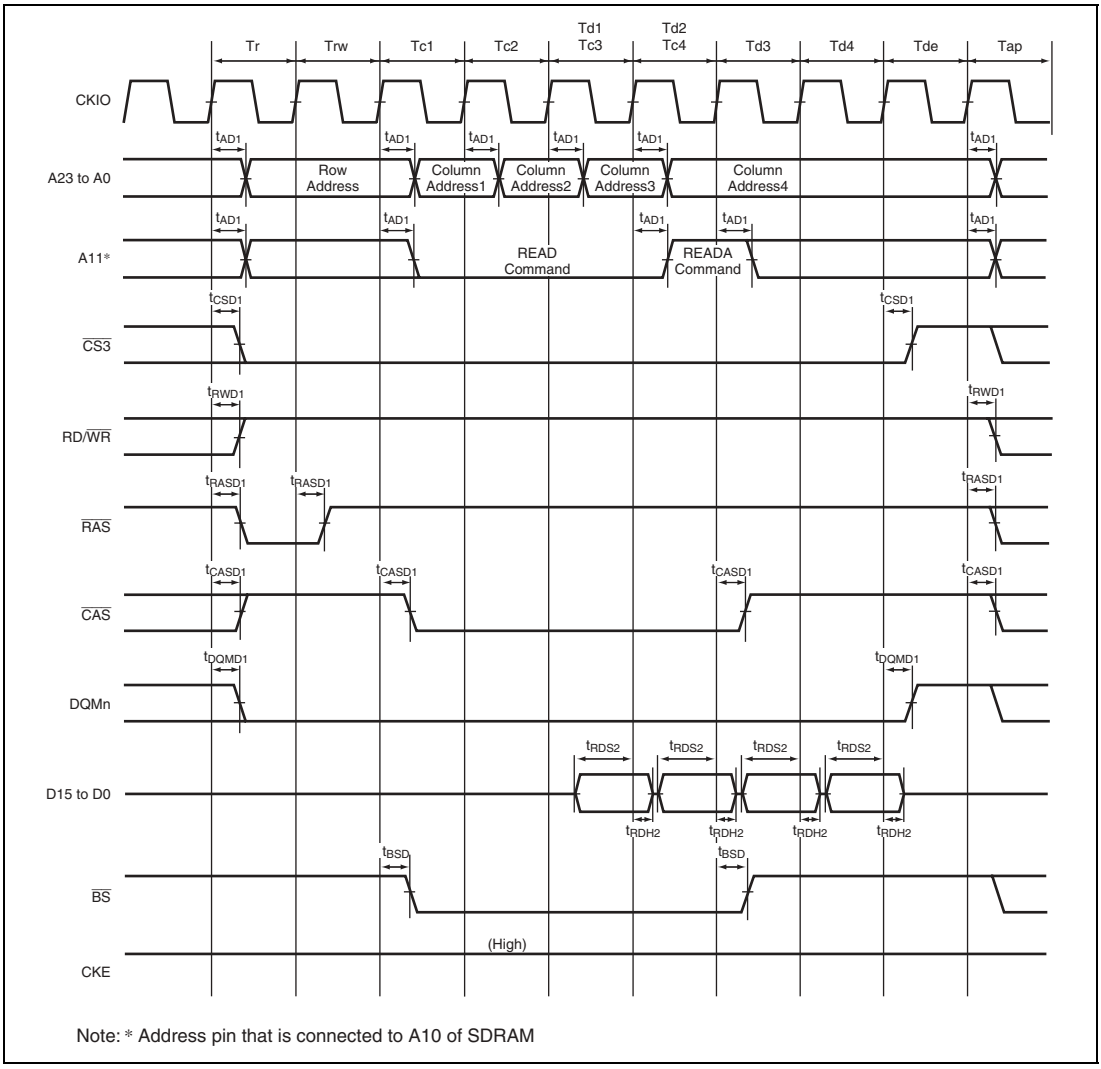
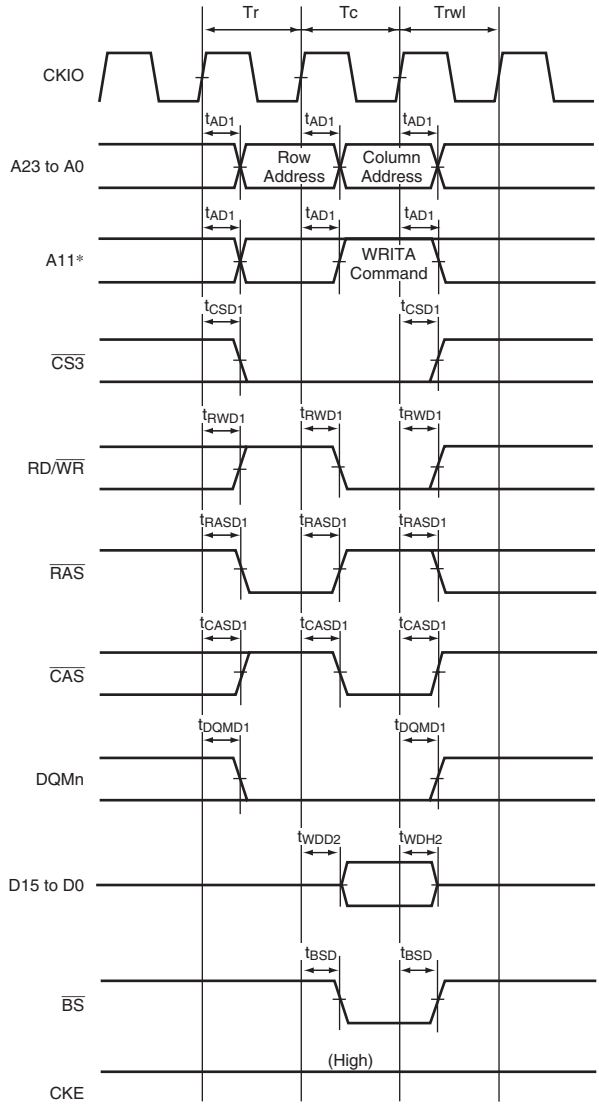
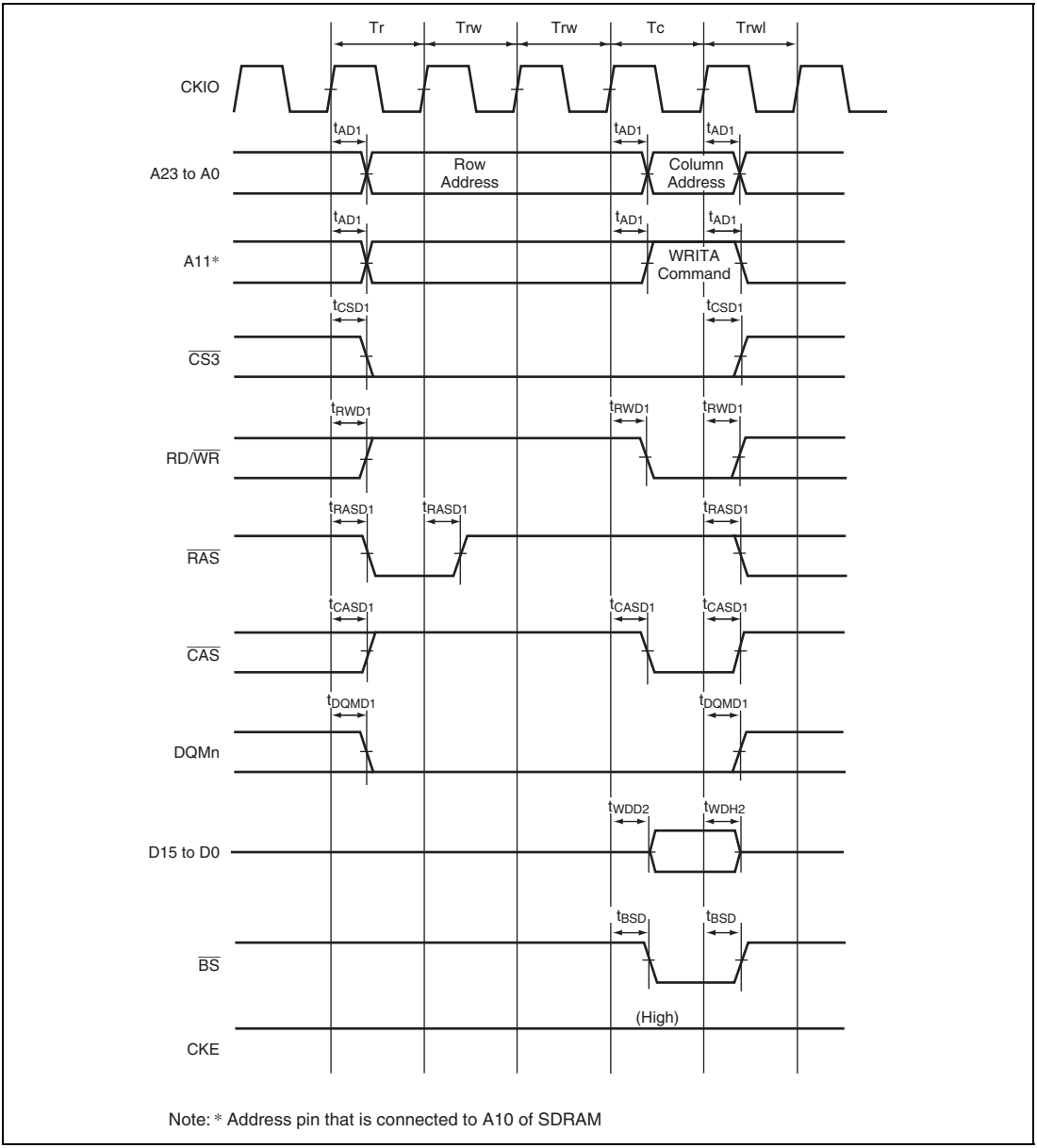


Figure 28.26 Burst Read Bus Cycle of Synchronous DRAM (Single Read × 4) (Auto Precharge Mode, CAS Latency 2, TRCD = 2 Cycle, TRP = 1 Cycle)



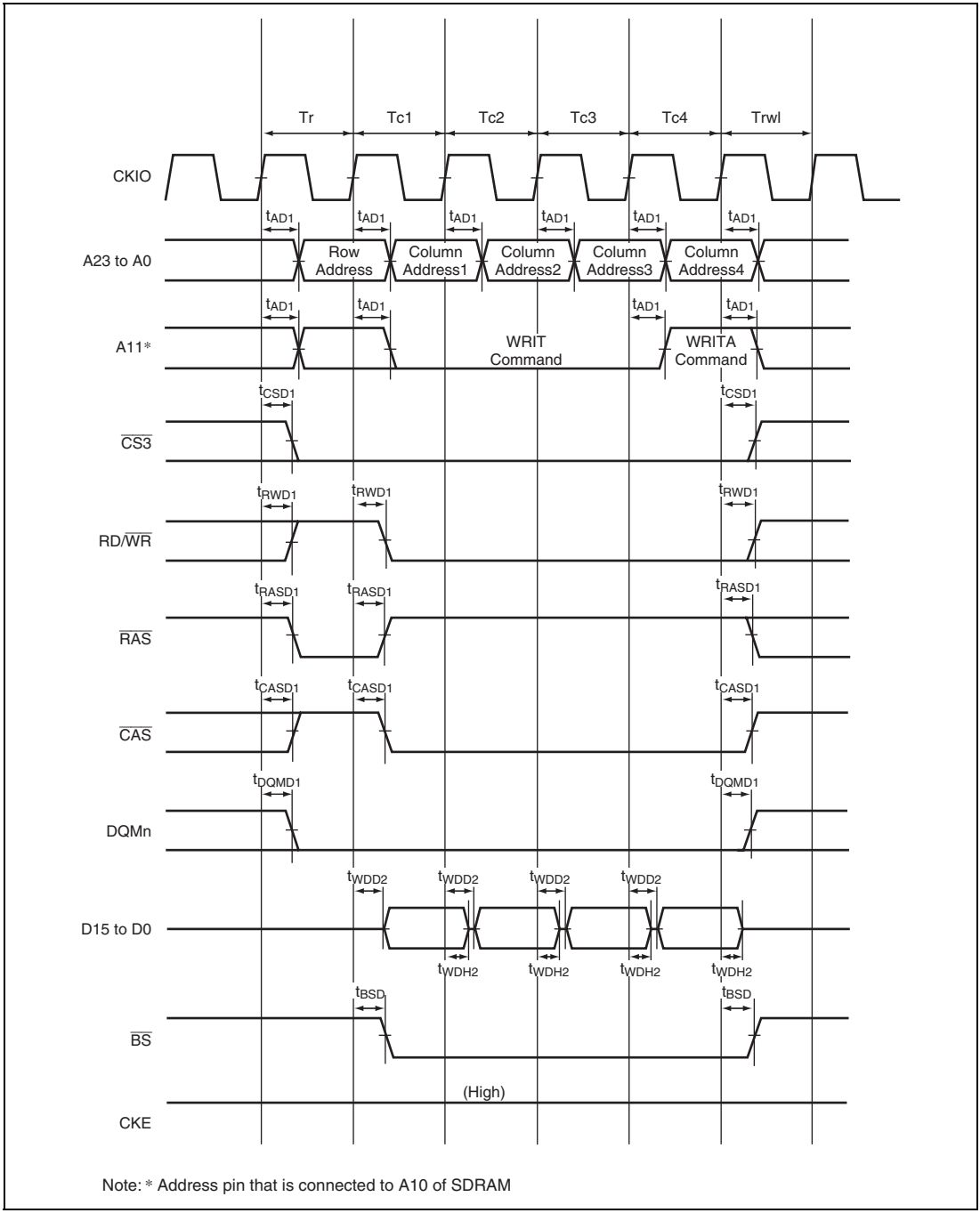
Note: * Address pin that is connected to A10 of SDRAM

Figure 28.27 Single Write Bus Cycle of Synchronous DRAM (Auto Precharge Mode, TRWL = 1 Cycle)



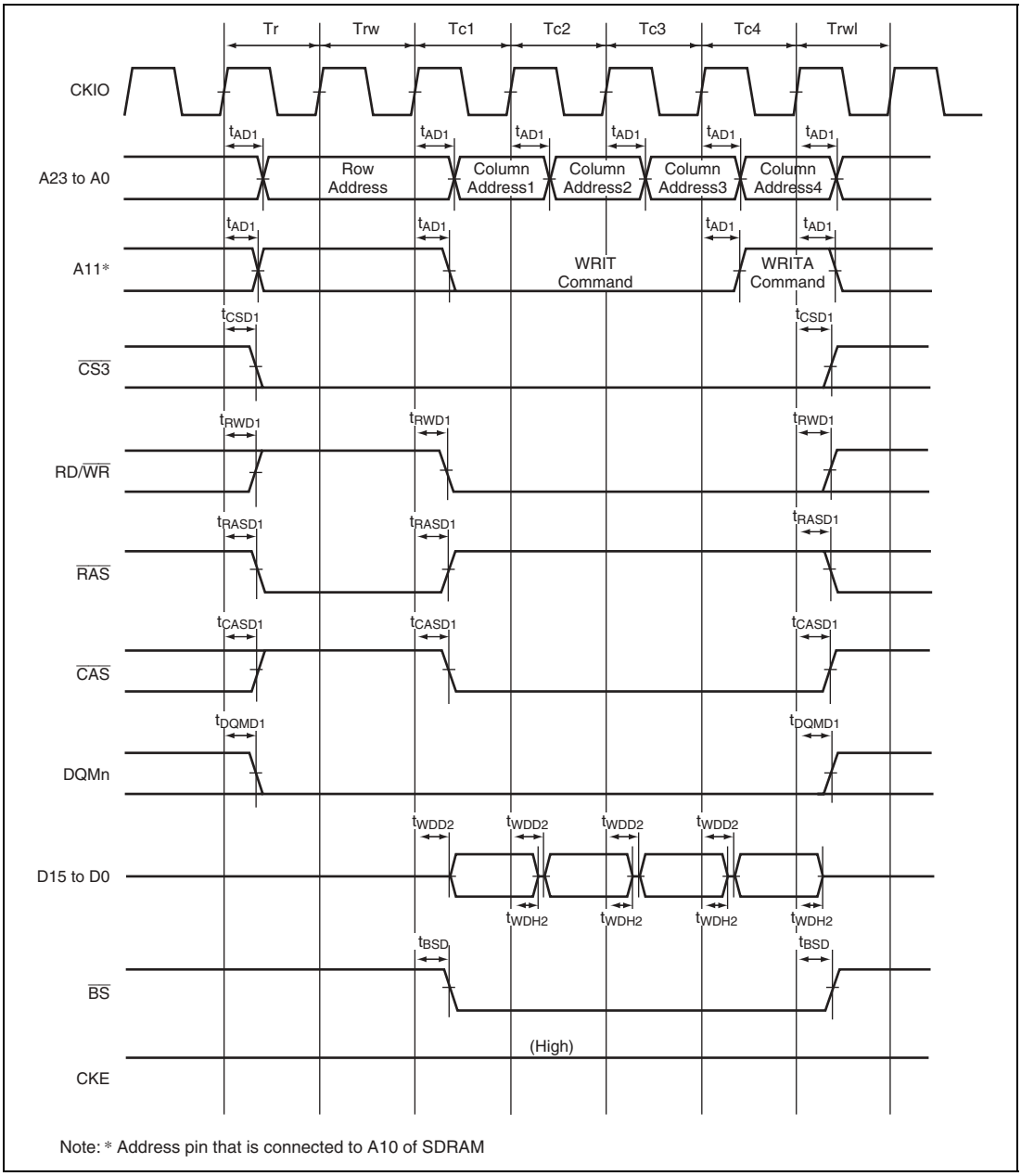
Note: * Address pin that is connected to A10 of SDRAM

Figure 28.28 Single Write Bus Cycle of Synchronous DRAM (Auto Precharge Mode, TRCD = 3 Cycle, TRWL = 1 Cycle)



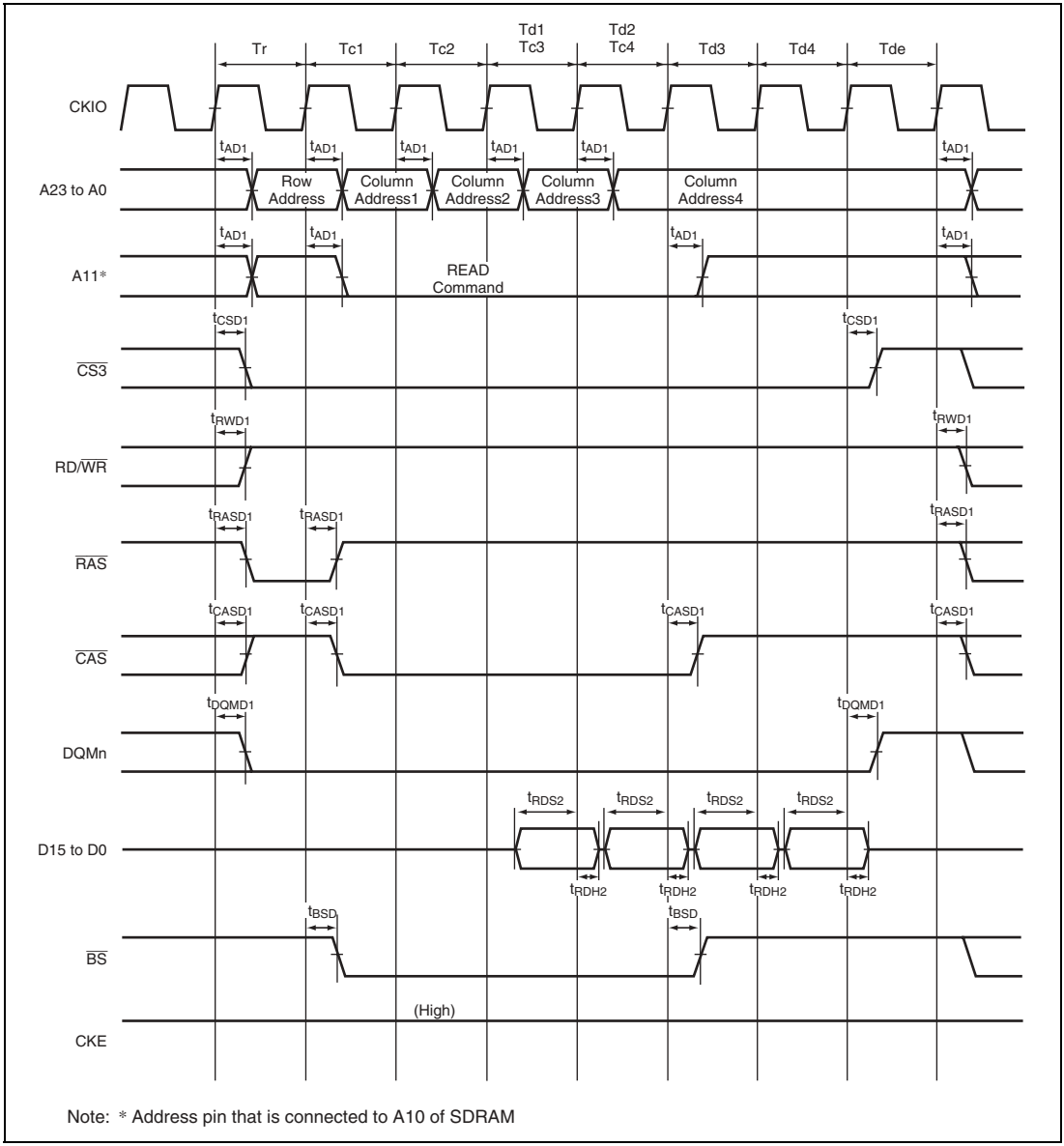
Note: * Address pin that is connected to A10 of SDRAM

Figure 28.29 Burst Write Bus Cycle of Synchronous DRAM (Single Write × 4) (Auto Precharge Mode, TRCD = 1 Cycle, TRWL = 1 Cycle)



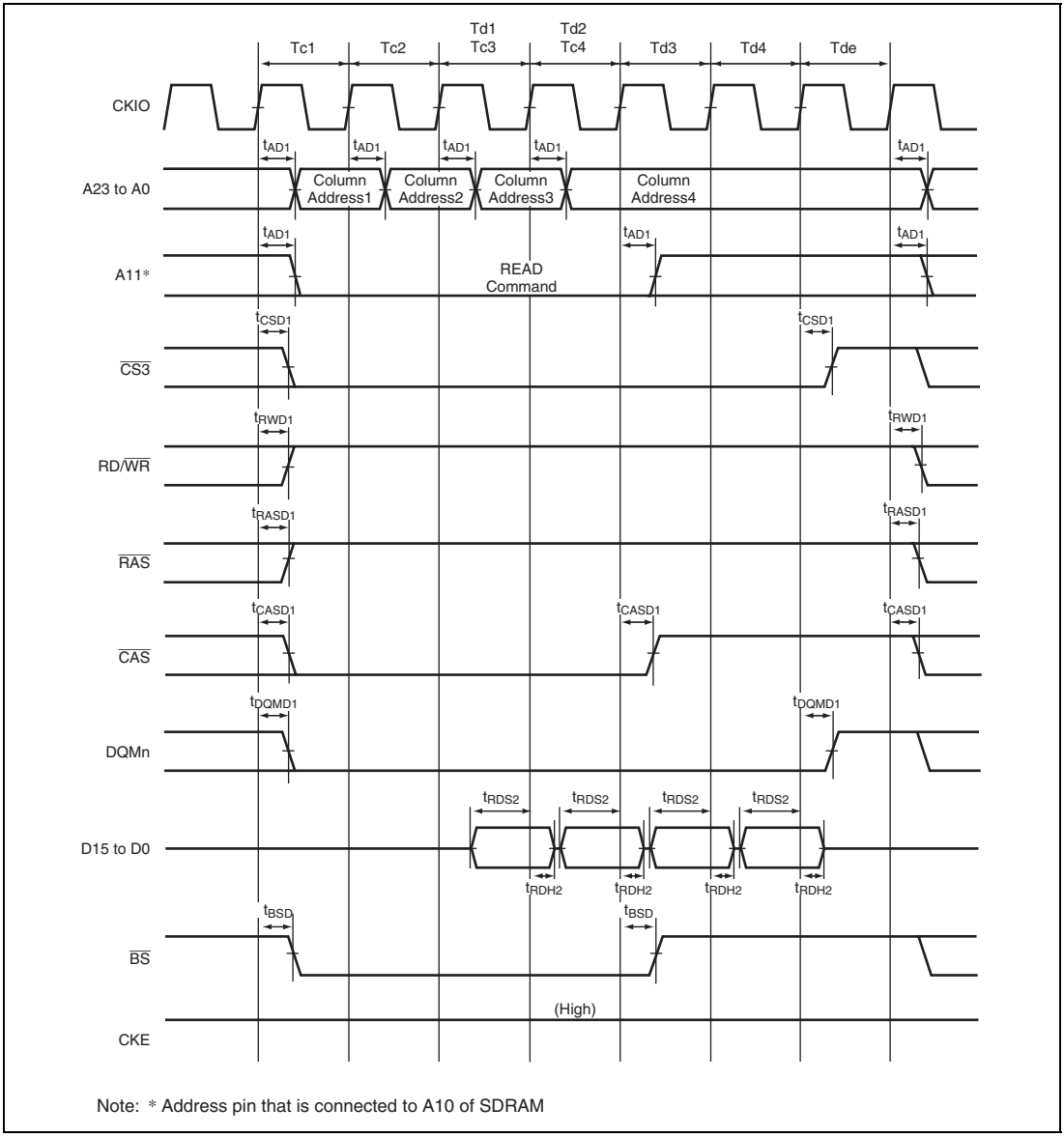
Note: * Address pin that is connected to A10 of SDRAM

Figure 28.30 Burst Write Bus Cycle of Synchronous DRAM (Single Write × 4) (Auto Precharge Mode, TRCD = 1 Cycle, TRWL = 1 Cycle)



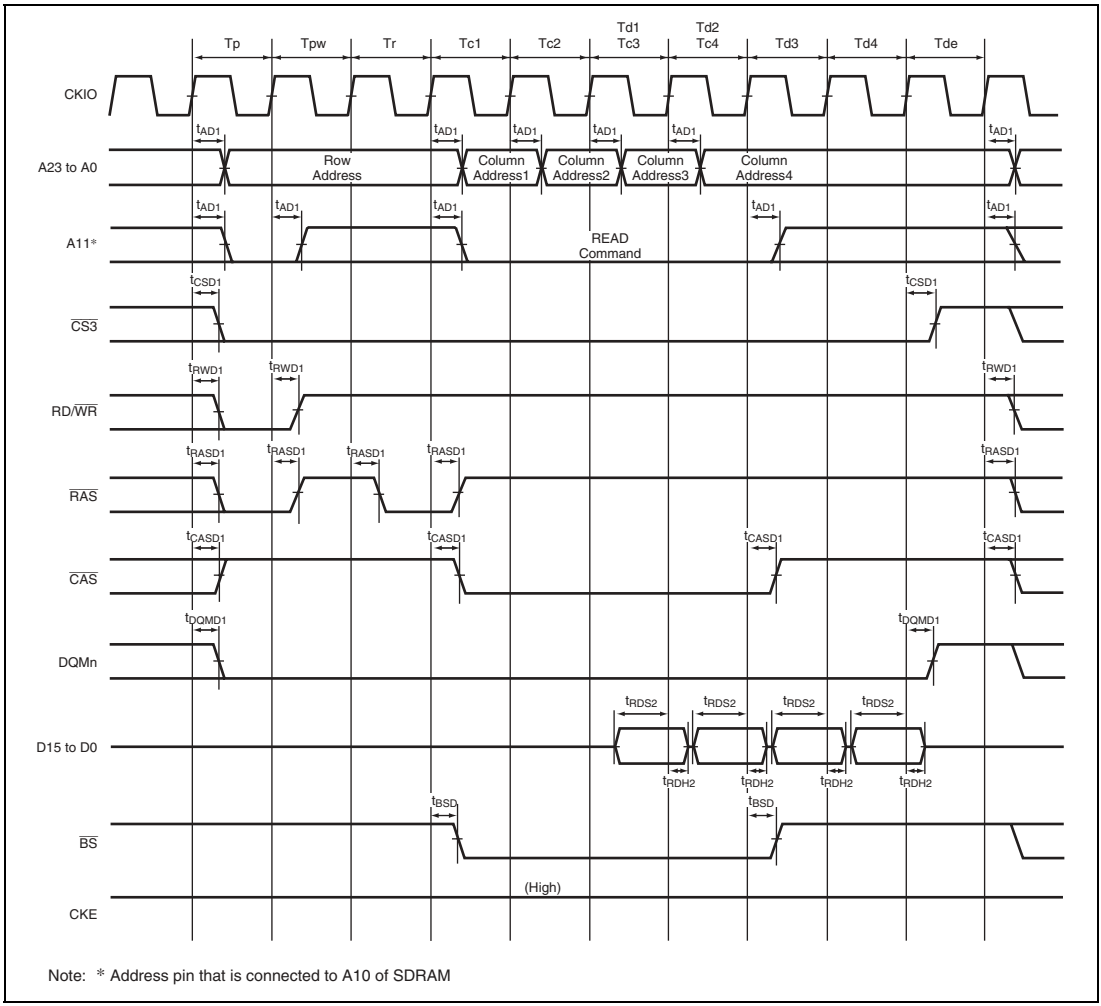
Note: * Address pin that is connected to A10 of SDRAM

Figure 28.31 Burst Read Bus Cycle of Synchronous DRAM (Single Read × 4)
(Bank Active Mode: ACTV + READ Command, CAS Latency 2, TRCD = 1 Cycle)



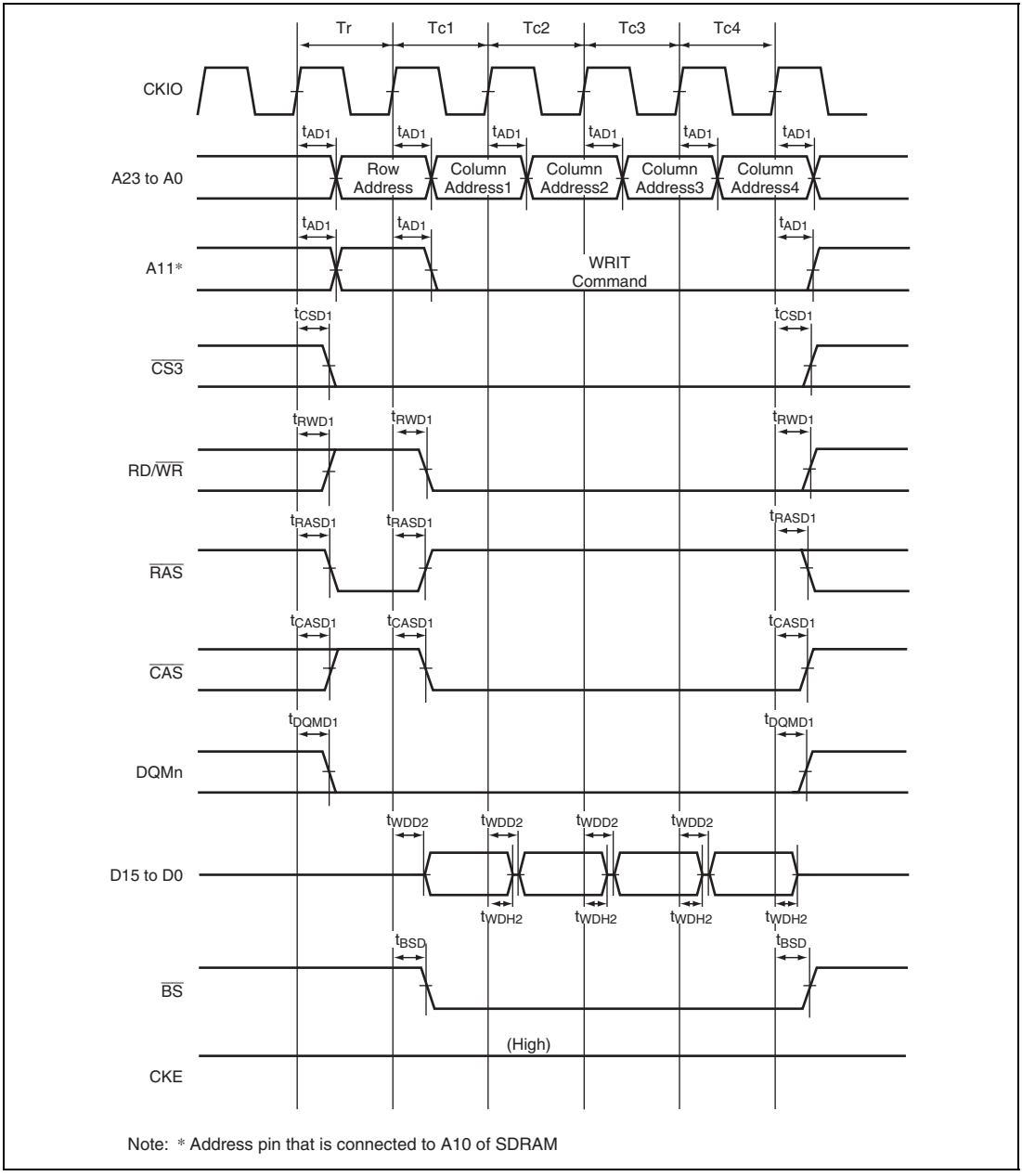
Note: * Address pin that is connected to A10 of SDRAM

Figure 28.32 Burst Read Bus Cycle of Synchronous DRAM (Single Read × 4)
(Bank Active Mode: READ Command,
Same Row Address, CAS Latency 2, TRCD = 1 Cycle)



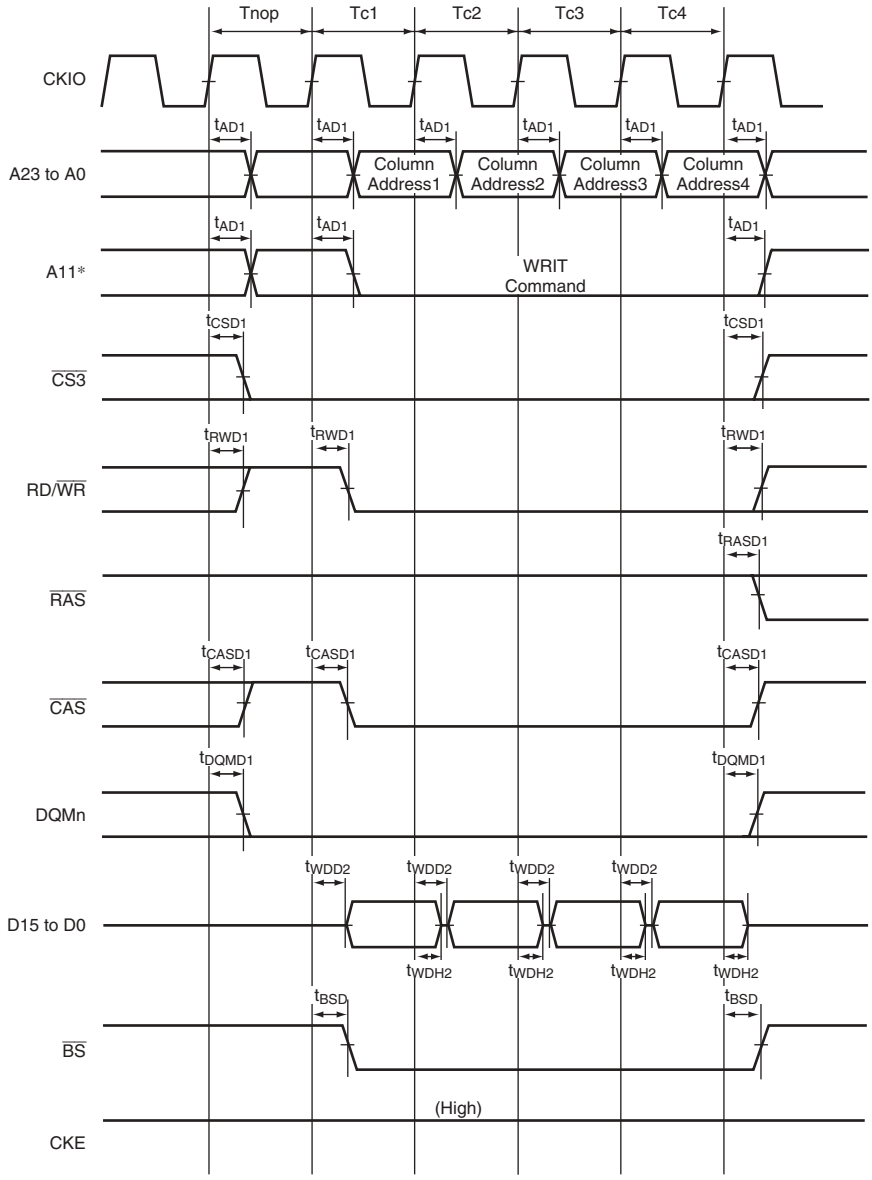
Note: * Address pin that is connected to A10 of SDRAM

Figure 28.33 Burst Read Bus Cycle of Synchronous DRAM (Single Read × 4)
(Bank Active Mode: PRE + ACTV + READ Command,
Different Row Address, CAS Latency 2, TRCD = 1 Cycle)



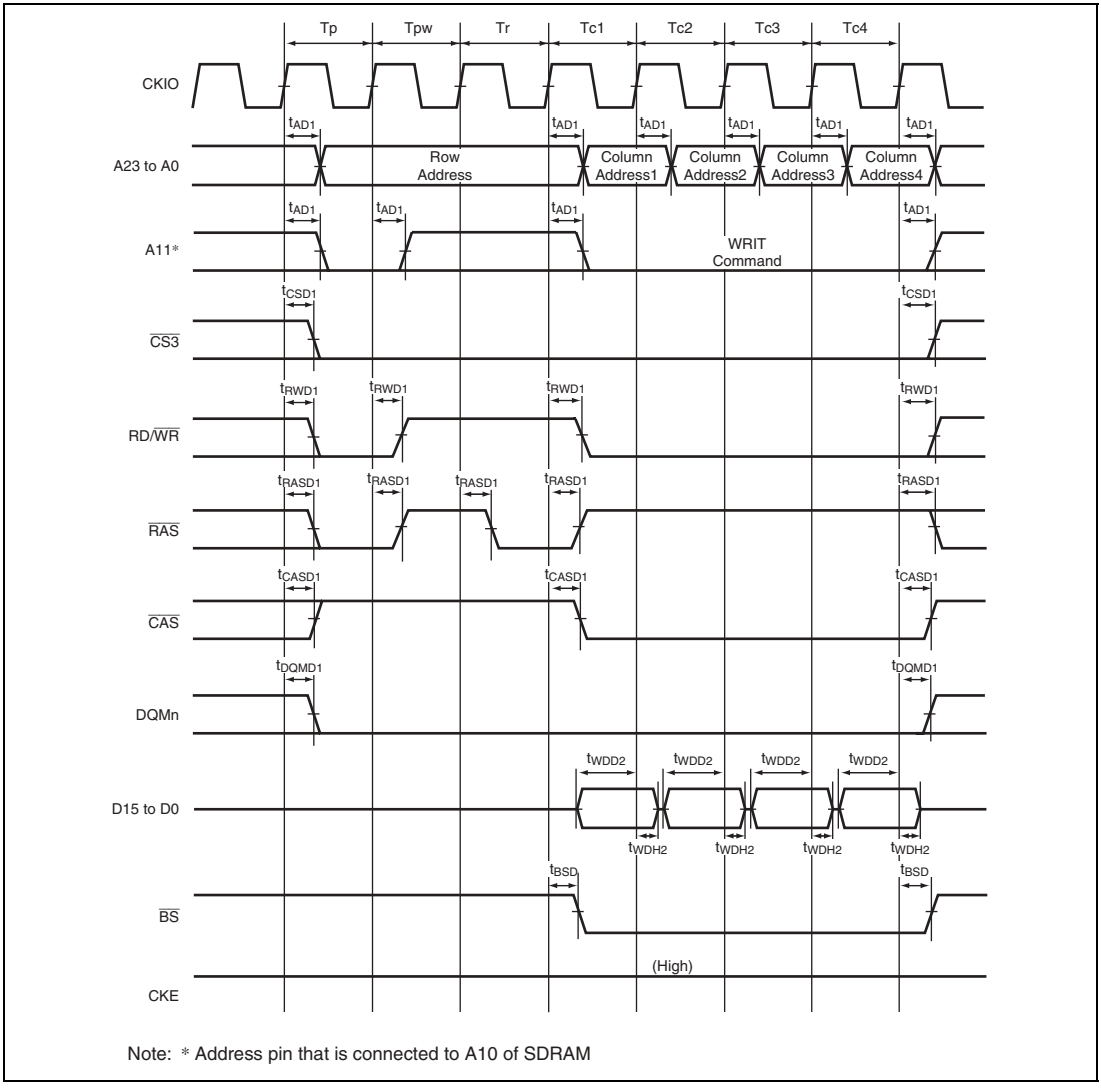
Note: * Address pin that is connected to A10 of SDRAM

Figure 28.34 Burst Write Bus Cycle of Synchronous DRAM (Single Write × 4)
(Bank Active Mode: ACTV + WRIT Command, TRCD = 1 Cycle)



Note: * Address pin that is connected to A10 of SDRAM

Figure 28.35 Burst Write Bus Cycle of Synchronous DRAM (Single Write × 4) (Bank Active Mode: ACTV + WRIT Command, TRCD = 1 Cycle)



Note: * Address pin that is connected to A10 of SDRAM

Figure 28.36 Burst Write Bus Cycle of Synchronous DRAM (Single Write × 4) (Bank Active Mode: PRE + ACTV + WRIT Command, TRCD = 1 Cycle)

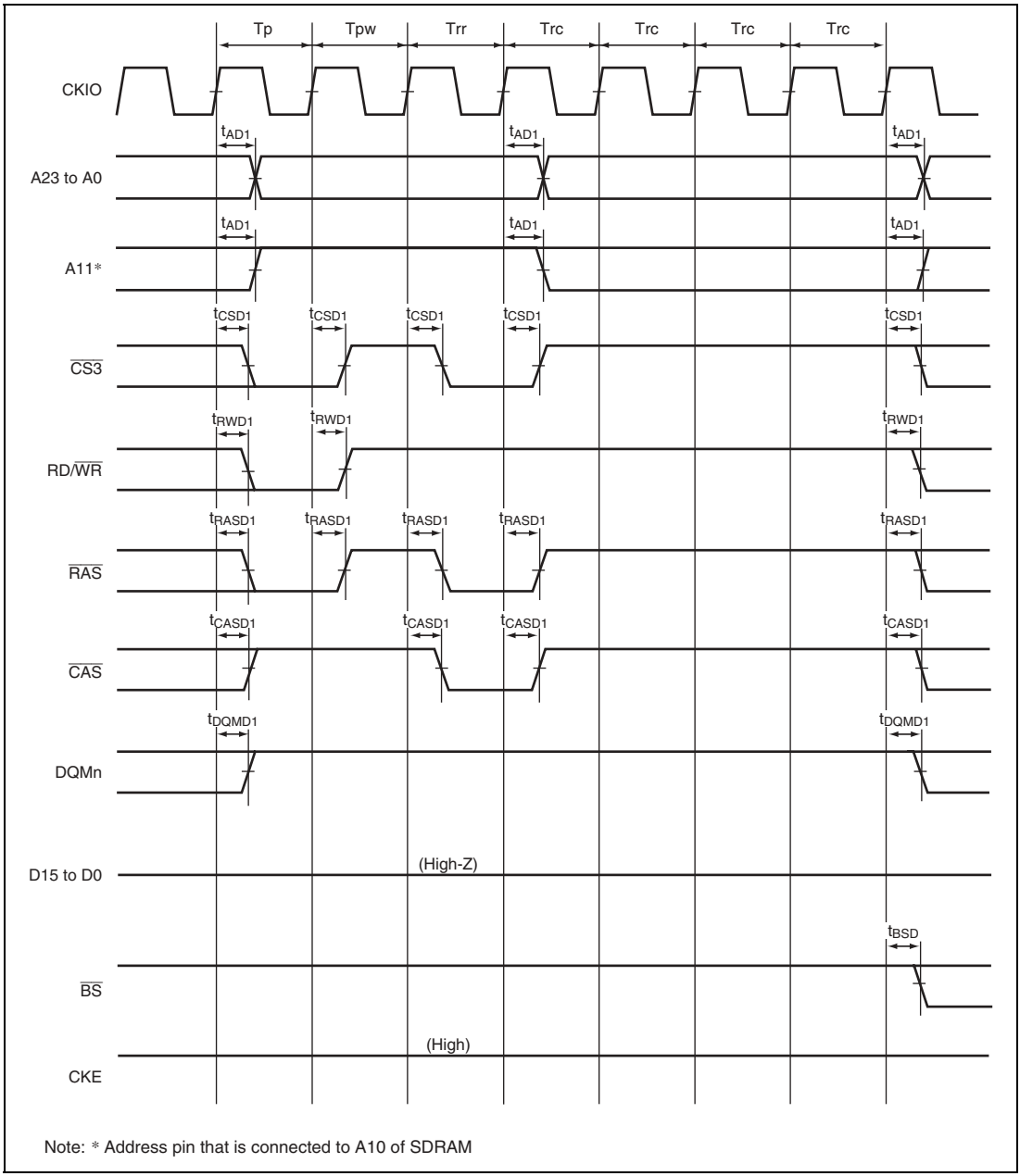


Figure 28.37 Auto Refresh Timing of Synchronous DRAM (TRP = 2 Cycle)

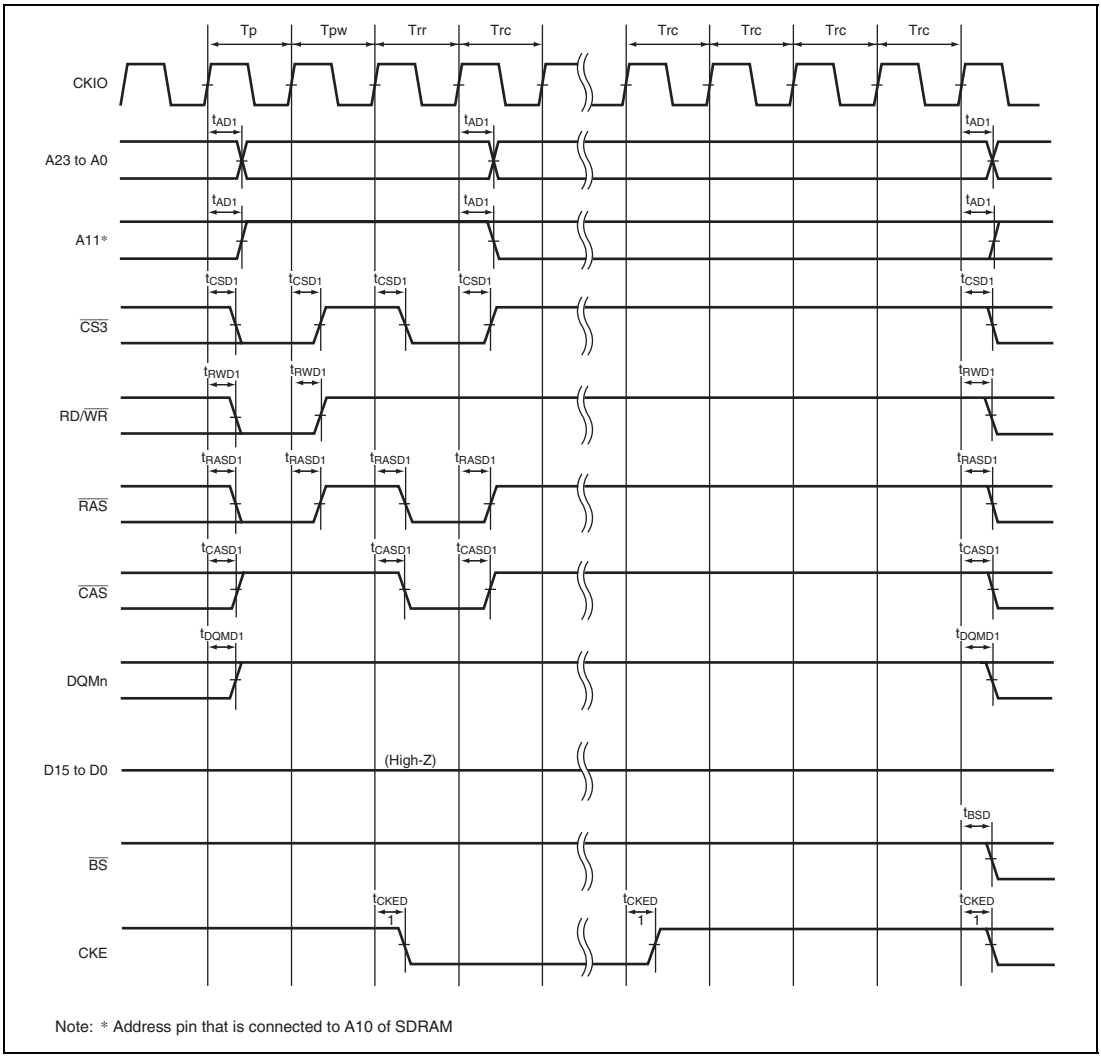
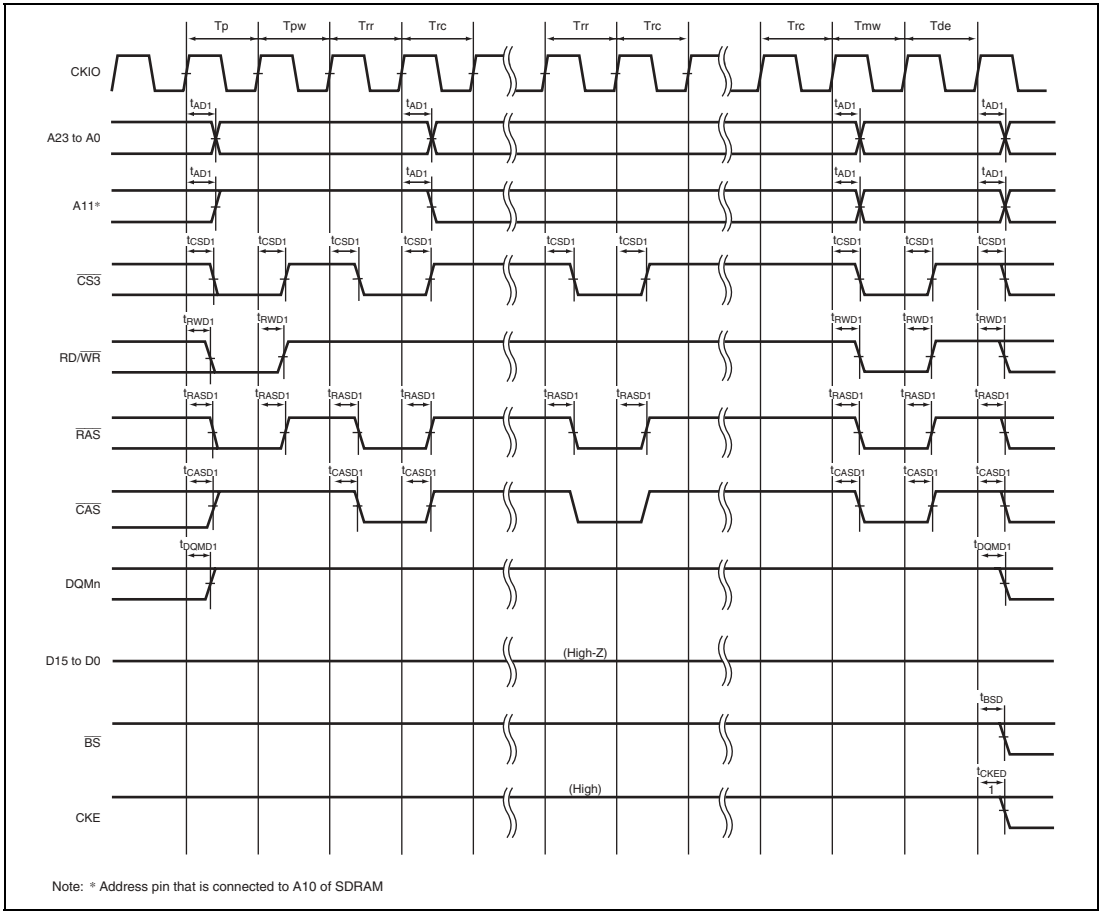


Figure 28.38 Self Refresh Timing of Synchronous DRAM (TRP = 2 Cycle)



Note: * Address pin that is connected to A10 of SDRAM

Figure 28.39 Power-On Sequence of Synchronous DRAM (Mode Write Timing, TRP = 2 Cycle)

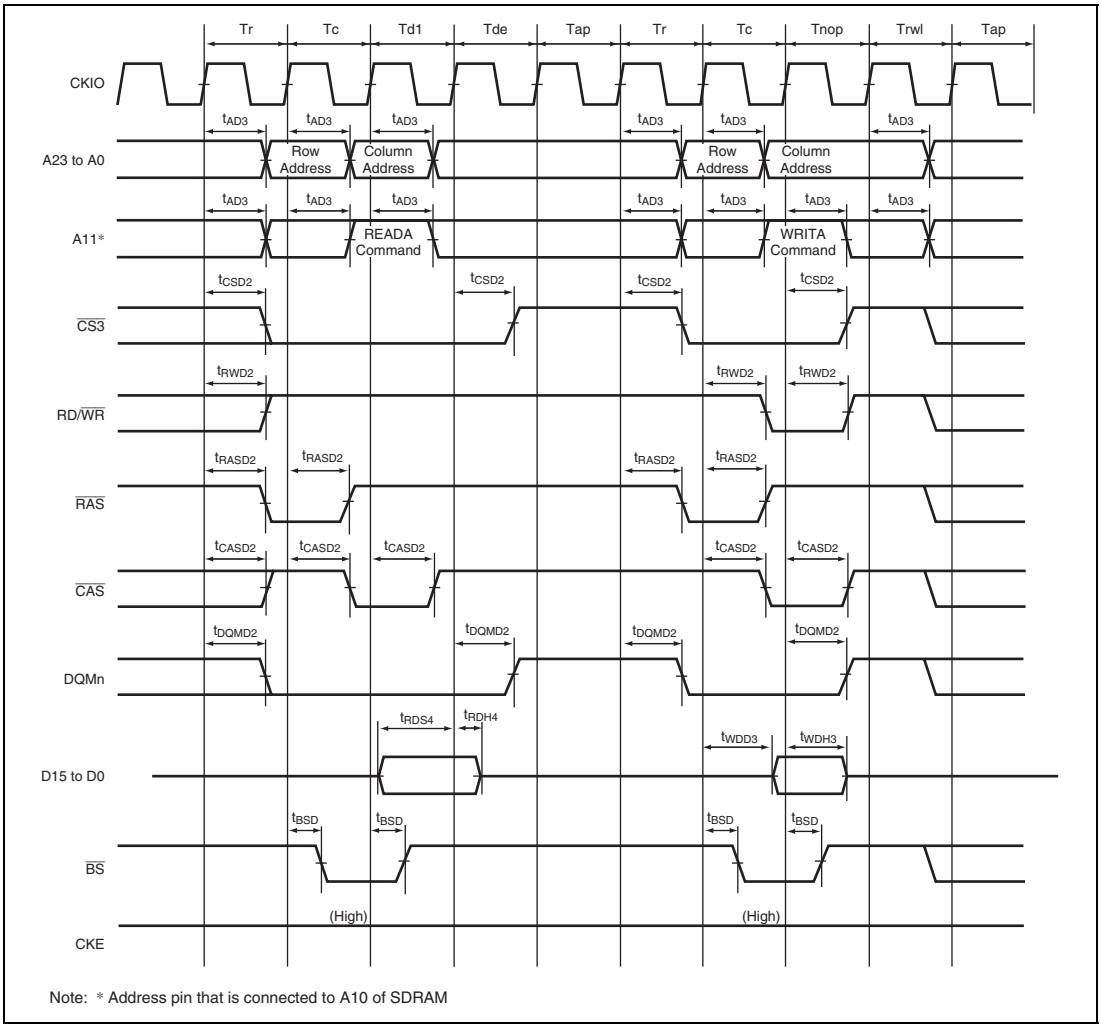


Figure 28.40 Access Timing in Low-Frequency Mode of Synchronous DRAM (Auto Precharge Mode, TRWL = 1 Cycle)

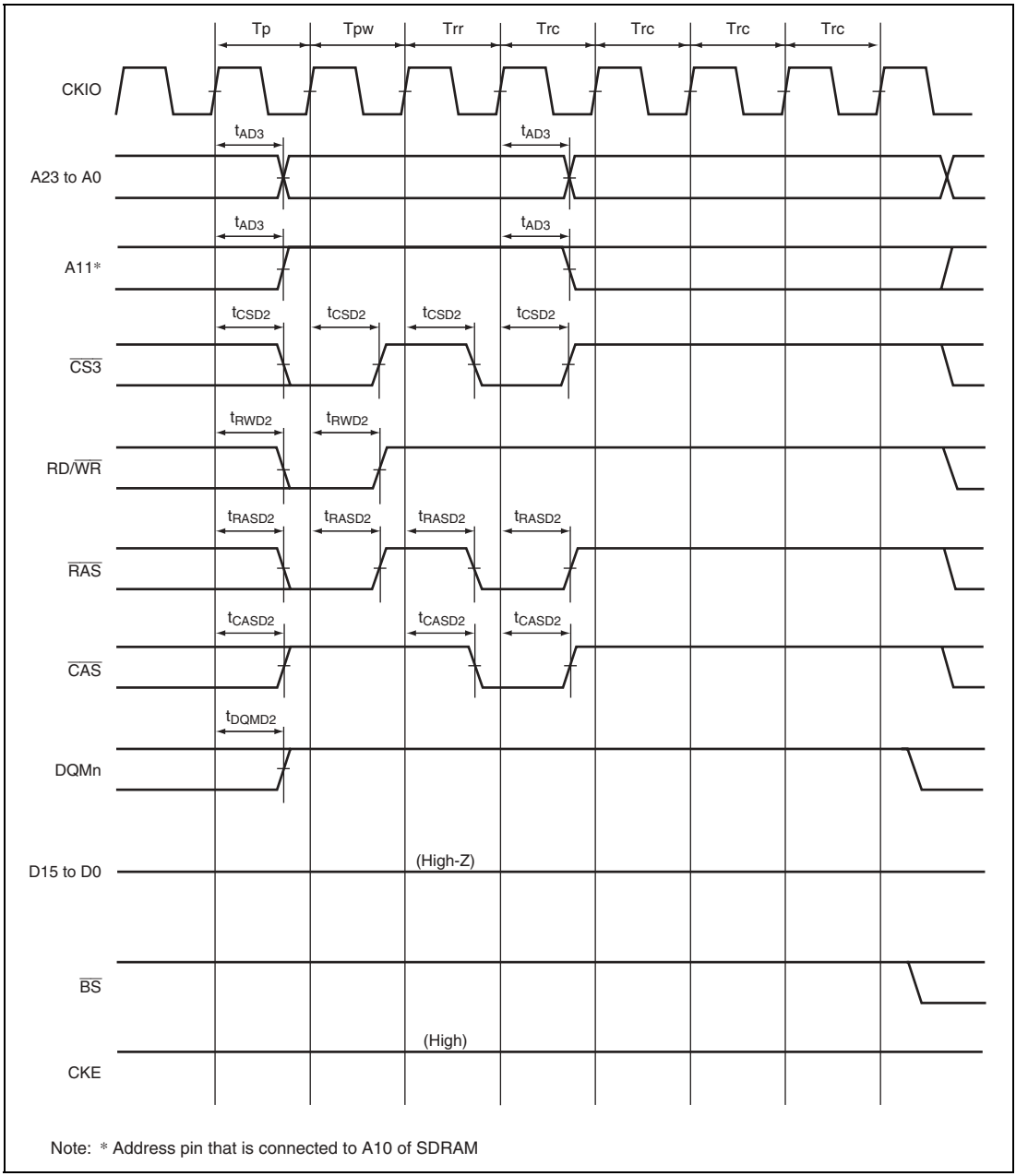


Figure 28.41 Auto Refresh Timing in Low-Frequency Mode of Synchronous DRAM (TRP = 2 Cycle)

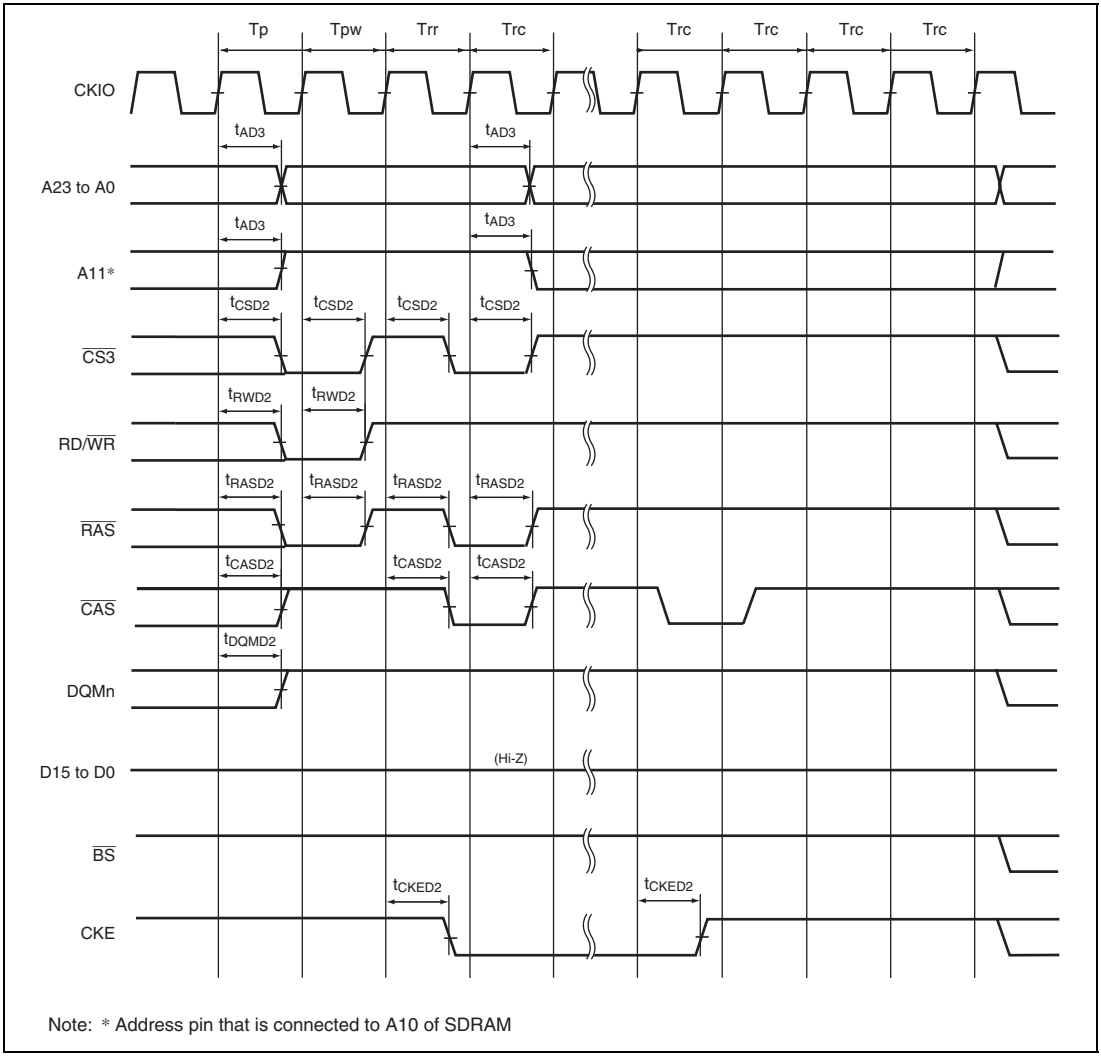
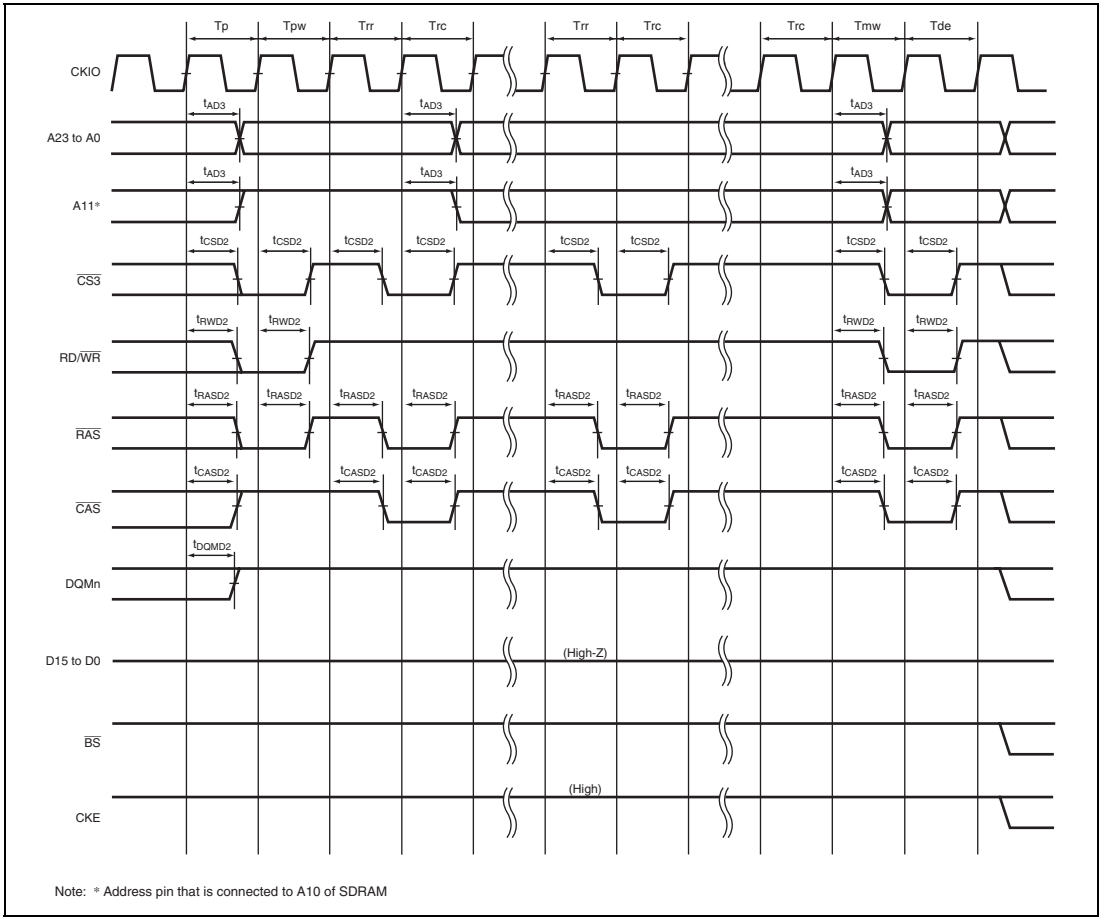


Figure 28.42 Self Refresh Timing in Low-Frequency Mode of Synchronous DRAM (TRP = 2 Cycle)



Note: * Address pin that is connected to A10 of SDRAM

Figure 28.43 Power-On Sequence in Low-Frequency Mode of Synchronous DRAM (Mode Write Timing, TRP = 2 Cycle)

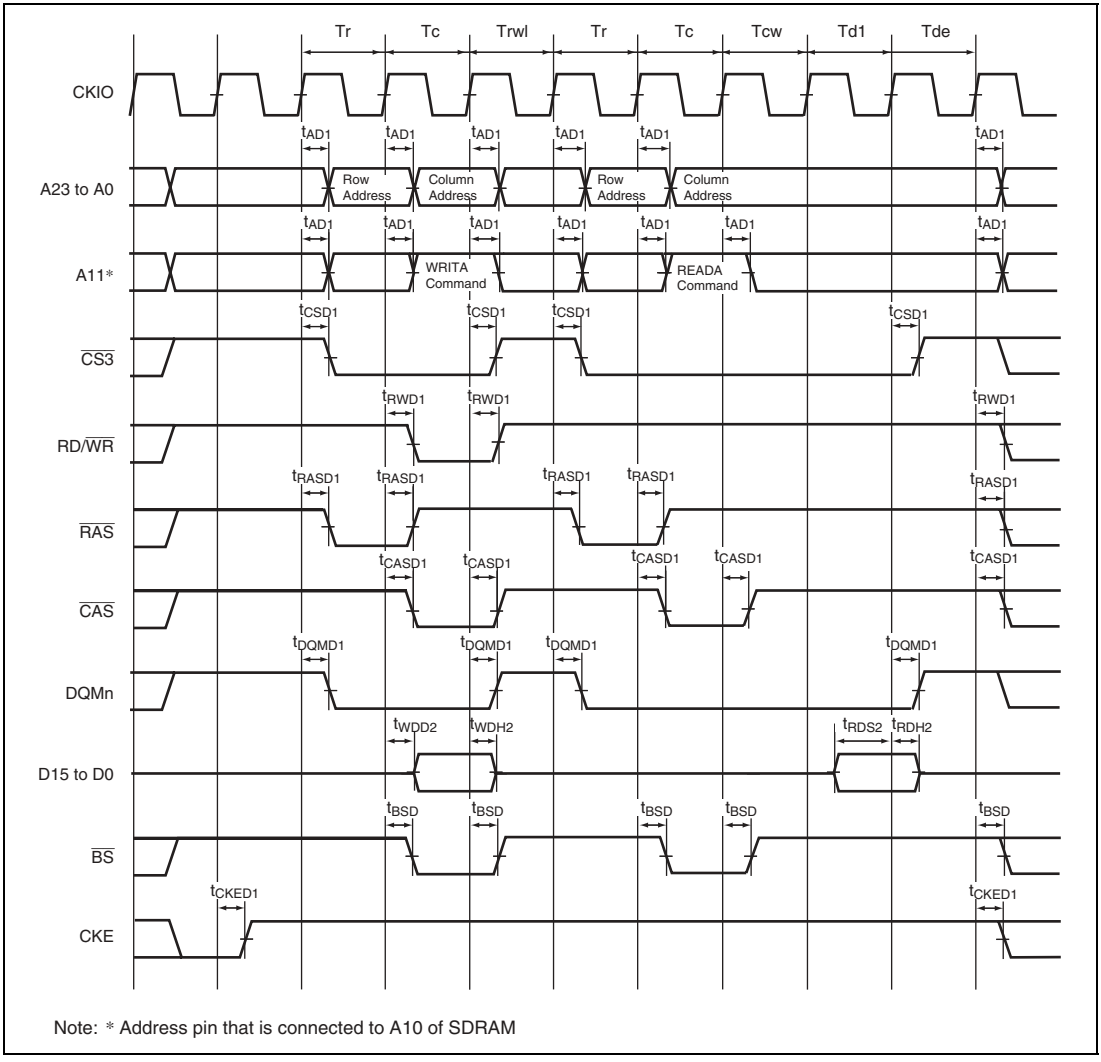
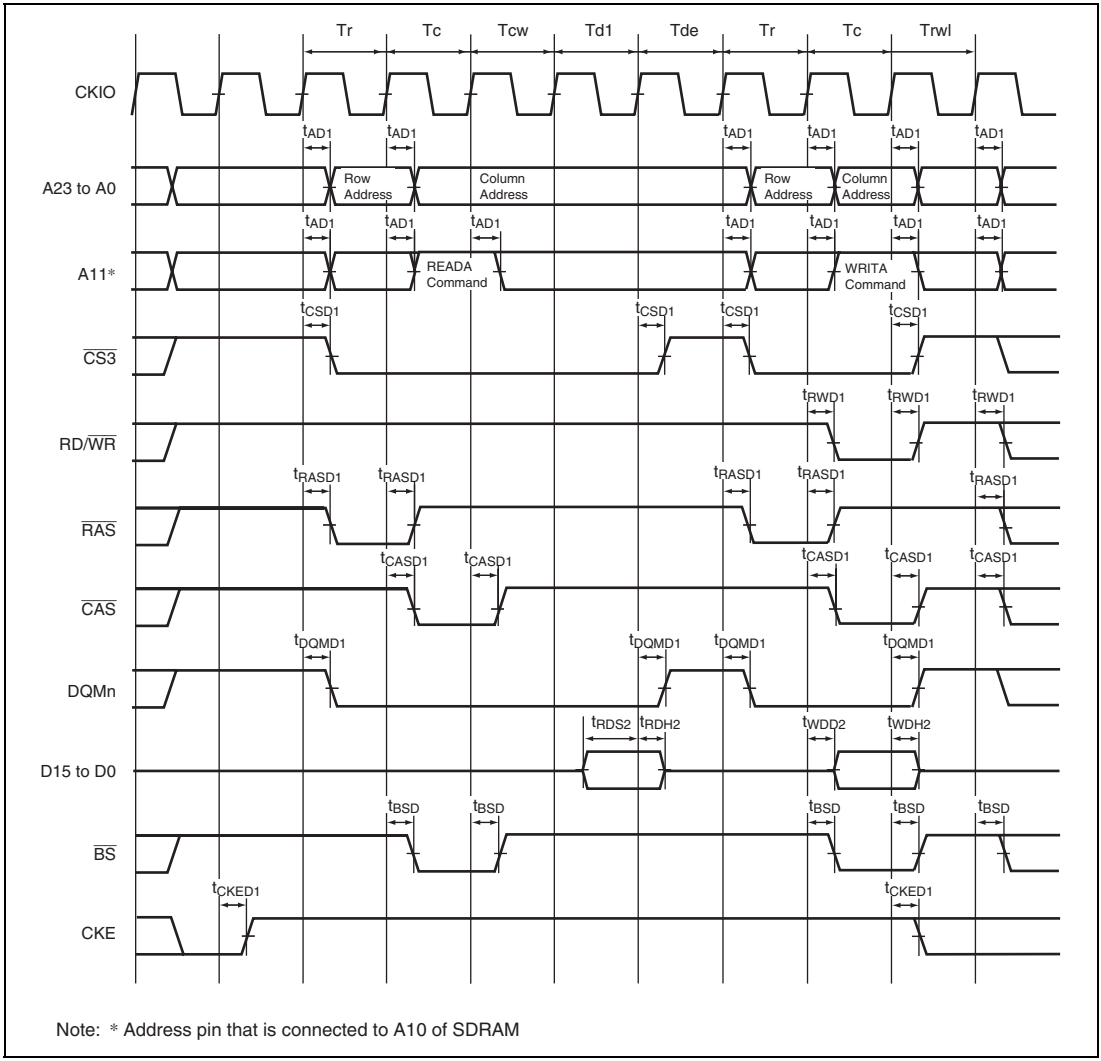


Figure 28.44 Write to Read Bus Cycle in Power-Down Mode of Synchronous DRAM (Auto Precharge Mode, TRCD = 1 Cycle, TRP = 1 Cycle, TRWL = 1 Cycle)



Note: * Address pin that is connected to A10 of SDRAM

Figure 28.45 Read to Write Bus Cycle in Power-Down Mode of Synchronous DRAM (Auto Precharge Mode, TRCD = 1 Cycle, TRP = 1 Cycle, TRWL = 1 Cycle)

28.3.7 Peripheral Module Signal Timing

Table 28.8 Peripheral Module Signal Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}–28 = 2.7 to 3.0 V, AV_{CC} = 2.7 to 3.6 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Module	Item	Symbol	Min	Max	Unit	Figure
I/O Port	Output data delay time	t_{PORTD}	—	17	ns	28.46
	Input data setup time	t_{PORTS}	17	—		
	Input data hold time	t_{PORTH}	10	—		
DMAC	DREQ setup time	t_{DREQS}	8	—	ns	28.47
	DREQ hold time	t_{DREQH}	8	—		
	DACK delay time	t_{DACKD}	—	14		28.48
	TEND delay time	t_{TENDD}	—	14		

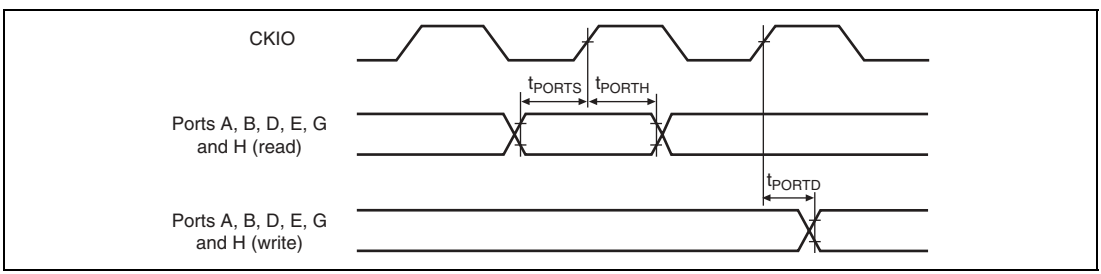


Figure 28.46 I/O Port Timing

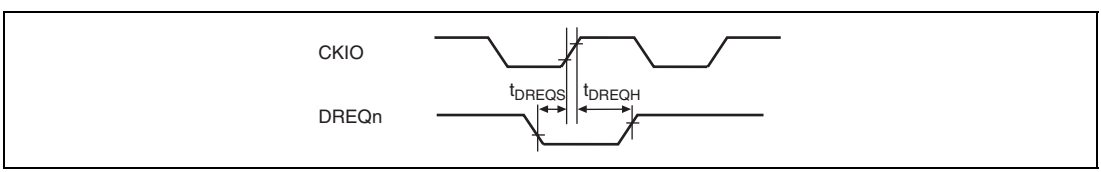


Figure 28.47 DREQ Input Timing (DREQ Low Level is Detected)

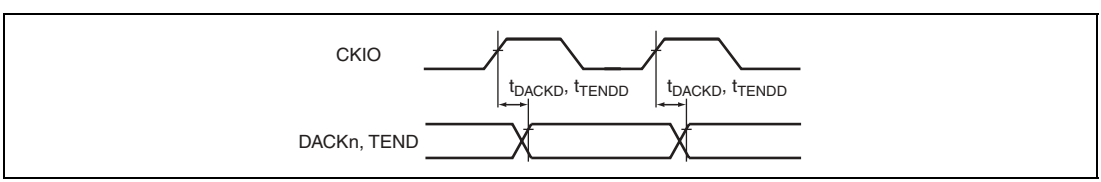


Figure 28.48 DACK and TEND Output Timing

28.3.8 SIOF Module Signal Timing

Table 28.9 SIOF Module Signal Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}_28= 2.7 to 3.0 V, AV_{CC} = 2.7 to 3.6 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Min	Max	Unit	Figure
SIOF_MCLK clock input cycle time	t_{MCYC}	t_{PCYC}	—	ns	28.49
SIOF_MCLK input high level width	t_{MWH}	$0.4 \times t_{MCYC}$	—	ns	28.49
SIOF_MCLK input low level width	t_{MWL}	$0.4 \times t_{MCYC}$	—	ns	28.49
SIOF_SCK clock cycle time	t_{SICYC}	t_{pcyc}	—	ns	28.50 to 28.54
SIOF_SCK output high level width	t_{SWHO}	$0.4 \times t_{SICYC}$	—	ns	28.50 to 28.53
SIOF_SCK output low level width	t_{SWLO}	$0.4 \times t_{SICYC}$	—	ns	28.50 to 28.53
SIOF_SYNC output delay time	t_{FSD}	—	20	ns	28.50 to 28.53
SIOF_SCK input high level width	t_{SWHI}	$0.4 \times t_{SICYC}$	—	ns	28.54
SIOF_SCK input low level width	t_{SWLI}	$0.4 \times t_{SICYC}$	—	ns	28.54
SIOF_SYNC input setup time	t_{FSS}	20	—	ns	28.54
SIOF_SYNC input hold time	t_{FSH}	20	—	ns	28.54
SIOF_TXD output delay time	t_{STDD}	—	20	ns	28.50 to 28.54
SIOF_RXD input setup time	t_{SRDS}	20	—	ns	28.50 to 28.54
SIOF_RXD input hold time	t_{SRDH}	10	—	ns	28.50 to 28.54
Slave select setup time	t_{SSS}	$t_{SICYC}/2 \times n$*^{2} –20	—	ns	28.55 to 28.56
Slave select hold time	t_{SSH}	$t_{SICYC}/2 \times n$*^{2}	—	ns	28.55 to 28.56

Notes: 1. t_{PCYC} is a cycle time of a peripheral clock (Pϕ).
2. $t_{SICYC}/2 \times n$ is defined according to the SSAST1 and SSAST0 bits in SPICR.

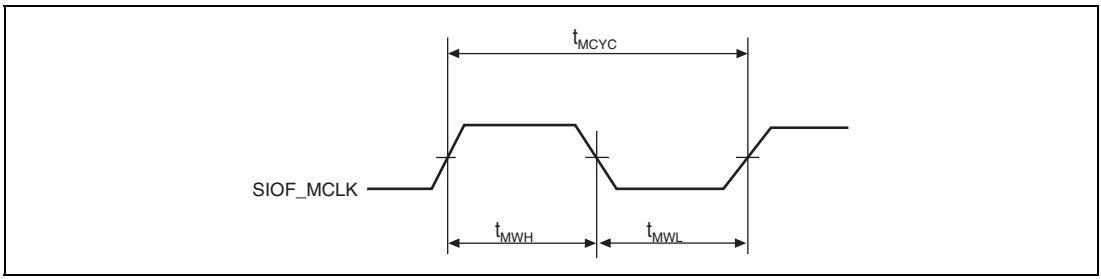


Figure 28.49 SIOF_MCLK Input Timing

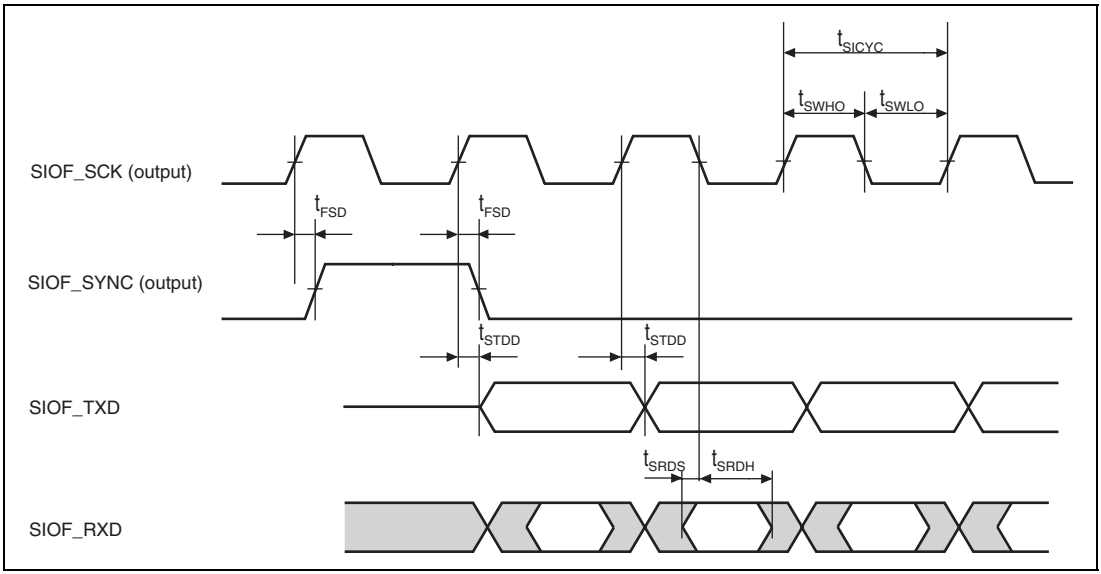


Figure 28.50 SIOF Transmission/Reception Timing (Master Mode 1, Falling Edge Sampling)

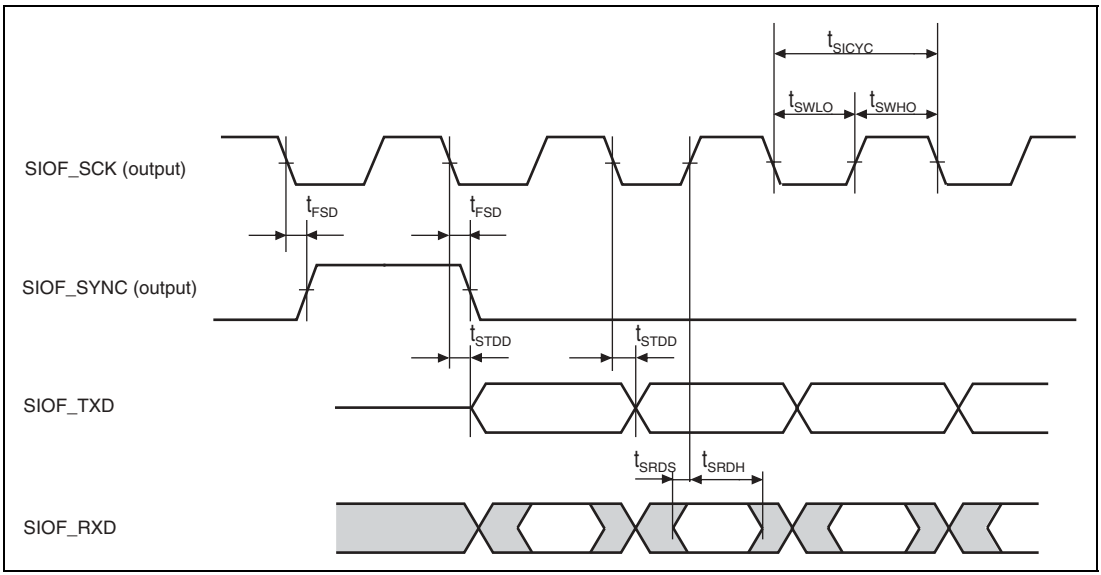


Figure 28.51 SIOF Transmission/Reception Timing (Master Mode 1, Rising Edge Sampling)

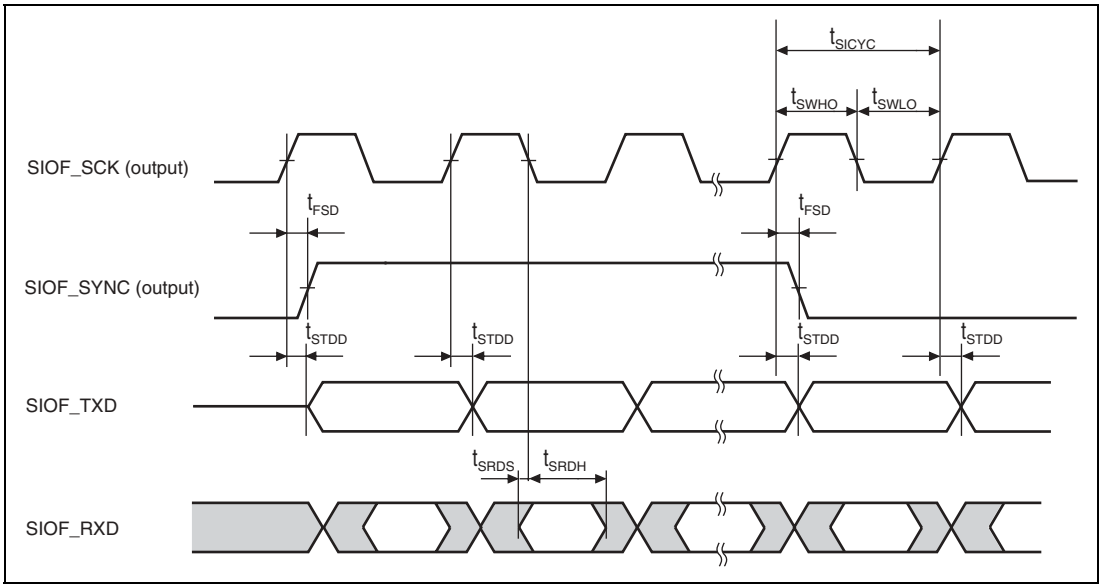


Figure 28.52 SIOF Transmission/Reception Timing (Master Mode 2, Falling Edge Sampling)

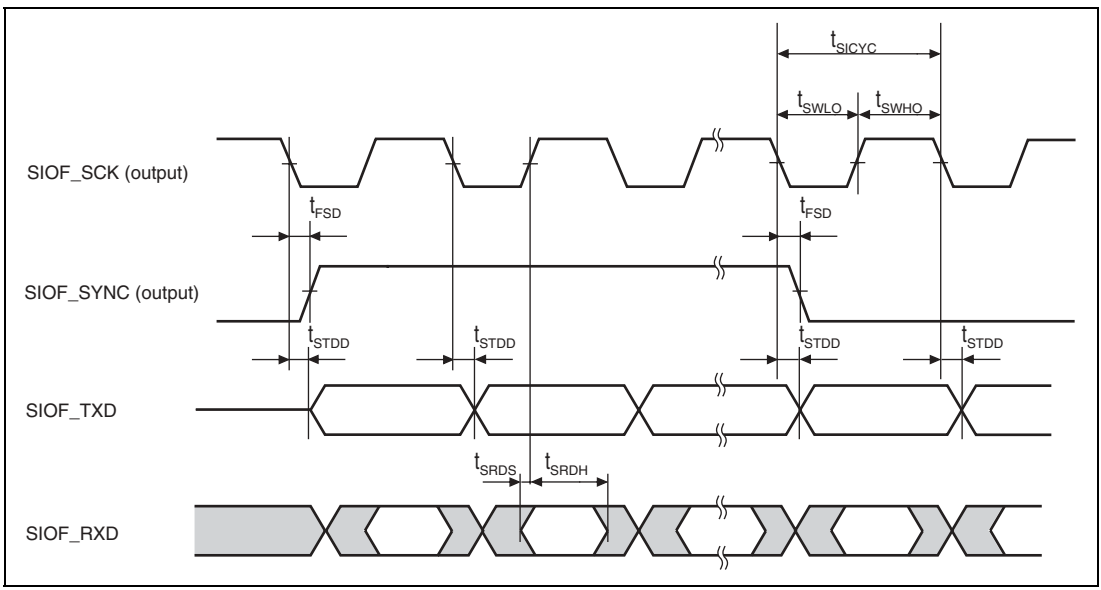


Figure 28.53 SIOF Transmission/Reception Timing (Master Mode 2, Rising Edge Sampling)

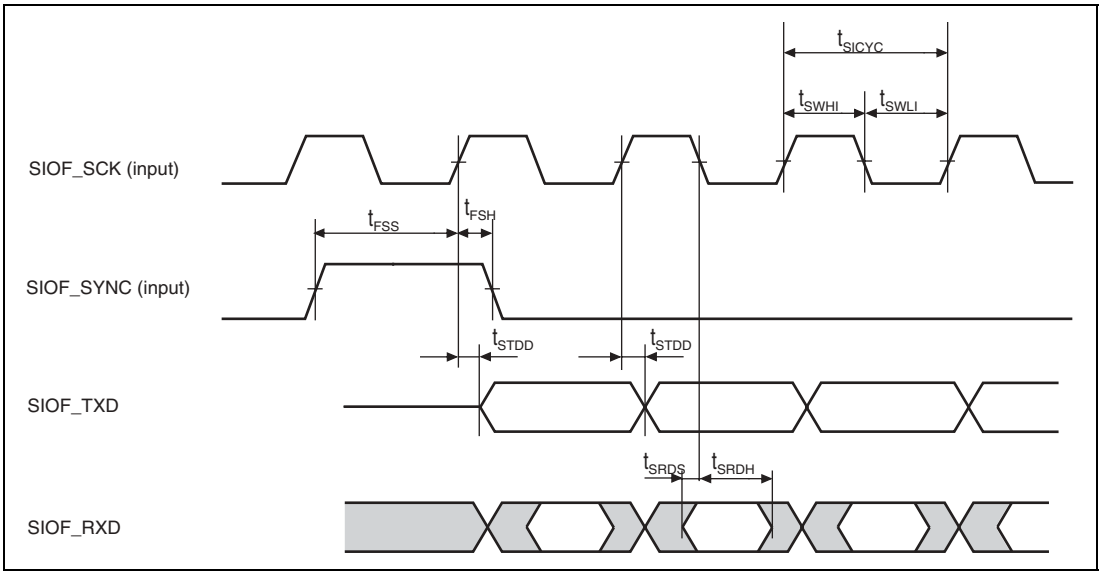


Figure 28.54 SIOF Transmission/Reception Timing (Slave Mode 1, Slave Mode 2)

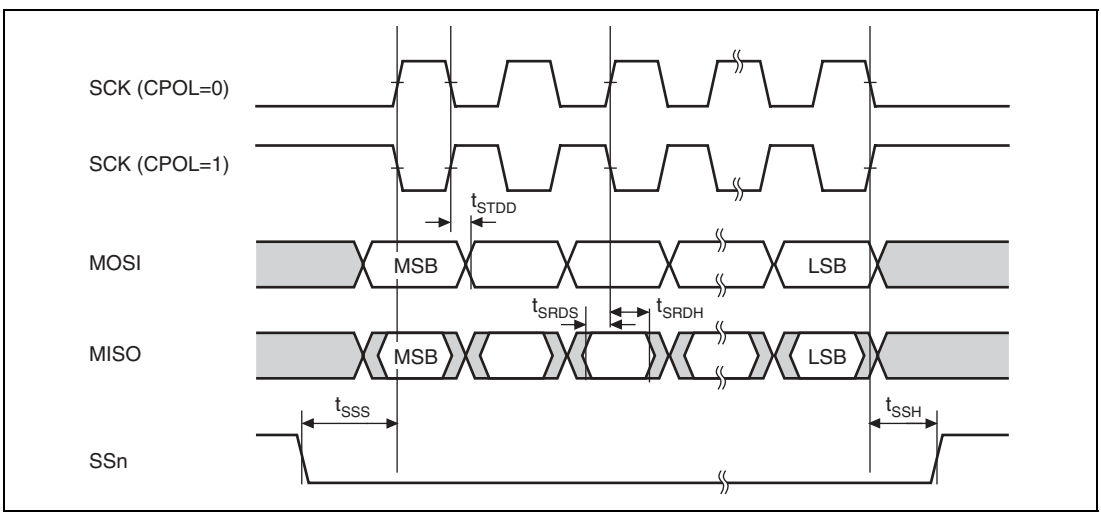


Figure 28.55 SIOF Transmission/Reception Timing (SPI Mode, CPHA = 0, SSAST1 and SSAST0 = 01)

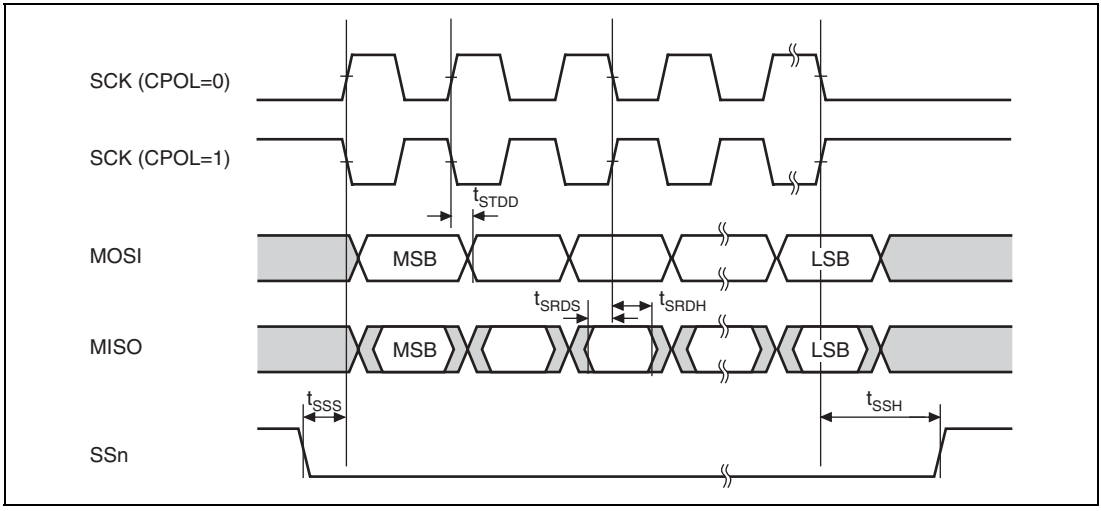


Figure 28.56 SIOF Transmission/Reception Timing (SPI Mode, CPHA = 1, SSAST1 and SSAST0 = 01)

28.3.9 SCIF Module Signal Timing

Table 28.10 SCIF Module Signal Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}–28= 2.7 to 3.0 V, AV_{CC} = 2.7 to 3.6 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item		Symbol	Min	Max	Unit	Figure
SCIF input clock cycle	Asynchronous	t_{SCYC}	$4 \times t_{PCYC}$	—	ns	28.57, 28.58
	Synchronous		$12 \times t_{PCYC}$	—	ns	
SCIF input clock high level width		t_{SCWH}	$0.4 \times t_{PCYC}$	—	ns	28.57
SCIF input clock low level width		t_{SCWL}	$0.4 \times t_{PCYC}$	—	ns	
SCIF transmit data delay time		t_{TXD}	—	$3t_{PCYC}$ + 50	ns	28.58
SCIF transmit data setup time (synchronous)		t_{RXS}	$2 \times t_{PCYC}$	—	ns	
SCIF transmit data hold time (synchronous)		t_{RXH}	$2 \times t_{PCYC}$	—	ns	

Note: t_{pcyc} is a cycle time of a peripheral clock (Pϕ).

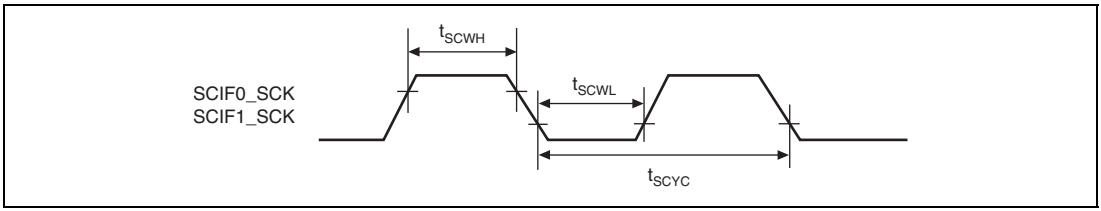


Figure 28.57 SCIF Input Clock Cycle

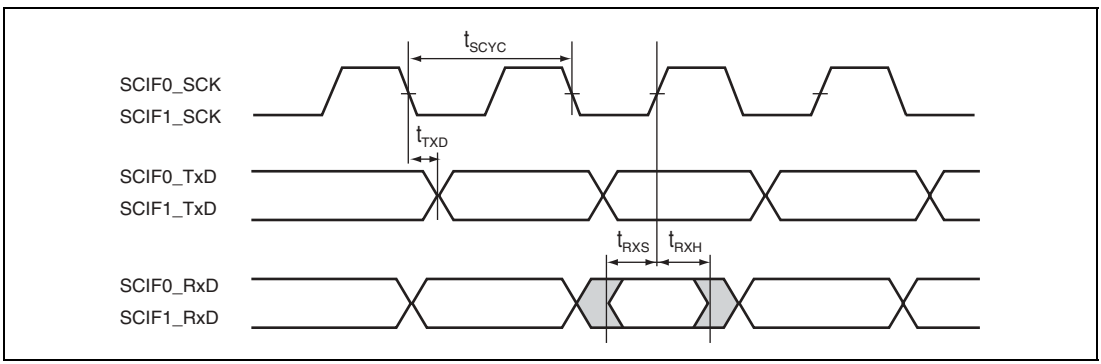


Figure 28.58 Input/Output Timing in SCIF Synchronous Mode

28.3.10 USB Module Signal Timing

Table 28.11 USB Module Clock Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}_28= 2.7 to 3.0 V, AV_{CC}(USB) = 3.0 to 3.6 V,
V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Min.	Max.	Unit	Figure
Clock rise time	t_{R48}	—	4	ns	28.59
Clock fall time	t_{F48}	—	4	ns	
Duty	t_{HIGH}/t_{LOW}	90	110	%	

Table 28.12 USB Module Clock Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}_28= 2.7 to 3.0 V, AV_{CC}(USB) = 3.0 to 3.6 V,
V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Frequency	Unit	Condition
Frequency	$1/t_{FREQ}$	48.0 (± 500 ppm)	MHz	USB Host is used
		48.0 (± 2500 ppm)	MHz	USB Function is used

Note: When USB is operated by a clock which is supplied from an external device to the UCLK pin without using the on-chip multiplier circuit, the clock should be in conformance with the USB specifications Ver. 1.1.

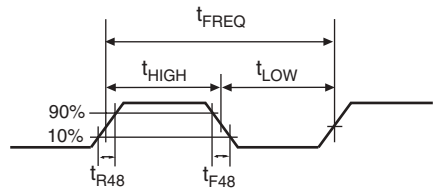


Figure 28.59 USB Clock Timing

28.3.11 USB Transceiver Timing

Table 28.13 USB Transceiver Timing (Full-Speed)

Conditions: V_{CC} = 2.7 to 3.6 V, AV_{CC}(USB) = 3.0 to 3.6 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Rise time	t_r	4	—	20	ns	C_L = 50 pF
Fall time	t_f	4	—	20	ns	C_L = 50 pF
Rise/fall time ratio	t_r/t_f	80	—	120	%	
Output signal crossover voltage	V_{CRS}	1.3	—	2.0	V	—
Output driver resistance*	Z_{DRU}	28	—	44	Ω	

Note: Includes an externally connected series resistance, RS = 27 Ω ± 1 %.

Table 28.14 USB Transceiver Timing (Low-Speed)

Conditions: V_{CC} = 2.7 to 3.6 V, AV_{CC}(USB) = 3.0 to 3.5 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Rise time	t_r	75	—	—	ns	C_L = 200 pF
		—	—	300	ns	C_L = 600 pF
Fall time	t_f	75	—	—	ns	C_L = 200 pF
		—	—	300	ns	C_L = 600 pF
Rise/fall time ratio	t_r/t_f	80	—	125	%	
Output signal crossover voltage	V_{CRS}	1.3	—	2.0	V	—

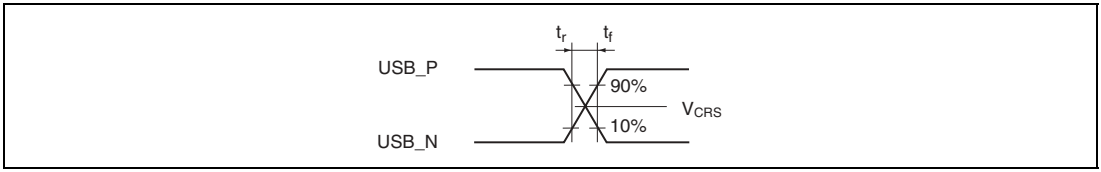


Figure 28.60 USB Transceiver Timing

Testing Circuits:

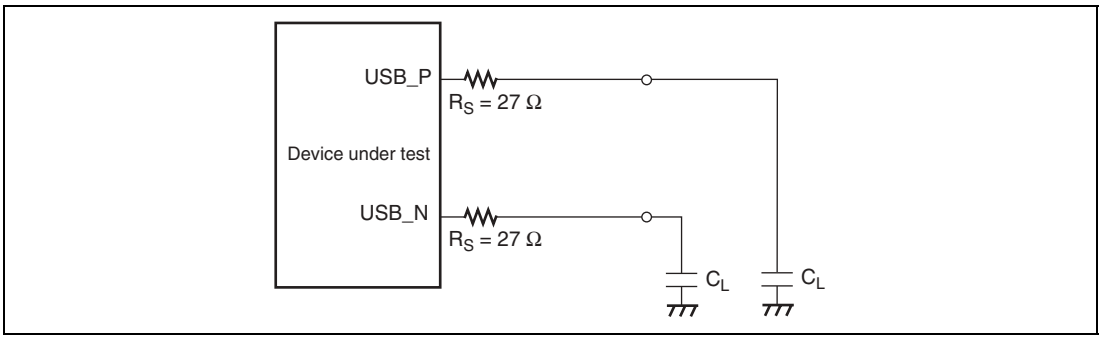


Figure 28.61 USB Transceiver Characteristics Testing Circuit (Full-Speed)

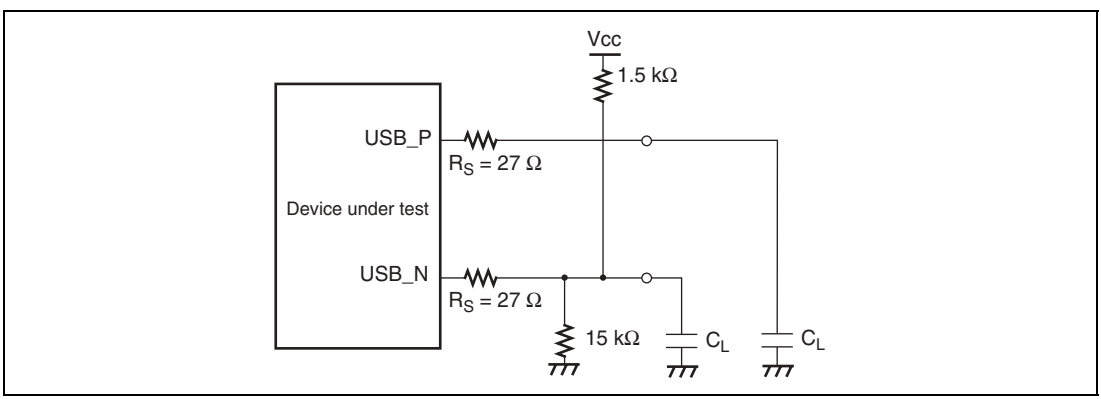


Figure 28.62 USB Transceiver Characteristics Testing Circuit (Low-Speed)

28.3.12 Bluetooth Interface (BT) Timing

Table 28.15 Bluetooth Interface Module Clock Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}–28 = 2.7 to 3.0 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit	Figure
Rise time	t_R	—	—	4	ns	28.63
Fall time	t_F	—	—	4	ns	
Duty	t_{HIGH}/t_{LOW}	90	100	110	%	

Table 28.16 Bluetooth Interface Module Clock Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}–28 = 2.7 to 3.0 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Frequency	Unit	Condition
Frequency	$1/t_{FREQ}$	13.0 (± 20 ppm)	MHz	When Renesas RF-IC is connected.*

Note: * When Renesas Technology RF-IC (HD157100NP or HD157102NP) is connected.

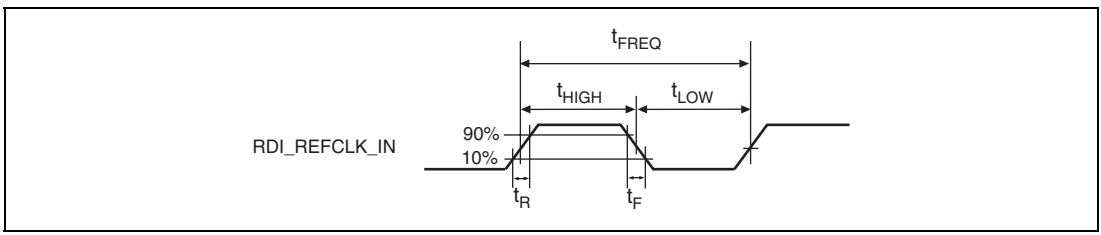


Figure 28.63 Bluetooth Module Clock Timing

Table 28.17 Bluetooth Interface Module Low Power Clock Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}–28 = 2.7 to 3.0 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit	Figure
Rise time	t_R	—	—	4	ns	28.64
Fall time	t_F	—	—	4	ns	
Duty	t_{HIGH}/t_{LOW}	90	100	110	%	

Table 28.18 Bluetooth Interface Module Low Power Clock Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}_28 = 2.7 to 3.0 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Frequency	Unit	Condition
Frequency	$1/t_{FREQ}$	32.0 (± 250 ppm)	kHz	Frequency conversion circuit is bypassed
		32.768 (± 250 ppm)	kHz	Frequency conversion circuit is used (maximum error is about 30 μs)

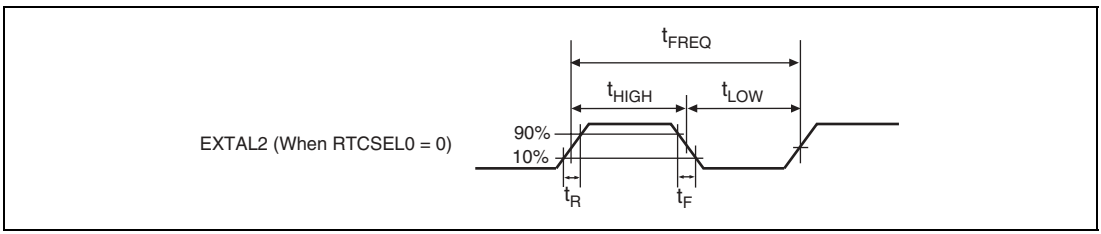


Figure 28.64 Bluetooth Interface Module Low Power Clock Timing

Table 28.19 Bluetooth Interface (BT) Voice CODEC Interface Signal Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}_28 = 2.7 to 3.0 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit	Figure
VCI_SCO_CLK_OUT clock output frequency	f_{VCI_CLK}	0.2	—	1	MHz	28.65, 28.66
VCI_SCO_SYNC_OUT output delay time	t_{SYNCD}	—	—	200	ns	28.65, 28.66
VCI_SCO_RX output delay time	t_{RXD}	—	—	200	ns	28.65, 28.66
VCI_SCO_TX setup time	t_{TXS}	100	—	—	ns	28.65, 28.66
VCI_SCO_TX hold time	t_{TXH}	100	—	—	ns	28.65, 28.66
VCI_HWC output delay time*	t_{HWCD}	—	—	200	ns	28.65

Note: * In case that the STLC7550 is connected as the external Voice CODEC IC.

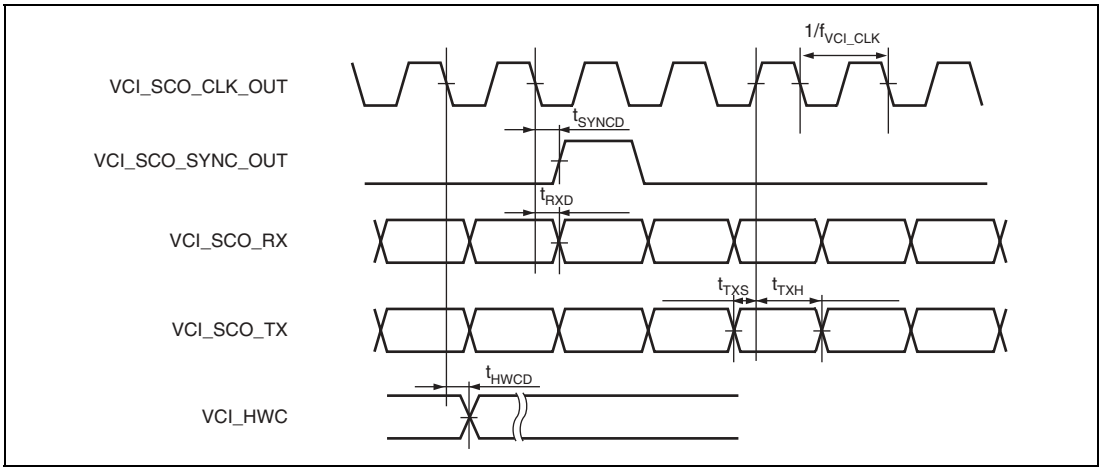


Figure 28.65 Bluetooth Interface (BT) Voice CODEC (STLC7550) Interface Signal Timing

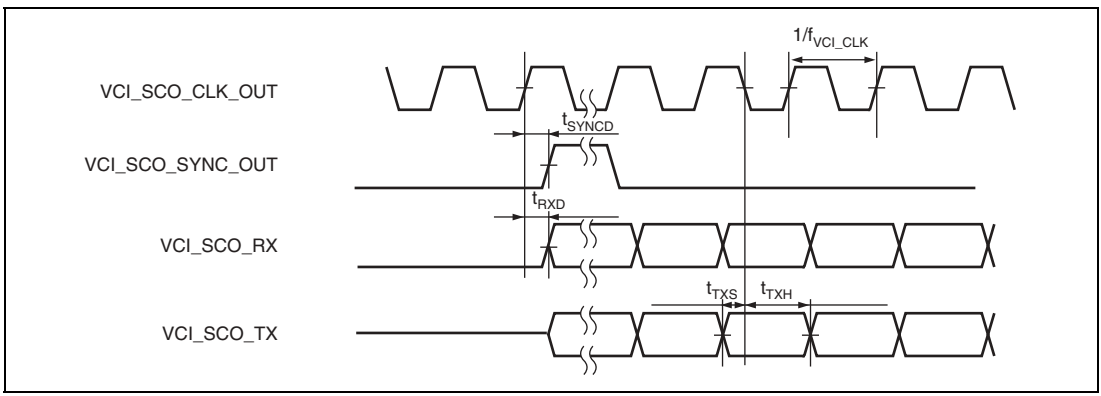


Figure 28.66 Bluetooth Interface (BT) Voice CODEC (MC145483) Interface Signal Timing

Table 28.20 SPI Interface Signal Timing for Bluetooth Interface (BT) RF

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}_28 = 2.7 to 3.0 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit	Figure
RCI_SPI_CLK clock output frequency*	f_{SPI_CLK}	—	6.5	—	MHz	28.67
RCI_SPI_CLK duty		45	50	55	%	28.67
RCI_SPI_TXRX output delay time	t_{DELAY}	—	—	30	ns	28.67
RCI_SPI_TXRX setup time	t_{SETUP}	40	—	—	ns	28.67
RCI_SPI_TXRX hold time	t_{HOLD}	45	—	—	ns	28.67

Note: * When Renesas Technology RF-IC (HD157100NP or HD157102NP) is connected.

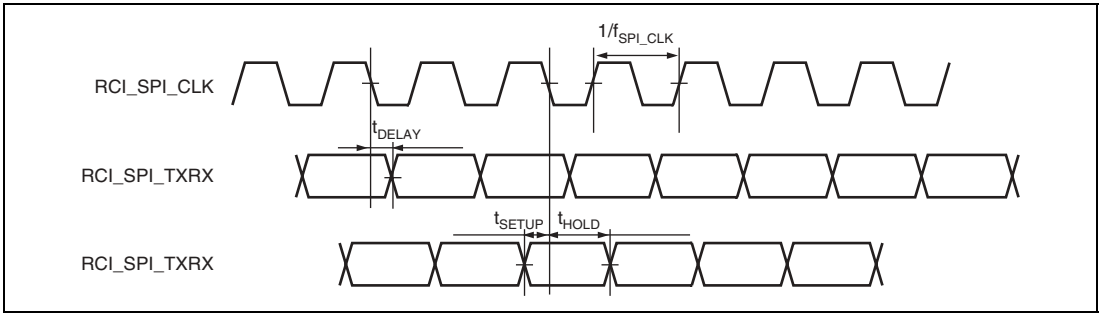


Figure 28.67 SPI Interface Signal Timing for Bluetooth Interface (BT) RF

Table 28.21 Receive Data Timing

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}_28 = 2.7 to 3.0 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Symbol	Min.	Typ.	Max.	Unit	Figure
RDI_TXTRDATA	$t_{TXTRDATA}$	0.7	1	1.3	µs	28.68

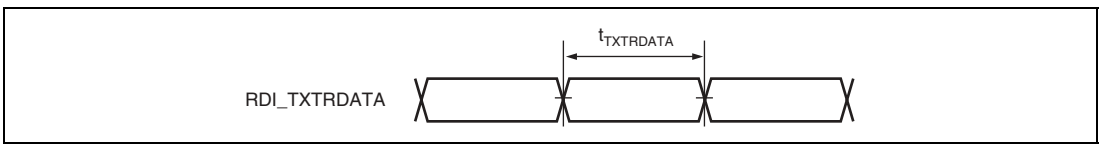
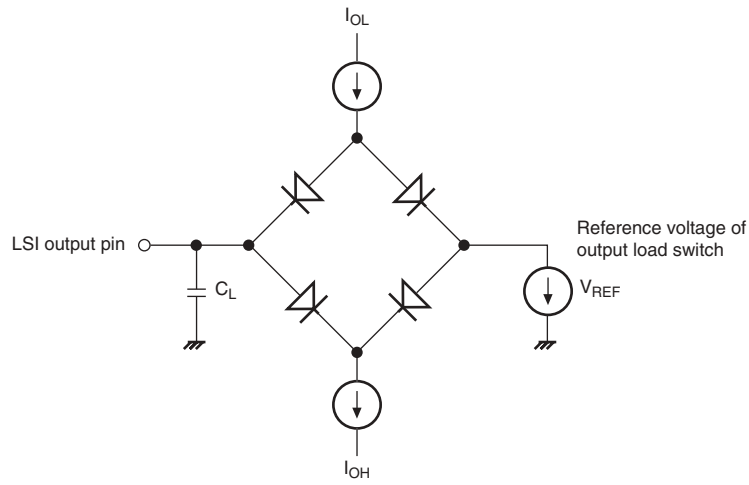


Figure 28.68 Bluetooth Interface (BT) Receive Data Timing

28.3.13 AC Characteristic Test Conditions

- I/O signal reference level: 1.5 V (V_{CC} = 2.7 to 3.6 V, V_{CC}_28 = 2.7 to 3.0 V)
- Input pulse level: V_{SS} to 3.3 V (where $\overline{\text{RESETP}}$, RDI_TXTRDATA, RDI_RXBDW_OUT, RDI_CTRL4, RDI_REFCLK_IN, RCI_SPI_CLK, RCI_SPI_ENB, and RCI_SPI_TXRX are within V_{SS} to 2.8 V)
- Input rise and fall times: 1 ns



Notes: 1. C_L is the total value that includes the capacitance of measurement instruments, etc., and is set as follows for each pin.
30pF: CKIO, $\overline{\text{CS0}}$, $\overline{\text{CS3}}$, $\overline{\text{CS4}}$
50pF: All other pins
2. I_{OL} = 0.2mA, I_{OH} = -0.2mA

Figure 28.69 Output Load Circuit

28.4 D/A Converter Characteristics

Table 28.22 lists the D/A converter characteristics.

Table 28.22 D/A Converter Characteristics

Conditions: V_{CC} = 2.7 to 3.6 V, V_{CC}_28 = 2.7 to 3.0 V, V_{DD} = 1.4 to 1.6 V, Ta = –40 to 85°C

Item	Min.	Typ.	Max.	Unit
Resolution	8	8	8	bits
Conversion time (20pF load capacitance)	—	—	10	µs
Absolute accuracy (2MΩ load resistance)	—	—	±4.0	LSB

Appendix

A. Pin States

The initial values differ in each operating mode (reset, power-down mode, bus-released state). The following symbols are used in the table.

Legend

O : Output
I : Input
IP : Input (Pull-up MOS on)
H : High-level output
L : Low-level output
Z1 : High impedance (I/O buffer off, and pull-up MOS off)
Z2 : High impedance (Output buffer off)
P : I or O, depending on register setting
K : Input pins are high-impedance, and output pins retain their state.
V : I/O buffer off, and pull-up MOS on

Table A.1

		Reset		Power-Down Mode			
Type	Pin Name	Power-On Reset	Manual Reset	Software Standby	Module Standby	Sleep	Bus-Released State
Clock	EXTAL	I	I	I	I		I
	XTAL	O	O	O	O		O
	CKIO	I, O*[1]	I, O*[1]	Z, I, O*[1]*[2]	I, O*[1]		Z, I, O*[1]*[2]
Operation mode Control	MD5, MD2, MD1	I	I	I	I		I
System control	$\overline{\text{RESETP}}$	I	I	I	I		I
	$\overline{\text{BOOT_E}}$	IP	IP	IP	IP		IP
	$\overline{\text{BREQ}}$	I	I	I	I		I
	$\overline{\text{BACK}}$	Z1	O	O	O		O
Interrupt	NMI	I	I	I	I		I
	IRQ4 to IRQ2, IRQ0	V	I	I	I		I
	$\overline{\text{IRQOUT}}$	V	I	O	O		O

Type	Pin Name	Reset		Power-Down Mode			Bus-Released State
		Power-On Reset	Manual Reset	Software Standby	Module Standby	Sleep	
Address bus	A23 to A19, A0	Z1	O	Z, O *3	O		Z1
	A18 to A1	O	O	Z, O *3	O		Z2
Data bus	D15 to D0	Z1	Z1	Z1	I, O		Z1
Bus control	$\overline{CS0}$, $\overline{CS3}$, $\overline{CS4}$	H	O	Z, O*3	O		Z2
	$\overline{RD}$	H	O	Z, O*3	O		Z2
	RD/$\overline{WR}$	H	O	Z, O*3	O		Z2
	$\overline{BS}$	H	O	Z, O*3	O		Z2
	$\overline{WE1}$/DQM1	H	O	Z, O*3	O		Z2
	$\overline{WE0}$/DQM0	H	O	Z, O*3	O		Z2
	$\overline{CAS}$	H	O	Z, O*3	O		Z, O*4
	$\overline{RAS}$	H	O	Z, O*3	O		Z, O*4
	CKE	H	O	Z, O*3	O		Z, O*4
	$\overline{REFOUT}$	O	O	O	O		O
	$\overline{WAIT}$	IP	I	IP	IP		IP
Direct Memory Access Controller (DMAC)	DREQ	V	I	Z1	I		I
	DACK	V	O	Z1	O		O
	TEND	V	O	Z1	O		O
Serial I/O with FIFO (SIOF)	SIOF_TXD(MOSI)	V	O	Z1	O		O
	SIOF_RXD(MISO)	V	I	Z1	I		I
	SIOF_SCK(SCK)	V	O	Z1	I, O		I, O
	SIOF_MCLK	V	I	Z1	I		I
	SIOF_SYNC(SIOF_$\overline{SS0}$)	V	O	Z1	I, O		I, O
	SIOF_$\overline{SS1}$	V	O	Z1	O		O
	SIOF_$\overline{SS2}$	V	O	Z1	O		O
Serial communication interface with FIFO (SCIF0, 1)	SCIF0_TxD, SCIF1_TxD	Z1	Z1	Z1	O		O
	SCIF0_RxD, SCIF1_RxD	V	Z1	Z1	I		I
	SCIF0_SCK, SCIF1_SCK	V	Z1	Z1	I, O		I, O
	SCIF0_$\overline{RTS}$, SCIF1_$\overline{RTS}$	V	O	Z1	O		O
	SCIF0_$\overline{CTS}$, SCIF1_$\overline{CTS}$	V	I	Z1	I		I

Type	Pin Name	Reset		Power-Down Mode			Bus-Released State
		Power-On Reset	Manual Reset	Software Standby	Module Standby	Sleep	
USB	UCLK	I	I	I	I		I
	$\overline{\text{USB_PWR_EN}}$/ USB_PULLUP	O	O	O	O		O
	$\overline{\text{USB_OVR_CRNT}}$/ USB_VBUS	I	I	I	I		I
	USB_N	I, O	I, O	I, O	I, O		I, O
	USB_P	I, O	I, O	I, O	I, O		I, O
Bluetooth interface (BT)	RDI_TXTRDATA	O	O	O	O		O
	RDI_RXBDW_OUT	O	O	O	O		O
	RDI_REFCLK_IN	I	I	I	I		I
	RDI_CTRL2	O	O	O	O		O
	RDI_CTRL3	O	O	O	O		O
	RDI_CTRL4	O	O	O	O		O
	RCI_SPI_CLK	O	O	O	O		O
	RCI_SPI_TXRX	O	O	O	O		O
	RCI_SPI_ENB	O	O	O	O		O
	VCI_SCO_CLK_OUT	Z1	O	O	O		O
	VCI_SCO_SYNC_OUT	Z1	O	O	O		O
	VCI_SCO_TX	V	I	I	I		I
	VCI_SCO_RX	V	O	O	O		O
	VCI_HWC	V	O	O	O		O
	$\overline{\text{VCI_CODEC_PWRDWN}}$	V	O	O	O		O
	RTCSEL0	I	I	I	I		I
	RREF	I	I	I	I		I
	EXTAL2	I	I	I	I		I
	XTAL2	O	O	O	O		O
D/A converter (DAC)	DA1, DA0	Z2	O	O	O		O
I/O port	PTA6 to PTA0	V	P	K	P		P
	PTB7 to PTB0*[5]	V	P	K	P		P
	PTD5 to PTD0*[5]	V	P	K	P		P
	PTE6 to PTE0*[5]	V	P	K	P		P

		Reset		Power-Down Mode			
Type	**Pin Name**	**Power-On Reset**	**Manual Reset**	**Software Standby**	**Module Standby**	**Sleep**	**Bus-Released State**
I/O port	PTG4 to PTG2, PTG0	V	P	K	P		P
	PTG1	Z1	P	K	P		P
	PTH4 to PTH2, PTH0	V	P	K	P		P
	PTH1	Z1	P	K	P		P
User debugging interface (H-UDI)	TCK	I	I	I	I		I
	TMS	IP	IP	IP	IP		IP
	TDI	IP	IP	IP	IP		IP
	TDO	Z2	Z2	Z2	Z2		Z2
	$\overline{\text{TRST}}$	IP	IP	IP	IP		IP
Advanced user debugger (AUD)	AUDATA3 to AUDATA0	O	O	O	O		O
	AUDCK	O	O	O	O		O
	$\overline{\text{AUDSYNC}}$	H	H	H	H		H
E10A interface	$\overline{\text{ASEBRKAK}}$	O	O	O	O		O
	$\overline{\text{ASEMD0}}$	I	I	I	I		I

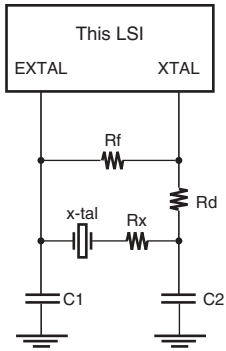
Notes: 1. Depends on clock operating mode.

2. I, O, or Z, depending on register setting.
   I or O, depending on clock operating mode.
3. O or Z, depending on register setting in software standby mode.
4. O or Z, depending on register setting in bus released state.
5. These pins are multiplexed with the BSC function, and there are some pins which initial becomes a bus controller function. Please refer to table 24.1 for details.

B. Usage Notes for Oscillator circuit with External Crystal Resonator

On a design of the oscillator circuit with the external crystal resonator, values of external component should be chosen from the evaluation results examined with the crystal resonator manufacturer. The evaluation result examples by Kinseki,Limited are shown below.

B.1 Recommended Oscillator Circuit



Crystal resonator specifications

Part number	CX-53G
Package size	5.0 × 3.2 × 1.3mm

Evaluation results:

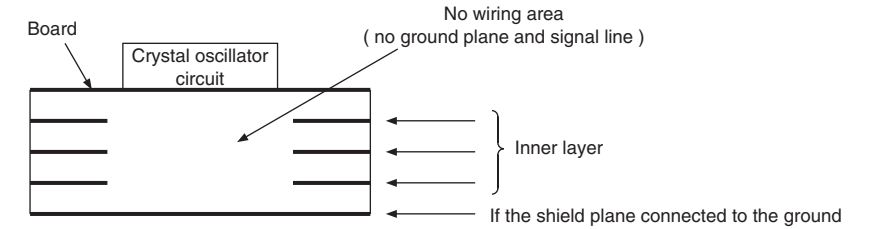
Crystal resonator part number		CX-53G	CX-53G	CX-53G
Oscillating frequency (kHz)		11289.6	18432	30000
Overtone number		fundamental oscillation	fundamental oscillation	fundamental oscillation
Shunt capacitance (pF)		8.0	8.0	8.0
Recommended external circuit parameters	Rf	No connect.	No connect.	No connect.
	Rd	0 Ω	0 Ω	0 Ω
	Rx	0 Ω	0 Ω	0 Ω
	C1	7pF	7pF	8pF
	C2	7pF	7pF	8pF
Characteristics under recommended parameters	Negative resistance (Ω)	-5442	-2826	-943
	Load capacitance (pF)	7.93	7.91	8.07
	Frequency deviation ($\times 10^{-6}$)	+1.1	+1.5	-1.7
	Drive level (μW)	83	125	65

Note: The characteristics of the oscillator circuit vary with the external circuit parameters and the PCB layout. After an evaluation of the matching between the crystal resonator and the oscillator circuit sufficiently, choose the circuit parameter and PCB layout. The recommended external circuit parameters and characteristics are only guideline and recommendations. They are not guaranteed on the user application systems.
For the designing the PCB layout, refer to section B2 Note for PCB design.

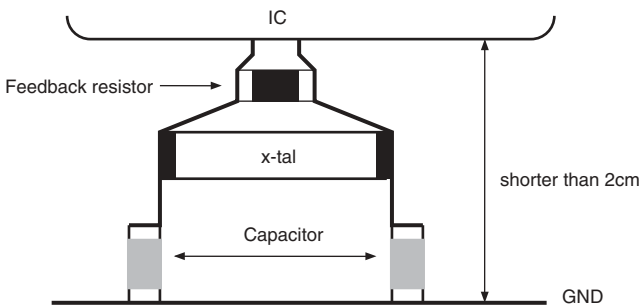
B.2 Note on PCB design

Following items should be considered for the PCB design.

- The external oscillator circuit should be placed as near from oscillator IC as possible.
- Ground plane should not be placed under the oscillator circuit area.
- Signal line length should be shorter (shorter than 2cm).
- Keep spacing between two signal lines connected to the crystal resonator. Do not place them close to each other.
- If a shield is required for oscillator circuit, the shield area should be as small as necessary.



PCB cross section (multi-layer board)

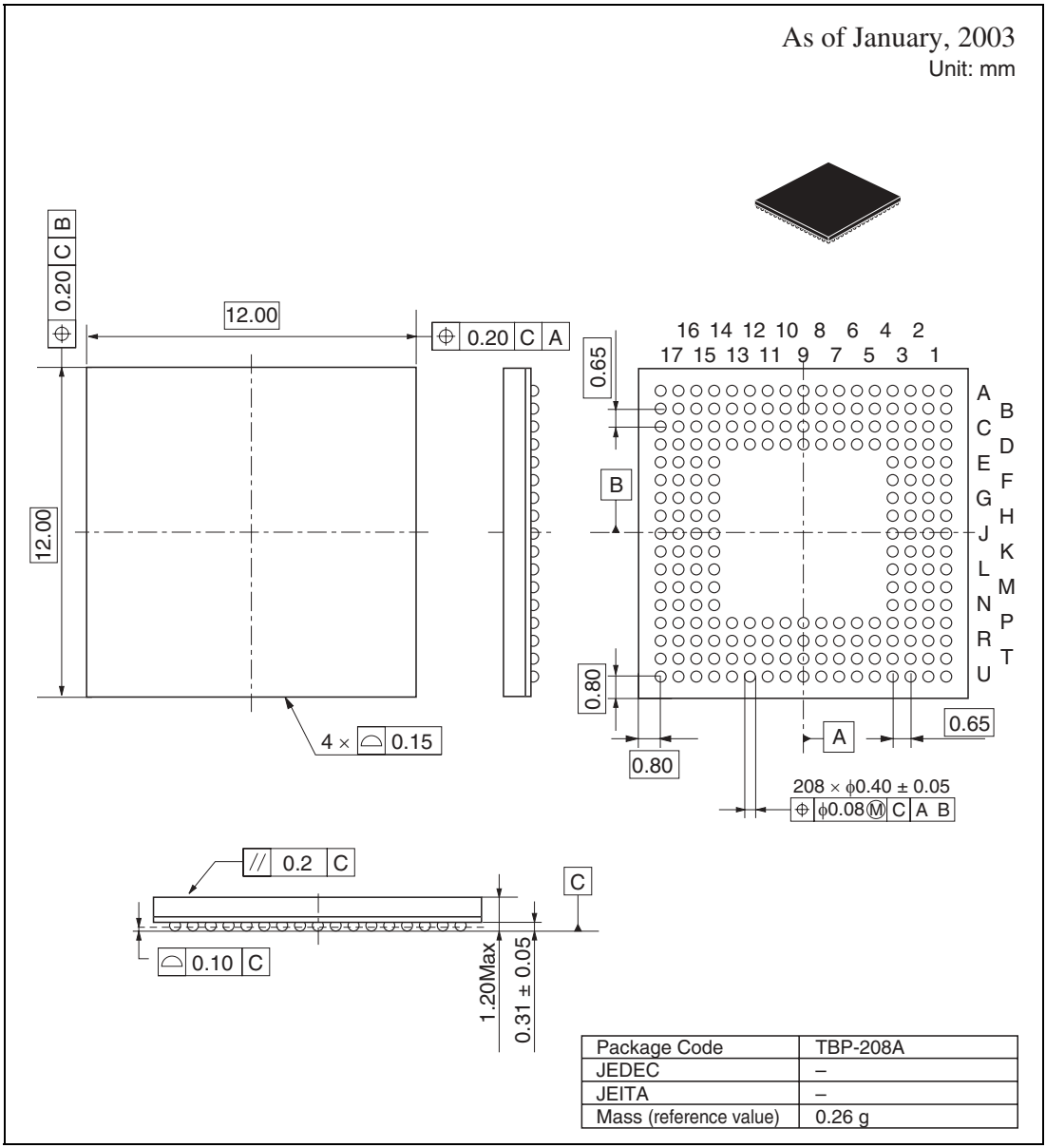


Top view of PCB

For further information on Appendix B, please contact:
Kinseki,Limited (Headquarter) 1-8-1, Izumihon-cho, Komae-shi, Tokyo, 201-8648 Japan
http://www.kinseki.co.jp/ E-mail:kanriks@kinseki.co.jp
Oversea sales TEL +81-3-5497-3111 FAX +81-3-5497-3209

C. Package Dimensions

For package dimensions that are shows in the Renesas Semiconductor Packages Data Book have priority.



Package Code	TBP-208A
JEDEC	–
JEITA	–
Mass (reference value)	0.26 g

Figure C.1 Package Dimensions (TBP-208A)

Index

16-Bit/32-bit displacement 48
Absolute addresses 48
Access wait control..... 256
Address array..... 178, 186
Address-array read..... 186
Address-array write (associative operation) 186
Address-array write (non-associative operation)..... 186
Addressing modes..... 49
Area division..... 221
Asynchronous mode 462
AUD 683
Auto-reload count operation 378
Auto-request mode 315
Baud rate generator..... 410
Big endian..... 45, 248
Bluetooth HCI/TCI commands..... 622
Bluetooth interface 611
Boot function 491
Bulk-in transfer..... 594
Bulk-out transfer..... 595
Burst mode..... 326
Burst ROM interface..... 285
Bus arbitration 291
Bus clock 333
Bus state controller 217
Byte-selection SRAM interface 287
Cache 177
Canceling software standby..... 352
Clock operating modes 337
Clock pulse generator 333
Clock synchronous mode..... 457, 475
Clock-resume control..... 622
Control registers 36, 41
Control transfer..... 587
CPU 29
CPU extended instructions 81
Cycle-steal mode 325
D/A converter 623
Data alignment..... 248
Data array..... 178, 187
Data-array read..... 187
Data-array write 187
Delayed branching 47
Direct memory access controller..... 297
Double data transfer instructions 97
DSP data operation instructions..... 107
DSP data transfer instructions..... 96
DSP instructions..... 111
DSP operating unit..... 73
DSP registers..... 80, 107
DSP repeat control 81
DSP status register 109
Dual address mode 321
Effective address..... 49
Effective addresses..... 49
Exception code..... 158
Exception codes 159
Exception handling 155
Exception handling state 29
Exception vector addresses 159
External address space 34
External request mode..... 315
General exceptions..... 165
General registers 36, 38
Global base register..... 44
I/O ports 627
Instruction formats 53
Intermittent mode..... 325
Internal clock 333
Interrupt controller 199
Interrupt-in transfer..... 593
Interval timer mode..... 353
IRQ interrupts 211
Isochronous out transfer..... 599
Isochronous-in transfer 597
JTAG..... 683

Literal constant 48
Little endian 46, 248
Load/store architecture 47
Logical address space 31
LRU 178
Manual reset 164
Module standby 368
Modulo addressing 103
Modulo register 78
Multiple interrupts 216
Multiply and accumulate registers 39
NMI interrupt 211
Notes on board design 344
On-chip peripheral module interrupts 212
On-chip peripheral module request 316
P0 area 32
P1 area 32
P2 area 32
P3 area 32
P4 area 32
PC trace 676
Peripheral clock 333
Physical address space 33
Pin function controller 639
Power-down modes 355
Power-down state 29
Power-on reset 164
Prefetch hit 184
Prefetch miss 184
Privileged mode 30
Procedure register 39
Processing modes 30
Program counter 36, 40
Read hit 184
Read miss 184
Receive margin 488
Refreshing 277
Registers
All registers 727
BAMRA 658, 704, 721, 728
BAMRB 661, 704, 721, 727
BARA 657, 704, 721, 728
BARB 660, 703, 720, 727
BBRA 658, 704, 721, 728
BBRB 663, 704, 721, 728
BDMRB 662, 703, 720, 727
BDRB 661, 703, 720, 727
BETR 667, 703, 720, 727
BRCR 664, 703, 720, 727
BRDR 670, 704, 721, 728
BRSR 669, 704, 721, 728
CCR1 ..... 179, 698, 705, 722
CCR2 ..... 180, 698, 705, 722
CHCR 302
CHCR_0 ..... 699, 723
CHCR_1 ..... 699, 710, 723
CHCR_2 ..... 699, 710, 723
CHCR_3 ..... 700, 711, 723
CHCR0 709
CLTR 576
CMNCR 225, 699, 707, 723
CS0BCR 699, 707, 723
CS0WCR 699, 707, 723
CS3BCR 699, 707, 723
CS3WCR 699, 707, 723
CS4BCR 699, 707, 723
CS4WCR 699, 708, 723
CSnBCR 227
CSnWCR 231
CTLR 703, 719, 727
CVR 575, 703, 719, 727
DACR 625, 703, 719, 727
DADR 624
DADR_0 ..... 703, 719, 727
DADR_1 ..... 703, 719, 727
DAR 301
DAR_0 ..... 699, 723
DAR_1 ..... 699, 709, 723
DAR_2 ..... 699, 710, 723
DAR_3 ..... 700, 711, 723
DAR0 709
DASTS 572, 703, 719, 726
DMA 573, 703, 719, 727
DMAOR 308, 700, 711, 724
DMARS 310
DMARS0 ..... 700, 711, 724

DMARS1 ..... 700, 711, 724
DMARS2 724
DMATCR 301
DMATCR_0 ..... 699, 723
DMATCR_1 ..... 699, 709, 723
DMATCR_2 ..... 699, 710, 723
DMATCR_3 ..... 700, 711, 723
DMATCR0 709
EPDR0i 566, 702, 718, 726
EPDR0o 567, 702, 718, 726
EPDR0s 567, 702, 718, 726
EPDR1 ..... 567, 702, 718, 726
EPDR2i 568, 702, 718, 726
EPDR2o 568, 702, 718, 726
EPDR3i 568, 702, 718, 726
EPDR3o 569, 702, 718, 726
EPDR4 ..... 569, 702, 718, 726
EPDR5 ..... 569, 702, 718, 726
EPDR6 ..... 570, 702, 718, 726
EPIR 703, 719, 727
EPIRn 578
EPSTL 574, 703, 719, 727
EPSZ0o 570, 702, 718, 726
EPSZ2o 570, 702, 718, 726
EPSZ3o 570, 702, 719, 726
EPSZ6 ..... 571, 703, 719, 726
EXCPGCR 502, 701, 715, 725
EXPEVT 157, 159, 698, 705, 722
FCLR 572, 703, 719, 727
FRQCR 340, 700, 711, 724
HBCEDR 528, 702, 716, 726
HBHEDR 528, 702, 716, 726
HCCEDR 527, 702, 716, 726
HCHEDR 527, 702, 716, 725
HCSR 516, 701, 715, 725
HCTLR 513, 701, 715, 725
HDHEDR 528, 702, 716, 726
HFIR 529, 702, 716, 726
HFNR 532, 702, 717, 726
HFRR 531, 702, 717, 726
HHCCAR 526, 702, 716, 725
HIDR 524, 701, 715, 725
HIER 521, 701, 715, 725
HISR 518, 701, 715, 725
HIZCRA 652, 703, 720, 727
HLSTR 534, 702, 717, 726
HPCEDR 527, 702, 716, 725
HPSR 533, 702, 717, 726
HRDRA 535, 702, 717, 726
HRDRB 537, 702, 717, 726
HREVR 512, 701, 715, 725
HRPSR 540, 702, 717, 726
HRSR 538, 702, 717, 726
IAR 308
IAR_0 ..... 699, 709, 723
IAR_1 ..... 699, 710, 723
IAR_2 ..... 699, 710, 723
IAR_3 ..... 700, 711, 723
ICR0 ..... 204, 699, 706, 722
ICR1 ..... 205, 699, 706, 722
ICR2 ..... 207, 699, 706, 722
IER0 ..... 563, 702, 718, 726
IFR0 ..... 551, 702, 718, 726
IMCR 210
IMCR0 ..... 698, 706, 722
IMCR1 ..... 698, 706, 722
IMCR4 ..... 698, 706, 722
IMCR5 ..... 698, 706, 722
IMCR6 ..... 698, 706, 722
IMCR9 ..... 698, 706, 722
IMR 208
IMR0 ..... 698, 706, 722
IMR1 ..... 698, 706, 722
IMR4 ..... 698, 706, 722
IMR5 ..... 698, 706, 722
IMR6 ..... 698, 706, 722
IMR9 ..... 698, 706, 722
INTEVT2 ..... 157, 159, 698, 705, 722
IPR 202
IPRA 698, 705, 722
IPRB 698, 706, 722
IPRC 698, 706, 722
IPRD 698, 706, 722
IPRE 698, 706, 722
IPRF 698, 706, 722
IPRG 698, 706, 722

IPRH 698, 706, 722
IRR0 ..... 208, 699, 706, 722
ISR0 ..... 558, 702, 718, 726
MCCR 365, 700, 711, 724
NCCR 654, 703, 720, 727
PACR 641, 703, 719, 727
PADR 628, 703, 719, 727
PBCR 643, 703, 719, 727
PBDR 629, 703, 719, 727
PDCR 644, 703, 719, 727
PDDR 631, 703, 719, 727
PECR 646, 703, 720, 727
PEDR 633, 703, 719, 727
PGCR 647, 703, 720, 727
PGDR 636, 703, 719, 727
PHCR 649, 703, 720, 727
PHDR 637, 703, 719, 727
PSELA 650, 703, 720, 727
RTCNT 246, 699, 708, 723
RTCOR 247, 699, 708, 723
RTCSR 245, 699, 708, 723
RWTCNT 247, 699, 708, 723
SAR 301
SAR_0 ..... 699, 723
SAR_1 ..... 699, 709, 723
SAR_2 ..... 699, 710, 723
SAR_3 ..... 699, 710, 723
SAR0 709
SCBRR 456
SCBRR_0 ..... 701, 714, 725
SCBRR_1 ..... 701, 714, 725
SCFCR 458
SCFCR_0 ..... 701, 714, 725
SCFCR_1 ..... 701, 715, 725
SCFDR 461
SCFDR_0 ..... 701, 714, 725
SCFDR_1 ..... 701, 715, 725
SCFER 449
SCFER_0 ..... 701, 714, 725
SCFER_1 ..... 701, 714, 725
SCFRDR 440
SCFRDR_0 ..... 701, 714, 725
SCFRDR_1 ..... 701, 715, 725
SCFTDR 441
SCFTDR_0 ..... 701, 714, 725
SCFTDR_1 ..... 701, 715, 725
SCRSR 440
SCSCR 445
SCSCR_0 ..... 701, 714, 725
SCSCR_1 ..... 701, 714, 725
SCSMR 442
SCSMR_0 ..... 701, 714, 725
SCSMR_1 ..... 701, 714, 725
SCSSR 450
SCSSR_0 ..... 701, 714, 725
SCSSR_1 ..... 701, 714, 725
SCTDSR 461
SCTDSR_0 ..... 701, 714, 725
SCTDSR_1 ..... 701, 714, 725
SCTSR 440
SDBPR 685
SDBSR 686
SDCR 242, 699, 708, 723
SDID 691
SDID/SDIDH 704, 721, 728
SDID/SDIDL 704, 721, 728
SDIR 685, 704, 721, 728
SDMR3 ..... 699, 708, 723
SICDAR 406, 700, 713, 724
SICTR 387, 700, 713, 724
SIFCTR 401, 700, 713, 724
SIIER 399, 701, 713, 724
SIMDR 384, 700, 713, 724
SIRCR 393, 701, 714, 725
SIRDAR 405, 700, 713, 724
SIRDR 391, 701, 713, 724
SISCR 403, 700, 713, 724
SISTR 394, 700, 713, 724
SITCR 392, 701, 713, 725
SITDAR 404, 700, 713, 724
SITDR 390, 701, 713, 724
SPICR 408, 701, 714, 725
STBCR 360, 700, 711, 724
STBCR2 ..... 361, 700, 711, 724
STBCR3 ..... 362, 700, 711, 724
STBCR4 ..... 363, 700, 711, 724

TCNT 376
TCNT_0 ..... 700, 712, 724
TCNT_1 ..... 700, 712, 724
TCNT_2 ..... 700, 712, 724
TCOR 376
TCOR_0 ..... 700, 712, 724
TCOR_1 ..... 700, 712, 724
TCOR_2 ..... 700, 712, 724
TCR 374
TCR_0 ..... 700, 712, 724
TCR_1 ..... 700, 712, 724
TCR_2 ..... 700, 712, 724
TEA 157, 698, 705, 722
TRA 156, 698, 705, 722
TRG 571, 703, 719, 726
TSR 576, 703, 719, 727
TSTR 374, 700, 712, 724
U memory 728
WTCNT 349, 700, 711, 724
WTCSR 349, 700, 711, 724
X/Y memory 728
Repeat end register 78
Repeat mode transfer 315
Repeat start register 78
Reset state 29
Resets 164
RF-IC 614
Round-robin mode 318
Save program counter 43
Save status register 43
SDRAM interface 259
Sequential break 674
Serial clocks 410
Serial communication interface with FIFO 435
Serial I/O with FIFO 381
Shadow area 222
Single address mode 323
Single data transfer instructions 98
Sleep mode 366
Software standby mode 366
SPI mode 430
Stall operations 602
State transitions 369
Status register 36, 41, 77
System registers 36, 39
T bit 47
TAP controller 692
Timer unit 371
U memory 195
U0 area 32
USB function controller 547
USB host controller 509
USB pin multiplex controller 499
USB standard commands 601
User break controller 655
User debugging interface 683
User mode 30
Uxy area 33
Vector base register 44
Voice codec IC 616
Watchdog timer 347
Watchdog timer mode 353
Write hit 184
Write miss 184
X/Y memory 191

SH7660 Hardware Manual

Publication Date:	Rev. 1.00, February 6, 2004
Published by:	Sales Strategic Planning Div. Renesas Technology Corp.
Edited by:	Technical Documentation & Information Department Renesas Kodaira Semiconductor Co., Ltd.

©2004 Renesas Technology Corp. All rights reserved. Printed in Japan.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

RENESAS SALES OFFICES

http://www.renesas.com

Renesas Technology America, Inc.
450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500 Fax: <1> (408) 382-7501

Renesas Technology Europe Limited.
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, United Kingdom
Tel: <44> (1628) 585 100, Fax: <44> (1628) 585 900

Renesas Technology Europe GmbH
Dornacher Str. 3, D-85622 Feldkirchen, Germany
Tel: <49> (89) 380 70 0, Fax: <49> (89) 929 30 11

Renesas Technology Hong Kong Ltd.
7/F., North Tower, World Finance Centre, Harbour City, Canton Road, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2375-6836

Renesas Technology Taiwan Co., Ltd.
FL 10, #99, Fu-Hsing N. Rd., Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

Renesas Technology (Shanghai) Co., Ltd.
26/F., Ruijin Building, No.205 Maoming Road (S), Shanghai 200020, China
Tel: <86> (21) 6472-1001, Fax: <86> (21) 6415-2952

Renesas Technology Singapore Pte. Ltd.
1, Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

SH7660
Hardware Manual



Renesas Technology Corp.
2-6-2, Ote-machi, Chiyoda-ku, Tokyo, 100-0004, Japan

REJ09B0032-0100Z