Hitachi SuperH™ RISC engine
SH7729

Hardware Manual
—Preliminary—

HITACHI

ADE-602-157
Rev. 0.5
4/23/99
Hitachi, Ltd.

Notice

When using this document, keep the following in mind:

1. This document may, wholly or partially, be subject to change without notice.

2. All rights are reserved: No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without Hitachi's permission.

3. Hitachi will not be held responsible for any damage to the user that may result from accidents or any other reasons during operation of the user's unit according to this document.

4. Circuitry and other examples described herein are meant merely to indicate the characteristics and performance of Hitachi's semiconductor products. Hitachi assumes no responsibility for any intellectual property claims or other problems that may result from applications based on the examples described herein.

5. No license is granted by implication or otherwise under any patents or other rights of any third party or Hitachi, Ltd.

6. MEDICAL APPLICATIONS: Hitachi's products are not authorized for use in MEDICAL APPLICATIONS without the written consent of the appropriate officer of Hitachi's sales company. Such use includes, but is not limited to, use in life support systems. Buyers of Hitachi's products are requested to notify the relevant Hitachi sales offices when planning to use the products in MEDICAL APPLICATIONS.

Contents

Section 1 Overview and Pin Functions.......... 1
1.1 SH7729 Features.......... 1
1.2 Block Diagram.......... 6
1.3 Pin Dcscription.......... 7
1.3.1 Pin Assignment.......... 7
1.3.2 Pin Function.......... 9

Section 2 Programming Model.......... 19
2.1 Programming Model.......... 19
2.1.1 Overview.......... 19
2.1.2 Execution Environment when DSP Extension is Disabled.......... 23
2.1.3 Execution Environment when DSP Extension is Enabled.......... 24
2.2 Registers.......... 24
2.2.1 General Purpose Registers.......... 25
2.2.2 Control Registers.......... 27
2.2.3 System Registers.......... 30
2.2.4 DSP Registers.......... 30
2.3 Data Format.......... 35
2.3.1 Data Format in Registers (Non-DSP Type).......... 35
2.3.2 DSP-Type Data Format.......... 35
2.3.3 Data Format in Memory.......... 37

Section 3 DSP Operation.......... 39
3.1 Data Operations of DSP Unit.......... 39
3.1.1 ALU Fixed-Point Operations.......... 39
3.1.2 ALU Integer Operations.......... 44
3.1.3 ALU Logical Operations.......... 46
3.1.4 Fixed-Point Multiply Operation.......... 47
3.1.5 Shift Operations.......... 49
3.1.6 Most Significant Bit Detection Operation.......... 53
3.1.7 Rounding Operation.......... 56
3.1.8 Overflow Protection.......... 58
3.1.9 Data Transfer Operation.......... 58
3.1.10 Local Data Move Operation.......... 62
3.1.11 Opcrand Conflict.......... 63
3.2 DSP Addressing.......... 64
3.2.1 DSP Loop Control.......... 64
3.2.2 DSP Data Addrcssing.......... 71

Section 4 Instruction Set 79
4.1 Basic Concept of SH7729 Instruction Set 79
4.2 SH1, SH2, SH3 Compatible Instruction Set 79
4.2.1 Instruction Set by Classification 79
4.3 Instructions for DSP Extension 94
4.3.1 Introduction 94
4.3.2 Additional Systcm Control Instruction for CPU 100
4.3.3 Single- and Double-Data Transfer for DSP Instructions 102
4.3.4 Parallel Operation for the DSP Unit 106

Section 5 Memory Management Unit (MMU) 119
5.1 Overview 119
5.1.1 Features 119
5.1.2 Role of MMU 119
5.1.3 Virtual Address Space 122
5.1.4 Register Configuration 125
5.2 Register Description 126
5.3 TLB Functions 128
5.3.1 Configuration of the TLB 128
5.3.2 TLB Indexing 130
5.3.3 TLB Address Comparison 131
5.3.4 Page Management Information 133
5.4 MMU Functions 134
5.4.1 MMU Hardware Management 134
5.4.2 MMU Software Management 134
5.4.3 MMU Instruction (LDTLB) 135
5.4.4 Avoiding Synonym Problems 136
5.5 MMU Exceptions 139
5.5.1 TLB Miss Exception 139
5.5.2 TLB Protection Violation Exception 140
5.5.3 TLB Invalid Exception 141
5.5.4 Initial Page Write Exception 142
5.5.5 Processing Flow in Event of MMU Exception
(Same Processing Flow for Address Error) 144
5.5.6 MMU Exception in Repeat Loop 146
5.6 Memory-Mapped TLB 148
5.6.1 Address Array 148
5.6.2 Data Array 149
5.6.3 Usage Examples 151
5.7 Usage Note 151

Section 6 Exception Processing 153
6.1 Overview 153
6.1.1 Features 153
6.1.2 Register Configuration 153
6.2 Exception Processing Function 153
6.2.1 Exception Processing Flow 153
6.2.2 Exception Processing Vector Addresses 154
6.2.3 Acceptance of Exceptions 156
6.2.4 Exception Codes 158
6.2.5 Exception Request Masks 159
6.2.6 Returning from Exception Handling 159
6.3 Register Description 160
6.4 Exception Handler Operation 160
6.4.1 Reset 160
6.4.2 Interrupts 161
6.4.3 General Exceptions 161
6.5 Individual Exception Operations 162
6.5.1 Resets 162
6.5.2 General Exceptions 163
6.5.3 Interrupts 167
6.6 Cautions 169

Section 7 Cache 171
7.1 Overview 171
7.1.1 Features 171
7.1.2 Cache Structure 171
7.1.3 Register Configuration 173
7.2 Register Description 173
7.2.1 Cache Control Register (CCR) 173
7.2.2 Cache Control Register 2 (CCR2) 174
7.3 Cache Operation 176
7.3.1 Searching the Cache 176
7.3.2 Read Access 178
7.3.3 Write Access 178
7.3.4 Write-Back Buffer 178
7.3.5 Coherency of Cache and External Memory 179
7.4 Memory-Mapped Cache 179
7.4.1 Address Array 179
7.4.2 Data Array 180
7.5 Usage Examples 182
7.5.1 Invalidating Specific Entries 182
7.5.2 Reading the Data of a Specific Entry 182

Section 8 X/Y Memory 183
8.1 Overview 183
8.1.1 Features 183
8.2 X/Y Memory Access from the CPU 183
8.3 X/Y Memory Access from the DSP 184
8.4 X/Y Memory Access from the DMAC 185

Section 9 Interrupt Controller (INTC) 187
9.1 Overview 187
9.1.1 Features 187
9.1.2 Block Diagram 188
9.1.3 Pin Configuration 189
9.1.4 Register Configuration 190
9.2 Interrupt Sources 191
9.2.1 NMI Interrupts 191
9.2.2 IRQ Interrupt 191
9.2.3 IRL Interrupts 192
9.2.4 PINT Interrupt 194
9.2.5 On-Chip Peripheral Module Interrupts 194
9.2.6 Interrupt Exception Processing and Priority 195
9.3 INTC Registers 201
9.3.1 Interrupt Priority Registers A to E (IPRA–IPRE) 201
9.3.2 Interrupt Control Register 0 (ICR0) 202
9.3.3 Interrupt Control Register 1 (ICR1) 203
9.3.4 Interrupt Control Register 2 (ICR2) 206
9.3.5 PINT Interrupt Enable Register (PINTER) 207
9.3.6 Interrupt Request Register 0 (IRR0) 208
9.3.7 Interrupt Request Register 1 (IRR1) 210
9.3.8 Interrupt Request Register 2 (IRR2) 211
9.4 INTC Operation 213
9.4.1 Interrupt Sequence 213
9.4.2 Multiple Interrupts 215
9.5 Interrupt Response Time 215

Section 10 User Break Controller 219
10.1 Overview 219
10.1.1 Features 219
10.1.2 Block Diagram 221
10.1.3 Register Configuration 222
10.2 Register Descriptions 223
10.2.1 Break Address Register A (BARA) 223
10.2.2 Break Address Mask Register A (BAMRA) 224
10.2.3 Break Bus Cycle Register A (BBRA) 225

10.2.4 Break Address Register B (BARB)... 227
10.2.5 Break Address Mask Register B (BAMRB).. 228
10.2.6 Break Data Register B (BDRB)... 229
10.2.7 Break Data Mask Register B (BDMRB).. 230
10.2.8 Break Bus Cycle Register B (BBRB).. 231
10.2.9 Break Control Register (BRCR)... 233
10.2.10 Exccution Timcs Brcak Rcgistcr (BETR).. 239
10.2.11 Branch Source Register (BRSR).. 240
10.2.12 Branch Destination Register (BRDR)... 241
10.2.13 Break ASID Register A (BASRA)... 242
10.2.14 Break ASID Register B (BASRB)... 242
10.2.15 I/O Break Address Register (IOBAR).. 243
10.2.16 I/O Break Data Register (IOBDR)... 244
10.3 Operation Description.. 245
10.3.1 Flow of the User Break Operation.. 245
10.3.2 Break on Instruction Fetch Cycle... 246
10.3.3 Break by Data Access Cycle.. 246
10.3.4 Break on X/Y-Memory Bus Cycle... 247
10.3.5 Sequential Break.. 247
10.3.6 Value of Saved Program Counter... 247
10.3.7 PC Trace.. 248
10.3.8 Usage Examples... 250
10.3.9 Notes.. 255

Section 11 Power-Down Modes.. 257
11.1 Overview.. 257
11.1.1 Power-Down Modes... 257
11.1.2 Register Configuration... 259
11.1.3 Pin Configuration... 259
11.2 Register Description... 259
11.2.1 Standby Control Register (STBCR).. 259
11.2.2 Standby Control Register 2 (STBCR2)... 261
11.3 Sleep Mode.. 263
11.3.1 Transition to Sleep Mode... 263
11.3.2 Canceling Sleep Mode.. 263
11.4 Standby Mode.. 264
11.4.1 Transition to Standby Mode... 264
11.4.2 Canceling Standby Mode.. 264
11.4.3 Clock Pause Function... 265
11.5 Module Standby Function... 266
11.5.1 Transition to Modulc Standby Function.. 266
11.5.2 Clearing the Module Standby Function... 266
11.6 Timing of STATUS Pin Changes... 267

11.6.1 Timing for Resets 267
11.6.2 Timing for Canceling Standbys 269
11.6.3 Timing for Canceling Sleep Mode 270
11.7 Hardware Standby Function 272
11.7.1 Transition to Hardware Standby Mode 272
11.7.2 Canceling Hardware Standby Mode 273
11.7.3 Hardwarc Standby Modc Timing 273

Section 12 On-Chip Oscillation Circuits 275
12.1 Overview 275
12.1.1 Features 275
12.2 Overview of the CPG 276
12.2.1 CPG Block Diagram 276
12.2.2 CPG Pin Configuration 278
12.2.3 CPG Register Configuration 278
12.3 Clock Operating Modes 279
12.4 Register Descriptions 284
12.4.1 Frequency Control Register (FRQCR) 284
12.5 Changing the Frequency 287
12.5.1 Changing the Multiplication Rate 287
12.5.2 Changing the Division Ratio 287
12.6 PLL Standby Function 288
12.6.1 Overview of the PLL Standby Function 288
12.6.2 Usage 288
12.7 Controlling Clock Output 289
12.7.1 Clock Modes 0–2 289
12.7.2 Clock Modes 3, 4 289
12.8 Overview of the WDT 290
12.8.1 Block Diagram of the WDT 290
12.8.2 Register Configurations 290
12.9 WDT Registers 291
12.9.1 Watchdog Timer Counter (WTCNT) 291
12.9.2 Watchdog Timer Control/Status Register (WTCSR) 291
12.9.3 Notes on Register Access 293
12.10 Using the WDT 294
12.10.1 Canceling Standbys 294
12.10.2 Changing the Frequency 294
12.10.3 Using Watchdog Timer Mode 295
12.10.4 Using Interval Timer Mode 295
12.11 Notes on Board Design 296

Section 13 Bus State Controller (BSC) 299
13.1 Overview 299
13.1.1 Features 299
13.1.2 Block Diagram 300
13.1.3 Pin Configuration 302
13.1.4 Register Configuration 304
13.1.5 Area Overview 305
13.1.6 PCMCIA Support 308
13.2 BSC Registers 312
13.2.1 Bus Control Register 1 (BCR1) 312
13.2.2 Bus Control Register 2 (BCR2) 316
13.2.3 Bus Control Register 3 (BCR3) 317
13.2.4 Wait State Control Register 1 (WCR1) 320
13.2.5 Wait State Control Register 2 (WCR2) 321
13.2.6 Individual Memory Control Register (MCR) 324
13.2.7 DRAM Control Register (DCR) 329
13.2.8 PCMCIA Control Register (PCR) 331
13.2.9 SDRAM Mode Register (SDMR) 335
13.2.10 Refresh Timer Control/Status Register (RTCSR) 336
13.2.11 Refresh Timer Counter (RTCNT) 338
13.2.12 Refresh Time Constant Register (RTCOR) 339
13.2.13 Refresh Count Register (RFCR) 339
13.2.14 Cautions on Accessing Refresh Control Related Registers 340
13.2.15 MCS0 Control Register (MCSCR0) 341
13.2.16 MCS1 Control Register (MCSCR1) 342
13.2.17 MCS2 Control Register (MCSCR2) 342
13.2.18 MCS3 Control Register (MCSCR3) 342
13.2.19 MCS4 Control Register (MCSCR4) 342
13.2.20 MCS5 Control Register (MCSCR5) 342
13.2.21 MCS6 Control Register (MCSCR6) 342
13.2.22 MCS7 Control Register (MCSCR7) 342
13.3 BSC Operation 343
13.3.1 Endian/Access Size and Data Alignment 343
13.3.2 Description of Areas 349
13.3.3 Basic Interface 352
13.3.4 DRAM Interface 360
13.3.5 Synchronous DRAM Interface 377
13.3.6 Burst ROM Interface 404
13.3.7 PCMCIA Interface 407
13.3.8 Waits between Access Cycles 419
13.3.9 Bus Arbitration 420
13.3.10 Bus Pull-Up 421
13.3.11 $\overline{\text{MCS0}}$ to $\overline{\text{MCS7}}$ Pin Control 423

Section 14 Direct Memory Access Controller (DMAC)........ 427
14.1 Overview........ 427
14.1.1 Features........ 427
14.1.2 Block Diagram........ 429
14.1.3 Pin Configuration........ 430
14.1.4 Register Configuration........ 431
14.2 Register Descriptions........ 432
14.2.1 DMA Source Address Registers 0–3 (SAR0–SAR3)........ 432
14.2.2 DMA Destination Address Registers 0–3 (DAR0–DAR3)........ 433
14.2.3 DMA Transfer Count Registers 0–3 (DMATCR0–DMATCR3)........ 434
14.2.4 DMA Channel Control Registers 0–3 (CHCR0–CHCR3)........ 435
14.2.5 DMA Operation Register (DMAOR)........ 441
14.3 Operation........ 443
14.3.1 DMA Transfer Flow........ 443
14.3.2 DMA Transfer Requests........ 445
14.3.3 Channel Priority........ 447
14.3.4 DMA Transfer Types........ 450
14.3.5 Number of Bus Cycle States and $\overline{\text{DREQ}}$ Pin Sampling Timing........ 463
14.3.6 Source Address Reload Function........ 469
14.3.7 DMA Transfer Ending Conditions........ 471
14.4 Compare Match Timer (CMT)........ 473
14.4.1 Overview........ 473
14.4.2 Register Descriptions........ 474
14.4.3 Operation........ 477
14.4.4 Compare Match........ 478
14.5 Examples of Use........ 480
14.5.1 Example of DMA Transfer between On-Chip IRDA and External Memory........ 480
14.5.2 Example of DMA Transfer between AD0 and External Memory (Address Reload on)........ 481
14.5.3 Example of DMA Transfer between AD0 and External Memory (Address Reload on)........ 483
14.5.4 Example of DMA Transfer between External Memory and SCIF Transmitter (Indirect Address on)........ 484
14.6 Cautions........ 486

Section 15 Timer (TMU)........ 487
15.1 Overview........ 487
15.1.1 Features........ 487
15.1.2 Block Diagram........ 488
15.1.3 Pin Configuration........ 489
15.1.4 Register Configuration........ 489
15.2 TMU Registers........ 490
15.2.1 Timer Output Control Register (TOCR)........ 490

15.2.2 Timer Start Register (TSTR) 490
15.2.3 Timer Control Register (TCR) 491
15.2.4 Timer Constant Register (TCOR) 494
15.2.5 Timer Counters (TCNT) 495
15.2.6 Input Capture Register (TCPR2) 497
15.3 TMU Operation 498
15.3.1 Overview 498
15.3.2 Basic Functions 498
15.4 Interrupts 502
15.4.1 Status Flag Set Timing 502
15.4.2 Status Flag Clear Timing 503
15.4.3 Interrupt Sources and Priorities 503
15.5 Usage Notes 504
15.5.1 Writing to Registers 504
15.5.2 Reading Registers 504

Section 16 Realtime Clock (RTC) 505
16.1 Overview 505
16.1.1 Features 505
16.1.2 Block Diagram 505
16.1.3 Pin Configuration 507
16.1.4 RTC Register Configuration 508
16.2 RTC Registers 509
16.2.1 64-Hz Counter (R64CNT) 509
16.2.2 Second Counter (RSECCNT) 509
16.2.3 Minute Counter (RMINCNT) 510
16.2.4 Hour Counter (RHRCNT) 510
16.2.5 Day of the Week Counter (RWKCNT) 511
16.2.6 Date Counter (RDAYCNT) 512
16.2.7 Month Counter (RMONCNT) 512
16.2.8 Year Counter (RYRCNT) 513
16.2.9 Second Alarm Register (RSECAR) 513
16.2.10 Minute Alarm Register (RMINAR) 514
16.2.11 Hour Alarm Register (RHRAR) 514
16.2.12 Day of the Week Alarm Register (RWKAR) 515
16.2.13 Date Alarm Register (RDAYAR) 516
16.2.14 Month Alarm Register (RMONAR) 516
16.2.15 RTC Control Register 1 (RCR1) 517
16.2.16 RTC Control Register 2 (RCR2) 518
16.3 RTC Operation 520
16.3.1 Initial Settings of Registers after Power-On 520
16.3.2 Setting the Time 520
16.3.3 Reading the Time 522

16.3.4 Alarm Function 523
16.3.5 Crystal Oscillator Circuit 524

Section 17 Serial Communication Interface (SCI) 525
17.1 Overview 525
17.1.1 Features 525
17.1.2 Block Diagram 526
17.1.3 Pin Configuration 529
17.1.4 Register Configuration 530
17.2 Register Descriptions 530
17.2.1 Receive Shift Register 530
17.2.2 Receive Data Register 530
17.2.3 Transmit Shift Register 531
17.2.4 Transmit Data Register 531
17.2.5 Serial Mode Register 532
17.2.6 Serial Control Register 534
17.2.7 Serial Status Register 537
17.2.8 Port SC Control Register (SCPCR)/Port SC Data Register (SCPDR) 541
17.2.9 Bit Rate Register (SCBRR) 543
17.3 Operation 551
17.3.1 Overview 551
17.3.2 Operation in Asynchronous Mode 553
17.3.3 Multiprocessor Communication 563
17.3.4 Clock Synchronous Operation 572
17.4 SCI Interrupt Sources 582
17.5 Notes on Use 583

Section 18 Smart Card Interface 587
18.1 Overview 587
18.1.1 Features 587
18.1.2 Block Diagram 588
18.1.3 Pin Configuration 589
18.1.4 Register Configuration 589
18.2 Register Descriptions 590
18.2.1 Smart Card Mode Register (SCSCMR) 590
18.2.2 Serial Status Register (SCSSR) 591
18.3 Operation 592
18.3.1 Overview 592
18.3.2 Pin Connections 593
18.3.3 Data Format 594
18.3.4 Register Settings 595
18.3.5 Clock 596
18.3.6 Data Transmission and Reception 599

18.4 Usage Notes 605
18.4.1 Receive Data Timing and Receive Margin in Asynchronous Mode........ 605
18.4.2 Retransmission (Receive and Transmit Modes)........ 607

Section 19 Serial Communication Interface with FIFO (SCIF)........ 609
19.1 Overview........ 609
19.1.1 Features........ 609
19.1.2 Block Diagram........ 610
19.1.3 Pin Configuration 613
19.1.4 Register Configuration........ 614
19.2 Register Descriptions........ 614
19.2.1 Receive Shift Register........ 614
19.2.2 Receive FIFO Data Register........ 615
19.2.3 Transmit Shift Register........ 615
19.2.4 Transmit FIFO Data Register 615
19.2.5 Serial Mode Register........ 616
19.2.6 Serial Control Register........ 618
19.2.7 Serial Status Register........ 620
19.2.8 Bit Rate Register (SCBRR)........ 625
19.2.9 FIFO Control Register (SCFCR)........ 632
19.2.10 Register of the Number of FIFO Data (SCFDR) 634
19.3 Operation........ 635
19.3.1 Overview 635
19.3.2 Serial Operation........ 636
19.4 SCIF Interrupts........ 645
19.5 Notes on Use........ 646

Section 20 IrDA........ 649
20.1 Overview........ 649
20.1.1 Features........ 649
20.1.2 Block Diagram........ 650
20.1.3 Pin Configuration 653
20.1.4 Register Configuration........ 654
20.2 Register Description 654
20.2.1 Serial Mode Register (SCSMR)........ 654
20.3 Operation Description 656
20.3.1 Overview........ 656
20.3.2 Transmit........ 657
20.3.3 Receive 657

Section 21 Pin Function Controller........ 659
21.1 Overview........ 659
21.2 Register Configuration........ 663

21.3 Register Descriptions.. 664
21.3.1 Port A Control Register (PACR).. 664
21.3.2 Port B Control Register (PBCR).. 665
21.3.3 Port C Control Register (PCCR).. 666
21.3.4 Port D Control Register (PDCR).. 667
21.3.5 Port E Control Register (PECR).. 669
21.3.6 Port F Control Register (PFCR)... 670
21.3.7 Port G Control Register (PGCR).. 671
21.3.8 Port H Control Register (PHCR).. 672
21.3.9 Port J Control Register (PJCR).. 674
21.3.10 Port K Control Register (PKCR).. 675
21.3.11 Port L Control Register (PLCR).. 676
21.3.12 SC Port Control Register (SCPCR).. 677

Section 22 I/O Ports... 681
22.1 Overview.. 681
22.2 Port A 681
22.2.1 Register Descriptions.. 681
22.2.2 Port A Data Register (PADR) 682
22.3 Port B... 683
22.3.1 Register Descriptions.. 683
22.3.2 Port B Data Register (PBDR).. 684
22.4 Port C 685
22.4.1 Register Descriptions.. 685
22.4.2 Port C Data Register (PCDR) 686
22.5 Port D 687
22.5.1 Register Descriptions.. 687
22.5.2 Port D Data Register (PDDR) 687
22.6 Port E... 689
22.6.1 Register Descriptions.. 689
22.6.2 Port E Data Register (PEDR).. 690
22.7 Port F... 691
22.7.1 Register Descriptions.. 691
22.7.2 Port F Data Register (PFDR).. 692
22.8 Port G 693
22.8.1 Register Descriptions.. 693
22.8.2 Port G Data Register (PGDR) 694
22.9 Port H 695
22.9.1 Register Descriptions.. 695
22.9.2 Port H Data Register (PHDR) 696
22.10 Port J 697
22.10.1 Register Descriptions.. 697
22.10.2 Port J Data Register (PJDR) 697

22.11 Port K 699
22.11.1 Register Descriptions 699
22.11.2 Port K Data Register (PKDR) 700
22.12 Port L 701
22.12.1 Register Descriptions 701
22.12.2 Port L Data Register (PLDR) 702
22.13 SC Port 703
22.13.1 Register Descriptions 703
22.13.2 Port SC Data Register (SCPDR) 703

Section 23 A/D Converter 705
23.1 Overview 705
23.1.1 Features 705
23.1.2 Block Diagram 706
23.1.3 Input Pins 707
23.1.4 Register Configuration 708
23.2 Register Descriptions 709
23.2.1 A/D Data Registers A to D (ADDRA to ADDRD) 709
23.2.2 A/D Control/Status Register (ADCSR) 710
23.3 Bus Master Interface 713
23.4 Operation 714
23.4.1 Single Mode (MULTI = 0) 714
23.4.2 Multi Mode (MULTI = 1) 716
23.4.3 Scan Mode (MULTI = 1, SCN = 1) 718
23.4.4 Input Sampling and A/D Conversion Time 718
23.4.5 External Trigger Input Timing 720
23.5 Interrupts 720
23.6 Definitions of A/D Conversion Accuracy 720
23.7 A/D Converter Usage Notes 721
23.7.1 Setting Analog Input Voltage 721
23.7.2 Processing of Analog Input Pins 722
23.7.3 Access Size and Read Data 722

Section 24 D/A Converter 725
24.1 Overview 725
24.1.1 Features 725
24.1.2 Block Diagram 725
24.1.3 I/O Pins 726
24.1.4 Register Configuration 726
24.2 Register Descriptions 727
24.2.1 D/A Data Registers 0 and 1 (DADR0/1) 727
24.2.2 D/A Control Register (DACR) 727
24.3 Operation 729

Section 25 Hitachi User-Debugging Interface and Advanced User Debugger... 731
25.1 Overview....... 731
25.2 Hitachi User Debugging Interface (H-UDI)....... 731
25.2.1 Hardware Resources and Pins....... 732
25.2.2 Pin Description 732
25.2.3 Block Diagram....... 733
25.3 Control Registers....... 734
25.3.1 Bypass Register (SDBPR)....... 735
25.3.2 Instruction Register (SDIR)....... 735
25.3.3 Status Register (SDSR)....... 736
25.3.4 Data Register (SDDR)....... 737
25.3.5 Address Register (SDAR and SDARE)....... 737
25.3.6 SDPC and SDR0....... 738
25.4 H-UDI Operations....... 739
25.4.1 Example of Data Transfer from Emulator to CPU....... 740
25.4.2 Reset Configuration....... 741
25.4.3 H-UDI Reset....... 742
25.4.4 H-UDI Break....... 742
25.4.5 H-UDI Interrupt....... 742
25.4.6 Bypass....... 743
25.4.7 ASERAM Write....... 743
25.4.8 H-UDI Boot....... 743
25.4.9 Boot Sequence....... 744
25.4.10 Using H-UDI to Recover from Sleep Mode....... 744
25.5 CPU Operations when ASE Break Occurs....... 745
25.5.1 ASEEVT Configuration....... 745
25.6 Notes on Use....... 746
25.7 Advanced User Debugger (AUD)....... 747
25.7.1 Block Diagram....... 748
25.7.2 Pin Configuration....... 748
25.7.3 Pin Description....... 749
25.8 Realtime Trace Mode....... 751
25.8.1 Overview....... 751
25.8.2 Description on Operation....... 751
25.9 Full Trace Mode....... 753
25.9.1 Overview....... 753
25.9.2 Description on Operation....... 753
25.10 Notes on Use....... 754

Section 26 Electrical Characteristics....... 755
26.1 Absolute Maximum Ratings....... 755
26.2 DC Characteristics....... 755
26.3 AC Characteristics....... 759

26.3.1 Clock Timing.. 760
26.3.2 Control Signal Timing.. 769
26.3.3 AC Bus Timing.. 772
26.3.4 Basic Timing.. 774
26.3.5 Burst ROM Timing.. 778
26.3.6 DRAM Timing.. 781
26.3.7 Synchronous DRAM Timing.. 793
26.3.8 PCMCIA Timing.. 811
26.3.9 Peripheral Module Signal Timing.. 818
26.3.10 ASE-Related Pin Timing.. 821
26.3.11 AC Characteristics Measurement Conditions.. 824
26.4 A/D Converter Characteristics.. 825
26.5 D/A Converter Characteristics.. 825

Appendix A Pin Functions.. 827
A.1 Pin States.. 827
A.2 Pin Specifications.. 831
A.3 Processing of Unused Pins.. 835
A.4 Pin States in Access to Each Address Space.. 836

Appendix B Memory-Mapped Control Registers.. 864
B.1 Register Address Map.. 864
B.2 Register Bits.. 872

Appendix C Package Dimensions.. 884

Section 1 Overview and Pin Functions

1.1 SH7729 Features

This LSI is a single-chip RISC microprocessor that integrates a 32-bit RISC-type SuperH RISC engine architecture CPU with digital signal processing (DSP) extension as its core that has a cache memory, an on-chip X/Y memory, and a memory management unit (MMU) as well as peripheral functions required for system configuration such as a timer, a realtime clock, an interrupt controller, and a serial communication interface. This LSI includes data protection and virtual memory functions, and was designed by building a memory management unit onto an SuperH RISC engine microprocessor (SH-1 or SH-2). SH7729 chip has the same peripheral modules as SH7709 but with larger cache size, on-chip DSP module, on-chip X/Y memory, and emulator support.

High-speed data transfers with a direct memory access controller (DMAC) are implemented. An external memory access support function enables direct connection to each memory. The SH7729 microprocessor also supports, an infrared communication function, an A/D converter, and a D/A converter.

A powerful built-in power management function keeps power consumption low, even during high-speed operation. This LSI can run at eight times the frequency of the system-bus operating speed, making it optimum for electrical devices such as PDA requiring both high speed and low power.

The features of this LSI are listed in table 1.1.

Table 1.1 SH7729 Features

Item	Features
CPU	• Original Hitachi SuperH architecture • Object code level compatible with SH-1, SH-2 and SH-3 (SH7708) • 32-bit internal data paths • General-register files — Sixteen 32-bit general registers (eight 32-bit shadow registers) — Eight 32-bit control registers — Four 32-bit system registers • RISC-type instruction set — Instruction length: 16-bit fixed length for improved code efficiency — Load-store architecture — Delayed branch instructions — Instruction set based on C language • Instruction execution time: one instruction/cycle for basic instructions • Logical address space: 4 Gbytes • Space identifier ASID: 8 bits, 256 logical address space • Five-stage pipeline
DSP	• Mixture of 16-bit and 32-bit instructions • 32-/40-bit internal data paths • Multiplier, ALU, barrel shifter and register file • 16 bits x 16 bits → 32-bit one cycle multiplier • Large DSP data register file — Six 32-bit data registers — Two 40-bit data registers • Extended Harvard Architecture for DSP datapath — Two data buses — One instruction bus • Max. 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, indexing (with or without modulo addressing) • Zero overhead repeat loop control • Conditional execution instructions • User-DSP mode and privileged-DSP mode

Table 1.1 SH7729 Features (cont)

Item	Features
Clock pulse generator (CPG)	• Clock mode: An input clock can be selected from the external input (EXTAL or CKIO) or crystal oscillator. • Three types of clocks generated: — CPU clock: 1–16 times the input clock, maximum 133 MHz — Bus clock: 1–4 times the input clock, maximum 66 MHz — Peripheral clock: 1/4–4 times the input clock, maximum 33 MHz • Power-down modes: — Sleep mode — Standby mode — Module standby mode • One-channel watchdog timer
Memory management unit (MMU)	• 4 Gbytes of address space, 256 address spaces (ASID 8 bits) • Page unit sharing • Supports multiple page sizes: 1, 4 Kbytes • 128-entry, 4-way set associative TLB • Supports software selection of replacement method and random-replacement algorithms • Contents of TLB are directly accessible by address mapping
Cache memory	• 16-kbyte cache, mixed instruction/data • 256 entries, 4-way set associative, 16-byte block length • Write-back, write-through, LRU replacement algorithm • 1-stage write-back buffer • Maximum 2 ways of the cache can be locked
X/Y memory	• User-selectable mapping mechanism — Fixed mapping for mission-critical realtime applications — Automatic mapping through TLB for easy to use • 3 independent read/write ports — 8-/16-/32-bit access from the CPU — Maximum two 16-bit accesses from the DSP — 8-/16-/32-access from the DMAC • 8-KB RAM for X and Y memory each

Table 1.1 SH7729 Features (cont)

Item	Features
Interrupt controller	• Eleven external interrupt pins (NMI, IRQ5–IRQ0, IRL3–IRL0) • On-chip peripheral interrupts: set priority levels for each module
User break controller	• 2 break channels • Addresses, data values, type of access, and data size can all be set as break conditions • Supports a sequential break function
Bus state controller	• Physical address space divided into six areas, each a maximum 64 Mbytes, with the following features settable for each area: — Bus size (8, 16, or 32 bits) — Number of wait cycles (also supports a hardware wait function) — Setting the type of space enables direct connection to SRAM, DRAM, SDRAM, and burst ROM — Supports PCMCIA interface (2 channels) — Outputs chip select signal (CS0, CS2–CS6) for corresponding area • DRAM/SDRAM refresh function — Programmable refresh interval — Supports CAS-before-RAS refresh and self-refresh modes — Supports power-down DRAM • DRAM/SDRAM burst access function • Usable as either big or little endian machine
User-debugging Interface	• E10A emulator support • JTAG-standard pin assignment • Realtime branch address trace • 1-KB on-chip RAM for fast emulation program execution
Timer	• 3-channel auto-reload-type 32-bit timer • Input capture function • 6 types of counter input clocks can be selected • Maximum resolution: 2 MHz
Realtime clock	• Built-in clock, calendar functions, and alarm functions • On-chip 32-kHz crystal oscillator circuit with a maximum resolution (cycle interrupt) of 1/256 second

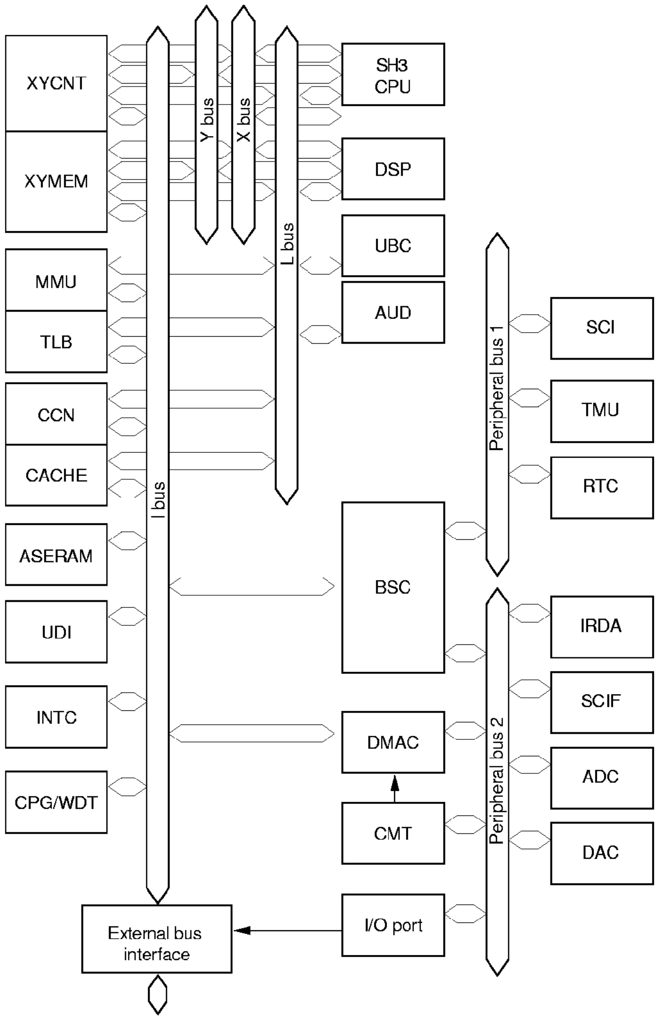
Table 1.1 SH7729 Features (cont)

Item	Features
Serial communication interface 0	• Asynchronous mode or clock synchronous mode can be selected • Full-duplex communication • Supports smart card interface
Serial communication interface 1	• 16-byte FIFO for transmission/reception • DMA can be transferred • IrDA: interface based on 1.0
Serial communication interface 2	• 16-byte FIFO for transmission/reception • DMA can be transferred • Hardware flow control
DMA controller	• 4 channels • Burst mode and cycle-steal mode
I/O port	• 16 bits (at 16-bit external bus)
A/D converter	• 10 bits ± 4 LSB, 8 channels • Conversion time: 10 µs • Input range: 0–Vcc (max. 3.6 V)
D/A converter	• 8 bits ± 4 LSB, 2 channels • Conversion time: 10 µs • Output range: 0–Vcc (max. 3.6 V)
Package	• 208-pin plastic LQFP (FP-208C) • 216-pin CSP (BT-216)

Table 1.2 Characteristics

Item	Characteristics
Power supply voltage	• I/O: 3.3 ± 0.3 V, Internal: 1.8 ± 0.3 V
Operating frequency	• Internal frequency: maximum 133 MHz, and external frequency: maximum 66 MHz
Process	• 0.25-µm CMOS/3-layer metal

1.2 Block Diagram



Notes:

ADC:	A/D converter
ASERAM:	ASE memory
AUD:	Advanced user debugger
BSC:	Bus state controller
CACHE:	Cache memory
CCN:	Cache memory controller
CMT:	Compare match timer
CPG/WDT:	Clock pulse generator/watchdog timer
CPU:	Central processing unit
DAC:	D/A converter
DMAC:	Direct memory access controller
DSP:	Digital signal processor
INTC:	Interrupt controller
IRDA:	Serial communicatiion interface (with IRDA)
MMU:	Memory management unit
RTC:	Realtime clock
SCI:	Serial communication interface (with smart card interface)
SCIF:	Serial communication interface (with FIFO)
TLB:	Address translation buffer
TMU:	Timer unit
UBC:	User break controller
UDI:	User-debugging interface
XYCNT:	XY memory controller
XYMEM:	XY memory

Figure 1.1 Block Diagram

1.3 Pin Description

1.3.1 Pin Assignment

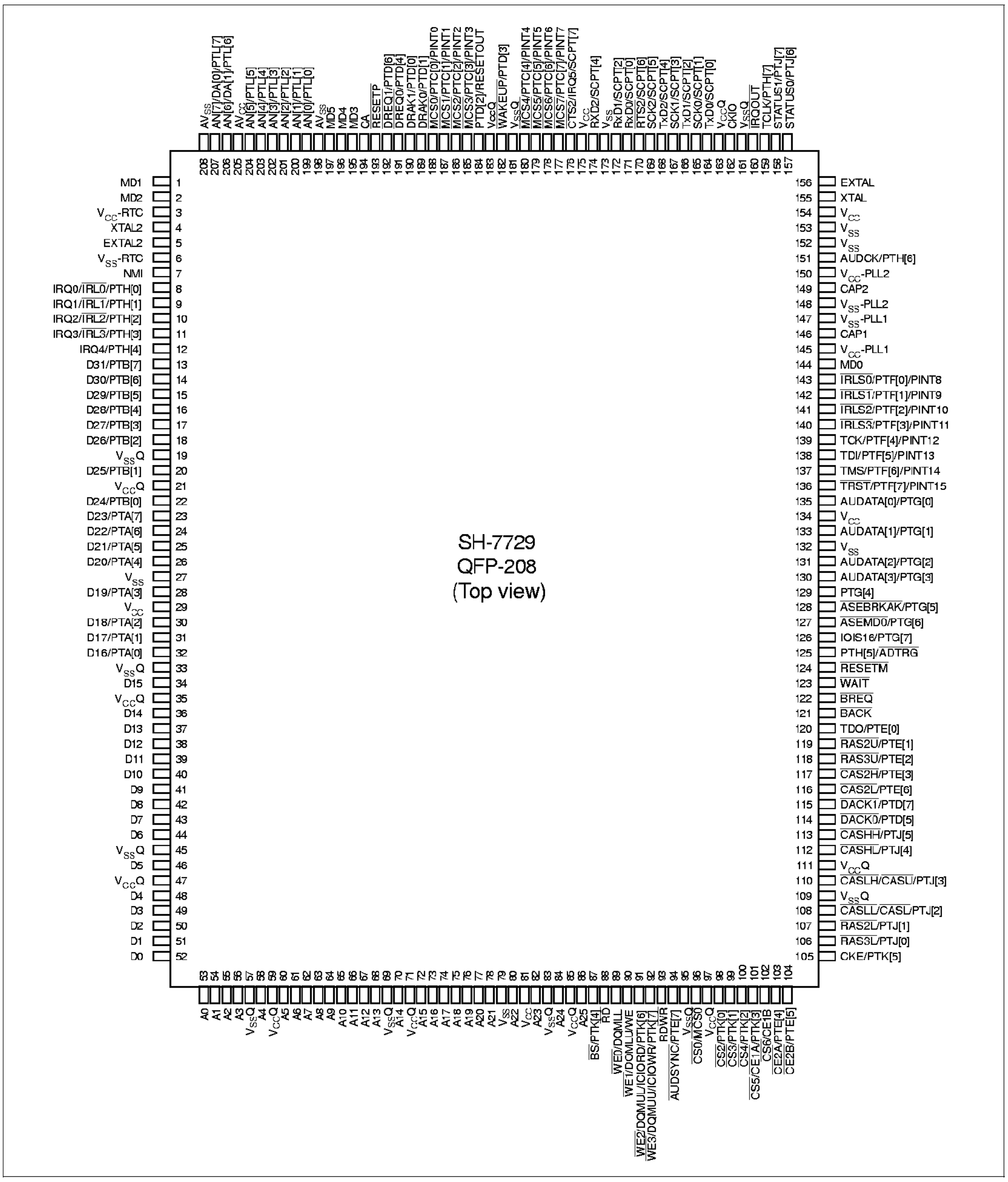
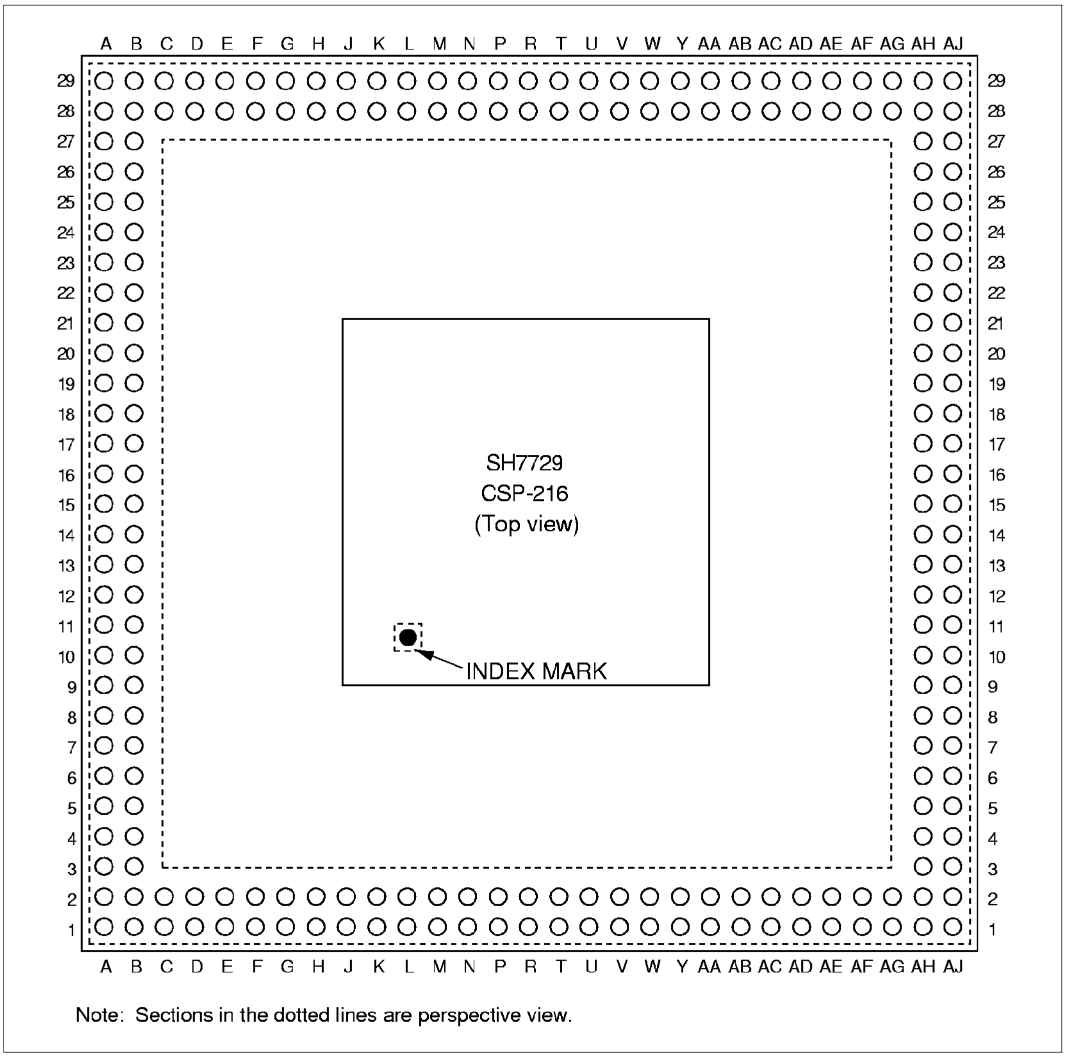

SH-7729
QFP-208
(Top view)
1 MD1
2 MD2
3 V_{CC}-RTC
4 XTAL2
5 EXTAL2
6 V_{SS}-RTC
7 NMI
8 IRQ0/$\overline{\text{IRL0}}$/PTH[0]
9 IRQ1/$\overline{\text{IRL1}}$/PTH[1]
10 IRQ2/$\overline{\text{IRL2}}$/PTH[2]
11 IRQ3/$\overline{\text{IRL3}}$/PTH[3]
12 IRQ4/PTH[4]
13 D31/PTB[7]
14 D30/PTB[6]
15 D29/PTB[5]
16 D28/PTB[4]
17 D27/PTB[3]
18 D26/PTB[2]
19 V_{SS}Q
20 D25/PTB[1]
21 V_{CC}Q
22 D24/PTB[0]
23 D23/PTA[7]
24 D22/PTA[6]
25 D21/PTA[5]
26 D20/PTA[4]
27 V_{SS}
28 D19/PTA[3]
29 V_{CC}
30 D18/PTA[2]
31 D17/PTA[1]
32 D16/PTA[0]
33 V_{SS}Q
34 D15
35 V_{CC}Q
36 D14
37 D13
38 D12
39 D11
40 D10
41 D9
42 D8
43 D7
44 D6
45 V_{SS}Q
46 D5
47 V_{CC}Q
48 D4
49 D3
50 D2
51 D1
52 D0
53 A0
54 A1
55 A2
56 A3
57 V_{SS}Q
58 A4
59 V_{CC}Q
60 A5
61 A6
62 A7
63 A8
64 A9
65 A10
66 A11
67 A12
68 A13
69 V_{SS}Q
70 A14
71 V_{CC}Q
72 A15
73 A16
74 A17
75 A18
76 A19
77 A20
78 A21
79 V_{SS}
80 A22
81 V_{CC}
82 A23
83 V_{SS}Q
84 A24
85 V_{CC}Q
86 A25
87 $\overline{\text{BS}}$/PTK[4]
88 $\overline{\text{RD}}$
89 $\overline{\text{WE0}}$/DQMLL
90 $\overline{\text{WE1}}$/DOMLU/$\overline{\text{WE}}$
91 $\overline{\text{WE2}}$/DQMUL/ICIORD/PTK[6]
92 $\overline{\text{WE3}}$/DQMUU/ICIOWR/PTK[7]
93 RD$\overline{\text{WR}}$
94 $\overline{\text{AUDSYNC}}$/PTE[7]
95 V_{SS}Q
96 $\overline{\text{CS0}}$/$\overline{\text{MCS0}}$
97 V_{CC}Q
98 $\overline{\text{CS2}}$/PTK[0]
99 $\overline{\text{CS3}}$/PTK[1]
100 $\overline{\text{CS4}}$/PTK[2]
101 $\overline{\text{CS5}}$/$\overline{\text{CE1A}}$/PTK[3]
102 $\overline{\text{CS6}}$/$\overline{\text{CE1B}}$
103 $\overline{\text{CE2A}}$/PTE[4]
104 $\overline{\text{CE2B}}$/PTE[5]
105 CKE/PTK[5]
106 $\overline{\text{RAS3L}}$/PTJ[0]
107 $\overline{\text{RAS2L}}$/PTJ[1]
108 $\overline{\text{CASLL}}$/$\overline{\text{CASL}}$/PTJ[2]
109 V_{SS}Q
110 $\overline{\text{CASLH}}$/$\overline{\text{CASU}}$/PTJ[3]
111 V_{CC}Q
112 $\overline{\text{CASHL}}$/PTJ[4]
113 $\overline{\text{CASHH}}$/PTJ[5]
114 $\overline{\text{DACK0}}$/PTD[5]
115 $\overline{\text{DACK1}}$/PTD[7]
116 $\overline{\text{CAS2L}}$/PTE[6]
117 $\overline{\text{CAS2H}}$/PTE[3]
118 $\overline{\text{RAS3U}}$/PTE[2]
119 $\overline{\text{RAS2U}}$/PTE[1]
120 TDO/PTE[0]
121 $\overline{\text{BACK}}$
122 $\overline{\text{BREQ}}$
123 $\overline{\text{WAIT}}$
124 $\overline{\text{RESETM}}$
125 PTH[5]/$\overline{\text{ADTRG}}$
126 IOIS16/PTG[7]
127 $\overline{\text{ASEMD0}}$/PTG[6]
128 $\overline{\text{ASEBRKAK}}$/PTG[5]
129 PTG[4]
130 AUDATA[3]/PTG[3]
131 AUDATA[2]/PTG[2]
132 V_{SS}
133 AUDATA[1]/PTG[1]
134 V_{CC}
135 AUDATA[0]/PTG[0]
136 $\overline{\text{TRST}}$/PTF[7]/PINT15
137 TMS/PTF[6]/PINT14
138 TDI/PTF[5]/PINT13
139 TCK/PTF[4]/PINT12
140 $\overline{\text{IRLS3}}$/PTF[3]/PINT11
141 $\overline{\text{IRLS2}}$/PTF[2]/PINT10
142 $\overline{\text{IRLS1}}$/PTF[1]/PINT9
143 $\overline{\text{IRLS0}}$/PTF[0]/PINT8
144 MD0
145 V_{CC}-PLL1
146 CAP1
147 V_{SS}-PLL1
148 V_{SS}-PLL2
149 CAP2
150 V_{CC}-PLL2
151 AUDCK/PTH[6]
152 V_{SS}
153 V_{SS}
154 V_{CC}
155 XTAL
156 EXTAL
157 STATUS0/PTJ[6]
158 STATUS1/PTJ[7]
159 TCLK/PTH[7]
160 $\overline{\text{IRQOUT}}$
161 V_{SS}Q
162 CKIO
163 V_{CC}Q
164 TxD0/SCPT[0]
165 SCK0/SCPT[1]
166 TxD1/SCPT[2]
167 SCK1/SCPT[3]
168 TxD2/SCPT[4]
169 SCK2/SCPT[5]
170 $\overline{\text{RTS2}}$/SCPT[6]
171 RxD0/SCPT[0]
172 RxD1/SCPT[2]
173 V_{SS}
174 RXD2/SCPT[4]
175 V_{CC}
176 $\overline{\text{CTS2}}$/IRQ5/SCPT[7]
177 $\overline{\text{MCS7}}$/PTC[7]/PINT7
178 $\overline{\text{MCS6}}$/PTC[6]/PINT6
179 $\overline{\text{MCS5}}$/PTC[5]/PINT5
180 $\overline{\text{MCS4}}$/PTC[4]/PINT4
181 V_{SS}Q
182 $\overline{\text{WAKEUP}}$/PTD[3]
183 VccQ
184 PTD[2]/RESETOUT
185 $\overline{\text{MCS3}}$/PTC[3]/PINT3
186 $\overline{\text{MCS2}}$/PTC[2]/PINT2
187 $\overline{\text{MCS1}}$/PTC[1]/PINT1
188 $\overline{\text{MCS0}}$/PTC[0]/PINT0
189 DRAK0/PTD[1]
190 DRAK1/PTD[0]
191 $\overline{\text{DREQ0}}$/PTD[4]
192 $\overline{\text{DREQ1}}$/PTD[6]
193 $\overline{\text{RESETP}}$
194 CA
195 MD3
196 MD4
197 MD5
198 AV_{SS}
199 AN[0]/PTL[0]
200 AN[1]/PTL[1]
201 AN[2]/PTL[2]
202 AN[3]/PTL[3]
203 AN[4]/PTL[4]
204 AN[5]/PTL[5]
205 AV_{CC}
206 AN[6]/DA[1]/PTL[6]
207 AN[7]/DA[0]/PTL[7]
208 AV_{SS}

Figure 1.2 Pin Assignment (FP-208C)



Note: Sections in the dotted lines are perspective view.

Figure 1.3 Pin Assignment (CSP-216)

1.3.2 Pin Function

Table 1.3 SH7729 Pin Function

Number of Pins				
QFP-208C	**CSP-216**	**Pin Name**	**I/O**	**Description**
1	B02	MD1	I	Clock mode setting
2	A02	MD2	I	Clock mode setting
3	B03	Vcc-RTC^{*1}	—	RTC power supply (1.8 V)
4	A03	XTAL2	O	On-chip RTC crystal oscillator pin
5	B04	EXTAL2	I	On-chip RTC crystal oscillator pin
6	A04	Vss-RTC^{*1}	—	RTC power supply (0 V)
7	B05	NMI	I	Nonmaskable interrupt request
8	A05	IRQ0/$\overline{\text{IRL0}}$/PTH[0]	I	External interrupt request/input port H
9	B06	IRQ1/$\overline{\text{IRL1}}$/PTH[1]	I	External interrupt request/input port H
10	A06	IRQ2/$\overline{\text{IRL2}}$/PTH[2]	I	External interrupt request/input port H
11	B07	IRQ3/$\overline{\text{IRL3}}$/PTH[3]	I	External interrupt request/input port H
12	A07	IRQ4/PTH[4]	I	External interrupt request/input port H
13	B08	D31/PTB[7]	I/O	Data bus / I/O port B
14	A08	D30/PTB[6]	I/O	Data bus / I/O port B
15	B09	D29/PTB[5]	I/O	Data bus / I/O port B
16	A09	D28/PTB[4]	I/O	Data bus / I/O port B
17	B10	D27/PTB[3]	I/O	Data bus / I/O port B
18	A10	D26/PTB[2]	I/O	Data bus / I/O port B
19	B11	VssQ	—	Input/output power supply (0 V)
20	A11	D25/PTB[1]	I/O	Data bus / I/O port B
21	B12	VccQ	—	Input/output power supply (3.3 V)
22	A12	D24/PTB[0]	I/O	Data bus / I/O port B
23	B13	D23/PTA[7]	I/O	Data bus / I/O port A
24	A13	D22/PTA[6]	I/O	Data bus / I/O port A
25	B14	D21/PTA[5]	I/O	Data bus / I/O port A

Table 1.3 SH7729 Pin Function (cont)

Number of Pins				
QFP-208C	**CSP-216**	**Pin Name**	**I/O**	**Description**
26	A14	D20/PTA[4]	I/O	Data bus / I/O port A
27	B15	Vss	—	Power supply (0 V)
28	A15	D19/PTA[3]	I/O	Data bus / I/O port A
29	B16	Vcc	—	Power supply (1.8 V)
30	A16	D18/PTA[2]	I/O	Data bus / I/O port A
31	B17	D17/PTA[1]	I/O	Data bus / I/O port A
32	A17	D16/PTA[0]	I/O	Data bus / I/O port A
33	B18	VssQ	—	Input/output power supply (0 V)
34	A18	D15	I/O	Data bus
35	B19	VccQ	—	Input/output power supply (3.3 V)
36	A19	D14	I/O	Data bus
37	B20	D13	I/O	Data bus
38	A20	D12	I/O	Data bus
39	B21	D11	I/O	Data bus
40	A21	D10	I/O	Data bus
41	B22	D9	I/O	Data bus
42	A22	D8	I/O	Data bus
43	B23	D7	I/O	Data bus
44	A23	D6	I/O	Data bus
45	B24	VssQ	—	Input/output power supply (0 V)
46	A24	D5	I/O	Data bus
47	B25	VccQ	—	Input/output power supply (3.3 V)
48	A25	D4	I/O	Data bus
49	B26	D3	I/O	Data bus
50	A26	D2	I/O	Data bus
51	B27	D1	I/O	Data bus
52	A27	D0	I/O	Data bus
53	B28	A0	O	Address bus
54	B29	A1	O	Address bus
55	C28	A2	O	Address bus

Table 1.3 SH7729 Pin Function (cont)

Number of Pins				
QFP-208C	**CSP-216**	**Pin Name**	**I/O**	**Description**
56	C29	A3	O	Address bus
57	D28	VssQ	—	Input/output power supply (0 V)
58	D29	A4	O	Address bus
59	E28	VccQ	—	Input/output power supply (3.3 V)
60	E29	A5	O	Address bus
61	F28	A6	O	Address bus
62	F29	A7	O	Address bus
63	G28	A8	O	Address bus
64	G29	A9	O	Address bus
65	H28	A10	O	Address bus
66	H29	A11	O	Address bus
67	J28	A12	O	Address bus
68	J29	A13	O	Address bus
69	K28	VssQ	—	Input/output power supply (0 V)
70	K29	A14	O	Address bus
71	L28	VccQ	—	Input/output power supply (3.3 V)
72	L29	A15	O	Address bus
73	M28	A16	O	Address bus
74	M29	A17	O	Address bus
75	N28	A18	O	Address bus
76	N29	A19	O	Address bus
77	P28	A20	O	Address bus
78	P29	A21	O	Address bus
79	R28	Vss	—	Power supply (0 V)
80	R29	A22	O	Address bus
81	T28	Vcc	—	Power supply (1.8 V)
82	T29	A23	O	Address bus
83	U28	VssQ	—	Input/output power supply (0 V)
84	U29	A24	O	Address bus
85	V28	VccQ	—	Input/output power supply (3.3 V)

Table 1.3 SH7729 Pin Function (cont)

Number of Pins				
QFP-208C	CSP-216	Pin Name	I/O	Description
86	V29	A25	O	Address bus
87	W28	BS/PTK[4]	O / I/O	Bus cycle start signal / I/O port K
88	W29	RD	O	Read strobe
89	Y28	WE0/DQMLL	O	D7–D0 select signal / DQM (SDRAM)
90	Y29	WE1/DQMLU/WE	O	D15–D8 select signal / DQM (SDRAM)
91	AA28	WE2/DQMUL/ICIORD/ PTK[6]	O / I/O	D23–D16 select signal / DQM (SDRAM) / PCMCIA I/O read / I/O port K
92	AA29	WE3/DQMUU/ICIOWR/ PTK[7]	O / I/O	D31–D24 select signal / DQM (SDRAM) / PCMCIA I/O read / I/O port K
93	AB28	RDWR	O	Read/write
94	AB29	AUDSYNC/PTE[7]	O / I/O	AUD synchronous / I/O port E
95	AC28	VssQ	—	Input/output power supply (0 V)
96	AC29	CS0/MCS[0]	O	Chip select 0/mask ROM chip select 0
97	AD28	VccQ	—	Input/output power supply (3.3 V)
98	AD29	CS2/PTK[0]	O / I/O	Chip select 2 / I/O port K
99	AE28	CS3/PTK[1]	O / I/O	Chip select 3 / I/O port K
100	AE29	CS4/PTK[2]	O / I/O	Chip select 4 / I/O port K
101	AF28	CS5/CE1A/PTK[3]	O / I/O	Chip select 5/CE1 (area 5 PCMCIA) / I/O port K
102	AF29	CS6/CE1B	O	Chip select 6/CE1 (area 6 PCMCIA) / I/O port K
103	AG28	CE2A/PTE[4]	O / I/O	Area 5 PCMCIA card enable / I/O port E
104	AG29	CE2B/PTE[5]	O / I/O	Area 6 PCMCIA card enable / I/O port E
105	AH28	CKE/PTK[5]	O / I/O	CK enable (SDRAM) / I/O port K
106	AJ28	RAS3L/PTJ[0]	O / I/O	Lower 32 MB address (area 3 DRAM, SDRAM) RAS / I/O port J

Table 1.3 SH7729 Pin Function (cont)

Number of Pins				
QFP-208C	**CSP-216**	**Pin Name**	**I/O**	**Description**
107	AH27	$\overline{\text{RAS2L}}$/ PTJ[1]	O / I/O	Lower 32 MB address (area 2 DRAM, SDRAM) RAS / I/O port J
108	AJ27	$\overline{\text{CASLL}}$/CASL/PTJ[2]	O / I/O	D7–D0 (DRAM) CAS / Lower 32 MB address (SDRAM) CAS / I/O port J
109	AH26	VssQ	—	Input/output power supply (0 V)
110	AJ26	$\overline{\text{CASLH}}$/CASU/PTJ[3]	O / I/O	D15–D8 (DRAM) CAS / Lower 32 MB address (SDRAM) CAS / I/O port J
111	AH25	VccQ	—	Input/output power supply (3.3 V)
112	AJ25	$\overline{\text{CASHL}}$/PTJ[4]	O / I/O	D23–D16 (DRAM) CAS / I/O port J
113	AH24	$\overline{\text{CASHH}}$/PTJ[5]	O / I/O	D31–D24 (DRAM) CAS / I/O port J
114	AJ24	$\overline{\text{DACK0}}$/PTD[5]	O / I/O	DMA acknowledge 0 / I/O port D
115	AH23	$\overline{\text{DACK1}}$/PTD[7]	O / I/O	DMA acknowledge 1 / I/O port D
116	AJ23	$\overline{\text{CAS2L}}$/PTE[6]	O / I/O	D7–D0 (area 2 DRAM) CAS / I/O port E
117	AH22	$\overline{\text{CAS2H}}$/PTE[3]	O / I/O	D15–D8 (area 2 DRAM) CAS / I/O port E
118	AJ22	$\overline{\text{RAS3U}}$/PTE[2]	O / I/O	Upper 32 MB address (area 3 DRAM, SDRAM) RAS / I/O port E
119	AH21	$\overline{\text{RAS2U}}$/PTE[1]	O / I/O	Upper 32 MB address (area 2 DRAM) RAS / I/O port E
120	AJ21	TDO/PTE[0]	I/O	Test data output / I/O port E
121	AH20	$\overline{\text{BACK}}$	O	Bus acknowledge
122	AJ20	$\overline{\text{BREQ}}$	I	Bus request
123	AH19	$\overline{\text{WAIT}}$	I	Hardware wait request
124	AJ19	$\overline{\text{RESETM}}$	I	Manual reset request
125	AH18	$\overline{\text{ADTRG}}$/PTH[5]	I	Analog trigger / input port H
126	AJ18	$\overline{\text{IOIS16}}$/PTG[7]	I	Write protect / area 6 16-bit input/output / input port G
127	AH17	$\overline{\text{ASEMD0}}$/PTG[6]	I	ASE mode / input port G
128	AJ17	$\overline{\text{ASEBRKAK}}$/PTG[5]	O/I	ASE break acknowledge / input port G

Table 1.3 SH7729 Pin Function (cont)

Number of Pins				
QFP-208C	CSP-216	Pin Name	I/O	Description
129	AH16	PTG[04]	I	Input port G
130	AJ16	AUDATA[3]/PTG[3]	O/I	AUD data / input port G
131	AH15	AUDATA[2]/PTG[2]	O/I	AUD data / input port G
132	AJ15	Vss	—	Power supply (0 V)
133	AH14	AUDATA[1]/PTG[1]	O/I	AUD data / input port G
134	AJ14	Vcc	—	Power supply (1.8 V)
135	AH13	AUDATA[0]/PTG[0]	O/I	AUD data / input port G
136	AJ13	$\overline{\text{TRST}}$/PTF[7]/PINT[15]	I	Test reset / input port F / port interrupt
137	AH12	TMS/PTF[6]/PINT[14]	I	Test mode switch / input port F / port interrupt
138	AJ12	TDI/PTF[5]/PINT[13]	I	Test data input / input port F / port interrupt
139	AH11	TCK/PTF[4]/PNT[12]	I	Test clock / input port F / port interrupt
140	AJ11	$\overline{\text{IRLS}}$[3]/PTF[3]/ PINT[11]	I	External interrupt request / input port F / port interrupt
141	AH10	$\overline{\text{IRLS}}$[2]/PTF[2]/ PINT[10]	I	External interrupt request / input port F / port interrupt
142	AJ10	$\overline{\text{IRLS}}$[1]/PTF[1]/PINT[9]	I	External interrupt request / input port F / port interrupt
143	AH09	$\overline{\text{IRLS}}$[0]/PTF[0]/PINT[8]	I	External interrupt request / input port F / port interrupt
144	AJ09	MD0	I	Clock mode setting
145	AH08	Vcc-PLL1[*2]	—	PLL1 power supply (1.8 V)
146	AJ08	CAP1	—	PLL1 external capacitance pin
147	AH07	Vss-PLL1[*2]	—	PLL1 power supply (0 V)
148	AJ07	Vss-PLL2[*2]	—	PLL2 power supply (0 V)
149	AH06	CAP2	—	PLL2 external capacitance pin
150	AJ06	Vcc-PLL2[*2]	—	PLL2 power supply (1.8 V)
151	AH05	AUDCK/PTH[6]	I	AUD clock / input port H
152	AJ05	Vss	—	Power supply (0 V)

Table 1.3 SH7729 Pin Function (cont)

Number of Pins				
QFP-208C	CSP-216	Pin Name	I/O	Description
153	AH04	Vss	—	Power supply (0 V)
154	AJ04	Vcc	—	Power supply (1.8 V)
155	AH03	XTAL	O	Clock oscillator pin
156	AJ03	EXTAL	I	External clock / crystal oscillator pin
157	AH02	STATUS0/PTJ[6]	O / I/O	Processor status / I/O port J
158	AH01	STATUS1/PTJ[7]	O / I/O	Processor status / I/O port J
159	AG02	TCLK/PTH[7]	I/O	TMU or RTC clock input/output / I/O port H
160	AG01	$\overline{\text{IRQOUT}}$	O	Interrupt request notification
161	AF02	VssQ	—	Power supply (0 V)
162	AF01	CKIO	I/O	System clock input/output
163	AE02	VccQ	—	Power supply (3.3 V)
164	AE01	TxD0/SCPT[0]	O	Transmit data 0 / SCI output port
165	AD02	SOK0/SCPT[1]	I/O	Serial clock 0 / SCI I/O port
166	AD01	TxD1/SCPT[2]	O	Transmit data 1 / SCI output port
167	AC02	SCK1/SCPT[3]	I/O	Serial clock 1 / SCI I/O port
168	AC01	TxD2/SCPT[4]	O	Transmit data 2 / SCI output port
169	AB02	SCK2/SCPT[5]	I/O	Serial clock 2 / SCI I/O port
170	AB01	$\overline{\text{RTS2}}$/SCPT[6]	O / I/O	Transmit request 2 / SCI I/O port
171	AA02	RxD0/SCPT[0]	I	Transmit data 0 / SCI output port
172	AA01	RxD1/SCPT[2]	I	Transmit data 1 / SCI output port
173	Y02	Vss	—	Power supply (0 V)
174	Y01	RxD2/SCPT[4]	I	Transmit data 2 / SCI output port
175	W02	Vcc	—	Power supply (1.8 V)
176	W01	$\overline{\text{CTS2}}$/IRQ5/SCP[7]	I	Transmit clear 2 / external interrupt request / SCI input port
177	V02	$\overline{\text{MCS[7]}}$/PTC[7]/PINT[7]	O / I/O / I	Mask ROM chip select / I/O port C / port interrupt
178	V01	$\overline{\text{MCS[6]}}$/PTC[6]/PINT[6]	O / I/O / I	Mask ROM chip select / I/O port C / port interrupt

Table 1.3 SH7729 Pin Function (cont)

Number of Pins				
QFP-208C	**CSP-216**	**Pin Name**	**I/O**	**Description**
179	U02	$\overline{\text{MCS[5]}}$/PTC[5]/PINT[5]	O / I/O / I	Mask ROM chip select / I/O port C / port interrupt
180	U01	$\overline{\text{MCS[4]}}$/PTC[4]/PINT[4]	O / I/O / I	Mask ROM chip select / I/O port C / port interrupt
181	T02	VssQ	—	Input/output power supply (0 V)
182	T01	$\overline{\text{WAKEUP}}$/PTD[3]	O / I/O	Standby mode interrupt request notification / I/O port D
183	R02	VccQ	—	Input/output power supply (3.3 V)
184	R01	$\overline{\text{RESETOUT}}$/PTD[2]	O / I/O	Reset output / I/O port D
185	P02	$\overline{\text{MCS}}$[3]/PTC[3]/PINT[3]	O / I/O / I	Mask ROM chip select / I/O port C / port interrupt
186	P01	$\overline{\text{MCS}}$[2]/PTC[2]/PINT[2]	O / I/O / I	Mask ROM chip select / I/O port C / port interrupt
187	N02	$\overline{\text{MCS}}$[1]/PTC[1]/PINT[1]	O / I/O / I	Mask ROM chip select / I/O port C / port interrupt
188	N01	$\overline{\text{MCS}}$[0]/PTC[0]/PINT[0]	O / I/O / I	Mask ROM chip select / I/O port C / port interrupt
189	M02	DRAK0/PTD[1]	O / I/O	DMA request acknowledge / I/O port D
190	M01	DRAK1/PTD[0]	O / I/O	DMA request acknowledge / I/O port D
191	L02	$\overline{\text{DREQ0}}$/PTD[4]	I	DMA request / input port D
192	L01	$\overline{\text{DREQ1}}$/PTD[6]	I	DMA request / input port D
193	K02	$\overline{\text{RESETP}}$	I	Power-on reset request
194	K01	CA	I	Chip activate / hardware standby request
195	J02	MD3	I	Area 0 bus width setting
196	J01	MD4	I	Area 0 bus width setting
197	H02	MD5	I	Endian setting
198	H01	AVss	—	Analog power supply (0 V)
199	G02	AN[0]/PTL[0]	I	A/D converter input / input port L
200	G01	AN[1]/PTL[1]	I	A/D converter input / input port L
201	F02	AN[2]/PTL[2]	I	A/D converter input / input port L

Table 1.3 SH7729 Pin Function (cont)

Number of Pins				
QFP-208C	**CSP-216**	**Pin Name**	**I/O**	**Description**
202	F01	AN[3]/PTL[3]	I	A/D converter input / input port L
203	E02	AN[4]/PTL[4]	I	A/D converter input / input port L
204	E01	AN[5]/PTL[5]	I	A/D converter input / input port L
205	D02	AVcc (3.3V)	—	Analog power supply (3.3 V)
206	D01	AN[6]/DA[1]/PTL[6]	I	A/D converter input / input port L
207	C02	AN[7]/DA[0]/PTL[7]	I	A/D converter input / input port L
208	C01	AVss	—	Analog power supply (0 V)

Notes: 1. Must be connected to the power supply even when the RTC is not used.

2. Must be connected to the power supply even when the on-chip PLL circuits are not used (except in hardware standby mode).

3. Except in hardware standby mode, all V_{CC}/V_{SS} pins must be connected to the system power supply. (Supply power constantly.) In hardware standby mode, power must be supplied at least to V_{CC} (RTC) and V_{SS} (RTC). If power is not supplied to V_{CC} and V_{SS} pins other than V_{CC} (RTC) and V_{SS} (RTC), hold the CA pin low.

4. A01, A28, A29, AH29, AJ29, AJ02, AJ01, and B01 are NC pins. Do not make any connection to these pins.

Section 2 Programming Model

2.1 Programming Model

2.1.1 Overview

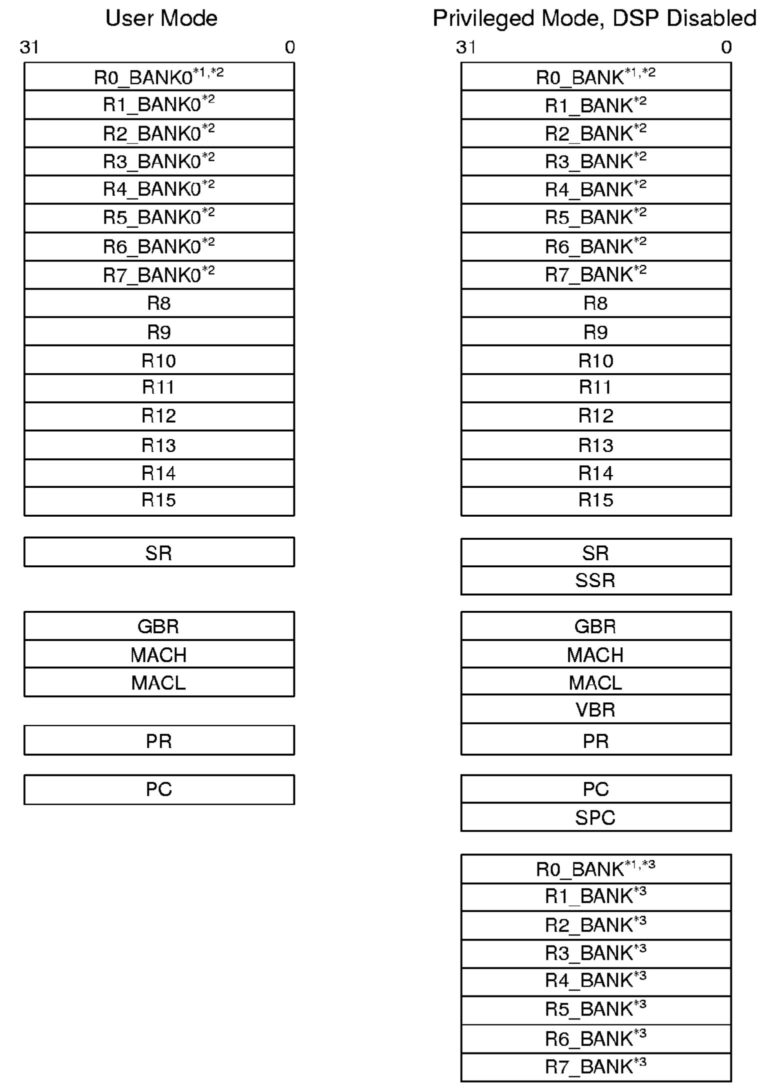
The SH7729 operates in user mode under normal conditions and enters privileged mode in response to an exception or interrupt. The DSP mode is enabled in privileged mode or in user mode under monitoring of the operating system. The processor mode is specified by the mode bit (MD) and DSP bit (DSP) in the status register (SR). SR.DSP can be changed only when MD = 1 (privileged mode).

1. User mode (without DSP extension): MD = 0, DSP = 0;
2. User-DSP mode (with DSP extension enabled): MD = 0, DSP = 1;
3. Privileged mode (without DSP extension): MD = 1, DSP = 0;
4. Privileged-DSP mode (with DSP extension enabled): MD = 1, DSP = 1;

The registers accessible to the programmer differ, depending on the processor mode. General-purpose registers R0 to R7 are banked registers which are switched by a processor mode change. Figure 2.1 shows the hardware resources for the user mode and privileged mode without DSP Extension. Figure 2.2 shows all resources available to software in the user-DSP mode. Figure 2.3 shows all resources available to software in the privileged-DSP mode. In the privileged mode (with or without DSP extension), the register bank (RB) bit in the SR defines which banked register set is accessed as general-purpose registers, and which set is accessed only through the load control register (LDC) and store control register (STC) instructions.

When the RB bit is 1, general-purpose registers R0_BANK1–R7_BANK1 in bank 1 and non-banked general-purpose registers R8–R15 function as the general-purpose register set, and general-purpose registers R0_BANK0–R7_BANK0 in bank 0 can be accessed only by the LDC/STC instructions.

When the RB bit is 0, general-purpose registers R0_BANK0–R7_BANK0 in bank 0 and non-banked general-purpose registers R8–R15 function as the general-purpose register set, and general-purpose registers R0_BANK1–R7_BANK1 in bank 1 can be accessed only by the LDC/STC instructions. In the user mode (MD = 0), general-purpose registers R0_BANK0–R7_BANK0 in bank 0 and non-banked general-purpose registers R8–R15 function as the general-purpose register set regardless of RB setting.

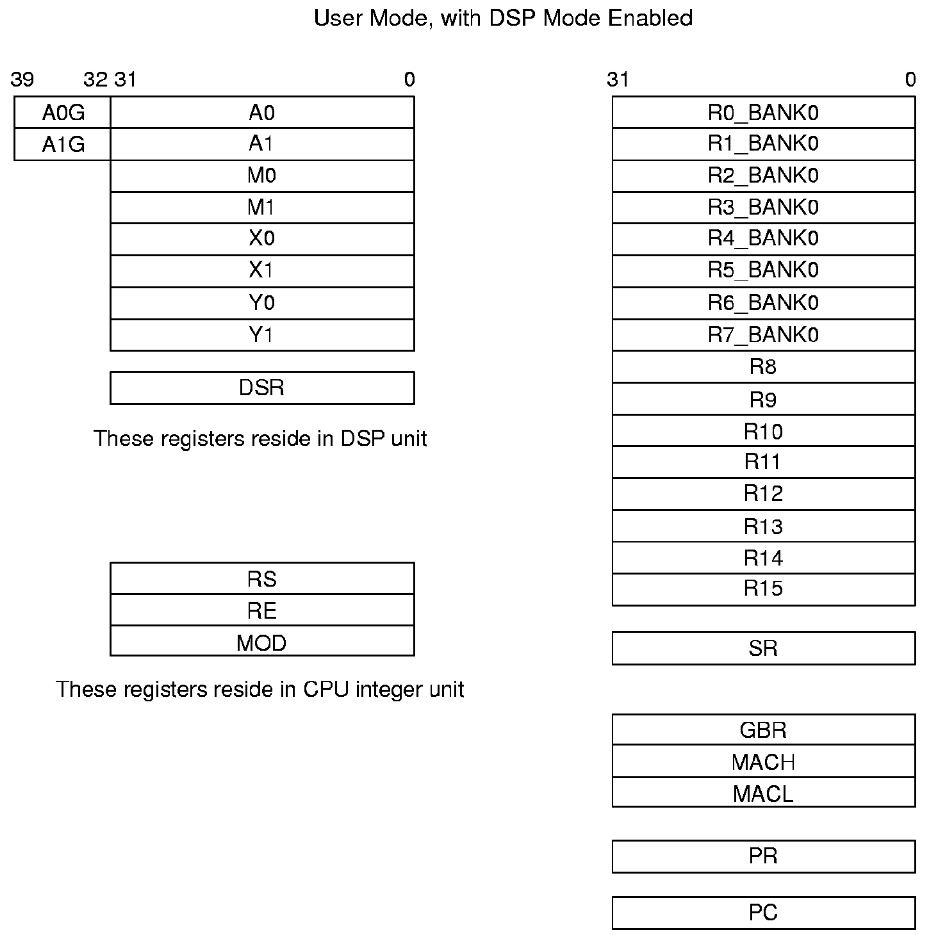


Note: 1. R0 functions as an index register in the indexed register-indirect addressing mode and indexed GBR-indirect addressing mode. In some instructions, only R0 can be used as the source register or destination register.

2. R0–R7 are banked registers. In user mode, BANK0 is used. In privileged mode, SR.RB specifies BANK.
SR.RB = 0; BANK0 is used
SR.RB = 1; BANK1 is used.

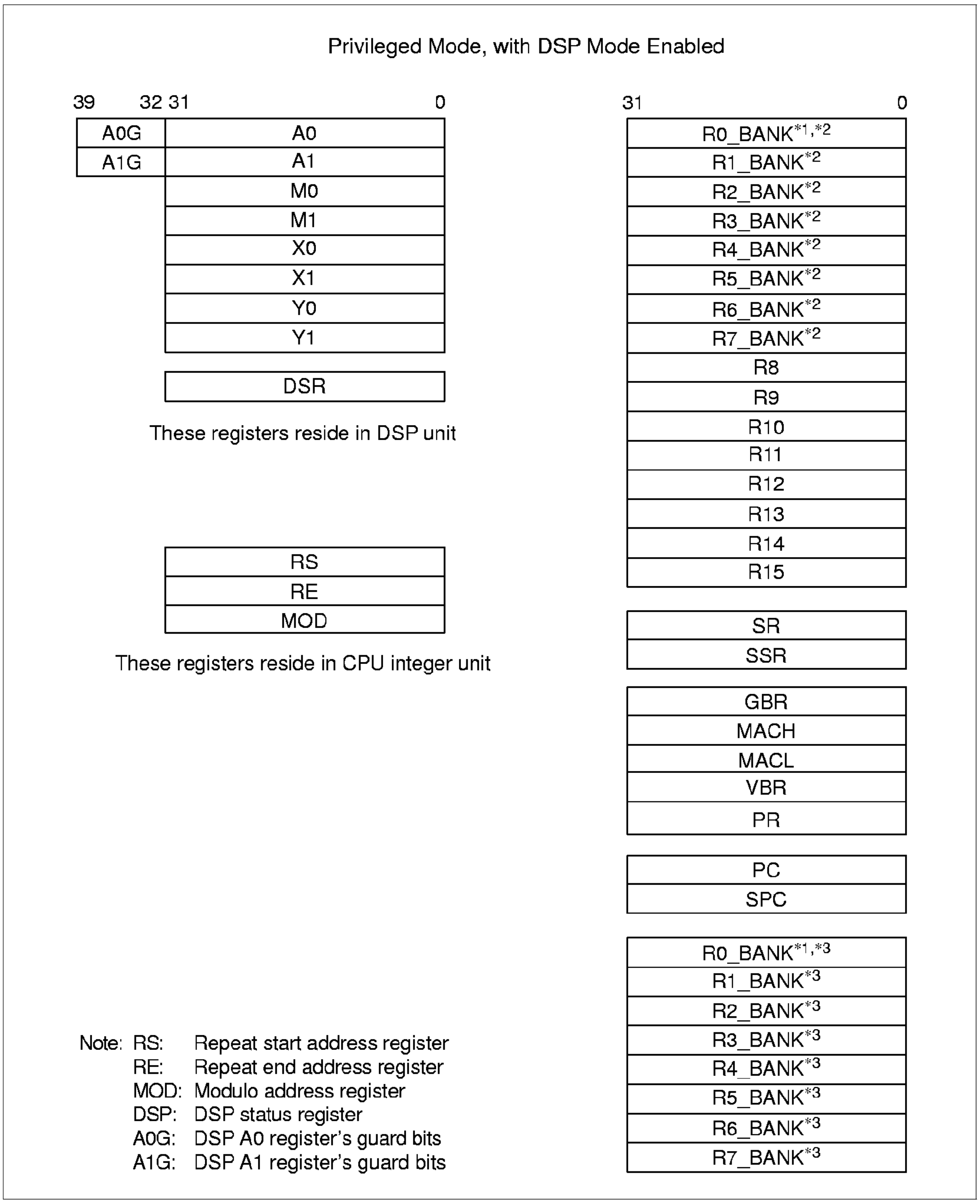
3. These registers are only accessed by LDC/STC instructions. SR.RB specifies BANK.
SR.RB = 0; BANK1 is used
SR.RB = 1; BANK0 is used

Figure 2.1 Programming Model (DSP Extension is Disabled)



Note: RS: Repeat start address register
RE: Repeat end address register
MOD: Modulo address register
DSP: DSP status register
A0G: DSP A0 register's guard bits
A1G: DSP A1 register's guard bits

Figure 2.2 Programming Model (User-DSP Mode)



Note: RS: Repeat start address register
RE: Repeat end address register
MOD: Modulo address register
DSP: DSP status register
A0G: DSP A0 register's guard bits
A1G: DSP A1 register's guard bits

Figure 2.3 Programming Model (Privileged-DSP Mode)

2.1.2 Execution Environment when DSP Extension is Disabled

When DSP extension is disabled, the SH7729 is identical to SH3. In this mode (DSP extension is disabled), the instruction set is implemented in fixed-length 16-bit wide instructions executed in a pipeline sequence with single-cycle execution for most instructions. All operations are executed in 32-bit longword units.

Memory can be accessed in 8-bit byte, 16-bit word, or 32-bit longword units, with byte or word data sign-extended into 32-bit longwords. Literals are sign-extended in arithmetic operations (MOV, ADD, and CMP/EQ instructions) and zero-extended in logical operations (TST, AND, OR, and XOR instructions).

The SH7729 features a load-store architecture in which basic operations are executed in registers. Operations requiring memory access are executed in registers following register loading, except for bit-manipulation operations such as logical AND instructions, which are executed directly in memory.

Unconditional branching is implemented as delayed branch operations. Pipeline disruptions due to branching are minimized by the execution of the instruction following the delayed branch instruction prior to branching, for example:

```
BRA      TRGET
ADD      R1, R0    ; ADD is executed prior to branching to TRGET
```

The T bit in the status register (SR) is used to indicate the result of comparison operations, and is read as a TRUE/FALSE condition determining if a conditional branch is taken or not. To improve processing speed, the T bit logic state is modified only by specific operations. An example of how to use the T bit in a sequence of operations:

```
ADD      #1, R0    ; T bit not modified by ADD operation
CMP/EQ   R1, R0    ; T bit set to 1 when R0 = R1
BT       TRGET     ; Branch taken to TRGET when T bit = 1 (R0 = 0)
```

Byte-width literals are inserted directly into the instruction code as immediate data. To maintain the 16-bit fixed-length instruction code, word or longword literals are stored in a table in main memory rather than inserted directly into the instruction code. The memory table is accessed by the MOV instruction using PC-relative addressing with displacement, as follows:

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

As with word and longword literals, absolute addresses must also be stored in a table in main memory. The value of the absolute address is transferred to a register and the operand access is specified by indexed register-indirect addressing, with the absolute address loaded (as with word and longword immediate data) during instruction execution.

In the same way, 16-bit and 32-bit displacements also must be stored in a table in main memory. Exactly like absolute addresses, the displacement value is transferred to a register and the operand access is specified by indexed register-indirect addressing, loading the displacement (as with word and longword immediate data) during instruction execution.

2.1.3 Execution Environment when DSP Extension is Enabled

When DSP = 1, the SH7729 operates in the DSP mode with DSP extension enabled. Additional DSP instructions and hardware resources are available for digital signal processing program. DSP instructions are divided into two groups: 16-bit and 32-bit lengths.

2.2 Registers

The SH7729 has the same registers as in SH3. In addition, the SH7729 also support the same DSP-related registers seen in SH-DSP. The basic software-accessible registers are divided into four distinct groups:

- General-purpose registers
- Control registers
- System registers
- DSP registers

All of these registers are 32-bit width, with the exception of two DSP registers which are 32-bit width with 8-bit guard bits. The general-purpose registers are accessible from the user mode, with R0–R7 banked to provide each processor mode access to a separate set of the R0–R7 registers (i.e. R0–R7_BANK0, and R0–R7_BANK1). In the privileged mode, the register bank (RB) bit in the status register (SR) defines which set of banked registers (R0–R7_BANK0 or R0–R7_BANK1) are accessed as general-purpose registers, and which are accessed only by the LDC/STC instructions.

The control registers (except for GBR) can only be accessed by the LDC/STC instructions in the privileged mode. Control registers are:

- SR: Status register
- SSR: Saved status register
- SPC: Saved program counter
- GBR: Global base register
- VBR: Vector base register
- RS: Repeat start register (DSP mode only)
- RE: Repeat end register (DSP mode only)
- MOD: Mode register (DSP mode only)

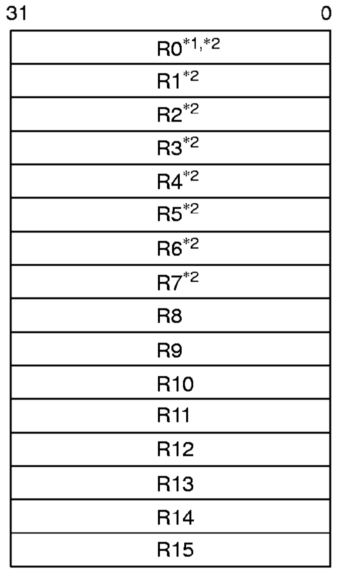
The system registers are accessed by the LDS/STS instructions (the PC is software-accessible, but is included here because its contents are saved in, and restored from, SPC in exception processing). The system registers are:

- MACH: Multiply and accumulate high register
- MACL: Multiply and accumulate low register
- PR: Procedure register
- PC: Program counter

This section explains the usage of these registers in different modes.

2.2.1 General Purpose Registers

Figure 2.4 shows the general purpose registers, which are identical to SH3's, when DSP extension is disabled.



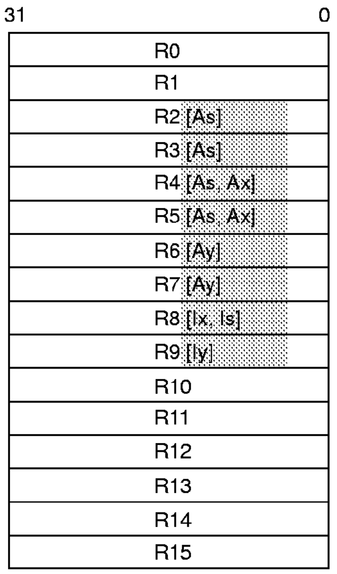
General-Purpose Registers (when not in DSP mode)

Note: 1. R0 functions as an index register in the indexed register-indirect addressing mode and indexed GBR-indirect addressing mode. In some instructions, only R0 can be used as the source register or destination register.

2. R0–R7 are banked registers. In user mode, BANK0 is used. In privileged mode, SR.RB specifies BANK.
SR.RB = 0; BANK0 is used
SR.RB = 1; BANK1 is used

Figure 2.4 General Purpose Register (Not in DSP Mode)

On the other hand, R2–R9 registers are also used for the DSP data address calculations, see figure 2.5, when DSP extension is enabled. Another symbol that represents the purpose of the registers in DSP type instruction is [ ].



General-Purpose Registers (when DSP mode is activated)

X and Y (dual) data transfer operation

R4, 5 [Ax]:	Address reg set for X data memory.
R8 [x]:	Index register for addr reg set Ax.
R6, 7 [Ay]:	Address reg set for Y data memory.
R9 [Iy]:	Index register for addr reg set Ay.

Single Data Transfer Operation

R2–5 [As]:	Address reg set for memory.
R8 [Is]:	Index register for addr reg set As.

Figure 2.5 General Purpose Register (DSP Mode)

DSP type instructions can access X and Y data memory simultaneously. To specify addresses for X and Y data memory, two address pointer sets are prepared. These are:

R8[Ix], R4,5[Ax] for X memory access, and R9[Iy], R6,7[Ay] for Y memory access.

R8[Ix] and R9[Iy] are index registers. R4–7[Ax,Ay] are address registers. In single data transfer operation, R8[Is] and R2–R5[As] can be used. The names (symbol) R2–R9 are used in the Assembler, but users can use other register names that represent the purpose of the register in the DSP instruction explicitly. In the assembly program, user can use an alias for the register. We recommend the following aliases for the registers.

```
Ax0:   .REG   (R4)
Ax1:   .REG   (R5)
Ix:    .REG   (R8)
Ay0:   .REG   (R6)
Ay1:   .REG   (R7)
Iy:    .REG   (R9)
As0:   .REG   (R4)   ; This is optional. If you need another alias for single data transfer.
As1:   .REG   (R5)   ; This is optional. If you need another alias for single data transfer.
As2:   .REG   (R2)
```

```
As3:    .REG    (R3)
Is:     .REG    (R8)    ; This is optional. If you need another alias for single data transfer.
```

2.2.2 Control Registers

SH3-DSP has 8 control registers: SR, SSR, SPC, GBR, VBR, RS, RE, and MOD (figure 2.6). SSR, SPC, GBR and VBR are the same as the SH3 registers. The DSP mode is activated only when SR.DSP = 1. Repeat start register RS, repeat end register RE, and repeat counter RC (12-bit part of the SR) and repeat control bits RF0 and RF1 are new registers and control bits which are used for repeat control. Modulo register MOD and modulo control bits DMX and DMY in the SR are also new register and control bits.

In the SR, there are five additional control bits: RF0, RF1, DMX, DMY and DSP. The DMX and DMY are used for modulo addressing control. If the DMX is 1 then the modulo addressing mode is effective for the X memory address pointer, Ax (R4 or R5). If the DMY is 1 then it is effective for the Y memory address pointer, Ay (R6 or R7). However, both X and Y address pointer cannot be operated under the modulo addressing mode even though both DMX and DMY bits are set. The case of DMX = DMY = 1 is reserved for future expansion. When both DMX and DMY are set simultaneously, the hardware will preliminary treat only address pointer as the modulo addressing mode. Modulo addressing is available for X and Y data transfer operation (MOVX and MOVY), but not for single data transfer operation (MOVS).

The RF1 and RF0 hold information of the number of repeat steps and they are set when a SETRC instruction is executed. When RF[1:0] shows 00, the current repeat module consists of one-step instruction. When RF[1:0] = 01, it means two-step instructions. When RF[1:0] = 11, it means three-step instruction. When RF[1:0] = 10, it means the current repeat module consists of four or more instructions.

The SR also has 12-bit repeat counter RC which is used for efficient loop control. Repeat start register (RS) and repeat end register (RE) are also introduced for the loop control. They keep the start and end addresses of a loop (the contents of the registers, RS and RE are slightly different from the actual loop start and end address). Modulo register,MOD is introduced to realize modulo addressing for circular data buffering. MOD keeps the modulo start address (MS) and the modulo end address (ME).

In order to access RS, RE and MOD, load/store (control register) instructions for them are introduced. An example for RS is as follows:

```
LDC Rm,RS;          Rm -> RS
LDC.L @Rm+,RS;      (Rm) -> RS, Rm+4 -> Rm
STC RS,Rn;          RS -> Rn
STC.L RS,@-Rn;      Rn-4 -> Rn, RS -> (Rn)
```

Address set instructions for the RS and RE are also prepared.

```
LDRS @(disp,PC);   disp X 2 + PC -> RS
LDRE @(disp,PC);   disp X 2 + PC -> RE
```

31 28	27 16	15 13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0 MD RB BL	RC	0-0	DSP	DM Y	DM X	M	Q	13	12	11	10	RF 1	RF 0	S	T	SR (Status register)

MD bit: Processor operation mode
MD = 1: privileged mode
MD = 0: user mode

RB bit: Register bank bit; used to define the general purpose registers in privileged mode.
RB = 1: R0_BANK1 to R7_BANK1 are used as general purpose reg.
R0_BANK0 to R7_BANK0 accessed by LDC/STC instructions.
RB = 0: R0_BANK0 to R7_BANK0 are used as general purpose reg.
R0_BANK1 to R7_BANK1 accessed by LDC/STC instructions.

BL bit: Block bit; used to mask exception in privileged mode.
BL = 1: interrupts are masked (not accepted)
BL = 0: interrupts are accepted

RC [11:0]: 12-bit repeat counter

DSP bit: DSP operation mode
DSP = 1: DSP instructions (LDS Rm, DSR/A0/X0/X1/Y0/Y1, LDS.L @Rm+, DSR/A0/X0/X1/Y0/Y1. STS DSR/A0/X0/X1/Y0/Y1, Rn, STS.L DSR/A0/X0/X1/Y0/Y1, @–Rn, LDC Rm, RS/RE/MOD, LDC.L @Rm+, RS/RE/MOD, STC RS/RE/MOD,Rn, STC.L RS/RE/MOD, @–Rn, LDRS, LDRE, SETRC, MOVS, MOVX, MOVY, Pxxx) are enabled.
DSP = 0: All DSP instructions are treated as illegal instructions, only SH3 instructions are supported.

DMY bit: Modulo addressing enable for Y side

DMX bit: Modulo addressing enable for X side

Q, M bit: Used by DIVOU/S and DIV1 instructions.

I [3:0]: 4-bit field indicating the interrupt request mask level.

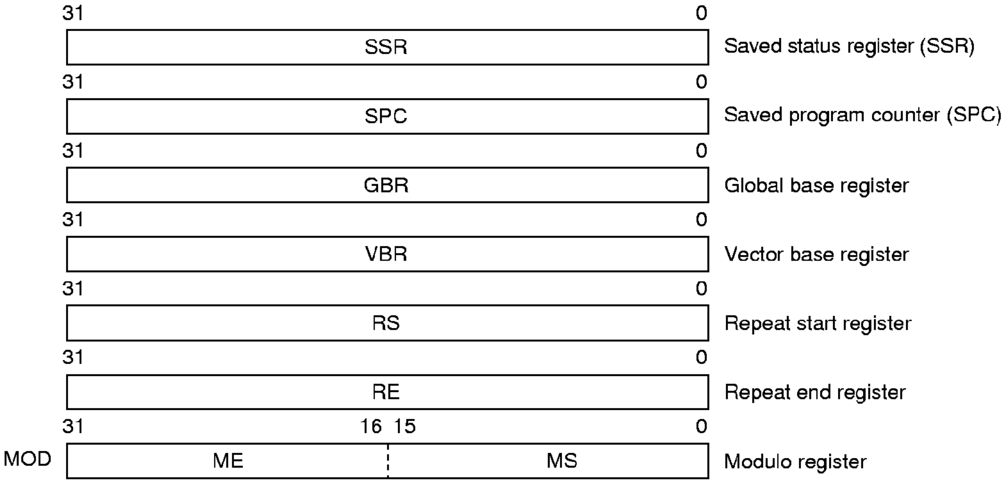
RF [1:0]: Used for repeat control

S bit: Used by the MAC instructions and DSP data.

T bit: The MOVT, CMP/cond. TAS.TST, BT, BF, SETT, CLRT and DT instructions use the T bit to indicate true (logic one) or false (logic zero). The ADDV/C, SUBV/C, DIVOU/S, DIV1, NEGC, SHAR/L, SHLR/L, ROTR/L and RO(TCR/L) instructions also use the T bit to indicate a carry, borrow, overflow or underflow.

0 bits: Always read as 0, and should always be written as 0.

Figure 2.6 Control Registers



ME: Modulo end address, MS: Modulo start address

Saved status register (SSR)
Stores current SR value at time of exception to indicate processor status in the return to instruction stream from exception handler.

Saved program counter (SPC)
Stores current PC value at time of exception to indicate the return address at the compeltion of exception processing.

Global base register (GBR)
Stores the base address of GBR-indirect addressing mode. The GBR-indirect addressing mode is used to transfer data to the register areas of the resident peripheral modules, and for logic operations.

Vector base register (VBR)
Stores the base address of the exception processing vector area.

Repeat start register (RS)
Used in DSP mode only. Indicates the starting address of repeat loop. The contents of RE register is loaded into PC when Repeat Loop proceed to the next iteration(when Repaeat Counter !=0).

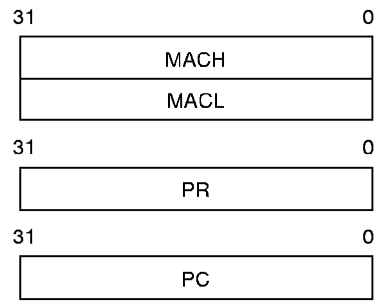
Repeat end register (RE)
Used in DSP mode only. Indicates the address of the repeat loop end.

Modulo register
Used in DSP mode only.
MD[31:16]: ME: Modulo end address, MD[15:0]: Modulo start address.
When doing X/Y operand's address generation, CPU compares the address to ME, if it is equal, CPU loads MS to either X or Y operand address register (depending on DMX and DMY bits in SR register).

Figure 2.7 Control Registers (continued)

2.2.3 System Registers

Like SH3 and SH-DSP, SH3-DSP has four system registers, MACL, MACH, PR and PC (figure 2.8).



Multiply and accumulate high and low registers (MACH/L)
Store the results of multiplicationand accumulation operations.

Procedure register (PR)
Stores the sbroutine procedure return address.

Program counter (PC)
Indicates the starting address of the current instruction.

Figure 2.8 System Registers

DSR, A0, X0, X1, Y0 and Y1 registers described in the following section are also treated as system registers from the CPU module in order to improve the performance. So, data move instructions between general registers and system registers are supported for them.

2.2.4 DSP Registers

The SH7729 has eight data registers and one control register (figure 2.9). The data registers are 32-bit width with the exception of registers A0 and A1. Registers A0 and A1 have guard bits.

Three types of operations access the DSP data registers. First one is the DSP data. When a DSP fixed-point data operation uses A0 or A1 for source register, it uses the guard bits (bits 39–32). When it uses A0 or A1 for destination register, bits 39–32 in the guard bit is valid. When a DSP fixed-point data operation uses the DSP registers other than A0 and A1 for source register, it sign-extends the source value to bits 39–32. When it uses them for destination register, the bits 39–32 of the result is discard.

Second one is X and Y data transfer operation,"MOVX.W MOVY.W". This operation accesses the X and Y memories through 16-bit X and Y data buses (figure 2.10). Registers to be loaded or stored by this operation are always upper 16 bits (bits 31–16). X0 and X1 can be destination of the X memory load and Y0 and Y1 can be destination of Y memory load, but other register cannot be destination register of this operation. When data is loaded into upper 16 bits of a register (bits 31–16), the lower half of the register (bits 15–0) is automatically cleared. A0 and A1 can be stored to the X or Y memory by this operation, but other registers cannot be stored.

Third one is single-data transfer instruction, “MOVS.W” and “MOVS.L”. This instruction accesses any memory location through LDB (figure 2.11). All DSP registers connect to the LDB and be able to be source and destination register of the data transfer. It has word and longword access modes. In the word mode, registers to be loaded or stored by this instruction are upper 16 bits (bits 31–16) for the DSP registers except A0G and A1G. When data is loaded into a register other than A0G and A1G in the word mode, lower half of the register is cleared. When it is A0 or A1, the data is sign-extended to bits 39–32 and lower half of it is cleared. When A0G or A1G is a destination register in the word mode, data is loaded into 8-bit register, but A0 or A1 is not cleared. In the longword mode, when a destination register is A0 or A1, it is sign-extended to bits 39–32.

Tables 2.1 and 2.2 show the data type of registers used in the DSP instructions. Some instructions cannot use some registers shown in the tables because of instruction code limitation. For example, PMULS can use A1 for source registers, but cannot use A0. These tables ignore details of the register selectability.

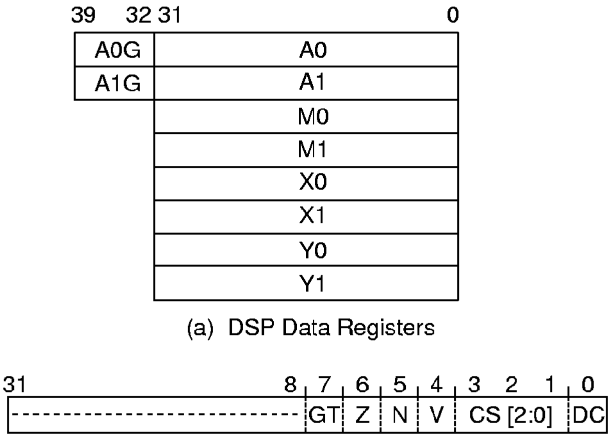
Table 2.1 Destination Register of DSP Instructions

			Guard Bits	Register Bits	
Registers	**Instructions**		**39 32**	**31 16**	**15 0**
A0/1	DSP	Fixed-point, PSHA, PMULS	Sign-extended	40-bit result	
	Data operation	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	Data	No update	
		MOVS.L	Data	No update	
X0/1 Y0/1 M0/1	DSP	Fixed-point, PSHA, PMULS		32-bit result	
	Data operation	Integer, logical, PDMSB, PSHL		16-bit result	Cleared
	Data transfer	MOVX/Y.W, MOVS.W		16-bit result	Cleared
		MOVS.L		32-bit data	

Table 2.2 Source Register of DSP Operations

			Guard Bits	Register Bits	
Registers	Instructions		39 32	31 16	15 0
A0/1	DSP	Fixed-point, PDMSB, PSHA		40-bit data	
	Data operation	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	Data	▨	▨
		MOVS.L	Data	▨	▨
X0/1 Y0/1 M0/1	DSP	Fixed-point, PDMSB, PSHA	Sign*	32-bit data	
	Data operation	Integer	Sign*	16-bit data	▨
		Logical, PSHL, PMULS	▨	16-bit data	▨
	Data transfer	MOVS.W	▨	16-bit data	▨
		MOVS.L	▨	32-bit data	

Note: *Sign-extend the data and feed to the ALU



(a) DSP Data Registers

(b) DSP Status Register (DSR)

Reset status
DSR: All zeros
Others: Undefined

Figure 2.9 DSP Registers

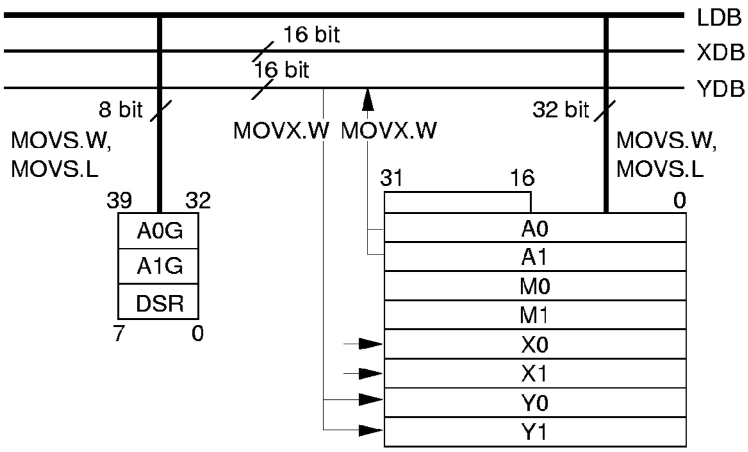


Figure 2.10 Connections of DSP Registers and Buses

The DSP engine has one control register DSP Status Register (DSR). The DSR has conditions of the DSP data operation result (zero, negative, and so on) and a DC bit which is similar to the T bit in the CPU. The DC bit indicates the one of the conditional flags. A DSP data processing instruction controls its execution based on the DC bit. This control affects only the operations in the DSP unit; it controls the update of DSP registers only. It cannot control operations in CPU, such as address register updating and load/store operations. The control bit CS[2:0] specifies the condition to be reflect to the DC bit (table 2.3).

The unconditional DSP type data operations, except PMULS, PWAD, PWSB, MOVX, MOVY and MOVS, update the conditional flags and DC bit, but no CPU instructions, including MAC instructions, update the DC bit. The conditional DSP type instructions do NOT update the DSR either.

Table 2.3 Mode of the DC Bit

CS [2:0]	Mode
000	Carry or borrow
001	Negative
010	Zero
011	Overflow
100	Signed greater than
101	Signed greater than or equal

DSR is assigned as a system register and load/store instructions are prepared as follows:

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

When DSR is read by the STS instructions, the upper bits (bit31~bit8) are all 0.

2.3 Data Format

2.3.1 Data Format in Registers (Non-DSP Type)

Register operands are always longwords (32 bits) (figure 2.11). When the memory operand is only a byte (8 bits) or a word (16 bits), it is sign-extended into a longword when loaded into a register.

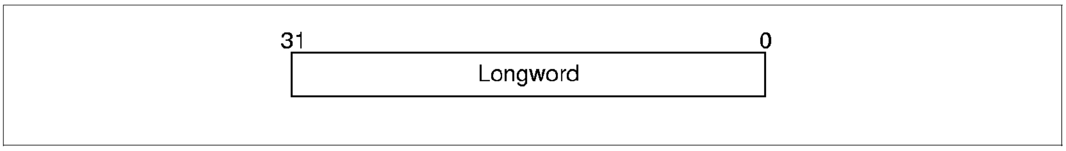


Figure 2.11 Longword Operand

2.3.2 DSP-Type Data Format

The SH7729 has several different data formats that depend on operations. This section explains the data formats for DSP type instructions.

Figure 2.12 shows three DSP type data formats according to the binary point position. 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 operations and the DSP registers.

Figure 2.12 also shows one of the CPU-type data formats that has the binary point between bit 0 and bit 1 as a reference.

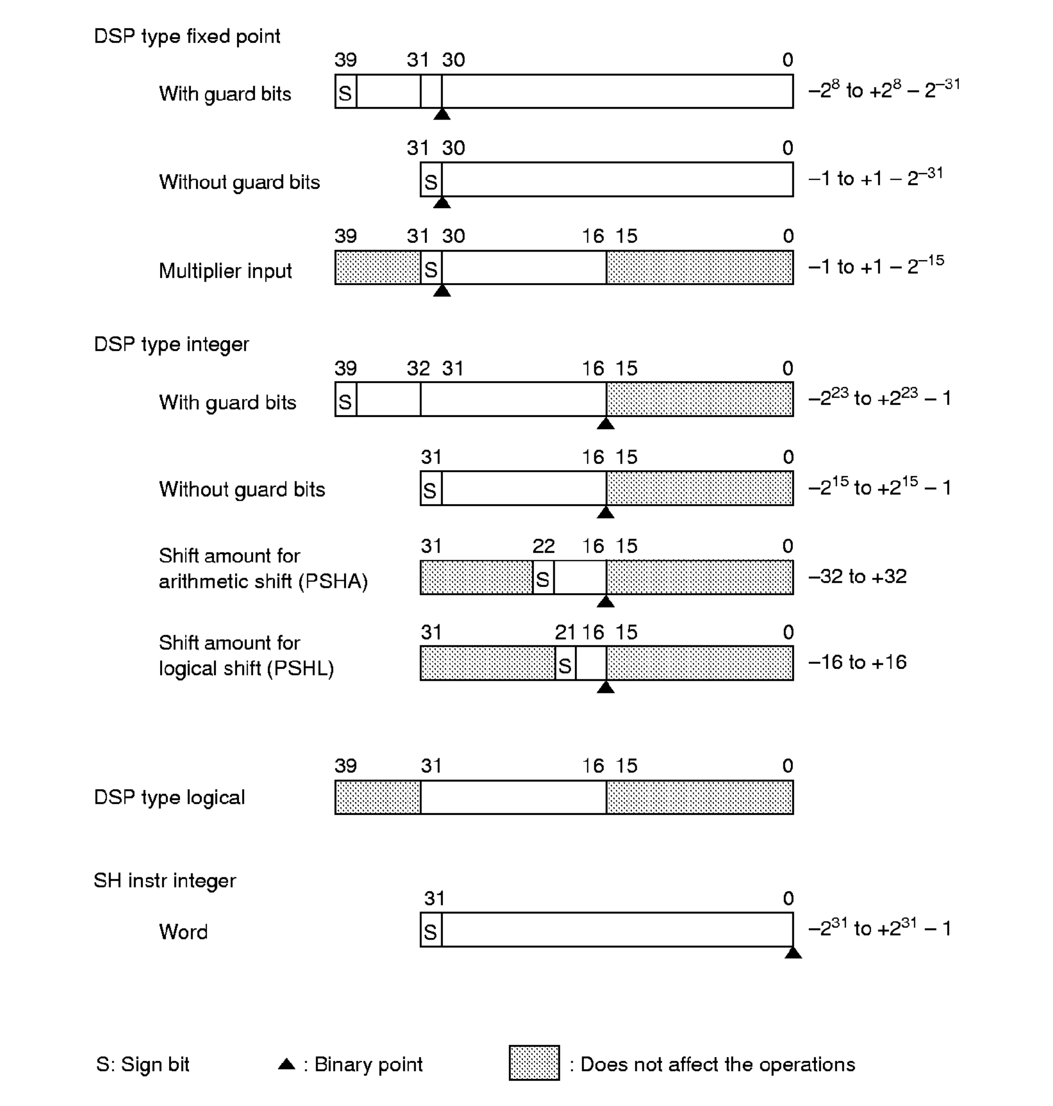


Figure 2.12 Data Format

Shift amount for arithmetic shift (PSHA) instruction has 7 bits filed that could represent –64 to +63, however –32 to +32 is the valid number for the operation. Also the shift amount for logical shift operation has 6-bits field, however –16 to +16 is the valid number for the instruction.

2.3.3 Data Format in Memory

Memory data formats are classified into bytes, words, and longwords. Byte data can be accessed from any address, but an address error will occur if the word data starting from an address other than 2n or longword data starting from an address other than 4n is accessed. In such cases, the data accessed cannot be guaranteed (figure 2.13).

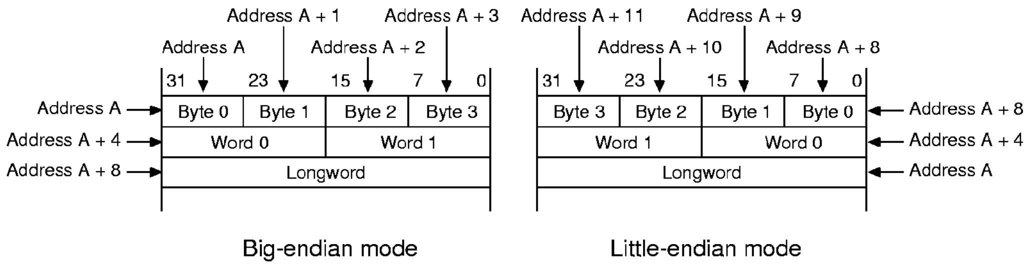


Figure 2.13 Byte, Word, and Longword Alignment

As the data format, either big endian or little endian byte order can be selected, according to the MD5 pin at reset. When MD5 is low at reset, the processor operates in big endian. When MD5 is high at reset, the processor operates in little endian.

Section 3 DSP Operation

3.1 Data Operations of DSP Unit

3.1.1 ALU Fixed-Point Operations

Figure 3.1 shows the ALU arithmetic operation flow. Table 3.1 shows the variation of this type of operation and table 3.2 shows the flexibility of each operand.

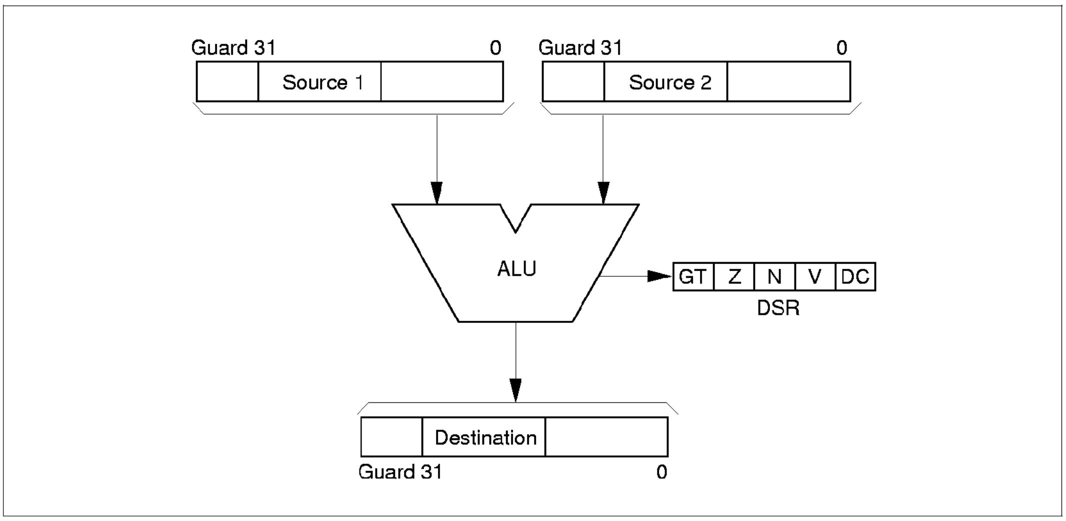


Figure 3.1 ALU Fixed-Point Arithmetic Operation Flow

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 are specified as the source operand. When a register not providing the guard-bit parts 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 final stage of the pipeline sequence, named DSP stage, as shown in figure 3.2.

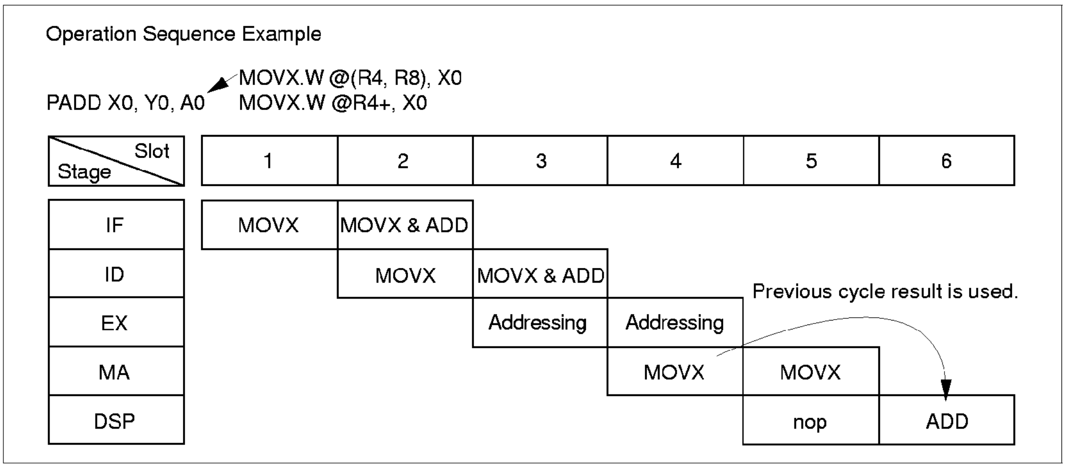
Table 3.1 Variation of ALU Fixed-Point Operation

Mnemonic	Function	Source 1	Source 2	Destination
PADD	Addition	Sx	Sy	Dz (Du)
PSUB	Subtraction	Sx	Sy	Dz (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 3.2 Operand Flexibility

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 3.2, loaded data from the memory at the MA stage, which is programmed at the same line as the ALU operation, is not used as a source operand for this operation, even though the destination operand of memory read operation is identical to the source operand of the ALU operation. In this case, previous operation results are used as the source operands for the ALU operation and then, updated as the destination operand of the data load operation.


Operation Sequence Example

```
                     MOVX.W @(R4, R8), X0
PADD X0, Y0, A0      MOVX.W @R4+, X0
```

Stage \ Slot	1	2	3	4	5	6
IF	MOVX	MOVX & ADD				
ID		MOVX	MOVX & ADD			
EX			Addressing	Addressing		
MA				MOVX	MOVX	
DSP					nop	ADD

Previous cycle result is used.

Figure 3.2 Operation Sequence Example

Every time an ALU arithmetic operation is executed, the DC, N, Z, V and GT bits in the DSR register 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 CS0–2 (condition selection) bits in the DSR register. The DC bit result is as follows:

Carry or Borrow Mode: CS [2:0] = 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 3.3. This mode is the default condition.

Negative Value Mode: CS [2:0] = 001: The DC flag indicates the same state 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 3.4.

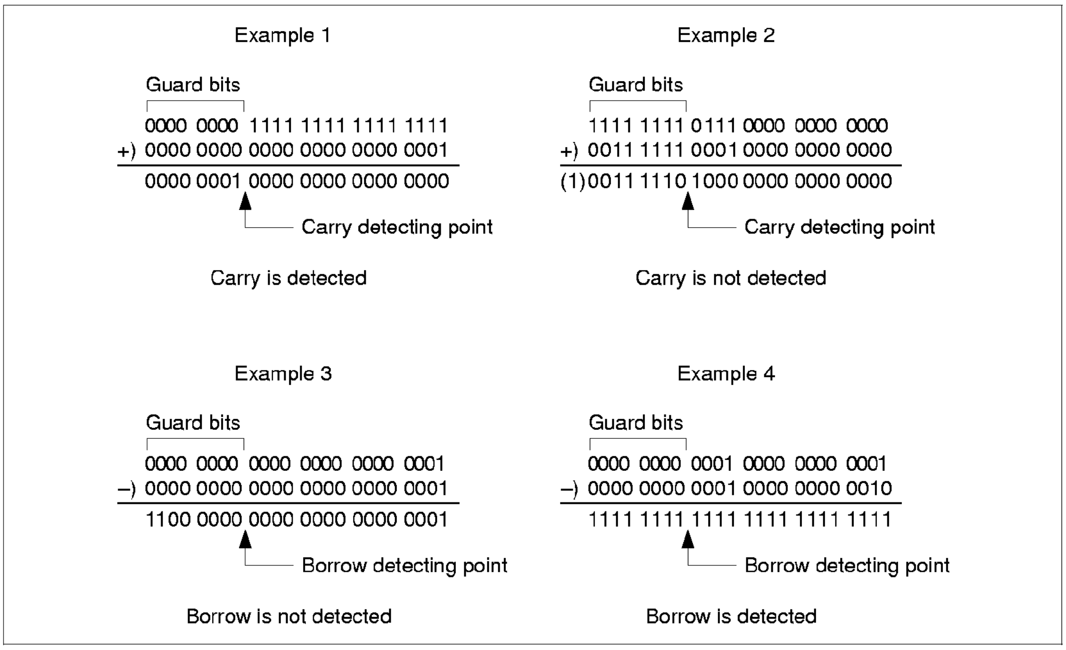


Figure 3.3 DC Bit Generation Examples in Carry or Borrow Mode

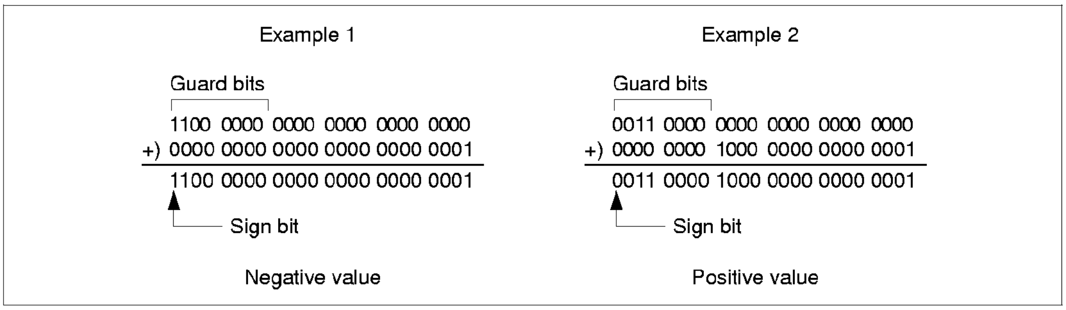


Figure 3.4 DC Bit Generation Examples in Negative Value Mode

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

Overflow Mode: CS [2:0] = 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, the DC bit always indicates the result of guard bits not provided case. So, the DC bit is always set if the guard-bit parts are used for large number representation. Some flag detection examples are shown in figure 3.5.

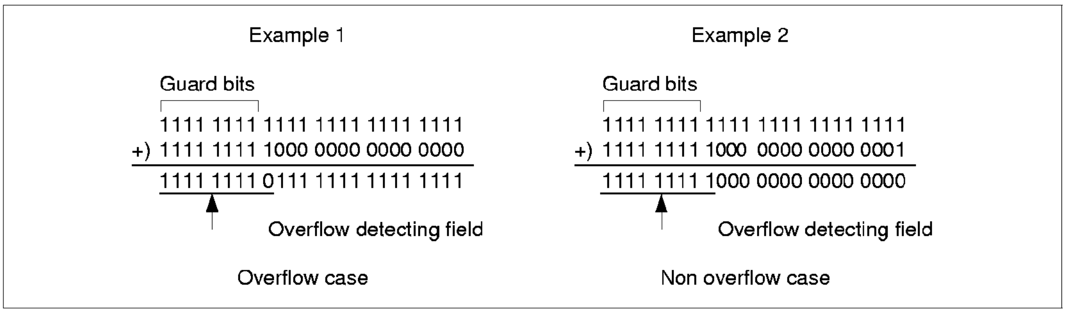


Figure 3.5 DC Bit Generation Examples in Overflow Mode

Signed Greater Than Mode: CS [2:0] = 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. So, a 'PCMP' operation should be executed in advance when a 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 usually 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 result of the PCMP/GT operation of the SH core instruction.

Signed Greater Than or Equal Mode: CS [2:0] = 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 a compare operation. So, a 'PCMP' operation should be executed in advance when a 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 T bit result of a PCMP/GE operation of the SH core instruction.

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

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

Overflow Protection: The S bit in SR register is effective for any ALU fixed-point arithmetic operations in the DSP unit. See section 3.1.8, Overflow Protection, for details.

3.1.2 ALU Integer Operations

Figure 3.6 shows the ALU integer arithmetic operation flow. Table 3.3 shows the variation of this type of operation. The flexibility of each operand is the same as ALU fixed-point operations as shown in table 3.2.

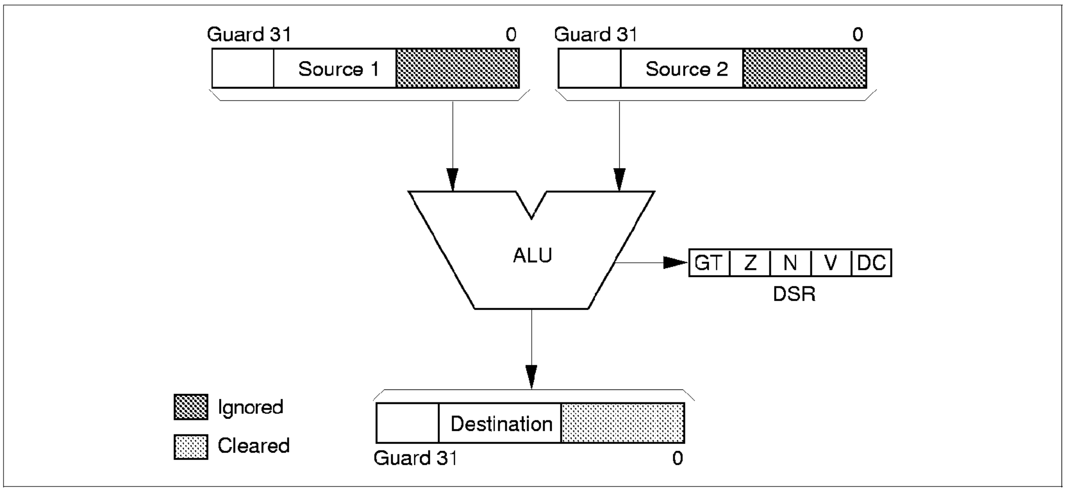


Figure 3.6 ALU Integer Arithmetic Operation Flow

Table 3.3 Variation of ALU Integer Operation

Mnemonic	Function	Source 1	Source 2	Destination
PINC	Increment by	Sx	+1	Dz
		+1	Sy	Dz
PDEC	Decrement by	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 as a destination operand, the lower 32 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 3.3, 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. So it is reasonable for increment and decrement operations to execute using the upper word in the DSP unit. When a register providing guard bits is specified as an operand, the guard bits are also activated. These operations are executed in the final stage of the pipeline sequence, named the DSP stage, as shown in figure 3.2, as well as fixed-point operations.

Every time an ALU arithmetic operation is executed, the DC, N, Z, V and GT bits in the DSR register are basically updated in accordance with the operation result. This is the same as fixed-point operations but lower word of each source and destination operand is not used in order to generate them. See section 3.1.1, ALU Fixed-Point 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 3.1.1, ALU Fixed-Point Operations, for details.

Overflow Protection: The S bit in the SR register is effective for any ALU integer arithmetic operations in DSP unit. See section 3.1.8, Overflow Protection, for details.

3.1.3 ALU Logical Operations

Figure 3.7 shows the ALU logical operation flow. Table 3.4 shows the variation of this type of operation. The flexibility of each operand is the same as the ALU fixed-point operations as shown in table 3.2. See note of section 3.1.1, ALU Fixed-Point Operations.

Logical operations are also executed between registers. Each source and destination operand are selected independently from one of the DSP registers. As shown in figure 3.7, this type of operation uses the upper word of each operand only. 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 final stage of the pipeline sequence, named DSP stage, as shown in figure 3.2.

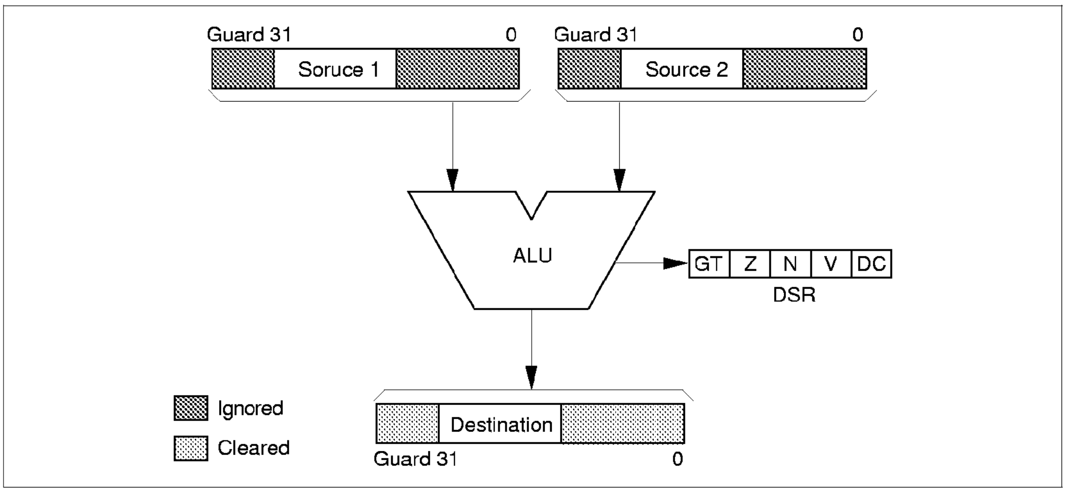


Figure 3.7 ALU Logical Operation Flow

Table 3.4 Variation of ALU Logical Operation

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 DC bit is selected by CS0–2 (condition selection) bits in the DSR register. The DC bit result is:

1. Carry or Borrow Mode: CS [2:0] = 000
   The DC bit is always cleared.
2. Negative Value Mode: CS [2:0] = 001
   The 31st bit of the operation result is loaded into the DC bit.
3. Zero Value Mode: CS [2:0] = 010
   The DC bit is set when the operation result is all zeros, otherwise cleared.
4. Overflow Mode: CS [2:0] = 011
   The DC bit is always cleared.
5. Signed Greater Than Mode: CS [2:0] = 100
   The DC bit is always cleared.
6. Signed Greater Than or Equal Mode: CS [2:0] = 101
   The DC bit is always cleared.

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

3.1.4 Fixed-Point Multiply Operation

Figure 3.8 shows the multiply operation flow. Table 3.5 shows the variation of this type of operation and table 3.6 shows the flexibility of each operand. The multiply operation of the DSP unit is single-word signed single-precision multiplication. If a double-precision multiply operation is needed, it is possible to make use of the SH-2's standard double-word multiply instructions.

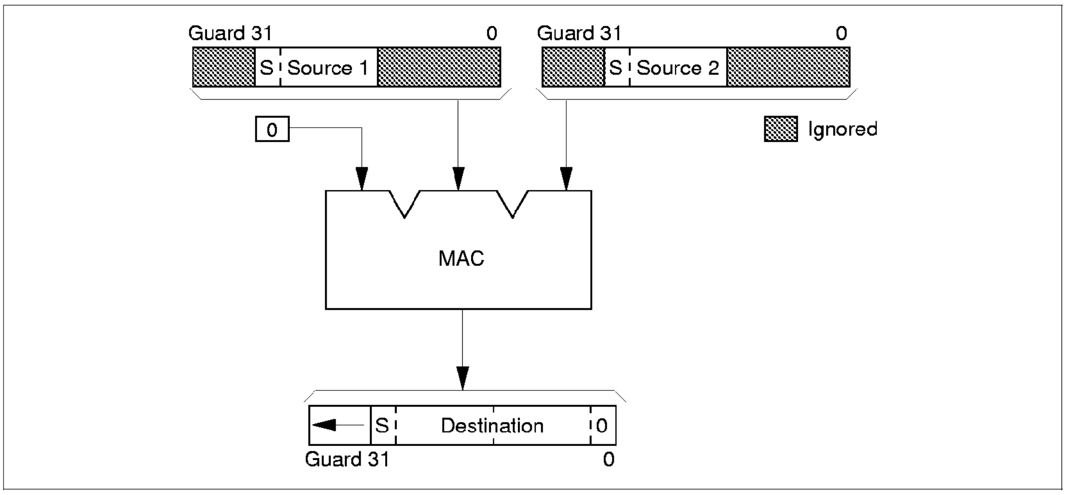


Figure 3.8 Fixed-Point Multiply Operation Flow

Table 3.5 Variation of Fixed-Point Multiply Operation

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

Table 3.6 Operand Flexibility

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 generates 32 bits of the operation result. So when a register providing the guard-bit parts are specified as a destination operand, the guard-bit parts will copy the 32nd bit (MSB) 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 3.8. In the SH's standard multiply operations, the lower words of both source operands are input into a MAC unit. The operation result is also different from the SH's case. The SH's multiply operation result is aligned to the LSB of the destination, but the fixed-point multiply operation result is aligned to the MSB, so that the LSB of the fixed-point multiply operation result is always 0.

This fixed-point multiply operation is executed in one cycle using 32 bits by 8-bit MAC unit. The other SH's multiply and MAC operations are executed as the SH2's are. Multiply operation doesn't affect any condition code bits, DC, N, Z, V and GT, in the DSR register.

Overflow Protection: The S bit in the SR register is effective for this multiply operation in the DSP unit. See section 3.1.8, Overflow Protection, for details.

If the S bit is '0', there is only one case to occur overflow when H'8000*H'8000, (-1.0)*(-1.0), operation is executed as signed by signed fixed-point multiply. The result is H'8000 0000 but it means (+1.0) not (-1.0). If the S bit is '1', it protects the overflow and the result is H'00 7FFF FFFF.

3.1.5 Shift Operations

Shift operations 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. Table 3.7 shows the variations of this type of operation. The flexibility of each operand, except for immediate operands, is the same as the ALU fixed-point operations as shown in table 3.2. See sections 5.1 and 5.4 for more detailed information about each instruction and operand. See note of section 3.1.1, ALU Fixed-Point Operations.

Table 3.7 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 #Imm, Dz	Arithmetic shift w/imm.	Dz	Imm1	Dz
PSHL #Imm, Dz	Logical shift w/imm.	Dz	Imm2	Dz

–32 <= Imm1 <= +32, –16 <= Imm2 <= +16

Arithmetic Shift: Figure 3.9 shows the arithmetic shift operation flow.

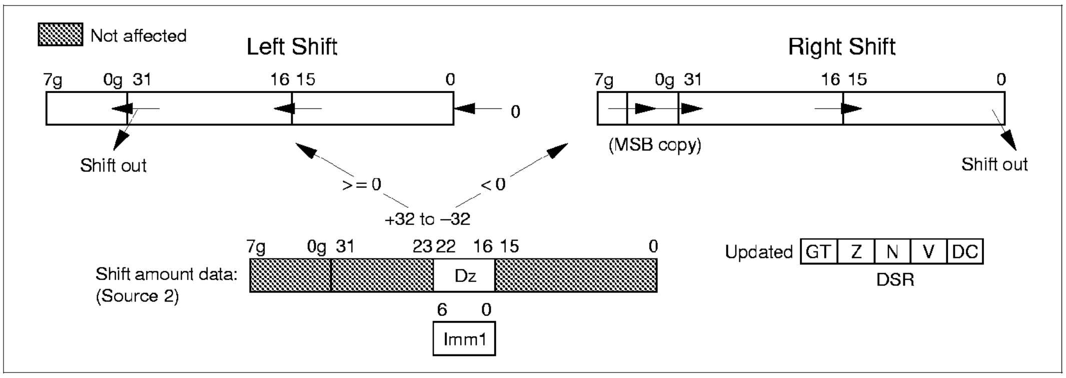


Figure 3.9 Shift Operation Flow

Note: The BPU arithmetic shift operations are basically 40-bit operation, the 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 as a destination operand, the lower 32 bits of the operation result are input into the destination register.

In this arithmetic shift operation, the full size 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 register or immediate operand. Available shift range is from −32 to +32. Here, negative value means the right shift, positive value means the left shift. It's possible for any source 2 operand to specify from –64 to +63 but the result is unknown if invalid shift value is specified. In case of the shift with immediate operand instruction, the source 1 operand must be the same register as the destination's. This operation is executed in the final stage of the pipeline sequence, named DSP stage, as shown in figure 3.2 as well as in fixed-point operations.

Every time an arithmetic shift 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 DC bit is selected by CS0–2 (condition selection) bits in the DSR register. The DC bit result is:

1. Carry or Borrow Mode: CS [2:0] = 000

   The DC bit indicates the last shifted out data as the operation result.

2. Negative Value Mode: CS [2:0] = 001

   Same definition as ALU fixed-point arithmetic operations.

3. Zero Value Mode: CS [2:0] = 010
   Same definition as ALU fixed-point arithmetic operations.
4. Overflow Mode: CS [2:0] = 011
   Same definition as ALU fixed-point arithmetic operations.
5. Signed Greater Than Mode: CS [2:0] = 100
   The DC bit is always cleared.
6. Signed Greater Than or Equal Mode: CS [2:0] = 101
   The DC bit is always cleared.

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

Overflow Protection: The S bit in the SR register is also effective for arithmetic shift operation in the DSP unit. See section 3.1.8, Overflow Protection, for details.

Logical Shift: Figure 3.10 shows the logical shift operation flow.

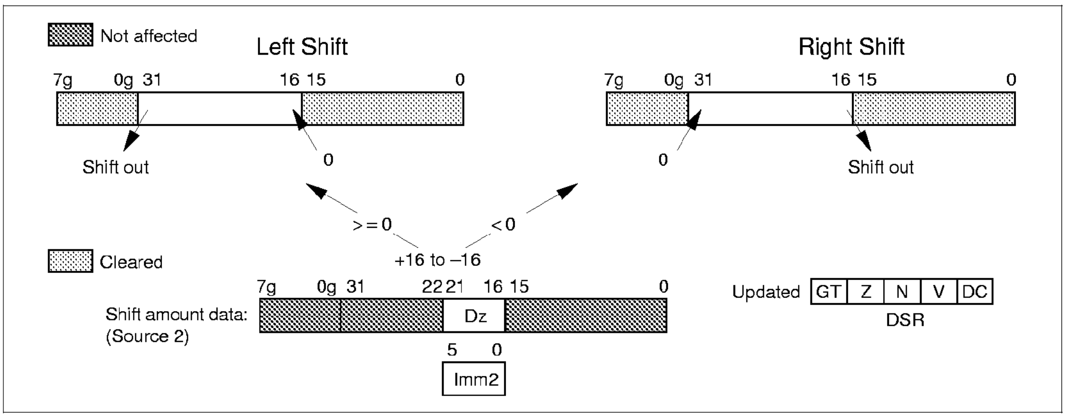


Figure 3.10 Logical Shift Operation Flow

As shown in figure 3.10, the logical shift operation uses the upper word of the source 1 and the destination operands. 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 register or immediate operand. Available shift range is from –16 to +16. Here, negative value means the right shift, positive value means the left shift. It's possible for any source 2 operand to specify from –32 to +31 but the result is unknown if invalid shift value is specified. In case of the shift with immediate operand instruction, the source 1 operand must be the same register as the destination's. These operations are executed in the final stage of the pipeline sequence, named DSP stage, as shown in figure 3.2.

Every time a logical shift operation is executed, the DC, N and Z bits in the DSR register are basically updated with the operation result but V and GT bits are always cleared. 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 unconditional operation, they are always updated with the operation result. The definition of the DC bit is selected by CS0–2 (condition selection) bits in the DSR register. The DC bit result is:

1. Carry or Borrow Mode: CS [2:0] = 000
   The DC bit indicates the last shifted out data as the operation result.
2. Negative Value Mode: CS [2:0] = 001
   Samc dcfinition as ALU logical opcrations.
3. Zero Value Mode: CS [2:0] = 010
   Same definition as ALU logical operations.
4. Overflow Mode: CS [2:0] = 011
   The DC bit is always cleared.
5. Signed Greater Than Mode: CS [2:0] = 100
   The DC bit is always cleared.
6. Signed Greater Than or Equal Mode: CS [2:0] = 101
   The DC bit is always cleared.

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

3.1.6 Most Significant Bit Detection Operation

The ‘PDMSB’, most significant bit detection operation, is used to calculate the shift amount for normalization. Figure 3.12 shows the ‘PDMSB’ operation flow and table 3.8 shows the operation definition. Table 3.9 shows the possible variations of this type of operation. The flexibility of each operand is the same as for ALU fixed-point operations, as shown in table 3.2. See note of section 3.1.1, ALU Fixed-Point Operations.

Note: The result of the priority encode operation is basically 24 bits as well as ALU integer operation, the upper 16-bits of the base precision and 8 bits of the guard-bit parts. When a register not providing the guard-bit parts as a destination operand, the lower 16 bits of the operation result are input into the destination register.

As shown in figure 3.11, the ‘PDMSB’ operation uses full-size data 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 3.1.5 (Arithmetic Shift). These operations are executed in the final stage of the pipeline sequence, named DSP stage, as shown in figure 3.11.

Every time a ‘PDMSB’ operation is executed, the DC, N, Z, V and GT bits in the DSR register are basically updated 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.

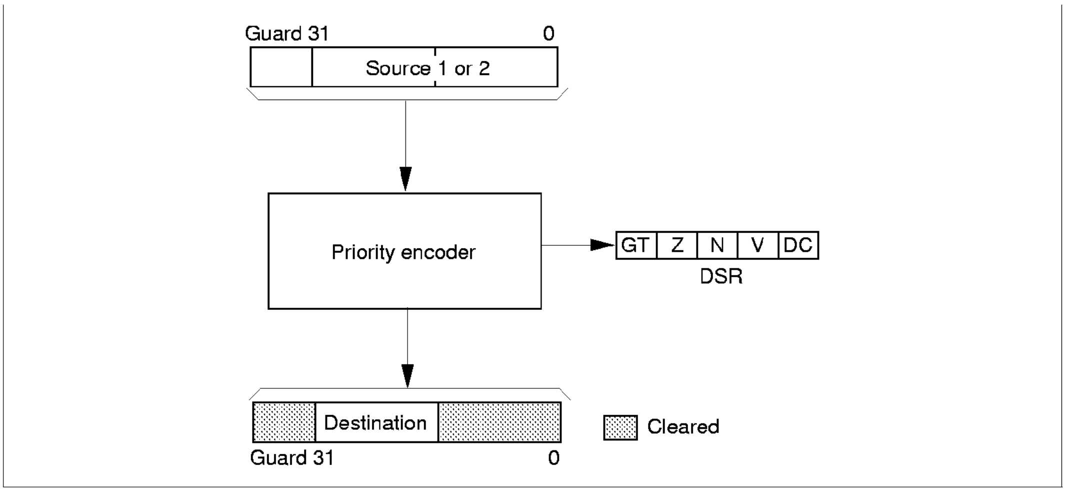


Figure 3.11 ‘PDMSB’ Operation Flow

The definition of the DC bit is selected by CS0–2 (condition selection) bits in the DSR register. The DC bit result is:

1. Carry or Borrow Mode: CS [2:0] = 000
   The DC bit is always cleared.
2. Negative Value Mode: CS [2:0] = 001
   Same definition as ALU integer arithmetic operations.
3. Zero Value Mode: CS [2:0] = 010
   Same definition as ALU integer arithmetic operations.
4. Overflow Mode: CS [2:0] = 011
   The DC bit is always cleared.
5. Signed Greater Than Mode: CS [2:0] = 100
   Same definition as ALU integer arithmetic operations.
6. Signed Greater Than or Equal Mode: CS [2:0] = 101
   Same definition as ALU integer arithmetic operations.

Table 3.8 Operation Definition of 'PDMSB'

Data in SRC															Result for DST							
Guard Bit					Upper Word					Lower Word				Guard Bit	Upper Word							
7g	**6g**	**…**	**1g**	**0g**	**31**	**30**	**29**	**28**	**…**	**3**	**2**	**1**	**0**	**7g–0g**	**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' bit.

Table 3.9 Variation of 'PDMSB' Operations

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

3.1.7 Rounding Operation

The DSP unit provides the rounding 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 3.12 shows the rounding operation flow and figure 3.13 shows the operation definition. Table 3.10 shows the variation of this type of operation. The flexibility of each operand is the same as ALU fixed-point operations as shown in table 3.2. See note of section 3.1.1, ALU Fixed-Point Operations.

As shown in figure 3.12, the rounding operation uses full-size data for both source and destination operands. These operations are executed in the final stage of the pipeline sequence, named DSP stage as shown in figure 3.2.

The rounding operation is always executed unconditionally, so that the DC, N, Z, V and GT bits in the DSR register are always updated in accordance with the operation result. The definition of the DC bit is selected by CS0–2 (condition selection) bits in the DSR register. These condition code bit result is the same as the ALU-fixed point arithmetic operations.

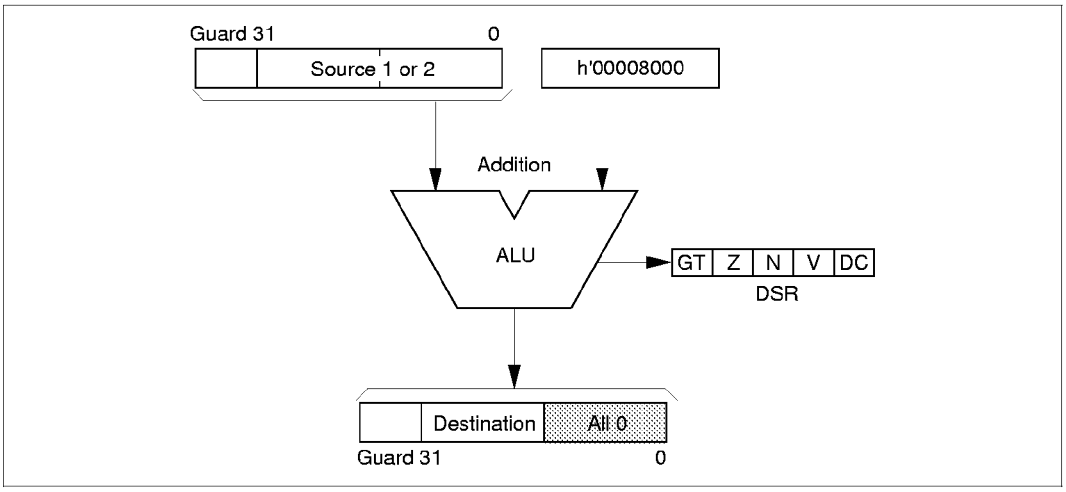


Figure 3.12 Rounding Operation Flow

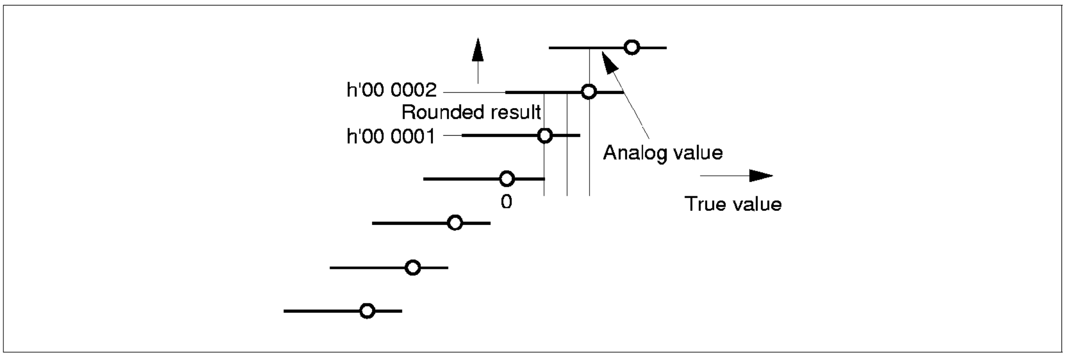


Figure 3.13 Definition of Rounding Operation

Table 3.10Variation of Rounding Operations

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

Overflow Protection: The S bit in the SR register is effective for any rounding operations in the DSP unit. See section 3.1.8, Overflow Protection, for details.

3.1.8 Overflow Protection

The S bit in the SR register is effective for any arithmetic operations executed in the DSP unit, including the conventional SH's multiply and the MAC operations. The S bit in the SR register, in SH's CPU core, is used as the overflow protection enable bit. The arithmetic operation overflows when the operation result exceeds the range of two's complement representation without guard-bit parts. Table 3.11 shows the definition of overflow protection for fixed-point arithmetic operations, including fixed-point signed by signed multiplication described in section 3.1.4, Fixed-Point Multiply Operation. Table 3.12 shows the definition of overflow protection for integer arithmetic operations. When a SH's conventional multiply or MAC operation is executed, the S bit function is completely the same as the current SH2's definition.

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

Table 3.11Definition of Overflow Protection for Fixed-Point Arithmetic

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 3.12Definition of Overflow Protection for Integer Arithmetic

Sign	Overflow Condition	Fixed Value	Hex Representation
Positive	Result > $2^{15} - 1$	$2^{15} - 1$	00 7FFF ****
Negative	Result < -2^{15}	-2^{15}	FF8000 ****

'*' means 'Don't care'.

3.1.9 Data Transfer Operation

The SH7729 can execute a maximum of two data transfer operations between the DSP register and the on-chip data memory in parallel for the DSP unit. This results almost the same performance as other DSPs. The SH7729 provides three types of data transfer instructions for the DSP unit.

1. Parallel operation type (using XDB and YDB)
2. Double data transfer type (using XDB or YDB)
3. Single data transfer type (using LDB)

The type 1 instructions execute both data processing and data transfer operations in parallel. The 32-bit instruction code is used for this type of instruction. Basically, two data transfer operations can be specified by this type of instruction, but they don't always have to be specified. One data transfer is for X memory and another is for Y memory. Both of these data transfer operations cannot be executed for one memory. Load operation for X memory can specify either the X0 or X1 register and for Y memory can specify either the Y0 or Y1 register as a destination operand. Both store operations for X and Y memories can specify either the A0 or A1 register as a source operand. This type of operation treats a word data only. When a word data transfer operation is executed, the upper word of the register operand is activated. In case of word data load, the data is loaded into the upper word of the destination register, and then the lower side of the destination is automatically cleared.

When a conditional operation is specified as a data processing operation, the specified condition also doesn't affect any data transfer operations. Figure 3.14 shows this type of data transfer operation flow.

This type of data transfer operation can access X or Y memory only. Any other memory space cannot be accessed.

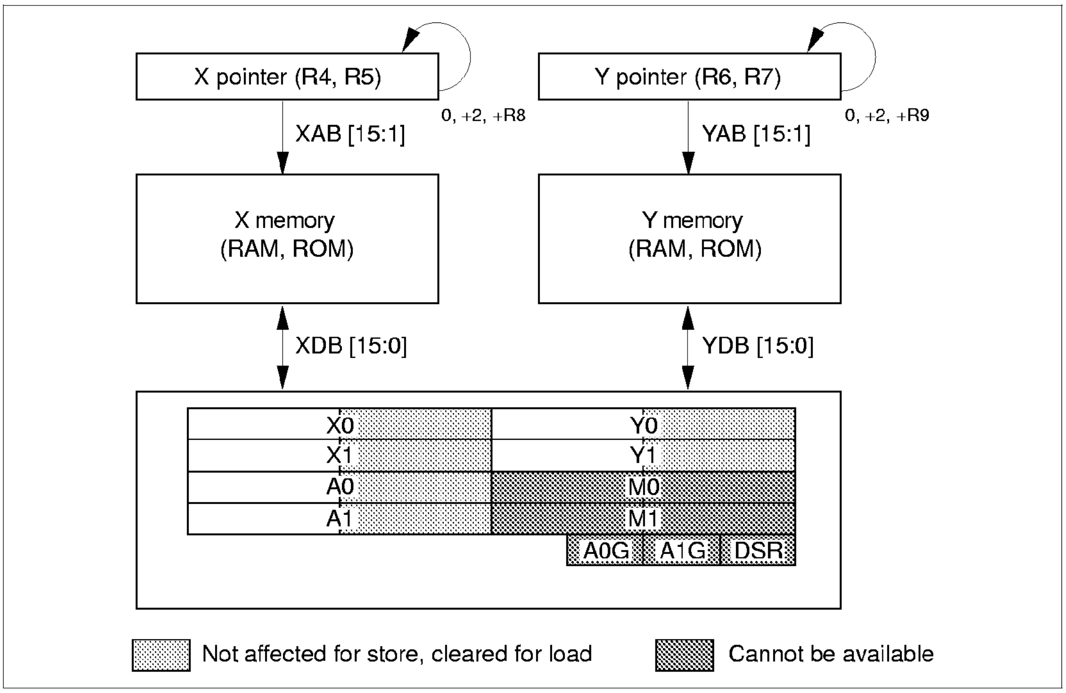


Figure 3.14 Data Transfer Operation Flow

Type 2 instructions execute just two data transfer operations. The 16-bit instruction code is used for this type of instructions. Basically, operation and operand flexibility are the same as in type 1 but conditional operation is not supported. This type of data transfer operation can also access X or Y memory only. Any other memory space cannot be accessed.

Type 3 instructions execute single data transfer operations only. The 16-bit instruction code is used for this type of instructions. X pointers and other two extra pointers are available for this type of operation but Y pointers are not available. This type of operation can access any memory address space, and all registers in the DSP unit, except for DSR, can be specified for both source and destination operands. The guard-bit registers, A0G and A1G, can also be specified as independent registers. In this type of operation, LAB and LDB are used instead of XAB, XDB, YAB and YDB to save hardware, so that the bus conflict might occur on LDB between data transfer and instruction fetch.

This type of operation can treat both single-word data and longword data. When a word-data transfer operation is executed, the upper word of the register operand is activated. In case of word data load, the data is loaded into the upper word of the destination register, the lower side of the destination is automatically cleared and the signed bit is copied into the guard-bit parts, if supported. In case of longword data load, the data is loaded into the upper word and the lower word of the destination register and the signed bit is copied into the guard-bit parts, if supported. In case of the guard register store, the sign is copied on the upper 24 bits of LDB. Figures 3.15 and 3.16 show this type of data transfer operation flows.

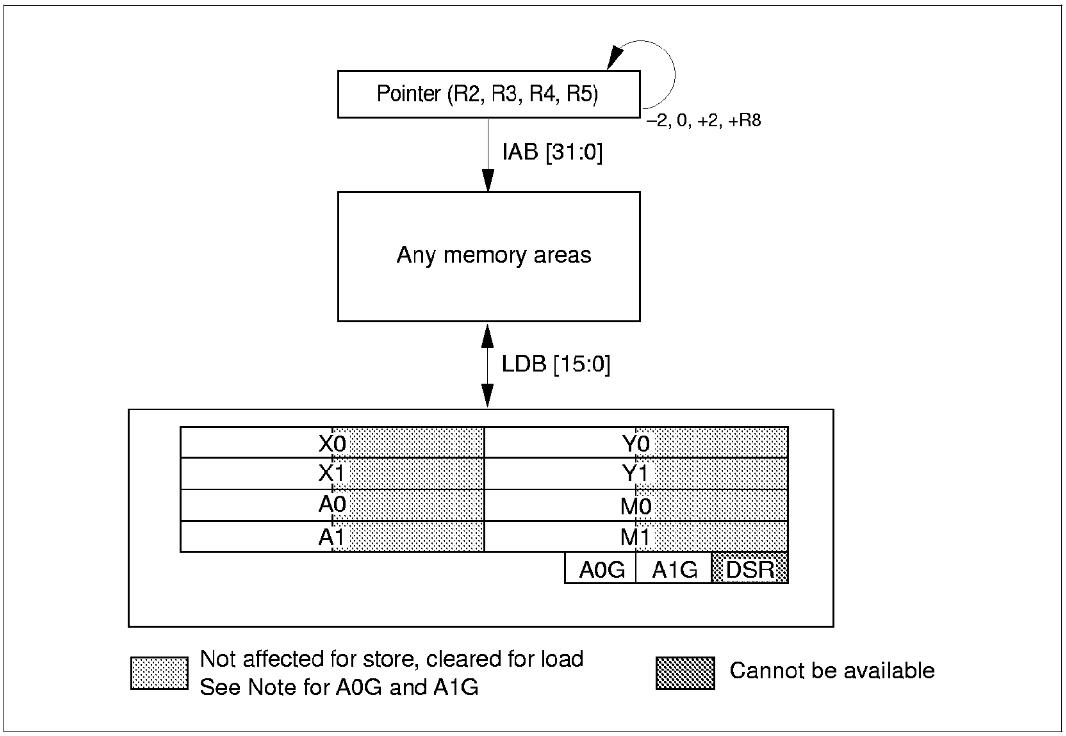


Figure 3.15 Word of Data-Transfer Operation Flow

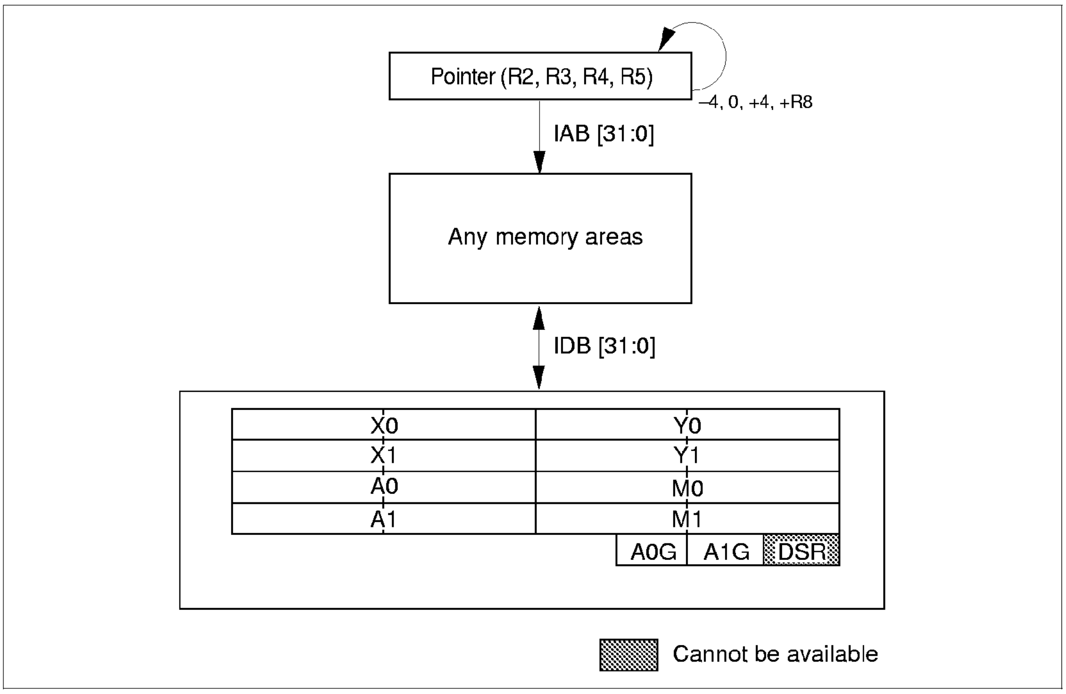


Figure 3.16 Longword of Data-Transfer Operation Flow

All data transfer operations are executed in the MA stage of the pipeline and all data processing operations execute in the DSP stage. When a store instruction is written the following step of the corresponding data processing instruction, one stall cycle is generated in order to execute properly because the data processing operation is not completed when the following data store operation starts the execution. So an instruction should be inserted between the data processing instruction and the data store instruction as shown in figure 3.17 in order to avoid such an overhead cycle.

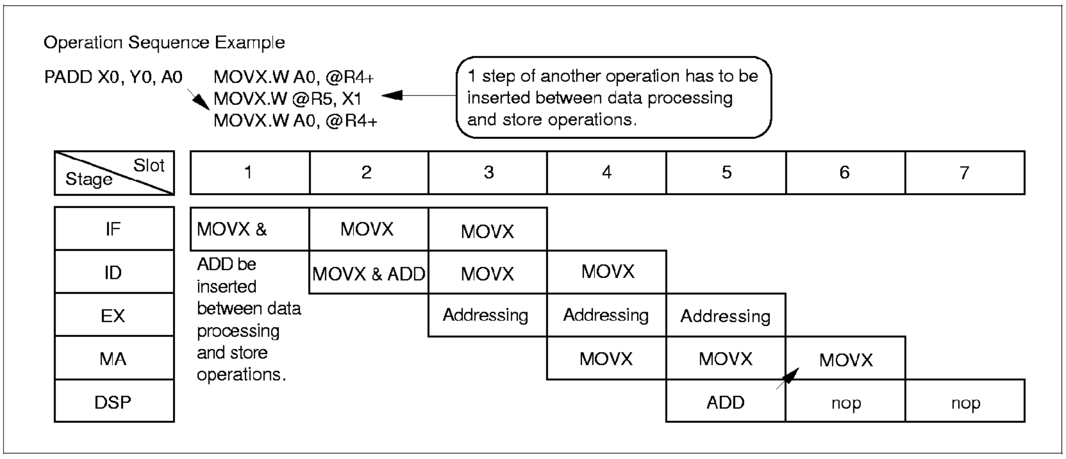
All data transfer operations don't update any condition code bits in the DSR register.

When one of the guard-bit register, A0G and A1G, is specified as the destination operand for a word data load operation, 'MOVS.W', the word data is input into the lower of the register.

A0 register can be loaded using both MOVS.L and LDS(.L)/STS(.L). Both operation is completely the same in the DSP module.

3.1.10 Local Data Move Operation

The DSP unit of the SH7729 provides additional two independent registers, MACL and MACH, in order to support conventional SH2's multiply/MAC operations. They can be also used as temporary storage registers. Local data move instruction between MACH/L and other DSP registers to make use of this benefit. Figure 3.18 shows the flow of seven local data move instructions. Table 3.13 shows the variation of this type of instruction.


Operation Sequence Example
PADD X0, Y0, A0
MOVX.W A0, @R4+
MOVX.W @R5, X1
MOVX.W A0, @R4+
1 step of another operation has to be inserted between data processing and store operations
MOVX & ADD be inserted between data processing and store operations.

Stage \ Slot	1	2	3	4	5	6	7
IF	MOVX &	MOVX	MOVX				
ID		MOVX & ADD	MOVX	MOVX			
EX			Addressing	Addressing	Addressing		
MA				MOVX	MOVX	MOVX	
DSP					ADD	nop	nop

Figure 3.17 Instruction Sequence Example Between Data Processing and Store

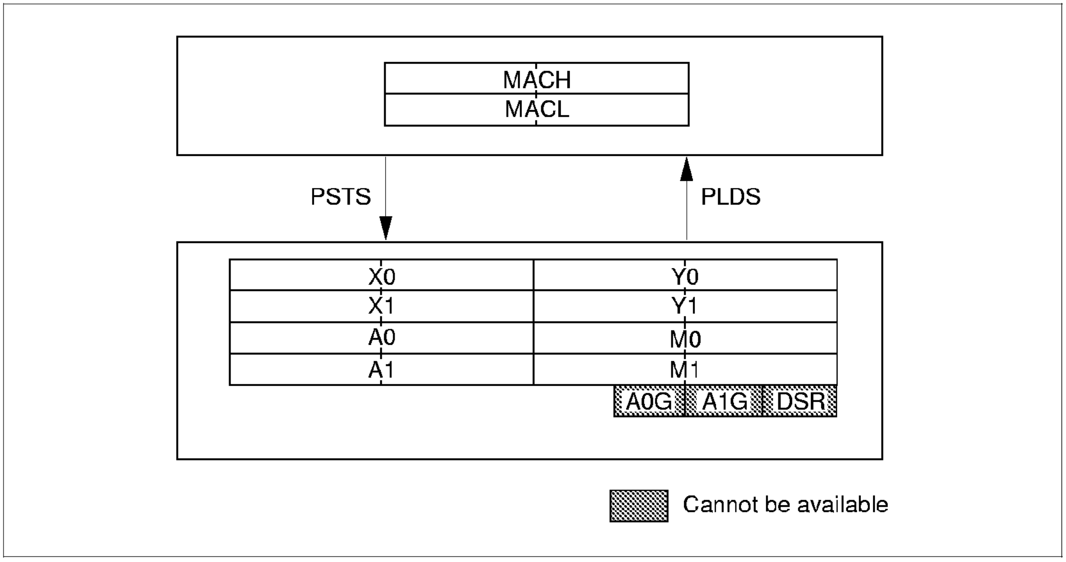


Figure 3.18 Local Data Move Instruction Flow

Table 3.13Variation of Local Data Move Operation

Mnemonic	Function	Operand
PLDS	Data Move from DSP reg. to MACL/H	Dz
PSTS	Data Move from MACL/H to DSP reg.	Dz

This instruction is very similar to other transfer instruction. If one of A0 and A1 registers is specified as the destination operand of PSTS, the signed bit is copied into the corresponding guard-bit parts, A0G or A1G. DC bit and other condition code bits are not updated based upon the instruction result. This instruction can operate conditionally.

Note: Basically, the local data move operation can be specified with MOVX and MOVY in parallel. However, MOVX and this local data move operation use the same hardware resource, so one cycle overhead is inserted when both are specified on the same instruction line.

3.1.11 Operand Conflict

When the identical destination operand is specified with multiple parallel operations, data conflict occurs. Table 3.14 shows the operand flexibility of each operation.

Table 3.14Operand Flexibility

		X-Side Load			Y-Side Load			6-Inst. ALU			6-Inst. Multi			3-Inst. ALU			3-Inst. Multi		
		Ax	Ix	Dx	Ay	Iy	Dy	Sx	Sy	Du	Se	Sf	Dg	Sx	Sy	Dz	Se	Sf	Dg
DSP register	A0							*		⊛			⊛	*		*			*
	A1							*		⊛	*	*	⊛	*		*	*	*	*
	M0								*				*		*	*			*
	M1								*				*		*	*			*
	X0			⊛				*		⊛	*	*		*		⊛	*	*	
	X1			⊛				*			*			*		⊛	*		
	Y0						⊛		*	⊛	*	*			*	⊛	*	*	
	Y1						⊛		*			*			*	⊛		*	

○ Operand confliction case.

'*' Available regs. for operand.

There are three cases of operand conflict problems. Actual hardware will avoid conflict ignoring either one even though such an instruction code is issued.

1. When ALU and multiply instructions specify the same destination operand (Du and Dg), the ALU instruction is executed normally and the multiplier instruction is ignored.
2. When X-side load and ALU instructions specify the same destination operand (Dx, Du, Dz), the ALU instruction is executed normally and the X-side load instruction is ignored.
3. When Y-side load and ALU instructions specify the same destination operand (Dy, Du, Dz), the ALU instruction is executed normally and the Y-side load instruction is ignored.

In these cases above, the result is unknown in the destination register on the specs.

Note: When a PLDS or an LDS instruction is specified in parallel with X-side load instruction, a kind of conflict occurs. However, it's not treated as an operand conflict but a resource conflict because both instructions have to use the same internal bus in the DSP module, so that conflict always occurs even though the specified operand is different.

3.2 DSP Addressing

3.2.1 DSP Loop Control

The SH7729 prepares a special control mechanism for efficient loop control. An instruction 'SETRC' sets repeat times into the repeat counter RC (12 bits), and an execution mode in which a program loop executes repetitively until RC is equal to '1'. After completion of the repeat instructions, the contents of the RC becomes 0.

Repeat start address register RS keeps the start address of a repeat loop. Repeat end register RE keeps the repeat end address. (There are some exceptions. See note, "Actual Implementation Options".) Repeat counter RC keeps the number of repeat times. In order to make this loop control, the following steps are required.

Step 1) Set loop start address into RS
Step 2) Set loop end address into RE
Step 3) Set repeat counter into RC
Step 4) Start repeat control

To do steps 1 and 2, use new instructions:

```
LDRS @(disp,PC); and
LDRE @(disp,PC);
```

For steps 3 and 4, use a new instruction, SETRC. An operand of SETRC is the immediate value or one of the general-purpose registers that will specify repeat times.

```
SETRC #imm;  #imm->RC, enable repeat control
SETRC Rm;    Rm->RC, enable repeat control
```

#imm is 8 bits while RC is 12 bits. Therefore, to set more than 256 into RC, use Rm. A sample program is shown below.

```
           LDRS   RptStart;
           LDRE   RptEnd3+4;
           SETRC  #imm;     RC = #imm
           instr0;
; instr1-5 executes repeatedly
RptStart:  instr1;
RptEnd3:   instr2;
           instr3;
           instr4;
RptEnd:    instr5;
           instr6;
```

In this implementation, there are some restrictions to use this repeat controls as follows:

1. There must be at least one instruction between SETRC and the first instruction of a repeat loop.
2. LDRS and LDRE must be executed before SETRC.
3. In a case that the repeat loop has four or more instructions in it, the one stall cycle is necessary at each iteration if repeat start address (address at the instr1 in above example) is not longword boundary.
4. If a repeat loop has less than four instructions in it, it can't have any branch instructions (BRA, BSR, BT, BF, BT/S, BF/S, BSRF, RTS, BRAF, RTE, JSR and JMP), repeat control instructions (SETRC, LDRS and LDRE), load instructions for SR, RS, RE and TRAPA in it. If these instructions are written, a reserved instruction code exception is executed and a certain address value shown in table 3.15 is stored into SPC.

Table 3.15 Address Value to be Stored into SPC (1)

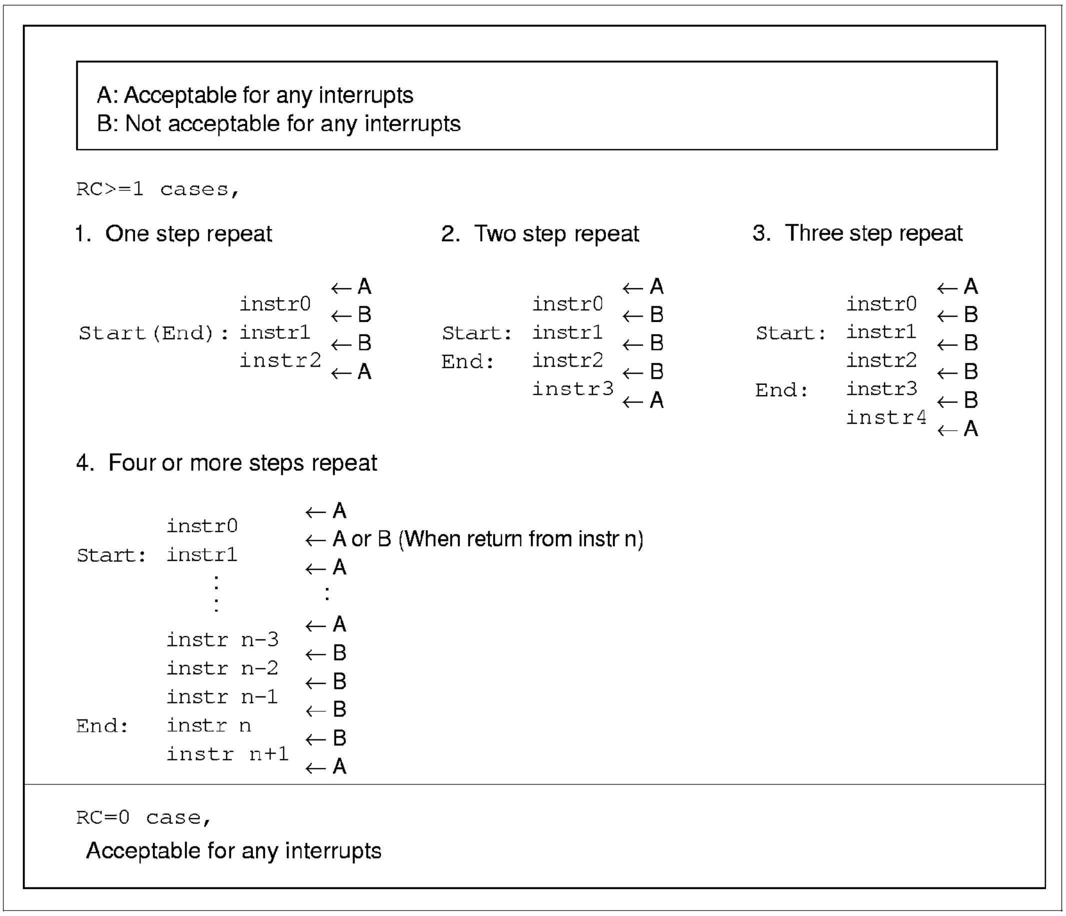
Condition	Location	Address to be Pushed
RC>=2	Any	RptStart
RC=1	Any	Prog. addr. of the illegal inst.

5. If a repeat loop has four or more instructions in it, any branch instructions (BRA, BSR, BT, BF, BT/S, BF/S, BSRF, RTS, BRAF, RTE, JSR and JMP), repeat control instructions (SETRC, LDRS and LDRE), load instructions for SR, RS, RE and TRAPA must not be written within the last three instructions from the bottom of a repeat loop. If written, a reserved instruction code exception is executed and a certain address value shown in table 3.16 is stored into SPC. In cases of repeat control instructions (SETRC, LDRS and LDRE) and load instructions for SR, RS, RE and TRAPA they cannot be placed in any other location of the repeat module, either. If they are, the operation is not guaranteed.

Table 3.16Address Value to be Stored into SPC (2)

Condition	Location	Address to be Pushed
RC>=2	instr3	Prog. addr. of the illegal inst.
	instr4	RptStart - 4
	instr5	RptStart - 2
RC=1	Any	Prog. addr. of the illegal inst.

6. If a repeat loop has less than four instructions in it, any PC relative instructions (MOVA @(disp,PC),R0, etc.) don't work properly except the instruction at the repeat top (instr1).
7. If a repeat loop has four or more instructions in it, any PC relative instructions (MOVA @(disp,PC),R0, etc.) don't work properly at two instructions from the repeat bottom.
8. The CPU has no repeat enable flag, however it uses the condition RC = 0 to disable repeat control. Whenever RC is not '0' and PC matches RE, the repeat control is alive. When '0' is set in the RC, the repeat control is disabled but the repeat loop is executed once and does not return to the repeat start as well as RC = 1 case. When RC = 1, the repeat loop is executed once and does not return to the repeat start but the RC becomes 0 after completing the execution of the repeat loop.
9. If a repeat loop has more than three instructions in it, any branch instructions, including subroutine call and return instructions, cannot specify the instruction from "inst3" to "inst5" in previous example as the branch target address. If executed, the repeat control doesn't work so the program go through the following instruction and the RC is also not updated. When a repeat loop has less than four instructions in it, the repeat control doesn't work properly and the contents of the RC in SR register is not updated if the branch target is the "RptStart" or a subsequent address.
10. Interrupt acceptance is restricted during repeat loop processing. See figure 3.19 for detail restrictions. Here, the flow of each case in figure 3.19 shows the EX stages. Usually interrupt starts right after the instruction's EX stage is finished. These are specified by "A" in the figure. However, at the point specified by "B", interrupt is not accepted.



A: Acceptable for any interrupts
B: Not acceptable for any interrupts

RC>=1 cases,

1. One step repeat

```
                     ← A
             instr0
                     ← B
Start(End) : instr1
                     ← B
             instr2
                     ← A
```

2. Two step repeat

```
                  ← A
        instr0
                  ← B
Start:  instr1
                  ← B
End:    instr2
                  ← B
        instr3
                  ← A
```

3. Three step repeat

```
                  ← A
        instr0
                  ← B
Start:  instr1
                  ← B
        instr2
                  ← B
End:    instr3
                  ← B
        instr4
                  ← A
```

4. Four or more steps repeat

```
                    ← A
        instr0
                    ← A or B (When return from instr n)
Start:  instr1
                    ← A
          :
                      :
          :
                    ← A
        instr n-3
                    ← B
        instr n-2
                    ← B
        instr n-1
                    ← B
End:    instr n
                    ← B
        instr n+1
                    ← A
```

RC=0 case,

Acceptable for any interrupts

Figure 3.19 Restriction of Interrupt acceptance in Repeat Loop

Note 1: Actual Implementation

Repeat start and repeat end registers, RS and RE keep repeat start address and repeat end address. The addresses that are kept in these registers depend on the number of instructions in the repeat loop. The rule is as follows,

Repeat_Start: An address of the instruction at the repeat top.
Repeat_Start0: An address of the instruction before one instruction at the repeat top.
Repeat_End3: An address of the instruction before three instruction at the repeat bottom.

Table 3.17RS and RE Setting Rule

	Number of Instructions in a Repeat Loop			
	1	**2**	**3**	**>=4**
RS	Repeat_start0 +8	Repeat_start0 +6	Repeat_start0 +4	Repeat_start
RE	Repeat_start0 +4	Repeat_start0 +4	Repeat_start0 +4	Repeat_End3+4

Based on this table, the actual repeat programming for various cases should be described as in the following examples:

CASE 1: 1 Repeated Instruction

```
             LDRS  RptStart0+8;
             LDRE  RptStart0+4;
             SETRC RptCount;
                - - - -
RptStart0:   instr0;
RptStart:    instr1;              Repeated instruction
             instr2;
```

CASE 2: 2 Repeated Instructions

```
             LDRS  RptStart0+6;
             LDRE  RptStart0+4;
             SETRC RptCount;
                - - - -
RptStart0:   instr0;
RptStart:    instr1;              Repeated instruction 1
RptEnd:      instr2;              Repeated instruction 2
             instr3;
```

CASE 3: 3 Repeated Instructions

```
             LDRS  RptStart0+4;
             LDRE  RptStart0+4;
             SETRC RptCount;
                - - - -
 RptStart0:  instr0;
 RptStart:   instr1;          Repeated instruction 1
             instr2;          Repeated instruction 2
 RptEnd:     instr3;          Repeated instruction 3
             instr4;
```

CASE 4: 4 or more Repeated Instructions

```
             LDRS  RptStart;
             LDRE  RptEnd3+4;
             SETRC RptCount;
                - - - -
 RptStart0:  instr0;
 RptStart:   instr1;          Repeated instruction 1
             instr2;          Repeated instruction 2
             instr3;          Repeated instruction 3
 ----------------------------------------------------------
 RptEnd3:    instrN-3;        Repeated instruction N
             instrN-2;        Repeated instruction N-2
             instrN-1;        Repeated instruction N-1
 RptEnd:     instrN;          Repeated instruction N
             instrN+1;
```

The examples above can be used as a template to program this repeat loop sequences. However, for easy programming, an extended instruction “REPEAT” will be provided to handle these complex labeling and offset issues. Details will be described in the following note 2.

Note 2: Extended Instruction REPEAT

This REPEAT extended instruction will handle all the delicate labeling and offset processing described in table 3.17 and note 1. The labels used here are:

Rptart: An address of the instruction at the top of the repeat loop.
RptEnd: An address of the instruction at the bottom of the repeat loop.
RptCount: Repeat count immediate number

This instruction can be used in the following way:

Here repeat count can be specified as an immediate value #Imm or a register indirect value Rn.

CASE 1: 1 Repeated Instruction

```
REPEAT RptStart, RptStart, RptCount;
               - - - -
            instr0;
RptStart:   instr1;              Repeated instruction
            instr2;
```

CASE 2: 2 Repeated Instruction

```
REPEAT RptStart, RptEnd, RptCount;
               - - - -
            instr0;
RptStart:   instr1;              Repeated instruction 1
RptEnd:     instr2;              Repeated instruction 2
```

CASE 3: 3 Repeated Instruction

```
REPEAT RptStart, RptEnd, RptCount;
               - - - -
            instr0;
RptStart:   instr1;              Repeated instruction 1
            instr2;              Repeated instruction 2
RptEnd:     instr3;              Repeated instruction 3
```

CASE 4: 4 or more Repeated Instructions

```
REPEAT RptStart, RptEnd, RptCount;
               - - - -
            instr0;
```

```
RptStart    instr1;         Repeated instruction 1
            instr2;         Repeated instruction 2
            instr3;         Repeated instruction 3
-----------------------------------------------------------
            instrN-3;       Repeated instruction N
            instrN-2;       Repeated instruction N-2
            instrN-1;       Repeated instruction N-1
RptEnd      instrN;         Repeated instruction N
            instrN+1;
```

The expanded results of each case corresponds to the same case numbers in note 1.

3.2.2 DSP Data Addressing

The SH7729 has two types of memory access instructions: one is “X and Y data transfer instruction” (MOVX.W and MOVY.W), and the other one is “single data transfer instruction” (MOVS.W and MOVS.L). Data addressing of these two types of instruction are different. Table 3.18 shows a summary of DSP data transfer instructions.

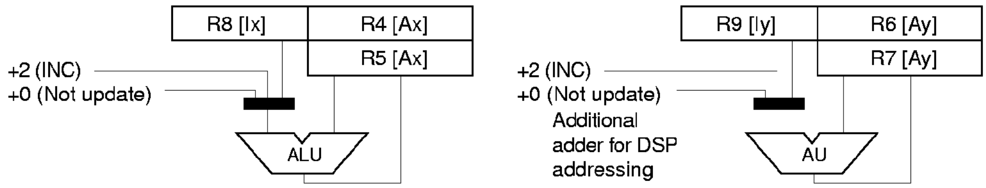
Table 3.18Summary of DSP Data Transfer Instructions

	X and Y Data Transfer Operation (MOVX.W MOVY.W)	**Single Data Transfer Operation (MOVS.W, MOVS.L)**
Address registers	Ax: R4 and R5, Ay: R6 and R7	As: R2, R3, R4 and R5
Index register(s)	Ix: R8, Iy: R9	Is: R8
Addressing operations	Nop/Inc (+2)/Add-index-reg: Post-update	Nop/Inc (+2, +4)/Add-index-reg: Post-update Dec (–2, –4): Pre-update
Modulo addressing	Yes	No
Data bus	XDB and YDB	IDB
Data length	16 bit (word)	16 bit/32 bit (word/longword)
Bus conflict	No	Possible (same as the SH)
Memory	X and Y data memories	All memory space
Source registers	Dx, Dy: A0 and A1	DS: A0/1, M0/1, X0/1, Y0/1, A0G, A1G
Destination registers	Dx: X0/1, Dy: Y0/1	Ds: A0/1, M0/1, X0/1, Y0/1, A0G, A1G

Addressing Instructions for MOVX.W and MOV.W: The SH7729 can access X and Y data memories simultaneously (MOVX.W and MOVY.W). The DSP instructions have two address pointers that *simultaneously* access X and Y data memories. The DSP instruction has only pointer-addressing (it does not have immediate-addressing). Address registers are divided into two sets, R4,5 (Ax: Address register for X memory) and R6,7 (Ay: Address register for Y memory). There are three data addressing operations for X and Y data transfer instruction.

1. Not-update address register
2. Add-index register
3. Increment address register

Each address pointer set has an index register, R8[Ix] for set Ax, and R9[Iy] for set Ay. Address instructions for set Ax use ALU in the CPU, and address instructions for set Ay use additional address unit (figure 3.20).



Three address operation types,
1. Increment
2. Add-index-register (Ix/Iy)
3. Not update

All operations are post-update type.
To decrement address pointer, set –2 to an index register.

Figure 3.20 DSP Addressing Instructions for MOVX.W and MOVY.W

Addressing in X and Y data transfer operation is always word mode; that is access to X and Y data memories are 16bit data width. Therefore, the increment operation adds '2' to an address register. To realize decrement, set '–2' to an index register and use add-index-register operation.

Addressing Instructions for MOVS: The SH7729 has single-data transfer instruction (`MOVS.W` and `MOVS.L`) to load/store DSP data registers. In this instruction, R2 to 5 (As: Address register for single-data transfer) are used for address pointer.

There are four data addressing instructions for single data transfer operation.

1. Not-update address register
2. Add-index register (post-update)
3. Increment address register (post-update)
4. Decrement address register (pre-update)

The address pointer set. As has an index register R8[Is] (figure 3.21)

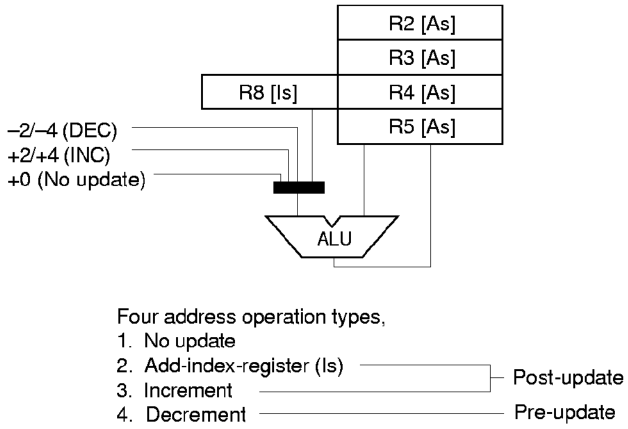


Figure 3.21 DSP Addressing Instructions for MOVS

Modulo Addressing: The SH7729 provides modulo addressing mode, which is common in DSPs. In modulo addressing mode, the address register is updated as explained above. When the address pointer reaches the pre-defined address (the modulo-end address), it goes to the modulo start address.

Modulo addressing is available for X and Y data transfer instruction (MOVX and MOVY), but not single-data transfer instruction (MOVS). The DMX and DMY in the SR register are used for the modulo addressing control. If the DMX is ‘1’ then the modulo addressing mode is effective for the X memory address pointer Ax (R4 or R5). If the DMY is ‘1’ then it is effective for the Y memory address pointer Ay (R6 or R7). Modulo addressing is available for one of X and Y address registers at one time. So DMX = DMY = 1 case is reserved for future expansion. When both DMX and DMY are set simultaneously, the hardware will preliminary treat as the modulo addressing mode for the Y address pointer only.

To specify the start and the end addresses of the modulo address area, the MOD register, which includes MS (modulo start) and ME (modulo end) is prepared. Following example shows a way to set the MOD (MS and ME) register.

```
            MOV.L ModAddr,Rn;        Rn=ModEnd, ModStart
            LDC Rn,MOD;              ME=ModEnd, MS=ModStart
ModAddr:    .DATA.W     mEnd;        Lower 16 bits of ModEnd
            .DATA.W     mStart;      Lower 16 bits of ModStart
```

```
ModStart:   .DATA
              :
ModEnd:     .DATA
```

MS and ME are set to specify the start and the end addresses, and then to set the DMX or DMY bit to '1.' The content of the address register is compared to ME. If it matches ME, the start address in MS is restored in the address register. Bits 1 to 15 of address register is compared to ME. The ME register holds the bit 0 also but it's not used. The maximum modulo size is 64 KB (it may exceed the memory size of the X or Y data memory through). Figure 3.22 shows block diagram of the modulo addressing.

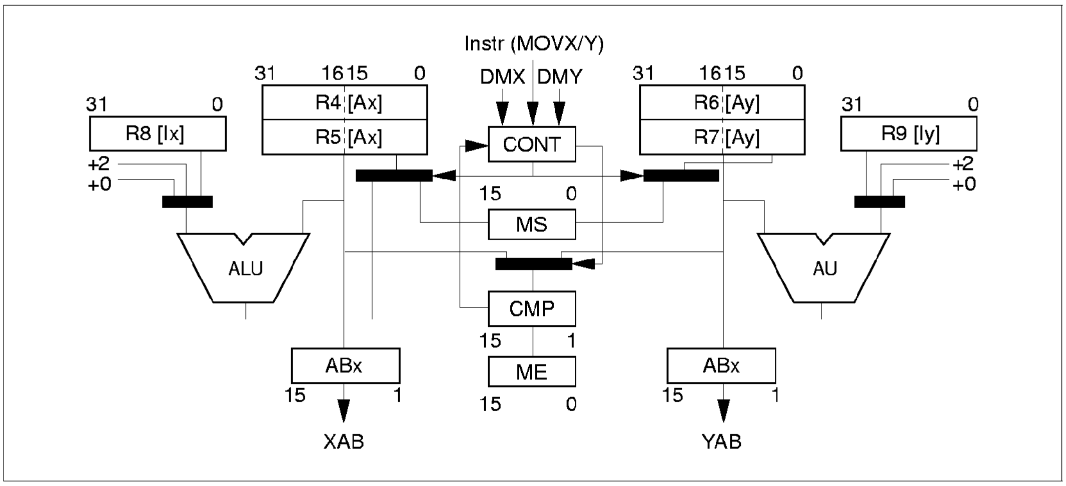


Figure 3.22 Modulo Addressing for Address Registers

An example is shown bellow.

```
MS=H'F000; ME=H'F004; R4=H'0800F000;
DMX=1; DMY=0 (modulo addressing for address register set Ax(R4,5))
                    ; R4: H'0800F000
MOVX.W @R4+,Dx      ; R4: H'0800F002
MOVX.W @R4+,Dx      ; R4: H'0800F004
MOVX.W @R4+,Dx      ; R4: H'0800F000
MOVX.W @R4+,Dx      ; R4: H'0800F002
```

The upper 16 bits of the modulo start and end addresses must be the same. Because the modulo start address replaces only 16 bits of the address register.

When the add-index register instruction is used for DSP data addressing, the address pointer might exceed ME instead of matching it. In this case, the address pointer does not return to the modulo start address.

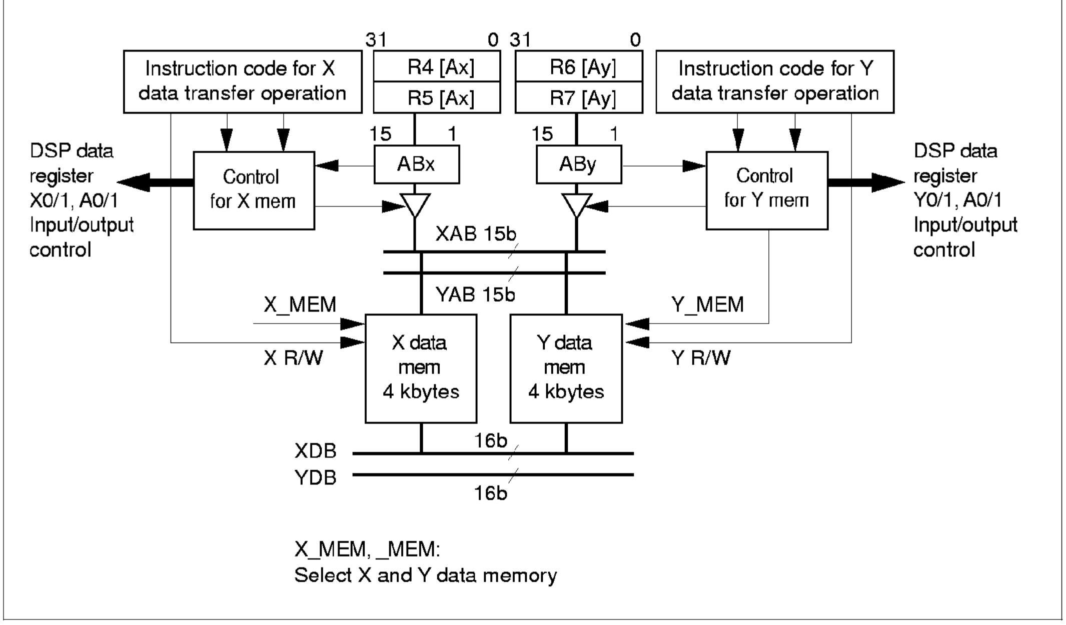
Addressing Instruction in the Execution Stage: Address instructions, including modulo addressing, are executed in the execution stage of the pipeline. Behavior of the DSP data addressing in the execution stage is:

```
if ( Operation is MOVX.W MOVY.W ) {
ABx=Ax; ABy=Ay;
/* Memory access cycle uses ABx and ABy. The addresses to be used have not
been updated. */

/* Ax is one of R4,5 */
if {DMX==0 || ( DMX==1 && DMY==1 )}Ax = Ax+(+2 or R8[Ix] or +0);
      /* Inc,Index,Not-Update */
else if (! not-update) Ax=modulo( Ax, (+2 or R8[Ix]) );

/* Ay is one of R6,7 */
if ( DMY==0 ) Ay=Ay+(+2 or R9[Iy] or +0); /* Inc,Index,Not-Update */
else if (! not-update) Ay=modulo( Ay, (+2 or R9[Iy]) );
}
else if ( Operation is MOVS.W or MOVS.L ) {
   if ( Addressing is Nop, Inc, Add-index-reg ) {
MAB=As;
/* Memory access cycle uses MAB. The address to be used has not been
updated. */

/* As is one of R2-5 */
As = As+(+2 or +4 or R8[Is] or +0); /* Inc,Index,Not-Update */
   else {/* Decrement, Pre-update */
/* As is one of R2-5 */
As=As+(-2 or -4);
MAB=As;
/* Memory access cycle uses MAB. The address to be used has been updated. */
}
/* The value to be added to the address register depends on addressing
instructions.

For example, (+2 or R8[Ix] or +0) means that
```

```
+2:     if instruction is increment
R8[Ix]: if instruction is add-index-register
+0:     if instruction is not-update
*/

function modulo ( AddrReg, Index ) {
if ( AddrReg[15:1]==ME[15:1] ) AddrReg[15:1]==MS[15:1];
else AddrReg=AddrReg+Index;
return AddrReg;
}
```

X and Y Data Transfer Instruction (MOVX.W and MOVY.W): This type of instruction uses the XDB and the YDB to access X and Y data memories (they cannot access other memory space). These two buses are separate bus from the instruction bus, therefore, there is no access conflict between the data memory access and instruction memory access.

Figure 3.23 shows load/store control for X and Y data transfer instruction. All memory accesses are word mode accesses.



31
0
31
0
Instruction code for X data transfer operation
R4 [Ax]
R5 [Ax]
R6 [Ay]
R7 [Ay]
Instruction code for Y data transfer operation
15
1
15
1
DSP data register X0/1, A0/1 Input/output control
Control for X mem
ABx
ABy
Control for Y mem
DSP data register Y0/1, A0/1 Input/output control
XAB 15b
YAB 15b
X_MEM
X R/W
X data mem 4 kbytes
Y data mem 4 kbytes
Y_MEM
Y R/W
XDB
YDB
16b
16b
X_MEM, _MEM:
Select X and Y data memory

Figure 3.23 Load/Store Control for X and Y Data-Transfer Instruction

Control for X mem

```
if ( !Nop ) {
      X_MEM=1; XAB=ABx;
      if ( load operation ) {
            Dx[31:16]=XDB;
            Dx[15:0]=0x0000;       /* Dx is X0 or X1 */
      }
      else XDB = Dx[31:16];        /* Dx is A0 or A1 */
}
else {X_MEM=0; XAB=0x000; }
```

The conditional execution based on a DC flag in the DSR cannot control any MOVX/MOVY instructions.

Single-Data Transfer Instruction (MOVS.W and MOVS.L): The SH7729 has single load/store instruction for the DSP registers. It is similar to a load/store instruction for a system register. It transfers data between memory and DSP data registers using IAB and LDB. There may be access conflict between the data access and the instruction fetch.

The single-data transfer instruction has word and longword access modes. Figure 3.24 shows a block diagram of single-data transfer. Control of the memory address buffer (MAB) and the memory select uses existing SH-CPU core design.

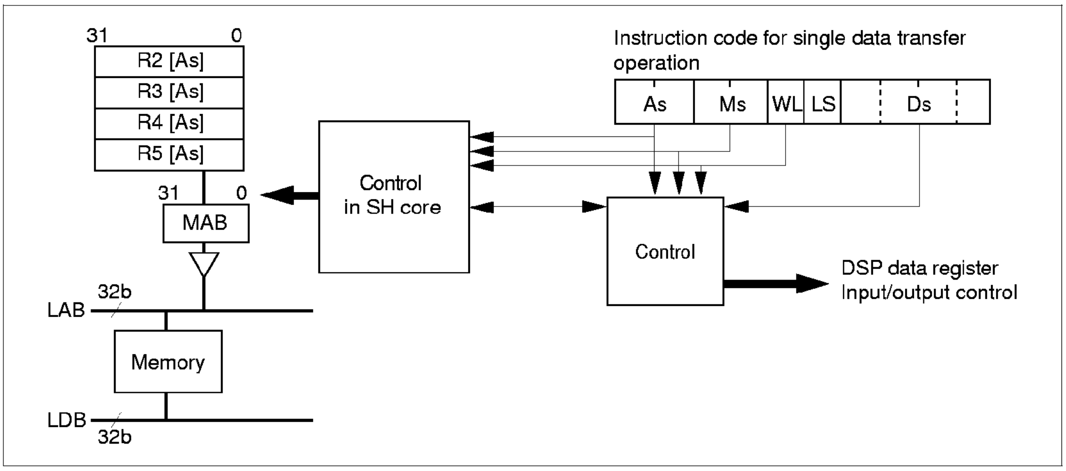


Figure 3.24 Load/Store Control for Single-Data Transfer Instruction

Control

```
LAB=MAB;
if ( Ms!=NLS && W/L is word access ) {/* MOVS.W */
      if (LS==load) {
            if (Ds!=A0G && Ds!=A1G) {
                  Ds[31:16]=LDB[15:0]; Ds[15:0]=0x0000;
                  if (Ds==A0) A0G[7:0]=sign-extension of LDB;
                  if (Ds==A1) A1G[7:0]=sign-extension of LDB;
            }
            else Ds[7:0]=LDB[7:0];      /* Ds is A0G or A1G */
      }
      else {/* Store */
            if (Ds!=A0G && Ds!=A1G) LDB[15:0]=Ds[31:16];
            /* Ds is A0G or A1G */
            else LDB[15:0]=Ds[7:0] with 8bit sign-extension;
      }
}
else if ( MA!=NLS && W/L is long-word access ) {/* MOVS.L */
      if (LS==load) {
            if (Ds!=A0G && Ds!=A1G) {
                  Ds[31:0]=LDB[31:0];
                  if (Ds==A0) A0G[7:0]=sign-extension of LDB;
                  if (Ds==A1) A1G[7:0]=sign-extension of LDB;
            }
            else Ds[7:0]=LDB[7:0]; /* Ds is A0G or A1G */
      }
      else {/* Store */
            if (Ds!=A0G && Ds!=A1G) LDB[31:0]=Ds[31:0];
            /* Ds is A0G or A1G */
            else LDB[31:0]=Ds[7:0] with 24bit sign-extension;
      }
}
```

Section 4 Instruction Set

4.1 Basic Concept of SH7729 Instruction Set

In order to improve digital signal processing performance, DSP type of instructions are added to form SH7729's ISA. Its relationship with the rest of the SH family is:

1. Object-code level upward compatible with SH1, SH2 and SH3.
2. Instructions of DSP extension are object-code level compatible with DSP extension in SH-DSP.

This section is organized in two parts: section 4.2, SH1, SH2, SH3 Compatible Instruction Set and section 4.3, Instructions for DSP Extension. Instructions described in section 4.2 are all 16-bit in length and are compatible to SH1, SH2 and SH3 (SH7708) series. DSP Extension instructions are divided into 3 groups:

1. Additional system control instructions for CPU unit, e.g. setting up repeat loop control and modulo addressing.
2. Single- or double-data transfer between memory and registers in the DSP unit
3. Parallel instruction for the DSP unit.

Groups 1 and 2 are 16-bit in length, while 3 are 32-bit instructions which can specify up to four parallel instructions (two load/store, one ALU and one Multiply) at the same time.

4.2 SH1, SH2, SH3 Compatible Instruction Set

4.2.1 Instruction Set by Classification

SH1, SH2 and SH3 instruction set includes 66 basic instruction types, divided into seven functional classifications, as listed in table 4.1. Tables 4.3 to 4.8 summarize instruction notation, machine mode, execution time, and function.

Table 4.1 Classification of Instructions

Classification	Types	Operation Code	Function	Number of Instructions
Data transfer	6	MOV	Data transfer Immediate data transfer Peripheral module data transfer Structure data transfer	40
		MOVA	Effective address transfer	
		MOVT	T-bit transfer	
		SWAP	Swap of upper and lower bytes	
		XTRCT	Extraction of the middle of registers connected	
		PREF	Prefetching data to cache	
Arithmetic operations	21	ADD	Binary addition	34
		ADDC	Binary addition with carry	
		ADDV	Binary addition with overflow check	
		CMP/cond	Comparison	
		DIV1	Division	
		DIV0S	Initialization of signed division	
		DIV0U	Initialization of unsigned division	
		DMULS	Signed double-length multiplication	
		DMULU	Unsigned double-length multiplication	
		DT	Decrement and test	
		EXTS	Sign extension	
		EXTU	Zero extension	
		MAC	Multiply/accumulate, double-length multiply/accumulate operation	

Table 4.1 Classification of Instructions (cont)

Classification	Types	Operation Code	Function	Number of Instruction s
Arithmetic operations (cont)	21	MUL	Double-length multiplication (32 × 32 bits)	34
		MULS	Signed multiplication (16 × 16 bits)	
		MULU	Unsigned multiplication (16 × 16 bits)	
		NEG	Negation	
		NEGC	Negation with borrow	
		SUB	Binary subtraction	
		SUBC	Binary subtraction with carry	
		SUBV	Binary subtraction with underflow check	
Logic operations	6	AND	Logical AND	14
		NOT	Bit inversion	
		OR	Logical OR	
		TAS	Memory test and bit set	
		TST	Logical AND and T-bit set	
		XOR	Exclusive OR	
Shift	12	ROTL	One-bit left rotation	16
		ROTR	One-bit right rotation	
		ROTCL	One-bit left rotation with T bit	
		ROTCR	One-bit right rotation with T bit	
		SHAL	One-bit arithmetic left shift	
		SHAR	One-bit arithmetic right shift	
		SHLL	One-bit logical left shift	
		SHLLn	n-bit logical left shift	
		SHLR	One-bit logical right shift	
		SHLRn	n-bit logical right shift	
		SHAD	Dynamic arithmetic shift	
		SHLD	Dynamic logical shift	

Table 4.1 Classification of Instructions (cont)

Classification	Types	Operation Code	Function	Number of Instructions
Branch	9	BF	Conditional branch, conditional branch with delay (T = 0)	11
		BT	Conditional branch, conditional branch with delay (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	14	CLRT	T-bit clear	74
		CLRMAC	MAC register clear	
		CLRS	S-bit clear	
		LDC	Load to control register	
		LDS	Load to system register	
		LDTLB	Load PTE to TLB	
		NOP	No operation	
		RTE	Return from exception processing	
		SETS	S-bit set	
		SETT	T-bit set	
		SLEEP	Shift into power-down mode	
		STC	Storing control register data	
		STS	Storing system register data	
		TRAPA	Trap exception handling	
Total:	66			189

Instruction codes, instruction, and execution states are listed in table 4.2 by classification. Tables 4.3 through 4.8 list the minimum number of clock cycles required for execution. In practice, the number of execution cycles increases when the instruction fetch is in contention with data access or when the destination register of a load instruction (memory → register) is the same as the register used by the next instruction.

Table 4.2 Instruction Code Format

Item	Format	Explanation

Instruction mnemonic	`OP.Sz`
	`SRC,DEST`
		OP: Operation code
		Sz: Size
		SRC: Source
		DEST: Destination
		Rm: Source register
		Rn: Destination register
		imm: Immediate data
		disp: Displacement

Instruction code	MSB ↔ LSB	mmmm: Source register nnnn: Destination register 0000: R0 0001: R1 . . . . . . 1111: R15 iiii: Immediate data dddd: Displacement
Operation summary	→, ← (xx) M/Q/T & | ^ ~ <<n, >>n	Direction of transfer Memory operand Flag bits in the SR Logical AND of each bit Logical OR of each bit Exclusive OR of each bit Logical NOT of each bit n-bit shift
Execution cycle		Value when no wait states are inserted
Instruction execution cycles		The execution cycles listed in the table are minimums. The actual number of cycles may be increased: 1. When contention occurs between instruction fetches and data access, or 2. When the destination register of the load instruction (memory → register) and the register used by the next instruction are the same.
T bit		Value of T bit after instruction is executed
—		No change

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

Table 4.3 Data Transfer Instructions

Instruction		Operation	Code	Cycles	T Bit
MOV	#imm,Rn	#imm → Sign extension → Rn	1110nnnniiiiiiii	1	—
MOV.W	@(disp,PC),Rn	(disp × 2 + PC) → Sign extension → Rn	1001nnnndddddddd	1	—
MOV.L	@(disp,PC),Rn	(disp × 4 + PC) → Rn	1101nnnndddddddd	1	—
MOV	Rm,Rn	Rm → Rn	0110nnnnmmmm0011	1	—
MOV.B	Rm,@Rn	Rm → (Rn)	0010nnnnmmmm0000	1	—
MOV.W	Rm,@Rn	Rm → (Rn)	0010nnnnmmmm0001	1	—
MOV.L	Rm,@Rn	Rm → (Rn)	0010nnnnmmmm0010	1	—
MOV.B	@Rm,Rn	(Rm) → Sign extension → Rn	0110nnnnmmmm0000	1	—
MOV.W	@Rm,Rn	(Rm) → Sign extension → Rn	0110nnnnmmmm0001	1	—
MOV.L	@Rm,Rn	(Rm) → Rn	0110nnnnmmmm0010	1	—
MOV.B	Rm,@-Rn	Rn–1 → Rn, Rm → (Rn)	0010nnnnmmmm0100	1	—
MOV.W	Rm,@-Rn	Rn–2 → Rn, Rm → (Rn)	0010nnnnmmmm0101	1	—
MOV.L	Rm,@-Rn	Rn–4 → Rn, Rm → (Rn)	0010nnnnmmmm0110	1	—
MOV.B	@Rm+,Rn	(Rm) → Sign extension → Rn, Rm + 1 → Rm	0110nnnnmmmm0100	1	—
MOV.W	@Rm+,Rn	(Rm) → Sign extension → Rn, Rm + 2 → Rm	0110nnnnmmmm0101	1	—
MOV.L	@Rm+,Rn	(Rm) → Rn, Rm + 4 → Rm	0110nnnnmmmm0110	1	—
MOV.B	R0,@(disp,Rn)	R0 → (disp + Rn)	10000000nnnndddd	1	—
MOV.W	R0,@(disp,Rn)	R0 → (disp × 2 + Rn)	10000001nnnndddd	1	—
MOV.L	Rm,@(disp,Rn)	Rm → (disp × 4 + Rn)	0001nnnnmmmmdddd	1	—
MOV.B	@(disp,Rm),R0	(disp + Rm) → Sign extension → R0	10000100mmmmdddd	1	—
MOV.W	@(disp,Rm),R0	(disp × 2 + Rm) → Sign extension → R0	10000101mmmmdddd	1	—
MOV.L	@(disp,Rm),Rn	(disp × 4 + Rm) → Rn	0101nnnnmmmmdddd	1	—
MOV.B	Rm,@(R0,Rn)	Rm → (R0 + Rn)	0000nnnnmmmm0100	1	—
MOV.W	Rm,@(R0,Rn)	Rm → (R0 + Rn)	0000nnnnmmmm0101	1	—
MOV.L	Rm,@(R0,Rn)	Rm → (R0 + Rn)	0000nnnnmmmm0110	1	—
MOV.B	@(R0,Rm),Rn	(R0 + Rm) → Sign extension → Rn	0000nnnnmmmm1100	1	—
MOV.W	@(R0,Rm),Rn	(R0 + Rm) → Sign extension → Rn	0000nnnnmmmm1101	1	—

Table 4.3 Data Transfer Instructions (cont)

Instruction		Operation	Code	Cycles	T Bit
MOV.L	@(R0,Rm),Rn	(R0 + Rm) → Rn	0000nnnnmmmm1110	1	—
MOV.B	R0,@(disp,GBR)	R0 → (disp + GBR)	11000000dddddddd	1	—
MOV.W	R0,@(disp,GBR)	R0 → (disp × 2 + GBR)	11000001dddddddd	1	—
MOV.L	R0,@(disp,GBR)	R0 → (disp × 4 + GBR)	11000010dddddddd	1	—
MOV.B	@(disp,GBR),R0	(disp + GBR) → Sign extension → R0	11000100dddddddd	1	—
MOV.W	@(disp,GBR),R0	(disp × 2 + GBR) → Sign extension → R0	11000101dddddddd	1	—
MOV.L	@(disp,GBR),R0	(disp × 4 + GBR) → R0	11000110dddddddd	1	—
MOVA	@(disp,PC),R0	disp × 4 + PC → R0	11000111dddddddd	1	—
MOVT	Rn	T → Rn	0000nnnn00101001	1	—
PREF	@Rn	(Rn) → cache	0000nnnn10000011	1	—
SWAP.B	Rm,Rn	Rm → Swap the bottom two bytes → REG	0110nnnnmmmm1000	1	—
SWAP.W	Rm,Rn	Rm → Swap two consecutive words → Rn	0110nnnnmmmm1001	2	—
XTRCT	Rm,Rn	Rm: Middle 32 bits of Rn → Rn	0010nnnnmmmm1101	1	—

Table 4.4 Arithmetic Instructions

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

Table 4.4 Arithmetic Instructions (cont)

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

Notes: 1. The normal minimum number of execution cycles is 2, but 5 cycles are required when the results of an operation are read from the MAC register immediately after the instruction.

2. The normal minimum number of execution cycles is 1, but 3 cycles are required when the results of an operation are read from the MAC register immediately after a MUL instruction.

Table 4.5 Logic Operation Instructions

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

Table 4.6 Shift Instructions

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

Table 4.7 Branch Instructions

Instruction		Operation	Code	Cycles	T Bit
BF	label	If T = 0, disp × 2 + PC → PC; if T = 1, nop (where label is disp + PC)	10001011dddddddd	3/1*	—
BF/S	label	Delayed branch, if T = 0, disp × 2 + PC → PC; if T = 1, nop	10001111dddddddd	2/1*	
BT	label	If T = 1, disp × 2 + PC → PC; if T = 0, nop	10001001dddddddd	3/1*	—
BT/S	label	Delayed branch, if T = 1, disp × 2 + PC → PC; if T = 0, nop	10001101dddddddd	2/1*	—
BRA	label	Delayed branch, disp × 2 + PC → PC	1010dddddddddddd	2	—
BRAF	Rn	Rn + PC → PC	0000nnnn00100011	2	—
BSR	label	Delayed branch, PC → PR, disp × 2 + PC → PC	1011dddddddddddd	2	—
BSRF	Rn	PC → PR, Rn + PC → PC	0000nnnn00000011	2	—
JMP	@Rn	Delayed branch, Rn → PC	0100nnnn00101011	2	—
JSR	@Rn	Delayed branch, PC → PR, Rn → PC	0100nnnn00001011	2	—
RTS		Delayed branch, PR → PC	0000000000001011	2	—

Note: * One state when it does not branch.

Table 4.8 System Control Instructions

Instruction		Operation	Code	Cycles	T Bit
CLRMAC		0 → MACH, MACL	0000000000101000	1	—
CLRS		0 → S	0000000001001000	1	—
CLRT		0 → T	0000000000001000	1	0
LDC	Rm,SR	Rm → SR	0100mmmm00001110	5	LSB
LDC	Rm,GBR	Rm → GBR	0100mmmm00011110	3	—
LDC	Rm,VBR	Rm → VBR	0100mmmm00101110	3	—
LDC	Rm,SSR	Rm → SSR	0100mmmm00111110	3	—
LDC	Rm,SPC	Rm → SPC	0100mmmm01001110	3	—
LDC	Rm,R0_BANK	Rm → R0_BANK	0100mmmm10001110	3	—
LDC	Rm,R1_BANK	Rm → R1_BANK	0100mmmm10011110	3	—
LDC	Rm,R2_BANK	Rm → R2_BANK	0100mmmm10101110	3	—
LDC	Rm,R3_BANK	Rm → R3_BANK	0100mmmm10111110	3	—
LDC	Rm,R4_BANK	Rm → R4_BANK	0100mmmm11001110	3	—
LDC	Rm,R5_BANK	Rm → R5_BANK	0100mmmm11011110	3	—
LDC	Rm,R6_BANK	Rm → R6_BANK	0100mmmm11101110	3	—
LDC	Rm,R7_BANK	Rm → R7_BANK	0100mmmm11111110	3	—
LDC.L	@Rm+,SR	(Rm) → SR, Rm + 4 → Rm	0100mmmm00000111	7	LSB
LDC.L	@Rm+,GBR	(Rm) → GBR, Rm + 4 → Rm	0100mmmm00010111	5	—
LDC.L	@Rm+,VBR	(Rm) → VBR, Rm + 4 → Rm	0100mmmm00100111	5	—
LDC.L	@Rm+,SSR	(Rm) → SSR, Rm + 4 → Rm	0100mmmm00110111	5	—
LDC.L	@Rm+,SPC	(Rm) → SPC, Rm + 4 → Rm	0100mmmm01000111	5	—
LDC.L	@Rm+,R0_BANK	(Rm) → R0_BANK, Rm + 4 → Rm	0100mmmm10000111	5	—
LDC.L	@Rm+,R1_BANK	(Rm) → R1_BANK, Rm + 4 → Rm	0100mmmm10010111	5	—
LDC.L	@Rm+,R2_BANK	(Rm) → R2_BANK, Rm + 4 → Rm	0100mmmm10100111	5	—
LDC.L	@Rm+,R3_BANK	(Rm) → R3_BANK, Rm + 4 → Rm	0100mmmm10110111	5	—
LDC.L	@Rm+,R4_BANK	(Rm) → R4_BANK, Rm + 4 → Rm	0100mmmm11000111	5	—
LDC.L	@Rm+,R5_BANK	(Rm) → R5_BANK, Rm + 4 → Rm	0100mmmm11010111	5	—

Table 4.8 System Control Instructions (cont)

Instruction		Operation	Code	Cycles	T Bit
LDC.L	@Rm+,R6_BANK	(Rm) → R6_BANK, Rm + 4 → Rm	0100mmmm11100111	5	—
LDC.L	@Rm+,R7_BANK	(Rm) → R7_BANK, Rm + 4 → Rm	0100mmmm11110111	5	—
LDS	Rm,MACH	Rm → MACH	0100mmmm00001010	1	—
LDS	Rm,MACL	Rm → MACL	0100mmmm00011010	1	—
LDS	Rm,PR	Rm → PR	0100mmmm00101010	1	—
LDS.L	@Rm+,MACH	(Rm) → MACH, Rm + 4 → Rm	0100mmmm00000110	1	—
LDS.L	@Rm+,MACL	(Rm) → MACL, Rm + 4 → Rm	0100mmmm00010110	1	—
LDS.L	@Rm+,PR	(Rm) → PR, Rm + 4 → Rm	0100mmmm00100110	1	—
NOP		No operation	0000000000001001	1	—
RTE		Delayed branch, SSR/SPC → SR/PC	0000000000101011	4	—
SETS		1 → S	0000000001011000	1	—
SETT		1 → T	0000000000011000	1	1
SLEEP		Sleep	0000000000011011	4*	—
STC	SR,Rn	SR → Rn	0000nnnn00000010	1	—
STC	GBR,Rn	GBR → Rn	0000nnnn00010010	1	—
STC	VBR,Rn	VBR → Rn	0000nnnn00100010	1	—
STC	SSR,Rn	SSR → Rn	0000nnnn00110010	1	—
STC	SPC,Rn	SPC → Rn	0000nnnn01000010	1	—
STC	R0_BANK,Rn	R0_BANK→ Rn	0000nnnn10000010	1	—
STC	R1_BANK,Rn	R1_BANK→ Rn	0000nnnn10010010	1	—
STC	R2_BANK,Rn	R2_BANK→ Rn	0000nnnn10100010	1	—
STC	R3_BANK,Rn	R3_BANK→ Rn	0000nnnn10110010	1	—
STC	R4_BANK,Rn	R4_BANK→ Rn	0000nnnn11000010	1	—
STC	R5_BANK,Rn	R5_BANK→ Rn	0000nnnn11010010	1	—
STC	R6_BANK,Rn	R6_BANK→ Rn	0000nnnn11100010	1	—
STC	R7_BANK,Rn	R7_BANK→ Rn	0000nnnn11110010	1	—
STC.L	SR,@–Rn	Rn–4 → Rn, SR → (Rn)	0100nnnn00000011	2	—
STC.L	GBR,@–Rn	Rn–4 → Rn, GBR → (Rn)	0100nnnn00010011	2	—
STC.L	VBR,@–Rn	Rn–4 → Rn, VBR → (Rn)	0100nnnn00100011	2	—

Table 4.8 System Control Instructions (cont)

Instruction		Operation	Code	Cycles	T Bit
STC.L	SSR,@–Rn	Rn–4 → Rn, SSR → (Rn)	0100nnnn00110011	2	—
STC.L	SPC,@–Rn	Rn–4 → Rn, SPC → (Rn)	0100nnnn01000011	2	—
STC.L	R0_BANK,@–Rn	Rn–4 → Rn, R0_BANK → (Rn)	0100nnnn10000011	2	—
STC.L	R1_BANK,@–Rn	Rn–4 → Rn, R1_BANK → (Rn)	0100nnnn10010011	2	—
STC.L	R2_BANK,@–Rn	Rn–4 → Rn, R2_BANK → (Rn)	0100nnnn10100011	2	—
STC.L	R3_BANK,@–Rn	Rn–4 → Rn, R3_BANK → (Rn)	0100nnnn10110011	2	—
STC.L	R4_BANK,@–Rn	Rn–4 → Rn, R4_BANK → (Rn)	0100nnnn11000011	2	—
STC.L	R5_BANK,@–Rn	Rn–4 → Rn, R5_BANK → (Rn)	0100nnnn11010011	2	—
STC.L	R6_BANK,@–Rn	Rn–4 → Rn, R6_BANK → (Rn)	0100nnnn11100011	2	—
STC.L	R7_BANK,@–Rn	Rn–4 → Rn, R7_BANK → (Rn)	0100nnnn11110011	2	—
STS	MACH,Rn	MACH → Rn	0000nnnn00001010	1	—
STS	MACL,Rn	MACL → Rn	0000nnnn00011010	1	—
STS	PR,Rn	PR → Rn	0000nnnn00101010	1	—
STS.L	MACH,@–Rn	Rn–4 → Rn, MACH → (Rn)	0100nnnn00000010	1	—
STS.L	MACL,@–Rn	Rn–4 → Rn, MACL → (Rn)	0100nnnn00010010	1	—
STS.L	PR,@–Rn	Rn–4 → Rn, PR → (Rn)	0100nnnn00100010	1	—
TRAPA	#imm	PC/SR → SPC/SSR, #imm << 2 → TRA VBR + H'0100 → PC	11000011iiiiiiii	8	—
LDTLB		PTEH/PTEL → TLB	0000000000111000	1	—

Note: *The number of execution states before the chip enters the sleep state.

This table lists the minimum execution cycles. In practice, the number of execution cycles increases when the instruction fetch is in contention with data access or when the destination register of a load instruction (memory → register) is the same as the register used by the next instruction.

4.3 Instructions for DSP Extension

4.3.1 Introduction

New instructions provided are classified in the following 3 groups:

1. Additional system control instructions for CPU unit
2. Single- or double-data transfer between memory and registers in the DSP unit
3. Parallel operation for the DSP unit

1 is provided to support loop control and data transfer between CPU core registers, or memory, and new control registers added to the CPU core. Digital signal processing operations have some levels of nested loop structure. In case of one-level loop, it's good enough to use the 'decrement and test', 'DTRn', and conditional delayed branch, 'BF/S', instructions supported by the SH3. For the nested loop, however, zero overhead loop control capability improves DSP performance.

The CPU core provides some new control registers, RS, RE and MOD, to support loop control and modulo addressing functions. Data transfer instructions between these new control registers and the general registers, or memory, are supported. Moreover, address calculation instructions LDRS and LDRE are added in order to save code size for initial setting of the zero overhead loop control.

The DSP engine provides an independent control register, DSR, and this register is treated as a system register, like MACL and MACH. The A0, X0, X1, Y0 and Y1 registers are also treated as a system register from the CPU side and LDS/STS instructions are supported for the same purpose. Table 4.13 shows the instruction code maps of the new system control instructions for the CPU core.

2 is provided to save program code size of the DSP operations. Data transfer operation without data processing are executed frequently in the DSP engine. In this case, the 32-bit instruction code is redundant and wastes program memory area. All instructions in this class are 16-bit code length, as are the conventional SH-core instructions. Single-data transfer instructions have more-flexible operands than either double-data transfer instructions or the parallel instruction class. Tables 4.9 and 4.10 of section 4.1.3 show the instruction code map of data-transfer instructions for the DSP engine. See section 4.3 for details.

Table 4.9 DSP 16-bit Instruction Code Map for Double-data Transfer

Class	Mnemonic	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X side	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 side	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

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 4.10DSP 16-bit Instruction Code Map for Single-data Transfer

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

(*) System reserved area.

3 is provided to accelerate the digital signal processing operation using the DSP engine. This class of instructions consists of a 32-bit instruction code length, so that it's possible to execute a maximum four instructions, ALU, multiply and two data transfer instructions, in parallel.

This class of instructions consists of two operation fields, A and B, as shown in figure 4.1. The A field specifies data transfer operations, and the B field specifies single or dual data processing operations. These two operations are executed independently, in parallel, so that instructions can also be specified independently. Tables 4.11 and 4.12 of section 4.1.3 show the instruction code map of parallel operation instructions for DSP engine. See section 4.3.4 for details.

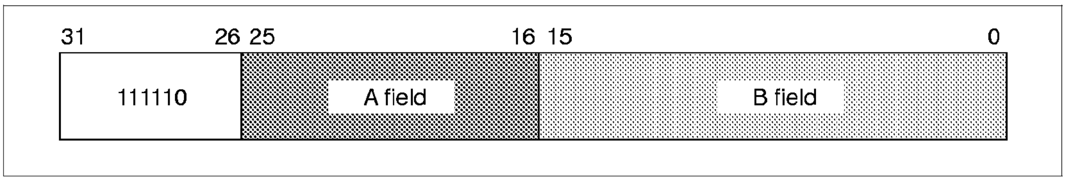


Figure 4.1 Separation of Operation Field

Table 4.11 DSP 32-bit Instruction Code Map for A Field of Parallel Operation

Class	Mnemonic	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S side of data transfer	NOPX	1	1	1	1	1	0	0	▨	0		0	▨	0	0	▨		B field															
	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 side of data transfer	NOPY							▨	0	▨	0	▨	0	▨		0	0	B field															
	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																

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 4.12 DSP 32-bit Instruction Code Map for B Field of Parallel Operation

Class	Mnemonic	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Imm. shift	PSHL #Imm, Dz	1	1	1	1	1	0											0	0	0	0	0	−16 < = Imm < = +16							Dz			
	PSHA #Imm, Dz																	0	0	0	1	0	−32 < = Imm < = +32										
	Vacancy							A field										0	0	0		1											
																		0	0	1													
6 operand parallel operation	PMULS Se, Sf, Dg																	0	1	0	0	Se		Sf		Sx		Sy		Dg		Du	
	Vacancy																	0	1	0	1	0:X0 1:X1 2:Y0 3:A1		0:Y0 1:Y1 2:X0 3:A1		0:X0 1:X1 2:A0 3:A1		0:Y0 1:Y1 2:M0 3:M1		0:M0 1:M1 2:A0 3:A1		0:X0 1:Y0 2:A0 3:A1	
	PSUB Sx, Sy, Du PMULS Se, Sf, Dg																	0	1	1	0												
	PADD Sx, Sy, Du PMULS Se, Sf, Dg																	0	1	1	1												
3 operand operation	Vacancy																	1	0	0	0	0	0	0	0					Dz 0:(*1) 1:(*1) 2:(*1) 3:(*1) 4:(*1) 5:A1 6:(*1) 7:A0 8:X0 9:X1 A:Y0 B:Y1 C:M0 D:(*1) E:M1 F:(*1)			
																				0	1												
	PSUBC Sx, Sy, Dz																			1	0												
	PADDC Sx, Sy, Dz																			1	1												
	PCMP Sx, Sy																			0	0	0	1										
	Vacancy																			0	1												
	Vacancy																			1	0												
	Vacancy																			1	1												
	PABS Sx, Dz																			0	0	1	0										
	PRND Sx, Dz																			0	1												
	PABS Sy, Dz																			1	0												
	PRND Sy, Dz																			1	1												
	Vacancy																			0	0	1	1										
																				0	1												
																				1	0												
																				1	1												

(*1) System reserved area.

Table 4.12 DSP 32-bit Instruction Code Map for B Field of Parallel Operation (cont)

Class	Mnemonic	31	30	29	28	27	26	25 24 23 22 21 20 19 18 17 16	15	14	13	12	11	10	9 8	7 6	5 4	3 2 1 0
Three operation with condition	[if cc] PSHL Sx, Sy, Dz	1	1	1	1	1	0	A field			0	0	0	0	if cc 01: Uncon-ditional 10: DCT 11: DCF	Sx 0:X0 1:X1 2:A0 3:A1	Sy 0:Y0 1:Y1 2:M0 3:M1	Dz 0:(*1) 1:(*1) 2:(*1) 3:(*1) 4:(*1) 5:A1 6:(*1) 7:A0 8:X0 9:X1 A:Y0 B:Y1 C:M0 D:(*1) E:M1 F:(*1)
	[if cc] PSHA Sx, Sy, Dz										0	1						
	[if cc] PSUB Sx, Sy, Dz										1	0						
	[if cc] PADD Sx, Sy, Dz										1	1						
	Vacancy										0	0	0	1				
	[if cc] PAND Sx, Sy, Dz										0	1						
	[if cc] PXOR Sx, Sy, Dz										1	0						
	[if cc] POR Sx, Sy, Dz										1	1						
	[if cc] PDEC Sx, Dz										0	0	1	0				
	[if cc] PINC Sx, Dz										0	1						
	[if cc] PDEC Sy, Dz										1	0						
	[if cc] PINC Sy, Dz										1	1						
	[if cc] PCLR Dz										0	0	1	1				
	[if cc] PDMSB Sx, Dz										0	1						
	Vacancy										1	0						
	[if cc] PDMSB Sy, Dz										1	1						
	[if cc] PNEG Sx, Dz								1	1	0	0	1	0				
	[if cc] PCOPY Sx, Dz										0	1						
	[if cc] PNEG Sy, Dz										1	0						
	[if cc] PCOPY Sy, Dz										1	1						
	Vacancy														0 0			
	[if cc] PSTS MACH, Dz										0	0	1	1	if cc			
	[if cc] PSTS MACL, Dz										0	1						
	[if cc] PLDS Dz, MACH										1	0						
	[if cc] PLDS Dz, MACL										1	1						
	(*2) Vacancy														0 0			
													0	*				
	Vacancy	1	1	1	1	1	1											

(*1) System reserved area.

(*2) [if cc] field can be selected from DCT (DC bit true), DCF (DC bit false) and omit (unconditional).

4.3.2 Additional System Control Instruction for CPU

This class of new instructions is treated as part of the CPU core functions, so that all instructions added here are 16-bit code length. All additional instructions belong to the group of system control instructions. Table 4.13 shows the abstract of additional system instructions. The CPU core is provided with some new control registers, RS, RE and MOD, to support loop control and modulo addressing function. So, LDC and STC types of instructions are provided.

The DSR, A0, X0, X1, Y0 and Y1 registers in the DSP engine are treated as a system register like MACH and MACL. So, STS and LDS instructions are supported for them. Digital signal processing operations have some levels of nested loop structure. So, zero overhead loop control capability improves DSP performance. The SETRC type instructions are provided to set the repeat count to RC, which is located in SR[27:16]. When the immediate operand type of SETRC is executed, eight bits of the immediate operand data is set to SR[23:16] and zeros are set to the rest of bits, SR[27:24]. When the register operand type of SETRC instruction is executed, Rn[11:0] is set to SR[27:16]. The start address and end address of repeat loop are set to the RS and the RE registers. There are two methods for the address setting. One is to use LDC type instructions and the other is to use LDRS and LDRE instructions. See 4.2.2 for details.

Table 4.13Additional System Control Instructions for CPU

Instruction		Operation	Code	Cycles	T Bit
SETRC	#imm	#imm → RC (of SR)	10000010iiiiiiii	3	—
SETRC	Rn	Rn[11:0] → RC (of SR)	0100nnnn00010100	3	—
LDRS	@(disp,PC)	(disp × 2 + PC) → RS	10001100dddddddd	3	—
LDRE	@(disp,PC)	(disp × 2 + PC) → RS	10001110dddddddd	3	—
STC	MOD,Rn	MOD → Rn	0000nnnn01010010	1	—
STC	RS,Rn	RS → Rn	0000nnnn01100010	1	—
STC	RE,Rn	RE → Rn	0000nnnn01110010	1	—
STS	DSR,Rn	DSR → Rn	0000nnnn01101010	1	—
STS	A0,Rn	A0 → Rn	0000nnnn01111010	1	—
STS	X0,Rn	X0 → Rn	0000nnnn10001010	1	—
STS	X1,Rn	X1 → Rn	0000nnnn10011010	1	—
STS	Y0,Rn	Y0 → Rn	0000nnnn10101010	1	—
STS	Y1,Rn	Y1 → Rn	0000nnnn10111010	1	—
STS.L	DSR,@–Rn	Rn – 4 → Rn, DSR → (Rn)	0100nnnn01100010	1	—
STS.L	A0,@–Rn	Rn – 4 → Rn, A0 → (Rn)	0100nnnn01110010	1	—
STS.L	X0,@–Rn	Rn – 4 → Rn, X0 → (Rn)	0100nnnn10000010	1	—
STS.L	X1,@–Rn	Rn – 4 → Rn, X1 → (Rn)	0100nnnn10010010	1	—
STS.L	Y0,@–Rn	Rn – 4 → Rn, Y0 → (Rn)	0100nnnn10100010	1	—
STS.L	Y1,@–Rn	Rn – 4 → Rn, Y1 → (Rn)	0100nnnn10110010	1	—
STC.L	MOD,@–Rn	Rn – 4 → Rn, MOD → (Rn)	0100nnnn01010011	2	—
STC.L	RS,@–Rn	Rn – 4 → Rn, RS → (Rn)	0100nnnn01100011	2	—
STC.L	RE,@–Rn	Rn – 4 → Rn, RE → (Rn)	0100nnnn01110011	2	—
LDS.L	@Rn+,DSR	(Rn) → DSR, Rn + 4 → Rn	0100nnnn01100110	1	—
LDS.L	@Rn+,A0	(Rn) → A0, Rn + 4 → Rn	0100nnnn01110110	1	—
LDS.L	@Rn+,X0	(Rn) → X0, Rn + 4 → Rn	0100nnnn10000110	1	—
LDS.L	@Rn+,X1	(Rn) → X1, Rn + 4 → Rn	0100nnnn10010110	1	—
LDS.L	@Rn+,Y0	(Rn) → Y0, Rn + 4 → Rn	0100nnnn10100110	1	—
LDS.L	@Rn+,Y1	(Rn) → Y1, Rn + 4 → Rn	0100nnnn10110110	1	—
LDC.L	@Rn+,MOD	(Rn) → MOD, Rn + 4 → Rn	0100nnnn01010111	3	—
LDC.L	@Rn+,RS	(Rn) → RS, Rn + 4 → Rn	0100nnnn01100111	3	—
LDC.L	@Rn+,RE	(Rn) → RE, Rn + 4 → Rn	0100nnnn01110111	3	—

Table 4.13Additional System Control Instructions for CPU (cont)

Instruction		Operation	Code	Cycles	T Bit
LDS	Rn,DSR	Rn → DSR	0100nnnn01101010	1	—
LDS	Rn,A0	Rn → A0	0100nnnn01111010	1	—
LDS	Rn,X0	Rn → X0	0100nnnn10001010	1	—
LDS	Rn,X1	Rn → X1	0100nnnn10011010	1	—
LDS	Rn,Y0	Rn → Y0	0100nnnn10101010	1	—
LDS	Rn,Y1	Rn → Y1	0100nnnn10111010	1	—
LDC	Rn,MOD	Rn → MOD	0100nnnn01011110	3	—
LDC	Rn,RS	Rn → RS	0100nnnn01101110	3	—
LDC	Rn,RE	Rn → RE	0100nnnn01111110	3	—

4.3.3 Single- and Double-Data Transfer for DSP Instructions

This class of new instructions is provided to save program code size of the DSP operation. All instructions added here are 16-bit code length. This class of instructions consists of two groups. One is single-data transfer instruction. The other is double-data transfer instruction. In double-transfer instructions, operand flexibility is the same as the data-transfer instruction field, A field, of parallel instruction class, described in section 4.3.4. Conditional load instruction, however, is not available in these 16-bit instructions. In single transfer, Ax pointers and extra two address pointers can be available as pointer operand As, but Ay pointers are not available. Tables 4.14 and 4.15 show the instruction table of single- or double-data transfer instructions. The flexibility of each operand is shown in table 4.16.

In the double-data transfer group, X memory and Y memory can be accessed in parallel. Ax pointers can be used for X memory access instructions only, and Ay pointers can be used for Y memory access instructions only. Double-data transfer operation can access on-chip X and Y memory area only. Single-data transfer operation, using 16-bit instruction code, can access any memory address space.

The Rn, n = 2-7, is usually used as the Ax, Ay and As pointers, but the pointer name itself can be changed with the renaming function on the assembler. The following is recommended:

R2:As2, R3:As3, R4:Ax0 (As0), R5:Ax1 (As1), R6:Ay0, R7:Ay1, R8:Ix, R9:Iy

Table 4.14 Table of Double-data Transfer Instructions

Instruction		Code	Operation	Cycle	DC
X memory no access group	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 no access group	NOPY	111100*0*0*0**00	Y memory no access by Aa pointers	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	—

Table 4.15Table of Single-data Transfer Instructions

Instruction	Code	Operation	Cycle	DC
`MOVS.W @-As,Ds`	111101AADDDD0000	As – 2 → As, (As) → MSW of Ds, 0 → LSW of Dd	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 Dd, As + 2 → As	1	—
`MOVS.W @As+Ix,Ds`	111101AADDDD1100	(Asc) → MSW of Ds, 0 → LSW of Dd, As + Ix → As	1	—
`MOVS.W Ds,@-As` *	111101AADDDD0001	As – 2 → As, MSW of Ds → (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 → As	1	—
`MOVS.L Ds,@As+Ix`	111101AADDDD1111	Ds → (As), As + Ix → As	1	—

Note: * When one of the guard-bit register, A0G and A1G, is specified as the source operand Ds, the data is output on LDB [7:0] and the signed bit is copied to the upper bits [31:8].

Table 4.16Operand Flexibility of Data Transfer Instructions

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

4.3.4 Parallel Operation for the DSP Unit

This class of new instructions is provided to accelerate digital signal processing operations using the DSP engine. This class of instructions has 32-bit code length in order to execute multiple operations in parallel. The instruction word consists of two instruction fields, A and B, as described in section 4.3. The A field specifies double-data transfer instructions, and the B field specifies single- or double-data processing instructions. These two instructions are executed independently in parallel, so that instructions can also be specified independently. The function of A field, data-transfer instruction field, is basically the same as the double-data transfer instruction of the previous section, 4.3.3, but load operation has a special function.

There are three kinds of instruction groups for B field, a double-data processing group, a single-data processing with condition group, and an unconditional single-data processing group. Each operand can be selected independently from data registers of the DSP engine. The flexibility of each operand is shown in the following tables 4.17 and 4.18.

The order of instruction description is B-field instruction first, and then, A-field instruction, from left side. Figure 4.2 shows some examples of mnemonics of this class.

Table 4.17Operand Symbol of Each Group

Group	Class	Operand Symbol	
Field B	Double data processing group		ALUop. Sx, Sy, Du MLTop. Se, Df, Dg
	Single data processing with condition group		ALUop. Sx, Sy, Dz
		DCT	ALUop. Sx, Sy, Dz
		DCF	ALUop. Sx, Sy, Dz
			ALUop. Sx, Dz
		DCT	ALUop. Sx, Dz
		DCF	ALUop. Sx, Dz
			ALUop. Sy, Dz
		DCT	ALUop. Sy, Dz
		DCF	ALUop. Sy, Dz
	Unconditional single data processing group		ALUop. Sx, Sy, Dz
			ALUop. Sx, Dz
			ALUop. Sy, Dz
			MLTop. Se, Sf, Dg

Table 4.18Operand Flexibility of Parallel Instructions

	ALU, BPU 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	

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

Figure 4.2 Examples of Parallel Instruction Program

Here, [ ] means that this part can be omitted.

No operation instructions, NOPX and NOPY can be omitted. Tables 4.19 to 4.22 shows the abstract of B field of parallel operation instructions. See sections 4.3 and 4.4.3 for A field of operations.

The symbol ‘;’ is used to separate instruction lines, but it can be omitted. When this separator ‘;’ is used, the space following can be used as a comment area. It has the same function as conventional SH tools.

Condition code bit (DC) in the DSR register is always updated based upon the result of the ALU or shift operation, if it is unconditional. If it is conditional instruction, then it does not update the DC bit. Multiply operation doesn’t affect the DC bit. Update of the DC bit depends on the states of CS0-2 bits in the DSR register. Table 4.23 shows the definition of the DC-bit update rule.

Table 4.19Table of B Field of Parallel Operation Instructions (1)

Instruction	Code	Operation	Cycle	DC
PMULS Se,Sf,Dg	111110********** 0100eeff0000gg00	Se * Sf → Dg (Signed)	1	—
PADD Sx,Sy,Du PMULS Se,Sf,Dg	111110********** 0111eeffxxyygguu	Sx + Sy → Du Se * Sf → Dg (Signed)	1	*
PSUB Sx,Sy,Du PMULS Se,Sf,Dg	111110********** 0110eeffxxyygguu	Sy – Sy → Du Se * Sf → Dg (Signed)	1	*
PADD Sx,Sy,Dz	111110********** 10110001xxyyzzzz	Sx + Sy → Dz	1	*
DCT PADD Sx,Sy,Dz	111110********** 10110010xxyyzzzz	If DC = 1, Sx + Sy → Dz If DC = 0, nop	1	*
DCF PADD Sx,Sy,Dz	111110********** 10110011xxyyzzzz	If DC = 0, Sx + Sy → Dz If DC = 1, nop	1	*
PSUB Sx,Sy,Dz	111110********** 10100001xxyyzzzz	Sx – Sy →Dz	1	*
DCT PSUB Sx,Sy,Dz	111110********** 10100010xxyyzzzz	If DC = 1, Sx – Sy → Dz If DC = 0, nop	1	*
DCF PSUB Sx,Sy,Dz	111110********** 10100011xxyyzzzz	If DC = 0, Sx – Sy → Dz If DC = 1, nop	1	*
PSHA Sx,Sy,Dz	111110********** 1010001xxyyzzzz	If Sy >= 0, Sx << Sy → Dz (arith. shift) If Sy < 0, Sx >> Sy → Dz	1	*
DCT PSHA Sx,Sy,Dz	111110********** 10010010xxyyzzzz	If DC = 1 & Sy >= 0, Sx << Sy → Dz (arith. shift) If DC = 1 & Sy < 0, Sx >> Sy → Dz If DC = 0, nop	1	*
DCF PSHA Sx,Sy,Dz	111110********** 10010011xxyyzzzz	If DC = 0 & Sy >= 0, Sx << Sy → Dz (arith. 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 (log. shift) If Sy < 0, Sx >> Sy → Dz	1	*

Note: * See table 4.23.

Table 4.19Table of B Field of Parallel Operation Instructions (1)

Instruction	Code	Operation	Cycle	DC
DCT PSHL Sx,Sy,Dz	111110********** 10000010xxyyzzzz	If DC = 1 & Sy >= 0, Sx << Sy → Dz (log. 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 (log. 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 Sx,Dz	111110********** 1111101100yyzzzz	If DC = 0, Sy → Dz If DC = 1, nop	1	*

Note: * See table 4.23.

Table 4.20Table of B Field of Parallel Operation Instructions (2)

Instruction	Code	Operation	Cycle	DC
PDMSB Sx,Dz	111110********** 10011101xx00zzzz	Count shift value for normalizing Sx → Dz	1	*
PDMSB Sy,Dz	111110********** 1011110100yyzzzz	Count shift value for normalizing Sy → Dz	1	*
DCT PDMSB Sx,Dz	111110********** 10011110xx00zzzz	If DC = 1, Count shift value for normalizing Sx → Dz If DC = 0, nop	1	*
DCT PDMSB Sy,Dz	111110********** 1011111000yyzzzz	If DC = 1, Count shift value for normalizing Sy → Dz If DC = 0, nop	1	*
DCF PDMSB Sx,Dz	111110********** 10011111xx00zzzz	If DC = 0, Count shift value for normalizing Sx → Dz If DC = 1, nop	1	*
DCF PDMSB Sy,Dz	111110********** 1011111100yyzzzz	If DC = 0, Count shift value for normalizing 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	*

Note: * See table 4.23.

Table 4.20Table of B Field of Parallel Operation Instructions (2) (cont)

Instruction	Code	Operation	Cycle	DC
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	*

Note: * See table 4.23.

Table 4.21Table of B Field of Parallel Operation Instructions (3)

Instruction	Code	Operation	Cycle	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 = 1, Sx ^ Sy → Dz If DC = 0, nop	1	*

Table 4.21Table of B Field of Parallel Operation Instructions (3)

Instruction	Code	Operation	Cycle	DC
PDEC Sx,Dz	111110********** 10001001xx00zzzz	Sx [39:16] – 1 → Dz	1	*
PDEC Sy,Dz	111110********** 1010100100yyzzzz	Sy [31:16] – 1 → Dz	1	*
DCT PDEC Sx,Dz	111110********** 10001010xx00zzzz	If DC = 1, Sx [39:16] – 1 → Dz If DC=0, nop	1	*
DCT PDEC Sy,Dz	111110********** 1010101000yyzzzz	If DC = 1, Sy [31:16] – 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	*
DCF PDEC Sy,Dz	111110********** 1010101100yyzzzz	If DC = 0, Sy [31:16] – 1 → Dz If DC = 1, nop	1	*
PCLR Dz	111110********** 100011010000zzzz	h'00000000 → Dz	1	*
DCT PCLR Dz	111110********** 100011100000zzzz	If DC = 1, h'00000000 → Dz If DC = 0, nop	1	*
DCF PCLR Dz	111110********** 100011110000zzzz	If DC = 0, h'00000000 → Dz If DC = 1, nop	1	*
PSHA #Imm,Dz	111110********** 00010iiiiiiizzzz	If Imm >= 0, Dz << Imm → (arith. shift) If Imm < 0, Dz >> Imm → Dz	1	*
PSHL #Imm,Dz	111110********** 00000iiiiiiizzzz	If Imm >= 0, Dz << Imm → Dz (logical shift) If Imm < 0, Dz >> Imm → Dz	1	*

Note: * See table 4.23.

Table 4.22Table of B Field of Parallel Operation Instructions (4)

Instruction	Code	Operation	Cycle	DC
PSTS MACH,Dz	111110********** 110011010000zzzz	MACH → Dz	1	—
DCT PSTS MACH,Dz	111110********** 110011100000zzzz	If DC = 1, MACH → Dz	1	—
DCF PSTS MACH,Dz	111110********** 110011110000zzzz	If DC = 0, MACH → Dz	1	—
PSTS MACL,Dz	111110********** 110111010000zzzz	MACL → Dz	1	—
DCT PSTS MACL,Dz	111110********** 110111100000zzzz	If DC = 1, MACL → Dz	1	—
DCF PSTS MACL,Dz	111110********** 110111110000zzzz	If DC = 0, MACL → Dz	1	—
PLDS Dz,MACH	111110********** 111011010000zzzz	Dz → MACH	1	—
DCT PLDS Dz,MACH	111110********** 111011100000zzzz	If DC = 1, Dz → MACH	1	—
DCF PLDS Dz,MACH	111110********** 111011110000zzzz	If DC = 0, Dz → MACH	1	—
PLDS Dz,MACL	111110********** 111111010000zzzz	Dz → MACL	1	—
DCT PLDS Dz,MACL	111110********** 111111100000zzzz	If DC = 1, Dz → MACL	1	—
DCF PLDS Dz,MACL	111110********** 111111110000zzzz	If DC = 0, Dz → MACL	1	—
PADDC Sx,Sy,Dz	111110********** 10110000xxyyzzzz	Sx + Sy + DC → Dz Carry → DC	1	Carry
PSUBC Sx,Sy,Dz	111110********** 10100000xxyyzzzz	Sx – Sy – DC → Dz Borrow → DC	1	Borrow
PCMP Sx,Sy	111110********** 10000100xxyy0000	Sx – Sy → Update DC*	1	*

Note: * See table 4.23.

Table 4.22Table of B Field of Parallel Operation Instructions (4)

Instruction	Code	Operation	Cycle	DC
PABS Sx,Dz	111110**********	If Sx < 0, 0 – Sx → Dz	1	*
	10001000xx00zzzz	If Sx >= 0, nop		
PABS Sy,Dz	111110**********	If Sy < 0, 0 – Sy → Dz	1	*
	1010100000yyzzzz	If Sx >= 0, nop		
PRND Sx,Dz	111110**********	Sx + h'00008000 → Dz	1	*
	10011000xx00zzzz	LSW of Dz → h'0000		
PRND Sy,Dz	111110**********	Sy + h'00008000 → Dz	1	*
	1011100000yyzzzz	LSW of Dz → h'0000		

Note: * See table 4.23.

Table 4.23Definition of DC-Bit Update Rule

CS [2:0]				
2	**1**	**0**	**Condition Mode**	**Explanation**
0	0	0	Carry or borrow mode	When carry or borrow is generated as the result of an ALU arithmetic operation, DC bit is set. Otherwise cleared. When a shift operation, PSHA or PSHL, is executed, the last shifted out bit data is copied into DC bit. When an ALU logical operation is executed, DC bit is always cleared.
0	0	1	Negative value mode	When an arithmetic operation of ALU or shift (PSHA) is executed, the MSB of the result, including guard bit part, is copied into DC bit. When a logical operation of ALU or shift (PSHL) is executed, the MSB of the result, excluding guard bit part, is copied into DC bit.
0	1	0	Zero value mode	When the result of ALU or shift operation is all zeros, DC bit is set. Otherwise cleared.
0	1	1	Overflow mode	When an arithmetic operation of ALU or shift (PSHA) yields a result beyond the range of the destination register, except for guard bit part, DC bit is set. Otherwise cleared. When a logical operation of ALU or shift (PSHL) is executed, DC bit is always cleared.

1	0	0	Signed greater than mode	This mode is similar to the signed greater than or equal mode but DC is cleared when the result is all zeros.
				DC = ~ {(Negative ^ Overflow) | Zero} ; Arithmetic operation cases
				DC = 0 ; Logical operation cases

1	0	1	Signed greater than or equal mode	In the case that the result of an arithmetic operation of ALU or shift (PSHA) is not overflow, the definition is opposite of negative value mode. In the case of overflow result, the definition is the same as negative value mode.
				When a logical operation of ALU or shift (PSHL) is executed, the DC bit is always cleared.
				DC = ~ {(Negative ^ Overflow) | Zero} ; Arithmetic operation cases
				DC = 0 ; Logical operation cases

1	1	0	Reserved	
1	1	1	Reserved	

Conditional Operation and Data Transfer: Some operations belonging to this class can execute with condition as described before. However, note that the specified condition is effective for B field of instruction only, not effective for any data transfer instructions specified in parallel. Figure 4.3 shows an example.

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

Condition True Case
Before execution:  X0=H©33333333, Y0=H©55555555, A0=H©123456789A,
                   R4=H©00008000, R6=H©00008233, R9=H©00000004
                   (R4)=H©1111, (R6)=H©2222
After execution:   X0=H©11110000, Y0=H©55555555, A0=H©0088888888,
                   R4=H©00008002, R6=H©00008237, R9=H©00000004
                   (R4)=H©1111, (R6)=H©3456

Condition False Case
Before execution:  X0=H©33333333, Y0=H©55555555, A0=H©123456789A,
                   R4=H©00008000, R6=H©00008233, R9=H©00000004
After execution:   (R4)=H©1111, (R6)=H©2222
                   X0=H©11110000, Y0=H©55555555, A0=H©123456789A,
                   R4=H©00008002, R6=H©00008237, R9=H©00000004
                   (R4)=H©1111, (R6)=H©3456
```

Figure 4.3 Example of Data Transfer Instruction with Condition

Instruction Code Assignment of NOPX and NOPY: There are several cases for instruction code assignment when a certain operation is unnecessary for this parallel operation. Table 4.24 shows the instruction code definition of some special cases.

Table 4.24Instruction Code Definition of Special Cases

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		Same as above
PADD X0,Y0,A0			Same as above
	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	Same as above
	NOPX	NOPY	1111000000000000
NOP			0000000000001001

Section 5 Memory Management Unit (MMU)

5.1 Overview

5.1.1 Features

The SH7729 has an on-chip memory management unit (MMU) that implements address translation. The SH7729 features a resident translation look-aside buffer (TLB) that caches information for user-created address translation tables located in external memory. It enables high-speed translation of virtual addresses into physical addresses. Address translation uses the paging system and supports two page sizes (1 Kbytes and 4 Kbytes). The access right to virtual address space can be set for privileged and user modes to provide memory protection.

5.1.2 Role of MMU

The MMU is a feature designed to make efficient use of physical memory. As shown in figure 5.1, if a process is smaller in size than the physical memory, the entire process can be mapped onto physical memory. However, if the process increases in size to the extent that it no longer fits into physical memory, it becomes necessary to partition the process and to map those parts requiring execution onto memory as occasion demands ((1)). Having the process itself consider this mapping onto physical memory would impose a large burden on the process. To lighten this burden, the idea of virtual memory was born as a means of performing en bloc mapping onto physical memory ((2)). In a virtual memory system, substantially more virtual memory than physical memory is provided, and the process is mapped onto this virtual memory. Thus a process only has to consider operation in virtual memory. Mapping from virtual memory to physical memory is handled by the MMU. The MMU is normally controlled by the operating system, switching physical memory to allow the virtual memory required by a process to be mapped onto physical memory in a smooth fashion. Switching of physical memory is carried out via secondary storage, etc.

The virtual memory system that came into being in this way is particularly effective in a time-sharing system (TSS) in which a number of processes are running simultaneously ((3)). If processes running in a TSS had to take mapping onto virtual memory into consideration while running, it would not be possible to increase efficiency. Virtual memory is thus used to reduce this load on the individual processes and so improve efficiency ((4)). In the virtual memory system, virtual memory is allocated to each process. The task of the MMU is to perform efficient mapping of these virtual memory areas onto physical memory. It also has a memory protection feature that prevents one process from inadvertently accessing another process's physical memory.

When address translation from virtual memory to physical memory is performed using the MMU, it may occur that the relevant translation information is not recorded in the MMU, with the result that one process may inadvertently access the virtual memory allocated to another process. In this case, the MMU will generate an exception, change the physical memory mapping, and record the new address translation information.

Although the functions of the MMU could also be implemented by software alone, the need for translation to be performed by software each time a process accesses physical memory would result in poor efficiency. For this reason, a buffer for address translation (translation look-aside buffer: TLB) is provided in hardware to hold frequently used address translation information. The TLB can be described as a cache for storing address translation information. Unlike cache memory, however, if address translation fails, that is, if an exception is generated, switching of address translation information is normally performed by software. This makes it possible for memory management to be performed flexibly by software.

The MMU has two methods of mapping from virtual memory to physical memory: a paging method using fixed-length address translation, and a segment method using variable-length address translation. With the paging method, the unit of translation is a fixed-size address space (usually of 1 to 64 Kbytes) called a page.

In the following text, the SH7729 address space in virtual memory is referred to as virtual address space, and address space in physical memory as physical memory space.

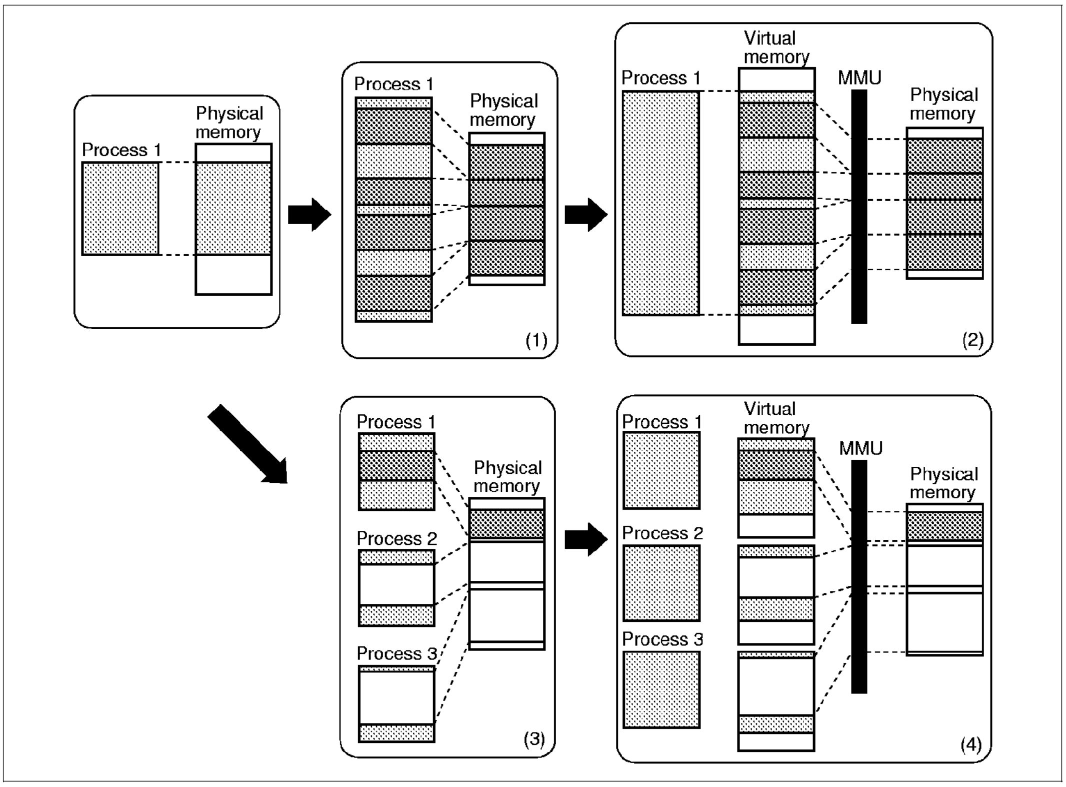


Figure 5.1 MMU Functions

5.1.3 Virtual Address Space

Virtual Address Map: The SH7729 uses 32-bit virtual addresses to access a 4-Gbyte virtual address space that is divided into several areas. Address space mapping is shown in figure 5.2.

In the privileged mode, there are five areas, P0–P4. The P0 and P3 areas are mapped onto physical address space in page units, in accordance with address translation table information. Addresses H'7F000000–H'7FFFFFFF in the P0 area can be used as on-chip RAM space by making a setting in the cache control register (CCR) (see section 5, Cache). In this case, mapping by means of the address translation table is not performed for the on-chip RAM space. Write-back or write-through can be selected for write access by means of a CCR setting.

Mapping of the P1 area is fixed to physical address space (H'00000000 to H'1FFFFFFF). In the P1 area, setting a virtual address MSBs (bit 31) to 0 generates the corresponding physical address. P1 area access can be cached, and the cache control register (CCR) is set to indicate whether to cache or not. Write access is processed as write-through.

Mapping of the P2 area is fixed to physical address space (H'00000000 to H'1FFFFFFF). In the P2 area, setting the top three virtual address bits (bits 31, 30, and 29) to 0 generates the corresponding physical address. P2 area access cannot be cached.

The P1 and P2 areas are not mapped by the address translation table, so the TLB is not used and no exceptions like TLB misses occur. Initialization of MMU-related registers, exception processing, and the like are located in the P1 and P2 areas. Because the P1 area is cached, handlers that require high-speed processing are placed there.

A part of the control register in the peripheral module is allocated in area 1 of the physical address space. When the physical address space is not used for address translation, allocate that part of the control register in the P2 area. When the physical address space is used for address translation, set no caching.

The P4 area is used for mapping on-chip control register addresses.

In the user mode, 2 Gbytes of the virtual address space from H'00000000 to H'7FFFFFFF (area U0) can be accessed. U0 is mapped onto physical address space in page units. Write-back or write-through mode can be selected for write accesses by means of a CCR setting. When SR.DSP is off, 2 Gbytes of the virtual address space from H'80000000 to H'FFFFFFFF cannot be accessed in the user mode. Attempting to do so creates an address error. When the SR.DSP is on, a new 16-MB address space, Uxy, is defined from address H'A500 0000 to H'A5FF FFFF for X/Y RAM. This Uxy space is non-cached, fixed physical address space. Any access to address space beyond U0 and Uxy creates an address error. For details of the X/Y RAM space, refer to section 8, X/Y RAM.

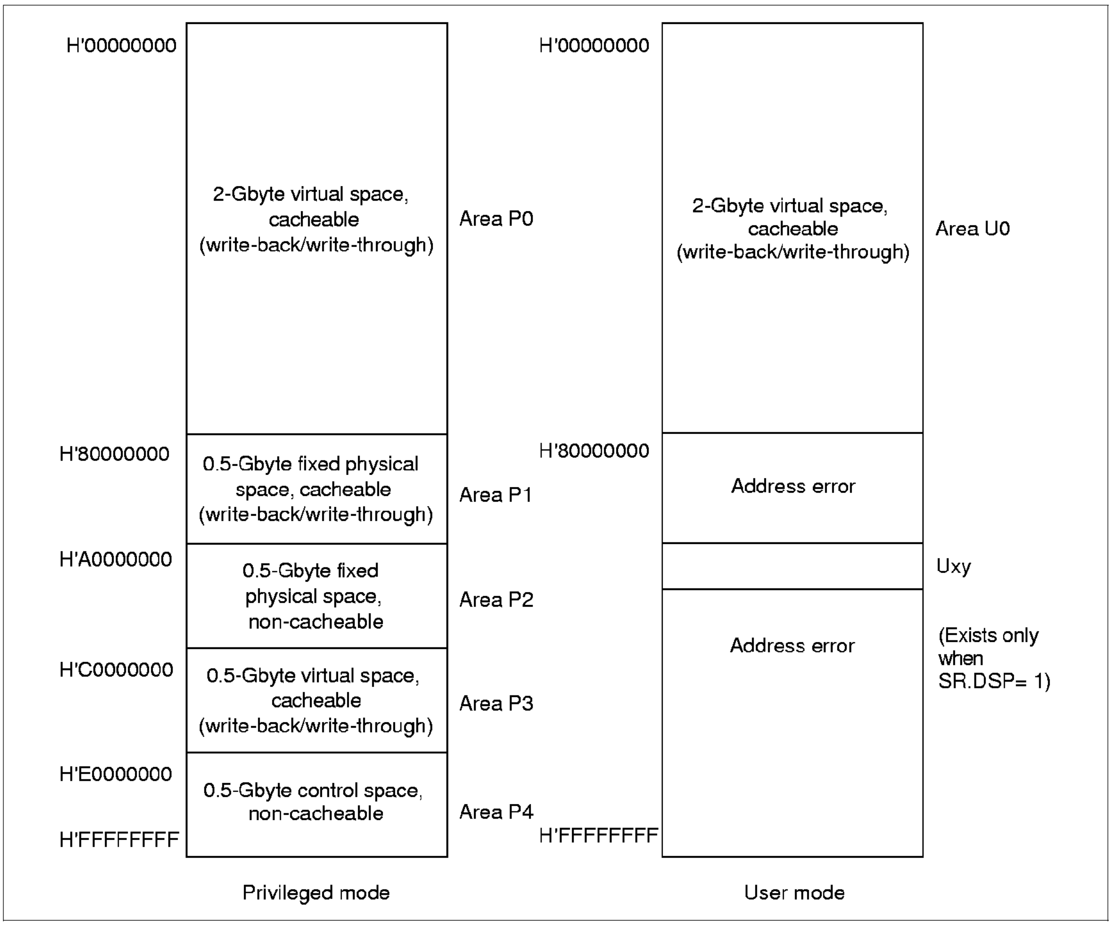

H'00000000
2-Gbyte virtual space, cacheable (write-back/write-through)
Area P0
H'80000000
0.5-Gbyte fixed physical space, cacheable (write-back/write-through)
Area P1
H'A0000000
0.5-Gbyte fixed physical space, non-cacheable
Area P2
H'C0000000
0.5-Gbyte virtual space, cacheable (write-back/write-through)
Area P3
H'E0000000
0.5-Gbyte control space, non-cacheable
Area P4
H'FFFFFFFF
Privileged mode
H'00000000
2-Gbyte virtual space, cacheable (write-back/write-through)
Area U0
H'80000000
Address error
Uxy
Address error
(Exists only when SR.DSP= 1)
H'FFFFFFFF
User mode

Figure 5.2 Virtual Address Space Mapping

Physical Address Space: The SH7729 supports a 32-bit physical address space, but the upper 3 bits are actually ignored and treated as a shadow. See section 10, Bus State Controller, for details.

Single Address Translation: When the MMU is enabled, the virtual address space is divided into units called pages. Physical addresses are translated in page units. Address translation tables in external memory hold information such as the physical address that corresponds to the virtual address and memory protection codes. When an access to areas P1 or P2 occurs, there is no TLB access and the physical address is defined uniquely by hardware. If it belongs to area P0, P3 or U0, the TLB is searched by virtual address and, if that virtual address is registered in the TLB, the access hits the TLB. The corresponding physical address and the page control information are read from the TLB and the physical address is determined.

If the virtual address is not registered in the TLB, a TLB miss exception occurs and processing will shift to the TLB miss handler. In the TLB miss handler, the TLB address translation table in external memory is searched and the corresponding physical address and the page control information are registered in the TLB. After returning from the handler, the instruction that caused the TLB miss is re-executed. When the MMU is enabled, address translation information that results in a physical address space of H'80000000–H'FFFFFFFF should not be registered in the TLB.

When the MMU is disabled, the virtual address is used directly as the physical address. As the SH7729 supports a 29-bit address space as the physical address space, the top 3 bits of the physical address are ignored, and constitute a shadow space (see section 10, Bus State Controller (BSC)). For example, addresses H'00001000 in the P0 area, H'80001000 in the P1 area, H'A0001000 in the P2 area, and H'C0001000 in the P3 area are all mapped onto the same physical address. When access to these addresses is performed with the cache enabled, an address with the top 3 bits of the physical address masked to 0 is stored in the cache address array to ensure data congruity.

Single Virtual Memory Mode and Multiple Virtual Memory Mode: There are two virtual memory modes: single virtual memory mode and multiple virtual memory mode. In single virtual memory mode, multiple processes run in parallel using the virtual address space exclusively and the physical address corresponding to a given virtual address is specified uniquely. In multiple virtual memory mode, multiple processes run in parallel sharing the virtual address space, so a given virtual address may be translated into different physical addresses depending on the process. By the value set to the MMU control register, either single or multiple virtual mode is selected.

Address Space Identifier (ASID): When multiple processes run in parallel sharing the same virtual address space and the processes have unique address translation tables, the virtual space can be multiplexed. When this is done, a given virtual address may be mapped to a different physical address depending on the process. This means that virtual addresses are expanded by using an address space identifier (ASID) and virtual addresses can be distinguished by ASID. The ASID is 8 bits in length and is held in PTEH within the MMU indicating the current process. With ASID, the TLB need not be purged when the process is switched.

When multiple processes run in parallel using the virtual address space exclusively, the physical address corresponding to a given virtual address is specified uniquely. For this kind of single virtual memory, the ASID becomes a key to protect memory (see section 5.4.2, MMU Software Management).

5.1.4 Register Configuration

A register that has an undefined initial value must be initialized by software. Table 5.1 shows the configuration of the MMU control registers.

Table 5.1 Register Configuration

Name	Abbreviation	R/W	Size	Initial Value*1	Address
Page table entry register high	PTEH	R/W	Longword	Undefined	H'FFFFFFF0
Page table entry register low	PTEL	R/W	Longword	Undefined	H'FFFFFFF4
Translation table base register	TTB	R/W	Longword	Undefined	H'FFFFFFF8
TLB exception address register	TEA	R/W	Longword	Undefined	H'FFFFFFFC
MMU control register	MMUCR	R/W	Longword	*2	H'FFFFFFE0

Notes: 1. Initialized by a power-on reset or manual reset.

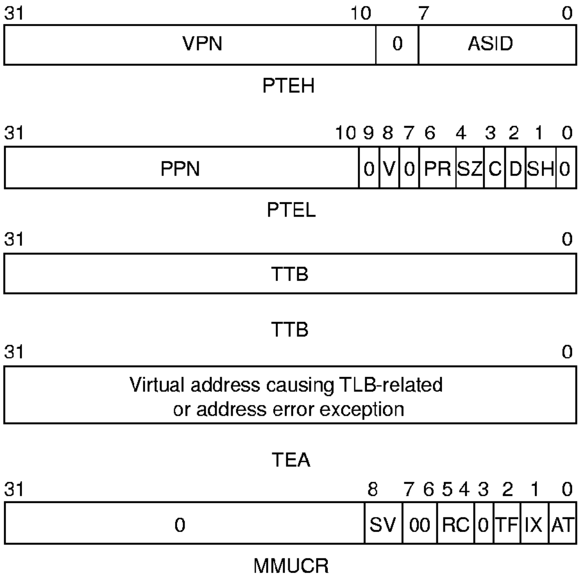
2. SV bit: undefined
Other bits: 0

5.2 Register Description

There are five registers for MMU processing. These are all peripheral module registers, so they are located in address space area P4 and can only be accessed from privileged mode by specifying the address. These registers consist of:

1. The page table entry register high (PTEH) register residing at address H'FFFFFFF0, which consists of a virtual page number (VPN) and ASID. The VPN set is the VPN of the virtual address at which the exception is generated in case of an MMU exception or address error exception. When the page size is 4 Kbytes, the VPN is the upper 20 bits of the virtual address, but in this case the upper 22 bits of the virtual address are set. The VPN can also be modified by software. As the ASID, software sets the number of the currently executing process. The VPN and ASID are recorded in the TLB by the LDTLB instruction.
2. The page table entry register low (PTEL) register residing at address H'FFFFFFF4, and used to store the physical page number and page management information to be recorded in the TLB by the LDTLB instruction. The contents of this register are only modified in response to a software command.
3. The translation table base register (TTB) residing at address H'FFFFFFF8, which points to the base address of the current page table. The hardware does not set any value in TTB automatically. TTB is available to software for general purposes.
4. The TLB exception address register (TEA) residing at address H'FFFFFFFC, which stores the virtual address corresponding to a TLB or address error exception. This value remains valid until the next exception or interrupt.
5. The MMU control register (MMUCR) residing at address H'FFFFFFE0, which makes the MMU settings described in figure 5.3. Any program that modifies MMUCR should reside in the P1 or P2 area.

The MMU registers are shown in figure 5.3.



0: Reserved bits. Always read as 0. Writing is ignored. However, 0 should also be specified in a write to MMUCR only.

SV: Single virtual memory mode bit. Set to 1 for the single virtual memory mode, cleared to 0 for the multiple virtual memory mode.

RC: A 2-bit random counter, automatically updated by hardware according to the following rules in the event of an MMU exception. When a TLB miss exception occurs, all TLB entry ways corresponding to the virtual address at which the exception occurred are checked, and if all ways are valid, 1 is added to RC; if there is one or more invalid way, they are set by priority from way 0, in the order: way 0, way 1, way 2, way 3. In the event of an MMU exception other than a TLB miss exception, the way which caused the exception is set in RC.

TF: TLB flush bit. Write 1 to flush the TLB (clear all valid bits of the TLB to 0). Always reads 0.

IX: Index mode bit. When 0, VPN bits 16–12 are used as the TLB index number. When 1, the value obtained by EX-ORing ASID bits 4–0 in PTEH and VPN bits 16–12 are used as the TLB index number.

AT: Address translation bit. Enables/disables the MMU.
0: MMU disabled
1: MMU enabled

Figure 5.3 MMU Register Contents

5.3 TLB Functions

5.3.1 Configuration of the TLB

The TLB caches address translation table information located in the external memory. The address translation table stores the physical page number translated from the virtual page number and the control information for the page, which is the unit of address translation. Figure 5.4 shows the overall TLB configuration. The TLB is 4-way set associative with 128 entries. There are 32 entries for each way. Figure 5.5 shows the configuration of virtual addresses and TLB entries.

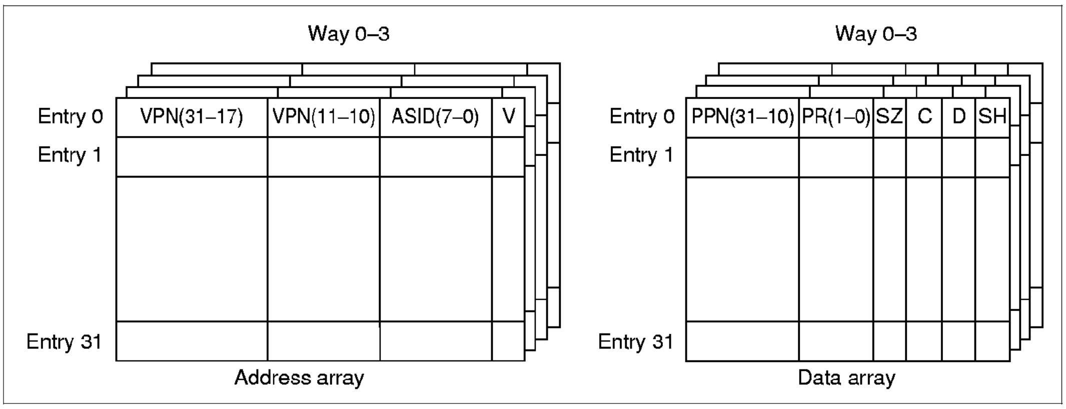
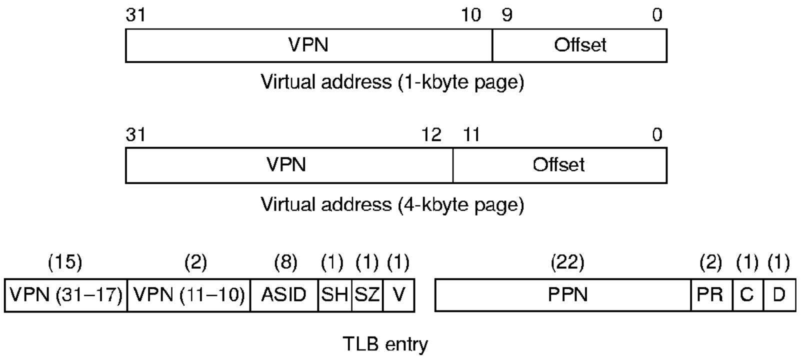


Figure 5.4 Overall Configuration of the TLB



VPN: Virtual page number. Top 22 bits of virtual address for a 1-kbyte page, or top 20 bits of virtual address for a 4-kbyte page. Since VPN bits 16-12 are used as the index number, they are not stored in the TLB entry.

ASID: Address space identifier. Indicates the process that can access a virtual page. In single virtual memory mode and user mode, or in multiple virtual memory mode, if the SH bit is 0, the address is compared with the ASID in PTEH when address comparison is performed.

SH: Share status bit
0 = Page not shared between processes
1 = Page shared between processes

SZ: Page-size bit
0 = 1-kbyte page
1 = 4-kbyte page

V: Valid bit. Indicates whether entry is valid.
0 = Invalid
1 = Valid
Cleared to 0 by a power-on reset. Not affected by a manual reset.

PPN: Physical page number. Top 22 bits of physical address. PPN bits 11-10 are not used in case of a 4-kbyte page. Attention must be paid to the synonym problem in case of a 1-kbyte page (see section 3.4.4).

PR: Set the most significant bit to 0.
Protection key field. 2-bit field encoded to define the access rights to the page.
00: Reading only is possible in privileged mode.
01: Reading/writing is possible in privileged mode.
10: Reading only is possible in privileged/user mode.
11: Reading/writing is possible in privileged/user mode.

C: Cacheable bit. Indicates whether the page is cacheable.
0: Non-cacheable
1: Cacheable

D: Dirty bit. Indicates whether the page has been written to.
0 = Not written to
1 = Written to

Figure 5.5 Virtual Address and TLB Structure

5.3.2 TLB Indexing

The TLB uses a 4-way set associative scheme, so entries must be selected by index. VPN bits 16 to 12 are used as the index number regardless of the page size. The index number can be generated in two different ways depending on the setting of the IX bit in MMUCR.

1. When IX = 0, VPN bits 16–12 alone are used as the index number
2. When IX = 1, VPN bits 16–12 are EX-ORed with ASID bits 4–0 to generate a 5-bit index number

The second method is used to prevent lowered TLB efficiency that results when multiple processes run simultaneously in the same virtual address space and a specific entry is selected by indexing of each process. Figures 5.6 and 5.7 show the indexing schemes.

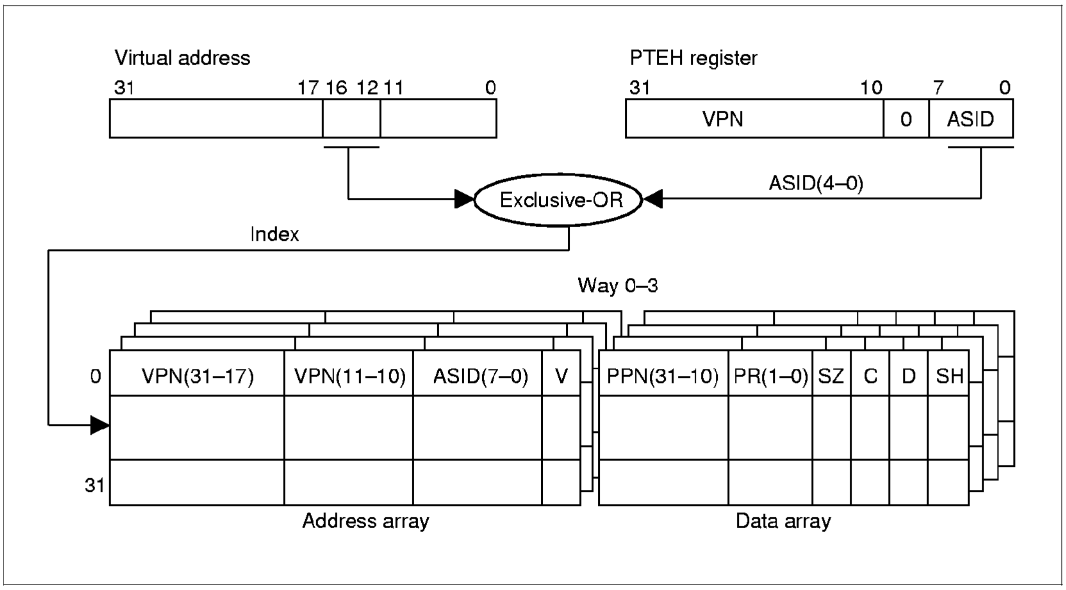


Figure 5.6 TLB Indexing (IX = 1)

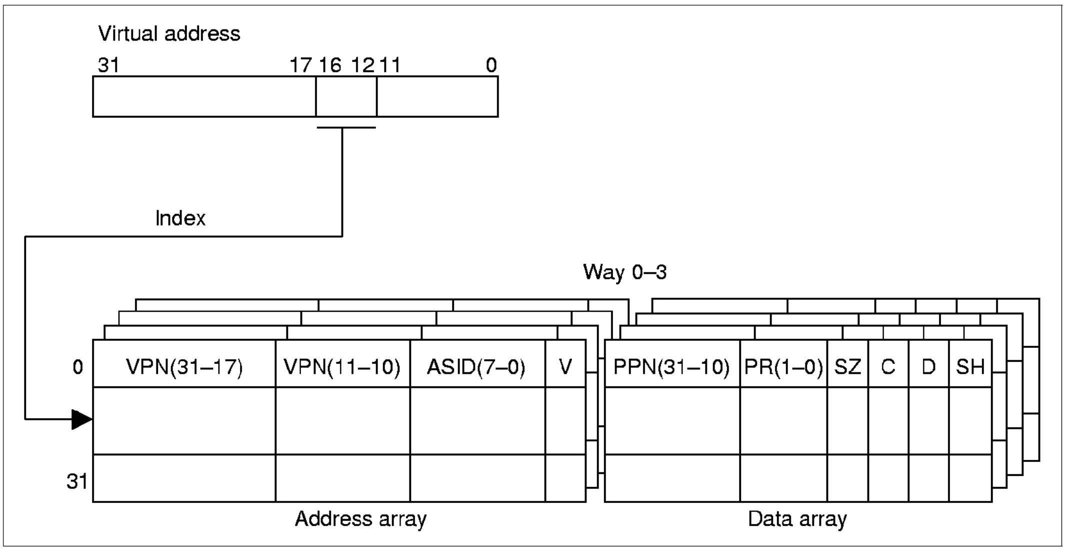


Figure 5.7 TLB Indexing (IX = 0)

5.3.3 TLB Address Comparison

The results of address comparison determine whether a specific virtual page number is registered in the TLB. The virtual page number of the virtual address that accesses external memory is compared to the virtual page number of the indexed TLB entry. The ASID within the PTEH is compared to the ASID of the indexed TLB entry. All four ways are searched simultaneously. If the compared values match, and the indexed TLB entry is valid (V bit = 1), the hit is registered.

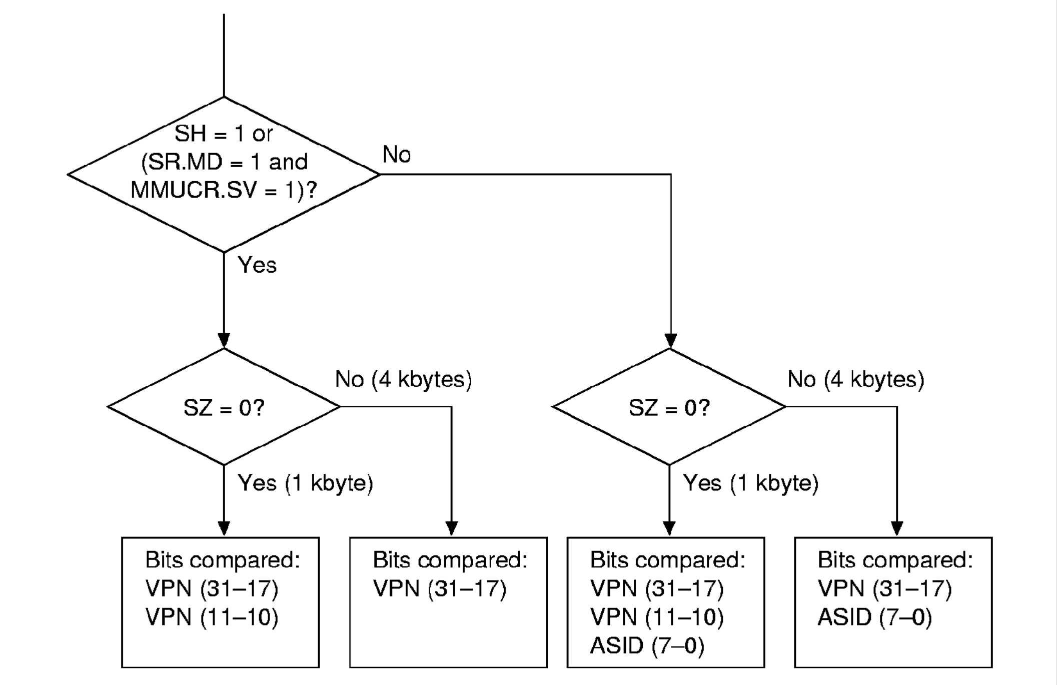
It is necessary to have software ensure that TLB hits do not occur simultaneously in more than one way, as hardware operation is not guaranteed if this occurs. For example, if there are two identical TLB entries with the same VPN and a setting is made such that a TLB hit is made only by a process with ASID = H'FF when one is in the shared state (SH = 1) and the other in the non-shared state (SH = 0), then if the ASID in PTEH is set to H'FF, there is a possibility of simultaneous TLB hits in both these ways. It is therefore necessary to ensure that this kind of setting is not made by software.

The object compared varies depending on the page management information (SZ, SH) in the TLB entry. It also varies depending on whether the system supports multiple virtual memory or single virtual memory.

The page-size information determines whether VPN (11–10) is compared. VPN (11–10) is compared for 1-kbyte pages (SZ = 0) but not for 4-kbyte pages (SZ = 1).

The sharing information (SH) determines whether the PTEH.ASID and the ASID in the TLB entry are compared. ASIDs are compared when there is no sharing between processes (SH= 0) but not when there is sharing (SH = 1).

When single virtual memory is supported (MMUCR.SV = 1) and privileged mode is engaged (SR.MD = 1), all process resources can be accessed. This means that ASIDs are not compared when single virtual memory is supported and privileged mode is engaged. The objects of address comparison are shown in figure 5.8.



SH = 1 or (SR.MD = 1 and MMUCR.SV = 1)?
No
Yes
SZ = 0?
No (4 kbytes)
Yes (1 kbyte)
SZ = 0?
No (4 kbytes)
Yes (1 kbyte)
Bits compared: VPN (31–17) VPN (11–10)
Bits compared: VPN (31–17)
Bits compared: VPN (31–17) VPN (11–10) ASID (7–0)
Bits compared: VPN (31–17) ASID (7–0)

Figure 5.8 Objects of Address Comparison

5.3.4 Page Management Information

In addition to the SH and SZ bits, the page management information of TLB entries also includes D, C, and PR bits.

The D bit of a TLB entry indicates whether the page is dirty (i.e., has been written to). If the D bit is 0, an attempt to write to the page results in an initial page write exception. For physical page swapping between secondary memory and main memory, for example, pages are controlled so that a dirty page is paged out of main memory only after that page is written back to secondary memory. To record that there has been a write to a given page in the address translation table in memory, an initial page write exception is used.

The C bit in the entry indicates whether the referenced page resides in a cacheable or non-cacheable area of memory. When the control register in area 1 is mapped, set the C bit to 0. The PR field specifies the access rights for the page in privileged and user modes and is used to protect memory. Attempts at nonpermitted accesses result in TLB protection violation exceptions.

Access states designated by the D, C, and PR bits are shown in table 5.2.

Table 5.2 Access States Designated by D, C, and PR Bits

		Privileged Mode		User Mode	
		Reading	**Writing**	**Reading**	**Writing**
D bit	0	Permitted	Initial page write exception	Permitted	Initial page write exception
	1	Permitted	Permitted	Permitted	Permitted
C bit	0	Permitted (no caching)	Permitted (no caching)	Permitted (no caching)	Permitted (no caching)
	1	Permitted (with caching)	Permitted (with caching)	Permitted (with caching)	Permitted (with caching)
PR bit	00	Permitted	TLB protection violation exception	TLB protection violation exception	TLB protection violation exception
	01	Permitted	Permitted	TLB protection violation exception	TLB protection violation exception
	10	Permitted	TLB protection violation exception	Permitted	TLB protection violation exception
	11	Permitted	Permitted	Permitted	Permitted

5.4 MMU Functions

5.4.1 MMU Hardware Management

There are two kinds of MMU hardware management as follows:

1. The MMU decodes the virtual address accessed by a process and performs address translation by controlling the TLB in accordance with the MMUCR settings.
2. In address translation, the MMU receives page management information from the TLB, and determines the MMU exception and whether the cache is to be accessed (using the C bit). For details of the determination method and the hardware processing, see section 5.5, MMU Exceptions.

5.4.2 MMU Software Management

There are three kinds of MMU software management, as follows.

1. MMU register setting. MMUCR setting, in particular, should be performed in areas P1 and P2 for which address translation is not performed. Also, since SV and IX bit changes constitute address translation system changes, in this case, TLB flushing should be performed by simultaneously writing 1 to the TF bit also. Since MMU exceptions are not generated in the MMU disabled state with the AT bit cleared to 0, use in the disabled state must be avoided with software that does not use the MMU.
2. TLB entry recording, deletion, and reading. TLB entry recording can be done in two ways by using the LDTLB instruction, or by writing directly to the memory-mapped TLB. For TLB entry deletion and reading, the memory allocation TLB can be accessed. See section 5.4.3, MMU Instruction (LDTLB), for details of the LDTLB instruction, and section 5.6, Memory-Mapped TLB Configuration, for details of the memory-mapped TLB.
3. MMU exception processing. When an MMU exception is generated, it is handled on the basis of information set from the hardware side. See section 5.5, MMU Exceptions, for details.

When single virtual memory mode is used, it is possible to create a state in which physical memory access is enabled in the privileged mode only by clearing the share status bit (SH) to 0 to specify recording of all TLB entries. This strengthens inter-process memory protection, and enables special access levels to be created in the privileged mode only.

Recording a 1-kbyte page TLB entry may result in a synonym problem. See section 5.4.4, Avoiding Synonym Problems.

5.4.3 MMU Instruction (LDTLB)

The load TLB instruction (LDTLB) is used to record TLB entries. When the IX bit in MMUCR is 0, the LDTLB instruction changes the TLB entry in the way specified by the RC bit in MMUCR to the value specified by PTEH and PTEL, using VPN bits 16–12 specified in PTEH as the index number. When the IX bit in MMUCR is 1, the EX-OR of VPN bits 16–12 specified in PTEH and ASID bits 4–0 in PTEH are used as the index number.

Figure 5.9 shows the case where the IX bit in MMUCR is 0.

When an MMU exception occurs, the virtual page number of the virtual address that caused the exception is set in PTEH by hardware. The way is set in the RC bit of MMUCR for each exception according to the rules shown in figure 5.9. Consequently, if the LDTLB instruction is issued after setting only PTEL in the MMU exception processing routine, TLB entry recording is possible. Any TLB entry can be updated by software rewriting of PTEH and the RC bits in MMUCR.

As the LDTLB instruction changes address translation information, there is a risk of destroying address translation information if this instruction is issued in the P0, U0, or P3 area. Make sure, therefore, that this instruction is issued in the P1 or P2 area. Also, an instruction associated with an access to the P0, U0, or P3 area (such as the RTE instruction) should be issued at least two instructions after the LDTLB instruction.

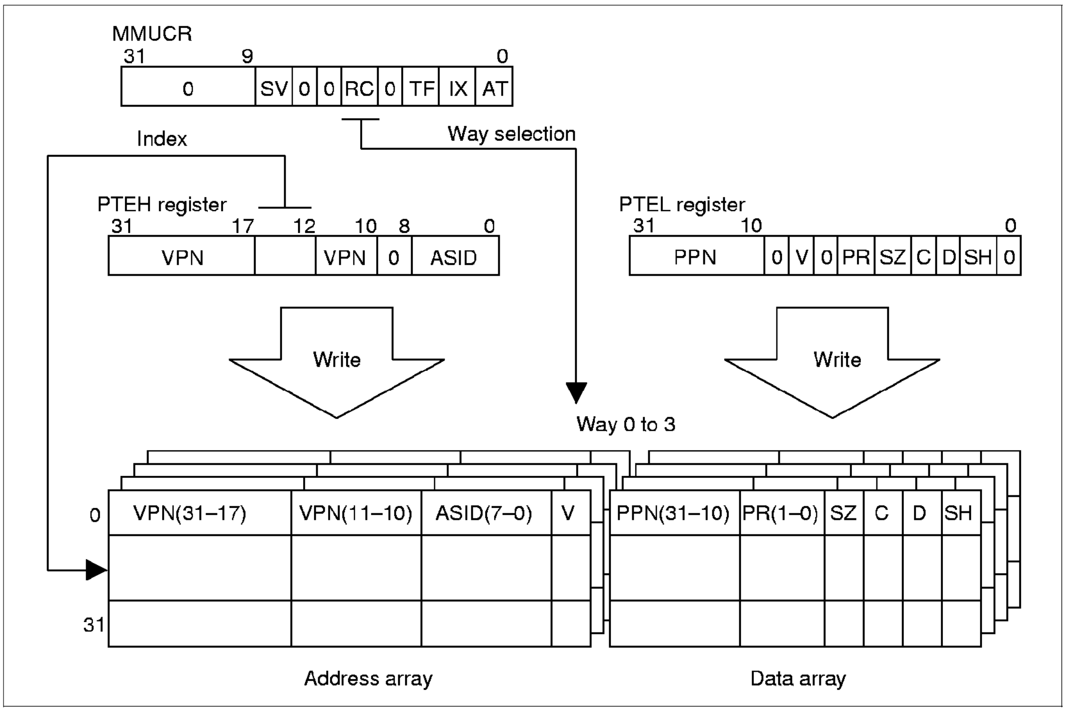


Figure 5.9 Operation of LDTLB Instruction

5.4.4 Avoiding Synonym Problems

When a 1-kbyte page is recorded in a TLB entry, a synonym problem may arise. If a number of virtual addresses are mapped onto a single physical address, the same physical address data will be recorded in a number of cache entries, and it will not be possible to guarantee data congruity. The reason why this problem only occurs when using a 1-kbyte page is explained below with reference to figure 5.10.

To achieve high-speed operation of the SH7729 cache, an index number is created using virtual address bits 10–4. When a 4-kbyte page is used, virtual address bits 10–4 are included in the offset, and since they are not subject to address translation, they are the same as physical address bits 10–4. In cache-based address comparison and recording in the address array, since the cache tag address is a physical address, physical address bits 31–10 are recorded.

When a 1-kbyte page is used, also, a cache index number is created using virtual address bits 10-4. However, in case of a 1-kbyte page, virtual address bit 10 is subject to address translation and therefore may not be the same as physical address bit 10. Consequently, the physical address is recorded in a different entry from that of the index number indicated by the physical address in the cache address array.

For example, assume that, with 1-kbyte page TLB entries, TLB entries for which the following translation has been performed are recorded in two TLBs:

Virtual address 1 H'00000000 → physical address H'00000400
Virtual address 2 H'00000400 → physical address H'00000400

Virtual address 1 is recorded in cache entry H'00, and virtual address 2 in cache entry H'40. Since two virtual addresses are recorded in different cache entries despite the fact that the physical addresses are the same, memory inconsistency will occur as soon as a write is performed to either virtual address. Therefore, when recording a 1-kbyte TLB entry, if the physical address is the same as a physical address already used in another TLB entry, it should be recorded in such a way that physical address bit 10 is the same.

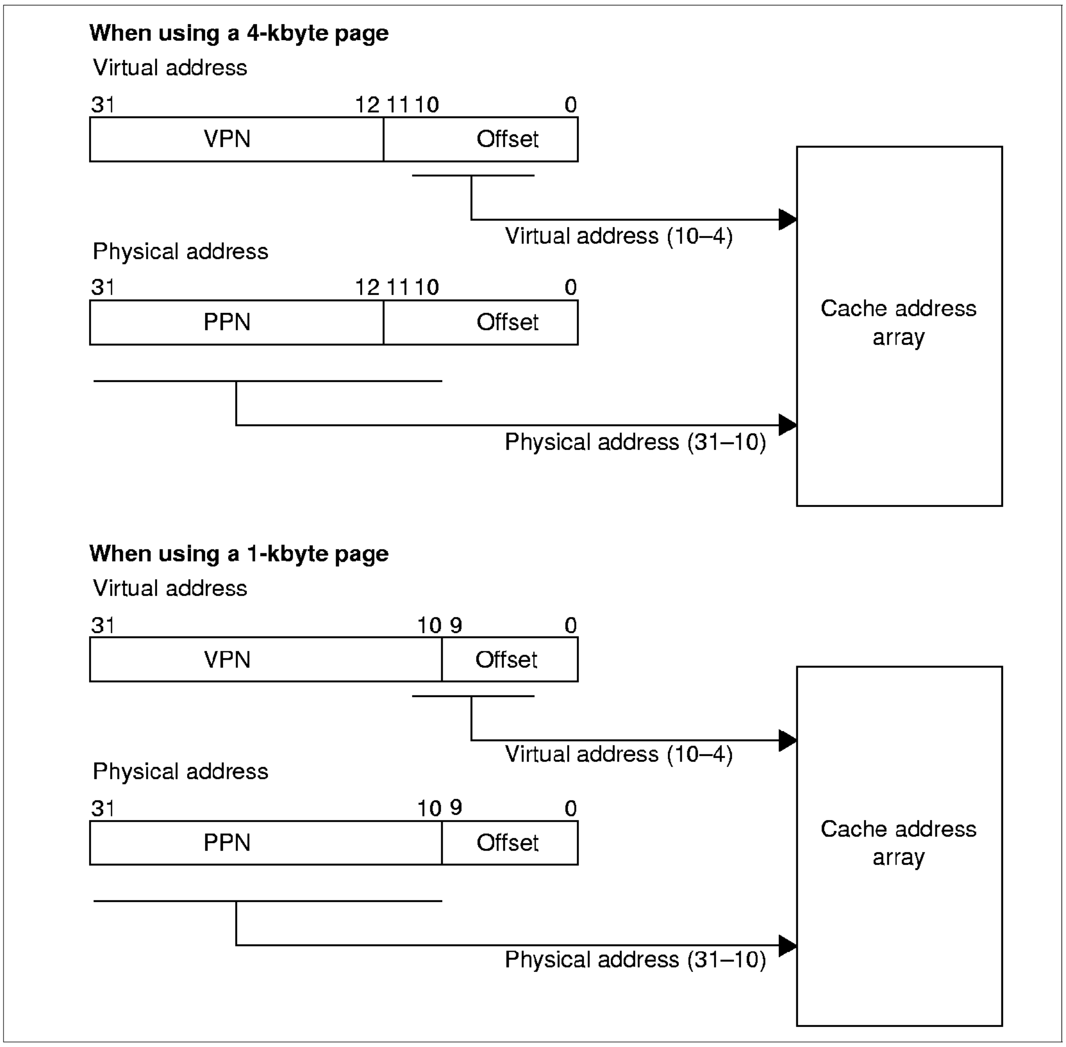


Figure 5.10 Synonym Problem

5.5 MMU Exceptions

There are four MMU exceptions: TLB miss, TLB protection violation, TLB invalid, and initial page write.

5.5.1 TLB Miss Exception

A TLB miss results when the virtual address and the address array of the selected TLB entry are compared and no match is found. TLB miss exception processing includes both hardware and software operations.

Hardware Operations: In a TLB miss, the SH7729 hardware executes a set of prescribed operations, as follows:

1. The VPN field of the virtual address causing the exception is written to the PTEH register.
2. The virtual address causing the exception is written to the TEA register.
3. Either exception code H'040 for a load access, or H'060 for a store access, is written to the EXPEVT register.
4. The PC value indicating the address of the instruction in which the exception occurred is written to the save program counter (SPC). If the exception occurred in a delay slot, the PC value indicating the address of the related delayed branch instruction is written to the SPC.
5. The contents of the status register (SR) at the time of the exception are written to the save status register (SSR).
6. The mode (MD) bit in SR is set to 1 to place the SH7729 in the privileged mode.
7. The block (BL) bit in SR is set to 1 to mask any further exception requests.
8. The register bank (RB) bit in SR is set to 1.
9. The RC field in the MMU control register (MMUCR) is incremented by 1 when all entries indexed are valid. When some entries indexed are invalid, the smallest way number of them is set in RC.
10. Execution branches to the address obtained by adding the value of the VBR contents and H'00000400 to invoke the user-written TLB miss exception handler.

Software (TLB Miss Handler) Operations: The software searches the page tables in external memory and allocates the required page table entry. Upon retrieving the required page table entry, software must execute the following operations:

1. Write the value of the physical page number (PPN) field and the protection key (PR), page size (SZ), cacheable (C), dirty (D), share status (SH), and valid (V) bits of the page table entry recorded in the address translation table in the external memory into the PTEL register in the SH7729.

2. If using software for way selection for entry replacement, write the desired value to the RC field in MMUCR.
3. Issue the LDTLB instruction to load the contents of PTEH and PTEL into the TLB.
4. Issue the return from exception handler (RTE) instruction to terminate the handler routine and return to the instruction stream.

5.5.2 TLB Protection Violation Exception

A TLB protection violation exception results when the virtual address and the address array of the selected TLB entry are compared and a valid entry is found to match, but the type of access is not permitted by the access rights specified in the PR field. TLB protection violation exception processing includes both hardware and software operations.

Hardware Operations: In a TLB protection violation exception, the SH7729 hardware executes a set of prescribed operations, as follows:

1. The VPN field of the virtual address causing the exception is written to the PTEH register.
2. The virtual address causing the exception is written to the TEA register.
3. Either exception code H'0A0 for a load access, or H'0C0 for a store access, is written to the EXPEVT register.
4. The PC value indicating the address of the instruction in which the exception occurred is written into SPC (if the exception occurred in a delay slot, the PC value indicating the address of the related delayed branch instruction is written into SPC).
5. The contents of SR at the time of the exception are written to SSR.
6. The MD bit in SR is set to 1 to place the SH7729 in the privileged mode.
7. The BL bit in SR is set to 1 to mask any further exception requests.
8. The register bank (RB) bit in SR is set to 1.
9. The way that generated the exception is set in the RC field in MMUCR.
10. Execution branches to the address obtained by adding the value of the VBR contents and H'00000100 to invoke the TLB protection violation exception handler.

Software (TLB Protection Violation Handler) Operations: Software resolves the TLB protection violation and issues the RTE (return from exception handler) instruction to terminate the handler and return to the instruction stream.

5.5.3 TLB Invalid Exception

A TLB invalid exception results when the virtual address is compared to a selected TLB entry address array and a match is found but the entry is not valid (the V bit is 0). TLB invalid exception processing includes both hardware and software operations.

Hardware Operations: In a TLB invalid exception, the SH7729 hardware executes a set of prescribed operations, as follows:

1. The VPN number of the virtual address causing the exception is written to the PTEH register.
2. The virtual address causing the exception is written to the TEA register.
3. The way number causing the exception is written to RC in MMUCR.
4. Either exception code H'040 for a load access, or H'060 for a store access, is written to the EXPEVT register.
5. The PC value indicating the address of the instruction in which the exception occurred is written to the SPC. If the exception occurred in a delay slot, the PC value indicating the address of the delayed branch instruction is written to the SPC.
6. The contents of SR at the time of the exception are written into SSR.
7. The mode (MD) bit in SR is set to 1 to place the SH7729 in the privileged mode.
8. The block (BL) bit in SR is set to 1 to mask any further exception requests.
9. The register bank (RB) bit in SR is set to 1.
10. Execution branches to the address obtained by adding the value of the VBR contents and H'00000100, and the TLB protection violation exception handler starts.

Software (TLB Invalid Exception Handler) Operations: The software searches the page tables in external memory and assigns the required page table entry. Upon retrieving the required page table entry, software must execute the following operations:

1. Write the values of the physical page number (PPN) field and the values of the protection key (PR), page size (SZ), cacheable (C), dirty (D), share status (SH), and valid (V) bits of the page table entry recorded in the external memory to the PTEL register.
2. If using software for way selection for entry replacement, write the desired value to the RC field in MMUCR.
3. Issue the LDTLB instruction to load the contents of PTEH and PTEL into the TLB.
4. Issue the RTE instruction to terminate the handler and return to the instruction stream. The RTE instruction should be issued after two LDTLB instructions.

5.5.4 Initial Page Write Exception

An initial page write exception results in a write access when the virtual address and the address array of the selected TLB entry are compared and a valid entry with the appropriate access rights is found to match, but the D (dirty) bit of the entry is 0 (the page has not been written to). Initial page write exception processing includes both hardware and software operations.

Hardware Operations: In an initial page write exception, the SH7729 hardware executes a set of prescribed operations, as follows:

1. The VPN field of the virtual address causing the exception is written to the PTEH register.
2. The virtual address causing the exception is written to the TEA register.
3. Exception code H'080 is written to the EXPEVT register.
4. The PC value indicating the address of the instruction in which the exception occurred is written to the SPC. If the exception occurred in a delay slot, the PC value indicating the address of the related delayed branch instruction is written to the SPC.
5. The contents of SR at the time of the exception are written to SSR.
6. The MD bit in SR is set to 1 to place the SH7729 in the privileged mode.
7. The BL bit in SR is set to 1 to mask any further exception requests.
8. The register bank (RB) bit in SR is set to 1.
9. The way that caused the exception is set in the RC field in MMUCR.
10. Execution branches to the address obtained by adding the value of the VBR contents and H'00000100 to invoke the user-written initial page write exception handler.

Software (Initial Page Write Handler) Operations: The software must execute the following operations:

1. Retrieve the required page table entry from external memory.
2. Set the D bit of the page table entry in the external memory to 1.
3. Write the value of the PPN field and the PR, SZ, C, D, SH, and V bits of the page table entry in the external memory to the PTEL register.
4. If using software for way selection for entry replacement, write the desired value to the RC field in MMUCR.
5. Issue the LDTLB instruction to load the contents of PTEH and PTEL into the TLB.
6. Issue the RTE instruction to terminate the handler and return to the instruction stream. The RTE instruction should be issued after two LDTLB instructions.

Figure 5.11 shows the flowchart for MMU exceptions.

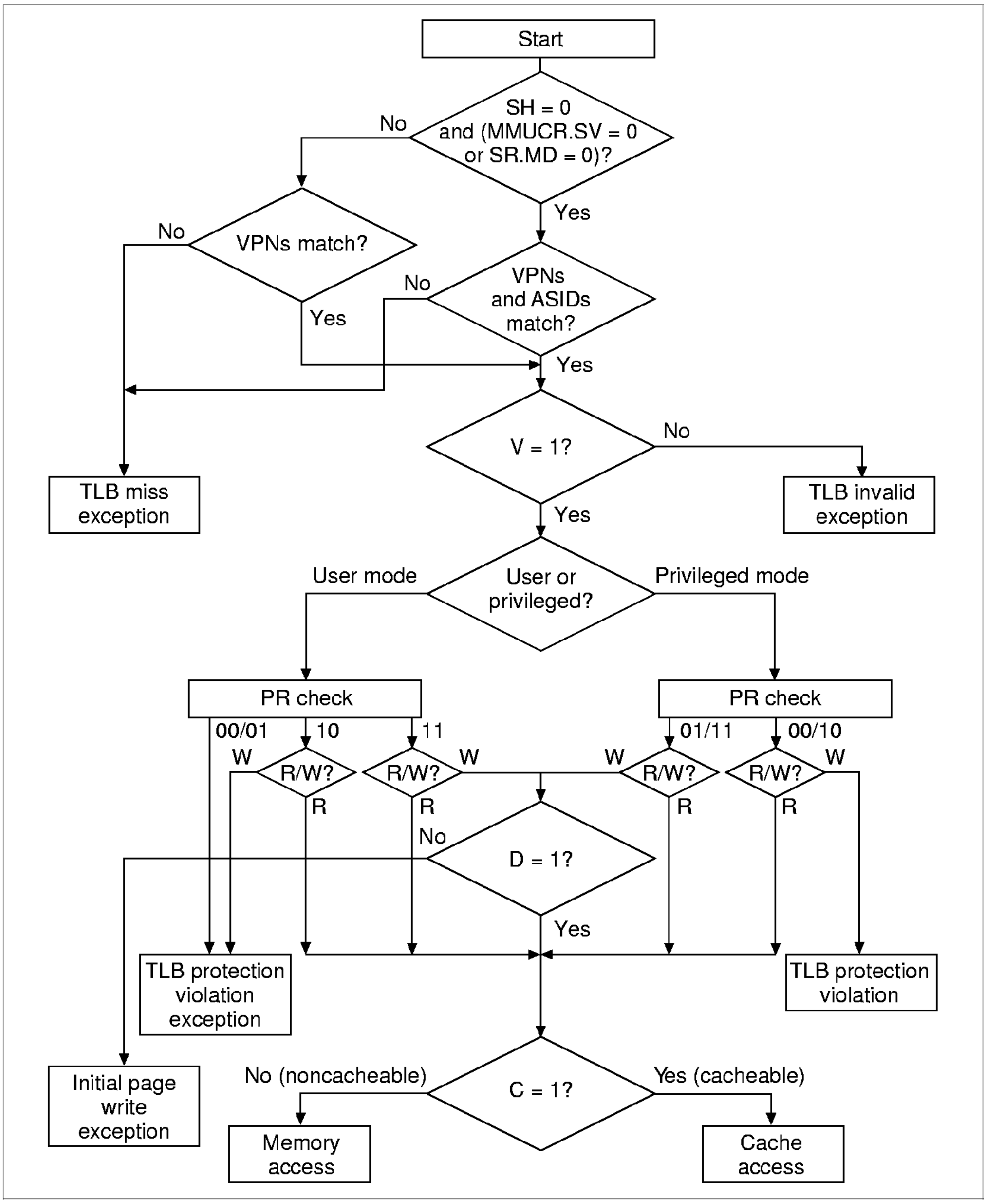


Figure 5.11 MMU Exception Generation Flowchart

5.5.5 Processing Flow in Event of MMU Exception (Same Processing Flow for Address Error)

Figure 5.12 shows the MMU exception signals in the instruction fetch mode.

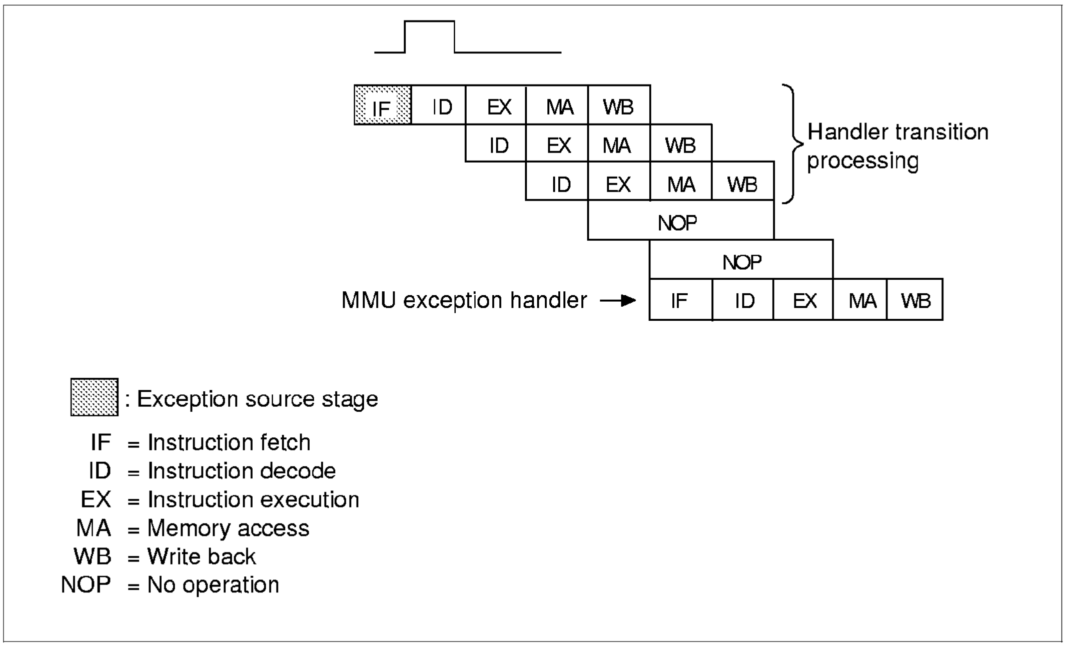


Figure 5.12 MMU Exception Signals in Instruction Fetch

Figure 5.13 shows the MMU exception signals in the data access mode.

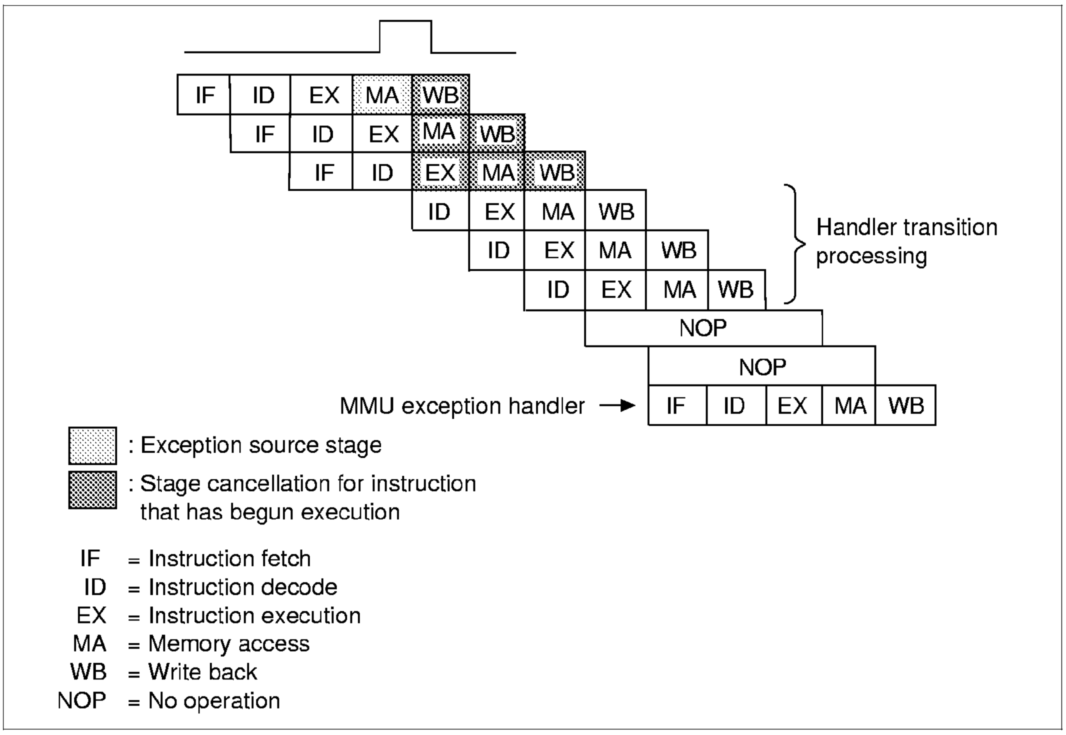
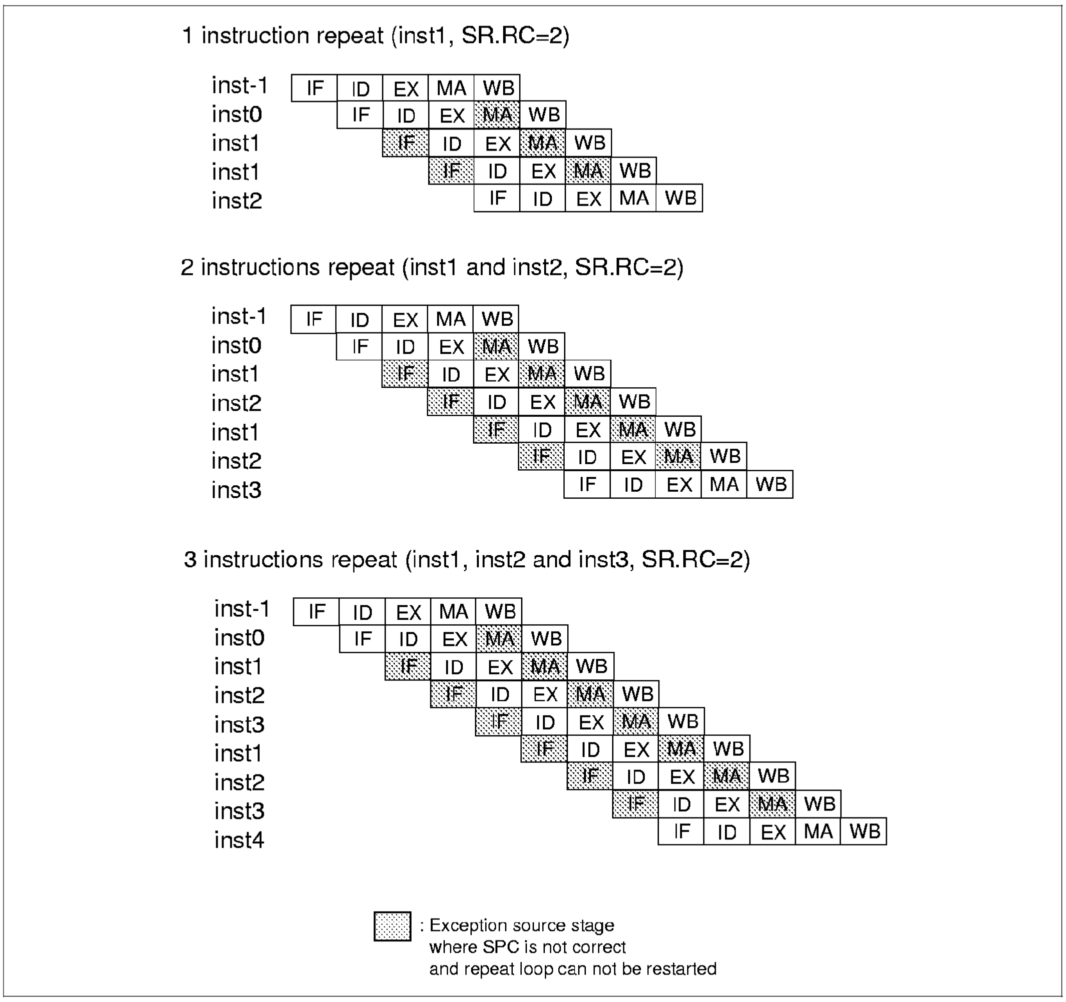


Figure 5.13 MMU Exception Signals in Data Access

5.5.6 MMU Exception in Repeat Loop

When MMU exception or CPU address error occurs immediately before or within a repeat loop, the PC of the instruction that generated the exception can not be saved in SPC correctly and repeat loop can not be restarted after returning from exception handler. EXPEVT is set to H'070 in cases of TLB miss, TLB invalid, and CPU address error. EXPEVT is set to H'0D0 in case of TLB protection violation. Figure 5.14 describes the places where this case occurs.



1 instruction repeat (inst1, SR.RC=2)

inst-1	IF	ID	EX	MA	WB				
inst0		IF	ID	EX	▨MA	WB			
inst1			▨IF	ID	EX	▨MA	WB		
inst1				▨IF	ID	EX	▨MA	WB	
inst2					IF	ID	EX	MA	WB

2 instructions repeat (inst1 and inst2, SR.RC=2)

inst-1	IF	ID	EX	MA	WB						
inst0		IF	ID	EX	▨MA	WB					
inst1			▨IF	ID	EX	▨MA	WB				
inst2				▨IF	ID	EX	▨MA	WB			
inst1					▨IF	ID	EX	▨MA	WB		
inst2						▨IF	ID	EX	▨MA	WB	
inst3							IF	ID	EX	MA	WB

3 instructions repeat (inst1, inst2 and inst3, SR.RC=2)

inst-1	IF	ID	EX	MA	WB								
inst0		IF	ID	EX	▨MA	WB							
inst1			▨IF	ID	EX	▨MA	WB						
inst2				▨IF	ID	EX	▨MA	WB					
inst3					▨IF	ID	EX	▨MA	WB				
inst1						▨IF	ID	EX	▨MA	WB			
inst2							▨IF	ID	EX	▨MA	WB		
inst3								▨IF	ID	EX	▨MA	WB	
inst4									IF	ID	EX	MA	WB

▨ : Exception source stage
where SPC is not correct
and repeat loop can not be restarted

Figure 5.14 MMU Exception in Repeat Loop

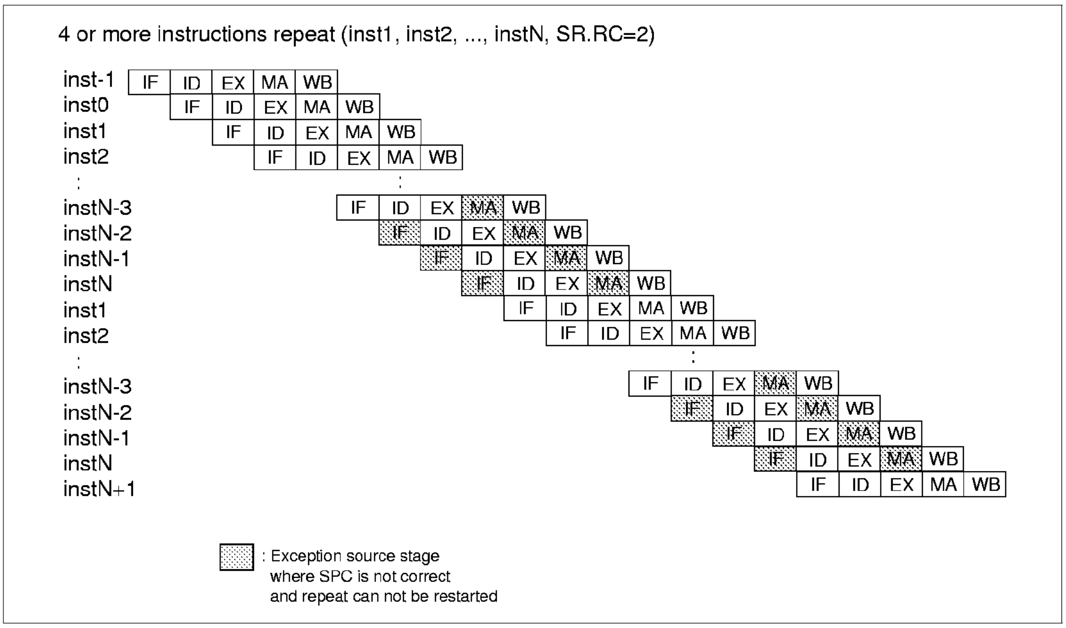


Figure 5.14 MMU Exception in Repeat Loop (cont)

5.6 Memory-Mapped TLB

In order for TLB operations to be managed by software, TLB contents can be read or written to in the privileged mode using the MOV instruction. The TLB is assigned to the P4 area in the virtual address space. The TLB address array (VPN, V bit, and ASID) is assigned to H'F2000000–H'F2FFFFFF, and the data array (PPN, PR, SZ, C, D, and SH bits) to H'F3000000–H'F3FFFFFF. The V bit in the address array can also be accessed from the data array. Only longword access is possible for both the address array and the data array.

5.6.1 Address Array

The address array is assigned to H'F2000000 to H'F2FFFFFF. To access an address array, the 32-bit address field (for read/write operations) and 32-bit data field (for write operations) must be specified. The address field specifies information for selecting the entry to be accessed; the data field specifies the VPN, V bit and ASID to be written to the address array (figure 5.15 (1)).

In the address field, specify the entry address for selecting the entry (bits 16–12), W for selecting the way (bits 9–8: 00 is way 0, 01 is way 1, 10 is way 2, 11 is way 3) and H'F2 to indicate address array access (bits 31–24). The IX bit in MMUCR indicates whether an EX-OR is taken of the entry address and ASID.

When writing, specify bit 7 as the A bit. The A bit indicates whether addresses are compared during writing. When the A bit is 1, the VPNs of the four entries selected by the entry addresses are compared to the VPN to be written into the address array specified in the data field. Writing takes place to the way that has a hit. When a miss occurs, nothing is written to the address array and no operation occurs. The way number specified in bits 9–8 is not used. The item compared is determined by the SZ and SH bits of the entry selected by the entry address, the SV bit in MMUCR and the MD bit in SR, just as in ordinary operations (see section 5.3.3, TLB Address Comparison).

When the A bit is 0, it is written to the entry selected with the entry address and way number without comparing addresses.

When reading, the VPN (31–17, 11–10), V bit, and ASID of the entry specified by the entry address and way number are read in the format of the data field in figure 5.15 without comparing addresses.

To invalidate a specific entry, specify the entry and write 0 to its V bit. When 1 is specified for the A bit, only the required VPN entry is invalidated.

5.6.2 Data Array

The data array is assigned to H'F3000000 to H'F3FFFFFF. To access a data array, the 32-bit address field (for read/write operations), and 32-bit data field (for write operations) must be specified. These are specified in the general register. The address section specifies information for selecting the entry to be accessed; the data section specifies the longword data to be written to the data array (figure 5.15 (2)).

In the address section, specify the entry address for selecting the entry (bits 16–12), W for selecting the way (bits 9–8: 00 is way 0, 01 is way 1, 10 is way 2, 11 is way 3), and H'F3 to indicate data array access (bits 31–24). The IX bit in MMUCR indicates whether an EX-OR is taken of the entry address and ASID.

Both reading and writing use the longword of the data array specified by the entry address and way number. The access size of the data array is fixed at longword.

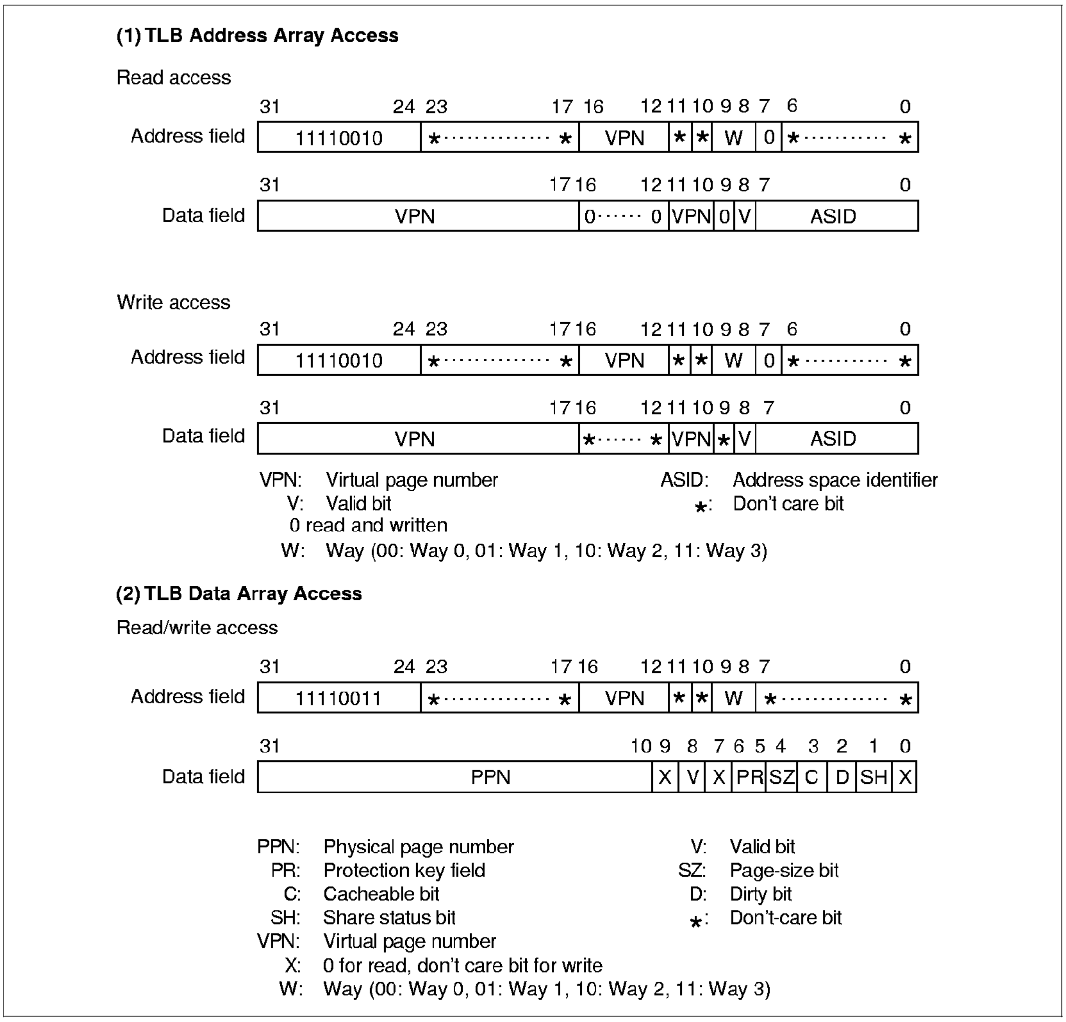

(1) TLB Address Array Access
Read access
31 24 23 17 16 12 11 10 9 8 7 6 0
Address field
11110010 | *············* | VPN | * | * | W | 0 | *···········*
31 17 16 12 11 10 9 8 7 0
Data field
VPN | 0······0 | VPN | 0 | V | ASID
Write access
31 24 23 17 16 12 11 10 9 8 7 6 0
Address field
11110010 | *············* | VPN | * | * | W | 0 | *···········*
31 17 16 12 11 10 9 8 7 0
Data field
VPN | *······* | VPN | * | V | ASID
VPN: Virtual page number
ASID: Address space identifier
V: Valid bit
*: Don't care bit
0 read and written
W: Way (00: Way 0, 01: Way 1, 10: Way 2, 11: Way 3)
(2) TLB Data Array Access
Read/write access
31 24 23 17 16 12 11 10 9 8 7 0
Address field
11110011 | *············* | VPN | * | * | W | *·············*
31 10 9 8 7 6 5 4 3 2 1 0
Data field
PPN | X | V | X | PR | SZ | C | D | SH | X
PPN: Physical page number
V: Valid bit
PR: Protection key field
SZ: Page-size bit
C: Cacheable bit
D: Dirty bit
SH: Share status bit
*: Don't-care bit
VPN: Virtual page number
X: 0 for read, don't care bit for write
W: Way (00: Way 0, 01: Way 1, 10: Way 2, 11: Way 3)

Figure 5.15 Specifying Address and Data for Memory-Mapped TLB Access

5.6.3 Usage Examples

Invalidating Specific Entries: Specific TLB entries can be invalidated by writing 0 to the entry's V bit. When the A bit is 1, the VPN and ASID specified by the write data is compared to the VPN and ASID within the TLB entry selected by the entry address and data is written to the matching way. If no match is found, there is no operation. R0 specifies the write data and R1 specifies the address.

```
; R0=H'1547 381C  R1=H'F201 3080
; MMUCR.IX=0
; VPN(31-17)=B'0001 0101 0100 011   VPN(11-10)=B'10   ASID=B'0001 1100
; corresponding entry association is made from the entry selected by
; the VPN(16-12)=B'1 0011 index, the V bit of the hit way is cleared to
; 0,achieving invalidation.
MOV.L  R0,@R1
```

Reading the Data of a Specific Entry: This example reads the data section of a specific TLB entry. The bit order indicated in the data field in figure 5.15 (2) is read. R0 specifies the address and the data section of a selected entry is read to R1.

```
; R1=H'F300 4300   VPN(16-12)=B'00100   Way 3
; MOV.L @R0,R1
```

5.7 Usage Note

Instructions that manipulate the MD or BL bit in register SR (the LDC Rm, SR instruction, LDC @Rm+, SR instruction, and RTE instruction) and the following instruction, or the LDTLB instruction, should be used with the TLB disabled or in a fixed physical address space (the P1 or P2 space).

Section 6 Exception Processing

6.1 Overview

6.1.1 Features

Exception processing is separate from normal program processing, and is performed by a routine separate from the normal program. In response to an exception processing request due to abnormal termination of the executing instruction, control is passed to a user-written exception handler. However, in response to an interrupt request, normal program execution continues until the end of the executing instruction. Here, all exceptions other than resets and interrupts will be called general exceptions. There are thus three types of exceptions: resets, general exceptions, and interrupts.

6.1.2 Register Configuration

Table 6.1 lists the registers used for exception processing. A register with an undefined initial value should be initialized by software.

Table 6.1 Register Configuration

Register	Abbr.	R/W	Size	Initial Value	Address
TRAPA exception register	TRA	R/W	Longword	Undefined	H'FFFFFFD0
Exception event register	EXPEVT	R/W	Longword	Power-on reset: H'000 Manual reset: H'020	H'FFFFFFD4
Interrupt event register	INTEVT	R/W	Longword	Undefined	H'FFFFFFD8
Interrupt event register2	INTEVT2	R	Longword	Undefined	H'04000000

6.2 Exception Processing Function

6.2.1 Exception Processing Flow

In exception processing, 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 at the completion of the routine, restoring the contents of the PC and SR to return to the processor state at the point of interruption and the address where the exception occurred.

A basic exception processing sequence consists of the following operations:

1. The contents of the PC and SR are saved in the SPC and SSR, respectively.
2. The block (BL) bit in SR is set to 1, masking any subsequent exceptions.
3. The mode (MD) bit in SR is set to 1 to place the SH7729 in the privileged mode.
4. The register bank (RB) bit in SR is set to 1.
5. An exception code identifying the exception event is written to bits 11–0 of the exception event (EXPEVT) or interrupt event (INTEVT and INTEVT2) register.
6. Instruction execution jumps to the designated exception processing vector address to invoke the handler routine.

6.2.2 Exception Processing Vector Addresses

The reset vector address is fixed at H'A0000000. The other three events are assigned offsets from the vector base address by software. Translation look-aside buffer (TLB) miss exceptions have an offset from the vector base address of H'00000400. The vector address offset for general exception events other than TLB miss exceptions is H'00000100. The interrupt vector address offset is H'00000600. The vector base address is loaded into the vector base register (VBR) by software. The vector base address should reside in P1 or P2 fixed physical address space. Figure 6.1 shows the relationship between the vector base address, the vector offset, and the vector table.

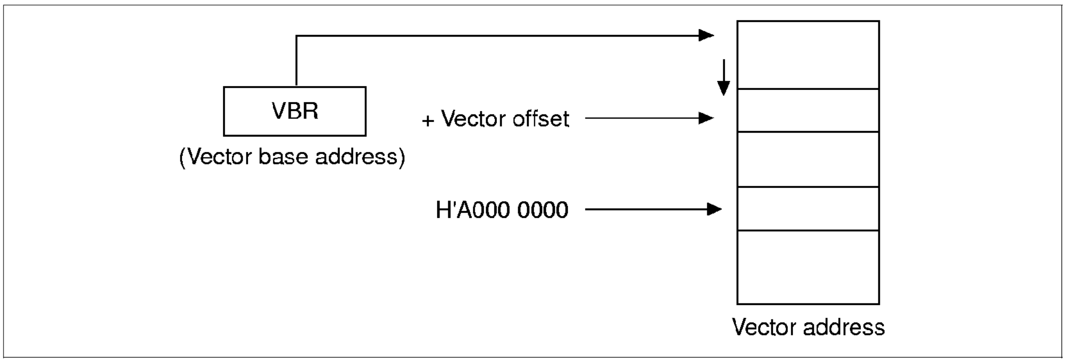


Figure 6.1 Vector Addresses

In table 6.2, exceptions and their vector addresses are listed by exception type, instruction completion state, relative acceptance priority, relative order of occurrence within an instruction execution sequence and vector address for exceptions and their vector addresses.

Table 6.2 Exception Event Vectors

Exception Type	Current Instruction	Exception Event	Priority*1	Exception Order	Vector Address	Vector Offset
Reset	Aborted	Power-on	1	—	H'A00000000	—
		Manual reset	1	—	H'A00000000	—
		H-UDI reset	2	—	H’A00000000	—
General exception events	Aborted and retried	CPU Address error (instruction access)	2	1	—	H'00000100
		TLB miss (instruction access not in repeat loop)	2	2	—	H'00000400
		TLB miss (instruction access in repeat loop)*4	2	2	---	H’00000100
		TLB invalid (instruction access)	2	3	—	H'00000100
		TLB protection violation (instruction access)	2	4	—	H'00000100
		Reserved instruction code exception	2	5	—	H'00000100
		Illegal slot instruction exception	2	5	—	H'00000100
		CPU_Address error (data access)	2	6	—	H'00000100
		TLB miss (data access not in repeat loop)	2	7	—	H'00000400
		TLB miss (data access in repeat loop)*4	2	7	—	H'00000100
		TLB invalid (data access)	2	8	—	H'00000100
		TLB protection violation (data access)	2	9	—	H'00000100
		Initial page write	2	10	—	H'00000100
	Completed	Unconditional trap (TRAPA instruction)	2	5	—	H'00000100

Table 6.2 Exception Event Vectors (cont)

Exception Type	Current Instruction	Exception Event	Priority*1	Exception Order	Vector Address	Vector Offset
General exception events	Completed	User breakpoint trap	2	n*2	—	H'00000100
General interrupt requests	Completed	DMA address error	2	12	—	H'00000100
		Nonmaskable interrupt	3	—	—	H'00000600
		External hardware interrupt	4*3	—	—	H'00000600
		H-UDI interrupt	4*3	—	—	H'00000600

Notes: 1. Priorities are indicated from high to low, 1 being highest and 4 being lowest.

2. The user defines the break point traps. 1 is a break point before instruction execution and 11 is a break point after instruction execution. For an operand break point, use 11.

3. Use software to specify relative priorities of external hardware interrupts and peripheral module interrupts (see section 9, Interrupt Controller (INTC)).

4. See section 6.5.2, General Exceptions for details.

6.2.3 Acceptance of Exceptions

Processor resets and interrupts are asynchronous events unrelated to the instruction stream. All exception events are prioritized to establish an acceptance order whenever two or more exception events occur simultaneously. The power-on reset and manual reset may not occur simultaneously, so they have the same priority.

All general exception events occur in a relative order in the execution sequence of an instruction (i.e. execution order), but are handled at priority level 2 in instruction-stream order (i.e. program order), where an exception detected in a preceding instruction is accepted prior to an exception detected in a subsequent instruction.

Three general exception events (reserved instruction code exception, unconditional trap, and illegal slot instruction exception) are detected in the decode stage (ID stage) of different instructions and are mutually exclusive events in the instruction pipeline. They have the same execution priority. Figure 6.2 shows the order of general exception acceptance.

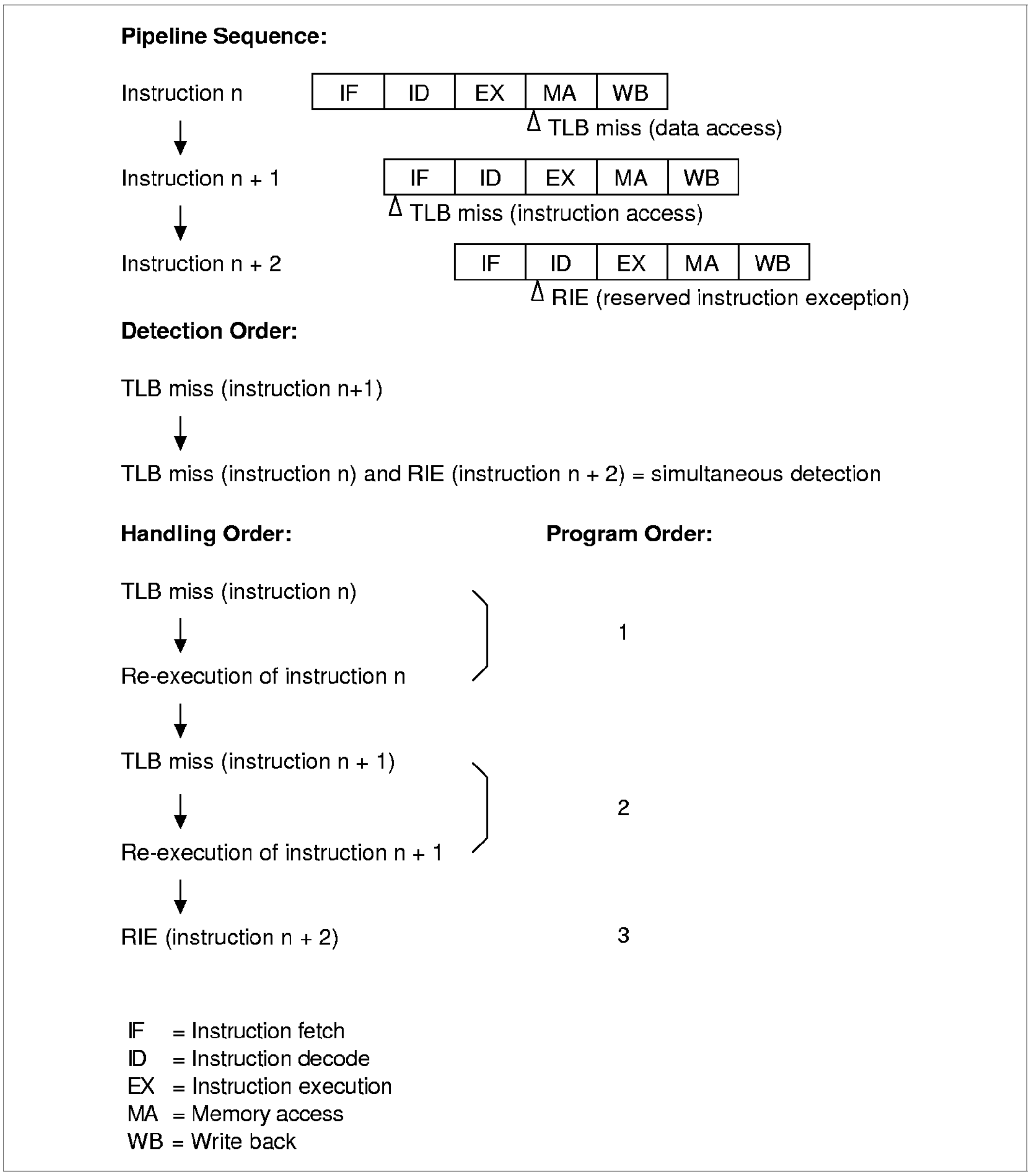


Figure 6.2 Example of Acceptance Order of General Exceptions

All exceptions other than a reset are detected in the pipeline ID stage, and accepted on instruction boundaries. However, an exception is not accepted between a delayed branch instruction and the delay slot. A re-execution type exception detected in a delay slot is accepted before execution of the delayed branch instruction. A completion type exception detected in a delayed branch instruction or delay slot is accepted after execution of the delayed branch instruction. The delay slot here refers to the next instruction after a delayed unconditional branch instruction, or the next instruction when a delayed conditional branch instruction is true.

6.2.4 Exception Codes

Table 6.3 lists the exception codes written to bits 11–0 of the EXPEVT register (for reset or general exceptions) or the INTEVT and INTEVT2 registers (for general interrupt requests) to identify each specific exception event. An additional exception register, the TRAPA (TRA) register, is used to hold the 8-bit immediate data in an unconditional trap (TRAPA instruction).

Table 6.3 Exception Codes

Exception Type	Exception Event	Exception Code
Reset	Power-on reset	H'000
	Manual reset	H'020
	H-UDI reset	H’000
General exception events	TLB miss/invalid (load)	H'040
	TLB miss/invalid (store)	H'060
	TLB miss/invalid/CPU Address error in repeat loop	H’070
	Initial page write	H'080
	TLB protection violation (load)	H'0A0
	TLB protection violation (store)	H'0C0
	TLB protection violation in repeat loop	H’0D0
	CPU Address error (load)	H'0E0
	CPU Address error (store)	H'100
	Unconditional trap (TRAPA instruction)	H'160
	Reserved instruction code exception	H'180
	Illegal slot instruction exception	H'1A0
	User breakpoint trap	H'1E0
	DMA address error	H'600
General interrupt requests	Nonmaskable interrupt	H'1C0
	H-UDI interrupt	H’620
	External hardware interrupts:	
	IRL3–IRL0 = 0000	H'200
	IRL3–IRL0 = 0001	H'220

Table 6.3 Exception Codes (cont)

Exception Type	Exception Event	Exception Code
General interrupt requests (cont)	External hardware interrupts (cont):	
	IRL3–IRL0 = 0010	H'240
	IRL3–IRL0 = 0011	H'260
	IRL3–IRL0 = 0100	H'280
	IRL3–IRL0 = 0101	H'2A0
	IRL3–IRL0 = 0110	H'2C0
	IRL3–IRL0 = 0111	H'2E0
	IRL3–IRL0 = 1000	H'300
	IRL3–IRL0 = 1001	H'320
	IRL3–IRL0 = 1010	H'340
	IRL3–IRL0 = 1011	H'360
	IRL3–IRL0 = 1100	H'380
	IRL3–IRL0 = 1101	H'3A0
	IRL3–IRL0 = 1110	H'3C0

Note: Exception codes H'120, H'140, and H'3E0 are reserved.

6.2.5 Exception Request Masks

If a general exception event occurs when the BL bit in SR is 1, the CPU's internal registers are set to their post-reset state, other module registers retain their contents prior to the general exception, and a branch is made to the same address (H'A0000000) as for a reset.

If a general interrupt occurs when BL= 1, the request is masked (held pending) and not accepted until the BL bit is cleared to 0 by software. For reentrant exception processing, the SPC and SSR must be saved and the BL bit in SR cleared to 0.

6.2.6 Returning from Exception Handling

The RTE instruction is used to return from exception handling. When RTE is executed, the SPC value is set in the PC, and the SSR value in SR, and the return from exception processing is performed by branching to the SPC address.

If the SPC and SSR have been saved in the external memory, set the BL bit in SR to 1, then restore the SPC and SSR, and issue an RTE instruction.

6.3 Register Description

There are three registers related to exception handling. These are peripheral module registers, and therefore reside in area P4. They can be accessed by specifying the address in the privileged mode only.

1. The exception event register (EXPEVT) resides at address H'FFFFFFD4, and contains a 12-bit exception code. The exception code set in EXPEVT is that for a reset or general exception event. The exception code is set automatically by hardware when an exception occurs. EXPEVT can also be modified by software.
2. The interrupt event register (INTEVT) resides at address H'FFFFFFD8, and contains a 12-bit exception code. The exception code set in EXPEVT is that for an interrupt request. The exception code is set automatically by hardware when an exception occurs.
3. The TRAPA exception register (TRA) resides at address H'FFFFFFD0, and contains 8-bit immediate data (imm) for the TRAPA instruction. TRA is set automatically by hardware when a TRAPA instruction is executed. TRA can also be modified by software.

The bit configurations of the EXPEVT, INTEVT, and TRA registers are shown in figure 6.3.

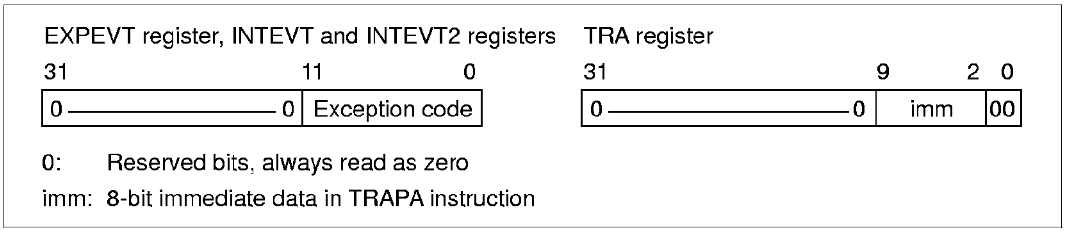

EXPEVT register, INTEVT and INTEVT2 registers
31
11
0
0 — 0
Exception code
TRA register
31
9
2
0
0 — 0
imm
00
0: Reserved bits, always read as zero
imm: 8-bit immediate data in TRAPA instruction

Figure 6.3 Bit Configurations of EXPEVT, INTEVT, INTEVT2, and TRA Registers

6.4 Exception Handler Operation

6.4.1 Reset

The reset sequence is used to power up or restart the SH7709 from the initialization state. The $\overline{\text{RESET}}$ signal is sampled every clock cycle, and in the case of a power-on reset, all processing being executed (excluding the RTC) is suspended, all unfinished events are canceled, and reset processing is executed immediately. In the case of a manual reset, however, processing to retain external memory contents is continued. The reset sequence consists of the following operations:

1. The MD bit in SR is set to 1 to place the SH7709 in privileged mode.
2. The BL bit in SR is set to 1, masking any subsequent exceptions.
3. The RB bit in SR is set to 1.

4. An encoded value of H'000 in a power-on reset or H'020 in a manual reset is written to bits 11–0 of the EXPEVT register to identify the exception event.
5. Instruction execution jumps to the user-written exception handler at address H'A0000000.

6.4.2 Interrupts

An interrupt processing request is accepted on completion of the current instruction. The interrupt acceptance sequence consists of the following operations:

1. The contents of the PC and SR are saved in SPC and SSR, respectively.
2. The BL bit in SR is set to 1, masking any subsequent exceptions.
3. The MD bit in SR is set to 1 to place the SH7709 in privileged mode.
4. The RB bit in SR is set to 1.
5. An encoded value identifying the exception event is written to bits 11–0 of the INTEVT and INTEVT2 registers.
6. Instruction execution jumps to the vector location designated by the sum of the value of the contents of the vector base register (VBR) and H'00000600 to invoke the exception handler.

6.4.3 General Exceptions

When the SH7729 encounters any exception condition other than a reset or interrupt request, it executes the following operations:

1. The contents of the PC and SR are saved in the SPC and SSR, respectively.
2. The BL bit in SR is set to 1, masking any subsequent exceptions.
3. The MD bit in SR is set to 1 to place the SH7729 in privileged mode.
4. The RB bit in SR is set to 1.
5. An encoded value identifying the exception event is written to bits 11–0 of the EXPEVT register.
6. Instruction execution jumps to the vector location designated by either the sum of the vector base address and offset H'00000400 in the vector table in a TLB miss trap, or by the sum of the vector base address and offset H'00000100 for exceptions other than TLB miss traps, to invoke the exception handler.

6.5 Individual Exception Operations

This section describes the conditions for specific exception processing, and the processor operations.

6.5.1 Resets

- Power-On Reset
  - — Conditions: $\overline{\text{RESETP}}$ low
  - — Operations: EXPEVT set to H'000, VBR and SR initialized, branch to PC = H'A0000000. Initialization sets the VBR register to H'0000000. In SR, the MD, RB and BL bits are set to 1 and the IMASK field is set to B'1111. The CPU and on-chip supporting modules are initialized. See the register descriptions in the relevant sections for details. A power-on reset must always be performed when powering on.
- Manual Reset
  - — Conditions: $\overline{\text{RESETM}}$ low
  - — Operations: EXPEVT set to H'020, VBR and SR initialized, branch to PC = H'A0000000. Initialization sets the VBR register to H'0000000. In SR, the MD, RB, and BL bits are set to 1 and the IMASK field is set to B'1111. The CPU and on-chip supporting modules are initialized. See the register descriptions in the relevant sections for details.
- H-UDI Reset
  - — Conditions: H-UDI reset command input (see section 25.4.3, H-UDI Reset)
  - — Operations: EXPEVT set to H'000, VBR and SR initialized, branch to PC = H'A0000000. Initialization sets the VBR register to H'0000000. In SR, the MD, RB and BL bits are set to 1 and the IMASK field is set to B'1111. The CPU and on-chip supporting modules are initialized. See the register descriptions in the relevant sections for details.

Table 6.4 Types of Reset

		Internal State	
Type	**Conditions for Transition to Reset State**	**CPU**	**On-Chip Supporting Modules**
Power-on reset	$\overline{\text{RESETP}}$ = Low	Initialized	(See register configuration in relevant sections)
Manual reset	$\overline{\text{RESETM}}$ = Low	Initialized	
H-UDI reset	H-UDI reset command input	Initialized	

6.5.2 General Exceptions

- TLB miss exception
  - Conditions: Comparison of TLB addresses shows no address match
  - Operations: The virtual address (32 bits) that caused the exception is set in TEA and the corresponding virtual page number (22 bits) is set in PTEH (31–10). The ASID of PTEH indicates the ASID at the time the exception occurred. The RC bit in MMUCR is incremented by one for replacement.

  The PC and SR of the instruction that generated the exception are saved to the SPC and SSR, respectively. If the exception occurred during a read, H'040 is set in EXPEVT; if the exception occurred during a write, H'060 is set in EXPEVT. The BL, MD and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0400.

  To speed up TLB miss processing, the offset differs from other exceptions.

- TLB invalid exception
  - Conditions: Comparison of TLB addresses shows address match but V = 0.
  - Operations: The virtual address (32 bits) that caused the exception is set in TEA and the corresponding virtual page number (22 bits) is set in PTEH (31–10). The ASID of PTEH indicates the ASID at the time the exception occurred. The way that generated the exception is set in the RC bits in MMUCR.

  The PC and SR of the instruction that generated the exception are saved in the SPC and SSR, respectively. If the exception occurred during a read, H'040 is set in EXPEVT; if the exception occurred during a write, H'060 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0100.

- TLB exception/CPU address error in repeat loop
  - Conditions: TLB miss, TLB invalid or CPU address error in the last several instructions of repeat loop (see section 5.5.6, MMU Exception in Repeat Loop)
  - Operations: TEA, PTEH and RC bit in MMUCR are set in the way of the type of exception.

  The SR of the instruction that generated the exception are saved in the SSR. But the SPC is not the PC of the instruction that generated the exception. Repeat loop can not be restarted after returning from exception handler. In order to complete a repeat loop, ensure not to cause TLB exceptions or CPU address error in the last several instructions of repeat loop (see section 5.5.6, MMU Exception in Repeat Loop). If the TLB exception or CPU address error occurred in the last several instructions of repeat loop, H'070 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0100.

- Initial page write exception
  - — Conditions: A hit occurred to the TLB for a store access, but D = 0.
    This occurs for initial writes to the page registered by the load.
  - — Operations: The virtual address (32 bits) that caused the exception is set in TEA and the corresponding virtual page number (22 bits) is set in PTEH (31–10). The ASID of PTEH indicates the ASID at the time the exception occurred. The way that generated the exception is set in the RC bit in MMUCR.

  The PC and SR of the instruction that generated the exception are saved to the SPC and SSR, respectively. H'080 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs in PC = VBR + H'0100.

- TLB protection exception
  - — Conditions: When a hit access violates the TLB protection information (PR bits) shown below:

PR	Privileged mode	User mode
00	Only read enabled	No access
01	Read/write enabled	No access
10	Only read enabled	Only read enabled
11	Read/write enabled	Read/write enabled

- Operations: The virtual address (32 bits) that caused the exception is set in TEA and the corresponding virtual page number (22 bits) is set in PTEH (31–10). The ASID of PTEH indicates the ASID at the time the exception occurred. The way that generated the exception is set in the RC bits in MMUCR.

  The PC and SR of the instruction that generated the exception are saved to the SPC and SSR, respectively. If the exception occurred during a read, H'0A0 is set in EXPEVT; if the exception occurred during a write, H'0C0 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0100.

- TLB protection violation in repeat loop
  - — Conditions: TLB protection violation in the last several instruction of repeat loop (see section 5.5.6, MMU Exception in Repeat Loop)
  - — Operations: TEA, PTEH and RC bit in MMUCR are set in the way of the type of exception.

The SR of the instruction that generated the exception are saved in the SSR. But the SPC is not the PC of the instruction that generated the exception. Repeat loop can not be restarted after returning from exception handler. In order to complete a repeat loop, ensure not to cause TLB exceptions or CPU address error in the last several instructions of repeat loop (see section 5.5.6, MMU Exception in Repeat Loop). If the TLB exception or CPU address error occurred in the last several instructions of repeat loop, H'070 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0100.

- CPU Address error
  - — Conditions:
  - a. Instruction fetch from odd address (4n + 1, 4n + 3)
  - b. Word data accessed from addresses other than word boundaries (4n + 1, 4n + 3)
  - c. Longword accessed from addresses other than longword boundaries (4n + 1, 4n + 2, 4n + 3)
  - d. Virtual space accessed in user mode in the area H'80000000 to H'FFFFFFFF.
  - — Operations: The virtual address (32 bits) that caused the exception is set in TEA. The PC and SR of the instruction that generated the exception are saved to the SPC and SSR, respectively. If the exception occurred during a read, H'0E0 is set in EXPEVT; if the exception occurred during a write, H'100 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0100.

- Unconditional trap
  - — Conditions: TRAPA instruction executed
  - — Operations: The exception is a processing-completion type, so the PC of the instruction after the TRAPA instruction is saved to the SPC. SR from the time when the TRAPA instruction was executing is saved to SSR. The 8-bit immediate value in the TRAPA instruction is quadrupled and set in TRA (9–0). H'160 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0100.

- Reserved instruction exception
  - — Conditions:
  - a. When undefined code not in a delay slot is decoded

    Delay branch instructions: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT/S, BF/S
  - b. 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.
  - c. When a DSP instruction not in a delay slot is decoded without DSP extension (SR.DSP=0)

DSP instructions: LDS Rm, DSR/A0/X0/X1/Y0/Y1, LDS.L @Rm+, DSR/A0/X0/X1/Y0/Y1, STS DSR/A0/X0/X1/Y0/Y1, Rn, STS.L DSR/A0/X0/X1/Y0/Y1, @-Rn, LDC Rm, RS/RE/MOD, LDC.L @Rm+, RS/RE/MOD, STC RS/RE/MOD, Rn, STC.L RS/RE/MOD, @-Rn, LDRS, LDRE, SETRC, MOVS, MOVX, MOVY, Pxxx

d. When an instruction that rewrites the PC/SR/RS/RE in the last three instructions of repeat loop is decoded.

Instructions that rewrite the PC: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT, BF, BT/S, BF/S, TRAPA, LDC Rm, SR, LDC.L @Rm+, SR

Instructions that rewrite the SR: LDC Rm, SR, LDC.L @Rm+, SR, SETRC

Instructions that rewrite the RS: LDC Rm, RS, LDC.L @Rm+, RS, LDRS

Instructions that rewrite the RE: LDC Rm, RE, LDC.L @Rm+, RE, LDRE

— Operations: The PC and SR of the instruction that generated the exception are saved to the SPC and SSR, respectively. H'180 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0100. When an undefined instruction other than H'Fxxx is decoded, operation cannot be guaranteed.

- Illegal slot instruction

— Conditions:

a. When undefined code in a delay slot is decoded

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

b. When an instruction that rewrites the PC in a delay slot is decoded

Instructions that rewrite the PC: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT, BF, BT/S, BF/S, TRAPA, LDC Rm, SR, LDC.L @Rm+, SR

c. 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.

d. When a DSP instruction in a delay slot is decoded without DSP extension (SR.DSP=0)

DSP instructions: LDS Rm, DSR/A0/X0/X1/Y0/Y1, LDS.L @Rm+, DSR/A0/X0/X1/Y0/Y1, STS DSR/A0/X0/X1/Y0/Y1, Rn, STS.L DSR/A0/X0/X1/Y0/Y1, @-Rn, LDC Rm, RS/RE/MOD, LDC.L @Rm+, RS/RE/MOD, STC RS/RE/MOD, Rn, STC.L RS/RE/MOD, @-Rn, LDRS, LDRE, SETRC, MOVS, MOVX, MOVY, Pxxx

— Operations: The PC of the previous delay branch instruction is saved to the SPC. SR of the instruction that generated the exception is saved to SSR. H'1A0 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0100. When an undefined instruction other than H'Fxxx is decoded, operation cannot be guaranteed.

- User break point trap

— Conditions: When a break condition set in the user break point controller is satisfied

— Operations: When a post-execution break occurs, the PC of the instruction immediately after the instruction that set the break point is set in the SPC. If a pre-execution break occurs, the PC of the instruction that set the break point is set in the SPC. SR when the break occurs is set in SSR. H'1E0 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0100. See section 10, User Break Controller, for more information.

- DMA Address error

— Conditions:

a. Word data accessed from addresses other than word boundaries (4n + 1, 4n + 3)

b. Longword accessed from addresses other than longword boundaries (4n + 1, 4n + 2, 4n + 3)

— Operations: The PC of the instruction immediately after the instruction executed before the exception occurs is saved to the SPC. SR when the exception occurs is saved to SSR. H'5C0 is set in EXPEVT. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to PC = VBR + H'0100.

6.5.3 Interrupts

1. NMI

Conditions: NMI pin edge detection

Operations: The PC and SR after the instruction that receives the interrupt are saved to the SPC and SSR, respectively. H'01C0 is set to INTEVT. The BL, MD, and RB bits of the SR are set to 1 and a branch occurs to PC = VBR + H'0600. This interrupt is not masked by SR.IMASK and received with top priority when the SR's BL bit in SR is 0. When the BL bit is 1, the interrupt is masked. When BLMSK in ICRI is a logic zero and not masked when BLMSK in ICRI is a logic one. See section 9, Interrupt Controller, for more information.

2. IRL Interrupts

Conditions: The value of the interrupt mask bits in SR is lower than the IRL3–IRL0 level and the BL bit in SR is 0. The interrupt is accepted at an instruction boundary.

Operations: The PC after the instruction that accepts the interrupt is saved to the SPC. SR at the time the interrupt is accepted is saved to SSR. The code corresponding to the IRL3–IRL0 level is set in INTEVT. The corresponding code is given as H'200 + B' (IRL3–IRL0) × H'20. The BL, MD, and RB bits in SR are set to 1 and a branch occurs to VBR + H'0600. The received level is not set in SR.IMASK. See section 9, Interrupt Controller, for more information.

3. IRQ Pin Interrupts

Conditions: IRQ pin is asserted and SR.IMASK is lower than the IRQ priority level and the BL bit in SR is 0. The interrupt is accepted at an instruction boundary.

Operations: The PC after the instruction that accepts the interrupt is saved to the SPC. The SR at the point the interrupt is accepted is saved to the SSR. The code corresponding to the interrupt source is set to INTEVT and INTEVT2. The BL, MD, and RB bits of the SR are set to 1 and a branch occurs to VBR + H'0600. The received level is not set to SR.IMASK. See section 9, Interrupt Controller, for more information.

4. PINT Pin Interrupts

   Conditions: The PINT pin is asserted and SR.IMASK is lower than the PINT priority level and the BL bit in SR is 0. The interrupt is accepted at an instruction boundary.

   Operations: The PC after the instruction that accepts the interrupt is saved to the SPC. The SR at the point the interrupt is accepted is saved to the SSR. The code corresponding to the interrupt source is set to INTEVT and INTEVT2. The BL, MD, and RB bits of the SR are set to 1 and a branch occurs to VBR + H'0600. The received level is not set to SR.IMASK. See section 9, Interrupt Controller, for more information.

5. On-Chip Module Interrupts

   Conditions: SR.IMASK is lower than the on-chip module (TMU, RTC, SCI0, SCI1, SCI2, A/D, LCDC, PCC, DMAC, CPG, REF) interrupt level and the BL bit in SR is 0. The interrupt is accepted at an instruction boundary.

   Operations: The PC after the instruction that accepts the interrupt is saved to the SPC. The SR at the point the interrupt is accepted is saved to the SSR. The code corresponding to the interrupt source is set to INTEVT and INTEVT2. The BL, MD, and RB bits of the SR are set to 1 and a branch occurs to VBR + H'0600. See section 9, Interrupt Controller, for more information.

6. H-UDI Interrupt

   Conditions: H-UDI interrupt command is input (see section 25.4.5, H-UDI Interrupt) and SR.IMASK is lower than 15 and the BL bit in SR is 0. The interrupt is accepted at an instruction boundary.

   Operations: The PC after the instruction that accepts the interrupt is saved to the SPC. The SR at the point the interrupt is accepted is saved to the SSR. H'5E0 is set to INTEVT and INTEVT2. The BL, MD, and RB bits of the SR are set to 1 and a branch occurs to VBR + H'0600. See section 9, Interrupt Controller, for more information.

6.6 Cautions

- Return from exception processing
  - Check the BL bit in SR with software. When the SPC and SSR have been saved to external memory, set the BL bit in SR to 1 before restoring them.
  - Issue an RTE instruction. Set the SPC in the PC and SSR in SR with the RTE instruction, branch to the SPC address, and return from exception processing.

- Operation when exception or interrupt occurs while SR.BL = 1
  - Interrupt: Acceptance is suppressed until the BL bit in SR is set to 0 by software. If there is a request and the reception conditions are satisfied, the interrupt is accepted after the execution of the instruction that sets the BL bit in SR to 0. During the sleep or standby mode, however, the interrupt will be accepted even when the BL bit in SR is 1.

    NMI is accepted when BLMSK in ICRI is 1.
  - Exception: No user break point trap will occur even when the break conditions are met. When one of the other exceptions occurs, a branch is made to the fixed address of the reset (H'A0000000). In this case, the values of the EXPEVT, SPC, and SSR registers are undefined.

- SPC when an Exception Occurs: The PC saved to the SPC when an exception occurs is as shown below:
  - Re-executing-type exceptions: The PC of the instruction that caused the exception is set in the SPC and re-executed after return from exception processing. If the exception occurred in a delay slot, however, the PC of the immediately prior delayed branch instruction is set in the SPC. If the condition of the conditional delayed branch instruction is not satisfied, the delay slot PC is set in SPC.
  - Completed-type exceptions and interrupts: The PC of the instruction after the one that caused the exception is set in the SPC. If the exception was caused by a delayed conditional instruction, however, the branch destination PC is set in SPC. If the condition of the conditional delayed branch instruction is not satisfied, the delay slot PC is set in SPC.

- Initial register values after reset
  - Undefined registers

    R0_BANK0/1–R7_BANK0/1, R8–R15, GBR, SPC, SSR, MACH, MACL, PR, RS, RE, MOD, A0, A0G, A1, A1G, M0, M1, X0, X1, Y0, Y1
  - Initialized registers

    VBR = H'00000000

    SR.MD = 1, SR.BL = 1, SR.RB = 1, SR.I3–SR.I0 = H'F. Other SR bits are undefined.

    PC = H'A0000000

    DSR = H'00000000

- Ensure that an exception is not generated at an RTE instruction delay slot, as operation is not guaranteed in this case.

  When the BL bit in the SRL register is set to 1, ensure that a TLB-related exception or address error does not occur at an LDC instruction that updates the SR register and the following instruction. This occurrence will be identified as multiple exceptions, and may initiate reset processing.

Section 7 Cache

7.1 Overview

7.1.1 Features

The cache specifications are listed in table 7.1.

Table 7.1 Cache Specifications

Parameter	Specification
Capacity	16 Kbytes
Structure	Instruction/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	P0, P1, P3, U0: Write-back/write-through selectable
Replacement method	Least-recently-used (LRU) algorithm

7.1.2 Cache Structure

The cache mixes data and instructions and uses a 4-way set associative system. It is composed of four ways (banks), 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 data section of the entry 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), with a total of 16 Kbytes in the cache as a whole (4 ways). Figure 7.1 shows the cache structure.

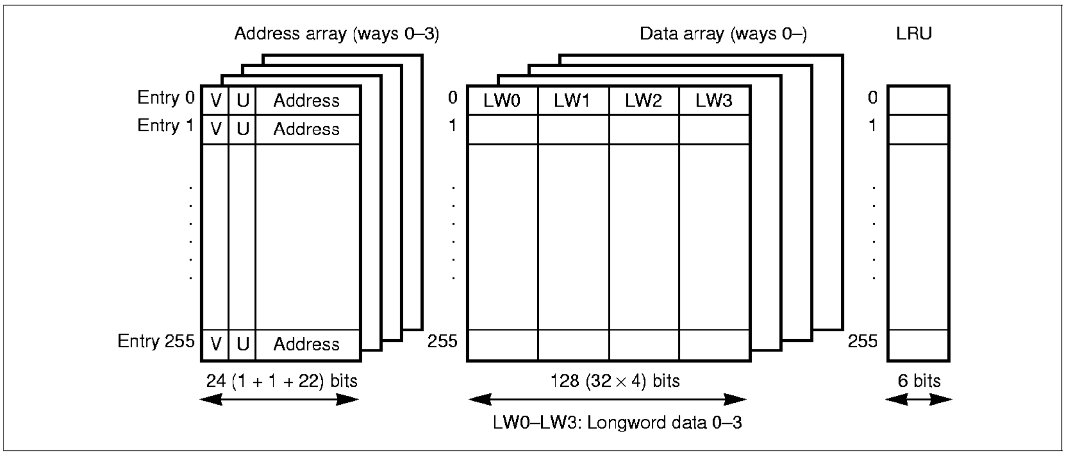


Figure 7.1 Cache Structure

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 address tag holds the physical address used in the external memory access. It is composed of 22 bits (address bits 31–10) used for comparison during cache searches.

In the SH7729, the top three of 32 physical address bits are used as shadow bits (see section 10, User Break Controller), and therefore in a normal replace operation the top three bits of the vector 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 (address bits 10–4) can be registered in the cache. When an entry is registered, the 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.

In normal operation, four ways are used as cache and six LRU bits indicate the way to be replaced (table 7.2). If a bit pattern other than those listed in table 7.2 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 7.2.

The LRU bits are initialized to 0 by a power-on reset, but are not initialized by a manual reset.

Table 7.2 LRU and Way Replacement (when lock is disabled)

LRU (5–0)	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

7.1.3 Register Configuration

Table 7.3 shows details of the cache control register.

Table 7.3 Register Configuration

Register	Abbr.	R/W	Size	Initial Value	Address
Cache control register	CCR	R/W	Longword	H'00000000	H'FFFFFFEC
Cache control register 2	CCR2	R/W	Longword	H’00000000	H’040000B0

7.2 Register Description

7.2.1 Cache Control Register (CCR)

The cache is enabled or disabled using the CE bit of the cache control register (CCR). CCR also has a CF bit (which invalidates all cache entries)*[1], and a WT bit (which selects either write-through mode or write-back mode*[2]). Programs that change the contents of the CCR register should be placed in address space that is not cached. When updating the contents of the CCR register, always set bits 4 to 0. Figure 7.2 shows the configuration of the CCR register.

Notes: 1. CB bit (which selects either write-through mode or write-back mode of P1)

2. P0, U0 and P3 modes

31		6	5	4	3	2	1	0
—	··· ··· ··· ··· ··· ··· ··· ··· ···	—	0	0	CF	CB	WT	CE

Bit 5, 4: Always set to 0 when setting the register.

CF: Cache flush bit. Invalidates all cache entries. 1 = flush (clears the V,U, and LRU bits of all entries to 0). Always reads 0. Write-back to external memory is not performed when the cache is flushed.

CB: Cache write-back bit. Indicates the cache's operating mode for area P1.
1 = write-back mode, 0 = write-through mode.

WT: Write-through bit. Indicates the cache's operating mode for area P0, U0 and P3.
1 = write-through mode, 0 = write-back mode.

CE: Cache enable bit. Indicates whether the cache function is used.
1 = cache used, 0 = cache not used.

Figure 7.2 CCR Register Configuration

7.2.2 Cache Control Register 2 (CCR2)

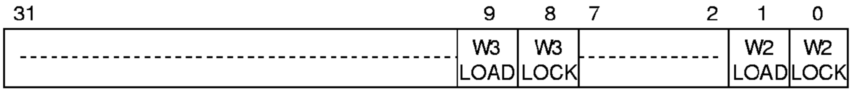
CCR2 register is used to enable or disable cache locking mechanism during DSP mode (CPU status register bit 12) only. Executing a prefetch instruction (PREF) during DSP mode will bring in one line size of data pointed by Rn to cache, according to the setting of CCR2 [9:8] (W3LOAD, W3LOCK) and [1:0] (W2LOAD, W2LOCK):

When CCR2[9:8]=11, during DSP mode PREF @Rn will bring the data into way 3. When CCR2[9:8]=00, 01 or 10 during DSP mode, or any setting during non-DSP mode, PREF @Rn will place the data into the way pointed by LRU.

When CCR2[1:0]=11, during DSP mode PREF @Rn will bring the data into way 2. When CCR2[1:0]=00, 01 or 10 during DSP mode, or any setting during non-DSP mode, PREF @Rn will place the data into the way pointed by LRU.

CCR2 must be set before cache is enabled (CCR.CE = 1).

Figure 7.3 shows the configuration of the CCR2 register.



W2LOCK: Way 2 lock bit. W2LOAD: Way 2 load bit.

When W2LOCK = 1 & W2LOAD = 1 & DSP = 1, the prefetched data will always be loaded into Way2. In all other conditions the prefetched data will be loaded into the way pointed by LRU.

W3LOCK: Way 3 lock bit. W3LOAD: Way 3 load bit.

When W3LOCK = 1 & W3LOAD = 1 & DSP = 1, the prefetched data will always be loaded into Way3. In all other conditions the prefetched data will be loaded into the way pointed by LRU.

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

Figure 7.3 CCR2 Register Configuration

Whenever CCR2 bit 8 (W3LOCK) or bit 0 (W2LOCK) is high the cache is locked. The locked data will not be overwritten unless W3LOCK bit and W2LOCK bit are reset or the PREF condition during DSP mode matched. During cache locking mode, the LRU in table 7.2 will be replaced by tables 7.4–7.6.

Table 7.4 LRU and Way Replacement (when W2LOCK=1)

LRU (5–0)	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 7.5 LRU and Way Replacement (when W3LOCK=1)

LRU (5–0)	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 7.6 LRU and Way Replacement (when W2LOCK=1 and W3LOCK=1)

LRU (5–0)	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

7.3 Cache Operation

7.3.1 Searching the Cache

If the cache is enabled, whenever instructions or data in memory are accessed the cache will be searched to see if the desired instruction or data is in the cache. Figure 7.4 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 11–4 of the address (virtual) of the access to memory and the address tag of that entry is read. In parallel to reading of the address tag, the virtual address is translated to a physical address in the MMU. The physical address after translation and the physical address read from the address section 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 7.4 shows a hit on way 1.

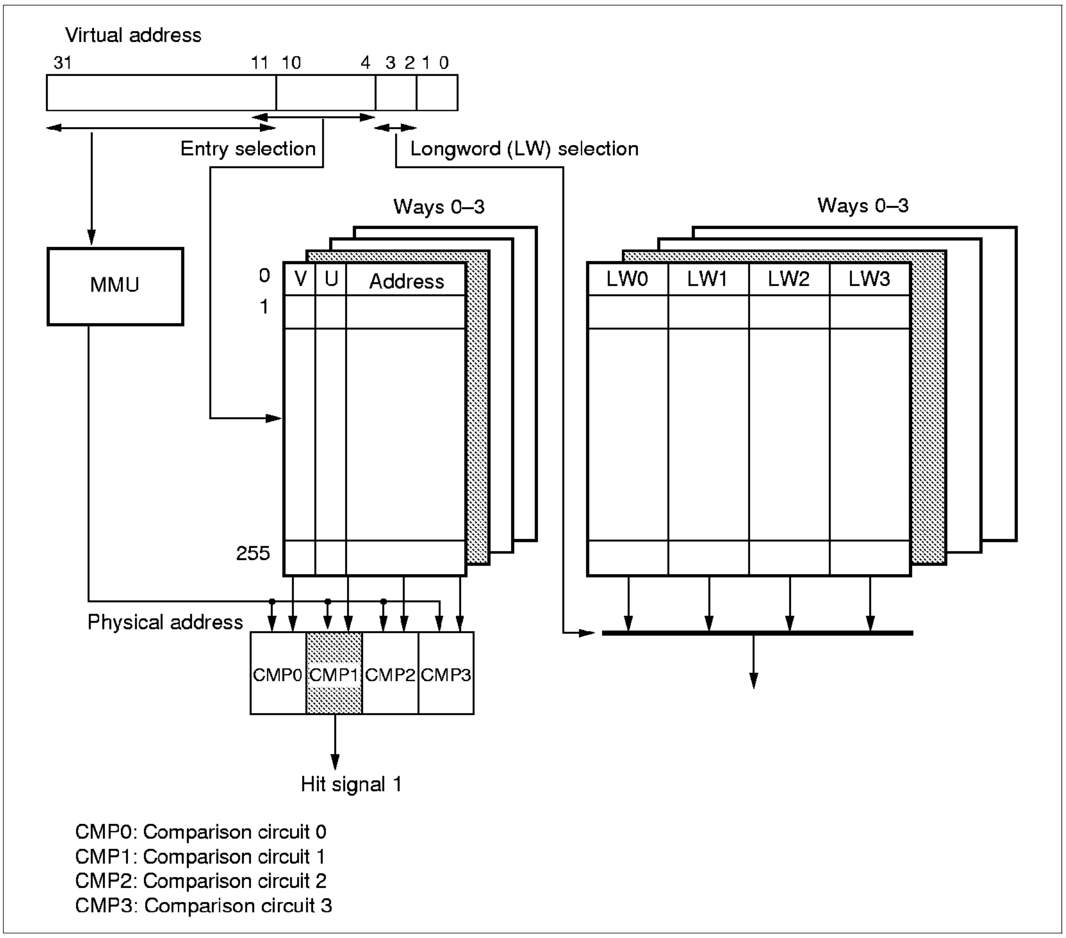


Figure 7.4 Cache Search Scheme

7.3.2 Read Access

Read Hit: In a read access, instructions and data are transferred from the cache to the CPU. The transfer unit is 32 bits. The LRU is updated.

Read Miss: An external bus cycle starts and the entry is updated. The way replaced is the one least recently used. 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 in the cache, the U bit is cleared to 0 and the V bit is set to 1. When the U bit of a replaced entry in write-back mode is 1, the cache fill cycle starts after the entry is transferred to the write-back buffer. After the cache completes its fill cycle, the write-back buffer writes back the entry to the memory. The write-back unit is 16 bytes.

7.3.3 Write Access

Write Hit: In a write access in the write-back mode, the data is written to the cache and the U bit of the entry written is set to 1. Writing occurs only to the cache; no external memory write cycle is issued. In the write-through mode, the data is written to the cache and an external memory write cycle is issued.

Write Miss: In the write-back mode, an external bus cycle starts when a write miss occurs and an entry with its U bit set to 1 is replaced. The way to be replaced is the one least recently used. When the U bit of the entry to be replaced is 1, the cache fill cycle starts after the entry is transferred to the write-back buffer. After the cache completes its fill cycle, the write-back buffer writes back the entry to the memory. The write-back unit is 16 bytes. Data is written to the cache and the U bit is set to 1. In the write-through mode, no write to cache occurs in a write miss; the write is only to the external memory.

7.3.4 Write-Back Buffer

When the U bit of the entry to be replaced in the write-back mode is 1, it 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. During the write back cycles, the cache can be accessed. The write-back buffer can hold one line of the cache data (16 bytes) and its physical address. Figure 7.5 shows the configuration of the write-back buffer.

PA (31–4)	Longword 0	Longword 1	Longword 2	Longword 3

PA (31–4): Physical address written to external memory
Longword 0–3: The line of cache data to be written to external memory

Figure 7.5 Write-Back Buffer Configuration

7.3.5 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 accessed, the latest data may be in a write-back mode cache, so invalidate the entry that includes the latest data in the cache, generate a write back, and update the data in memory before using it. When the caching area is updated by a device other than the SH7729, invalidate the entry that includes the updated data in the cache.

7.4 Memory-Mapped Cache

To allow software management of the cache, it is mapped onto virtual address space P4. The address array is mapped onto addresses H'F0000000 to H'F0FFFFFF and the data array onto addresses H'F1000000 to H'F1FFFFFF. In the privileged mode, the cache contents can be read or written using the MOV instruction.

7.4.1 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 address, V bit, U bit, and LRU bits to be written to the address array (figure 7.6 (1)).

In the address field, specify the entry address for selecting the entry (bits 11–4), W for selecting the way (bits 12–11: 00 is way 0, 01 is way 1, 10 is way 2, 11 is way 3), and H'F0 to indicate address array access (bits 31–24).

When writing, specify bit 2 as the A bit. The A bit indicates whether addresses are compared during writing. When the A bit is 1, the addresses of four entries selected by the entry addresses are compared to the addresses to be written into the address array specified in the data field. Writing takes place to the way that has a hit. When a miss occurs, nothing is written to the address array and no operation occurs. The way number (W) specified in bits 12–11 is not used. When the A bit is 0, it is written to the entry selected with the entry address and way number without comparing addresses. The address specified by bits 31–10 in the data specification in figure 7.6 (1), address array access, is a virtual address. When the MMU is enabled, the address is translated into a physical address, then the physical address is used in comparing addresses when the A bit is 1. The physical address is written into the address array.

When reading, the address tag, V bit, U bit, and LRU bits of the entry specified by the entry address and way number (W) are read using the data format shown in figure 7.6 without comparing addresses. To invalidate a specific entry, specify the entry by its entry address and way number, and write 0 to its V bit. To invalidate only an entry for an address to be invalidated, specify 1 for the A bit.

When an entry for which 0 is written to the V bit has a U bit set to 1, it will be written back. This allows coherency to be achieved between the external memory and cache by invalidating the entry. However, when 0 is written to the V bit, 0 must also be written to the U bit of that entry.

7.4.2 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 (figure 7.6 (2)).

In the address field, specify the entry address for selecting the entry (bits 11–4), L for indicating the longword position within the (16-byte) line (bits 3–2: 00 is longword 0, 01 is longword 1, 10 is longword 2, 11 is longword 3), W for selecting the way (bits 13–12: 00 is way 0, 01 is way 1, 10 is way 2, 11 is way 3), and H'F1 to indicate data array access (bits 31–24).

Both reading and writing use the longword of the data array specified by the entry address, way number and longword address. The access size of the data array is fixed at longword.

1. Address array access

Address specification

Read access

31 24	23 14	13 12	11 4	3	2	0
1111 0000	*·············*	W	Entry	0	*	*

Write access

31 24	23 14	13 12	11 4	3	2	0
1111 0000	*·············*	W	Entry	A	*	*

Data specification

31	30	29	10	9 4	3	2	1	0
0	0	0	Address tag (31–10)	LRU	X	X	U	V

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

Address specification

31 24	23 14	13 12	11 4	3 2	1 0
1111 0001	*·············*	W	Entry	L	* *

Data specification

31 0
Longword

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

Note: Address tag [31:29] are reserved for future use.
Programmer shall always write 0s to these bits.

Figure 7.6 Specifying Address and Data for Memory-Mapped Cache Access

7.5 Usage Examples

7.5.1 Invalidating Specific Entries

Specific cache entries can be invalidated by writing 0 to the entry's V bit. When the A bit is 1, the address tag specified by the write data is compared to the address tag within the cache selected by the entry address, and data is written when a match is found. If no match is found, there is no operation. R0 specifies the write data in R0 and R1 specifies the address. When the V bit of an entry in the address array is set to 0, the entry is written back if the entry's U bit is 1.

```
; R0=H'01100010; VPN=B'0000 0001 0001 0000 0000 00, U=0, V=0
; R1=H'F0000088; address array access, entry=B'00001000, A=1
;
MOV.L R0,@R1
```

7.5.2 Reading the Data of a Specific Entry

This example reads the data section of a specific cache entry. The longword indicated in the data field of the data array in figure 7.6 is read to the register. R0 specifies the address and R1 is read.

```
; R1=H'F100 004C; data array access, entry=B'00000100, Way = 0,
; longword address = 3
;
MOV.L @R0,R1 ; Longword 3 is read.
```

Section 8 X/Y Memory

8.1 Overview

The SH7729 has on-chip X-RAM and Y-RAM. It can be used by both CPU and DSP to store instructions or data.

8.1.1 Features

The X/Y Memory specifications are listed in table 8.1.

Table 8.1 X/Y Memory Specifications

Parameter	Specification
Addressing method	User selectable mapping mechanism • Fixed mapping for mission-critical realtime applications (P2/Uxy area) • Automatic mapping through TLB for easy to use (P0/P3/U0 area)
Ports	3 independent read/write ports • 8-/16-/32-bit access from the CPU • Maximum two 16-bit access from the DSP • 8-/16-/32-bit access from the DMAC
Size	8-Kbytes RAM for X and Y memory each

8.2 X/Y Memory Access from the CPU

The X/Y memory can be located in either map-enabled area or fixed-mapped area, depending on the mode bit (MD) and DSP bit (DSP) setting in the status register (SR). Figure 8.1 shows X/Y memory logical mapping.

1. Privileged Mode

   MD = 1, DSP = 0; Any virtual address in space P0 or P3 can map to X/Y memory through TLB translation. Addresses ranging from H'A500 0000 to H'A5FF FFFF in the P2 space can also fixed map to X/Y memory. Since the DSP extension is disabled, the DSP instruction set and registers are not available to the programmer.

2. User Mode

   MD = 0, DSP = 0; Any address in the U0 space can access X/Y memory through TLB translation. Any access to addresses beyond the U0 space will cause an address error. Since the DSP extension is disabled, the DSP instruction set and registers are not available to the programmer.

3. Privileged-DSP Mode

MD = 1, DSP = 1; Any virtual address in space P0 or P3 can map to X/Y memory through TLB translation. Addresses ranging from H'A500 0000 to H'A5FF FFFF in the P2 space can also fixed-map to X/Y memory. Since the DSP extension is enabled, the DSP instruction set and registers are available to the programmer.

4. User-DSP Mode

   MD = 0, DSP = 1; Any virtual address in space U0 can map to X/Y memory through TLB translation. Addresses ranging from H'A500 0000 to H'A5FF FFFF in the Uxy spaces can also fixed map to X/Y memory. Any access to outside of U0 and Uxy space will cause an address error. Since the DSP extension is enabled, the DSP instruction set and registers are available to the programmer.

It is recommended that for the mappable area, the C (cacheable) bit in the TLB entry must be set to 0 to guarantee a two-cycle access.

Mapping through TLB translation provides a flexible X/Y memory addressing scheme but takes two cycles even when the C bit in the TLB entry is set to 0. Fixed mapping provides a one-cycle access for read and two-cycle access for write, which is the appropriate method for mission-critical realtime operations.

The X/Y memory resides on the second 16 MB of physical address space area 1, from H'A500 0000 to H'A5FF FFFF. These 16-MB address spaces are shadowed and maps to the same 128-KB X/Y ROM/RAM. Figures 8.1 and 8.2 show X/Y memory physical mapping.

8.3 X/Y Memory Access from the DSP

The X/Y memory can be accessed by the DSP through the X bus and Y bus. Each access is 16-bit unit.

8.4 X/Y Memory Access from the DMAC

The X/Y memory also exists on the I bus and can be accessed by the DMAC. The DMAC access is 8-/16-/32-bit unit. If the I bus accesses X/Y memory simultaneously with an access from X bus/Y bus or L bus, the I bus master has a higher priority.

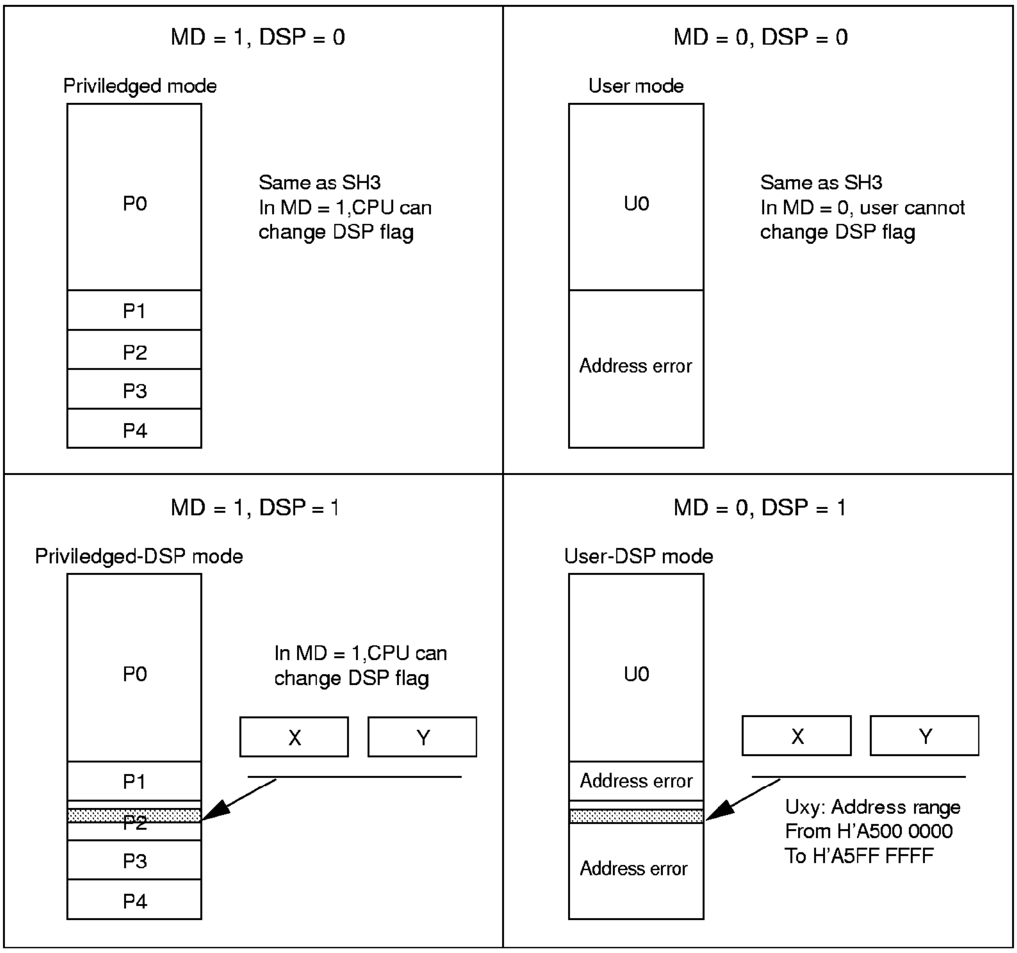


Figure 8.1 X/Y Memory Logical Address Mapping

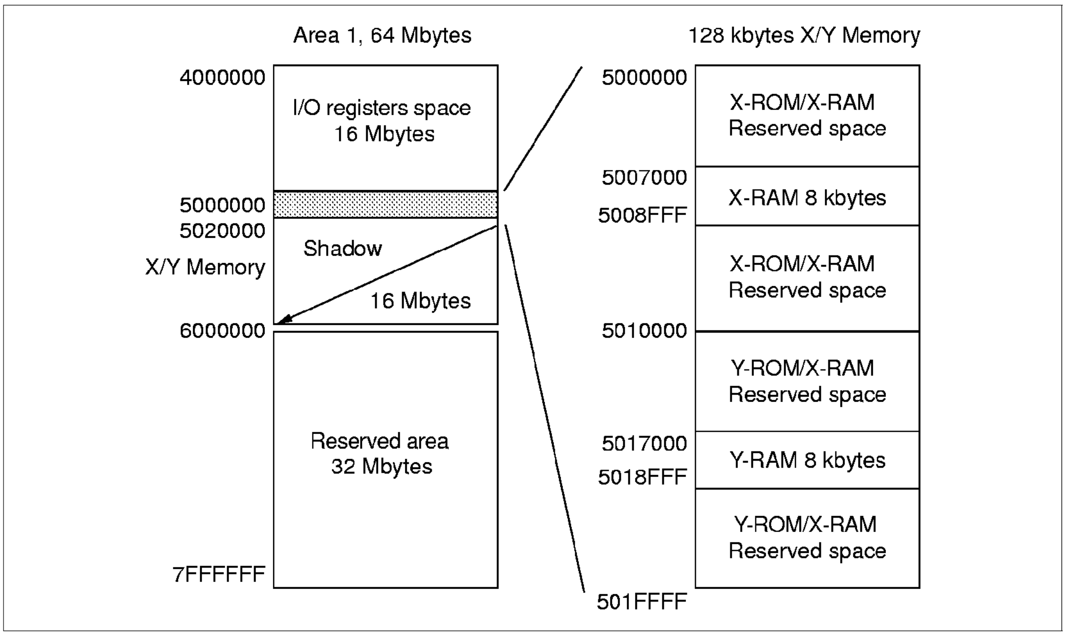


Figure 8.2 X/Y Memory Physical Address Mapping

Section 9 Interrupt Controller (INTC)

9.1 Overview

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.

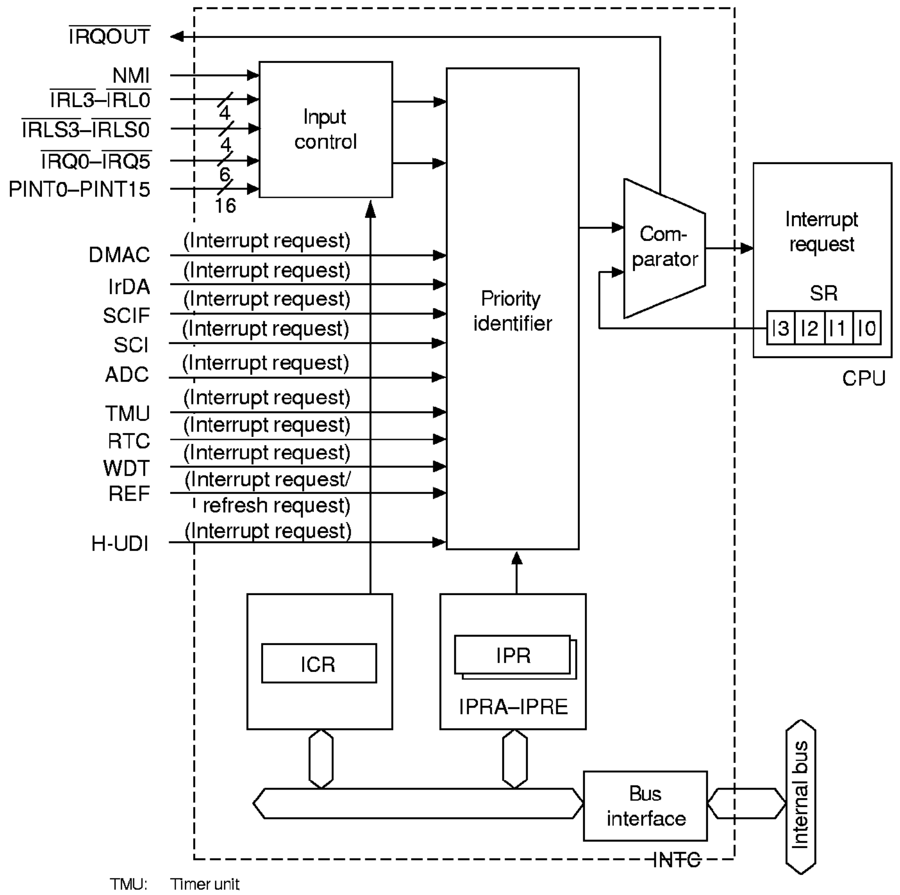
9.1.1 Features

INTC has the following features:

- 16 levels of interrupt priority can be set: By setting the five interrupt-priority registers, the priorities of on chip peripheral module interrupts can be selected from 16 levels for different request sources.
- NMI noise canceler function: NMI input-level bit indicates NMI pin states. 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.
- External devices can be notified that an interrupt has been received (IRQOUT): For example, when the SH7729 has released the bus right, the external bus master can be notified that an external interrupt, an on-chip peripheral module interrupt or a memory refresh request has occurred, enabling this LSI to request the bus right.

9.1.2 Block Diagram

Figure 9.1 is a block diagram of the INTC.



TMU:	Timer unit
RTC:	Realtime clock unit
SCI:	Serial communication interface
IrDA:	Serial communication interface (with IrDA)
SCIF:	Serial communication interface (with FIFO)
WDT:	Watchdog timer
REF:	Refresh requests in the bus state controller
ICR:	Interrupt control register
IPRA-IPRE:	Registers A-E for setting the interrupt proprity levels
SR:	Status register
DMAC:	Direct memory access controller
ADC:	Analog-to-digital converter
H-UDI:	Hitachi user-debugging interface

Figure 9.1 INTC Block Diagram

9.1.3 Pin Configuration

Table 9.1 lists the INTC pin configuration.

Table 9.1 Pin Configuration

Name	Abbreviation	I/O	Description
Nonmaskable interrupt input pin	NMI	I	Input of interrupt request signal, which is nonmaskable by SR.IMASK
Interrupt input pins	IRQ5–IRQ0/ $\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ $\overline{\text{IRLS3}}$-$\overline{\text{IRLS0}}$	I	Input of interrupt request signals, which is maskable by SR.IMASK
Port interrupt input pins	PINT0–PINT15	I	Port input of interrupt request signals, which is maskable by SR.IMASK
Interrupt request output pin	$\overline{\text{IRQOUT}}$	O	Output of signal that notifies external devices that an interrupt source or memory refresh has occurred

9.1.4 Register Configuration

The INTC has 12 registers listed in table 9.2.

Table 9.2 Register Configuration

Name	Abbr.	R/W	Initial Value*1	Address	Access Size
Interrupt control register 0	ICR0	R/W	*2	H'FFFFFEE0	16
Interrupt control register 1	ICR1	R/W	H'0000	H'4000010	16
Interrupt control register 2	ICR2	R/W	H'0000	H'4000012	16
PINT interrupt enable register	PINTER	R/W	H'0000	H'4000014	16
Interrupt priority level setting register A	IPRA	R/W	H'0000	H'FFFFFEE2	16
Interrupt priority level setting register B	IPRB	R/W	H'0000	H'FFFFFEE4	16
Interrupt priority level setting register C	IPRC	R/W	H'0000	H'4000016	16
Interrupt priority level setting register D	IPRD	R/W	H'0000	H'4000018	16
Interrupt priority level setting register E	IPRE	R/W	H'0000	H'400001A	16
Interrupt request register 0	IRR0	R/W	H'00	H'4000004	8
Interrupt request register 1	IRR1	R	H'00	H'4000006	8
Interrupt request register 2	IRR2	R	H'00	H'4000008	8

Notes: 1. Initialized by a power-on or manual reset.

2. H'8000 when the NMI pin is at high level. H'0000 when the NMI pin is at low level.

9.2 Interrupt Sources

There are five types of interrupt sources: NMI, IRQ, IRL,PINT, and on-chip peripheral modules. Each interrupt has priority levels (0–16) with 0 the lowest and 16 the highest. Priority level 0 masks an interrupt.

9.2.1 NMI Interrupts

The NMI interrupt has the highest priority level of 16. When the BLMSK bit of the interrupt control register (ICR1) is 1 or the BL bit of the status register (SR) is 0, NMI interrupts are accepted when the MAI bit of the ICR1 register is 0. NMI interrupts are edge-detected. In sleep or standby mode, the interrupt is accepted regardless of the BL. The NMI edge select bit (NMIE) in the interrupt control register 0 (ICR0) is used to select either the rising or falling edge. When the NMIE bit of the ICR0 register is changed, the NMI interrupt is not detected for 20 cycles after changing the ICR.NMIE to avoid a false detection of the NMI interrupt. NMI interrupt exception processing does not affect the interrupt mask level bits (I3–I0) in the status register (SR).

When the BLMSK bit of the ICR1 register is set to 1 and only NMI interrupts are accepted, the SPC register and SSR register are updated by the NMI interrupt handler, making it impossible to return to the original processing from exception processing initiated prior to the NMI. Use should therefore be restricted to cases where return is not necessary.

It is possible to wake the chip up from the standby state with an NMI interrupt (except when the MAI bit of the ICR1 register is set to 1).

9.2.2 IRQ Interrupt

IRQ interrupts are input by priority from pins IRQ0–IRQ5 with a level or an edge. The priority level can be set by priority setting registers C–D (IPRC–IPRD) in a range from levels 0–15.

To clear IRQ interrupt input with an edge, write 0 to the corresponding bits in IRR0 after reading 1.

When the ICR1 register is rewritten, IRQ interrupts may be mistakenly detected, depending on the 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 interrupt request register 0 (IRR0).

It is necessary for an edge input interrupt detection to input a pulse width more than two-cycle width by P clock basis.

The interrupt mask bits (I3–I0) of the status register (SR) are not affected by IRQ interrupt processing.

Interrupts IRQ4–IRQ0 can wake the chip up from the standby state when the relevant interrupt level is higher than I3–I0 in the SR register (but only when the RTC 32-kHz oscillator is used).

9.2.3 IRL Interrupts

IRL interrupts are input by level at pins IRL3–IRL0 and IRLS3–IRLS0. IRLS3–IRLS0 is enabled when IRQLVL bit and IRLSEN bit in interrupt control register 1 (ICR1) are both 1. The priority level is the higher level indicated by pins IRL3–IRL0 and IRLS3–IRLS0. An IRL3–IRL0/IRLS3–IRLS0 value of 0 (0000) indicates the highest-level interrupt request (interrupt priority level 15). A value of 15 (1111) indicates no interrupt request (interrupt priority level 0). Figure 9.2 shows an examples of an IRL interrupt connection. Table 9.3 shows IRL/IRLS pins and interrupt levels.

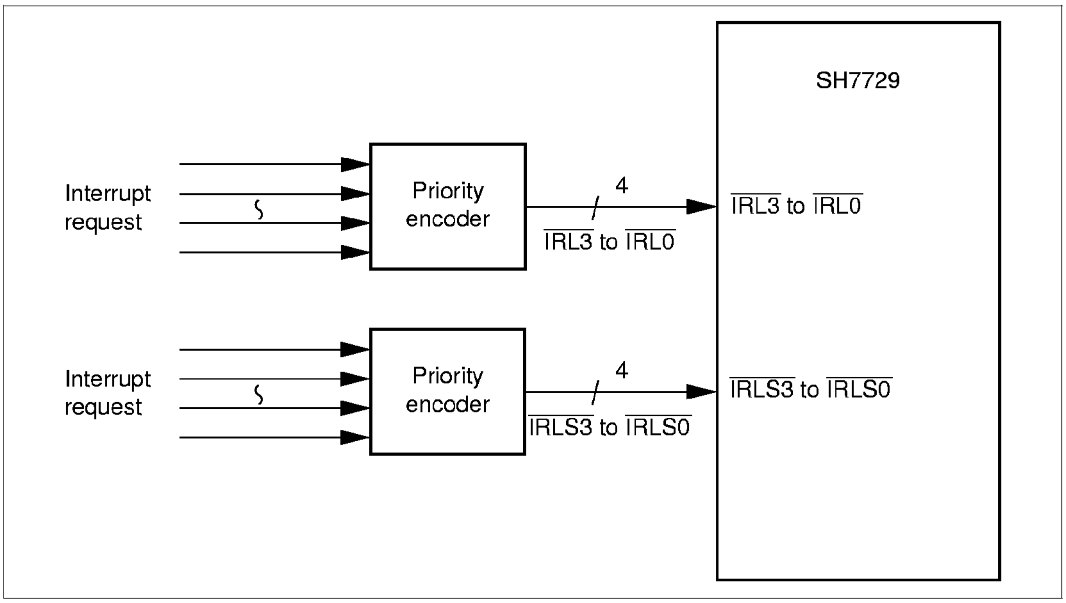


Figure 9.2 Example of IRL Interrupt Connection

Table 9.3 $\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$/$\overline{\text{IRLS3}}$–$\overline{\text{IRLS0}}$ Pins and Interrupt Levels

$\overline{\text{IRL3}}$/ $\overline{\text{IRLS3}}$	$\overline{\text{IRL2}}$/ $\overline{\text{IRLS2}}$	$\overline{\text{IRL1}}$/ $\overline{\text{IRLS1}}$	$\overline{\text{IRL0}}$/ $\overline{\text{IRLS0}}$	Interrupt Priority Level	Interrupt Request
0	0	0	0	15	Level 15 interrupt request
0	0	0	1	14	Level 14 interrupt request
0	0	1	0	13	Level 13 interrupt request
0	0	1	1	12	Level 12 interrupt request
0	1	0	0	11	Level 11 interrupt request
0	1	0	1	10	Level 10 interrupt request
0	1	1	0	9	Level 9 interrupt request
0	1	1	1	8	Level 8 interrupt request
1	0	0	0	7	Level 7 interrupt request
1	0	0	1	6	Level 6 interrupt request
1	0	1	0	5	Level 5 interrupt request
1	0	1	1	4	Level 4 interrupt request
1	1	0	0	3	Level 3 interrupt request
1	1	0	1	2	Level 2 interrupt request
1	1	1	0	1	Level 1 interrupt request
1	1	1	1	0	No interrupt request

A noise-cancellation feature is built in, and the IRL interrupt is not detected unless the levels sampled at every supporting module cycle remain unchanged for two consecutive cycles, so that no transient level on the $\overline{\text{IRL}}$/$\overline{\text{IRLS}}$ pin change is detected. In the standby mode, as the peripheral clock is stopped, noise cancellation is performed using the 32-kHz clock for the RTC instead. Therefore when the RTC is not used, interruption by means of IRL interrupts cannot be performed in standby mode.

The priority level of the IRL interrupt must not be lowered unless the interrupt is accepted and the interrupt processing starts. However, the priority level can be changed to a higher one.

The interrupt mask bits (I3–I0) in the status register (SR) are not affected by $\overline{\text{IRL}}$/$\overline{\text{IRLS}}$ interrupt processing.

9.2.4 PINT Interrupt

PINT interrupts are input by priority from pins PINT0–PINT15 with a level. The priority level can be set by priority setting registers D (IPRD) in a range from levels 0–15, in the unit of PINT0–PINT7 or PINT8–PINT15.

The interrupt mask bits (I3–I0) of the status register (SR) are not affected by PINT interrupt processing.

PINT interrupts can wake the chip up from the standby state when the relevant interrupt level is higher than I3–I0 in the SR register.

9.2.5 On-Chip Peripheral Module Interrupts

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

- Timer unit (TMU)
- Realtime clock (RTC)
- Serial communication interface (SCI, IrDA, SCIF)
- Bus state controller (BSC)
- Watchdog timer (WDT)
- Direct memory access controller (DMAC)
- Analog-to-digital converter (ADC)
- User-debugging interface (H-UDI)

Not every interrupt source is assigned a different interrupt vector. Sources are reflected on the interrupt event register (INTEVT and INTEVT2). It is easy to identify sources by using the values of the INTEVT or INTEVT2 register as branch offsets (in the exception service routine).

The priority level (from 0–15) can be set for each module except for H-UDI by writing to the interrupt priority setting registers A, B and E (IPRA, IPRB and IPRE). The priority level of H-UDI interrupt is 15 (fixed).

The interrupt mask bits (I3–I0) of the status register are not affected by the on-chip peripheral module interrupt processing.

TMU and RTC interrupts can wake the chip up from the standby state when the relevant interrupt level is higher than I3–I0 in the SR register (but only when the RTC 32-kHz oscillator is used).

9.2.6 Interrupt Exception Processing and Priority

Table 9.4 lists the codes for the interrupt event register (INTEVT and INTEVT2), and the order of interrupt priority. Each interrupt source is assigned unique code. The start address of the interrupt service routine is common to each interrupt source. This is why, for instance, the value of INTEVT or INTEVT2 is used as offset at the start of the interrupt service routine and branched to identify the interrupt source.

The order of priority of the on-chip peripheral module is set within the priority levels 0–15 at will by using the interrupt priority level set to registers A–E (IPRA–IPRE). The order of priority of the on-chip peripheral module is set to zero by RESET.

When the order of priorities for multiple interrupt sources are set to the same level and such interrupts are generated at the same time, they are processed according to the default order listed in tables 9.4 and 9.5.

Table 9.4 Interrupt Exception Handling Sources and Priority (IRQ Mode)

Interrupt Source		INTEVT Code (INTEVT2 Code)	Interrupt Priority (Initial Value)	IPR (Bit Numbers)	Priority within IPR Setting Unit	Default Priority
NMI		0x1C0 (0x1C0)	16	—	—	High
H-UDI		0x5E0 (0x5E0)	15	—	—	
IRQ	IRQ0	0x200–3C0* (0x600)	0–15 (0)	IPRC (3–0)	—	
	IRQ1	0x200–3C0* (0x620)	0–15 (0)	IPRC (7–4)	—	
	IRQ2	0x200–3C0* (0x640)	0–15 (0)	IPRC (11–8)	—	
	IRQ3	0x200–3C0* (0x660)	0–15 (0)	IPRC (15–12)	—	
	IRQ4	0x200–3C0* (0x680)	0–15 (0)	IPRD (3–0)	—	
	IRQ5	0x200–3C0* (0x6A0)	0–15 (0)	IPRD (7–4)	—	
PINT	PINT0-7	0x200–3C0* (0x700)	0–15 (0)	IPRD (15–12)	—	
	PINT8-15	0x200–3C0* (0x720)	0–15 (0)	IPRD (11–8)	—	
DMAC	DEI0	0x200–3C0* (0x800)	0–15 (0)	IPRE (15–12)	High	
	DEI1	0x200–3C0* (0x820)				
	DEI2	0x200–3C0* (0x840)				
	DEI3	0x200–3C0* (0x860)			Low	
IrDA	ERI1	0x200–3C0* (0x880)	0–15 (0)	IPRE (11–8)	High	
	RXI1	0x200–3C0* (0x8A0)				
	BRI1	0x200–3C0* (0x8C0)				
	TXI1	0x200–3C0* (0x8E0)			Low	
SCIF	ERI2	0x200–3C0* (0x900)	0–15 (0)	IPRE (7–4)	High	
	RXI2	0x200–3C0* (0x920)				
	BRI2	0x200–3C0* (0x940)				
	TXI2	0x200–3C0* (0x960)			Low	
ADC	ADI	0x200–3C0* (0x980)	0–15 (0)	IPRE (3–0)	—	
TMU0	TUNI0	0x400 (0x400)	0–15 (0)	IPRA (15–12)	—	
TMU1	TUNI1	0x420 (0x420)	0–15 (0)	IPRA (11–8)	—	
TMU2	TUNI2	0x440 (0x440)	0–15 (0)	IPRA (7–4)	High	
	TICPI2	0x460 (0x460)			Low	Low

Table 9.4 Interrupt Exception Handling Sources and Priority (IRQ Mode) (cont)

Interrupt Source		INTEVT Code (INTEVT2 Code)	Interrupt Priority (Initial Value)	IPR (Bit Numbers)	Priority within IPR Setting Unit	Default Priorit y
RTC	ATI	0x480 (0x480)	0–15 (0)	IPRA (3–0)	High	High
	PRI	0x4A0 (0x4A0)				
	CUI	0x4C0 (0x4C0)			Low	
SCI0	ERI	0x4E0 (0x4E0)	0–15 (0)	IPRB (7–4)	High	
	RXI	0x500 (0x500)				
	TXI	0x520 (0x520)				
	TEI	0x540 (0x540)			Low	
WDT	ITI	0x560 (0x560)	0–15 (0)	IPRB (15–12)	—	
REF	RCMI	0x580 (0x580)	0–15 (0)	IPRB (11–8)	High	
	ROVI	0x5A0 (0x5A0)			Low	Low

Note: * The code corresponding to an interrupt level shown in table 9.6 is set.

Table 9.5 Interrupt Exception Handling Sources and Priority (IRL Mode)

Interrupt Source		INTEVT Code (INTEVT2 Code)	Interrupt Priority (Initial Value)	IPR (Bit Numbers)	Priority within IPR Setting Unit	Default Priority
NMI		0x1C0 (0x1C0)	16	—	—	High
H-UDI		0x5E0 (0x5E0)	15	—	—	
IRL	IRL(3:0)*2 = 0000	0x200 (0x200)	15	—	—	
	IRL(3:0)*2 = 0001	0x220 (0x220)	14	—	—	
	IRL(3:0)*2 = 0010	0x240 (0x240)	13	—	—	
	IRL(3:0)*2 = 0011	0x260 (0x260)	12	—	—	
	IRL(3:0)*2 = 0100	0x280 (0x280)	11	—	—	
	IRL(3:0)*2 = 0101	0x2A0 (0x2A0)	10	—	—	
	IRL(3:0)*2 = 0110	0x2C0 (0x2C0)	9	—	—	
	IRL(3:0)*2 = 0111	0x2E0 (0x2E0)	8	—	—	
	IRL(3:0)*2 = 1000	0x300 (0x300)	7	—	—	
	IRL(3:0)*2 = 1001	0x320 (0x320)	6	—	—	
	IRL(3:0)*2 = 1010	0x340 (0x340)	5	—	—	
	IRL(3:0)*2 = 1011	0x360 (0x360)	4	—	—	
	IRL(3:0)*2 = 1100	0x380 (0x380)	3	—	—	
	IRL(3:0)*2 = 1101	0x3A0 (0x3A0)	2	—	—	
	IRL(3:0)*2 = 1110	0x3C0 (0x3C0)	1	—	—	
IRQ	IRQ4	0x200–3C0*1 (0x680)	0–15 (0)	IPRD (3–0)	—	
	IRQ5	0x200–3C0*1 (0x6A0)	0–15 (0)	IPRD (7–4)	—	
PINT	PINT0–7	0x200–3C0*1 (0x700)	0–15 (0)	IPRD (15–12)	—	
	PINT8–15	0x200–3C0*1 (0x720)	0–15 (0)	IPRD (11–8)	—	
DMAC	DEI0	0x200–3C0*1 (0x800)	0–15 (0)	IPRE (15–12)	High	
	DEI1	0x200–3C0*1 (0x820)				
	DEI2	0x200–3C0*1 (0x840)				
	DEI3	0x200–3C0*1 (0x860)			Low	
IrDA	ERI1	0x200–3C0*1 (0x880)	0–15 (0)	IPRE (11–8)	High	
	RXI1	0x200–3C0*1 (0x8A0)				
	BRI1	0x200–3C0*1 (0x8C0)				
	TXI1	0x200–3C0*1 (0x8E0)			Low	Low

Table 9.5 Interrupt Exception Handling Sources and Priority (IRL Mode) (cont)

Interrupt Source		INTEVT Code (INTEVT2 Code)	Interrupt Priority (Initial Value)	IPR (Bit Numbers)	Priority within IPR Setting Unit	Default Priority
SCIF	ERI2	0x200–3C0*1 (0x900)	0–15 (0)	IPRE (7–4)	High	High
	RXI2	0x200–3C0*1 (0x920)				
	BRI2	0x200–3C0*1 (0x940)				
	TXI2	0x200–3C0*1 (0x960)			Low	
ADC	ADI	0x200–3C0*1 (0x980)	0–15 (0)	IPRE (3–0)	—	
TMU0	TUNI0	0x400 (0x400)	0–15 (0)	IPRA (15–12)	—	
TMU1	TUNI1	0x420 (0x420)	0–15 (0)	IPRA (11–8)	—	
TMU2	TUNI2	0x440 (0x440)	0–15 (0)	IPRA (7–4)	High	
	TICPI2	0x460 (0x460)			Low	
RTC	ATI	0x480 (0x480)	0–15 (0)	IPRA (3–0)	High	
	PRI	0x4A0 (0x4A0)				
	CUI	0x4C0 (0x4C0)			Low	
SCI0	ERI	0x4E0 (0x4E0)	0–15 (0)	IPRB (7–4)	High	
	RXI	0x500 (0x500)				
	TXI	0x520 (0x520)				
	TEI	0x540 (0x540)			Low	
WDT	ITI	0x560 (0x560)	0–15 (0)	IPRB (15–12)	—	
REF	RCMI	0x580 (0x580)	0–15 (0)	IPRB (11–8)	High	
	ROVI	0x5A0 (0x5A0)			Low	Low

Notes: 1. The code corresponding to an interrupt level shown in table 9.6 is set.

2. When $\overline{\text{IRLS3}}$–$\overline{\text{IRLS0}}$ are enabled, IRL is higher level of $\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ and $\overline{\text{IRLS3}}$–$\overline{\text{IRLS0}}$.

Table 9.6 Interrupt Level and INTEVT Code

Interrupt level	INTEVT Code
15	H’200
14	H’220
13	H’240
12	H’260
11	H’280
10	H’2A0
9	H’2C0
8	H’2E0
7	H’300
6	H’320
5	H’340
4	H’360
3	H’380
2	H’3A0
1	H’3C0

9.3 INTC Registers

9.3.1 Interrupt Priority Registers A to E (IPRA–IPRE)

Interrupt priority registers A to E (IPRA to IPRE) are 16-bit read/write registers that set priority levels from 0 to 15 for on-chip peripheral module interrupts. These registers are initialized to H'0000 at power-on reset, manual reset, or in hardware standby mode, but is not initialized in standby mode.

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

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9.7 lists the relationship between the interrupt sources and the IPRA and IPRE bits.

Table 9.7 Interrupt Request Sources and IPRA–IPRE

Register	Bits 15 to 12	Bits 11 to 8	Bits 7 to 4	Bits 3 to 0
IPRA	TMU0	TMU1	TMU2	RTC
IPRB	WDT	REF	SCI0	Reserved*
IPRC	IRQ3	IRQ2	IRQ1	IRQ0
IPRD	PINT0 to PINT7	PINT8 to PINT15	IRQ5	IRQ4
IPRE	DMAC	IrDA	SCIF	ADC

Note: * Always read as 0. Only 0 should be written in.

As listed in table 9.7, four sets of on-chip peripheral modules are assigned to each register. 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 is priority level 15 (the highest level). A reset initializes IPRA–IPRE to H'0000.

H'0 should be set into bits corresponding to an unused interrupt.

9.3.2 Interrupt Control Register 0 (ICR0)

The ICR0 is a 16-bit register that sets the input signal detection mode of the external interrupt input pin NMI and indicates the input signal level to the NMI pin. This register is initialized to H'0000 at power-on reset or manual reset, but is not initialized in standby mode.

Bit:	15	14	13	12	11	10	9	8
Bit name:	NMIL	—	—	—	—	—	—	NMIE
Initial value:	0/1*	0	0	0	0	0	0	0
R/W:	R	—	—	—	—	—	—	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	—	—	—	—	—	—	—	—

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

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

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

Bit 8—NMI Edge Select (NMIE): Selects whether the falling or rising edge of the interrupt request signal to the NMI is detected.

Bit 8: NMIE	Description
0	Interrupt request is detected on the falling edge of NMI input
1	Interrupt request is detected on rising edge of NMI input

Bits 14–9 and 7–0—Reserved: Writing is invalid. Always read as 0.

9.3.3 Interrupt Control Register 1 (ICR1)

The ICR1 is a 16-bit register that specifies the detection mode to external interrupt input pins, IRQ0 to IRQ5 individually: rising edge, falling edge, or low level. This register, initialized to H'4000 at power-on reset or manual reset, is not initialized in the standby mode. Bits 15 and 13 must be cleared. Write 1 to these bits is inhibited.

Bit:	15	14	13	12	11	10	9	8
Bit name:	MAI	IRQLVL	BLMSK	IRLSEN	IRQ51S	IRQ50S	IRQ41S	IRQ40S
Initial value:	0	1	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	RW	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	IRQ31S	IRQ30S	IRQ21S	IRQ20S	IRQ11S	IRQ10S	IRQ01S	IRQ00S
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 15—Mask All Interrupts (MAI): Masks NMI interrupts when set to 1. Also selects whether or not all interrupt requests are masked when a low level is being input to the NMI pin.

Bit 15: MAI	Description	
0	All interrupt requests are not masked	(Initial value)
1	All interrupt requests are masked	

Bit 14—Interrupt Request Level Detect (IRQLVL): Selects whether the IRQ3–IRQ0 pins are used as four independent interrupt pins or as 15-level interrupt pins encoded as$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$.

Bit 14: IRQLVL	Description	
0	Used as four independent interrupt request pins IRQ3–IRQ0	
1	Used as encoded 15-level interrupt pins as $\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$	(Initial value)

Bit 13—BL Bit Mask (BLMSK): Specifies whether NMI interrupts are masked when the BL bit of the SR register is 1.

Bit 13: BLMSK	Description	
0	NMI interrupts are masked when the BL bit is 1	(Initial value)
1	NMI interrupts are accepted regardless of the BL bit setting	

Bit 12—$\overline{\text{IRLS}}$ Enable (IRLSEN): Enable $\overline{\text{IRLS3}}$–$\overline{\text{IRLS0}}$ pins. This bit is effective only when IRQLVL bit is 1.

Bit 12: IRLSEN	Description	
0	Disable $\overline{\text{IRLS3}}$–$\overline{\text{IRLS0}}$ pins	(Initial value)
1	Enable $\overline{\text{IRLS3}}$–$\overline{\text{IRLS0}}$ pins	

Bits 11 and 10—IRQ5 Sense Select (IRQ51S and IRQ50S): Select whether the interrupt signal to the IRQ5 pin is detected at the rising edge, at the falling edge, or at low level.

Bit 11: IRQ51S	Bit 10: IRQ50S	Description
0	0	An interrupt request is detected at IRQ5 input falling edge (Initial value)
	1	An interrupt request is detected at IRQ5 input rising edge
1	0	An interrupt request is detected at IRQ5 input low level
	1	Reserved

Bits 9 and 8—IRQ4 Sense Select (IRQ41S and IRQ40S): Select whether the interrupt signal to the IRQ4 pin is detected at the rising edge, at the falling edge, or at low level.

Bit 9: IRQ41S	Bit 8: IRQ40S	Description
0	0	An interrupt request is detected at IRQ4 input falling edge (Initial value)
	1	An interrupt request is detected at IRQ4 input rising edge
1	0	An interrupt request is detected at IRQ4 input low level
	1	Reserved

Bits 7 and 6—IRQ3 Sense Select (IRQ31S and IRQ30S): Select whether the interrupt signal to the IRQ3 pin is detected at the rising edge, at the falling edge, or at low level.

Bit 7: IRQ31S	Bit 6: IRQ30S	Description
0	0	An interrupt request is detected at IRQ3 input falling edge (Initial value)
	1	An interrupt request is detected at IRQ3 input rising edge
1	0	An interrupt request is detected at IRQ3 input low level
	1	Reserved

Bits 5 and 4—IRQ2 Sense Select (IRQ21S and IRQ20S): Select whether the interrupt signal to the IRQ2 pin is detected at the rising edge, at the falling edge, or at low level.

Bit 5: IRQ21S	Bit 4: IRQ20S	Description
0	0	An interrupt request is detected at IRQ2 input falling edge (Initial value)
	1	An interrupt request is detected at IRQ2 input rising edge
1	0	An interrupt request is detected at IRQ2 input low level
	1	Reserved

Bits 3 and 2—IRQ1 Sense Select (IRQ11S and IRQ10S): Select whether the interrupt signal to the IRQ1 pin is detected at the rising edge, at the falling edge, or at low level.

Bit 3: IRQ11S	Bit 2: IRQ10S	Description
0	0	An interrupt request is detected at IRQ1 input falling edge (Initial value)
	1	An interrupt request is detected at IRQ1 input rising edge
1	0	An interrupt request is detected at IRQ1 input low level
	1	Reserved

Bits 1 and 0—IRQ0 Sense Select (IRQ01S and IRQ00S): Select whether the interrupt signal to the IRQ0 pin is detected at the rising edge, at the falling edge, or at low level.

Bit 1: IRQ01S	Bit 0: IRQ00S	Description
0	0	An interrupt request is detected at IRQ0 input falling edge (Initial value)
	1	An interrupt request is detected at IRQ0 input rising edge
1	0	An interrupt request is detected at IRQ0 input low level
	1	Reserved

9.3.4 Interrupt Control Register 2 (ICR2)

The ICR2 is a 16-bit read/write register that sets the detection mode to external interrupt input pins PINT0 to PINT15. This register is initialized to H'0000 at power-on reset or manual reset, but is not initialized in software standby mode.

Bit:	15	14	13	12	11	10	9	8
Bit name:	PINT15S	PINT14S	PINT13S	PINT14S	PINT11S	PINT10S	PINT9S	PINT8S
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	PINT7S	PINT6S	PINT5S	PINT4S	PINT3S	PINT2S	PINT1S	PINT0S
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15–0—PINT15 to PINT0 Sense Select (PINT15S to PINT0S): Select whether interrupt request signals to PINT15 to PINT0 are detected at low levels or high levels.

Bits 15–0: PINT15S to PINT0S	Description
0	Interrupt requests are detected at low level input to the PINT pins (Initial value)
1	Interrupt requests are detected at high level input to the PINT pins

9.3.5 PINT Interrupt Enable Register (PINTER)

The PINTER is a 16-bit read/write register that enables interrupt requests input to external interrupt input pins PINT0 to PINT15. This register is initialized to H'0000 at power-on reset or manual reset, but is not initialized in standby mode.

Bit:	15	14	13	12	11	10	9	8
Bit name:	PINT15E	PINT14E	PINT13E	PINT12E	PINT11E	PINT10E	PINT9E	PINT8E
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	PINT7E	PINT6E	PINT5E	PINT4E	PINT3E	PINT2E	PINT1E	PINT0E
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15–0—PINT15 to PINT0 Interrupt Enable (PINT15E to PINT0E): Enable whether the interrupt requests input to the PINT15 to PINT0 pins.

Bits 15–0: PINT15E to PINT0E	Description	
0	Disables PINT input interrupt requests	(Initial value)
1	Enables PINT input interrupt requests	

When all or some of these pins, PINT0–PINT15 are not used as an interrupt input, a bit corresponding to a pin unused as an interrupt request should be set to 0.

9.3.6 Interrupt Request Register 0 (IRR0)

The IRR0 is an 8-bit register that indicates interrupt requests from external input pins IRQ0 to IRQ5 and PINT0 to PINT15. This register is initialized to H'00 at power-on reset or manual reset, but is not initialized in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	PINT0R	PINT1R	IRQ5R	IRQ4R	IRQ3R	IRQ2R	IRQ1R	IRQ0R
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

When clearing IRQ5R–IRQ0R bit to 0, 0 should be written to the bit after the bit is set to 1 and the contents of 1 are read. Only 0 can be written to IRQ5R–IRQ0R.

Bit 7—PINT0 to PINT7 Interrupt Request (PINT0R): Indicates whether interrupt requests are input to PINT0 to PINT7 pins.

Bit 7: PINT0R	Description	
0	Interrupt requests are not input to PINT0 to PINT7 pins	(Initial value)
1	Interrupt requests are input to PINT0 to PINT7 pins.	

Bit 6—PINT8 to PINT15 Interrupt Request (PINT1R): Indicates whether interrupt requests are input to PINT8 to PINT15 pins.

Bit 6: PINT1R	Description	
0	Interrupt requests are not input to PINT8 to PINT15 pins	(Initial value)
1	Interrupt requests are input to PINT8 to PINT15 pins.	

Bit 5—IRQ5 Interrupt Request (IRQ5R): Indicates whether an interrupt request is input to the IRQ5 pin. When edge detection mode is set for IRQ5, an interrupt request is cleared by clearing the IRQ5R bit.

Bit 5: IRQ5R	Description	
0	An interrupt request is not input to IRQ5 pin	(Initial value)
1	An interrupt request is input to IRQ5 pin	

Bit 4—IRQ4 Interrupt Request (IRQ4R): Indicates whether an interrupt request is input to the IRQ4 pin. When edge detection mode is set for IRQ4, an interrupt request is cleared by clearing the IRQ4R bit.

Bit 4: IRQ4R	Description	
0	An interrupt request is not input to IRQ4 pin	(Initial value)
1	An interrupt request is input to IRQ4 pin	

Bit 3—IRQ3 Interrupt Request (IRQ3R): Indicates whether an interrupt request is input to the IRQ3 pin. When edge detection mode is set for IRQ3, an interrupt request is cleared by clearing the IRQ3R bit.

Bit 3: IRQ3R	Description	
0	An interrupt request is not input to IRQ3 pin	(Initial value)
1	An interrupt request is input to IRQ3 pin	

Bit 2—IRQ2 Interrupt Request (IRQ2R): Indicates whether an interrupt request is input to the IRQ2 pin. When edge detection mode is set for IRQ2, an interrupt request is cleared by clearing the IRQ2R bit.

Bit 2: IRQ2R	Description	
0	An interrupt request is not input to IRQ2 pin	(Initial value)
1	An interrupt request is input to IRQ2 pin	

Bit 1—IRQ1 Interrupt Request (IRQ1R): Indicates whether an interrupt request is input to the IRQ1 pin. When edge detection mode is set for IRQ1, an interrupt request is cleared by clearing the IRQ1R bit.

Bit 1: IRQ1R	Description	
0	An interrupt request is not input to IRQ1 pin	(Initial value)
1	An interrupt request is input to IRQ1 pin	

Bit 0—IRQ0 Interrupt Request (IRQ0R): Indicates whether an interrupt request is input to the IRQ0 pin. When edge detection mode is set for IRQ0, an interrupt request is cleared by clearing the IRQ0R bit.

Bit 0: IRQ0R	Description	
0	An interrupt request is not input to IRQ0 pin	(Initial value)
1	An interrupt request is input to IRQ0 pin	

9.3.7 Interrupt Request Register 1 (IRR1)

The IRR1 is an 8-bit read-only register that indicates whether DMAC or IrDA interrupt requests are generated. This register is initialized to H'00 at power-on reset or manual reset, but is not initialized in software mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	TXI1R	BRI1R	RXI1R	ERI1R	DEI3R	DEI2R	DEI1R	DEI0R
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit 7—TXI1 Interrupt Request (TXI1R): Indicates whether a TXI1 (IrDA) interrupt request is generated.

Bit 7: TXI1	Description	
0	A TXI1 interrupt request is not generated	(Initial value)
1	A TXI1 interrupt request is generated	

Bit 6—BRI1 interrupt request (BRI1R): Indicates whether a BRI1 (IrDA) interrupt request is generated.

Bit 6: BRI1R	Description	
0	A BRI1 interrupt request is not generated	(Initial value)
1	A BRI1 interrupt request is generated	

Bit 5—RXI1 interrupt request (RXI1R): Indicates whether an RXI1 (IrDA) interrupt request is generated.

Bit 5: RXI1R	Description	
0	An RXI1 interrupt request is not generated	(Initial value)
1	An RXI1 interrupt request is generated	

Bit 4—ERI1 Interrupt Request (ERI1R): Indicates whether an ERI1 (IrDA) interrupt request is generated.

Bit 4: ERI1R	Description	
0	An ERI1 interrupt request is not generated	(Initial value)
1	An ERI1 interrupt request is generated	

Bit 3—DEI3 Interrupt Request (DEI3R): Indicates whether a DEI3 (DMAC) interrupt request is generated.

Bit 3: DEI3R	Description	
0	A DEI3 interrupt request is not generated	(Initial value)
1	A DEI3 interrupt request is generated	

Bit 2—DEI2 Interrupt Request (DEI2R): Indicates whether a DEI2 (DMAC) interrupt request is generated.

Bit 2: DEI2R	Description	
0	A DEI2 interrupt request is not generated	(Initial value)
1	A DEI2 interrupt request is generated	

Bit 1—DEI1 Interrupt Request (DEI1R): Indicates whether a DEI1 (DMAC) interrupt request is generated.

Bit 1: DEI1R	Description	
0	A DEI1 interrupt request is not generated	(Initial value)
1	A DEI1 interrupt request is generated	

Bit 0—DEI0 Interrupt Request (DEI0R): Indicates whether a DEI0 (DMAC) interrupt request is generated.

Bit 0: DEI0R	Description	
0	A DEI0 interrupt request is not generated	(Initial value)
1	A DEI0 interrupt request is generated	

9.3.8 Interrupt Request Register 2 (IRR2)

The IRR2 is an 8-bit read-only register that indicates whether A/D converter, or SCIF interrupt requests are generated. This register is initialized to H'00 at power-on reset, manual reset, or in hardware standby mode, but is not initialized in software standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	ADIR	TXI2R	BRI2R	RXI2R	ERI2R
Initial value:	0	0	0	0	0	0	0	0
R/W:	—	—	—	R	R	R	R	R

Bits 7 to 5—Reserved Bits: Writing is invalid. Always read as 0.

Bit 4—ADI Interrupt Request (ADIR): Indicates whether an ADI (ADC) interrupt request is generated.

Bit 4: ADIR	Description	
0	An ADI interrupt request is not generated	(Initial value)
1	An ADI interrupt request is generated	

Bit 3—TXI2 Interrupt Request (TXI2R): Indicates whether a TXI2 (SCIF) interrupt request is generated.

Bit 3: TXI2R	Description	
0	A TXI2 interrupt request is not generated	(Initial value)
1	A TXI2 interrupt request is generated	

Bit 2—BRI2 Interrupt Request (BRI2R): Indicates whether a BRI2 (SCIF) interrupt request is generated.

Bit 2: BRI2R	Description	
0	A BRI2 interrupt request is not generated	(Initial value)
1	A BRI2 interrupt request is generated	

Bit 1—RXI2 Interrupt Request (RXI2R): Indicates whether an RXI2 (SCIF) interrupt request is generated.

Bit 1: RXI2R	Description	
0	An RXI2 interrupt request is not generated	(Initial value)
1	An RXI2 interrupt request is generated	

Bit 0—ERI2 Interrupt Request (ERI2R): Indicates whether an ERI2 (SCIF) interrupt request is generated.

Bit 0: ERI2R	Description	
0	An ERI2 interrupt request is not generated	(Initial value)
1	An ERI2 interrupt request is generated	

9.4 INTC Operation

9.4.1 Interrupt Sequence

Thc scqucncc of intcrrupt opcrations is cxplaincd bclow. Figurc 9.3 is a flowchart of thc 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 E (IPRA to IPRE). 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 default priority or the highest priority within its IPR setting unit (as indicated in table 9.4) is selected.
3. The priority level of the interrupt selected by the interrupt controller is compared with the interrupt mask bits (I3–I0) in the status register (SR) of the CPU. If the request priority level is higher than the level in bits I3–I0, the interrupt controller accepts the interrupt and sends an interrupt request signal to the CPU.
4. When the interrupt controller receives an interrupt, a low level is output from the $\overline{\text{IRQOUT}}$ pin.
5. The CPU receives an interrupt at a break in instructions.
6. The interrupt source code is set in the interrupt event registers (INTEVT and INTEVT2).
7. The status register (SR) and program counter (PC) are saved to SSR and SPC, respectively.
8. The block bit (BL), mode bit (MD), and register bank bit (RB) in SR are set to 1.
9. 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). The interrupt handler may branch with the INTEVT register value as its offset in order to identify the interrupt source. This enables it to branch to the processing routine for the individual interrupt source.

Notes: 1. The interrupt mask bits (I3–I0) in the status register (SR) are not changed by acceptance of an interrupt in the SH7729.

2. $\overline{\text{IRQOUT}}$ outputs a low level until the interrupt request is cleared. However, if the interrupt source is masked by an interrupt mask bit, the $\overline{\text{IRQOUT}}$ pin returns to the high level. The level is output without regard to the BL bit.

3. 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, then wait for the interval shown in table 9.8 (Time for priority decision and SR mask bit comparison) before clearing the BL bit or executing an RTE instruction.

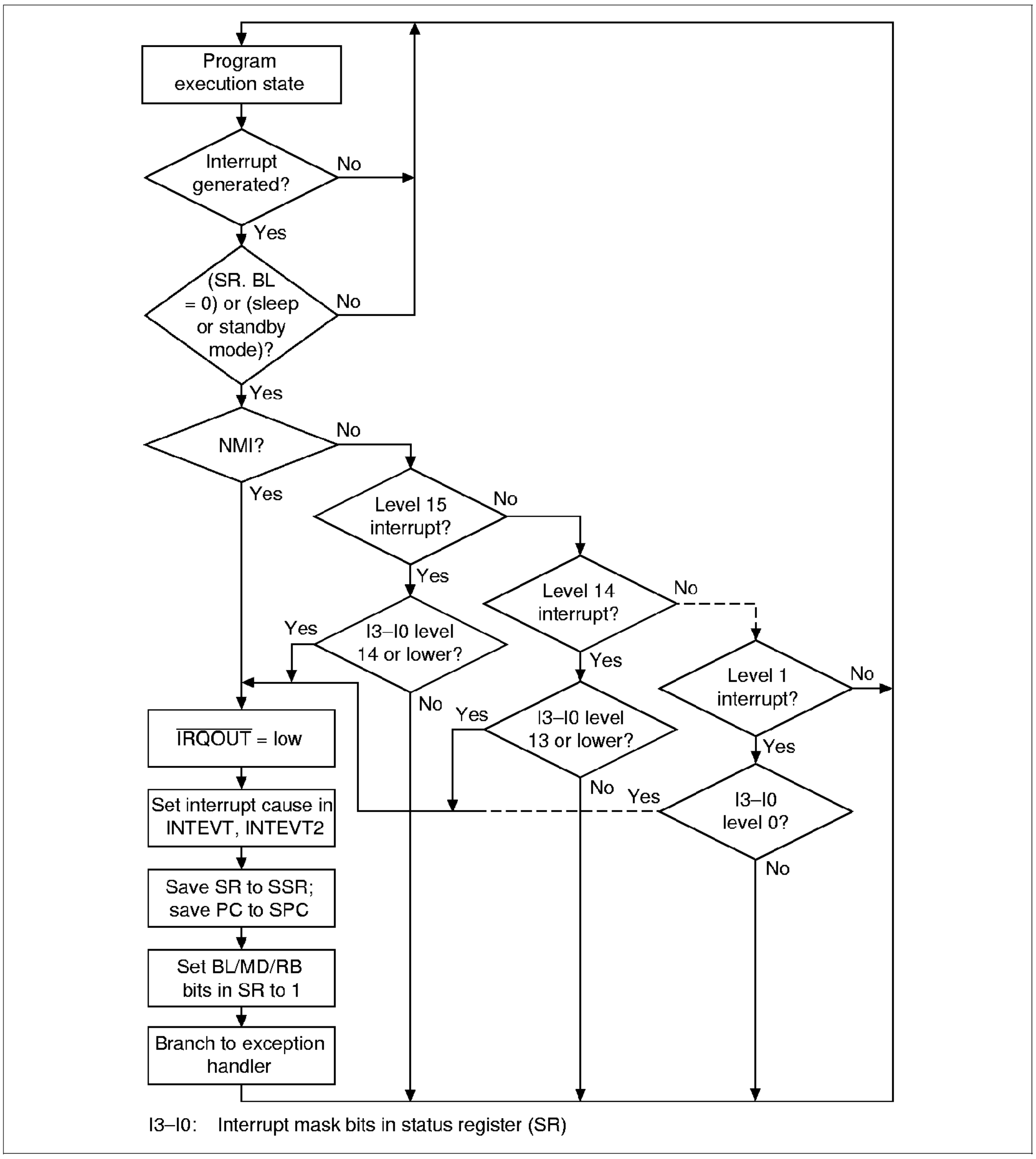


Figure 9.3 Interrupt Operation Flowchart

9.4.2 Multiple Interrupts

When processing multiple interrupts, an interrupt handler should include the following procedures:

1. Branch to a specific interrupt handler corresponding to a code set in INTEVT and INTEVT2. The code in INTEVT and INTEVT2 can be used as a branch-offset for branching to the specific handler.
2. Clear the cause of the interrupt in each specific handler.
3. Save SSR and SPC to the 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 these procedures are followed in order, an interrupt of higher priority than the one being handled can be accepted after clearing BL in step 4. Figure 9.3 shows a sample interrupt operation flowchart.

9.5 Interrupt Response Time

The time from generation of an interrupt request until interrupt exception processing is performed and fetching of the first instruction of the exception handler is started (the interrupt response time) is shown in table 9.8. Figure 9.4 shows an example of pipeline operation when an IRL interrupt is accepted. When SR.BL is 1, interrupt exception processing is masked, and is kept waiting until completion of an instruction that clears BL to 0.

Table 9.8 Interrupt Response Time

	Number of States				
Item	**NMI**	**IRQ**	**PINT**	**Peripheral Modules**	**Notes**
Time for priority decision and SR mask bit comparison	0.5 × Icyc + 0.5 × Bcyc + 0.5 × Pcyc	0.5 × Icyc + 1 × Bcyc + 4.5× Pcyc*4 0.5 × Icyc + 1 × Bcyc + + 2.5 × Pcyc*5	0.5 × Icyc + 3.5 × Pcyc	0.5 × Icyc + 1.5 × Pcyc*6 0.5 × Icyc + 3 × Pcyc*7	
Wait time until end of sequence being executed by CPU	X (≥ 0) × Icyc	X (≥ 0) × Icyc	X (≥ 0) × Icyc	X (≥ 0) × Icyc	Interrupt exception processing is kept waiting until the executing instruction ends. If the number of instruction execution states is S*1, the maximum wait time is: X = S – 1. However, if BL is set to 1 by instruction execution or by an exception, interrupt exception processing is deferred until completion of an instruction that clears BL to 0. If the following instruction masks interrupt exception processing, the processing may be further deferred.
Time from interrupt exception processing (save of SR and PC) until fetch of first instruction of exception handler is started	5 × Icyc	5 × Icyc	5 × Icyc	5 × Icyc	

Table 9.8 Interrupt Response Time (cont)

Item		Number of States				Notes
		NMI	IRQ	PINT	Peripheral Modules	
Response time	Total	(5.5 + X) × Icyc + 0.5 × Bcyc + 0.5 × Pcyc	(5.5 + X) × Icyc + 1 × Bcyc + 2.5 × Pcyc*6	(5.5 + X) × Icyc + 1.5 × Pcyc*6	(5.5 + X) × Icyc + 1.5 × Pcyc*6	
			(5.5 + X) × Icyc + 3 × Pcyc*5	(5.5 + X) × Icyc + 3 × Pcyc*7	(5.5 + X) × Icyc + 3 × Pcyc*7	
	Minimum case*2	7.5	16.5	12.5	8.5*6/11.5*7	At 60-MHz (CKIO = 30) operation: 0.13–0.28 µs
	Maximum case*3	7 + S	26.5 + S	18.5 + S	10.5 + S*6 16.5 + S*7	At 60-MHz (CKIO = 15) operation: 0.26–0.56 µs (in case of operand cache-hit) At 60-MHz (CKIO = 15) operation: 0.29–0.59 µs (when external memory access is performed with wait = 0)

Icyc: Duration of one cycle of internal clock supplied to CPU.

Bcyc: Duration of one CKIO cycle.

Pcyc: Duration of one cycle of peripheral clock supplied to peripheral modules.

Notes: *1. S also includes the memory access wait time.

The processing requiring the maximum execution time is LDC.L @Rm+, SR. When the memory access is a cache-hit, this requires seven instruction execution cycles. When the external access is performed, the corresponding number of cycles must be added. There are also instructions that perform two external memory accesses; if the external memory access is slow, the number of instruction execution cycles will increase accordingly.

*2. The internal clock: CKIO: peripheral clock ratio is 2: 1: 1.

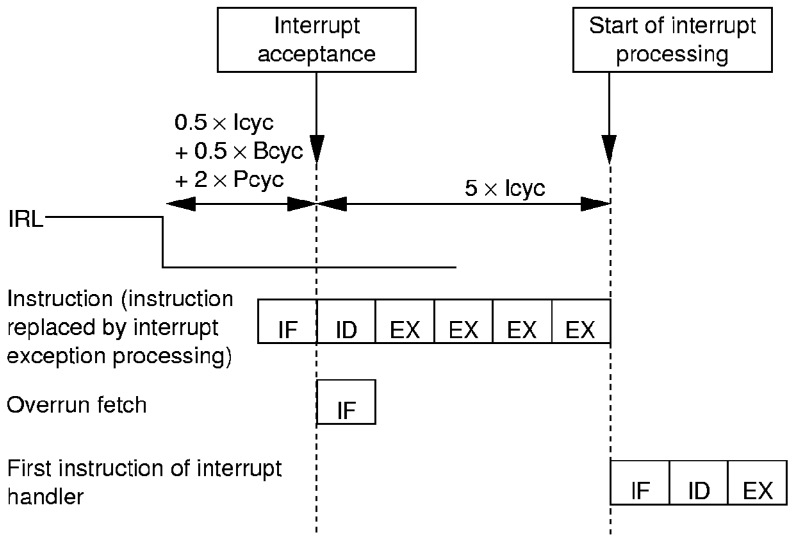
*3. The internal clock: CKIO: peripheral clock ratio is 4: 1: 1.

*4. IRQ mode

*5. IRL mode

*6. Modules: TMU, RTC, SCI, WDT, REFC

*7. Modules: DMAC, ADC, IrDA, SCIF


Interrupt acceptance
Start of interrupt processing
0.5 × Icyc
+ 0.5 × Bcyc
+ 2 × Pcyc
5 × Icyc
IRL
Instruction (instruction replaced by interrupt exception processing)
IF
ID
EX
EX
EX
EX
Overrun fetch
IF
First instruction of interrupt handler
IF
ID
EX

IF: Instruction fetch: Instruction is fetched from memory in which program is stored.
ID: Instruction decode: Fetched instruction is decoded.
EX: Instruction execution: Data operation and address calculation are performed in accordance with result of decoding.

Figure 9.4 Example of Pipeline Operations when IRL Interrupt is Accepted

Section 10 User Break Controller

10.1 Overview

The user break controller (UBC) provides functions that simplify program debugging. Break conditions are set in the UBC and a user break is generated according to the conditions of the bus cycle generated by the CPU or on-chip DMAC. The breakpoint check function monitors instruction fetches and operand read/writes, generating a variable combination of pre-execution instruction fetch, post-execution instruction fetch, and post-execution operand access breakpoint traps under designated read/write conditions.

This function makes it easy to design an effective self-monitoring debugger, enabling the chip to debug programs without using an in-circuit emulator.

10.1.1 Features

The user break controller has the following features:

- 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 logical AND, but not in the same bus cycle).
  - — Address (Compares 40 bits comprised of a 32-bit logical address prefixed with an ASID address

    Comparison bits are maskable in 32-bit units, user can easily program it to mask addresses at bottom 12 bits (4-k page), bottom 10 bits (1-k page), or any size of page, etc.

    The 8-bit ASID checking is from MMU control to indicate hit or not hit.)

    One of four address buses (CPU address bus (LAB), cache address bus (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 (CPU data bus (LDB), cache data bus (IDB), X-memory data bus (XDB) and Y-memory data bus (YDB)) can be selected.
  - — Bus master: CPU cycle or DMAC cycle
  - — Bus cycle: instruction fetch or data access
  - — Read/write
  - — Operand size: byte, word, or longword
- User break is generated upon satisfying break conditions. A user-designed user-break condition exception processing routine can be run.

- In an instruction fetch cycle, it can be selected that a break is set before or after an instruction is executed.
- Breaks can be specified for on-chip I/O accesses or LDTLB instruction execution in ASE mode.
- The number of repeat times can be specified as a break condition. (It is only for channel B)
- Maximum repeat times for the break condition: $2^{12} - 1$ times.
- Eight pairs of branch source/destination buffers.

10.1.2 Block Diagram

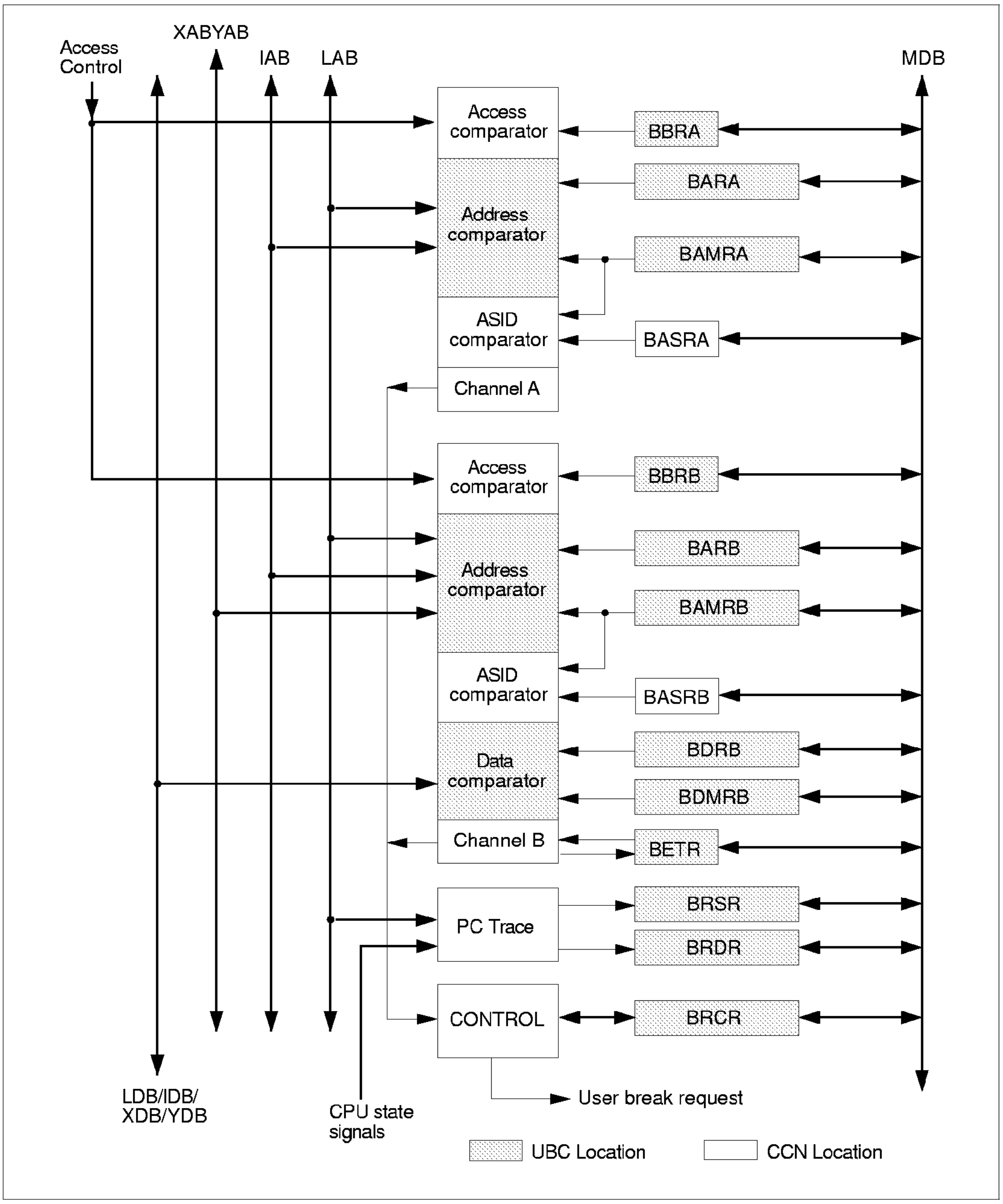


Figure 10.1 Block Diagram of User Break Controller

10.1.3 Register Configuration

Table 10.1Register Configuration

Name	Abbr.	R/W	Initial Value*1	Address	Access Size	Location	Section
Break address register A	BARA	R/W	H'00000000	H'FFFFFFB0	Word/long	UBC	10.2.1
Break address mask register A	BAMRA	R/W	H'00000000	H'FFFFFFB4	Word/long	UBC	10.2.2
Break bus cycle register A	BBRA	R/W	H'0000	H'FFFFFFB8	Word	UBC	10.2.3
Break address register B	BARB	R/W	H'00000000	H'FFFFFFA0	Word/long	UBC	10.2.4
Break address mask register B	BAMRB	R/W	H'00000000	H'FFFFFFA4	Word/long	UBC	10.2.5
Break bus cycle register B	BBRB	R/W	H'0000	H'FFFFFFA8	Word	UBC	10.2.6
Break data register B	BDRB	R/W	H'00000000	H'FFFFFF90	Word/long	UBC	10.2.7
Break data mask register B	BDMRB	R/W	H'00000000	H'FFFFFF94	Word/long	UBC	10.2.8
Break control register	BRCR	R/W	H'00000000	H'FFFFFF98	Word/long	UBC	10.2.9
Execution count break register	BETR	R/W	H'0000	H'FFFFFF9C	Word	UBC	10.2.10
Branch source register	BRSR	R	Undefined*2	H'FFFFFFAC	Word/long	UBC	10.2.11
Branch destination register	BRDR	R	Undefined*2	H'FFFFFFBC	Word/long	UBC	10.2.12
Break ASID register A	BASRA	R/W	Undefined	H'FFFFFFE4	Byte	CCN	10.2.13
Break ASID register B	BASRB	R/W	Undefined	H'FFFFFFE8	Byte	CCN	10.2.14

Notes: 1. Initialized by power-on reset. Values held in standby state and undefined by manual resets.

2. Bit 31 of BRSR and BRDR (valid flag) is initialized by power-on resets. But other bits are not initialized.

10.2 Register Descriptions

10.2.1 Break Address Register A (BARA)

Bit:	31	30	29	28	27	26	25	24
BARA:	BAA31	BAA30	BAA29	BAA28	BAA27	BAA26	BAA25	BAA24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
BARA:	BAA23	BAA22	BAA21	BAA20	BAA19	BAA18	BAA17	BAA16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
BARA:	BAA15	BAA14	BAA13	BAA12	BAA11	BAA10	BAA9	BAA8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
BARA:	BAA7	BAA6	BAA5	BAA4	BAA3	BAA2	BAA1	BAA0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BARA is a 32-bit read/write register. BARA specifies the address used as a break condition in channel A. A power-on reset initializes BARA to H'00000000.

BARA—Break Address A31–A0 (BAA31–BAA0): Stores the address on the LAB or IAB specifying break conditions of channel A.

10.2.2 Break Address Mask Register A (BAMRA)

Bit:	31	30	29	28	27	26	25	24
BAMRA:	BAMA31	BAMA30	BAMA29	BAMA28	BAMA27	BAMA26	BAMA25	BAMA24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
BAMRA:	BAMA23	BAMA22	BAMA21	BAMA20	BAMA19	BAMA18	BAMA17	BAMA16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
BAMRA:	BAMA15	BAMA14	BAMA13	BAMA12	BAMA11	BAMA10	BAMA9	BAMA8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
BAMRA:	BAMA7	BAMA6	BAMA5	BAMA4	BAMA3	BAMA2	BAMA1	BAMA0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BAMRA is a 32-bit read/write register. BAMRA specifies bits masked in the break address specified by BARA. A power-on reset initializes BAMRA to H'00000000.

BAMRA—Break Address Mask Register A31–A0 (BAMA31–BAMA0): Specifies bits masked in the channel A break address bits specified by BARA (BAA31–BAA0).

Bits 31 to 0: BAMAn	**Description**
0	Break address bit BAAn of channel A is included in the break condition (Initial value)
1	Break address bit BAAn of channel A is masked and is not included in the break condition

n = 31–0

10.2.3 Break Bus Cycle Register A (BBRA)

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

Bit:	7	6	5	4	3	2	1	0
BBRA:	CDA1	CDA0	IDA1	IDA0	RWA1	RWA0	SZA1	SZA0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Break bus cycle register A (BBRA) is a 16-bit read/write register, which specifies (1) CPU cycle or DMAC cycle, (2) instruction fetch or data access, (3) read or write, and (4) operand size in the break conditions of channel A. A power-on reset initializes BBRA to H'0000.

Bits 15 to 8—Reserved Bits: Writing is disabled. These bits are always read as 0.

Bits 7 and 6—CPU Cycle/DMAC Cycle Select A (CDA1, CDA0): Selects the CPU cycle or DMAC cycle as the bus cycle of the channel A break condition.

Bit 7: CDA1	Bit 6: CDA0	Description	
0	0	Condition comparison is not performed	(Initial value)
*	1	The break condition is the CPU cycle	
1	0	The break condition is the DMAC cycle	

Bits 5 and 4—Instruction Fetch/Data Access Select A (IDA1, IDA0): Selects the instruction fetch cycle or data access cycle as the bus cycle of the channel A break condition.

Bit 5: IDA1	Bit 4: IDA0	Description	
0	0	Condition comparison is not performed	(Initial value)
	1	The break condition is the instruction fetch cycle	
1	0	The break condition is the data access cycle	
	1	The break condition is the instruction fetch cycle or data access cycle	

Bits 3 and 2—Read/Write Select A (RWA1, RWA0): Selects the read cycle or write cycle as the bus cycle of the channel A break condition.

Bit 3: RWA1	Bit 2: RWA0	Description
0	0	Condition comparison is not performed (Initial value)
	1	The break condition is the read cycle
1	0	The break condition is the write cycle
	1	The break condition is the read cycle or write cycle

Bits 1 and 0—Operand Size Select A (SZA1, SZA0): Selects the operand size of the bus cycle for the channel A break condition.

Bit 1: SZA1	Bit 0: SZA0	Description
0	0	The break condition does not include operand size (Initial value)
	1	The break condition is byte access
1	0	The break condition is word access
	1	The break condition is longword access

10.2.4 Break Address Register B (BARB)

Bit:	31	30	29	28	27	26	25	24
BARB:	BAB31	BAB30	BAB29	BAB28	BAB27	BAB26	BAB25	BAB24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
BARB:	BAB23	BAB22	BAB21	BAB20	BAB19	BAB18	BAB17	BAB16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
BARB:	BAB15	BAB14	BAB13	BAB12	BAB11	BAB10	BAB9	BAB8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
BARB:	BAB7	BAB6	BAB5	BAB4	BAB3	BAB2	BAB1	BAB0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BARB is a 32-bit read/write register. BARB specifies the address used as a break condition in channel B. Control bits XYE and XYS in the BBRB selects an address bus for break condition B. If the XYE is 0, then BARB specifies the break address on logic or internal bus, LAB or IAB. If the XYE is 1, then the BAB 31-16 specifies the break address on XAB (bits 15-1) and the BAB 15-0 specifies the break address on YAB (bits 15-1). However, you have to choose one of two address buses for the break. A power-on reset initializes BARB to H'00000000.

	BAB31–16	BAB15–0
XYE = 0	L(I) AB31–16	L(I) AB15–0
XYE = 1	XAB15–1 (XYS = 0)	YAB15–1 (XYS = 1)

10.2.5 Break Address Mask Register B (BAMRB)

Bit:	31	30	29	28	27	26	25	24
BAMRB:	BAMB31	BAMB30	BAMB29	BAMB28	BAMB27	BAMB26	BAMB25	BAMB24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
BAMRB:	BAMB23	BAMB22	BAMB21	BAMB20	BAMB19	BAMB18	BAMB17	BAMB16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
BAMRB:	BAMB15	BAMB14	BAMB13	BAMB12	BAMB11	BAMB10	BAMB9	BAMB8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
BAMRB:	BAMB7	BAMB6	BAMB5	BAMB4	BAMB3	BAMB2	BAMB1	BAMB0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BAMRB is a 32-bit read/write register. BAMRB specifies bits masked in the break address specified by BARB. A power-on reset initializes BAMRB to H'00000000.

	BAMB31–16	BAMB15–0
XYE = 0	Mask L(I) AB31–16	Mask L(I) AB15–0
XYE = 1	Mask XAB15–1 (XYS = 0)	Mask YAB15–1 (XYS = 1)

Bits 31–0: BAMBn	Description
0	Break address BABn of channel B is included in the break condition (Initial value)
1	Break address BABn of channel B is masked and is not included in the break condition

n = 31–0

10.2.6 Break Data Register B (BDRB)

Bit:	31	30	29	28	27	26	25	24
BDRBH:	BDB31	BDB30	BDB29	BDB28	BDB27	BDB26	BDB25	BDB24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
BDRBH:	BDB23	BDB22	BDB21	BDB20	BDB19	BDB18	BDB17	BDB16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
BDRBH:	BDB15	BDB14	BDB13	BDB12	BDB11	BDB10	BDB9	BDB8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
BDRBH:	BDB7	BDB6	BDB5	BDB4	BDB3	BDB2	BDB1	BDB0
Initial value:	0	0	0	0	0	0	0	0

BDRB is a 32-bit read/write register. The control bits XYE and XYS in BBRB select a data bus for break condition B. If the XYE is 0, then BDRB specifies the break data on LDB or IDB. If the XYE is 1, then BDB 31-16 specifies the break data on XDB (bits 15-0) and BDB 15-0 specifies the break data on YDB (bits 15-0). However, you have to choose one of two data buses for the break. A power-on reset initializes BDRB to H'00000000.

	BDB31–16	BDB15–0
XYE = 0	L(I) DB31–16	L(I) DB15–0
XYE = 1	XDB15–0 (XYS = 0)	YDB15–0 (XYS = 1)

10.2.7 Break Data Mask Register B (BDMRB)

Bit:	31	30	29	28	27	26	25	24
BDMRBH	BDMB31	BDMB30	BDMB29	BDMB28	BDMB27	BDMB26	BDMB25	BDMB24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
BDMRBH:	BDMB23	BDMB22	BDMB21	BDMB20	BDMB19	BDMB18	BDMB17	BDMB16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
BDMRBL:	BDMB15	BDMB14	BDMB13	BDMB12	BDMB11	BDMB10	BDMB9	BDMB8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
BDMRBL:	BDMB7	BDMB6	BDMB5	BDMB4	BDMB3	BDMB2	BDMB1	BDMB0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BDMRB is a 32-bit read/write register. BDMRB specifies bits masked in the break data specified by BDRB. A power-on reset initializes BDMRB to H'00000000.

	BDMB31–16	**BDMB15–0**
XYE = 0	Mask L(I) DB31–16	Mask L(I) DB15–0
XYE = 1	Mask XDB15–0 (XYS = 0)	Mask YDB15–0 (XYS = 1)

Bits 31–0: BDMBn	**Description**
0	Break data BDBn of channel B is included in the break condition (Initial value)
1	Break data BDBn of channel B is masked and is not included in the break condition

n = 31–0

Notes: 1. Specify an operand size when including the value of the data bus in the break condition.

2. When a byte size is selected as a break condition, the break data must be set in bits 15-8 in BDRB for an even break address and bits 7-0 for an odd break address. Another 8 bits have no influence on a break condition.

10.2.8 Break Bus Cycle Register B (BBRB)

Bit:	15	14	13	12	11	10	9	8
BBRB:	—	—	—	—	—	—	XYE	XYS
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
BBRB:	CDB1	CDB0	IDB1	IDB0	RWB1	RWB0	SZB1	SZB0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Break bus cycle register B (BBRB) is a 16-bit read/write register, which specifies (1) logic or internal bus (L or I bus), X bus, of Y bus, (2) CPU cycle or DMAC cycle, (3) instruction fetch or data access, (4) read/write, and (5) operand size in the break conditions of channel B. A power-on reset initializes BBRB to H'0000.

Bits 15 to 10—Reserved Bits: Writing is disabled. These bits are always read as 0.

Bit 9—X/Y Memory Bus Enable (XYE): Selects the logic or internal bus (L or I bus) or X/Y memory bus as the bus of the channel B break condition.

Bit 9: XYE	Description
0	Select internal bus (I bus) for the channel B break condition
1	Select X/Y memory bus (X/Y bus) for the channel B break condition

Bits 8—X or Y Memory Bus Select (XYS): Selects the X bus or the Y bus as the bus of the channel B break condition.

Bit 8: XYS	Description
0	Select the X bus for the channel B break condition
1	Select the Y bus for the channel B break condition

Bits 7 and 6—CPU Cycle/DMAC Cycle Select B (CDB1 and CDB0): Select the CPU cycle or DMAC cycle as the bus cycle of the channel B break condition.

Bit 7: CDB1	Bit 6: CDB0	Description
0	0	Condition comparison is not performed (Initial value)
*	1	The break condition is the CPU cycle
1	0	The break condition is the DMAC cycle

Bits 5 and 4—Instruction Fetch/Data Access Select B (IDB1 and IDB0): Select the instruction fetch cycle or data access cycle as the bus cycle of the channel B break condition.

Bit 5: IDB1	Bit 4: IDB0	Description
0	0	Condition comparison is not performed (Initial value)
	1	The break condition is the instruction fetch cycle
1	0	The break condition is the data access cycle
	1	The break condition is the instruction fetch cycle or data access cycle

Bits 3 and 2—Read/Write Select B (RWB1 and RWB0): Select the read cycle or write cycle as the bus cycle of the channel B break condition.

Bit 3: RWB1	Bit 2: RWB0	Description
0	0	Condition comparison is not performed (Initial value)
	1	The break condition is the read cycle
1	0	The break condition is the write cycle
	1	The break condition is the read cycle or write cycle

Bits 1 and 0—Operand Size Select B (SZB1 and SZB0): Select the operand size of the bus cycle for the channel B break condition.

Bit 1: SZB1	Bit 0: SZB0	Description
0	0	The break condition does not include operand size (Initial value)
	1	The break condition is byte access
1	0	The break condition is word access
	1	The break condition is longword access

10.2.9 Break Control Register (BRCR)

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

Bit:	23	22	21	20	19	18	17	16
BRCR:	—	—	BASMA	BASMB	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R	R	R	R

Bit:	15	14	13	12	11	10	9	8
BRCR:	SCMFCA	SCMFCB	SCMFDA	SCMFDB	PCTE	PCBA	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R	R

Bit:	7	6	5	4	3	2	1	0
BRCR:	DBEB	PCBB	—	—	SEQ	—	—	ETBE
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R/W	R	R	R/W

BRCR sets the following conditions:

1. Channels A and B are used in two independent channels condition or under the sequential condition.
2. A break is set before or after instruction execution.
3. A break is set by the number of execution times.
4. Determine whether to include data bus on channel B in comparison conditions.
5. Enable PC trace.
6. Enable on-chip I/O break.
7. Enable LDTLB execution break.
8. Enable the ASID check.
9. Enable the trigger output.
10. Indicate whether to mask user interrupts after RTB.
10. Allow user to use UBC in the ASE user mode.

The break control register (BRCR) is a 32-bit read/write register that has break conditions match flags and bits for setting a variety of break conditions. The reading of SCMFIOB, SCMFLDT, TOEA, BIEA, TOEB, BIEB, IOBE, LDTE, NPROT and RTBMK bits are meaningless in main chip mode. They are always read as 0.

BRCR has many condition match flags, SCMFCA, SCMFCB, SCMFDA, SCMFDB, SCMFLDT, and SCMFIOB. A power-on reset initializes BRCR to H'00000000.

Bits 31 to 22—Reserved Bits: Writing is disabled. These bits are always read as 0.

Bit 21—Break ASID Mask A (BASMA): Specifies whether the bits of the channel A break ASID7-ASID0 (BASA7 to BASA0) set in BASRA are masked or not.

Bit 21: BASMA	Description	
0	All BASRA bits are included in break condition, ASID is checked	(Initial value)
1	No BASRA bits are included in break condition, ASID is not checked	

Bit 20—Break ASID Mask B (BASMB): Specifies whether the bits of channel B break ASID7-ASID0 (BASB7 to BASB0) set in BASRB are masked or not.

Bit 20: BASMB	Description	
0	All BASRB bits are included in break condition, ASID is checked	(Initial value)
1	No BASRB bits are included in break condition, ASID is not checked	

Bit 19—LDTLB Condition Match Flag (SCMFLDT): Indicates that LDTLB execution break condition has been met. Not cleared to 0. (To check a flag setting after it has been set, clear it by writing 0.)

Bit 19: SCMFLDT	Description	
0	LDTLB execution break condition does not match	(Initial value)
1	LDTLB execution break condition matches	

Bit 18—On-chip I/O Condition Match Flag (SCMFIOB): Indicates that on-chip I/O break conditions have been met. Not cleared to 0. (To check a flag setting after it has been set, clear it by writing 0.)

Bit 18: SCMFIOB	Description	
0	On-chip I/O break conditions do not match	(Initial value)
1	On-chip I/O break conditions match	

Bit 17—LDTLB Condition Break Enable (LDTE): Specifies whether to include LDTLB execution in break conditions. This bit is meaningless in the main chip mode.

Bit 17: LDTE	Description	
0	LDTLB execution is not included in break conditions	(Initial value)
1	LDTLB execution is included in break conditions	

Bit 16—On-Chip I/O Break Enable (IOBE): Specifies whether to include on-chip I/O access in break conditions. This bit is meaningless in the main chip mode.

Bit 16: IOBE	Description	
0	On-chip I/O access is not included in break conditions	(Initial value)
1	On-chip I/O access is included in break conditions	

Bit 15—CPU Condition Match Flag A (SCMFCA): When the CPU 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.

Bit 15: SCMFCA	Description	
0	The CPU cycle condition for channel A does not match	(Initial value)
1	The CPU cycle condition for channel A matches	

Bit 14—CPU Condition Match Flag B (SCMFCB): When the CPU 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.

Bit 14: SCMFCB	Description	
0	The CPU cycle condition for channel B does not match	(Initial value)
1	The CPU cycle condition for channel B matches	

Bit 13—DMAC Condition Match Flag A (SCMFDA): When the on-chip DMAC 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.

Bit 13: SCMFDA	Description	
0	The DMAC cycle condition for channel A does not match	(Initial value)
1	The DMAC cycle condition for channel A matches	

Bit 12—DMAC Condition Match Flag B (SCMFDB): When the on-chip DMAC 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.

Bit 12: SCMFDB	Description	
0	The DMAC cycle condition for channel B does not match	(Initial value)
1	The DMAC cycle condition for channel B matches	

Bit 11—PC Trace Enable (PCTE): Enables PC trace.

Bit 11: PCTE	Description	
0	Disables PC trace	(Initial value)
1	Enables PC trace	

Bit 10—PC Break Select A (PCBA): Selects the break timing of the instruction fetch cycle for channel A as before or after instruction execution.

Bit 10: PCBA	Description	
0	PC break of channel A is set before instruction execution	(Initial value)
1	PC break of channel A is set after instruction execution	

Bit 9—Trigger Output Enable A (TOEA): Selects whether to enable output of a trigger to the external pin when channel A set conditions are met. This bit is meaningless in the SH7729.

Bit 9: TOEA	Description	
0	Trigger is not output to external pin on channel A condition match	(Initial value)
1	Trigger is output to external pin on channel A condition match	

Bit 8—ASE Break Enable A (BIEA): Specifies whether to request an ASE break to the CPU when channel A set conditions are met. This bit is meaningless in the main chip mode.

Bit 8: BIEA	Description
0	An ASE break is not requested to the CPU on channel A condition match (Initial value)
1	An ASE break is requested on channel A condition match

Bit 7—Data Break Enable B (DBEB): Selects whether or not the data bus condition is included in the break condition of channel B.

Bit 7: DBEB	Description
0	No data bus condition is included in the condition of channel B (Initial value)
1	The data bus condition is included in the condition of channel B

Bit 6—PC Break Select B (PCBB): Selects the break timing of the instruction fetch cycle for channel B as before or after instruction execution.

Bit 6: PCBB	Description
0	PC break of channel B is set before instruction execution (Initial value)
1	PC break of channel B is set after instruction execution

Bit 5—Trigger Output Enable B (TOEB): Selects whether to enable output of a trigger to the external pin when channel B set conditions are met. This bit is meaningless in the SH7729.

Bit 5: TOEB	Description
0	Trigger is not output to external pin on channel B condition match (Initial value)
1	Trigger is output to external pin on channel B condition match

Bit 4—ASE Break Enable B (BIEB): Specifies whether to request an ASE break to the CPU when channel B set conditions are met. This bit is meaningless in the main chip mode.

Bit 4: BIEB	Description
0	An ASE break is not requested to the CPU on channel B condition match (Initial value)
1	An ASE break is requested on channel B condition match

Bit 3—Sequence Condition Select (SEQ): Selects two conditions of channels A and B as independent or sequential.

Bit 3: SEQ	Description	
0	Channels A and B are compared under the independent condition	(Initial value)
1	Channels A and B are compared under the sequential condition	

Bit 2—NPROT Bit: Indicates whether to release UBC to user. This bit is meaningless in the main chip mode. The reading of BRCR.NPROT bit could be impossible in the ASE break mode if the value is 1 because the mode has switched to the ASE user mode.

Bit 2: NPROT	Description	
0	UBC is not released to user	(Initial value)
1	UBC is released to user	

Bit 1—RTB Interrupt Mask (RTBMK): Indicates whether to mask user interrupts after RTB. This bit is meaningless in the main chip mode.

Bit 1: RTBMK	Description	
0	Interrupts are accepted after RTB	(Initial value)
1	Interrupts are not accepted after RTB	

Bit 0—The Number of Execution Times Break Enable (ETBE): Enable 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 the BETR register.

Bit 0: ETBE	Description	
0	The execution-times break condition is masked on channel B	(Initial value)
1	The execution-times break condition is enabled on channel B	

10.2.10 Execution Times Break Register (BETR)

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

Bit:	7	6	5	4	3	2	1	0
BETR:								
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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. A power-on reset initializes BETR to H'0000. When a break condition is satisfied, it decreases the BETR. A break is issued when the break condition is satisfied after the BETR becomes H'0001. Bits 15-12 are always read as 0 and 0 should always be written in these bits.

10.2.11Branch Source Register (BRSR)

Bit:	31	30	29	28	27	26	25	24
BRSR:	SVF	PID2	PID1	PID0	BSA27	BSA26	BSA25	BSA24
Initial value:	0	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	23	22	21	20	19	18	17	16
BRSR:	BSA23	BSA22	BSA21	BSA20	BSA19	BSA18	BSA17	BSA16
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	15	14	13	12	11	10	9	8
BRSR:	BSA15	BSA14	BSA13	BSA12	BSA11	BSA10	BSA9	BSA8
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
BRSR:	BSA7	BSA6	BSA5	BSA4	BSA3	BSA2	BSA1	BSA0
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

BRSR is a 32-bit read register. BRSR stores the last fetched address before branch and the pointer (3 bits) which indicates the number of cycles from fetch to execution for the last executed instruction. BRSR has the flag bit that is set to 1 when branch occurs. This flag bit is cleared to 0, when BRSR is read and also initialized by power-on resets or manual resets. Other bits are not initialized by reset. Four BRSR registers have queue structure and a stored register is shifted every branch.

Bit 31—BRSR Valid Flag (SVF): Indicates whether the address and the pointer by which the branch source address can be calculated. When a branch source address is fetched, this flag is set to 1. This flag is cleared to 0 in reading BRSR.

Bit 31: SVF	Description
0	The value of BRSR register is invalid
1	The value of BRSR register is valid

Bits 30 to 28—Instruction Decode Pointer (PID2–0): PID is a 3-bit binary pointer (0–7). These bits indicate the instruction buffer number which stores the last executed instruction before branch.

Bits 30 to 28: PID	Description
Even	PID indicates the instruction buffer number.
Odd	PiD+2 indicates the instruction buffer number

Bits 27 to 0—Branch Source Address (BSA27–BSA0): These bits store the last fetched address before branch.

10.2.12 Branch Destination Register (BRDR)

Bit:	31	30	29	28	27	26	25	24
BRDR:	DVF	—	—	—	BDA27	BDA26	BDA25	BDA24
Initial value:	0	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	23	22	21	20	19	18	17	16
BRDR:	BDA23	BDA22	BDA21	BDA20	BDA19	BDA18	BDA17	BDA16
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	15	14	13	12	11	10	9	8
BRDR:	BDA15	BDA14	BDA13	BDA12	BDA11	BDA10	BDA9	BDA8
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
BRDR:	BDA7	BDA6	BDA5	BDA4	BDA3	BDA2	BDA1	BDA0
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

BRDR is a 32-bit read register. BRDR stores the branch destination fetch address. BRDR has the flag bit that is set to 1 when branch occurs. This flag bit is cleared to 0, when BRDR is read and also initialized by power-on resets or manual resets. Other bits are not initialized by resets. Four BRDR registers have queue structure and a stored register is shifted every branch.

Bit 31—BRDR Valid Flag (DVF): Indicates whether a branch destination address is stored. When a branch destination address is fetched, this flag is set to 1. This flag is set to 0 in reading BRDR.

Bit 31: DVF	Description
0	The value of BRDR register is invalid
1	The value of BRDR register is valid

Bits 30 to 28—Reserved Bits: These bits are always read as 0. Writing is disabled.

Bits 27 to 0—Branch Destination Address (BDA27–BDA0): These bits store the first fetched address after branch.

10.2.13 Break ASID Register A (BASRA)

Bit:	7	6	5	4	3	2	1	0
BASRA:	BASA7	BASA6	BASA5	BASA4	BASA3	BASA2	BASA1	BASA0
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Break ASID register A (BASRA) is an 8-bit read/write register that specifies the ASID that serves as the break condition for channel A. It is not initialized by resets. It is located in CCN.

Bits 7 to 0—Break ASID A7 to 0 (BASA7 to BASA0): These bits store the ASID (bits 7 to 0) that is the channel A break condition.

10.2.14 Break ASID Register B (BASRB)

Bit:	7	6	5	4	3	2	1	0
BASRB:	BASB7	BASB6	BASB5	BASB4	BASB3	BASB2	BASB1	BASB0
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Break ASID register B (BASRB) is an 8-bit read/write register that specifies the ASID that serves as the break condition for channel B. It is not initialized by resets. It is located in CCN.

Bits 7 to 0: Break ASID A7 to 0 (BASB7 to BASB0): These bits store the ASID (bits 7 to 0) that is the channel B break condition.

10.2.15 I/O Break Address Register (IOBAR)

Bit:	31	30	29	28	27	26	25	24
IOBAR:	IOBA31	IOBA30	IOBA29	IOBA28	IOBA27	IOBA26	IOBA25	IOBA24
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
IOBAR:	IOBA23	IOBA22	IOBA21	IOBA20	IOBA19	IOBA18	IOBA17	IOBA16
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
IOBAR:	IOBA15	IOBA14	IOBA13	IOBA12	IOBA11	IOBA10	IOBA9	IOBA8
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
IOBAR:	IOBA7	IOBA6	IOBA5	IOBA4	IOBA3	IOBA2	IOBA1	IOBA0
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The I/O break address register (IOBAR) is a 32-bit read/write register in the ASE mode. It writes the on-chip I/O address that was accessed in the on-chip I/O break. It is not initialized by resets.

Bits 31 to 0—I/O break address 31 to 0 (IOBA31 to IOBA0): These bits store the on-chip I/O address when an on-chip I/O break occurs.

10.2.16 I/O Break Data Register (IOBDR)

Bit:	31	30	29	28	27	26	25	24
IOBDR:	IOBD31	IOBD30	IOBD29	IOBD28	IOBD27	IOBD26	IOBD25	IOBD24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
IOBDR:	IOBD23	IOBD22	IOBD21	IOBD20	IOBD19	IOBD18	IOBD17	IOBD16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
IOBDR:	IOBD15	IOBD14	IOBD13	IOBD12	IOBD11	IOBD10	IOBD9	IOBD8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
IOBDR:	IOBD7	IOBD6	IOBD5	IOBD4	IOBD3	IOBD2	IOBD1	IOBD0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The I/O break data register (IOBDR) is a 32-bit read/write register in the ASE mode. It writes the on-chip I/O data that was accessed in the on-chip I/O break. It is not initialized by resets.

Bits 31 to 0—I/O break data 31 to 0 (IOBD31 to IOBD0): These bits store the on-chip I/O data when an on-chip I/O break occurs.

10.3 Operation Description

10.3.1 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 and the corresponding ASIDs are loaded in the break address registers (BARA and BARB) and break ASID registers (BASRA and BASRB in CCN). The masked addresses are set in the break address mask registers (BAMRA and 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 breaking bus conditions are set in the break bus cycle registers (BBRA and BBRB). Three groups of the BBRA and BBRB (CPU cycle/DMAC 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 BRCR.
2. When the break conditions are satisfied, the UBC sends a user break request to the interrupt controller. The break type will be sent to CPU indicating the instruction fetch, pre/post instruction break, data access break, or on-chip I/O access/LDTLB break. When conditions match up, the CPU condition match flags (SCMFCA and SCMFCB) and DMAC condition match flags (SCMFDA and SCMFDB) for the respective channels are set.
3. When the ASE mode is set (not in the main chip mode) and the set conditions are satisfied, the UBC sends an ASE break request to interrupt controller. When conditions match up, the corresponding match flags (SCMFIOB and SCMFLDT) or other flags are set.
4. 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.
5. There is a chance that the data access break and its following instruction fetch break occur around the same time, there will be only one break request to the CPU, but these two break channel match flags could be both set.

10.3.2 Break on Instruction Fetch Cycle

1. When CPU/instruction fetch/read/word or longword is set in the break bus cycle registers (BBRA/BBRB), the break condition becomes the CPU instruction fetch cycle. Whether it then breaks before or after the execution of the instruction can then be selected with the PCBA/PCBB bits of the break control register (BRCR) for the appropriate channel.
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 delay branch instruction, the break is generated prior to execution of the instruction that then first accepts the break. Meanwhile, the break set for pre-instruction-break on delay slot instruction and post-instruction-break on SLEEP instruction are also prohibited.
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 delay branch instruction, the break is generated at the instruction that then first accepts the break.
4. When an instruction fetch cycle is set for channel B, break data register B (BDRB) is ignored. There is thus no need to set break data for the break of the instruction fetch cycle.

10.3.3 Break by Data Access Cycle

1. The memory cycles in which CPU data access breaks occur are from instructions.
2. The relationship between the data access cycle address and the comparison condition for operand size are listed in table 10.2:

Table 10.2Data Access Cycle Addresses and Operand Size Comparison Conditions

Access Size	Address Compared
Longword	Compares break address register bits 31–2 to address bus bits 31–2
Word	Compares break address register bits 31–1 to address bus bits 31–1
Byte	Compares break address register bits 31–0 to address bus bits 31–0

This means that when address H'00001003 is set without specifying the size condition, 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 B channel:

   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 registers (BBRA and 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 15–8 and bits 7–0 of the break data register B (BDRB) and break data mask register B (BDMRB). When word or byte is set, bits 31–16 of BDRB and BDMRB are ignored.
4. When the DMAC data access is included in the break condition:

   When the address is included in the break condition on DMAC data access, the operand size of the break bus cycle registers (BBRA and BBRB) should be byte, word or no specified operand size. When the data value is included, select either byte or word.

10.3.4 Break on X/Y-Memory Bus Cycle

1. The break condition on X/Y-memory bus cycle is specified only in channel B. If XYE in BBRB is set to 1, break address and break data on X/Y-memory bus are selected. At this time, select X-memory bus or Y-memory bus by specifying XYS in BBRB. The Break condition cannot include both X-memory and Y-memory at the same time. The break condition is applied to X/Y-memory bus cycle by specifying CPU/data access/read or write/word or no specified operand size in the break bus cycle register B (BBRB).
2. When X-memory address is selected as the break condition, specify X-memory address in upper 16 bits in BARB and BAMRB. When Y-memory address is selected, specify Y-memory address in lower 16 bits. Specification of X/Y-memory data is the same for BDRB and BDMRB.

10.3.5 Sequential Break

1. By specifying SEQ in BRCR is set to 1, the sequential break is issued when channel B break condition matches after channel A break condition matches. A user break is ignored even if channel B break condition matches before channel A break condition matches. When channels A and B condition match at the same time, the sequential break is not issued.
2. In sequential break specification, internal/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 at channel B condition match with BETR = H'0001 after channel A condition match.

10.3.6 Value of Saved Program Counter

The PC when a break occurs is saved to the SPC in user breaks but saved to a fixed address (H'FD000000) in the ASE space in ASE break. The PC value saved is as follows depending on the type of break.

1. When instruction fetch (before instruction execution) is specified as a break condition:

The value of the program counter (PC) saved is the address of the instruction that matches the break condition. The fetched instruction is not executed, and a break occurs before it.

2. When instruction fetch (after instruction execution) is specified as a break condition:

   The PC value saved is the address of the instruction to be executed following the instruction in which the break condition matches. The fetched instruction is executed, and a break occurs before the execution of the next instruction.

3. When data access (address only) is specified as a break condition:

   The PC value is the address of the instruction to be executed following the instruction that matched the break condition. The instruction that matched the condition is executed and the break occurs before the next instruction is executed.

4. When data access (address + data) is specified as a break condition:

   The PC value is the start address of the instruction that follows the instruction already executed when break processing started up. When a data value is added to the break conditions, the place where the break will occur cannot be specified exactly. The break will occur before the execution of an instruction fetched around the data access where the break occurred.

5. Software ASE breaks (BRK instructions):

   The PC value saved is the address of the instruction following the BRK instruction.

6. On-chip I/O breaks and LDTLB ASE breaks:

   Same as 3 above.

10.3.7 PC Trace

1. Setting PCTE in BRCR to 1 enables PC traces. When branch (branch instruction, repeat, and interrupt) is generated, the address from which the branch source address can be calculated and the branch destination address are stored in BRSR and BRDR, respectively. The branch address and the pointer, which corresponds to the branch, are included in BRSR.

2. The branch address before branch occurs can be calculated from the address and the pointer stored in BRSR. The expression from BSA (the address in BRSR), PID (the pointer in BRSR), and IA (the instruction address before branch occurs) is as follows: IA = BSA – 2 * PID.

   Notes are needed when an interrupt (a branch) is issued before the branch destination instruction is executed. In case of the next figure, the instruction “Exec” executed immediately before branch is calculated by IA = BSA – 2 * PID. However, when branch “branch” has delay slot and the destination address is 4n + 2 address, the address “Dest” which is specified by branch instruction is stored in BRSR (Dest = BSA). Therefore, as IA = BSA – 2 * PID is not applied to this case, this PID is invalid. The case where BSA is 4n + 2 boundary is applied only to this case and then some cases are classified as follows:

```
Exec:branch  Dest
Dest:instr          (not executed)
     interrupt
```

```
Int: interrupt routine
```

If the PID value is odd, instruction buffer indicates PID+2 buffer. However, these expressions in this table are accounted for it. Therefore, the true branch source address is calculated with BSA and PID values stored in BRSR.

3. The branch address before branch occurrence, IA, has different values due to some kinds of branch.

   a. Branch instruction

      The branch instruction address

   b. Repeat

      The instruction before the last instruction of a repeat loop

      ```
      Repeat_Start:inst (1);   ---> BRDR
                   inst (2);
                     :
                   inst (n-1); --> the address calculated from BRSR
      Repeat_End: inst (n);
      ```

   c. Interrupt

      The last instruction executed before interrupt

      Thc top addrcss of intcrrupt routinc is storcd in BRDR.

   In a repeat loop with instructions less than three, no instruction fetch cycle appears and branch source address is unknown. Therefore, PC trace is disabled.

4. 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. When reading BRDR, longword access should be used. Also, the PC trace has a trace pointer, which initially points to the bottom of the queues. The first pair of branch addresses will be stored at the bottom of thc qucucs, thcn push up whcn ncxt pairs comc into thc qucucs. Thc tracc pointcr will points to the next branch address to be executed, unless it got push out of the queues. When the branch address has been executed, the trace pointer will shift down to next pair of addresses, until it reaches the bottom of the queues. After switching the PCTE bit (in BRCR) off and on, the values in the queues are invalid. The read pointer stay at the position before PCTE is switched, but the trace pointer restart at the bottom of the queues.

10.3.8 Usage Examples

Break Condition Specified to a CPU Instruction Fetch Cycle

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: CPU/instruction fetch (after instruction execution)/read (operand size is not included in the condition)

     No ASID check is included

   - Channel B

     Address: H'00008010, Address mask: H'00000006

     Data: H'00000000, Data mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/read (operand size is not included in the condition)

     No ASID check is included

   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.

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, BASRA = H'80, BASRB = H'70

   Specified conditions: Channel A/channel B sequence mode

   - Channel A

     Address: H'00037226, Address mask: H'00000000, ASID = H'80

     Bus cycle: CPU/instruction fetch (before instruction execution)/read/word

   - Channel B

     Address: H'0003722E, Address mask: H'00000000, ASID = H'70

     Data: H'00000000, Data mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/read/word

   An instruction with ASID = H'80 and address H'00037226 is executed, and a user break occurs before an instruction with ASID = H'70 and address H'0003722E is executed.

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: CPU/instruction fetch (before instruction execution)/write/word

     No ASID check is included

   - Channel B

     Address: H'00031415, Address mask: H'00000000

     Data: H'00000000, Data mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/read (operand size is not included in the condition)

     No ASID check is included

   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.

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, BASRA = H'80, BASRB = H'70

   Specified conditions: Channel A/channel B sequence mode

   - Channel A

     Address: H'00037226, Address mask: H'00000000, ASID: H'80

     Bus cycle: CPU/instruction fetch (before instruction execution)/write/word

   - Channel B

     Address: H'0003722E, Address mask: H'00000000, ASID: H'70

     Data: H'00000000, Data mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/read/word

   Since instruction fetch is not a write cycle on channel A, a sequence condition does not match. Therefore, no user break occurs.

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: CPU/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: CPU/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 before the fifth instruction execution after instructions of address H'00001000 are executed four times.

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, BASRA = H'80, BASRB = H'70

   Specified conditions: Channel A/channel B independent mode

   - Channel A

     Address: H'00008404, Address mask: H'00000FFF, ASID: H'80

     Bus cycle: CPU/instruction fetch (after instruction execution)/read (operand size is not included in the condition)
   - Channel B

     Address: H'00008010, Address mask: H'00000006, ASID: H'70

     Data: H'00000000, Data mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/read (operand size is not included in the condition)

   A user break occurs after an instruction with ASID = H'80 and address H'00008000 to H'00008FFE is executed or before instructions with ASID = H'70 and addresses H'00008010 to H'00008016 are executed.

Break Condition Specified to a CPU Data Access Cycle

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, BASRA = H'80, BASRB = H'70

   Specified conditions: Channel A/channel B independent mode

   - Channel A

     Address: H'00123456, Address mask: H'00000000, ASID: H'80

     Bus cycle: CPU/data access/read (operand size is not included in the condition)

   - Channel B

     Address: H'000ABCDE, Address mask: H'000000FF, ASID: H'70

     Data: H'0000A512, Data mask: H'00000000

     Bus cycle: CPU/data access/write/word

   On channel A, a user break occurs with ASID = H'80 during longword read to address H'00123454, word read to address H'00123456, or byte read to address H'00123456. On channel B, a user break occurs with ASID = H'70 when word H'A512 is written in addresses H'000ABC00 to H'000ABCFE.

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: CPU/data access/read/word

     No ASID check is included

   - Channel B

     Y Address: H'0001F000, Address mask: H'FFFF0000

     Data: H'00004567, Data mask: H'00000000

     Bus cycle: CPU/data access/write/word

     No ASID check is included

   On channel A, a user break occurs during word read to address H'01000000 on the memory space. On channel B, a user break occurs when word H'4567 is written in address H'0001F000 on Y memory space. The X/Y-memory space is changed by a mode specification. Section 3.1, Memory Map, describes the memory space in detail.

Break Condition Specified to a DMAC Data Access Cycle

1. Register specifications:
   BARA = H'00314156, BAMRA = H'00000000, BBRA = H'0094, BARB = H'00055555, BAMRB = H'00000000, BBRB = H'00A9, BDRB = H'00000078, BDMRB = H'0000000F, BRCR = H'00000080, BASRA = H'80, BASRB = H'70

   Specified conditions: Channel A/channel B independent mode

   - Channel A
     Address: H'00314156, Address mask: H'00000000, ASID: H'80
     Bus cycle: DMAC/instruction fetch/read (operand size is not included in the condition)
   - Channel B
     Address: H'00055555, Address mask: H'00000000, ASID: H'70
     Data: H'00000078, Data mask: H'0000000F
     Bus cycle: DMAC/data access/write/byte

   On channel A, no user break occurs since instruction fetch is not performed in DMAC cycles. On channel B, a user break occurs with ASID = H'70 when the DMAC writes byte H'7* in address H'00055555.

Break Condition Specified to On-Chip I/O Access or LDTLB Instruction Execution

1. Break condition set for on-chip I/O access
   Register specifications:

Register	Value
BRCR	H'00010000

   Specified conditions: On-chip I/O break enabled (other register settings have no effect).
   When an access to the register occurs, on-chip I/O access break is generated.

2. Break condition set for LDTLB instruction execution
   Register specifications:

Register	Value
BRCR	H'00020000

   Specified conditions: LDTLB execution break enabled (other register settings have no effect).
   When an LDTLB instruction is executed, an LDTLB execution break is generated.

10.3.9 Notes

1. Only CPU can read/write UBC registers.
2. UBC cannot monitor CPU and DMAC access in the same channel.
3. Notes in specification of sequential break are described below:
    a. A condition match occurs when B-channel match occurs in a bus cycle after an A-channel match occurs in another bus cycle in sequential break setting. Therefore, no condition match occurs even if a bus cycle, in which an A-channel match and a channel B match occur simultaneously, is set.
    b. Since the CPU has a pipeline configuration, the pipeline determines the order of an instruction fetch cycle and a memory cycle. Therefore, when a channel condition matches in the order of bus cycles, a sequential condition is satisfied.
    c. When the bus cycle condition for channel A is specified as a break before execution (PCBA = 0 in BRCR) and an instruction fetch cycle (in BBRA), the attention is as follows. A break is issued and condition match flags in BRCR are set to 1, when the bus cycle conditions both for channels A and B match simultaneously.
4. The change of a UBC register value is executed in MA (memory access) stage. Therefore, even if the break condition matches in the instruction fetch address following the instruction in which the pre-execution break is specified as the break condition, no break occurs. In order to know the timing UBC register is changed, read the last written register. Instructions after then are valid for the newly written register value.
5. Notes in specifying the instruction during repeat execution with repeat instruction as the break condition are as follows: When the instruction during repeat execution is specified as the break condition,
    a. The break is not issued during repeat execution, which has fewer than three instructions.
    b. When the execution times break is set, no instruction fetch from memory occurs during repeat execution under three instructions. Therefore, the execution times register BETR is not decreased.
6. The branch instruction should not be executed as soon as PC trace register BRSR and BRDR are read.
7. When PC breaks and TLB exceptions or errors occur in the same instruction. The priority is as follows:
    a. Break and instruction fetch exceptions: Instruction fetch exception occurs first.
    b. Break before execution and operand exception: Break before execution occurs first.
    c. Break after execution and operand exception: Operand exception occurs first.

Section 11 Power-Down Modes

11.1 Overview

In the power-down modes, all CPU and some on-chip supporting module functions are halted. This lowers power consumption.

11.1.1 Power-Down Modes

The SH7729 has three power-down modes:

1. Sleep mode
2. Standby mode
3. Module standby function (TMU, RTC, and SCI on-chip supporting modules)

Table 11.1 shows the transition conditions for entering the modes from the program execution state, as well as the CPU and supporting module states in each mode and the procedures for canceling each mode.

Table 11.1Power-Down Modes

		State							
Mode	**Transition Conditions**	**CPG**	**CPU**	**CPU Reg-ister**	**On-Chip Memory**	**On-Chip Peripheral Modules**	**Pins**	**External Memory**	**Canceling Procedure**
Sleep mode	Execute SLEEP instruction with STBY bit cleared to 0 in STBCR	Runs	Halts	Held	Held	Run	Held	Refresh	1. Interrupt 2. Reset
Standby mode	Execute SLEEP instruction with STBY bit set to 1 in STBCR	Halts	Halts	Held	Held	Halt*1	Held	Self-refresh	1. Interrupt 2. Reset
Module standby	Set MSTP bit of STBCR to 1	Runs	Runs	Held	Held	Specified module halts	*2	Refresh	1. Clear MSTP bit to 0 2. Reset
Hardware standby mode	Drive CA pin low	Halts	Halts	Held	Held	Halt*3	Held	Self-refresh	Power-on reset

Notes: 1. The RTC still runs if the START bit in RCR2 is set to 1 (see section 12, Realtime Clock (RTC)). TMU still runs when output of the RTC is used as input to its counter (see section 11, Timer (TMU)).

2. Depends on the on-chip supporting module.

TMU external pin: Held

SCI external pin: Reset

3. The RTC still runs if the START bit in RCR2 is set to 1. TMU does not run.

11.1.2 Register Configuration

Table 11.2 shows the configuration of the control register for the power-down modes.

Table 11.2Register Configuration

Name	Abbreviation	R/W	Initial Value	Access Size	Address
Standby control register	STBCR	R/W	H'00	Byte	H'FFFFFF82
Standby control register 2	STBCR2	R/W	H'00	Byte	H'FFFFFF88

Note: Initialized by power-on resets. This value is not initialized by manual resets but the contents are held.

11.1.3 Pin Configuration

Table 11.3 lists the pins used for the power-down modes.

Table 11.3Pin Configuration

Pin Name	Symbol	I/O	Description
Processing state 1	STATUS1	O	Operating state of the processor.
Processing state 0	STATUS0		HH: Reset, HL: Sleep mode, LH: Standby mode, LL: Normal operation
Wakeup from standby mode	WAKEUP	0	Low active assert after accepting wakeup interrupt in standby mode until returning to normal operation with WDT overflow.*

Note: H means high level, and L means low level. In the sleep mode that sets the powerdown mode. The STBCR is initialized to H'00 by a power-on reset. Always set bits 6–3 to 0 when writing to the STBCR register.

11.2 Register Description

11.2.1 Standby Control Register (STBCR)

The standby control register (STBCR) is an 8-bit read/write register that sets the power-down mode. STBCR is initialized to H'00 by a power-on reset. Always set bits 6–3 to 0 when writing to the STBCR register.

Bit:	7	6	5	4	3	2	1	0
Bit name:	STBY	—	—	—	—	MSTP2	MSTP1	MSTP0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R	R	R	R/W	R/W	R/W

Bit 7—Standby (STBY): Specifies transition to standby mode.

Bit 7: STBY	Description
0	Executing SLEEP instruction puts the chip into sleep mode. (Initial value)
1	Executing SLEEP instruction puts the chip into standby mode.

Bits 6 to 3—Reserved: These bits always read 0. The write value should always as 0.

Bit 2—Module Standby 2 (MSTP2): Specifies halting the clock supply to the timer unit TMU (an on-chip supporting module). When the MSTP2 bit is set to 1, the supply of the clock to the TMU is halted.

Bit 2: MSTP2	Description
0	TMU runs. (Initial value)
1	Clock supply to TMU is halted.

Bit 1—Module Standby 1 (MSTP1): Specifies halting the clock supply to the realtime clock RTC (an on-chip supporting module). When the MSTP1 bit is set to 1, the supply of the clock to RTC is halted. When the clock halts, all RTC registers become inaccessible, but the counter keeps running.

Bit 1: MSTP1	Description
0	RTC runs. (Initial value)
1	Clock supply to RTC is halted.

Bit 0—Module Standby 0 (MSTP0): Specifies halting the clock supply to the serial communication interface SCI (an on-chip supporting module). When the MSTP0 bit is set to 1, the supply of the clock to the SCI is halted.

Bit 0: MSTP0	Description
0	SCI operates. (Initial value)
1	Clock supply to SCI is halted.

11.2.2 Standby Control Register 2 (STBCR2)

The standby control register 2 (STBCR2) is a read/write 8-bit register that sets the power-down mode. The STBCR2 is initialized to H'00 during a power-on reset.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	MDCHG	MSTP8	MSTP7	MSTP6	MSTP5	MSTP4	MSTP3
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 6—Pin MD5 to MD0 Control (MDCHG): Specifies whether or not pins MD5 to MD0 are changed in standby mode. When this bit is set to 1, the MD5 to MD0 pin values are latched when returning from standby mode by means of a reset or interrupt.

Bit 6: MDCHG	Description
0	Pin MD5 to MD0 change is not performed in standby mode (Initial value)
1	Pin MD5 to MD0 change is performed in standby mode

Bit 5— Module Standby 8 (MSTP8): Specifies halting the clock supply to the user break controller UBC (an on-chip supporting module). When the MSTP8 bit is set to 1, the supply of the clock to the UBC is halted.

Bit 5: MSTP8	Description
0	UBC runs (Initial value)
1	Clock supply to UBC is halted

Bit 4—Module Stop 7 (MSTP7): Specifies halting of clock supply to the DMAC (an on-chip peripheral module). When the MSTP7 bit is set to 1, the supply of the clock to the DMAC is halted.

Bit 4: MSTP7	Description
0	DMAC runs (Initial value)
1	Clock supply to DMAC halted

Bit 3—Module Stop 6 (MSTP6): Specifies halting of clock supply to the DAC (an on-chip peripheral module). When the MSTP6 bit is set to 1, the supply of the clock to the DAC is halted.

Bit 3: MSTP6	Description
0	DAC runs (Initial value)
1	Clock supply to DAC halted

Bit 2—Module Stop 5 (MSTP5): Specifies halting of clock supply to the ADC (an on-chip peripheral module). When the MSTP5 bit is set to 1, the supply of the clock to the ADC is halted.

Bit 2: MSTP5	Description
0	ADC runs (Initial value)
1	Clock supply to ADC halted

Bit 1—Module Stop 4 (MSTP4): Specifies halting the clock supply to the serial communication interface with FIFO (an on-chip peripheral module). When the MSTP1 bit is set to 1, the supply of the clock to the SCIF is halted.

Bit 1: MSTP4	Description
0	SCIF runs (Initial value)
1	Clock supply to SCIF halted

Bit 0—Module Stop 3 (MSTP3): Specifies halting the clock supply to the infrared data association interface with FIFO (an on-chip peripheral module). When the MSTP1 bit is set to 1, the supply of the clock to the IrDA is halted.

Bit 0: MSTP3	Description
0	IrDA runs (Initial value)
1	Clock supply to IrDA halted

11.3 Sleep Mode

11.3.1 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 supporting modules continue to run during sleep mode and the clock continues to be output to the CKIO pin. In sleep mode, the STATUS1 pin is set high and the STATUS0 pin low. However, during a refresh cycle, the STATUS1 pin and STATUS0 pin are both set low.

11.3.2 Canceling Sleep Mode

Sleep mode is canceled by an interrupt (NMI, IRL, on-chip supporting module) or reset. Interrupts are accepted during sleep mode even when the BL bit in the SR register is 1. If necessary, save SPC and SSR in the stack before excuting the SLEEP instruction.

Canceling with an Interrupt: When an NMI, IRL or on-chip supporting module interrupt occurs, sleep mode is canceled and interrupt exception handling is executed. A code indicating the interrupt source is set in the INTEVT and INTEVT2 registers.

Canceling with a Reset: Sleep mode is canceled by a power-on reset or a manual reset.

11.4 Standby Mode

11.4.1 Transition to Standby Mode

To enter standby mode, set the STBY bit to 1 in STBCR, then execute the SLEEP instruction. The chip moves from the program execution state to standby mode. In standby mode, power consumption is greatly reduced by halting not only the CPU, but the clock and on-chip supporting modules as well. The clock output from the CKIO pin also halts. CPU and cache register contents are held, but some on-chip supporting modules are initialized. Table 11.4 lists the states of registers in standby mode.

Table 11.4Register States in Standby Mode

Module	Registers Initialized	Registers Retaining Data
Interrupt controller	—	All registers
Break controller	—	All registers
Bus state controller	—	All registers
On-chip clock pulse generator	—	All registers
Timer unit	TSTR register	Registers other than TSTR
Realtime clock	—	All registers
A/D converter	All registers	—
D/A converter	—	All registers

The procedure for moving to standby mode is as follows:

1. Clear the TME bit in the WDT's timer control register (WTCSR) to 0 to stop the WDT. Set the WDT's timer counter (WTCNT) and the CKS2–CKS0 bits of the WTCSR register to appropriate values to secure the specified oscillation settling time.
2. When PLL circuit 1 is running in clock modes 3–6, clear the PSTBY and PLLEN bits in the frequency control register (FRQCR) to 0 to stop PLL circuit 1.
3. After the STBY bit in the STBCR register is set to 1, a SLEEP instruction is executed.
4. Standby mode is entered and the clocks within the chip are halted. The STATUS1 pin output goes low and the STATUS0 pin output goes high.

11.4.2 Canceling Standby Mode

Standby mode is canceled by an interrupt (NMI, IRQ, IRL, PINT, or on-chip supporting module) or a reset.

Canceling with an Interrupt: The on-chip WDT can be used for hot starts. When the chip detects an NMI, IRL, IRQ, PINT,*1 or on-chip supporting module (except the interval timer)*2 interrupt, the clock will be supplied to the entire chip and standby mode canceled after the time set in the WDT's timer control/status register has elapsed. The STATUS1 and STATUS0 pins both go low. Interrupt handling then begins and a code indicating the interrupt source is set in the INTEVT and INTEVT2 registers. After branching to the interrupt processing routine occurs, clear the STBY bit in the STBCR register. The WTCNT stops automatically. If the STBY bit is not cleared, WTCNT continues operation and transits to the standby mode *3 when it reaches H'80. This function prevents the data from being destroyed due to a rising voltage under an unstable power supply. Interrupts are accepted during standby mode even when the BL bit in the SR register is 1. Immediately after an interrupt is detected, the phase of the clock output of the CKIO pin may be unstable, until the processor starts interrupt handling. (The canceling condition is that the IRL3–IRL0 level is higher than the mask level in the I3–I0 bits in the SR register.)

Notes: 1. When the RTC is being used, standby mode can be canceled using IRL3–IRL0, IRQ4–IRQ0 or PINT0/1.

2. Standby mode can be canceled with an RTC or TMU (only when running on the RTC clock) interrupt.

Canceling with a Reset: Standby mode can be canceled with a reset (power-on or manual). Keep the RESET pin low until the clock oscillation settles. The internal clock will continue to be output to the CKIO pin.

11.4.3 Clock Pause Function

In standby mode, the clock input from the EXTAL pin or CKIO pin can be halted and the frequency can be changed. This function is used as follows:

1. Enter standby mode using the appropriate procedures.
2. Once standby mode is entered and the clock stopped within the chip, the STATUS1 pin output is low and the STATUS0 pin output is high.
3. Once the STATUS1 pin goes low and the STATUS0 pin goes high, the input clock is stopped or the frequency is changed.
4. When the frequency is changed, an NMI, IRL, IRQ, PINT or on chip supporthing mudule (except the internal timer) interrupt is input after the change. When the clock is stopped, the same interrupts are input after the clock is applied.
5. After the time set in the WDT has elapsed, the clock starts being applied internally within the chip, the STATUS1–STATUS0 pins both go low, interrupts are handled, and operation resumes.

11.5 Module Standby Function

11.5.1 Transition to Module Standby Function

Setting the standby control register MSTP2–MSTP0 bits to 1 halts the supply of clocks to the corresponding on-chip supporting modules. This function can be used to reduce the power consumption in sleep mode. The module standby function holds the status prior to halt of the external pins of the on-chip supporting modules. TMU external pins hold their status prior to the halt. SCI external pins go to the reset state. With a few exceptions, all registers hold their values.

Bit	Value	Description
MSTP8	0	UBC runs.
	1	Supply of clock to UBC halted.
MSTP7	0	DMAC runs.
	1	Supply of clock to DMAC halted.
MSTP6	0	DAC runs.
	1	Supply of clock to DAC halted.
MSTP5	0	ADC runs.
	1	Supply of clock to ADC halted.
MSTP4	0	SCIF runs.
	1	Supply of clock to SCIF halted.
MSTP3	0	IRDA runs.
	1	Supply of clock to SCIF1 halted.
MSTP2	0	TMU runs.
	1	Supply of clock to TMU halted. Registers initialized.*1
MSTP1	0	RTC runs.
	1	Supply of clock to RTC halted. Register access prohibited.*2
MSTP0	0	SCI operates.
	1	Supply of clock to SCI halted.

Notes: 1. The registers initialized are the same as in standby mode (table 11.4).
2. The counter runs.

11.5.2 Clearing the Module Standby Function

The module standby function can be cleared by clearing the MSTPSLP0 MSTP8–MSTP0 bits to 0, or by a power-on reset or manual reset.

11.6 Timing of STATUS Pin Changes

The timing of STATUS1 and STATUS0 pin changes is shown in figures 11.1 through 11.9.

The meaning of the STATUS descriptions is as follows:

Reset:	HH (STATUS1 high, STATUS0 high)
Sleep:	HL (STATUS1 high, STATUS0 low)
Standby:	LH (STATUS1 low, STATUS0 high)
Normal:	LL (STATUS1 low, STATUS0 low)

The meaning of the clock units is as follows:

Bcyc:	Bus clock cycle
Pcyc:	Peripheral clock cycle

11.6.1 Timing for Resets

Power-On Reset (Clock Modes 0, 1, 2, and 7):

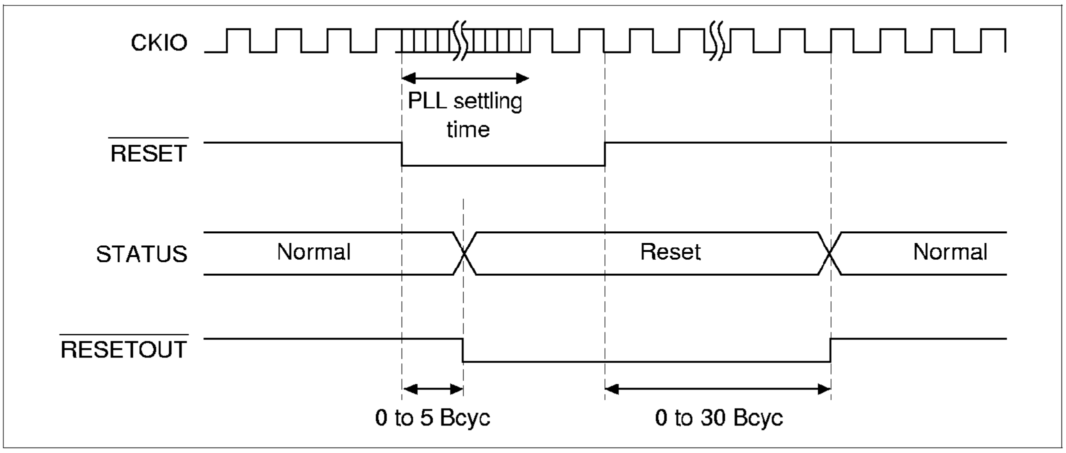


Figure 11.1 Power-On Reset (Clock Mode 0, 1, 2, and 7) STATUS Output

Power-On Reset (Clock Modes 3 and 4):

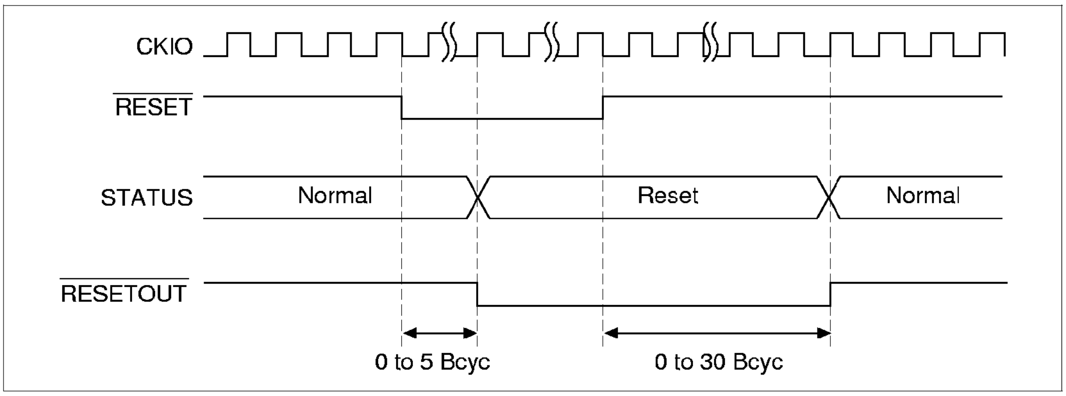
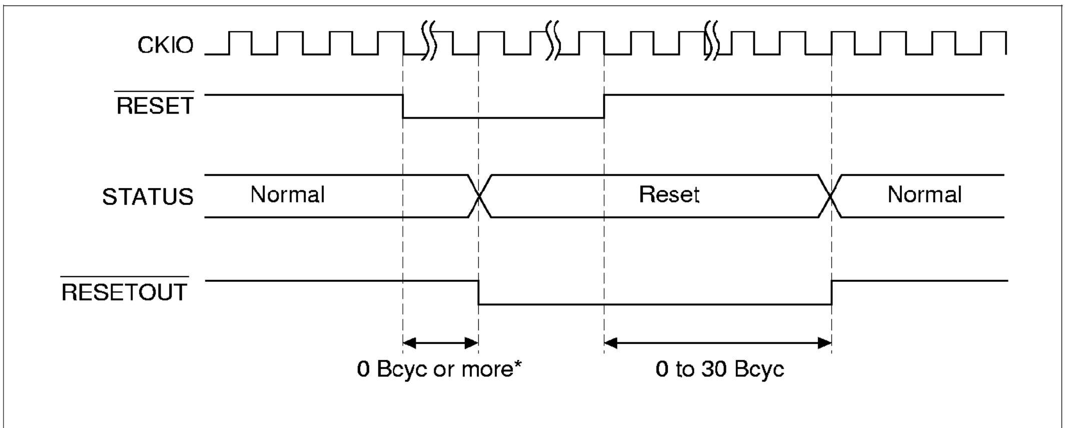


Figure 11.2 Power-On Reset (Clock Mode 3 and 4) STATUS Output

Manual Reset:



Note: During manual reset, STATUS becomes HH (reset) and the internal reset begins after waiting for the executing bus cycle to end.

Figure 11.3 Manual Reset STATUS Output

11.6.2 Timing for Canceling Standbys

Standby to Interrupt:

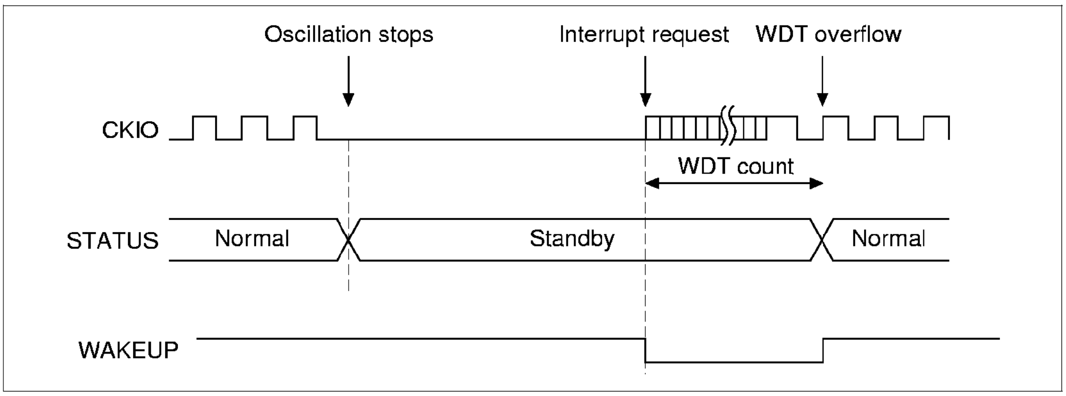
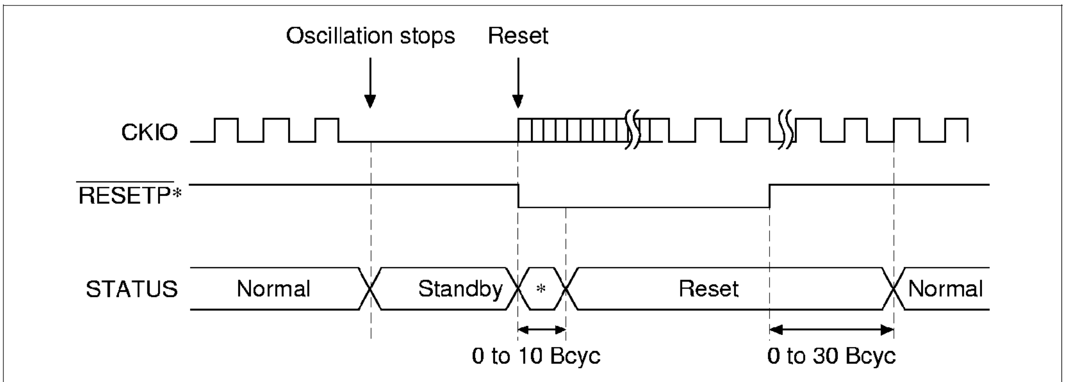


Figure 11.4 Standby to Interrupt STATUS Output

Standby to Power-On Reset:

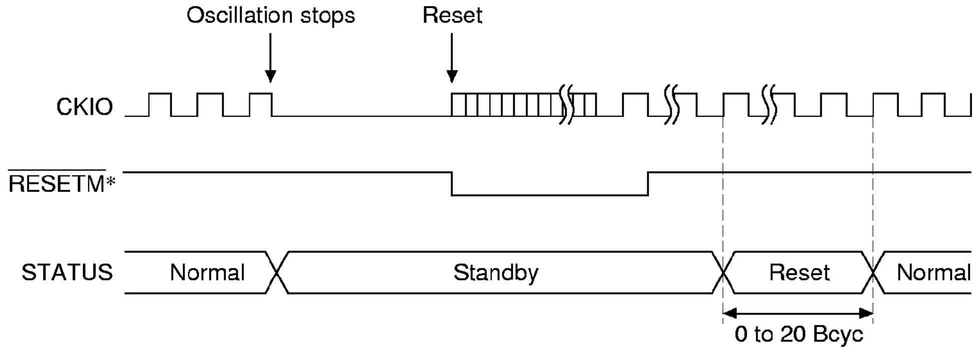


Note: When standby mode is cleared with a power-on reset, the WDT does not count. Keep RESETP low during the PLL's oscillation settling time.

*: Undefined

Figure 11.5 Standby to Power-On Reset STATUS Output

Standby to Manual Reset:



Note: When standby mode is cleared with a manual reset, the WDT does not count. Keep RESETM low during the PLL's oscillation settling time.

Figure 11.6 Standby to Manual Reset STATUS Output

11.6.3 Timing for Canceling Sleep Mode

Sleep to Interrupt:

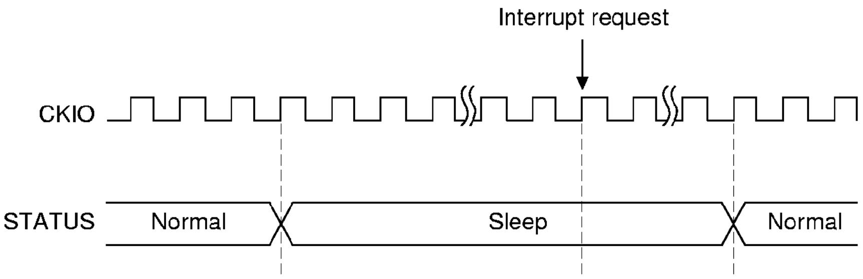
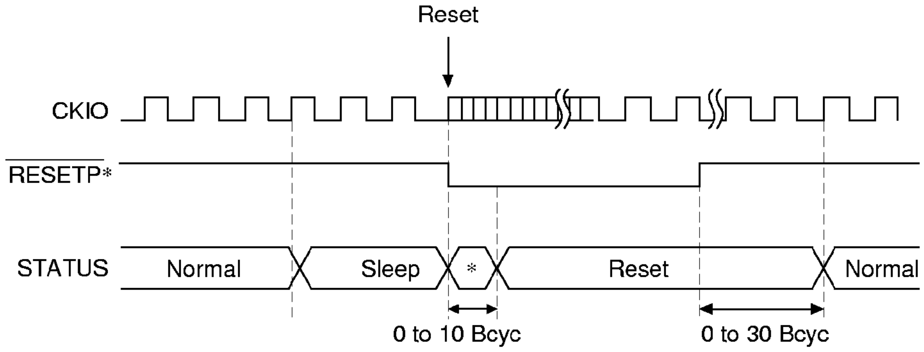


Figure 11.7 Sleep to Interrupt STATUS Output

Sleep to Power-On Reset:

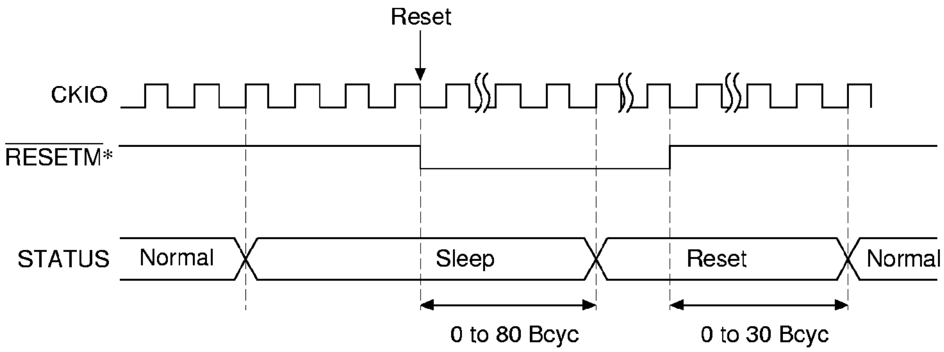


Note: When the PLL1's multiplication ratio is changed by a power-on reset, keep RESETP low during the PLL's oscillation settling time.

*: Undefined

Figure 11.8 Sleep to Power-On Reset STATUS Output

Sleep to Manual Reset:



Note: Keep RESETM low until the STATUS becomes reset.

Figure 11.9 Sleep to Manual Reset STATUS Output

11.7 Hardware Standby Function

11.7.1 Transition to Hardware Standby Mode

Driving the CA pin low causes a transition to hardware standby mode. In hardware standby mode, all modules except those operating on an RTC clock are halted, as in the standby mode entered on execution of a SLEEP instruction ((software) standby mode).

Hardware standby mode differs from (software) standby mode as follows.

1. Interrupts and manual resets are not accepted.
2. The TMU does not operate.
3. The RTC continues to operate even if power is not supplied to power supply pins other than those for RTC power. In this case, all output pins go to the non-drive state.

Operation when a low-level signal is input at the CA pin depends on the CPG state, as follows.

1. In standby mode

   The clock remains stopped and the chip enters the hardware standby state. Acceptance of interrupts and manual resets is disabled, TCLK output is fixed low, and the TMU halts.
2. During WDT operation when standby mode is canceled by an interrupt

   The chip enters hardware standby mode after standby mode is canceled and the CPU resumes operation.
3. In sleep mode

   The chip enters hardware standby mode after sleep mode is canceled and the CPU resumes operation.
4. During PLL standby (see section 9.6 for the PLL standby function)

   The chip enters hardware standby mode after the PLL turned off.

Hold the CA pin low in hardware standby mode.

11.7.2 Canceling Hardware Standby Mode

Hardware standby mode can only be canceled by a power-on reset.

When the CA pin is driven high while the RESETP pin is low, clock oscillation is started. Hold the RESETP pin low until clock oscillation stabilizes. When the RESETP pin is driven high, the CPU begins power-on reset processing.

Hardware standby mode cannot be canceled by an interrupt or manual reset.

11.7.3 Hardware Standby Mode Timing

Figures 11.10 and 11.11 show examples of pin timing in hardware standby mode.

The CA pin is sampled using EXTAL2 (32.768 kHz), and a hardware standby request is only recognized when the pin is low for two consecutive clock cycles.

The CA pin must be held low while the chip is in hardware standby mode.

Clock oscillation starts when the CA pin is driven high after the RESETP pin is driven low.

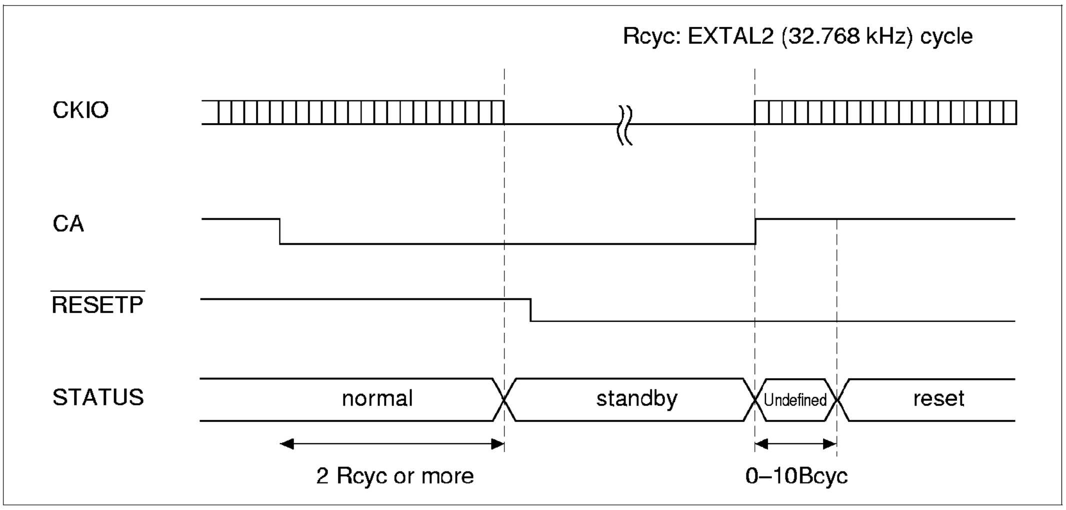


Figure 11.10 Hardware Standby Mode (When CA Goes Low in Normal Operation)

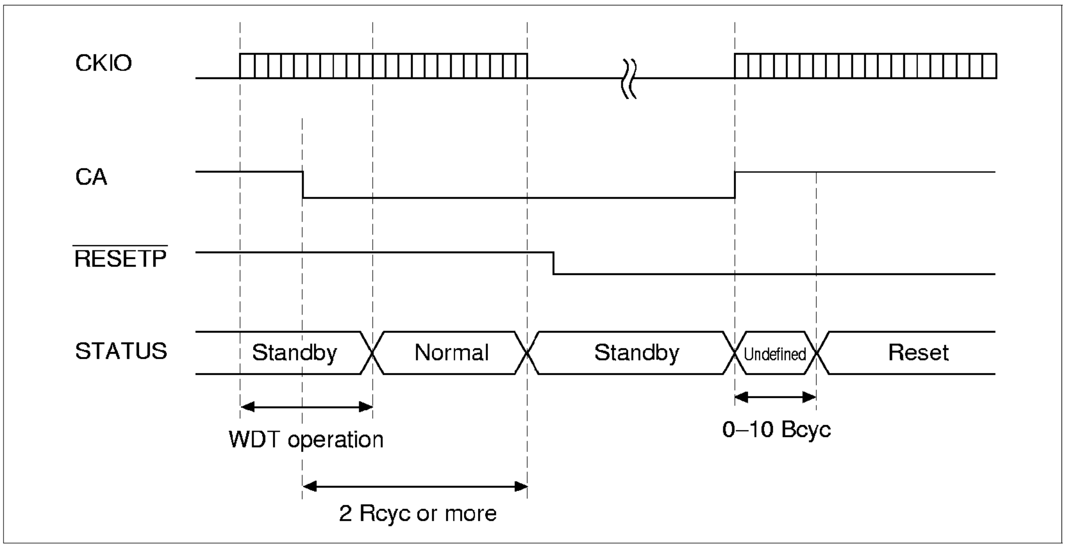


**Figure 11.11 Hardware Standby Mode Timing
(When CA Goes Low during WDT Operation on Standby Mode Cancellation)**

Section 12 On-Chip Oscillation Circuits

12.1 Overview

The clock pulse generator (CPG) supplies all clocks to the processor and controls the power-down modes. The watchdog timer (WDT) is a single-channel timer that counts the clock settling time and is used when clearing standby mode and temporary standbys, such as frequency changes. It can also be used as an ordinary watchdog timer or interval timer.

12.1.1 Features

The CPG has the following features:

- Six clock modes: Selection of six clock modes for different frequency ranges, power consumption, direct crystal input, and external clock input.
- Three clocks generated independently: An internal clock for the CPU, cache, and TLB (Iφ); a peripheral clock (Pφ) for the on-chip supporting modules; and a bus clock (CKIO) for the external bus interface.
- Frequency change function: Internal and peripheral clock frequencies can be changed independently using the PLL circuit and divider circuit within the CPG. Frequencies are changed by software using frequency control register (FRQCR) settings.
- PLL on/off function: Power consumption can be decreased by stopping the PLL circuit when operating at low frequencies.
- Power-down mode control: The clock can be stopped for sleep mode and standby mode and specific modules can be stopped using the module standby function.

The WDT has the following features:

- Can be used to ensure the clock settling time: Use the WDT to cancel 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. Selection of power-on reset or manual reset.
- Generates interrupts in interval timer mode: Internal timer interrupts occur after counter overflow.
- Selection of eight counter input clocks. Eight clocks (×1 to ×1/4096) can be obtained by dividing the peripheral clock.

12.2 Overview of the CPG

12.2.1 CPG Block Diagram

A block diagram of the on-chip clock pulse generator is shown in figure 12.1.

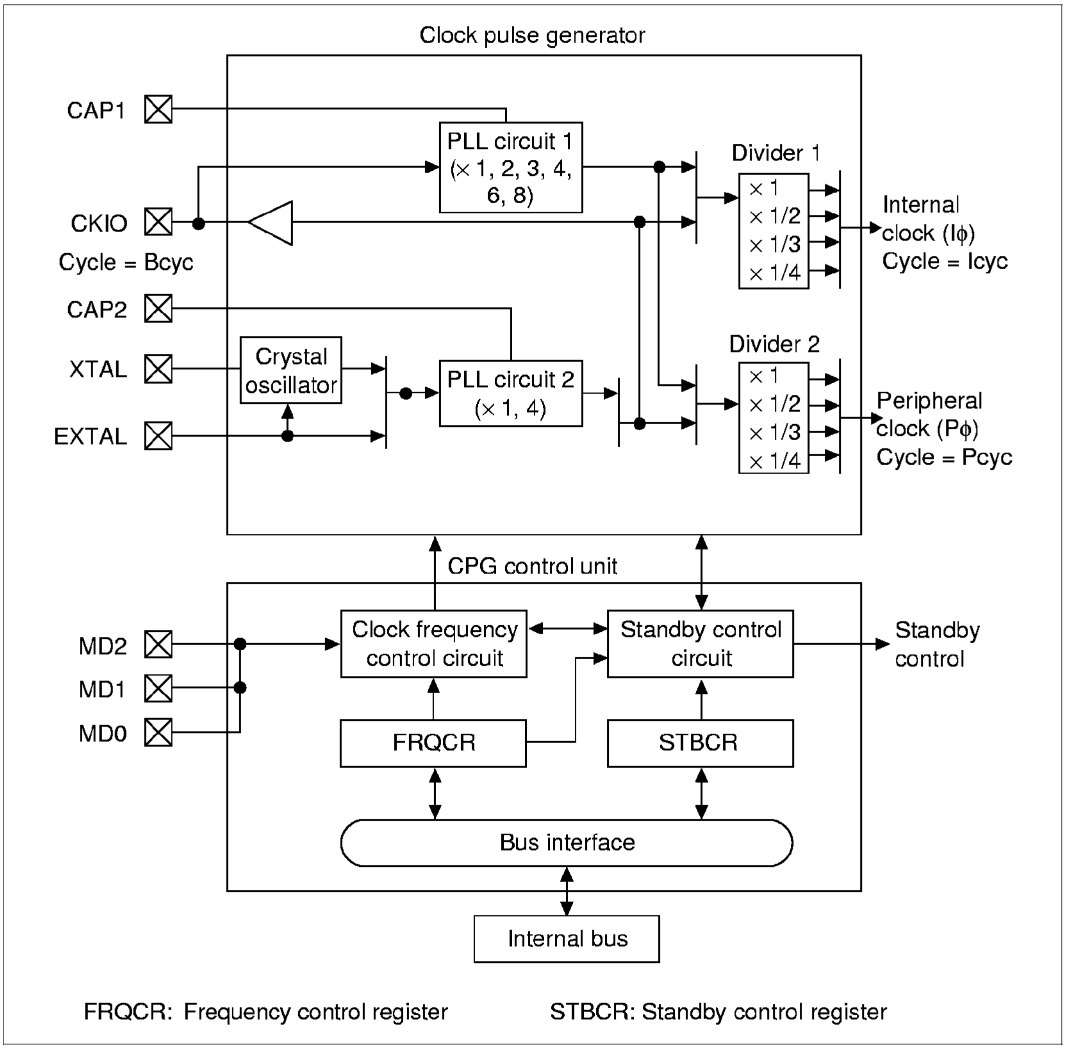


Figure 12.1 Block Diagram of Clock Pulse Generator

The clock pulse generator blocks function as follows:

1. PLL Circuit 1: PLL circuit 1 doubles, triples, quadruples, sextuples, octuples, or leaves unchanged the input clock frequency from the CKIO terminal. The multiplication rate is set by the frequency control register. When this is done, the phase of the leading edge of the internal clock is controlled so that it will agree with the phase of the leading edge of the CKIO pin.
2. PLL Circuit 2: PLL circuit 2 leaves unchanged or quadruples the frequency of the crystal oscillator or the input clock frequency coming from the EXTAL pin. The multiplication ratio is fixed by the clock operation mode. The clock operation mode is set by pins MD0, MD1, and MD2. See table 12.3 for more information on clock operation modes.
3. Crystal Oscillator: This oscillator is used when a crystal oscillator element is connected to the XTAL and EXTAL pins. It operates according to the clock operating mode setting.
4. Divider 1: Divider 1 generates a clock at the operating frequency used by the internal clock. The operating frequency can be 1, 1/2, 1/3, or 1/4 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.
5. Divider 2: Divider 2 generates a clock at the operating frequency used by the peripheral clock. The operating frequencies can be 1, 1/2, 1/3, or 1/4 times the output frequency of PLL Circuit 1 or the clock frequency of the CKIO pin, as long as it stays at or below the clock frequency of the CKIO pin. The division ratio is set in the frequency control register.
6. Clock Frequency Control Circuit: The clock frequency control circuit controls the clock frequency using the MD pin and the frequency control register.
7. Standby Control Circuit: The standby control circuit controls the state of the clock pulse generator and other modules during clock switching and sleep/standby modes.
8. Frequency Control Register: The frequency control register has control bits assigned for the following functions: clock output/non-output from the CKIO pin, on/off control of PLL circuit 1, PLL standby, the frequency multiplication ratio of PLL 1, and the frequency division ratio of the internal clock and the peripheral clock.
9. Standby Control Register: The standby control register has bits for controlling the power-down modes. See section 11, Power-Down Modes, for more information.

12.2.2 CPG Pin Configuration

Table 12.1 lists the CPG pins and their functions.

Table 12.1Clock Pulse Generator Pins and Functions

Pin Name	Symbol	I/O	Description
Mode control pins	MD0	I	Set the clock operating mode.
	MD1	I	
	MD2	I	
Crystal I/O pins (clock input pins)	XTAL	O	Connects a crystal oscillator.
	EXTAL	I	Connects a crystal oscillator. Also used to input an external clock.
Clock I/O pin	CKIO	I/O	Inputs or outputs an external clock. Level can be fixed during output.
Capacitor connection pins for PLL	CAP1	I	Connects capacitor for PLL circuit 1 operation (recommended value 470 pF).
	CAP2	I	Connects capacitor for PLL circuit 2 operation (recommended value 470 pF).

12.2.3 CPG Register Configuration

Table 12.2 shows the CPG register configuration.

Table 12.2Register Configuration

Register Name	Abbreviation	R/W	Initial Value	Address	Access Size
Frequency control register	FRQCR	R/W	H'0102	H'FFFFFF80	16

12.3 Clock Operating Modes

Table 12.3 shows the relationship between the mode control pin (MD2–MD0) combinations and the clock operating modes. Table 12.4 shows the usable frequency ranges in the clock operating modes.

Table 12.3Clock Operating Modes

	Pin Values			Clock I/O		PLL2	PLL1	Divider 1	Divider 2	CKIO
Mode	MD2	MD1	MD0	Source	Output	On/Off	On/Off	Input	Input	Frequency
0	0	0	0	EXTAL	CKIO	On, multiplication ratio: 1	On	PLL1 output	PLL1	(EXTAL)
1	0	0	1	EXTAL	CKIO	On, multiplication ratio: 4	On	PLL1 output	PLL1	(EXTAL) × 4
2	0	1	0	Crystal oscillator	CKIO	On, multiplication ratio: 4	On	PLL1 output	PLL1	(Crystal) × 4
3	0	1	1	EXTAL	CKIO	On, multiplication ratio: 1	Off (initial value)	PLL2 output	PLL2	(EXTAL) × 1
							On	PLL1 output		
4	1	0	0	Crystal oscillator	CKIO	On, multiplication ratio: 1	Off (initial value)	PLL2 output	PLL2	(Crystal) × 1
							On	PLL1 output		
7	1	1	1	CKIO	–	Off	On	PLL1 output	PLL1	(CKIO)

Mode 0: An external clock is input from the EXTAL pin and undergoes waveform shaping by PLL circuit 2 before being supplied inside this LSI. PLL circuit 1 is constantly on, and there are no frequency range restrictions compared to mode 3. An input clock frequency of 16 MHz to 66 MHz can be used, and the CKIO frequency range is 10 MHz to 66 MHz.

As PLL circuit 1 compensates for fluctuations in the CKIO pin load, this mode is suitable for connection of synchronous DRAM.

Mode 1: An external clock is input from the EXTAL pin and its frequency is multiplied by 4 by PLL circuit 2 before being supplied inside this LSI, allowing a low-frequency external clock to be used. An input clock frequency of 5 MHz to 16.7 MHz can be used, and the CKIO frequency range is 20 MHz to 66 MHz.

As PLL circuit 1 compensates for fluctuations in the CKIO pin load, this mode is suitable for connection of synchronous DRAM.

Mode 2: The on-chip crystal oscillator operates, with the oscillation frequency being multiplied by 4 by PLL circuit 2 before being supplied inside this LSI, allowing a low crystal frequency to be used. A crystal oscillation frequency of 5 MHz to 16.7 MHz can be used, and the CKIO frequency range is 20 MHz to 66 MHz.

As PLL circuit 1 compensates for fluctuations in the CKIO pin load, this mode is suitable for connection of synchronous DRAM.

Mode 3: An external clock is input from the EXTAL pin and undergoes waveform shaping by PLL circuit 2 before being supplied inside this LSI. PLL circuit 1 is off in the default state at power-on reset, and PLL circuit 1 can be selected as on or off, enabling power consumption to be kept lower than in mode 0. An input clock frequency of 16 MHz to 33 MHz can be used, and the CKIO frequency range is 16 MHz to 33 MHz.

Mode 4: The on-chip crystal oscillator operates, with its output supplied inside this LSI as a square waveform by PLL circuit 2. PLL circuit 1 is off in the default state at power-on reset, and PLL circuit 1 can be selected as on or off, enabling power consumption to be reduced accordingly. A crystal oscillation frequency of 16 MHz to 33 MHz can be used, and the CKIO frequency range is 16 MHz to 33 MHz.

Mode 7: In this mode, the CKIO pin is an input, an external clock is input to this pin, and undergoes waveform shaping, and also frequency multiplication according to the setting, by PLL circuit 1 before being supplied to this LSI. In modes 0 to 4, the system clock is generated from the output of this LSI's CKIO pin. Consequently, if a large number of ICs are operating on the clock cycle, the CKIO pin load will be large. This mode, however, assumes a comparatively large-scale system. If a large number of ICs are operating on the clock cycle, a clock generator with a number of low-skew clock outputs can be provided, so that the ICs can operate synchronously by distributing the clocks to each one.

As PLL circuit 1 compensates for fluctuations in the CKIO pin load, this mode is suitable for connection of synchronous DRAM.

Table 12.4Available Combination of Clock Mode and FRQCR Values

Clock Mode	FRQCR	PLL1	PLL2	Clock Rate* (I:B:P)	Input Frequency Range	CKIO Frequency Range
0	H'0100	ON (×1)	ON (×1)	1:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz
	H'0101	ON (×1)	ON (×1)	1:1:1/2	20 MHz to 66 MHz	20 MHz to 66 MHz
	H'0102	ON (×1)	ON (×1)	1:1:1/4	20 MHz to 66 MHz	20 MHz to 66 MHz
	H'0111	ON (×2)	ON (×1)	2:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz
	H'0112	ON (×2)	ON (×1)	2:1:1/2	20 MHz to 66 MHz	20 MHz to 66 MHz
	H'0115	ON (×2)	ON (×1)	1:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz
	H'0116	ON (×2)	ON (×1)	1:1:1/2	20 MHz to 66 MHz	20 MHz to 66 MHz
	H'0122	ON (×4)	ON (×1)	4:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz
	H'0126	ON (×4)	ON (×1)	2:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz
	H'012A	ON (×4)	ON (×1)	1:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz
	H'A100	ON (×3)	ON (×1)	3:1:1	25 MHz to 33 MHz	25 MHz to 33 MHz
	H'A101	ON (×3)	ON (×1)	3:1:1/2	25 MHz to 44 MHz	25 MHz to 44 MHz
	H'E100	ON (×3)	ON (×1)	1:1:1	25 MHz to 33 MHz	25 MHz to 33 MHz
	H'E101	ON (×3)	ON (×1)	1:1:1/2	25 MHz to 44 MHz	25 MHz to 44 MHz
	H'A111	ON (×6)	ON (×1)	6:1:1	20 MHz to 22 MHz	20 MHz to 22 MHz
1, 2	H'0100	ON (×1)	ON (×4)	4:4:4	5 MHz to 8.3 MHz	20 MHz to 33 MHz
	H'0101	ON (×1)	ON (×4)	4:4:2	5 MHz to 16.7 MHz	20 MHz to 66 MHz
	H'0102	ON (×1)	ON (×4)	4:4:1	5 MHz to 16.7 MHz	20 MHz to 66 MHz
	H'0111	ON (×2)	ON (×4)	8:4:4	5 MHz to 8.3 MHz	20 MHz to 33 MHz
	H'0112	ON (×2)	ON (×4)	8:4:2	5 MHz to 16.6 MHz	20 MHz to 66 MHz
	H'0115	ON (×2)	ON (×4)	4:4:4	5 MHz to 8.3 MHz	20 MHz to 33 MHz
	H'0116	ON (×2)	ON (×4)	4:4:2	5 MHz to 16.6 MHz	20 MHz to 66 MHz
	H'0122	ON (×4)	ON (×4)	16:4:4	5 MHz to 8.3 MHz	20 MHz to 33 MHz
	H'0126	ON (×4)	ON (×4)	8:4:4	5 MHz to 8.3 MHz	20 MHz to 33 MHz
	H'012A	ON (×4)	ON (×4)	4:4:4	5 MHz to 8.3 MHz	20 MHz to 33 MHz
	H'A100	ON (×3)	ON (×4)	12:4:4	5 MHz to 8.3 MHz	20 MHz to 33 MHz
	H'A101	ON (×3)	ON (×4)	12:4:2	5 MHz to 11 MHz	20 MHz to 44 MHz
	H'E100	ON (×3)	ON (×4)	4:4:4	5 MHz to 8.3 MHz	20 MHz to 33 MHz
	H'E101	ON (×3)	ON (×4)	4:4:2	5 MHz to 11 MHz	20 MHz to 44 MHz

Table 12.4Available Combination of Clock Mode and FRQCR Values (cont)

Clock Mode	FRQCR	PLL1	PLL2	Clock Rate* (I:B:P)	Input Frequency Range	CKIO Frequency Range
3, 4	H'0100	OFF	ON (×1)	1:1:1	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'0101	OFF	ON (×1)	1:1:1/2	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'0102	OFF	ON (×1)	1:1:1/4	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01D0	ON (×2)	ON (×1)	2:1:1	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01D1	ON (×2)	ON (×1)	2:1:1/2	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01D2	ON (×2)	ON (×1)	2:1:1/4	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01D4	ON (×2)	ON (×1)	1:1:1	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01D5	ON (×2)	ON (×1)	1:1:1/2	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01D6	ON (×2)	ON (×1)	1:1:1/4	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'81C0	ON (×3)	ON (×1)	3:1:1	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'81C1	ON (×3)	ON (×1)	3:1:1/2	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'C1C0	ON (×3)	ON (×1)	1:1:1	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'C1C1	ON (×3)	ON (×1)	1:1:1/2	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01E0	ON (×4)	ON (×1)	4:1:1	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01E1	ON (×4)	ON (×1)	4:1:1/2	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01E4	ON (×4)	ON (×1)	2:1:1	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01E5	ON (×4)	ON (×1)	2:1:1/2	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01E6	ON (×4)	ON (×1)	2:1:1/4	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01E8	ON (×4)	ON (×1)	1:1:1	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01E9	ON (×4)	ON (×1)	1:1:1/2	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01EA	ON (×4)	ON (×1)	1:1:1/4	16 MHz to 33 MHz	16 MHz to 33 MHz
	H'01F0	ON (×8)	ON (×1)	8:1:1	16 MHz to 16.6 MHz	16 MHz to 16.6 MHz
	H'81D0	ON (×6)	ON (×1)	6:1:1	16 MHz to 22 MHz	16 MHz to 22 MHz
7	H'0100	ON (×1)	OFF	1:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz
	H'0101	ON (×1)	OFF	1:1:1/2	20 MHz to 66 MHz	20 MHz to 66 MHz
	H'0102	ON (×1)	OFF	1:1:1/4	20 MHz to 66 MHz	20 MHz to 66 MHz
	H'0111	ON (×2)	OFF	2:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz
	H'0112	ON (×2)	OFF	2:1:1/2	20 MHz to 66 MHz	20 MHz to 66 MHz
	H'0115	ON (×2)	OFF	1:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz
	H'0116	ON (×2)	OFF	1:1:1/2	20 MHz to 66 MHz	20 MHz to 66 MHz
	H'0122	ON (×4)	OFF	4:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz
	H'0126	ON (×4)	OFF	2:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz

	H'012A	ON (× 4)	OFF	1:1:1	20 MHz to 33 MHz	20 MHz to 33 MHz

Table 12.4Available Combination of Clock Mode and FRQCR Values (cont)

Clock Mode	FRQCR	PLL1	PLL2	Clock Rate* (I:B:P)	Input Frequency Range	CKIO Frequency Range
7	H'A100	ON (× 3)	OFF	3:1:1	25 MHz to 33 MHz	25 MHz to 33 MHz
	H'E100	ON (× 3)	OFF	1:1:1	25 MHz to 33 MHz	25 MHz to 33 MHz
	H'E101	ON (× 3)	OFF	1:1:1/2	25 MHz to 44 MHz	25 MHz to 44 MHz
	H'A111	ON (× 6)	OFF	6:1:1	20 MHz to 22 MHz	20 MHz to 22 MHz

Max. frequency: Iφ = 133 MHz, Bφ (CKIO) = 66 MHz, Pφ = 33 MHz

Note: * Taking input clock as 1

Cautions:

1. When clock operating modes 3, 4 are used:
   - The on/off state of PLL circuit 1 is set by the frequency control register.
   - PLL circuit 1 is initialized to the off state by a power-on reset.
   - Always turn PLL circuit 1 off before going into standby mode.
2. The input to divider 1 becomes the output of:
   - PLL circuit 1 when PLL circuit 1 is on.
   - PLL circuit 2 when PLL circuit 1 is off and PLL circuit 2 is on.
3. The input of divider 2 becomes the output of:
   - PLL circuit 1 when the clock operating mode is 0–2 or 7.
   - PLL circuit 2 when the clock operating mode is 3, 4 and PLL circuit 2 is on.
4. The frequency of the internal clock (Iφ) 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 when PLL circuit 1 is on.
   - Equal to the frequency of CKIO pin when PLL circuit 1 is off.
   - Do not set the internal clock frequency lower than the CKIO pin frequency.
5. The frequency of the peripheral clock (Pφ) becomes:
   - The product of the frequency of the CKIO pin, the frequency multiplication ratio of PLL circuit 1, and the division ratio of divider 2 when the clock operating mode is 0–2 or 7.
   - The product of the frequency of the CKIO pin and the division ratio of divider 2 when the clock operating mode is 3, 4.
   - The peripheral clock frequency should not be set higher than the frequency of the CKIO pin, higher than 33 MHz, or lower than 1/8 the internal clock (Iφ).

6. The output frequency of PLL circuit 1 is the product of the CKIO frequency and the multiplication ratio of PLL circuit 1. This frequency should be equal to or lower than 133 MHz.
7. × 1, × 2, × 3, × 4, × 6, or × 8 can be used as the multiplication ratio of PLL circuit 1. × 1, × 1/2, × 1/3, × 1/4, and × 1/6 can be selected as the division ratios of dividers 1 and 2. Set the rate in the frequency control register. The on/off state of PLL circuit 2 is determined by the mode.

12.4 Register Descriptions

12.4.1 Frequency Control Register (FRQCR)

The frequency control register (FRQCR) is a 16-bit read/write register used to specify whether a clock is output from the CKIO pin, the on/off state of PLL circuit 1, PLL standby, 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. FRQCR is initialized to H'0102 by a power-on reset, but retains its value in a manual reset and in standby mode.

FRQCR:

Bit:	15	14	13	12	11	10	9	8
Bit name:	STC2	IFC2	PFC2	—	—	—	SLPFRQ	CKOEN
Initial value:	0	0	0	0	0	0	0	1
R/W:	R/W	R/W	R/W	R	R	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	PLLEN	PSTBY	STC1	STC0	IFC1	IFC0	PFC1	PFC0
Initial value:	0	0	0	0	0	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 9—Divide Ratio of the External Bus Clock in the Sleep Mode (SLPFRQ): SLPFRQ specifies the divide ratio of the external bus clock in the sleep mode in clock modes 3 to 7. The clock frequency output from the CKIO is not changed. Since the memory refresh cycle is fixed by the bus state controller, the RTSCR register or other setting needs not to be changed.

Bit 9: SLPFRQ	Description
0	External bus clock is not changed in the sleep mode. (Initial value)
1	External bus clock is multiplied by 1/4 in the sleep mode.

Bit 8—Clock Output Enable (CKOEN): Used to output a clock from the CKIO pin or to fix the level of the CKIO pin. Even when the level is fixed, the SH7729 will operate internally at the frequency before the level was fixed. Set this bit to 1 in clock operating modes 0–2. In case of clock operating mode 7, the CKIO pin becomes an input pin irrespective of the value of this bit.

Bit 8: CKOEN	Description
0	Fixes the level of CKIO pin.
1	Outputs a clock from the CKIO pin. (Initial value)

Bit 7—PLL Circuit Enable (PLLEN): Specifies the on/off state of PLL circuit 1. This bit is valid in clock operating modes 3 and 4. PLL circuit 1 goes on when the clock operating mode is 0–2 or 7 irrespective of the value of PLLEN.

Bit 7: PLLEN	Description
0	PLL circuit 1 is not used. (Initial value)
1	PLL circuit 1 is used.

Bit 6—PLL Standby (PSTBY): Specifies PLL standby. When PLL standby is active, PLL circuit 1 will be in standby mode at the frequency specified by the STC bit. This function is valid in clock operating modes 3 and 4.

Bit 6: PSTBY	Description
0	PLL is not in standby mode. (Initial value)
1	PLL is in standby mode.

Bits 15, 5 and 4—Frequency Multiplication Ratio (STC2, STC1, STC0): These bits specify the frequency multiplication ratio of PLL circuit 1.

Bit 15: STC2	Bit 5: STC1	Bit 4: STC0	Description	
0	0	0	× 1	(Initial value)
0	0	1	× 2	
1	0	0	× 3	
0	1	0	× 4	
1	0	1	× 6	
0	1	1	× 8	

Note: Do not set the output frequency of PLL circuit 1 higher than 133 MHz.

Bits 14, 3 and 2—Internal Clock Frequency Division Ratio (IFC2, IFC1, IFC0): These bits specify the frequency division ratio of the internal clock with respect to the output frequency of PLL circuit 1. When PLL circuit 1 is off or in standby mode, set × 1.

Bit 14: IFC2	Bit 3: IFC1	Bit 2: IFC0	Description	
0	0	0	× 1	(Initial value)
0	0	1	× 1/2	
1	0	0	× 1/3	
0	1	0	× 1/4	

Note: Do not set the internal clock frequency lower than the CKIO frequency.

Bits 13, 1 and 0—Peripheral Clock Frequency Division Ratio (PFC2, PFC1, PFC0): These bits specify the division ratio of the peripheral clock frequency with respect to the frequency of the output frequency of PLL circuit 1 or the frequency of the CKIO pin.

Bit 13: PFC2	Bit 1: PFC1	Bit 0: PFC0	Description
0	0	0	× 1
0	0	1	× 1/2
1	0	0	× 1/3
0	1	0	× 1/4
1	0	1	× 1/6 (Initial value)

Note: Do not set the peripheral clock frequency higher than the frequency of the CKIO pin.

12.5 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 dividers 1 and 2. All of these are controlled by software through the frequency control register. The methods are described below. In modes 3, 4 the frequency can also be changed by turning PLL circuit 1 on and off, as described in section 12.6, PLL Standby Function.

12.5.1 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–CKS0 bits: Division ratio of WDT count clock
   WTCNT counter: Initial counter value
3. Set the desired value in the STC2, STC1 and STC0 bits. The division ratio can also be set in the IFC2–IFC0 bits and PFC2–PFC0 bits.
4. The processor pauses internally and the WDT starts incrementing. In clock modes 0–2 and 7, the internal and peripheral clocks both stop. In clock modes 3 and 4, only the internal clock stops. The clock will continue to be output at the CKIO pin as long as the CKOEN bit in the FRQCR register is set to 1.
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.

12.5.2 Changing the Division Ratio

The WDT will not count unless the multiplication rate is changed simultaneously.

1. In the initial state, IFC2–IFC0 = 000 and PFC2–PFC0 = 010.
2. Set the IFC2, IFC1, IFC0, PFC2, PFC1, and PFC0 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.

12.6 PLL Standby Function

12.6.1 Overview of the PLL Standby Function

When operating in clock modes 3 and 4, the internal clock can be controlled by turning the PLL1 circuit on and off. A long oscillation settling time is required, however, when the PLL circuit is started up from a complete halt. During this time, processor operation halts. To enable fast on/off switching of the PLL1 circuit, the PLL standby function is provided. This function is controlled by software using the frequency control register. The use of the PLL standby function is described below.

12.6.2 Usage

From Off to On:

1. Initially, PSTBY = 0, PLLEN = 0, and PLL circuit 1 is stopped. The output of PLL circuit 2 is used for divider 1 input.
2. When the multiplication rate of PLL circuit 1 is set in the STC2–STC0 bits and PSTBY is set to 1, PLL circuit 1 begins oscillating at the specified multiplication rate. The input to divider 1 is still the output of PLL circuit 2 at this point.
3. After PLL circuit 1 oscillation has stabilized, the input of divider 1 switches when PLLEN is set to 1 and the oscillation output of PLL circuit 1 is divided and becomes the internal clock. At this time, the division ratio can be changed by changing the settings of IFC2–IFC0 and PFC2–PFC0. For several cycles before and after the clock switches, the internal clock will be stopped, but the peripheral clock and CKIO output do not stop.

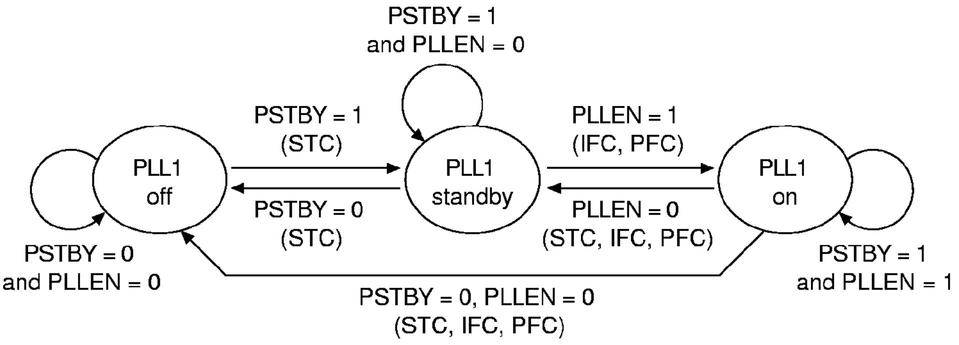
From On to Off:

1. When PLLEN is set to 0, the input of divider 1 switches to the output of PLL circuit 2. At this time, the division ratio can be changed by changing the settings of IFC2–IFC0 and PFC2–PFC0.
2. When PSTBY is set to 0, PLL circuit 1 stops. This setting can be performed simultaneously (and with the same instruction as) the setting in 1 above.

Notes: 1. There are some restrictions on the PLL standby state (PSTBY = 1, PLLEN = 0) as follows: The settings of the frequency control register's CKOEN, STC2–STC0, IFC2–IFC0 and PFC2–PFC0 bits generally cannot be changed. In some cases, however, they can be changed if the PSTBY and PLLEN bit settings are also changed simultaneously (figure 12.2). The SLEEP instruction cannot be executed.

2. It is the responsibility of software to ensure the oscillation settling time. If PLLEN is set to 1 before the oscillation has settled, malfunctions may be caused by an unstable clock.

3. In clock modes 3 and 4, this LSI cannot go to standby mode while PLL circuit 1 is on. Always set PSTBY and PLLEN to 0 to stop PLL circuit 1 before going to standby mode.
4. When PSTBY and PLLEN are both changed from 0 to 1 together, the WDT will automatically start counting and the clock will switch when the WDT overflows. See section 12.5, Changing the Frequency, for setting the WDT.



Note: Bits in parentheses can be changed simultaneously.

Figure 12.2 State Transitions for the PLL Standby Function

12.7 Controlling Clock Output

The CKOEN bit in the FRQCR register can be used to switch between outputting a clock to the CKIO pin or having the level fixed.

12.7.1 Clock Modes 0–2

The CKIO pin level cannot be fixed. Always set the CKOEN bit in FRQCR to 1 (clock output).

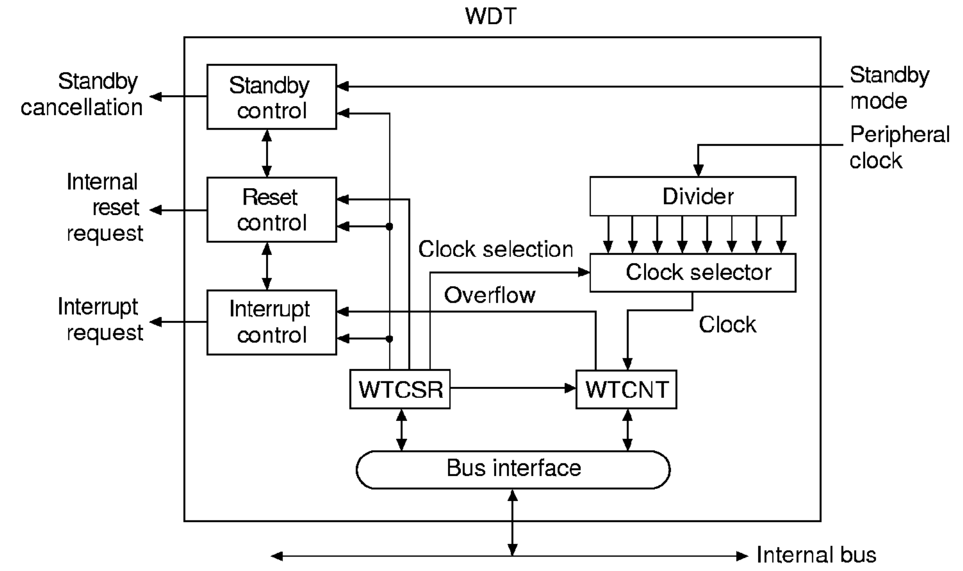
12.7.2 Clock Modes 3, 4

The CKIO output changes as soon as the CKOEN bit is changed. When the WDT is started by simultaneously changing the multiplication rate of PLL circuit 1 or switching PLL circuit 1 on or off, the WDT starts running after the CKIO output is switched, and then the internal clock changes.

12.8 Overview of the WDT

12.8.1 Block Diagram of the WDT

Figurc 12.3 shows a block diagram of thc WDT.



WTCSR: Watchdog timer control/status register
WTCNT: Watchdog timer counter

Figure 12.3 Block Diagram of the WDT

12.8.2 Register Configurations

The WDT has two registers that select the clock, switch the timer mode, and perform other functions. Table 12.5 shows the WDT register.

Table 12.5Register Configuration

Name	Abbreviation	R/W	Access Size	Initial Value	Address
Watchdog timer counter	WTCNT	R/W*	R: byte; W: word*	H'00	H'FFFFFF84
Watchdog timer control/status register	WTCSR	R/W*	R: byte; W: word*	H'00	H'FFFFFF86

Note: Write with a word access. Write H'5A and H'A5, respectively, in the upper bytes. Byte or longword writes are not possible. Read with a byte access.

12.9 WDT Registers

12.9.1 Watchdog Timer Counter (WTCNT)

The watchdog timer counter (WTCNT) is an 8-bit read/write counter that increments on the selected clock. When an overflow occurs, it generates a reset in watchdog timer mode and an interrupt in interval time mode. Its address is H'FFFFFF84. The WTCNT counter is initialized to H'00 only by a power-on reset through the RESET pin. Use a word access to write to the WTCNT counter, with H'5A in the upper byte. Use a byte access to read WTCNT.

Bit:	7	6	5	4	3	2	1	0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

12.9.2 Watchdog Timer Control/Status Register (WTCSR)

The watchdog timer control/status register (WTCSR) is an 8-bit read/write register composed of bits to select the clock used for the count, bits to select the timer mode, and overflow flags. Its address is H'FFFFFF86. The WTCSR register is initialized to H'00 only by a power-on reset through the RESET pin. When a WDT overflow causes an internal reset, the WTCSR retains its value. 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, with H'A5 in the upper byte. Use a byte access to read WTCSR.

Bit:	7	6	5	4	3	2	1	0
	TME	WT/$\overline{\text{IT}}$	RSTS	WOVF	IOVF	CKS2	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7—Timer Enable (TME): Starts and stops timer operation. Clear this bit to 0 when using the WDT in standby mode or when changing the clock frequency.

Bit 7: TME	Description
0	Timer disabled: Count-up stops and WTCNT value is retained (Initial value)
1	Timer enabled

Bit 6—Timer Mode Select (WT/$\overline{\text{IT}}$): Selects whether to use the WDT as a watchdog timer or an interval timer.

Bit 6: WT/$\overline{\text{IT}}$	Description
0	Use as interval timer (Initial value)
1	Use as watchdog timer

Note: If WT/$\overline{\text{IT}}$ is modified when the WDT is running, the up-count may not be performed correctly.

Bit 5—Reset Select (RSTS): Selects the type of reset when the WTCNT overflows in watchdog timer mode. In interval timer mode, this setting is ignored.

Bit 5: RSTS	Description
0	Power-on reset (Initial value)
1	Manual reset

Bit 4—Watchdog Timer Overflow (WOVF): Indicates that the WTCNT has overflowed in watchdog timer mode. This bit is not set in interval timer mode.

Bit 4: WOVF	Description
0	No overflow (Initial value)
1	WTCNT has overflowed in watchdog timer mode

Bit 3—Interval Timer Overflow (IOVF): Indicates that the WTCNT has overflowed in interval timer mode. This bit is not set in watchdog timer mode.

Bit 3: IOVF	Description
0	No overflow (Initial value)
1	WTCNT has overflowed in interval timer mode

Bits 2 to 0—Clock Select 2–0 (CKS2–CKS0): 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 in the table is the value when the peripheral clock (Pϕ) is 15 MHz.

Bit 2: CKS2	Bit 1: CKS1	Bit 0: CKS0	Clock Division Ratio	Overflow Period (when Pϕ = 15 MHz)
0	0	0	1 (Initial value)	17 μs
		1	1/4	68 μs
	1	0	1/16	273 μs
		1	1/32	546 μs
1	0	0	1/64	1.09 ms
		1	1/256	4.36 ms
	1	0	1/1024	17.46 ms
		1	1/4096	69.84 ms

Note: If bits CKS2–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.

12.9.3 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 procedure for 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 12.4. 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.

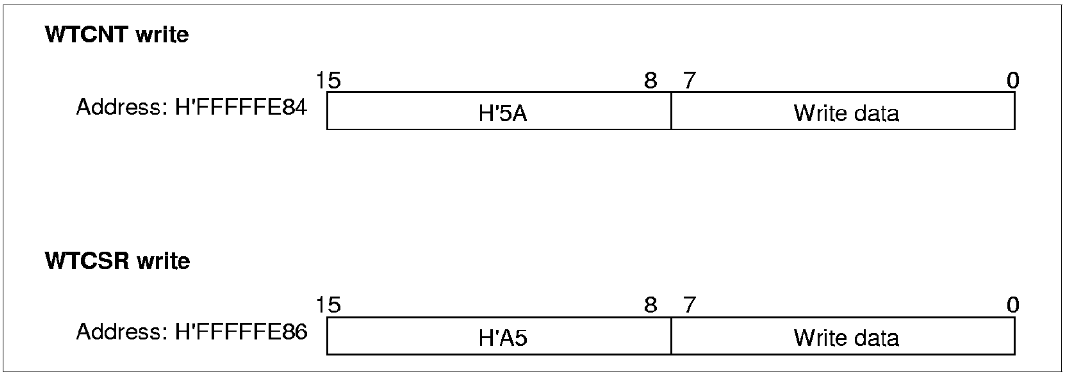


Figure 12.4 Writing to WTCNT and WTCSR

12.10 Using the WDT

12.10.1 Canceling Standbys

The WDT can be used to cancel standby mode with an NMI or other interrupts. The procedure is described below. (The WDT does not run when resets are used for canceling, so keep the RESET pin low until the clock stabilizes.)

1. Before transitioning to 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–CKS0 bits in 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.
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 or detecting interrupts.
5. When the WDT count overflows, the CPG starts supplying the clock and the processor resumes operation. The WOVF flag in WTCSR is 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 SH7729 again enters the standby mode when the WDT has counted up to H'80. This standby mode can be canceled by power-on resets.

12.10.2 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–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.
3. When the frequency control register (FRQCR) is written, the clock stops and the processor enters standby mode temporarily. The WDT starts counting.
4. When the WDT count overflows, the CPG resumes supplying the clock and the processor resumes operation. The WOVF flag in WTCSR is not set when this happens.
5. The counter stops at the values H'00–H'01. The stop value depends on the clock ratio.

12.10.3 Using Watchdog Timer Mode

1. Set the WT/$\overline{\text{IT}}$ bit in the WTCSR register 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 counter.
2. Set the TME bit in WTCSR to 1 to start the count in watchdog timer mode.
3. While operating in watchdog timer mode, rewrite the counter periodically to H'00 to prevent the counter from overflowing.
4. 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.

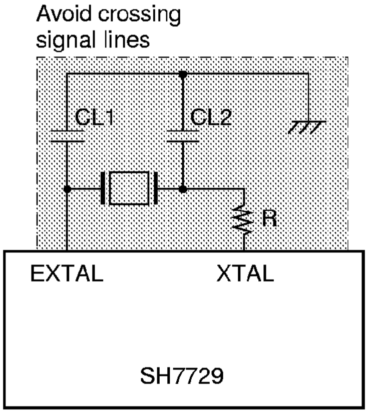
12.10.4 Using Interval Timer Mode

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 WT/$\overline{\text{IT}}$ bit in the WTCSR register to 0, set the type of count clock in the CKS2–CKS0 bits, and set the initial value of the counter in the WTCNT counter.
2. Set the TME bit in WTCSR to 1 to start the count in interval timer mode.
3. When the counter overflows, the WDT sets the IOVF flag in WTCSR to 1 and an interval timer interrupt request is sent to INTC. The counter then resumes counting.

12.11 Notes on Board Design

When Using an External Crystal Resonator: Place the crystal resonator, capacitors CL1 and CL2, and damping resistor R close to the EXTAL and XTAL pins. To prevent induction from interfering with correct oscillation, use a common grounding point for the capacitors connected to the resonator, and do not locate a wiring pattern near these components.



Note: The values for CL1, CL2, and the damping resistance should be determined after consultation with the crystal manufacturer.

Figure 12.5 Points for Attention when Using Crystal Resonator

Decoupling Capacitors: As far as possible, insert a laminated ceramic capacitor of 0.01 to 0.1 µF as a passive capacitor for each V_{SS}/V_{CC} pair. Mount the passive capacitors as close as possible to the SH3 power supply pins, and use components with a frequency characteristic suitable for the SH3 operating frequency, as well as a suitable capacitance value.

Digital system V_{SS}/V_{CC} pairs: 19-21, 27-29, 33-35, 45-47, 57-59, 69-71, 79-81, 83-85, 95-97, 109-111, 132-134, 153-154, 161-163, 173-175, 181-183, 205-208

On-chip oscillator V_{SS}/V_{CC} pairs: 3-6, 145-147, 148-150

When Using a PLL Oscillator Circuit: Keep the wiring from the PLL V_{CC} and V_{SS} connection pattern to the power supply pins short, and make the pattern width large, to minimize the inductance component. Ground the oscillation stabilization capacitors C1 and C2 to V_{SS} (PLL1) and V_{SS} (PLL2), respectively. Place C1 and C2 close to the CAP1 and CAP2 pins and do not locate a wiring pattern in the vicinity. In clock mode 7, connect the EXTAL pin to V_{CC} or V_{SS} and leave the XTAL pin open.

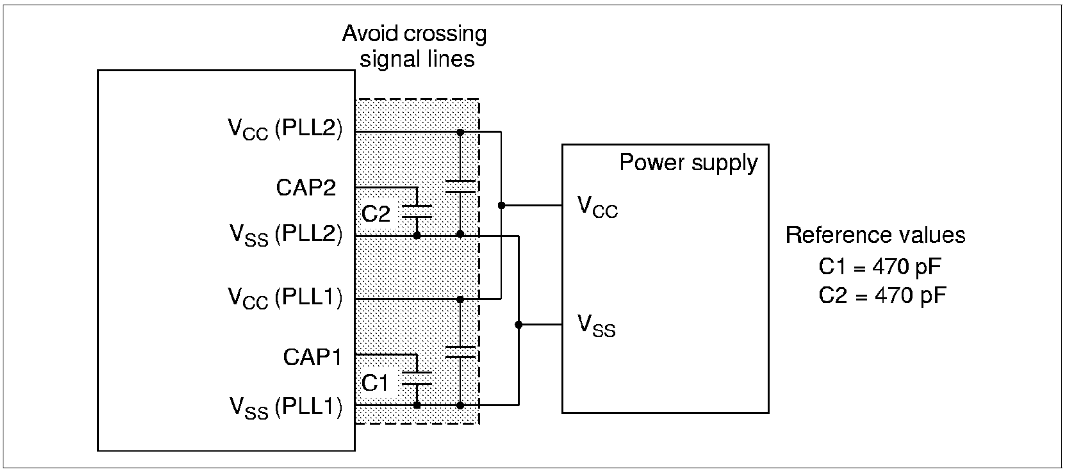


Figure 12.6 Points for Attention when Using PLL Oscillator Circuit

Section 13 Bus State Controller (BSC)

13.1 Overview

The bus state controller (BSC) divides physical address space and output control signals for various types of memory and bus interface specifications. BSC functions enable this LSI to link directly with DRAM, SDRAM, SRAM, ROM, and other memory storage devices without an external circuit. The BSC also allows direct connection to PCMCIA interfaces, simplifying system design and allowing high-speed data transfers in a compact system.

13.1.1 Features

The BSC has the following features:

- Physical address space is divided into six areas
  - A maximum 64 Mbytes for each of the six areas, 0, 2–6
  - Area bus width can be selected by register (area 0 is set by external pin)
  - Wait states can be inserted using the $\overline{\text{WAIT}}$ pin
  - Wait state insertion can be controlled through software. Register settings can be used to specify the insertion of 1–10 cycles independently for each area (1–38 cycles for areas 5 and 6 and the PCMCIAT interface only)
  - The type of memory connected can be specified for each area, and control signals are output for direct memory connection
  - Wait cycles are automatically inserted to avoid data bus conflict for continuous memory accesses to different areas or writes directly following reads of the same area

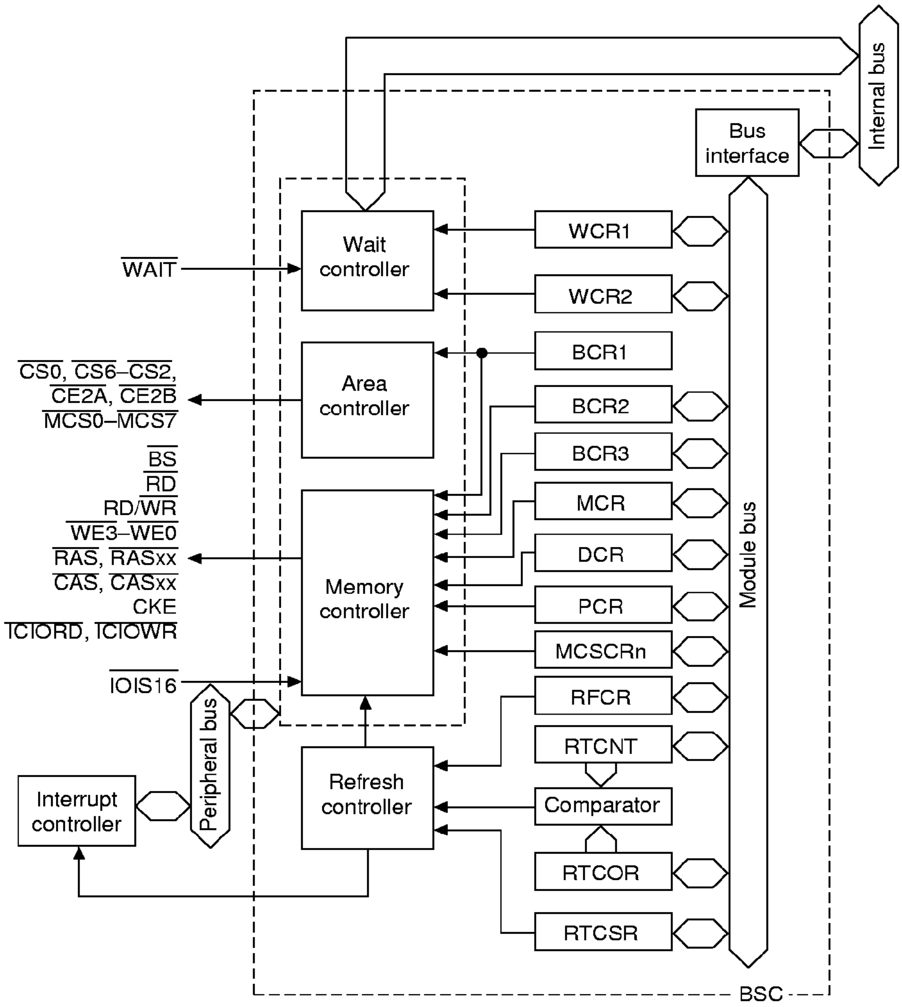
- Direct interface to DRAM
  - Multiplexes row/column addresses according to DRAM capacity
  - Supports burst operation (high-speed page mode, EDO mode, and short-pitch access)
  - Supports CAS-before-RAS refresh and self-refresh
  - Controls timing of DRAM direct-connection control signals according to register settings

- Direct interface to SDRAM
  - Multiplexes row/column addresses according to SDRAM capacity
  - Supports burst operation
  - Supports bank active mode
  - Has both auto-refresh and self-refresh functions
  - Controls timing of SDRAM direct-connection control signals according to register setting

- ROM burst interface
  - Insertion of wait states controllable through software
  - Register setting control of burst transfers
- PCMCIA direct-connection interface
  - Insertion of wait states controllable through software
  - Bus sizing function for I/O bus width (only in the little endian mode)
- Refresh function
  - Refresh cycles will be automatically maintained in the sleep mode even after the external bus frequency is reduced to 1/4 of its normal operating frequency
- Short refresh cycle control
  - The overflow interrupt function of the refresh counter enables the refresh function immediately after the self-refresh operation using the low power-consumption DRAM
- The refresh counter can be used as an interval timer
  - Outputs an interrupt request signal using the compare-matching function
  - Outputs an interrupt request signal when the refresh counter overflows
- Automatically disables the output of clock signals to anywhere but the refresh counter, except during execution of external bus cycles

13.1.2 Block Diagram

Figure 13.1 shows the functional block diagram of the bus state controller.



WCR:	Wait state control register	RFCR:	Refresh count register
BCR:	Bus control register	RTCNT:	Refresh timer count register
MCR:	Memory control register	RTCOR:	Refresh time constant register
DCR:	DRAM control register	RTCSR:	Refresh timer control/status register
PCR:	PCMCIA control register	MCSCRn:	MCSn control register (n:0-7)

Figure 13.1 BSC Functional Block Diagram

13.1.3 Pin Configuration

Table 13.1 lists the BSC pin configuration.

Table 13.1Pin Configuration (Preliminary)

Pin Name	Signal	I/O	Description
Address bus	A25–A0	O	Address output
Data bus	D15–D0	I/O	Data I/O
	D31–D16	I/O	When 32-bit bus width, data I/O
Bus cycle start	$\overline{\text{BS}}$	O	Shows start of bus cycle. During burst transfers, asserts every data cycle.
Chip select 0, 2–4	$\overline{\text{CS0}}$, $\overline{\text{CS2}}$–$\overline{\text{CS4}}$	O	Chip select signal to indicate area being accessed.
Chip select 5, 6	$\overline{\text{CS5}}$/$\overline{\text{CE1A}}$, $\overline{\text{CS6}}$/$\overline{\text{CE1B}}$	O	Chip select signal to indicate area being accessed. CS5/CE1A and CS6/CE1B can also be used as CE1A and CE1B of PCMCIA.
Read/write	RDWR	O	Data bus direction indicator signal. DRAM/PCMCIA write indicator signal.
Row address strobe 3L	$\overline{\text{RAS3L}}$	O	When DRAM or SDRAM is used in area 3, RAS3L for lower 32-Mbyte address.
Row address strobe 3U	$\overline{\text{RAS3U}}$	O	When DRAM or SDRAM is used in area 3, RAS3U for upper 32-Mbyte address.
Column address strobe	$\overline{\text{CASLL}}$/$\overline{\text{CASL}}$	O	When DRAM is used, CASLL signal for D7–D0. When SDRAM is used, CASL signal for lower 32-Mbyte address.
Column address strobe LH	$\overline{\text{CASLH}}$/$\overline{\text{CASU}}$	O	When DRAM is used, CASLH signal for D15–D8. When SDRAM is used, CASU signal for upper 32-Mbyte address.
Column address strobe HL	$\overline{\text{CASHL}}$	O	When DRAM is used, CASHL signal for D23–D16.
Column address strobe HH	$\overline{\text{CASHH}}$	O	When DRAM is used, CASHH signal for D31–D24.
Column address strobe 2L	$\overline{\text{CAS2L}}$	O	When the area 2 DRAM is used, CAS2L signal for D7–D0.
Column address strobe 2H	$\overline{\text{CAS2H}}$	O	When the area 2 DRAM is used, CAS2H signal for D15–D8.
Mask ROM chip select	$\overline{\text{MCS0}}$–$\overline{\text{MCS7}}$	O	Chip select signal for mask ROM connected to area 0 or 2.

Table 13.1Pin Configuration (Preliminary) (cont)

Pin Name	Signal	I/O	Description
Data enable 0	$\overline{\text{WE0}}$/DQMLL	O	When memory other than SDRAM is used, selects D7–D0 write strobe signal. When SDRAM is used, selects D7–D0.
Data enable 1	$\overline{\text{WE1}}$/DQMLU/ $\overline{\text{WE}}$	O	When memory other than SDRAM and PCMCIA is used, selects D15–D8 write strobe signal. When SDRAM is used, selects D15–D8. When PCMCIA is used, strobe signal that indicates the write cycle.
Data enable 2	$\overline{\text{WE2}}$/DQMUL/ $\overline{\text{ICIORD}}$	O	When memory other than SDRAM and PCMCIA is used, selects D23–D16 write strobe signal. When SDRAM is used, selects D23–D16. When PCMCIA is used, strobe signal indicating I/O read.
Data enable 3	$\overline{\text{WE3}}$/DQMUU/ $\overline{\text{ICIOWR}}$	O	When memory other than SDRAM and PCMCIA is used, selects D31–D24 write strobe signal. When SDRAM is used, selects D31–D24. When PCMCIA is used, strobe signal indicating I/O write.
Read	$\overline{\text{RD}}$	O	Strobe signal indicating read cycle
Wait	$\overline{\text{WAIT}}$	I	Wait state request signal
IOIS16	$\overline{\text{IOIS16}}$	I	Signal indicating PCMCIA 16-bit I/O. Valid only in little-endian mode.
Clock enable	CKE	O	Clock enable control signal of SDRAM
Bus release request	$\overline{\text{BREQ}}$	I	Bus release request signal
Bus release acknowledgment	$\overline{\text{BACK}}$	O	Bus release acknowledge signal
PCMCIA card select	$\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$	O	When PCMCIA is used, CE2A and CE2B
Row address strobe 2L	$\overline{\text{RAS2L}}$	O	When DRAM is used in area 2, RAS2L signal for lower 32-Mbyte address
Row address strobe 2U	$\overline{\text{RAS2U}}$	O	When DRAM is used in area 2, RAS2U signal for upper 32-Mbyte address

13.1.4 Register Configuration

The BSC has 11 registers (table 13.2). The SDRAM also has a built-in SDRAM mode register. These registers control direct connection interfaces to memory, wait states, refreshes, and PCMCIA devices.

Table 13.2Register Configuration

Name	Abbr.	R/W	Initial Value*	Address	Bus Width
Bus control register 1	BCR1	R/W	H'0000	H'FFFFFF60	16
Bus control register 2	BCR2	R/W	H'3FF0	H'FFFFFF62	16
Wait state control register 1	WCR1	R/W	H'3FF3	H'FFFFFF64	16
Wait state control register 2	WCR2	R/W	H'FFFF	H'FFFFFF66	16
Individual memory control register	MCR	R/W	H'0000	H'FFFFFF68	16
DRAM control register	DCR	R/W	H'0000	H'FFFFFF6A	16
PCMCIA control register	PCR	R/W	H'0000	H'FFFFFF6C	16
Refresh timer control/status register	RTCSR	R/W	H'0000	H'FFFFFF6E	16
Refresh timer counter	RTCNT	R/W	H'0000	H'FFFFFF70	16
Refresh time constant register	RTCOR	R/W	H'0000	H'FFFFFF72	16
Refresh count register	RFCR	R/W	H'0000	H'FFFFFF74	16
Bus control register 3	BCR3	R/W	H'0000	H'FFFFFF7E	16
SDRAM mode register, area 2	SDMR	W	—	H'FFFFD000– H'FFFFDFFF	8
SDRAM mode register, area 3				H'FFFFE000– H'FFFFEFFF	
MCS0 control register	MCSCR0	R/W	H'0000	H'FFFFFF50	16
MCS1 control register	MCSCR1	R/W	H'0000	H'FFFFFF52	16
MCS2 control register	MCSCR2	R/W	H'0000	H'FFFFFF54	16
MCS3 control register	MCSCR3	R/W	H'0000	H'FFFFFF56	16
MCS4 control register	MCSCR4	R/W	H'0000	H'FFFFFF58	16
MCS5 control register	MCSCR5	R/W	H'0000	H'FFFFFF5A	16
MCS6 control register	MCSCR6	R/W	H'0000	H'FFFFFF5C	16
MCS7 control register	MCSCR7	R/W	H'0000	H'FFFFFF5E	16

Notes: 1. Initialized by power-on resets.

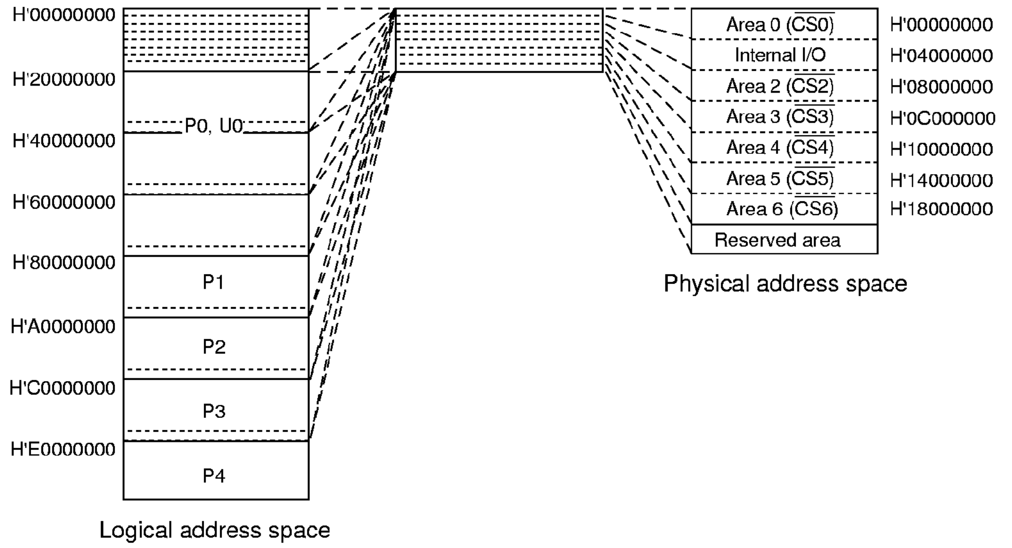
2. For details, see section 13.2.9, SDRAM Mode Register.

13.1.5 Arca Ovcrvicw

Space Allocation: In the architecture of this LSI, both logical spaces and physical spaces have 32-bit address spaces. The logical space is divided into five areas by the value of the upper bits of the address. The physical space is divided into eight areas.

Logical space can be allocated at physical spaces using a memory management unit (MMU). For dctails, rcfcr to scction 3, Mcmory Managcmcnt Unit, which dcscribcs arca allocation for physical spaces.

As listed in table 13.3, this LSI can be connected directly to seven areas of memory/PCMCIA interface, and it outputs chip select signals ($\overline{\text{CS0}}$, $\overline{\text{CS2}}$–$\overline{\text{CS6}}$, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$) for each of them. $\overline{\text{CS0}}$ is asserted during area 0 access; $\overline{\text{CS6}}$ is asserted during area 6 access. When DRAM is connected to area 2 or 3, signals such as $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and RD/$\overline{\text{WR}}$ are also asserted. When PCMCIA interface is selected in area 5 or 6, in addition to $\overline{\text{CS5}}$/$\overline{\text{CS6}}$, $\overline{\text{CE2A}}$/$\overline{\text{CE2B}}$ are asserted for the corresponding bytes accessed.



Note: For logical address spaces P0 and P3, when the memory management unit (MMU) is on, it can optionally generate a physical address for the logical address. It can be applied when the MMU is off and when the MMU is on and each physical address for the logical address is equal except for upper three bits.

Figure 13.2 Corresponding to Logical Address Space and Physical Address Space

Table 13.3Physical Address Space Map

Area	Physical Address	Connectable Memory	Capacity	Access Size
0	H'00000000 to H'03FFFFFF	Ordinary memory*1, burst ROM	64 Mbytes	8, 16, 32*2
	H'00000000 + H'20000000 × n to H'03FFFFFF + H'20000000 × n		Shadow	n: 1–6
1	H'04000000 to H'07FFFFFF	Internal I/O registers*8	64 Mbytes	8, 16, 32*3
	H'04000000 + H'20000000 × n to H'07FFFFFF + H'20000000 × n		Shadow	n: 1–6
2	H'08000000 to H'0BFFFFFF	Ordinary memory*1, SDRAM, DRAM	64 Mbytes	8, 16, 32*3, *4
	H'08000000 + H'20000000 × n to H'0BFFFFFF + H'20000000 × n		Shadow	n: 1–6
3	H'0C000000 to H'0FFFFFFF	Ordinary memory, SDRAM, DRAM	64 Mbytes	8, 16, 32*3, *5
	H'0C000000 + H'20000000 × n to H'0FFFFFFF + H'20000000 × n		Shadow	n: 1–6
4	H'10000000 to H'13FFFFFF	Ordinary memory	64 Mbytes	8, 16, 32*3
	H'10000000 + H'20000000 × n to H'13FFFFFF + H'20000000 × n		Shadow	n: 1–6
5	H'14000000 to H'15FFFFFF	Ordinary memory, PCMCIA, burst ROM	32 Mbytes	8, 16, 32 *3, *6
	H'16000000 to H'17FFFFFF	Ordinary memory, burst ROM	32 Mbytes	
	H'14000000 + H'20000000 × n to H'17FFFFFF + H'20000000 × n		Shadow	n: 1–6
6	H'18000000 to H'19FFFFFF	Ordinary memory, PCMCIA, burst ROM	32 Mbytes	8, 16, 32 *3, *6
	H'1A000000 to H'1BFFFFFF			
	H'18000000 + H'20000000 × n to H'1BFFFFFF + H'20000000 × n		Shadow	n: 1–6
7*7	H'1C000000 + H'20000000 × n to H'1FFFFFFF + H'20000000 × n	Reserved area		n: 0–7

Notes: 1. Memory with interface such as SRAM or ROM.

2. Use external pin to specify memory bus width.

3. Use register to specify memory bus width.

4. With SDRAM interfaces, bus width must be 16 or 32 bits. With DRAM interfaces, bus width must be 16 bits.

5. With SDRAM interfaces, bus width must be 16 or 32 bits. With DRAM interfaces, bus width must be 16 or 32 bits.

6. With PCMCIA interface, bus width must be 8 or 16 bits.

7. Do not access the reserved area. If the reserved area is accessed, the correct operation cannot be guaranteed.

8. When the control register in area 1 is not used for address translation by the MMU, set the top three bits of the logical address to 101 to allocate in the P2 space.

Area 0: H'00000000	Ordinary memory/ burst ROM	
Area 1: H'04000000	Internal I/O	
Area 2: H'08000000	Ordinary memory/ SDRAM, DRAM	Only DRAM with a 16-bit bus can be connected to area 2
Area 3: H'0C000000	Ordinary memory/ SDRAM, DRAM	
Area 4: H'10000000	Ordinary memory	
Area 5: H'14000000	Ordinary memory/ burst ROM/PCMCIA	The PCMCIA interface is for the memory card only
Area 6: H'18000000	Ordinary memory/ burst ROM/PCMCIA	The PCMCIA interface is shared by the memory and I/O card

Figure 13.3 Physical Space Allocation

Memory Bus Width: The memory bus width in this LSI can be set for each area. In area 0, an external pin can be used to select byte (8 bits), word (16 bits), or longword (32 bits) on power-on reset. The correspondence between the external pins (MD4 and MD3) and memory size is listed in table below.

Table 13.4Correspondence between External Pins (MD4 and MD3) and Memory Size

MD4	MD3	Memory Size
0	0	Reserved (Do not set)
0	1	8 bits
1	0	16 bits
1	1	32 bits

For areas 2–6, byte, word, and longword may be chosen for the bus width using bus control register 2 (BCR2) whenever ordinary memory, ROM, or burst ROM are used. When the DRAM or SDRAM interface is used, word or longword can be chosen as the bus width.

When area 2 is used as a DRAM area, set the bus width of area 2 to word. When the PCMCIA interface is used, set the bus width to byte or word. When SDRAM is connected to both area 2 and area 3, set the same bus width for areas 2 and 3. When using the port function, set each of the bus widths to byte or word for all areas. For more information, see section 13.2.2, Bus Control Register 2 (BCR2).

Shadow Space: Areas 0, 2–6 are decoded by physical addresses A28–A26, which correspond to areas 000 to 110. Address bits 31–29 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 = 1–6). The address range for area 7, which is on-chip I/O space, is H'1C000000 to H'1FFFFFFF. The address space H'1C000000 + H'20000000 × n–H'1FFFFFFF + H'20000000 × n (n = 0–7) corresponding to the area 7 shadow space is reserved, so do not use it.

13.1.6 PCMCIA Support

This LSI supports PCMCIA standard interface specifications in physical space areas 5 and 6.

Table 13.5PCMCIA Interface Characteristics

Item	Feature
Access	Random access
Data bus	8/16 bits
Memory type	Mask ROM, OTPROM, EPROM, EEPROM, flash memory, SRAM
Common memory capacity	Maximum 64 Mbytes (Supports full PCMCIA specifications by using a segment bit (an address bit for the PC card))
Attribute memory capacity	Maximum 32 Mbytes
I/O space capacity	Maximum 32 Mbytes
Others	Dynamic bus sizing of I/O bus width* The PCMCIA interface can be accessed from the address translation area or non-address translation area.

Note: Dynamic bus sizing of I/O bus width is supported only in the little endian mode.

Area 5: H'14000000	Commom memory/Attribute memory
Area 5: H'16000000	I/O space
Area 6: H'18000000	Commom memory/Attribute memory
Area 6: H'1A000000	I/O space

Figure 13.4 PCMCIA Space Allocation

Table 13.6PCMCIA Support Interface

	IC Memory Card Interface			I/O Card Interface			
Pin	Signal	I/O	Function	Signal	I/O	Function	SH7729 Pin
1	GND	—	Ground	GND	—	Ground	—
2	D3	I/O	Data	D3	I/O	Data	D3
3	D4	I/O	Data	D4	I/O	Data	D4
4	D5	I/O	Data	D5	I/O	Data	D5
5	D6	I/O	Data	D6	I/O	Data	D6
6	D7	I/O	Data	D7	I/O	Data	D7
7	$\overline{CE1}$	I	Card enable	$\overline{CE1}$	I	Card enable	$\overline{CE1A}$ or $\overline{CE1B}$
8	A10	I	Address	A10	I	Address	A10
9	OE	I	Output enable	$\overline{OE}$	I	Output enable	$\overline{RD}$
10	A11	I	Address	A11	I	Address	A11
11	A9	I	Address	A9	I	Address	A9
12	A8	I	Address	A8	I	Address	A8
13	A13	I	Address	A13	I	Address	A13
14	A14	I	Address	A14	I	Address	A14
15	$\overline{WE}/\overline{PGM}$	I	Write enable	$\overline{WE}/\overline{PGM}$	I	Write enable	$\overline{WE}$
16	RDY/$\overline{BSY}$	O	Ready/Busy	$\overline{IREQ}$	O	Ready/Busy	—
17	V_{CC}		Operation power	V_{CC}		Operation power	—
18	VPP1		Program power	VPP1		Program/ peripheral power	—

Table 13.6PCMCIA Support Interface (cont)

	IC Memory Card Interface			I/O Card Interface			
Pin	**Signal**	**I/O**	**Function**	**Signal**	**I/O**	**Function**	**SH7729 Pin**
19	A16	I	Address	A16	I	Address	A16
20	A15	I	Address	A15	I	Address	A15
21	A12	I	Address	A12	I	Address	A12
22	A7	I	Address	A7	I	Address	A7
23	A6	I	Address	A6	I	Address	A6
24	A5	I	Address	A5	I	Address	A5
25	A4	I	Address	A4	I	Address	A4
26	A3	I	Address	A3	I	Address	A3
27	A2	I	Address	A2	I	Address	A2
28	A1	I	Address	A1	I	Address	A1
29	A0	I	Address	A0	I	Address	A0
30	D0	I/O	Data	D0	I/O	Data	D0
31	D1	I/O	Data	D1	I/O	Data	D1
32	D2	I/O	Data	D2	I/O	Data	D2
33	WP	O	Write protect	$\overline{\text{IOIS16}}$	O	16-bit I/O port	$\overline{\text{IOIS16}}$
34	GND		Ground	GND		Ground	—
35	GND		Ground	GND		Ground	—
36	$\overline{\text{CD1}}$	O	Card detection	$\overline{\text{CD1}}$	O	Card detection	—
37	D11	I/O	Data	D11	I/O	Data	D11
38	D12	I/O	Data	D12	I/O	Data	D12
39	D13	I/O	Data	D13	I/O	Data	D13
40	D14	I/O	Data	D14	I/O	Data	D14
41	D15	I/O	Data	D15	I/O	Data	D15
42	$\overline{\text{CE2}}$	I	Card enable	$\overline{\text{CE2}}$	I	Card enable	$\overline{\text{CE2A}}$ or $\overline{\text{CE2B}}$
43	$\overline{\text{VS1}}$	I	Voltage sense 1	$\overline{\text{VS1}}$	I	Voltage sense 1	—
44	RFU		Reserved	$\overline{\text{IORD}}$	I	I/O read	$\overline{\text{ICIORD}}$
45	RFU		Reserved	$\overline{\text{IOWR}}$	I	I/O write	$\overline{\text{ICIOWR}}$
46	A17	I	Address	A17	I	Address	A17
47	A18	I	Address	A18	I	Address	A18
48	A19	I	Address	A19	I	Address	A19

Table 13.6PCMCIA Support Interface (cont)

	IC Memory Card Interface			I/O Card Interface			
Pin	Signal	I/O	Function	Signal	I/O	Function	SH7729 Pin
49	A20	I	Address	A20	I	Address	A20
50	A21	I	Address	A21	I	Address	A21
51	V_{CC}		Power supply	V_{CC}		Power supply	—
52	VPP2		Program power	VPP2		Program/ peripheral power	—
53	A22	I	Address	A22	I	Address	A22
54	A23	I	Address	A23	I	Address	A23
55	A24	I	Address	A24	I	Address	A24
56	A25	I	Address	A25	I	Address	A25
57	$\overline{\text{VS2}}$	I	Voltage sense 2	$\overline{\text{VS2}}$	I	Voltage sense 2	—
58	RESET	I	Reset	RESET	I	Reset	—
59	$\overline{\text{WAIT}}$	O	Wait request	$\overline{\text{WAIT}}$	O	Wait request	—
60	RFU		Reserved	$\overline{\text{INPACK}}$	O	Input acknowledge	—
61	$\overline{\text{REG}}$	I	Attribute memory space select	$\overline{\text{REG}}$	I	Attribute memory space select	—
62	BVD2	O	Battery voltage detection	$\overline{\text{SPKR}}$	O	Digital voice signal	—
63	BVD1	O	Battery voltage detection	$\overline{\text{STSCHG}}$	O	Card state change	—
64	D8	I/O	Data	D8	I/O	Data	D8
65	D9	I/O	Data	D9	I/O	Data	D9
66	D10	I/O	Data	D10	I/O	Data	D10
67	$\overline{\text{CD2}}$	O	Card detection	$\overline{\text{CD2}}$	O	Card detection	—
68	GND		Ground	GND		Ground	—

13.2 BSC Registers

13.2.1 Bus Control Register 1 (BCR1)

Bus control register 1 (BCR1) is a 16-bit read/write register that sets the functions and bus cycle state for each area. It is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or by standby mode. Do not access external memory outside area 0 until BCR1 register initialization is complete.

Bit:	15	14	13	12	11	10	9	8
Bit name:	PULA	PULD	HIZMEM	HIZCNT	ENDIAN	A0BST1	A0BST0	A5BST1
Initial value:	0	0	0	0	0/1*	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	A5BST0	A6BST1	A6BST0	DRAM TP2	DRAM TP1	DRAM TP0	A5 PCM	A6 PCM
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Samples the value of the external pin (MD5) designating endian at power-on reset.

Bit 15—Pin A25 to A0 Pull-Up (PULA): Specifies whether or not pins A25 to A0 are pulled up for 4 cycles immediately after $\overline{\text{BACK}}$ is asserted.

Bit 15: PULA	Description
0	Not pulled up (Initial value)
1	Pulled up

Bit 14—Pin D31 to D0 Pull-Up (PULD): Specifies whether or not pins D31 to D0 are pulled up when not in use.

Bit 14: PULD	Description
0	Not pulled up (Initial value)
1	Pulled up

Bit 13—Hi-Z memory control (HIZMEM): Specifies the state of A25-0, BS, CS, RD/WR, WE/DQM, RD, CE2A, CE2B and DRAK0/1 in standby mode.

Bit 13: HIZMEM	Description
0	A25-0, BS, CS, RD/WR, WE/DQM, RD, MD3/CE2A, MD4/CE2B and DRAK0/1 are Hi-Z in standby mode. (Initial value)
1	A25-0, BS, CS, RD/WR, WE/DQM, RD, MD3/CE2A, MD4/CE2B and DRAK0/1 are High in standby mode.

Bit 12—High-Z Control (HIZCNT): Specifies the state of the RAS and the CAS signals at standby and bus right release.

Bit 12: HIZCNT	Description
0	The RAS and the CAS signals are high-impedance state (High-Z) at standby and bus right release. (Initial value)
1	The RAS and the CAS signals are driven at standby and bus right release.

Bit 11—Endian Flag (ENDIAN): Samples the value of the external pin designating endian upon a power-on reset. Endian for all physical spaces is decided by this bit, which is read-only.

Bit 11: ENDIAN	Description
0	(On reset) Endian setting external pin (MD5) is low. Indicates the SH7729 is set as big endian.
1	(On reset) Endian setting external pin (MD5) is high. Indicates the SH7729 is set as little endian.

Bits 10, 9—Area 0 Burst ROM Control (A0BST1–A0BST0): Specify whether to use burst ROM in physical space area 0. When burst ROM is used, set the number of burst transfers.

Bit 10: A0BST1	Bit 9: A0BST0	Description
0	0	Access area 0 as ordinary memory (initial value)
	1	Access area 0 as burst ROM (4 consecutive accesses). Can be used when bus width is 8, 16, or 32.
1	0	Access area 0 as burst ROM (8 consecutive accesses). Can be used when bus width is 8 or 16.
	1	Access area 0 as burst ROM (16 consecutive accesses). Can be used only when bus width is 8.

Bits 8, 7—Area 5 Burst Enable (A5BST1–A5BST0): Specify whether to use burst ROM and PCMCIA burst mode in physical space area 5. When burst ROM and PCMCIA burst mode are used, set the number of burst transfers.

Bit 8: A5BST1	Bit 7: A5BST0	Description
0	0	Access area 5 as ordinary memory (initial value)
	1	Burst access of area 5 (4 consecutive accesses). Can be used when bus width is 8, 16, or 32.
1	0	Burst access of area 5 (8 consecutive accesses). Can be used when bus width is 8 or 16.
	1	Burst access of area 5 (16 consecutive accesses). Can be used only when bus width is 8.

Bits 6, 5—Area 6 Burst Enable (A6BST1–A6BST0): Specify whether to use burst ROM and PCMCIA burst mode in physical space area 6. When burst ROM and PCMCIA burst mode are used, set the number of burst transfers.

Bit 6: A6BST1	Bit 5: A6BST0	Description
0	0	Access area 6 as ordinary memory (initial value)
	1	Burst access of area 6 (4 consecutive accesses). Can be used when bus width is 8, 16, or 32.
1	0	Burst access of area 6 (8 consecutive accesses). Can be used when bus width is 8 or 16.
	1	Burst access of area 6 (16 consecutive accesses). Can be used only when bus width is 8.

Bits 4–2—Area 2, Area 3 Memory Type (DRAMTP2, DRAMTP1, DRAMTP0): Designate the types of memory connected to physical space areas 2 and 3. Ordinary memory, such as ROM, SRAM, or flash ROM, can be directly connected. DRAM, and SDRAM can also be directly connected.

Bit 4: DRAMTP2	Bit 3: DRAMTP1	Bit 2: DRAMTP0	Description
0	0	0	Areas 2 and 3 are ordinary memory (Initial value)
		1	Reserved (Setting disabled)
	1	0	Area 2: ordinary memory; area 3: SDRAM
		1	Areas 2 and 3 are SDRAM*[1]
1	0	0	Area 2: ordinary memory; area 3: DRAM
		1	Areas 2 and 3 are DRAM *[2]
	1	0	Reserved (Setting disabled)
		1	Reserved (Setting disabled)

Notes: 1. When selecting this mode, set the same bus width for area 2 and area 3.

2. When selecting this mode, set the area 2 bus width as word.

Bit 1—Area 5 Bus Type (A5PCM): Designates whether to access physical space area 5 as PCMCIA space.

Bit 1: A5PCM	Description
0	Access physical space area 5 as ordinary memory (Initial value)
1	Access physical space area 5 as PCMCIA space

Bit 0—Area 6 Bus Type (A6PCM): Designates whether to access physical space area 6 as PCMCIA space.

Bit 0: A6PCM	Description
0	Access physical space area 6 as ordinary memory (Initial value)
1	Access physical space area 6 as PCMCIA space

13.2.2 Bus Control Register 2 (BCR2)

The bus control register 2 (BCR2) is a 16-bit read/write register that selects the bus-size width of each area. It is initialized to H'3FF0 by a power-on reset, but is not initialized by a manual reset or by standby mode. Do not access external memory outside area 0 until BCR2 register initialization is complete.

Bit:	15	14	13	12	11	10	9	8
Bit name:	—	—	A6SZ1	A6SZ0	A5SZ1	A5SZ0	A4SZ1	A4SZ0
Initial value:	0	0	1	1	1	1	1	1
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	A3SZ1	A3SZ0	A2SZ1	A2SZ0	—	—	—	—
Initial value:	1	1	1	1	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R	R	R

Bits 15, 14, 3, 2, 1, and 0—Reserved: Bits 15, 14, 3, 2, 1, and 0 cannot be modified and always read as 0.

Bits 2n + 1, 2n—Area n (2–6) Bus Size Specification (AnSZ1, AnSZ0): Specify the bus sizes of physical space area n (n = 2 to 6).

Bit 2n + 1: AnSZ1	Bit 2n: AnSZ0	Port A / B	Description
0	0	Unused	Reserved (Setting disabled)
	1		Byte (8-bit) size
1	0		Word (16-bit) size
	1		Longword (32-bit) size
0	0	Used	Reserved (Setting disabled)
	1		Byte (8-bit) size
1	0		Word (16-bit) size
	1		Reserved (Setting disabled)

13.2.3 Bus Control Register 3 (BCR3)

Bus control register 3 (BCR3) is a 16-bit read/write register that specifies RAS and CAS timing for the DRAM (areas 2 and 3). This enables a large amount of data to be transferred efficiently, for example, when transferring image data. The BCR3 is initialized to H'0000 by power-on resets, but is not initialized by manual resets or in the standby mode. The bits EXTEND, TPC31–30, RCD31–30, TRAS31–30, TPC21–20, RCD21–20, and TRAS21–20 are written to during the initialization after a power-on reset and are not modified again.

Bit:	15	14	13	12	11	10	9	8
Bit name:	EXTEND	—	TPC31	TPC30	RCD31	RCD30	TRAS31	TRAS30
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	TPC21	TPC20	RCD21	RCD20	TRAS21	TRAS20
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit 15—BSC DRAM Access Mode Extended Control (EXTEND): The EXTEND bit specifies the short pitch access mode to the DRAM (areas 2 and 3). Setting the EXTEND bit to 1 enables a single access to be performed in a minimum of two cycles (RCDn (1–0) = 00) to the DRAM, and a burst access to be performed in a pitch of one cycle for the second and later data. In addition, by setting the EXTEND bit to 1, other bits of the BCR3 register (TPC31–30, SCD31–30, TRAS31–30, TPC21–20, RCD21–20, and TRAS21–20 become valid.

Bit 15: EXTEND	Description
0	The BSC is set to normal mode. (Initial value)
1	The BSC is set to extended mode. DRAM is accessed in a short pitch.

Bits 13 and 12—RAS Precharge Time (TPC31, TPC30): When the EXTEND bit in BCR3 is set to 1, the BSC uses the TPC (31-30) bits. When the DRAM interface is selected as the connected memory of area 3, the TPC bits set the minimum number of cycles for RAS precharge until the next RAS assertion after RAS negation. Set the same value for TPC31–30 and TPC21–20 in BCR3, TPC1–0 in the MCR, and TPC1–0 in the DCR.

Bit 13	Bit 12	
TPC31	**TPC30**	**Function**
0	0	1 cycle (Initial value)
0	1	2 cycles
1	0	3 cycles
1	1	4 cycles

Bits 11 and 10—RAS-CAS Delay (RCD31, RCD30): When the EXTEND bit in BCR3 is set to 1, the BSC uses the RCD (31-30) bits in place of the RCD (1-0) bits in the MCR. When the DRAM interface is selected as the connected memory of area 3, these bits set the RAS-CAS assert delay.

Bit 11	Bit 10	
RCD31	**RCD30**	**Function**
0	0	1 cycle (Initial value)
0	1	2 cycles
1	0	3 cycles
1	1	4 cycles

Bits 9 and 8—CAS-Before-RAS Refresh RAS Assert Time (TRAS31, TRAS30): When the EXTEND bit in BCR3 is set to 1, the BSC uses the TRAS (31–30) bits. When the DRAM interface is selected as the connected memory of area 3, these bits set the RAS assert period for CAS-before-RAS refreshes. Set the same value for TRAS31–30 and TRAS21–20 in BCR3, TRAS 1–0 in the MCR, and TRAS 1–0 in the DCR.

Bit 9	Bit 8	
TRAS31	**TRAS30**	**Function**
0	0	2 cycles (Initial value)
0	1	3 cycles
1	0	4 cycles
1	1	5 cycles

Bits 5 and 4—RAS Precharge Time (TPC21, TPC20): When the EXTEND bit in BCR3 is set to 1, the BSC uses the TPC (21–20) bits. When the DRAM interface is selected as the connected memory of area 2, the TPC bits set the minimum number of cycles for RAS precharge until the next RAS assertion after RAS negation.

Bit 5	Bit 4	Function	
TPC21	TPC20	Normal	After self refresh
0	0	1 cycle (Initial value)	2 cycles (Initial value)
0	1	2 cycles	5 cycles
1	0	3 cycles	8 cycles
1	1	4 cycles	11 cycles

Bits 3 and 2—RAS-CAS Delay (RCD21, RCD20): When the EXTEND bit in BCR3 is set to 1, the BSC uses the RCD (21–20) bits in place of the RCD (1–0) bits in the DCR. The RAS-CAS delay is set for the DRAM interface connected to area 2.

Bit 3	Bit 2	
RCD21	RCD20	Function
0	0	1 cycle (Initial value)
0	1	2 cycles
1	0	3 cycles
1	1	4 cycles

Bits 1 and 0—CAS-Before-RAS Refresh RAS Assert Time (TRAS21, TRAS20): When the EXTEND bit in BCR3 is set to 1, the BSC uses the TRAS (21–20) bits in place of the TRAS (1–0) bits in the DCR. The RAS assert time is set for the DRAM interface connected to area 2 when CAS-before-RAS refreshes.

Bit 1	Bit 0	
TRAS21	TRAS20	Function
0	0	2 cycles (Initial value)
0	1	3 cycles
1	0	4 cycles
1	1	5 cycles

Bits 14, 7 and 6—Reserved: Always read 0. In addition, 0 is always written to these bits.

13.2.4 Wait State Control Register 1 (WCR1)

Wait state control register 1 (WCR1) is a 16-bit read/write register that specifies the number of idle (wait) state cycles inserted for each area. For some memories, the drive of the data bus may not be turned off quickly even when the read signal from the external device is turned off. This can result in conflicts between data buses when consecutive memory accesses are to different memories or when a write immediately follows a memory read. This LSI automatically inserts idle states equal to the number set in WCR1 in those cases.

WCR1 is initialized to H'3FF3 by a power-on reset. It is not initialized by a manual reset or by standby mode.

Bit:	15	14	13	12	11	10	9	8
Bit name:	WAITSEL	—	A6IW1	A6IW0	A5IW1	A5IW0	A4IW1	A4IW0
Initial value:	0	0	1	1	1	1	1	1
R/W:	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	A3IW1	A3IW0	A2IW1	A2IW0	—	—	A0IW1	A0IW0
Initial value:	1	1	1	1	0	0	1	1
R/W:	R/W	R/W	R/W	R/W	R	R	R/W	R/W

Bit 15—WAIT Sampling Timing Select (WAITSEL): Specifies the WAIT signal sampling timing.

Bit 15: WAITSEL	Description	
0	Sampled at rise of CKIO	(Initial value)
1	Sampled at fall of CKIO	

Bits 14, 3, 2 —Reserved: Bits 15, 14, 3 and 2 cannot be modified and always read as 0.

Bits 2n + 1, 2n—Area n (6–2, 0) Intercycle Idle Specification (AnIW1, AnIW0): Specify the number of idles inserted between bus cycles when switching between physical space area n (6–2, 0) to another space or between a read access to a write access in the same physical space.

Bit 2n + 1: AnIW1	Bit 2n: AnIW0	Description
0	0	1 idle cycle inserted
	1	1 idle cycle inserted
1	0	2 idle cycles inserted
	1	3 idle cycles inserted (Initial value)

13.2.5 Wait State Control Register 2 (WCR2)

Wait state control register 2 (WCR2) is a 16-bit read/write register that specifies the number of wait state cycles inserted for each area. It also specifies the pitch of data access for burst memory accesses. This allows direct connection of even low-speed memories without an external circuit. WCR2 is initialized to H'FFFF by a power-on reset. It is not initialized by a manual reset or by standby mode.

Bit:	15	14	13	12	11	10	9	8
Bit name:	A6 W2	A6 W1	A6 W0	A5 W2	A5 W1	A5 W0	A4 W2	A4 W1
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	A4 W0	A3 W1	A3 W0	A2 W1	A2 W0	A0 W2	A0 W1	A0 W0
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15–13—Area 6 Wait Control (A6W2, A6W1, A6W0): Specify the number of wait states inserted into physical space area 6. Also specify the burst pitch for burst transfer.

			Description			
			First Cycle		Burst Cycle (Excluding First Cycle)	
Bit 15: A6W2	Bit 14: A6W1	Bit 13: A6W0	Inserted Wait States	$\overline{\text{WAIT}}$ Pin	Number of States Per Data Transfer	$\overline{\text{WAIT}}$ Pin
0	0	0	0	Disable	2	Enable
		1	1	Enable	2	Enable
	1	0	2	Enable	3	Enable
		1	3	Enable	4	Enable
1	0	0	4	Enable	4	Enable
		1	6	Enable	6	Enable
	1	0	8	Enable	8	Enable
		1	10 (Initial value)	Enable	10	Enable

Bits 12–10—Area 5 Wait Control (A5W2, A5W1, A5W0): Specify the number of wait states inserted into physical space area 5. Also specify the burst pitch for burst transfer.

			Description			
			First Cycle		Burst Cycle (Excluding First Cycle)	
Bit 12: A5W2	Bit 11: A5W1	Bit 10: A5W0	Inserted Wait States	$\overline{\text{WAIT}}$ Pin	Number of States Per Data Transfer	$\overline{\text{WAIT}}$ Pin
0	0	0	0	Disable	2	Enable
		1	1	Enable	2	Enable
	1	0	2	Enable	3	Enable
		1	3	Enable	4	Enable
1	0	0	4	Enable	4	Enable
		1	6	Enable	6	Enable
	1	0	8	Enable	8	Enable
		1	10 (Initial value)	Enable	10	Enable

Bits 9–7—Area 4 Wait Control (A4W2, A4W1, A4W0): Specify the number of wait states inserted into physical space area 4.

			Description	
Bit 9: A4W2	Bit 8: A4W1	Bit 7: A4W0	Inserted Wait State	$\overline{\text{WAIT}}$ Pin
0	0	0	0	Ignored
		1	1	Enable
	1	0	2	Enable
		1	3	Enable
1	0	0	4	Enable
		1	6	Enable
	1	0	8	Enable
		1	10	Enable (Initial value)

Bits 6, 5—Area 3 Wait Control (A3W1, A3W0): Specify the number of wait states inserted into physical space area 3. When the DRAM is used for area 3 and the EXTEND bit of BCR3 is set, the BSC ignores bits 6 and 5.

- For Ordinary memory

		Description	
Bit 6: A3W1	**Bit 5: A3W0**	**Inserted Wait States**	**$\overline{\text{WAIT}}$ Pin**
0	0	0	Ignored
	1	1	Enable
1	0	2	Enable
	1	3	Enable (Initial value)

- For DRAM, SDRAM

		Description	
Bit 6: A3W1	**Bit 5: A3W0**	**DRAM: CAS Assert Period**	**SDRAM: CAS Latency**
0	0	1	1
	1	1	1
1	0	2	2
	1	3	3 (Initial value)

Bits 4, 3—Area 2 Wait Control (A2W1, A2W0): Specify the number of wait states inserted into physical space area 2. When the DRAM is used for area 2 and the EXTEND bit of the BCR3 is set, the BSC ignores bits 4 and 3.

- For Ordinary memory

		Description	
Bit 4: A2W0	**Bit 3: A2W0**	**Inserted Wait States**	**$\overline{\text{WAIT}}$ Pin**
0	0	0	Ignored
	1	1	Enable
1	0	2	Enable
	1	3	Enable (Initial value)

- For DRAM, SDRAM

		Description	
Bit 4: A2W1	Bit 3: A2W0	DRAM: CAS Assert Period	SDRAM: CAS Latency
0	0	1	1
	1	1	1
1	0	2	2
	1	3	3 (Initial value)

Bits 2–0—Area 0 Wait Control (A0W2, A0W1, A0W0): Specify the number of wait states inserted into physical space area 0. Also specify the burst pitch for burst transfer.

			Description			
			First Cycle		Burst Cycle (Excluding First Cycle)	
Bit 2: A0W2	Bit 1: A0W1	Bit 0: A0W0	Inserted Wait States	$\overline{\text{WAIT}}$ Pin	Number of States Per Data Transfer	$\overline{\text{WAIT}}$ Pin
0	0	0	0	Ignored	2	Enable
		1	1	Enable	2	Enable
	1	0	2	Enable	3	Enable
		1	3	Enable	4	Enable
1	0	0	4	Enable	4	Enable
		1	6	Enable	6	Enable
	1	0	8	Enable	8	Enable
		1	10 (Initial value)	Enable	10	Enable

13.2.6 Individual Memory Control Register (MCR)

The individual memory control register (MCR) is a 16-bit read/write register that specifies$\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ timing and burst control for DRAM(area 3 only), SDRAM(areas 2 and 3), specifies address multiplexing, and controls refresh. This enables direct connection of DRAM, and SDRAM without external circuits.

The MCR is initialized to H'0000 by power-on resets, but is not initialized by manual resets or standby mode. The bits TPC1–TPC0, RCD1–RCD0, TRWL1–TRWL0, TRAS1–TRAS0, RASD, BE, SZ, AMX1–AMX0, and EDOMODE are written to at the initialization after a power-on reset and are not then modified again. When RFSH and RMODE are written to, write the same values to the other bits. When using DRAM, and SDRAM, do not access areas 2 and 3 until this register is initialized.

Bit:	15	14	13	12	11	10	9	8
Bit name:	TPC1	TPC0	RCD1	RCD0	TRWL1	TRWL0	TRAS1	TRAS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	RASD	BE	AMX2	AMX1	AMX0	RFSH	RMODE	EDO MODE
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 and 14—RAS Precharge Time (TPC1, TPC0): When DRAM interface is selected as connected memory, the TPC bits set the minimum number of cycles until the next RAS assertion after RAS negation. When SDRAM interface is selected, they set the minimum number of cycles until output of the next bank-active command after precharge. Set the same value for TPC31–30 and TPC21–20 in BCR3, TPC1–0 in the MCR, and TPC1–0 in the DCR.

Bit 15: TPC1	Bit 14: TPC0	Description
0	0	1 cycle (Initial value)
	1	2 cycles
1	0	3 cycles
	1	4 cycles

Bits 13 and 12—RAS–CAS Delay (RCD1, RCD0): The RCD bits set the RAS–CAS assert delay time for the connected memory when DRAM interface is selected. When SDRAM interface is selected, sets the bank active read/write command delay time. When the EXTEND bit of BCR3 is set, the BSC ignores the RCD bits.

Bit 13: RCD1	Bit 12: RCD0	Description
0	0	1 cycle (Initial value)
	1	2 cycles
1	0	3 cycles
	1	4 cycles

Bits 11 and 10—Write-Precharge Delay (TRWL1, TRWL0): The TRWL bits set the SDRAM write-precharge delay time. This designates the time between the end of a write cycle and the next bank-active command. This is valid only when SDRAM is connected. After the write cycle, the next bank-active command is not issued for the period TPC + TRWL.

Bit 11: TRWL1	Bit 10: TRWL0	Description
0	0	1 cycle (Initial value)
	1	2 cycles
1	0	3 cycles
	1	Reserved (Setting disabled)

Bits 9 and 8—$\overline{\text{CAS}}$-Before-$\overline{\text{RAS}}$ Refresh $\overline{\text{RAS}}$ Assert Time (TRAS1, TRAS0): When DRAM interface is selected as connected memory, the TRAS bits set the $\overline{\text{RAS}}$ assertion period for $\overline{\text{CAS}}$-before-$\overline{\text{RAS}}$ refreshes. When SDRAM interface is selected, no bank-active command is issues during the period TPC + TRAS after an auto-refresh command. Set the same value for TRAS31–30 and TRAS21–20 in BCR3, TRAS1–0 in the MCR, and TRAS1–0 in the DCR.

Bit 9: TRAS1	Bit 8: TRAS0	Description
0	0	2 cycles (Initial value)
	1	3 cycles
1	0	4 cycles
	1	5 cycles

Bit 7—SDRAM Bank Active (RASD): Specifies whether SDRAM is used in bank active mode or auto-precharge mode. Set areas 2 and 3 as auto-precharge modes when both of these areas arc sct in thc SDRAM spacc.

Bit 7: RASD	Description
0	Auto-precharge mode (Initial value)
1	Bank active mode

Bit 6—Burst Enable (BE): The BE bit specifies whether to conduct a burst access of DRAM. When accessing SDRAM, burst access is always carried out, regardless of this bit's designation.

Bit 6: BE	Description
0	Burst disabled (Initial value)
1	When DRAM interface, do a high-speed page mode access

Bit 5—Reserved: Bit 5 cannot be modified.

Bits 4 and 3—Address Multiplex (AMX1, AMX0): The AMX bits specify address multiplexing for DRAM and SDRAM. The actual address shift value differs between DRAM interface and SDRAM interface.

For DRAM Interface:

Bit 5: AMX2	Bit 4: AMX1	Bit 3: AMX0	Description
0	0	0	The row address begins with A9. (The A9 value is output at A1 when the row address is output.) (Initial value)
		1	The row address begins with A10. (The A10 value is output at A1 when the row address is output.)
	1	0	The row address begins with A11. (The A11 value is output at A1 when the row address is output.)
		1	(Reserved)
1	0	0	(Reserved)
		1	(Reserved)
	1	0	(Reserved)
		1	(Reserved)

For SDRAM Interface:

Bit5: AMX2	Bit 4: AMX1	Bit 3: AMX0	Description
1	0	0	The row address begins with A9. (The A9 value is output at A1 when the row address is output. 4 M × 16-bit products) (Initial value)
		1	The row address begins with A10. (The A10 value is output at A1 when the row address is output. 8 M × 8-bit products)
	1	1	The row address begins with A9. (The A11 value is output at A1 when the row address is output. 2 M × 32-bit products)
0	0	0	The row address begins with A9. (The A9 value is output at A1 when the row address is output. 1 M × 16-bit products) (Initial value)
		1	The row address begins with A10. (The A10 value is output at A1 when the row address is output. 2 M × 8-bit products)
	1	0	The row address begins with A11. (The A11 value is output at A1 when the row address is output. 4 M × 4-bit products)
		1	The row address begins with A9. (The A9 value is output at A1 when the row address is output. 256 K × 16-bit products)

Bit 2—Refresh Control (RFSH): The RFSH bit determines whether or not the refresh operation of the DRAM and SDRAM is performed. The timer for generation of the refresh request frequency can also be used as an interval timer.

Bit 2: RFSH	Description
0	No refresh (Initial value)
1	Refresh

Bit 1—Refresh Mode (RMODE): The RMODE bit selects whether to perform an ordinary refresh or a self-refresh when the RFSH bit is 1. When the RFSH bit is 1 and this bit is 0, a CAS-before-RAS refresh or an auto-refresh is performed on DRAM or SDRAM at the period set by the refresh-related registers RTCNT, RTCOR and RTCSR. When a refresh request occurs during an external bus cycle, the bus cycle will be ended and the refresh cycle performed. When the RFSH bit is 1 and this bit is also 1, the DRAM or SDRAM will wait for the end of any executing external bus cycle before going into a self-refresh. All refresh requests to memory that is in the self-refresh state are ignored.

Bit 1: RMODE	Description
0	CAS-before-RAS refresh (RFSH must be 1) (Initial value)
1	Self-refresh (RFSH must be 1)

Bit 0—EDO Mode (EDOMODE): The EDOMODE bit specifies the data sampling timing during data reads when using DRAM in EDO mode. Operating timing of memory other than DRAM does not change even if this bit is set. This bit is only valid for DRAM connected to area 3. Do not set this bit to 1 when using SDRAM.

Bit 0: EDOMODE	Description
0	Set when using normal DRAM. Data sampling timing during read cycle is on the falling edge of BCLK. (Initial value)
1	Set when using the EDO-mode DRAM. Data sampling timing during read cycle is on the rising edge of BCLK. Also, RAS signal negate timing is delayed 1/2 machine cycle.

13.2.7 DRAM Control Register (DCR)

The DRAM control register (DCR) is a 16-bit read/write register that specifies$\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ timing and burst control for DRAM connected to area 2. It also specifies address multiplexing and controls refresh. When DRAM is connected to area 2, the bus width is fixed at 16 bits. The DCR is initialized to H'0000 by power-on resets, but is not initialized by manual resets or standby mode. Do not access external memory outside area 2 until this register's initialization is complete.

Bit:	15	14	13	12	11	10	9	8
Bit name:	TPC1	TPC0	RCD1	RCD0	—	—	TRAS1	TRAS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	BE	—	AMX1	AMX0	RFSH	RMODE	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R	R/W	R/W	R/W	R/W	R

Bits 15 and 14—RAS Precharge Time (TPC1, TPC0): The TPC bits set the RAS precharge time for the DRAM connected to area 2. Set the same value for TPC31–30 and TPC21–20 in BCR3, TPC1–0 in the MCR, and TPC1–0 in the DCR.

Bit 15: TPC1	Bit 14: TPC0	Description
0	0	1 cycle (Initial value)
	1	2 cycles
1	0	3 cycles
	1	4 cycles

Bits 13 and 12—RAS-CAS Delay (RCD1, RCD0): The RCD bits set the RAS-CAS delay time for the DRAM connected to area 2. When the BCR3 EXTEND bit is set, the BSCP ignores bits 13 and 12.

Bit 13: RCD1	Bit 12: RCD0	Description
0	0	1 cycle (Initial value)
	1	2 cycles
1	0	3 cycles
	1	4 cycles

Bits 9 and 8—CAS-Before-RAS Refresh RAS Assert Time (TRAS1, TRAS0): The TRAS bits set timing for the DRAM connected to area 2. These bits set the RAS assert period for CAS-before-RAS refreshes. Set the same value for TRAS31–30 and TRAS21–20 in BCR3, TRAS1–0 in the MCR, and TRAS1–0 in the DCR.

Bit 9: TRAS1	Bit 8: TRAS0	Description
0	0	2 cycles (Initial value)
	1	3 cycles
1	0	4 cycles
	1	5 cycles

Bit 6—Burst Enable (BE): The BE bit specifies whether to conduct a burst access of the DRAM connected to area 2.

Bit 6: BE	Description
0	Burst disabled (Initial value)
1	High-speed page mode access

Bits 4 and 3—Address Multiplex (AMX1, AMX0): The AMX bits specify address multiplexing for the DRAM connected to area 2.

Bit 4: AMX1	Bit 3: AMX0	Description
0	0	8-bit column-address products (Initial value)
	1	9-bit column-address products
1	0	10-bit column-address products
	1	11-bit column-address products

Bit 2—Refresh Control (RFSH): The RFSH bit determines whether or not the refresh operation of the DRAM connected to area 2 is performed.

Bit 2: RFSH	Description
0	No refresh (Initial value)
1	Refresh

Bit 1—Refresh Mode (RMODE): The RMODE bit selects the refresh mode for the DRAM connected to area 2.

Bit 1: RMODE	Description
0	CAS-before-RAS refresh (RFSH must be 1) (Initial value)
1	Self-refresh (RFSH must be 1)

Bits 11, 10, 7, 5, and 0—Reserved: Bits 11, 10, 7, 5, and 0 cannot be modified. They always read 0.

13.2.8 PCMCIA Control Register (PCR)

Bit:	15	14	13	12	11	10	9	8
Bit name:	A6W3	A5W3	—	—	A5TED2	A6TED2	A5TEH2	A6TEH2
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	A5TED1	A5TED0	A6TED1	A6TED0	A5TEH1	A5TEH0	A6TEH1	A6TEH0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 15—Area 6 Wait Control (A6W3): The A6W3 bit specifies the number of inserted wait states for area 6 combined with bits A6W2–A6W0 in WCR2. It also specifies the number of transfer states in burst transfer. Set this bit to 0 when area 6 is not set to PCMCIA.

				Top Cycle		Burst Cycle	
A6W3	**A6W2**	**A6W1**	**A6W0**	**Inserted Wait State**	**WAIT Pin**	**Number of States per One-data Transfer**	**WAIT Pin**
0	0	0	0	0	Ignored	2	Enabled
0	0	0	1	1	Enabled	2	Enabled
0	0	1	0	2	Enabled	3	Enabled
0	0	1	1	3	Enabled	4	Enabled
0	1	0	0	4	Enabled	5	Enabled
0	1	0	1	6	Enabled	7	Enabled
0	1	1	0	8	Enabled	9	Enabled
0	1	1	1	10 (Initial value)	Enabled	11	Enabled
1	0	0	0	12	Enabled	13	Enabled
1	0	0	1	14	Enabled	15	Enabled
1	0	1	0	18	Enabled	19	Enabled
1	0	1	1	22	Enabled	23	Enabled
1	1	0	0	26	Enabled	27	Enabled
1	1	0	1	30	Enabled	31	Enabled
1	1	1	0	34	Enabled	35	Enabled
1	1	1	1	38	Enabled	39	Enabled

Bit 14—Area 5 Wait Control (A5W3): The A5W3 bit specifies the number of inserted wait states for area 5 combined with bits A5W2–A5W0 in WCR2. It also specifies the number of transfer states in burst transfer. Set this bit to 0 when area 5 is not set to PCMCIA.

The relationship between the setting value and the number of waits is the same as A6W3.

Bits 11, 7, and 6—Area 5 Address OE/WE Assert Delay (A5TED2, A5TED1, and A5TED0): The A5TED bits specify the address to OE/WE assert delay time for the PCMCIA interface connected to area 5.

Bit 11: A5TED2	Bit 7: A5TED1	Bit 6: A5TED0	Description
0	0	0	0.5-cycle delay (Initial value)
		1	1.5-cycle delay
	1	0	2.5-cycle delay
		1	3.5-cycle delay
1	0	0	4.5-cycle delay
		1	5.5-cycle delay
	1	0	6.5-cycle delay
		1	7.5-cycle delay

Bits 10, 5 and 4—Area 6 Address OE/WE Assert Delay (A6TED2, A6TED1, and A6TED0): The A6TED bits specify the address to OE/WE assert delay time for the PCMCIA interface connected to area 6.

Bit 10: A6TED2	Bit 5: A6TED1	Bit 4: A6TED0	Description
0	0	0	0.5-cycle delay (Initial value)
		1	1.5-cycle delay
	1	0	2.5-cycle delay
		1	3.5-cycle delay
1	0	0	4.5-cycle delay
		1	5.5-cycle delay
	1	0	6.5-cycle delay
		1	7.5-cycle delay

Bits 9, 3, and 2—Area 5 OE/WE Negate Address Delay (A5TEH2, A5TEH1, and A5TEH0): The A5TEH bits specify the OE/WE negate address delay time for the PCMCIA interface connected to area 5.

Bit 9: A5TEH2	Bit 3: A5TEH1	Bit 2: A5TEH0	Description
0	0	0	0.5-cycle delay (Initial value)
		1	1.5-cycle delay
	1	0	2.5-cycle delay
		1	3.5-cycle delay
1	0	0	4.5-cycle delay
		1	5.5-cycle delay
	1	0	6.5-cycle delay
		1	7.5-cycle delay

Bits 8, 1, and 0—Area 6 OE/WE Negate Address Delay (A6TEH2, A6TEH1, and A6TEH0): The A6TEH bits specify the OE/WE negate address delay time for the PCMCIA interface connected to area 6.

Bit 8: A6TEH2	Bit 1: A6TEH1	Bit 0: A6TEH0	Description
0	0	0	0.5-cycle delay (Initial value)
		1	1.5-cycle delay
	1	0	2.5-cycle delay
		1	3.5-cycle delay
1	0	0	4.5-cycle delay
		1	5.5-cycle delay
	1	0	6.5-cycle delay
		1	7.5-cycle delay

13.2.9 SDRAM Modc Rcgistcr (SDMR)

The SDRAM mode register (SDMR) is written to via the SDRAM address bus and is an 8-bit write-only register. It sets SDRAM mode for areas 2 and 3. SDMR is undefined after a power-on reset. The register contents are not initialized by a manual reset or standby mode; values remain unchanged.

Writcs to thc SDRAM modc rcgistcr usc thc addrcss bus rathcr than thc data bus. If thc valuc to bc set is X and the SDMR address is Y, the value X is written in the SDRAM mode register by writing in address X + Y. Since, with a 32-bit bus width, A0 of the SDRAM is connected to A2 of the chip and A1 of the SDRAM is connected to A3 of the chip, the value actually written to the SDRAM is the X value shifted two bits right. With a 16-bit bus width, the value written is the X value shifted one bit right. For example, with a 32-bit bus width, when H'0230 is written to the SDMR register of area 2, random data is written to the address H'FFFFD000 (address Y) + H'08C0 (value X), or H'FFFFD8C0. As a result, H'0230 is written to the SDMR register. The range for value X is H'0000 to H'0FFC. When H'0230 is written to the SDMR register of area 3, random data is written to the address H'FFFFE000 (address Y) + H'08C0 (value X), or H'FFFFE8C0. As a result, H'0230 is written to the SDMR register. The range for value X is H'0000 to H'0FFC.

Bit:	31		12	11	10	9	8
Bit name:	SDMR address			—	—	—	—
Initial value:	—	....	—	—	—	—	—
R/W:	—	....	—	W*	W*	W	W

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	—	—	—	—	—
Initial value:	—	—	—	—	—	—	—	—
R/W:	W	W	W	W	W	W	—	—

Note: Depending on the type of SDRAM.

13.2.10 Refresh Timer Control/Status Register (RTCSR)

The refresh timer control/status register (RTSCR) is a 16-bit read/write register that specifies the refresh cycle, whether to generate an interrupt, and that interrupt's cycle. It is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or standby mode.

Note: Writing to the RTCSR differs from that to general registers to ensure the RTCSR is not rewritten incorrectly. Use the word-transfer instruction to set the upper byte as B'10100101 and the lower byte as the write data. For details, see section 13.2.14, Cautions on Accessing Refresh Control Related Registers.

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

Bit:	7	6	5	4	3	2	1	0
Bit name:	CMF	CMIE	CKS2	CKS1	CKS0	OVF	OVIE	LMTS
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15–8—Reserved: Bits 15–8 cannot be modified. They always read 0.

Bit 7—Compare Match Flag (CMF): The CMF status flag indicates that the values of RTCNT and RTCOR match.

Bit 7: CMF	Description
0	The values of RTCNT and RTCOR do not match. Clear condition: When a refresh is performed After 0 has been written in CMF and RFSH = 1 and RMODE = 0 (to perform a CBR refresh). (Initial value)
1	The values of RTCNT and RTCOR match. Set condition: RTCNT = RTCOR *

Note : Contents do not change when 1 is written to CMF.

Bit 6—Compare Match Interrupt Enable (CMIE): Enables or disables an interrupt request caused when the CMF of RTCSR is set to 1. Do not set this bit to 1 when using CAS-before-RAS refresh or auto-refresh.

Bit 6: CMIE	Description
0	Disables an interrupt request caused by CMF (Initial value)
1	Enables an interrupt request caused by CMF

Bits 5–3—Clock Select Bits (CKS2–CKS0): Select the clock input to RTCNT. The source clock is the external bus clock (BCLK). The RTCNT count clock is CKIO divided by the specified ratio. The specified ratios are shown below in the normal external bus clock and when setting the external bus clock to 1/4 by setting the SLPFRQ bit in FRQCR to 1 in sleep mode.

			Description	
Bit 5: CKS2	Bit 4: CKS1	Bit 3: CKS0	Normal external bus clock	1/4-bus clock
0	0	0	Disables clock input	Disables clock input (Initial value)
		1	Bus clock (CKIO)/4	CKIO/1
	1	0	CKIO/16	CKIO/4
		1	CKIO/64	CKIO/16
1	0	0	CKIO/256	CKIO/64
		1	CKIO/1024	CKIO/256
	1	0	CKIO/2048	CKIO/512
		1	CKIO/4096	CKIO/1024

Bit 2—Refresh Count Overflow Flag (OVF): The OVF status flag indicates when the number of refresh requests indicated in the refresh count register (RFCR) exceeds the limit set in the LMTS bit of RTCSR.

Bit 2: OVF	Description
0	RFCR has not exceeded the count limit value set in LMTS Clear Conditions: When 0 is written to OVF (Initial value)
1	RFCR has exceeded the count limit value set in LMTS Set Conditions: When the RFCR value has exceeded the count limit value set in LMTS*

Note: Contents don't change when 1 is written to OVF.

Bit 1—Refresh Count Overflow Interrupt Enable (OVIE): OVIE selects whether to suppress generation of interrupt requests by OVF when the OVF bit of RTCSR is set to 1.

Bit 1: OVIE	Description
0	Disables interrupt requests from the OVF (Initial value)
1	Enables interrupt requests from the OVF

Bit 0—Refresh Count Overflow Limit Select (LMTS): Indicates the count limit value to be compared to the number of refreshes indicated in the refresh count register (RFCR). When the value RFCR overflows the value specified by LMTS, the OVF flag is set.

Bit 0: LMTS	Description
0	Count limit value is 1024 (Initial value)
1	Count limit value is 512

13.2.11 Refresh Timer Counter (RTCNT)

RTCNT is a 16-bit read/write register. RTCNT is an 8-bit counter that counts up with input clocks. The clock select bits (CKS2–CKS0) of RTCSR select the input clock. When RTCNT matches RTCOR, the CMF bit of RTCSR is set and RTCNT is cleared. RTCNT is initialized to H'00 by a power-on reset; it continues incrementing after a manual reset; it is not initialized by standby mode and holds its values unchanged.

Note: Writing to the RTCNT differs from that to general registers to ensure the RTCNT is not rewritten incorrectly. Use the word-transfer instruction to set the upper byte as B'10100101 and the lower byte as the write data. For details, see section 13.2.14, Cautions on Accessing Refresh Control Related Registers.

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

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

13.2.12 Refresh Time Constant Register (RTCOR)

The refresh time constant register (RTCOR) is a 16-bit read/write register. The values of RTCOR and RTCNT (bottom 8 bits) are constantly compared. When the values match, the compare match flag (CMF) of RTCSR is set and RTCNT is cleared to 0. When the refresh bit (RFSH) of the individual memory control register (MCR) is set to 1 and the refresh mode is set to CAS-before-RAS refresh, a memory refresh cycle occurs when the CMF bit is set. RTCOR is initialized to H'00 by a power-on reset. It is not initialized by a manual reset or standby mode, but holds its contents. Make the RTCOR setting before setting bits CKS2 to CKS0 in RTCSR.

Note: Writing to the RTCOR differs from that to general registers to ensure the RTCOR is not rewritten incorrectly. Use the word-transfer instruction to set the upper byte as B'10100101 and the lower byte as the write data. For details, see section 13.2.14, Cautions on Accessing Refresh Control Related Registers.

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

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

13.2.13 Refresh Count Register (RFCR)

The refresh count register (RFCR) is a 16-bit read/write register. It is a 10-bit counter that increments every time RTCOR and RTCNT match. When RFCR exceeds the count limit value set in the LMTS of RTCSR, RTCSR's OVF bit is set and RFCR clears. RFCR is initialized to H'0000 when a power-on reset is performed. It is not initialized by a manual reset or standby mode, but holds its contents.

Note: Writing to the RFCR differs from that to general registers to ensure the RFCR is not rewritten incorrectly. Use the word-transfer instruction to set the MSB and followed six bits of upper bytes as B'101001 and remaining bits as the write data. For details, see section 13.2.14, Cautions on Accessing Refresh Control Related Registers.

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

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

13.2.14 Cautions on Accessing Refresh Control Related Registers

RFCR, RTCSR, RTCNT, and RTCOR require that a specific code be appended to the data when it is written to prevent data from being mistakenly overwritten by program overruns or other write operations (figure 13.5). Perform reads and writes using the following methods:

1. When writing to RFCR, RTCSR, RTCNT, and RTCOR, use only word transfer instructions. You cannot write with byte transfer instructions.

   When writing to RTCNT, RTCSR, or RTCOR, place B'10100101 in the upper byte and the write data in the lower byte. When writing to RFCR, place B'101001 in the top 6 bits and the write data in the remaining bits, as shown in figure 13.5.

2. When reading from RFCR, RTCSR, RTCNT, and RTCOR, carry out reads with 16-bit width. 0 is read out from undefined bit sections.

	15							8	7 — 0
RTCSR, RTCNT, RTCOR	1	0	1	0	0	1	0	1	Write data

	15					10	9 — 0
RFCR	1	0	1	0	0	1	Write data

Figure 13.5 Writing to RFCR, RTCSR, RTCNT, and RTCOR

13.2.15MCS0 Control Register (MCSCR0)

The MCS0 control register (MCSCR0) is a 16-bit read/write register that specifies the $\overline{\text{MCS0}}$ pin output conditions.

$\overline{\text{MCSCR0}}$ is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or in standby mode.

As the $\overline{\text{MCS0}}$ pin is multiplexed as the PTC0 pin, when using the pin as $\overline{\text{MCS0}}$, bits PC0MD[1:0] in the PCCR register should be set to 00 (other function).

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

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	CS2/0	CAP1	CAP0	A25	A24	A23	A22
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 to 7—Reserved: Only 0 should be written to these bits.

Bit 6—CS2/CS0 Select (CS2/0): Selects whether an area 2 or area 0 address is to be decoded.

Bit 6: CS2/0	Description
0	Area 0 is selected
1	Area 2 is selected

Bits 5, 4—Connected Memory Size Specification (CAP1, CAP0)

Bit 5: CAP1	Bit 4: CAP0	Description
0	0	32-Mbit memory is connected
0	1	64-Mbit memory is connected
1	0	128-Mbit memory is connected
1	1	256-Mbit memory is connected

Bits 3 to 0—Start Address Specification (A25, A24, A23, A22): These bits specify the start address of the memory area for which $\overline{\text{MCS0}}$ is asserted.

13.2.16MCS1 Control Register (MCSCR1)

The MCS1 control register (MCSCR1) specifies the $\overline{\text{MCS1}}$ pin output conditions.

The bit configuration and functions are the same as those of MCSCR0.

13.2.17MCS2 Control Register (MCSCR2)

The MCS2 control register (MCSCR2) specifies the $\overline{\text{MCS2}}$ pin output conditions.

The bit configuration and functions are the same as those of MCSCR0.

13.2.18MCS3 Control Register (MCSCR3)

The MCS3 control register (MCSCR3) specifies the $\overline{\text{MCS3}}$ pin output conditions.

The bit configuration and functions are the same as those of MCSCR0.

13.2.19MCS4 Control Register (MCSCR4)

The MCS4 control register (MCSCR4) specifies the $\overline{\text{MCS4}}$ pin output conditions.

The bit configuration and functions are the same as those of MCSCR0.

13.2.20MCS5 Control Register (MCSCR5)

The MCS5 control register (MCSCR5) specifies the $\overline{\text{MCS5}}$ pin output conditions.

The bit configuration and functions are the same as those of MCSCR0.

13.2.21MCS6 Control Register (MCSCR6)

The MCS6 control register (MCSCR6) specifies the $\overline{\text{MCS6}}$ pin output conditions.

The bit configuration and functions are the same as those of MCSCR0.

13.2.22MCS7 Control Register (MCSCR7)

The MCS7 control register (MCSCR7) specifies the $\overline{\text{MCS7}}$ pin output conditions.

The bit configuration and functions are the same as those of MCSCR0.

13.3 BSC Operation

13.3.1 Endian/Access Size and Data Alignment

This LSI supports both big endian, in which the 0 address is the most significant byte in the byte data, and little endian, in which the 0 address is the least significant byte. This switchover is designated by an external pin (MD5 pin) at the time of a power-on reset. After a power-on reset, big endian is engaged when MD5 is low; little endian is engaged when MD5 is high.

Three data bus widths are available for ordinary memory (byte, word, longword) and two data bus widths (word and longword) for DRAM. Only longword is available for SDRAM. For the PCMCIA interface, choose from byte and word. This means data alignment is done by matching the device's data width and endian. The access unit must also be matched to the device's bus width. This also means that when longword data is read from a byte-width device, the read operation must happen 4 times. In this LSI, data alignment and conversion of data length is performed automatically between the respective interfaces.

Tables 13.7 through 13.12 show the relationship between endian, device data width, and access unit.

Table 13.732-Bit External Device/Big Endian Access and Data Alignment

$\overline{\text{WE3}}$, $\overline{\text{CASHH}}$, DQMUU	$\overline{\text{WE2}}$, $\overline{\text{CASHL}}$, DQMUL	$\overline{\text{WE1}}$, $\overline{\text{CASLH}}$, DQMLU	$\overline{\text{WE0}}$, $\overline{\text{CASLL}}$, DQMLL	D31–D24	D23–D16	D15–D8	D7–D0	Operation
Assert	—	—	—	Data 7–0	—	—	—	Byte access at 0
—	Assert	—	—	—	Data 7–0	—	—	Byte access at 1
—	—	Assert	—	—	—	Data 7–0	—	Byte access at 2
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 3
Assert	Assert	—	—	Data 15–8	Data 7–0	—	—	Word access at 0
—	—	Assert	Assert	—	—	Data 15–8	Data 7–0	Word access at 2
Assert	Assert	Assert	Assert	Data 31–24	Data 23–16	Data 15–8	Data 7–0	Longword access at 0

Table 13.816-Bit External Device/Big Endian Access and Data Alignment

$\overline{\text{WE3}}$, $\overline{\text{CASHH}}$, DQMUU	$\overline{\text{WE2}}$, $\overline{\text{CASHL}}$, DQMUL	$\overline{\text{WE1}}$, $\overline{\text{CASLH}}$, DQMLU	$\overline{\text{WE0}}$, $\overline{\text{CASLL}}$, DQMLL	D31–D24	D23–D16	D15–D8	D7–D0	Operation	
—	—	Assert	—	—	—	Data 7–0	—	Byte access at 0	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 1	
—	—	Assert	—	—	—	Data 7–0	—	Byte access at 2	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 3	
—	—	Assert	Assert	—	—	Data 15–8	Data 7–0	Word access at 0	
—	—	Assert	Assert	—	—	Data 15–8	Data 7–0	Word access at 2	
—	—	Assert	Assert	—	—	Data 31–24	Data 23–16	Longword access at 0	1st time at 0
—	—	Assert	Assert	—	—	Data 15–8	Data 7–0		2nd time at 2

Table 13.98-Bit External Device/Big Endian Access and Data Alignment

$\overline{\text{WE3}}$,	$\overline{\text{WE2}}$,	$\overline{\text{WE1}}$,	$\overline{\text{WE0}}$,						
$\overline{\text{CASHH}}$,	$\overline{\text{CASHL}}$,	$\overline{\text{CASLH}}$,	$\overline{\text{CASLL}}$,	D31–	D23–	D15–			
DQMUU	DQMUL	DQMLU	DQMLL	D24	D16	D8	D7–D0	Operation	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 0	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 1	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 2	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 3	
—	—	—	Assert	—	—	—	Data 15–8	Word access at 0	1st time at 0
—	—	—	Assert	—	—	—	Data 7–0		2nd time at 1
—	—	—	Assert	—	—	—	Data 15–8	Word access at 2	1st time at 2
—	—	—	Assert	—	—	—	Data 7–0		2nd time at 3
—	—	—	Assert	—	—	—	Data 31–24	Longword access at 0	1st time at 0
—	—	—	Assert	—	—	—	Data 23–16		2nd time at 1
—	—	—	Assert	—	—	—	Data 15–8		3rd time at 2
—	—	—	Assert	—	—	—	Data 7–0		4th time at 3

Table 13.10 32-Bit External Device/Little Endian Access and Data Alignment

$\overline{\text{WE3}}$, $\overline{\text{CASHH}}$, DQMUU	$\overline{\text{WE2}}$, $\overline{\text{CASHL}}$, DQMUL	$\overline{\text{WE1}}$, $\overline{\text{CASLH}}$, DQMLU	$\overline{\text{WE0}}$, $\overline{\text{CASLL}}$, DQMLL	D31–D24	D23–D16	D15–D8	D7–D0	Operation
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 0
—	—	Assert	—	—	—	Data 7–0	—	Byte access at 1
—	Assert	—	—	—	Data 7–0	—	—	Byte access at 2
Assert	—	—	—	Data 7–0	—	—	—	Byte access at 3
—	—	Assert	Assert	—	—	Data 15–8	Data 7–0	Word access at 0
Assert	Assert	—	—	Data 15–8	Data 7–0	—	—	Word access at 2
Assert	Assert	Assert	Assert	Data 31–24	Data 23–16	Data 15–8	Data 7–0	Longword access at 0

Table 13.11 16-Bit External Device/Little Endian Access and Data Alignment

$\overline{\text{WE3}}$, $\overline{\text{CASHH}}$, DQMUU	$\overline{\text{WE2}}$, $\overline{\text{CASHL}}$, DQMUL	$\overline{\text{WE1}}$, $\overline{\text{CASLH}}$, DQMLU	$\overline{\text{WE0}}$, $\overline{\text{CASLL}}$, DQMLL	D31–D24	D23–D16	D15–D8	D7–D0	Operation	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 0	
—	—	Assert	—	—	—	Data 7–0	—	Byte access at 1	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 2	
—	—	Assert	—	—	—	Data 7–0	—	Byte access at 3	
—	—	Assert	Assert	—	—	Data 15–8	Data 7–0	Word access at 0	
—	—	Assert	Assert	—	—	Data 15–8	Data 7–0	Word access at 2	
—	—	Assert	Assert	—	—	Data 15–8	Data 7–0	Longword access at 0	1st time at 0
—	—	Assert	Assert	—	—	Data 31–24	Data 23–16		2nd time at 2

Table 13.12 8-Bit External Device/Little Endian Access and Data Alignment

$\overline{\text{WE3}}$, $\overline{\text{CASHH}}$, DQMUU	$\overline{\text{WE2}}$, $\overline{\text{CASHL}}$, DQMUL	$\overline{\text{WE1}}$, $\overline{\text{CASLH}}$, DQMLU	$\overline{\text{WE0}}$, $\overline{\text{CASLL}}$, DQMLL	D31– D24	D23– D16	D15– D8	D7–D0	Operation	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 0	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 1	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 2	
—	—	—	Assert	—	—	—	Data 7–0	Byte access at 3	
—	—	—	Assert	—	—	—	Data 7–0	Word access at 0	1st time at 0
—	—	—	Assert	—	—	—	Data 15–8		2nd time at 1
—	—	—	Assert	—	—	—	Data 7–0	Word access at 2	1st time at 2
—	—	—	Assert	—	—	—	Data 15–8		2nd time at 3
—	—	—	Assert	—	—	—	Data 7–0	Longword access at 0	1st time at 0
—	—	—	Assert	—	—	—	Data 15–8		2nd time at 1
—	—	—	Assert	—	—	—	Data 23–16		3rd time at 2
—	—	—	Assert	—	—	—	Data 31–24		4th time at 3

13.3.2 Description of Areas

Area 0: Area 0 physical addresses A28–A26 are 000. Addresses A31–A29 are ignored and the address range is H'00000000 + H'20000000 × n – H'03FFFFFF + H'20000000 × n (n = 0–6 and n = 1–6 are the shadow spaces).

Ordinary memories such as SRAM, ROM, and burst ROM can be connected to this space. Byte, word, or longword can be selected as the bus width using external pins MD3 and MD4. When the area 0 space is accessed, a $\overline{\text{CS0}}$ signal is asserted. An $\overline{\text{RD}}$ signal that can be used as $\overline{\text{OE}}$ and the $\overline{\text{WE0}}$–$\overline{\text{WE3}}$ signals for write control are also asserted. The number of bus cycles is selected between 0 and 10 wait cycles using the A0W2–A0W0 bits of WCR2. When the burst function is used, the bus cycle pitch of the burst cycle is determined within a range of 2–10 according to the number of waits.

Area 1: Area 1 physical addresses A28–A26 are 001. Addresses A31–A29 are ignored and the address range is H'04000000 + H'20000000 × n – H'07FFFFFF + H'20000000 × n (n = 0–6 and n = 1–6 are the shadow spaces).

Area 1 is the area specifically for the internal peripheral modules. The external memories cannot be connected.

Control registers of peripheral modules shown below are mapped to this area 1. Their addresses are physical address, to which logical addresses can be mapped with the MMU enabled:
DMAC, PORT, IrDA, SCIF, ADC, DAC, INTC (except INTEVT, IPRA, IPRB)
Those registers must not be cached.

Area 2: Area 2 physical addresses A28–A26 are 010. Addresses A31–A29 are ignored and the address range is H'08000000 + H'20000000 × n – H'0BFFFFFF + H'20000000 × n (n = 0–6 and n = 1–6 are the shadow spaces).

Ordinary memories like SRAM and ROM, as well as DRAM and SDRAM, can be connected to this space. Byte, word, or longword can be selected as the bus width using the A2SZ1–A2SZ0 bits of BCR2 for ordinary memory. When DRAM is connected to Area 2, the bus width is fixed at 16 bits.

When the area 2 space is accessed, a $\overline{\text{CS2}}$ signal is asserted. When ordinary memories are connected, an $\overline{\text{RD}}$ signal that can be used as $\overline{\text{OE}}$ and the $\overline{\text{WE0}}$–$\overline{\text{WE3}}$ signals for write control are also asserted and the number of bus cycles is selected between 0 and 3 wait cycles using the A2W1 to A2W0 bits of WCR2.

When SDRAM is connected, the $\overline{\text{RAS3U}}$, $\overline{\text{RAS3L}}$ signal, $\overline{\text{CASU}}$, $\overline{\text{CASL}}$ signal, RD/$\overline{\text{WR}}$ signal, and byte controls DQMHH, DQMHL, DQMLH, and DQMLL are all asserted and addresses multiplexed. Control of $\overline{\text{RAS3U}}$, $\overline{\text{RAS3L}}$, $\overline{\text{CASU}}$, $\overline{\text{CASL}}$, data timing, and address multiplexing is set with MCR.

When DRAM is connected, the RAS2 signal, CAS2H signal, CAS2L signal, and RD/WR signal are all asserted and addresses multiplexed. Control of RAS2, CAS, data timing, and address multiplexing is set with DCR.

Area 3: Area 3 physical addresses A28–A26 are 011. Addresses A31–A29 are ignored and the address range is H'0C000000 + H'20000000 × n – H'0FFFFFFF + H'20000000 × n (n = 0–6 and n = 1–6 are the shadow spaces).

Ordinary memories like SRAM and ROM, as well as DRAM, and SDRAM, can be connected to this space. Byte, word or longword can be selected as the bus width using the A3SZ1–A3SZ0 bits of BCR2 for ordinary memory. For DRAM, word or longword can be selected using the SZ bit of MCR.

When area 3 space is accessed, CS3 is asserted.

When ordinary memories are connected, an RD signal that can be used as OE and the WE0–WE3 signals for write control are asserted and the number of bus cycles is selected between 0 and 3 wait cycles using the A3W1 to A3W0 bits of WCR2.

When SDRAM is connected, the RAS3U, RAS3L signal, CASU, CASL signal, RD/WR signal, and byte controls DQMHH, DQMHL, DQMLH, and DQMLL are all asserted and addresses multiplexed. When DRAM is connected, the RAS3U, RAS3L signal, CASHH signal, CASHL signal, CASLH signal, CASLL signal, and RD/WR signal are all asserted and addresses multiplexed. For all of these, the RAS signal, CASHH signal, CASHL signal, CASLH signal, CASLL signal, and RD/WR signal are all asserted and addresses multiplexed. Control of RAS, CAS, and data timing and of address multiplexing is set with MCR.

Area 4: Area 4 physical addresses A28–A26 are 100. Addresses A31–A29 are ignored and the address range is H'10000000 + H'20000000 × n – H'13FFFFFF + H'20000000 × n (n = 0–6 and n = 1–6 are the shadow spaces).

Only ordinary memories like SRAM and ROM can be connected to this space. Byte, word, or longword can be selected as the bus width using the A4SZ1–A4SZ0 bits of BCR2. When the area 4 space is accessed, a CS4 signal is asserted. An RD signal that can be used as OE and the WE0–WE3 signals for write control are also asserted. The number of bus cycles is selected between 0 and 10 wait cycles using the A4W2–A4W0 bits of WCR2.

Area 5: Area 5 physical addresses A28–A26 are 101. Addresses A31–A29 are ignored and the address range is the 64 Mbytes at H'14000000 + H'20000000 × n – H'17FFFFFF + H'20000000 × n (n = 0–6 and n = 1–6 are the shadow spaces).

Ordinary memories like SRAM and ROM as well as burst ROM and PCMCIA interfaces can be connected to this space. When the PCMCIA interface is used, the IC memory card interface address range comprises the 32 Mbytes at H'14000000 + H'20000000 × n to H'15FFFFFF + H'20000000 × n (n = 0–6 and n = 1–6 are the shadow spaces), and the I/O card interface address range comprises the 32 Mbytes at H'16000000 + H'20000000 × n to H'17FFFFFF + H'20000000 × n (n = 0–6 and n = 1–6 are the shadow spaces).

For ordinary memory and burst ROM, byte, word, or longword can be selected as the bus width using the A5SZ1–A5SZ0 bits of BCR2. For the PCMCIA interface, byte, and word can be selected as the bus width using the A5SZ1–A5SZ0 bits of BCR2.

When the area 5 space is accessed and ordinary memory is connected, a $\overline{\text{CS5}}$ signal is asserted. An RD signal that can be used as $\overline{\text{OE}}$ and the $\overline{\text{WE0}}$–$\overline{\text{WE3}}$ signals for write control are also asserted. When the PCMCIA interface is used, the $\overline{\text{CE1A}}$ signal, $\overline{\text{CE2A}}$ signal, $\overline{\text{RD}}$ signal as $\overline{\text{OE}}$ signal, and $\overline{\text{WE1}}$ signal are asserted.

The number of bus cycles is selected between 0 and 10 wait cycles using the A5W2–A5W0 bits of WCR2. With the PCMCIA interface, from 0 to 38 wait cycles can be selected using the A5W2–A5W0 bits of WCR2 and the A5W3 bit of PCR. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{WAIT}}$). When a burst function is used, the bus cycle pitch of the burst cycle is determined within a range of 2–11 (2–39 for the PCMCIA interface) according to the number of waits. The setup and hold times of address/$\overline{\text{CS5}}$ for the read/write strobe signals can be set in the range 0.5–7.5 using A5TED2–A5TED0 and A5TEH2 to A5TEH0 bits of the PCR register.

Area 6: Area 6 physical addresses A28–A26 are 110. Addresses A31–A29 are ignored and the address range is the 64 Mbytes at H'18000000 + H'20000000 × n – H'1BFFFFFF + H'20000000 × n (n = 0–6 and n = 1–6 are the shadow spaces).

Ordinary memories like SRAM and ROM as well as burst ROM and PCMCIA interfaces can be connected to this space. When the PCMCIA interface is used, the IC memory card interface address range is 32 Mbytes at H'18000000 + H'20000000 × n – H'19FFFFFF + H'20000000 × n and the I/O card interface address range is 32 Mbytes at H'1A000000 + H'20000000 × n – H'1BFFFFFF + H'20000000 × n (n = 0–6 and n = 1–6 are the shadow spaces).

For ordinary memory and burst ROM, byte, word, or longword can be selected as the bus width using the A6SZ1–A6SZ0 bits of BCR2. For the PCMCIA interface, byte, and word can be selected as the bus width using the A6SZ1–A6SZ0 bits of BCR2.

When the area 6 space is accessed and ordinary memory is connected, a $\overline{\text{CS6}}$ signal is asserted. An $\overline{\text{RD}}$ signal that can be used as $\overline{\text{OE}}$ and the $\overline{\text{WE0}}$–$\overline{\text{WE3}}$ signals for write control are also asserted. When the PCMCIA interface is used, the $\overline{\text{CE1B}}$ signal, $\overline{\text{CE2B}}$ signal, $\overline{\text{RD}}$ signal as $\overline{\text{OE}}$ signal, and $\overline{\text{WE}}$, $\overline{\text{ICIORD}}$, and $\overline{\text{ICIOWR}}$ signals are asserted.

The number of bus cycles is selected between 0 and 10 wait cycles using the A6W2–A6W0 bits of WCR2. With the PCMCIA interface, from 0 to 38 wait cycles can be selected using the A6W2–A6W0 bits of WCR2 and the A6W3 bit of PCR. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{WAIT}}$). The bus cycle pitch of the burst cycle is determined within a range of 2–11 (2–39 for the PCMCIA interface) according to the number of waits. The setup and hold times of address/$\overline{\text{CS6}}$ for the read/write strobe signals can be set in the range 0.5–7.5 using A6TED2–A6TED0 and A6TEH2–A6TEH0 bits of the PCR register.

13.3.3 Basic Interface

Basic Timing: The basic interface of the SH7729 uses strobe signal output in consideration of the fact that mainly static RAM will be directly connected. Figure 13.6 shows the basic timing of normal space accesses. A no-wait normal access is completed in two cycles. The $\overline{\text{BS}}$ signal is asserted for one cycle to indicate the start of a bus cycle. The $\overline{\text{CSn}}$ signal is negated on the T2 clock falling edge to secure the negation period. Therefore, in case of access at minimum pitch, there is a half-cycle negation period.

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, 32 bits are always read in case of a 32-bit device, and 16 bits in case of a 16-bit device. When writing, only the $\overline{\text{WE}}$ signal for the byte to be written is asserted. For details, see section 13.3.1, Endian/Access Size and Data Alignment.

Read/write for cache fill or write-back follows the set bus width and transfers a total of 16 bytes continuously. The bus is not released during this transfer. For cache misses that occur during byte or word operand accesses or branching to odd word boundaries, the fill is always performed by longword accesses on the chip-external interface. Write-through-area write access and non-cacheable read/write access are based on the actual address size.

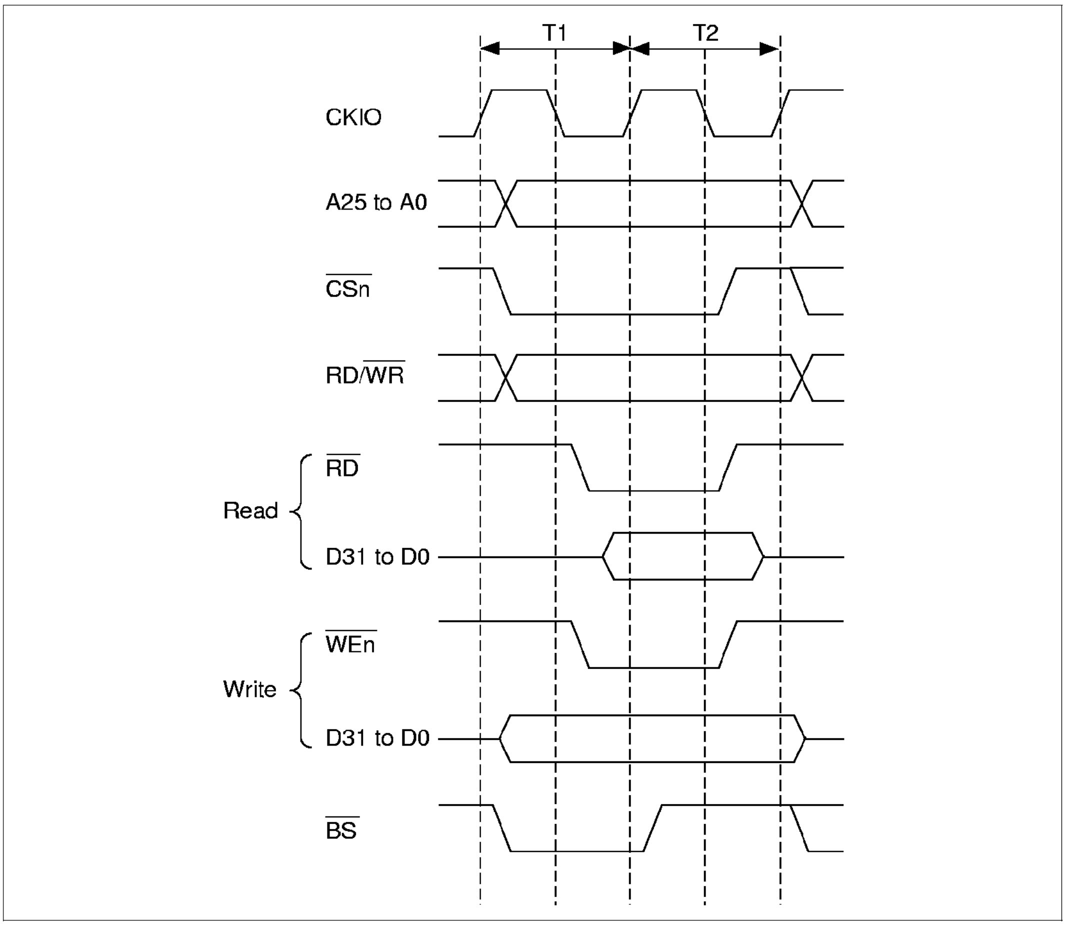


Figure 13.6 Basic Timing of Basic Interface

Figures 13.7, 13.8, and 13.9 show examples of connection to 32, 16, and 8-bit data-width static RAM, respectively.

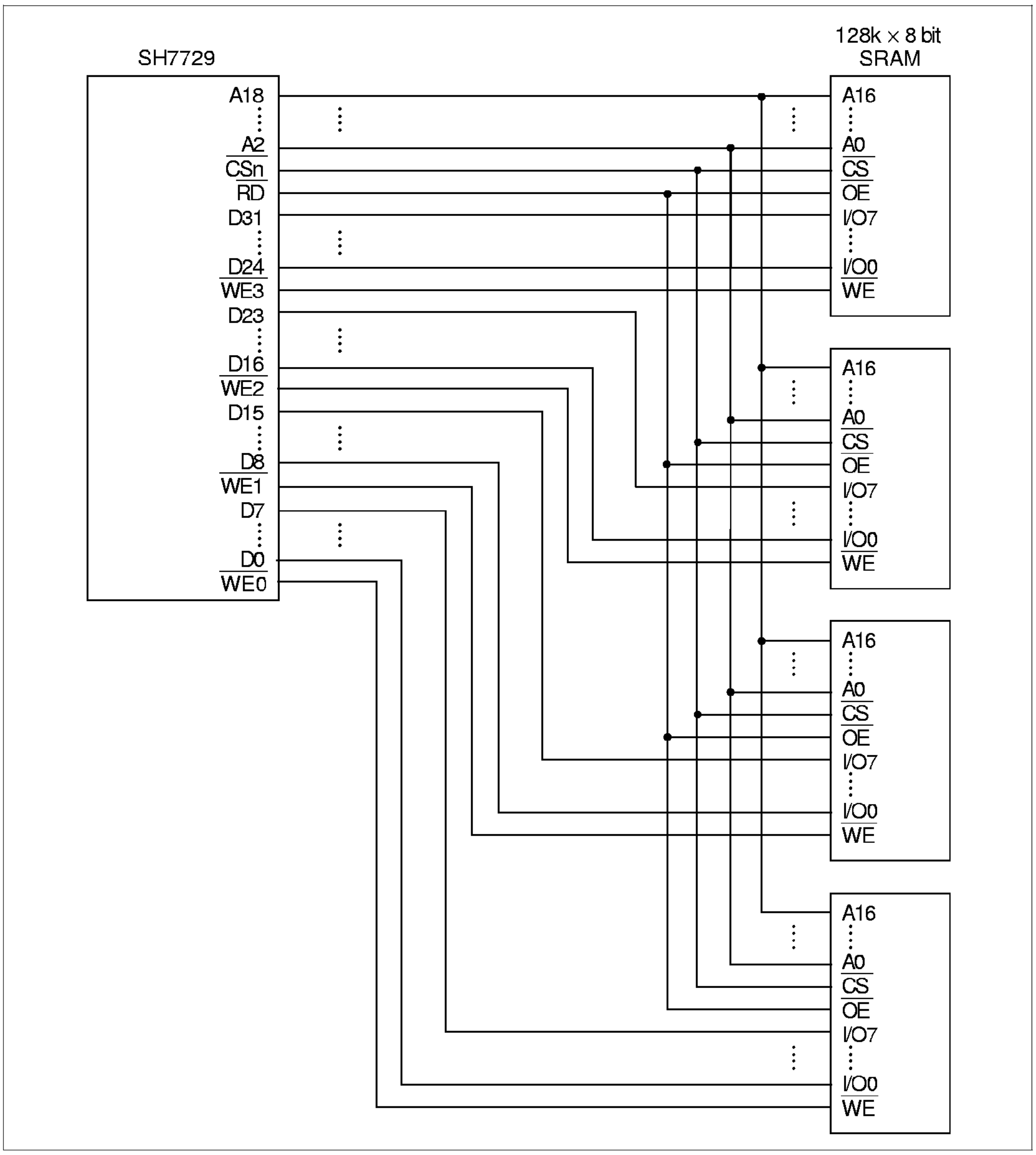


Figure 13.7 Example of 32-Bit Data-Width Static RAM Connection

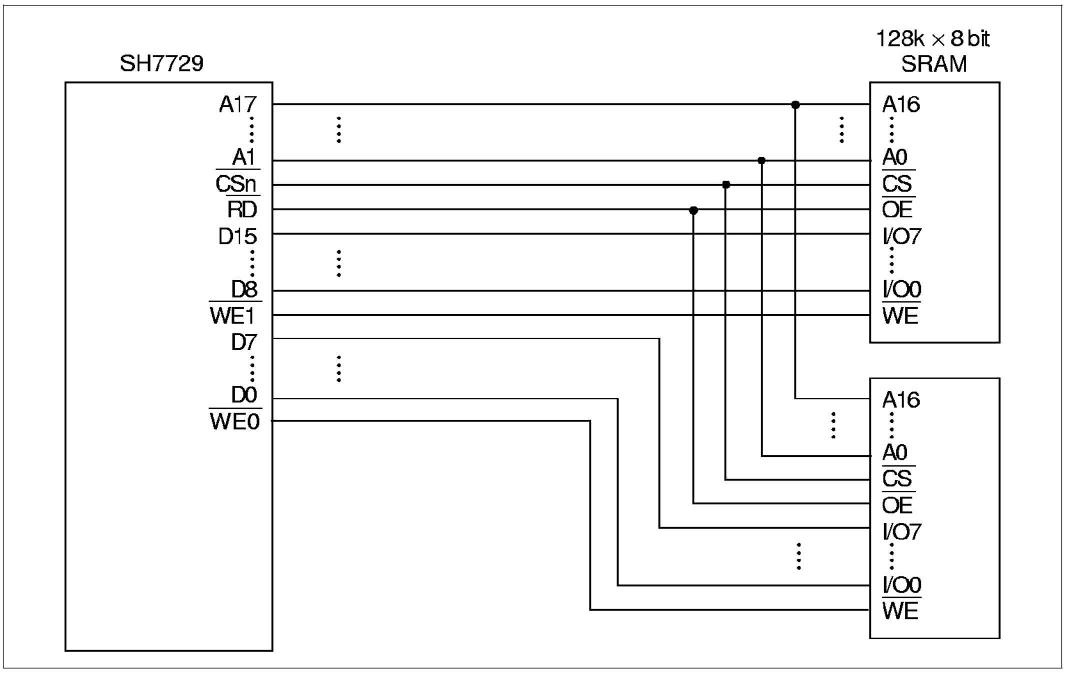


Figure 13.8 Example of 16-Bit Data-Width Static RAM Connection

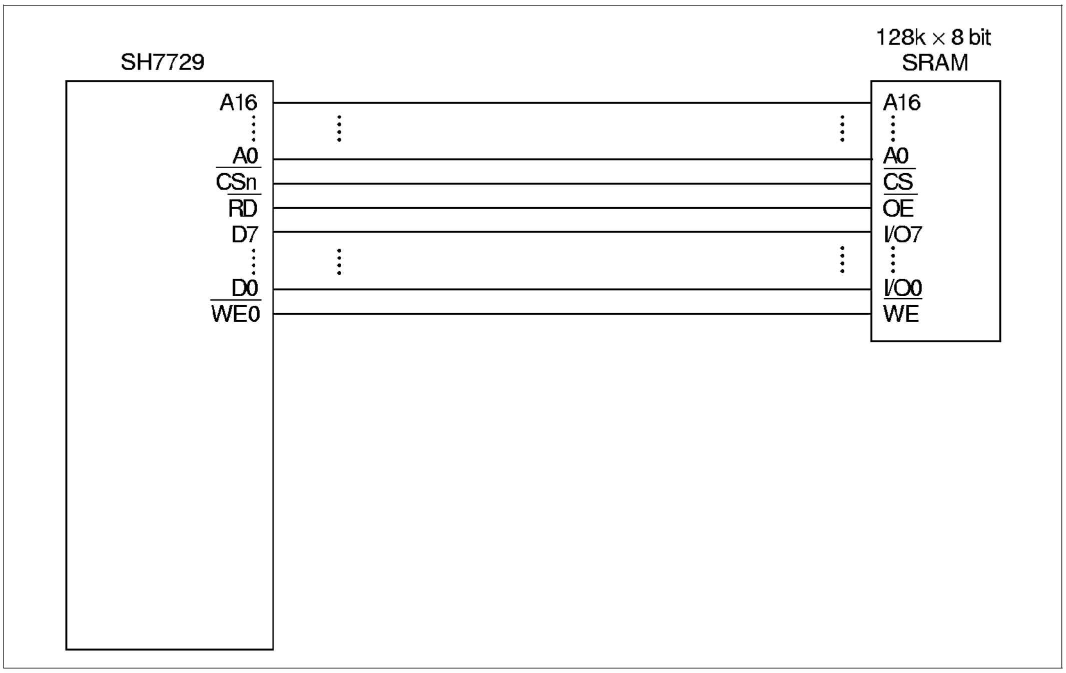


Figure 13.9 Example of 8-Bit Data-Width Static RAM Connection

Wait State Control: Wait state insertion on the basic interface can be controlled by the WCR2 settings. If the WCR2 wait specification bits corresponding to a particular area are not zero, a software wait is inserted in accordance with that specification. For details, see section 13.2.5, Wait Control Register 2 (WCR2).

The specified number of Tw cycles are inserted as wait cycles using the basic interface wait timing shown in figure 13.10.

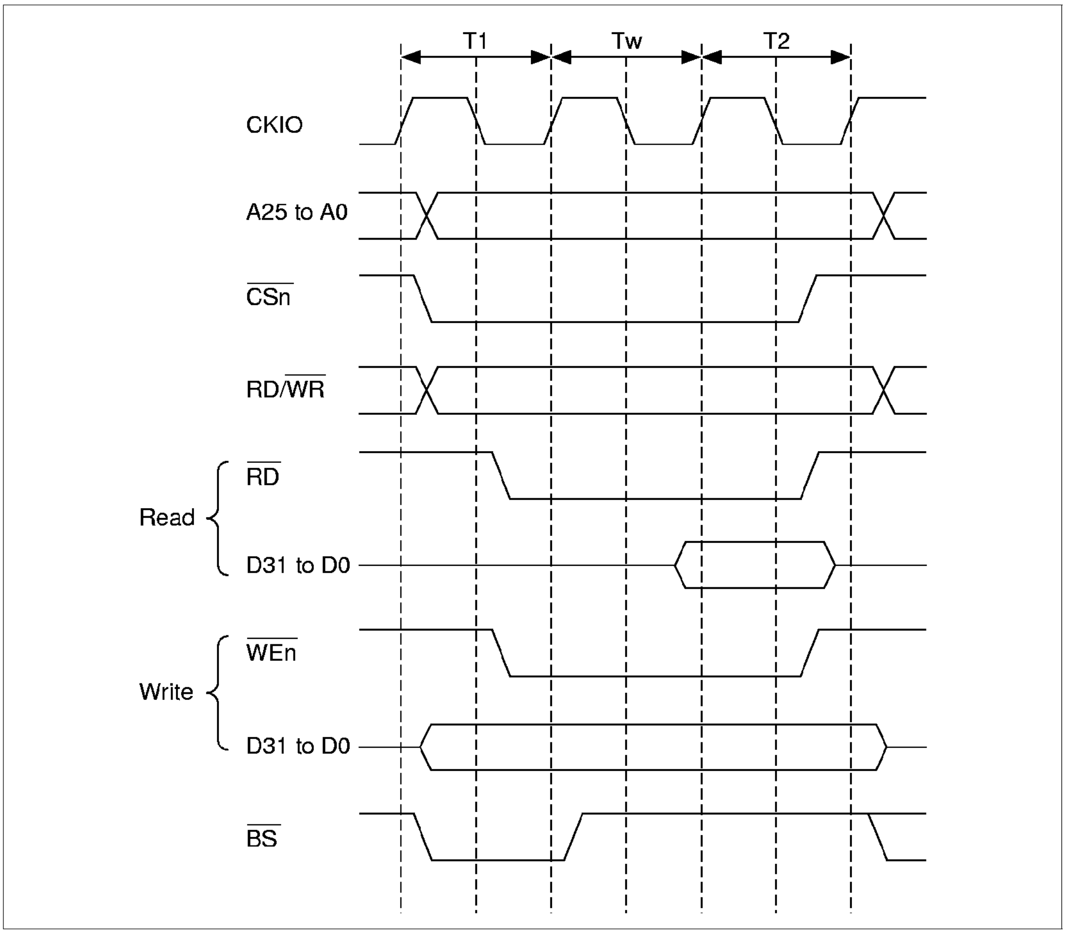


Figure 13.10 Basic Interface Wait Timing (Software Wait Only)

When software wait insertion is specified by WCR2, the external wait input WAIT signal is also sampled. WAIT pin sampling is shown in figure 13.11. A 2-cycle wait is specified as a software wait. Sampling is performed at the transition from the Tw state to the T2 state; therefore, if the WAIT signal has no effect if asserted in the T1 cycle or the first Tw cycle. When the WAITSEL bit in WCR1 is cleared to 0, the WAIT signal is sampled on the rising edge of the clock.

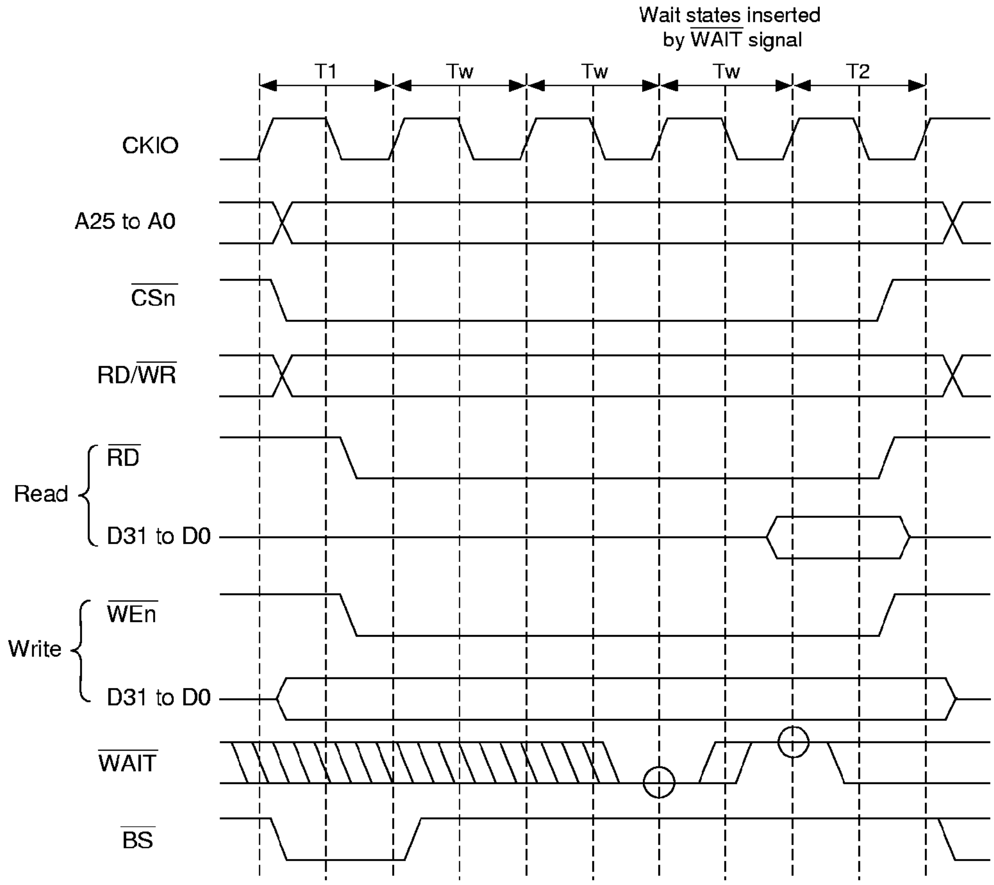


Figure 13.11 Basic Interface Wait State Timing (Wait State Insertion by WAIT Signal WAITSEL = 0)

When the WAITSEL bit in the WCR1 register is set to 1, the $\overline{\text{WAIT}}$ signal is sampled at the falling edge of the clock. If the setup time and hold times with respect to the falling edge of the clock are not satisfied, the value sampled at the next falling edge is used.

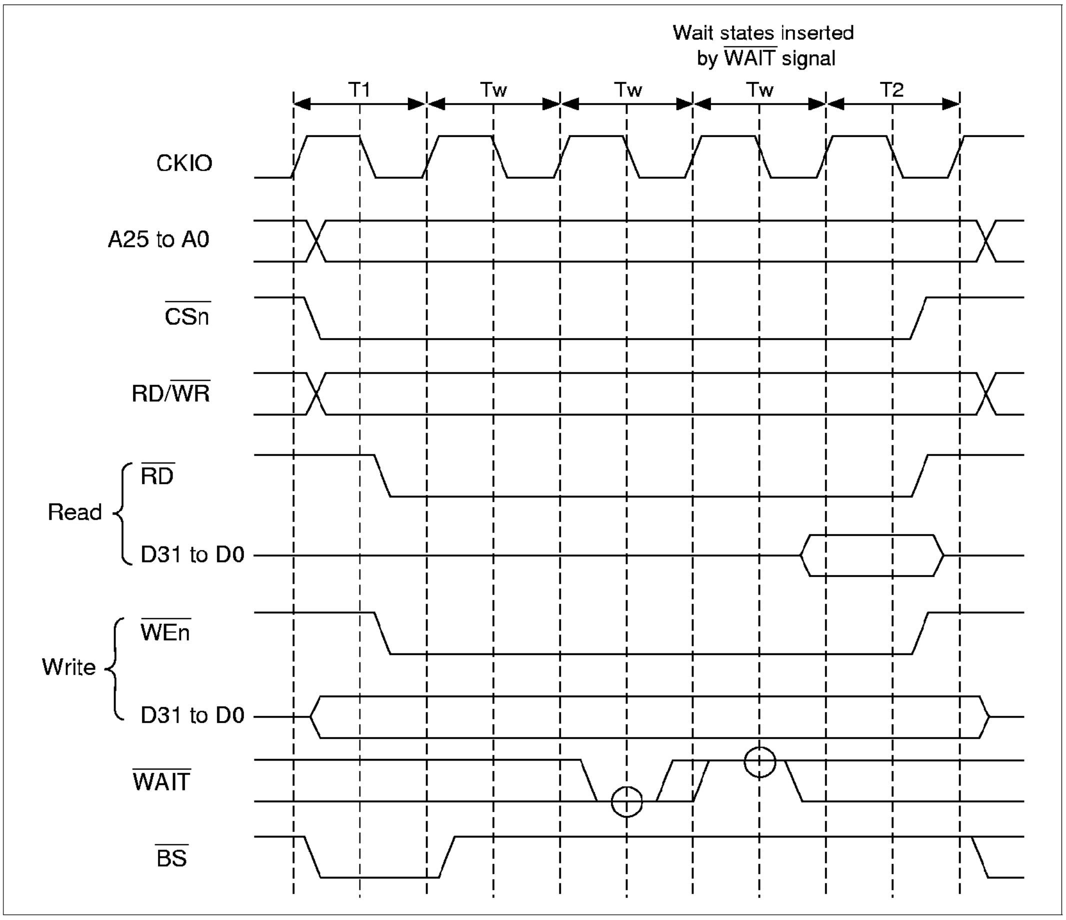


Figure 13.12 Basic Interface Wait State Timing (Wait State Insertion by $\overline{\text{WAIT}}$ Signal WAITSEL = 1)

13.3.4 DRAM Interface

DRAM Connection Method: When the memory type bits (DRAMTP2–0) in BCR1 are set to 100, area 3 becomes DRAM space; when set to 101, area 2 and area 3 become DRAM space. The DRAM interface function can then be used to connect the SH7729 directly to DRAM.

16 or 32 bits can be selected as the interface data width for area 3 when bits DRAMTP2–0 are set to 100, and 16 bits can be used for area 2 and 16 or 32 bits can be used for area 3 when bits DRAMTP2–0 are set to 101.

2-CAS 16-bit DRAMs can be connected, since $\overline{\text{CAS}}$ is used to control byte access.

Signals used for connection when DRAM is connected to area 3 are$\overline{\text{RAS3L}}$, $\overline{\text{RAS3U}}$, $\overline{\text{CASHH}}$, $\overline{\text{CASHL}}$, $\overline{\text{CASLH}}$, $\overline{\text{CASLL}}$, and RD/$\overline{\text{WR}}$. $\overline{\text{CASHH}}$ and $\overline{\text{CASHL}}$ are not used when the data width is 16 bits. When DRAM is connected to areas 2 and 3, the signals for DRAM connection in area 2 are $\overline{\text{RAS2L}}$, $\overline{\text{RAS2U}}$, $\overline{\text{CAS2H}}$, $\overline{\text{CAS2L}}$, and RD/$\overline{\text{WR}}$, and those for DRAM connection in area 3 are $\overline{\text{RAS3L}}$, $\overline{\text{RAS3U}}$, $\overline{\text{CASLH}}$, $\overline{\text{CASLL}}$, and RD/$\overline{\text{WR}}$.

In addition to normal read and write access modes, high-speed page mode is supported for burst access. Also, for DRAM connected to area 3, EDO mode, which enables the DRAM access time to be increased by delaying the data sampling timing by 1/2 clock when reading, is supported in addition to normal read and write access for burst mode.

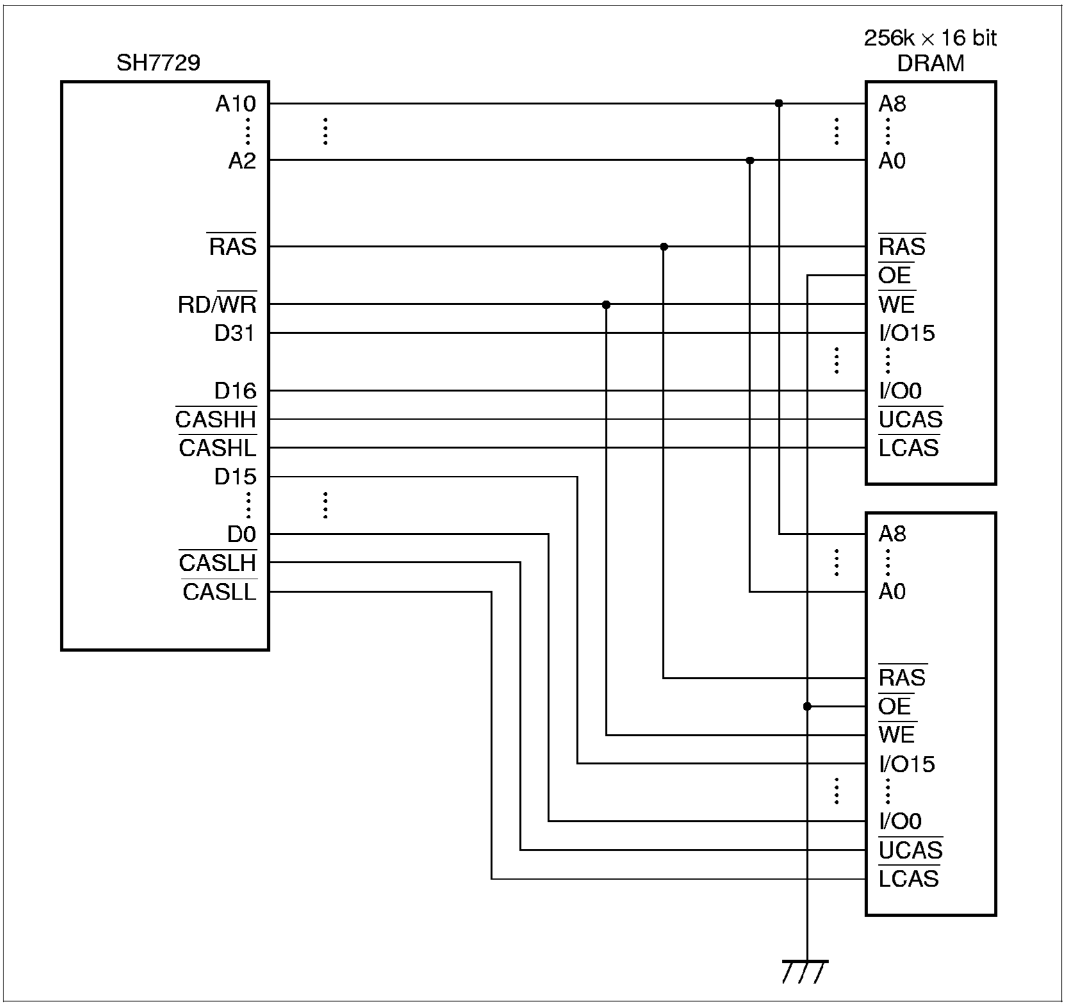


Figure 13.13 Example of DRAM Connection (32-Bit Data Width)

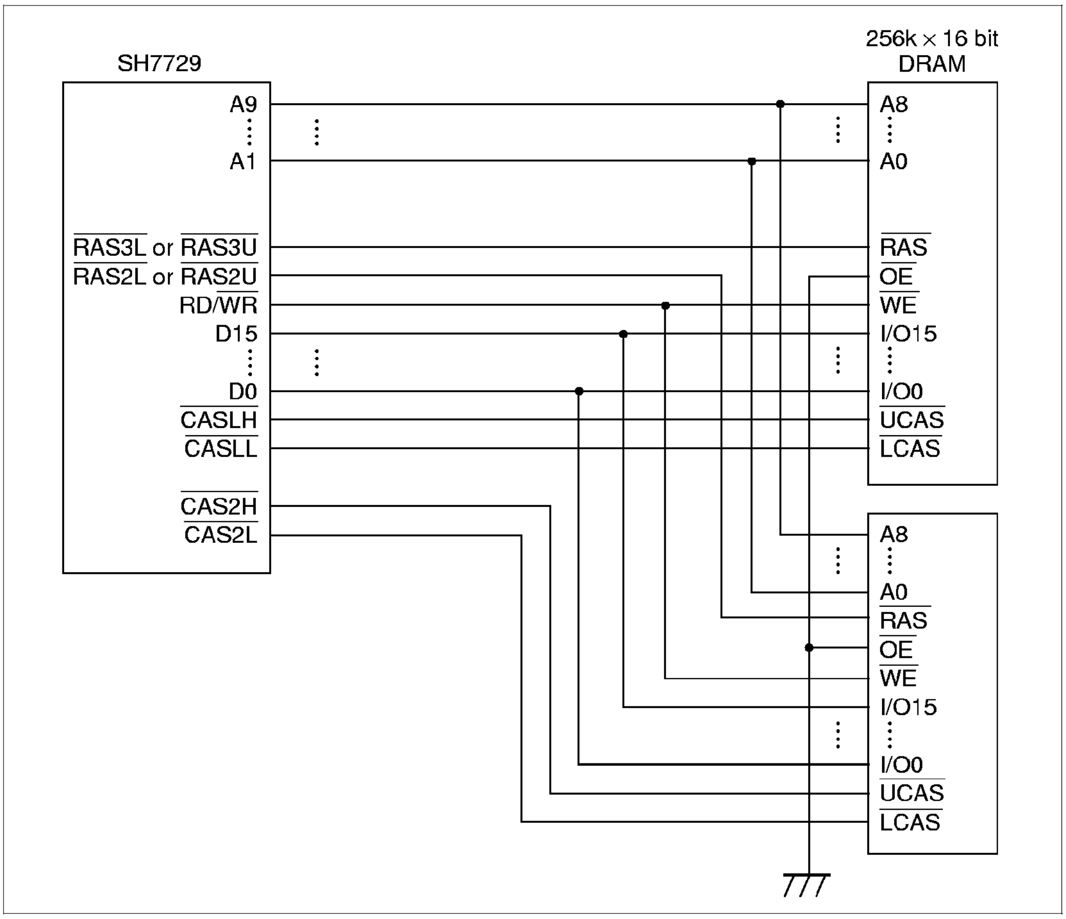


Figure 13.14 Example of DRAM Connection (16-Bit Data Width)

Address Multiplexing: When area 2 and area 3 are designated as DRAM space, address multiplexing is always performed in accesses to DRAM. This enables DRAM, which requires row and column address multiplexing, to be connected directly to the SH7729 without using an external address multiplexer. Any of the four multiplexing methods shown below can be selected, by setting bits AMX1-0 in MCR for area 3 DRAM, or bits AMX1-0 in DCR for area 2 DRAM. The relationship between bits AMX1-0 and address multiplexing is shown in table 13.13. The address output pins subject to address multiplexing are A15 to A1. Pins A25 to A16 carry the original address.

Table 13.13 Relationship between AMX1-0 and Address Multiplexing

Setting		Number of Column		External Address Pins	
AMX1	AMX0	Address Bits	Output Timing	A1 to A14	A15
0	0	8 bits	Column address	A1 to A14	A15
			Row address	A9 to A22	A23
0	1	9 bits	Column address	A1 to A14	A15
			Row address	A10 to A23	A24
1	0	10 bits	Column address	A1 to A14	A15
			Row address	A11 to A24	A25
1	1	11 bits	Column address	A1 to A14	A15
			Row address	A12 to A25	A15

Basic Timing: The basic timing for DRAM access is 3 cycles (RCD (1–0) = 00, AnW (1–0) = 00). This basic timing is shown in figure 13.15. Tpc is the precharge cycle, Tr the RAS assert cycle, Tc1 the CAS assert cycle, and Tc2 the read data latch cycle.

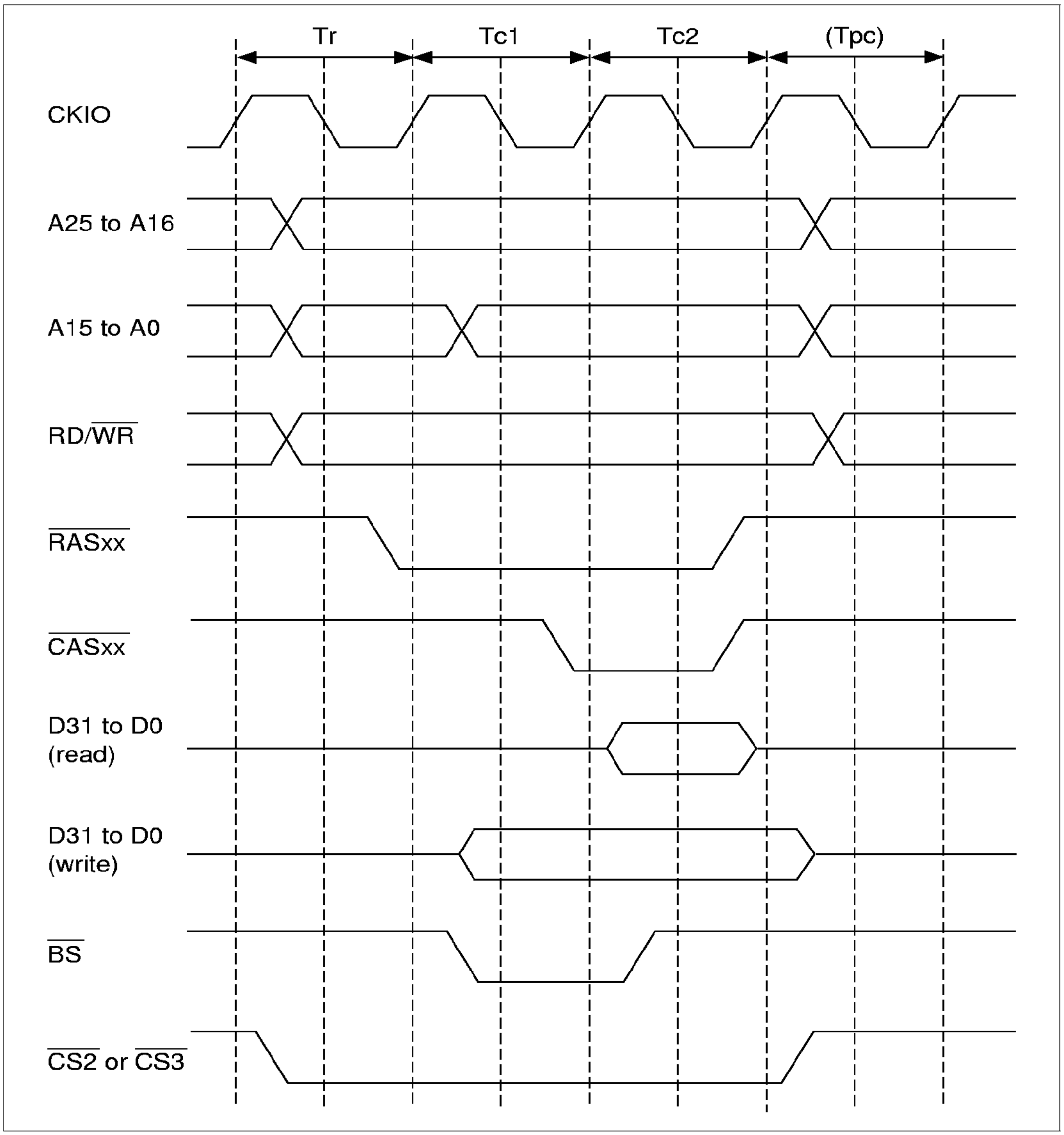

Tr
Tc1
Tc2
(Tpc)
CKIO
A25 to A16
A15 to A0
RD/$\overline{\text{WR}}$
$\overline{\text{RASxx}}$
$\overline{\text{CASxx}}$
D31 to D0 (read)
D31 to D0 (write)
$\overline{\text{BS}}$
$\overline{\text{CS2}}$ or $\overline{\text{CS3}}$

Figure 13.15 Basic Timing for DRAM Access

Wait State Control: As the clock frequency increases, it becomes impossible to complete all states in one cycle as in basic access. Therefore, provision is made for state extension by using the setting bits in WCR2, MCR, DCR, and BCR3. The timing with state extension using these settings is shown in figure 13.16. Additional Tpc cycles (cycles used to secure the RAS precharge time) can be inserted by means of the TPC bits in MCR and DCR, giving from 1 to 4 cycles. The number of cycles from RAS assertion to CAS assertion can be set to between 1 and 4 by inserting Trw cycles by means of the RCD bits in MCR and DCR. And the number of cycles from CAS assertion to the end of the access can be varied between 1 and 3 according to the setting of A2W (1–0) or A3W (1–0) in WCR2.

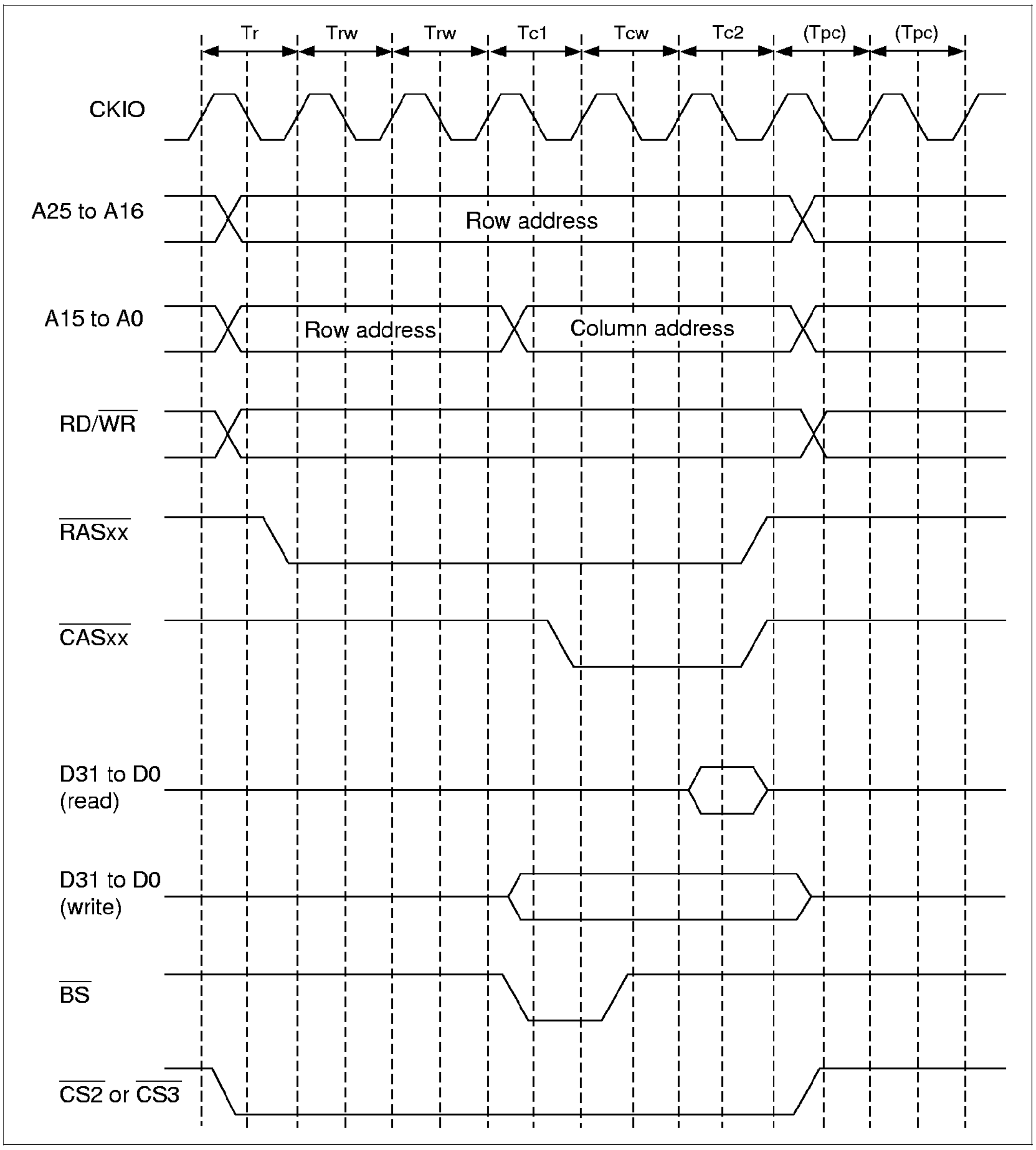
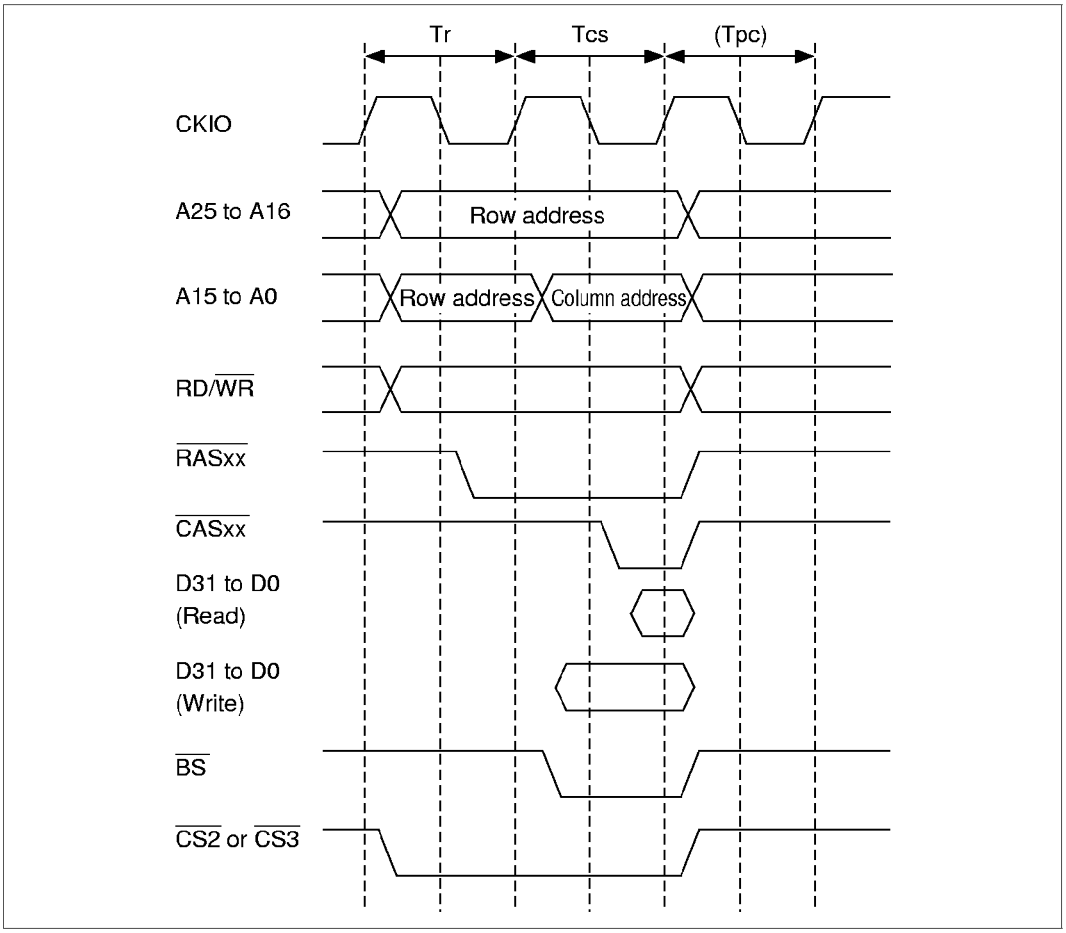


Figure 13.16 DRAM Wait State Timing

Short Pitch Access: The bus state controller of the SH7729 can perform short-pitch access to DRAM as shown in figure 13.17, when AnW (1–0) is set to 00. Tcs is a CAS assert and data access state.



Tr
Tcs
(Tpc)
CKIO
A25 to A16
Row address
A15 to A0
Row address
Column address
RD/$\overline{\text{WR}}$
$\overline{\text{RASxx}}$
$\overline{\text{CASxx}}$
D31 to D0 (Read)
D31 to D0 (Write)
$\overline{\text{BS}}$
$\overline{\text{CS2}}$ or $\overline{\text{CS3}}$

Figure 13.17 DRAM Short-Pitch Access Timing

Burst Access: In addition to the normal DRAM access mode in which a row address is output in each data access, a high-speed page mode is also provided for the case where consecutive accesses are made to the same row. This mode allows fast access to data by outputting the row address only once, then changing only the column address for each subsequent access. Normal access or burst access using high-speed page mode can be selected by means of the burst enable (BE) bit in MCR and DCR. The timing for burst access using high-speed page mode is shown in figure 13.18.

In burst transfer, 4 (longword access) or 16 (cache fill or cache write-back) bytes of data are burst-transferred in case of a 16-bit bus size. With a 32-bit bus size, 16 bytes of data are burst-transferred (cache fill or cache write-back). In a 16-byte burst transfer (cache fill), the first access comprises a longword that includes the data requiring access. The remaining accesses are performed on 16-byte boundary data that includes the relevant data. In burst transfer (cache write-back), sequential writing is performed from first-to-last order for 16-byte boundary data.

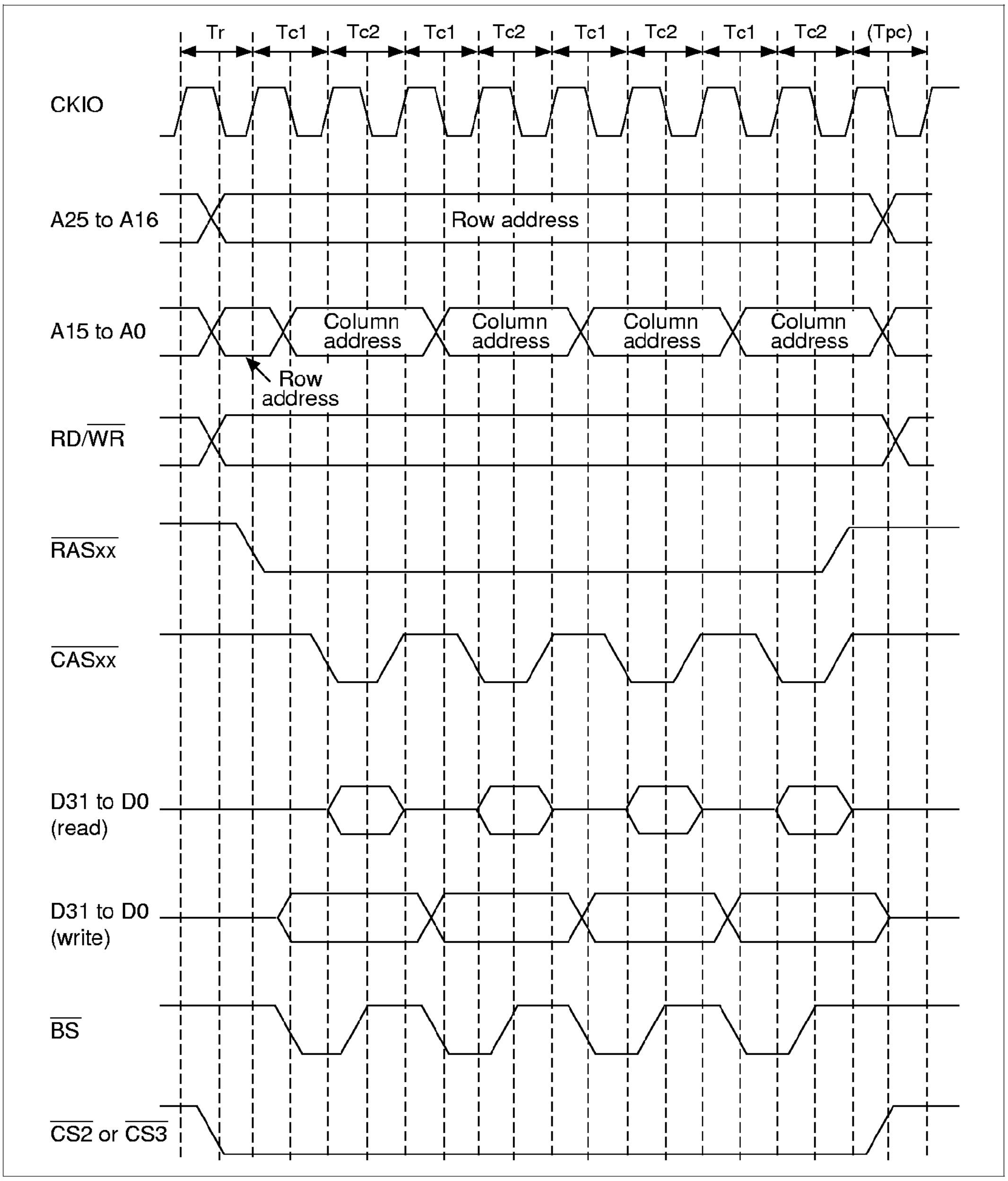


Figure 13.18 DRAM Burst Access Timing

Short-Pitch Burst Access: The bus state controller of the SH7729 can perform short-pitch burst access to DRAM as shown in figure 13.19, when AnW (1–0) and BE are set to 00 and 1, respectively. Tcs is a $\overline{\text{CAS}}$ assert and data access state.

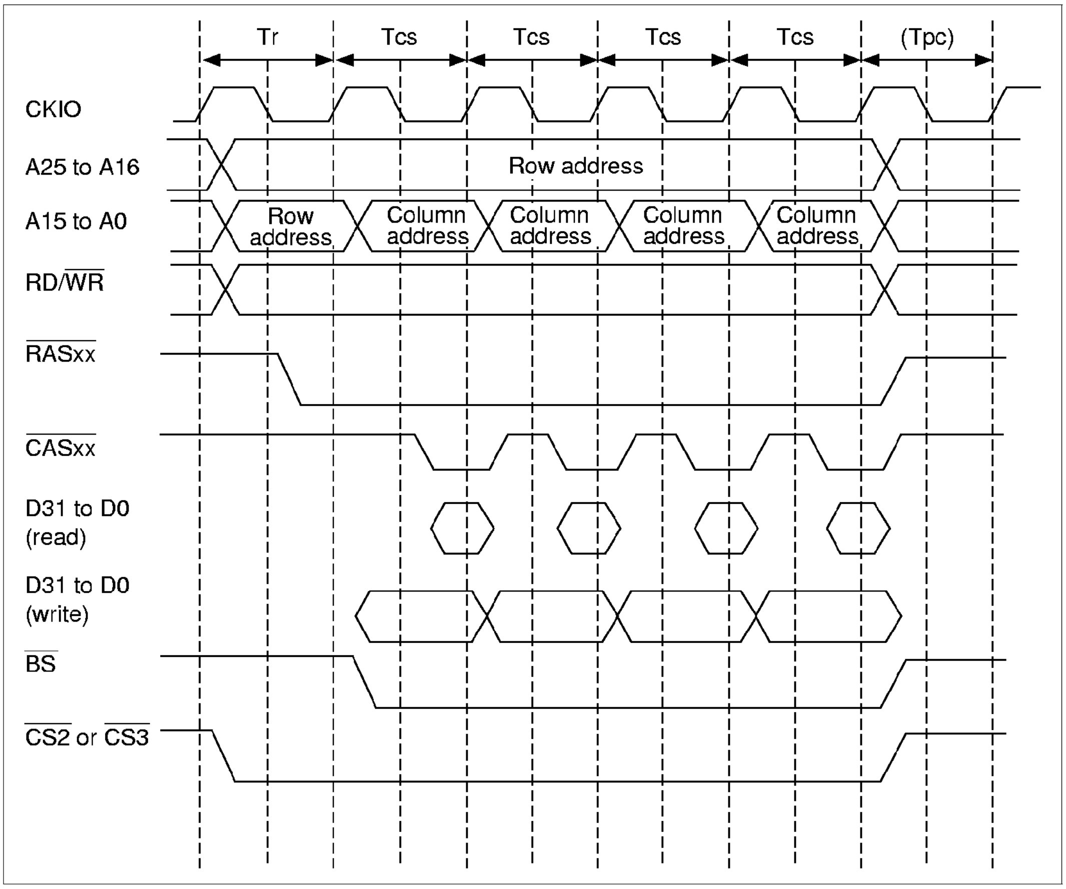


Figure 13.19 DRAM Short-Pitch Burst Access Timing

EDO Mode: With DRAM, in addition to the mode in which data is output to the data bus only while the $\overline{\text{CAS}}$ signal is asserted in a data read cycle, an EDO (extended data out) mode is also provided in which, once the $\overline{\text{CAS}}$ signal is asserted while the $\overline{\text{RAS}}$ signal is asserted, even if the $\overline{\text{CAS}}$ signal is negated, data is output to the data bus until the $\overline{\text{CAS}}$ signal is next asserted. In the SH7729, the EDO mode bit (EDOMODE) in MCR enables selection, for DRAM in area 3 only, of either normal access/burst access using high-speed page mode or EDO mode normal access/burst access. EDO mode normal access is shown in figure 13.20, and burst access in figure 13.21.

In EDO mode, the timing for data output to the data bus in a read cycle is extended as far as the next assertion of the $\overline{\text{CAS}}$ signal, so that the data latch timing is delayed by 1/2 cycle and made the rising edge of the CKIO clock, enabling the DRAM access time to be increased.

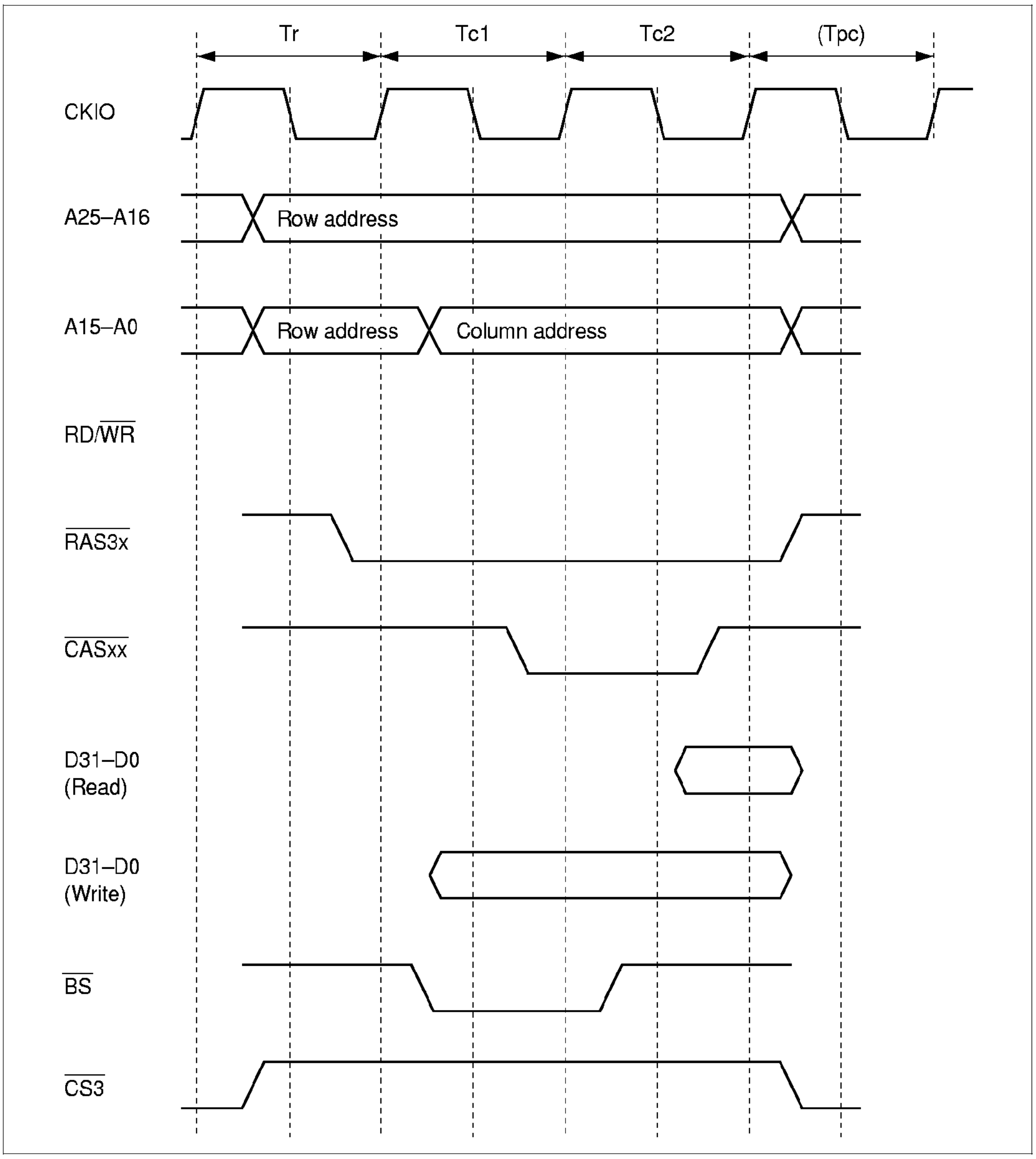


Figure 13.20 Normal Access Timing in DRAM EDO Mode

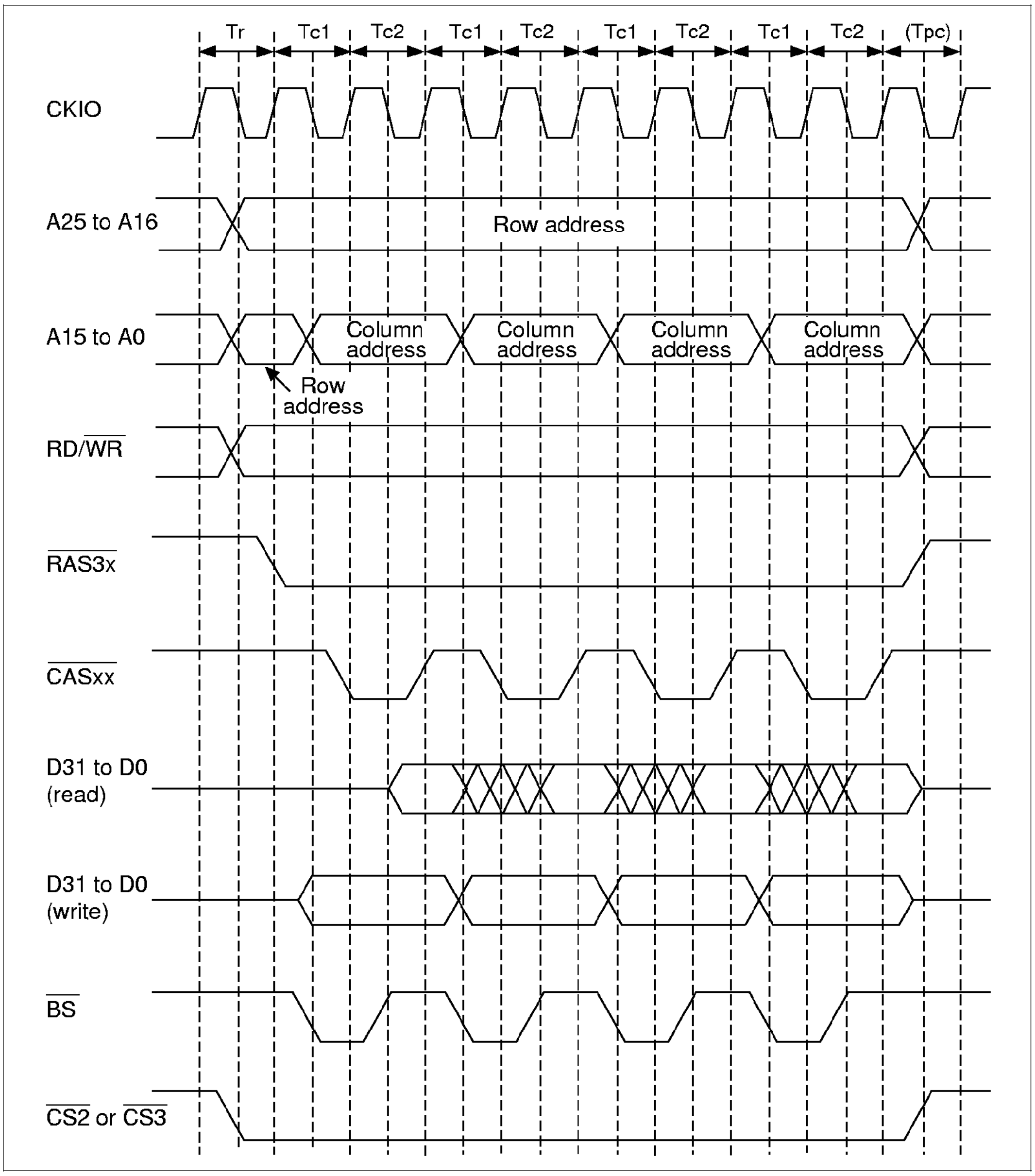


Figure 13.21 Burst Access Timing in DRAM EDO Mode

Refresh Timing: The bus state controller includes a function for controlling DRAM refreshing. Distributed refreshing using a CAS-before-RAS cycle can be performed by clearing the RMODE bit to 0 and setting the RFSH bit to 1 in MCR for DRAM in area 3, or by clearing the RMODE bit to 0 and setting the RFSH bit to 1 in DCR for DRAM in area 2.

When CAS-before-RAS refresh cycles are executed, refreshing is performed at intervals determined by the input clock selected by bits CKS2-0 in RTCSR, and the value set in RTCOR. The value of bits CKS2-0 in RTCOR should be set so as to satisfy the stipulation for the DRAM refresh interval. First make the settings for RTCOR, RTCNT, and the RMODE and RFSH bits in MCR, then make the CKS2-CKS0 setting. When the clock is selected by CKS2-CKS0, 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 the IRQOUT pin goes low. If the SH7729's external bus can be used, CAS-before-RAS refreshing is performed, and if there is no other interrupt request the IRQOUT pin goes high. At the same time, RTCNT is cleared to zero and the count-up is restarted. Figure 13.22 shows the operation of CAS-before-RAS refreshing.

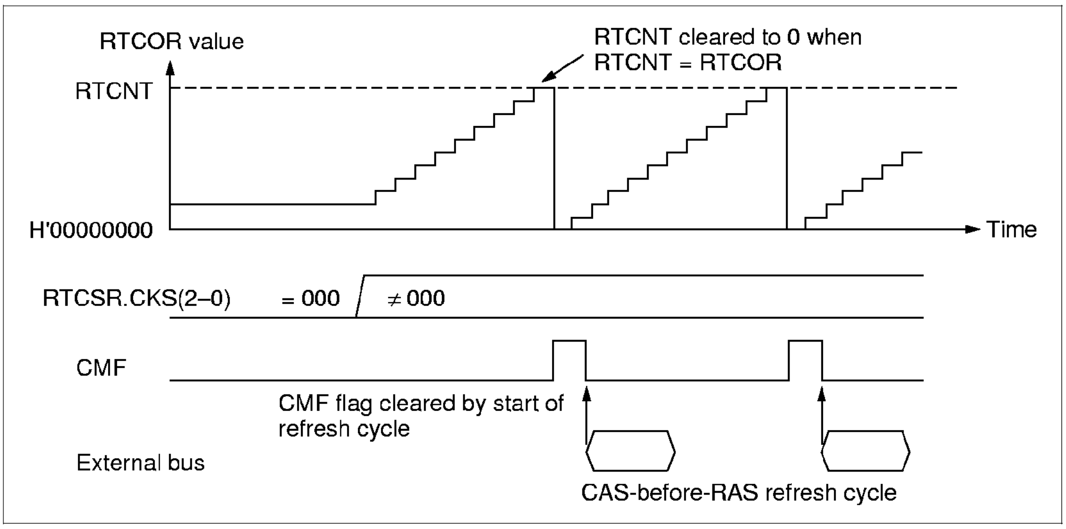

RTCOR value
RTCNT
H'00000000
RTCNT cleared to 0 when RTCNT = RTCOR
Time
RTCSR.CKS(2–0)
= 000
≠ 000
CMF
CMF flag cleared by start of refresh cycle
External bus
CAS-before-RAS refresh cycle

Figure 13.22 CAS-Before-RAS Refresh Operation

Figure 13.23 shows the timing of the CAS-before-RAS refresh cycle.

The number of RAS assert cycles in the refresh cycle is specified by the TRAS bits in MCR and DCR. The specification of the RAS precharge time in the refresh cycle is determined by the setting of the TPC bits in MCR and DCR in the same way as for normal access.

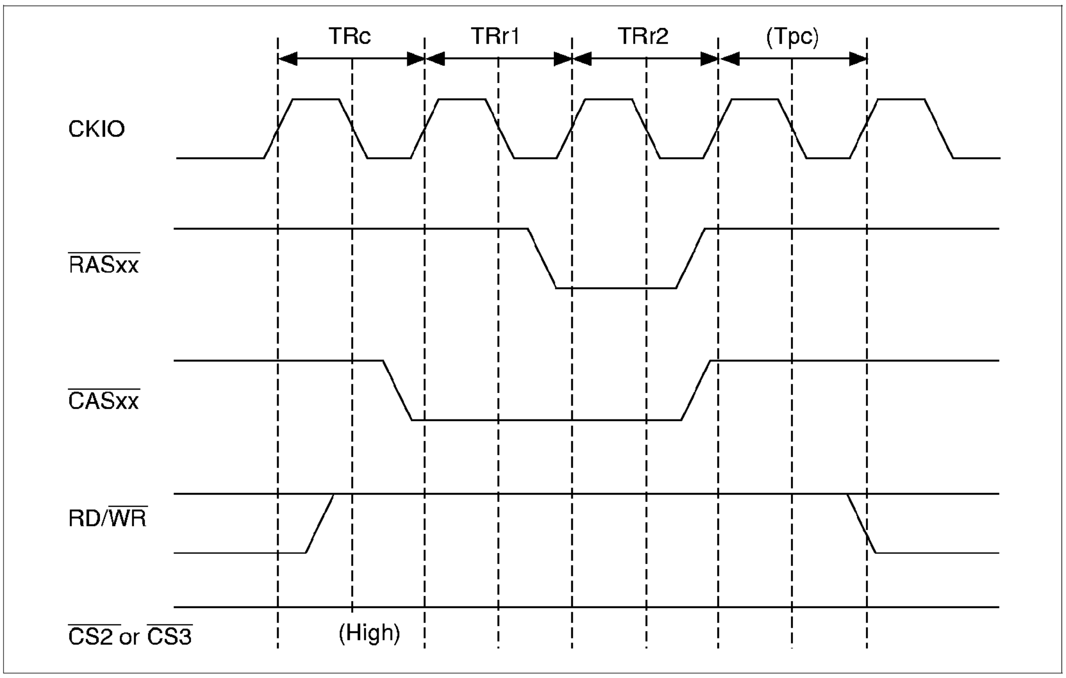


Figure 13.23 DRAM CAS-Before-RAS Refresh Cycle Timing

The self-refreshing supported by the SH7729 is shown in figure 13.24.

After the self-refresh is cleared, the refresh controller immediately generates a refresh request. The RAS precharge time immediately after the end of the self-refreshing can be set by the TPC bits in MCR and DCR.

DRAMs include low-power products (L versions) with a long refresh cycle time (for example, the L version of the HM51W4160AL has a refresh cycle of 1024 cycles/128 ms compared with 1024 cycles/16 ms for the normal version). With these DRAMs, however, the same refresh cycle as for the normal version is requested only in case of refreshing immediately following self-refreshing. To ensure efficient DRAM refreshing, therefore, processing is needed to generate an overflow interrupt and restore the refresh cycle to the proper value, after the necessary CAS-before-RAS refreshing has been performed following self-refreshing of an L-version DRAM, using RFCR and the OVF, OVIE, and LMTS bits in RTCSR. The necessary procedure is as follows.

1. Normally, set the refresh counter count value to the optimum value for the L version (e.g. 1024 cycles/128 ms).

2. When a transition is made to self-refreshing:

   a. Provide an interrupt handler to restore the refresh counter count value to the optimum value for the L version (e.g. 1024 cycles/128 ms) when a refresh counter overflow interrupt is generated.
   b. Reset the refresh counter count value to the requested short cycle (e.g. 1024 cycles/16 ms), set refresh controller overflow interruption, and clear the refresh count register (RFCR) to 0.
   c. Set self-refresh mode.

By using this procedure, the refreshing immediately following a self-refresh will be performed in a short cycle, and when adequate refreshing ends, an interrupt is generated and the setting can be restored to the original refresh cycle.

CAS-before-RAS refreshing is performed in normal operation, in sleep mode, and in case of a manual reset.

Self-refreshing is performed in normal operation, in sleep mode, in standby mode, and in case of a manual reset.

When the bus has been released in response to a bus arbitration request, or when a transition is made to standby mode, signals generally become high-impedance, but whether the $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ signals become high-impedance or continue to be output can be controlled by the HIZCNT bit in BCR1. This enables the DRAM to be kept in the self-refreshing state.

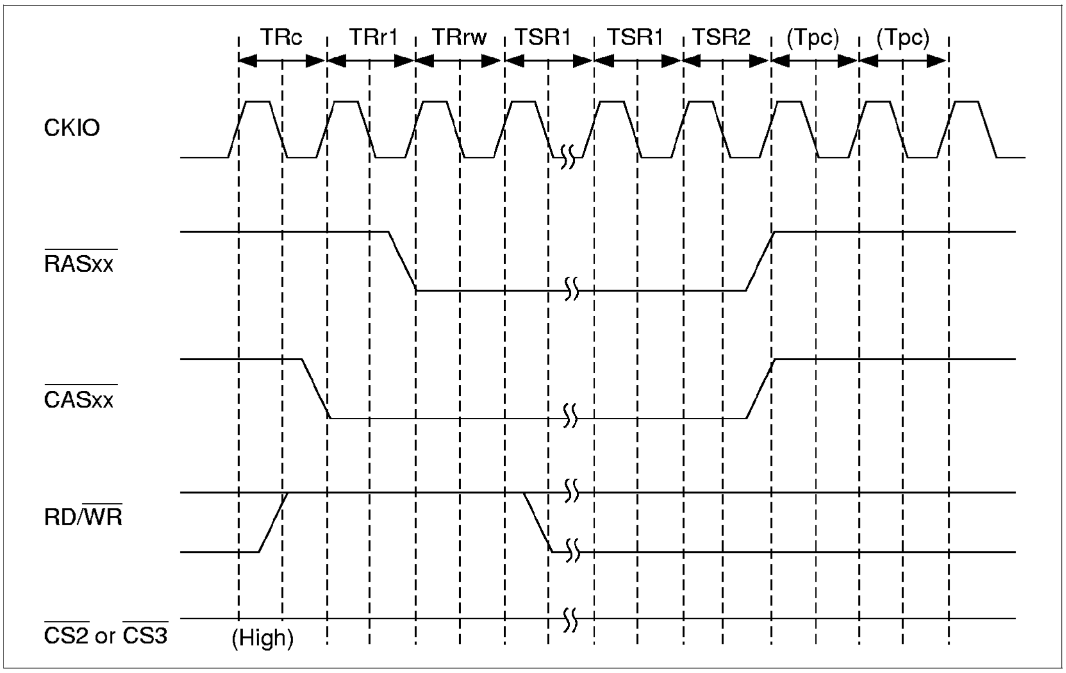


Figure 13.24 DRAM Self-Refresh Cycle Timing

Power-On Sequence: Regarding use of DRAM after powering on, it is requested that a wait time (at least 100 µs or 200 µs) during which no access can be performed be provided, followed by the prescribed number (usually 8) or more dummy CAS-before-RAS refresh cycles. As the bus state controller does not perform any special operations for a power-on reset, the necessary power-on sequence must be carried out by the initialization program executed after a power-on reset.

13.3.5 Synchronous DRAM Interface

Synchronous DRAM Direct Connection: Since SDRAM can be selected by the CS signal, physical space areas 2 and 3 can be connected using RAS and other control signals in common. If the memory type bits (DRAMTP2–0) in BCR1 are set to 010, area 2 is ordinary memory space and area 3 is SDRAM space; if set to 011, areas 2 and 3 are both SDRAM space.

With the SH7729, burst length 1 burst read/single write mode is supported as the SDRAM operating mode. A data bus width of 16 or 32 bits can be selected. The burst enable bit (BE) in MCR is ignored, a 16-bit burst transfer is performed in a cache fill/write-back cycle, and only one access is performed in a write-through area write or a non-cacheable area read/write.

The control signals for direct connection of SDRAM are RAS3L, RAS3U, CASL, CASU, RD/WR, CS2 or CS3, DQMUU, DQMUL, DQMLU, DQMLL, and CKE. All the signals other than CS2 and CS3 are common to all areas, and signals other than CKE are valid and fetched to the SDRAM only when CS2 or CS3 is asserted. SDRAM can therefore be connected in parallel to a number of areas. CKE is negated (low) only when self-refreshing is performed, and is always asserted (high) at other times.

However, which of the RAS3L, RAS3U, CASL, and CASU signals are output is determined by whether the address is in the upper or lower 32 Mbytes in each area. When the address is in upper 32 Mbytes, RAS3U and CASU are output; when in lower 32 Mbytes, RAS3L and CASL are output. In the refresh cycle and mode-register write cycle, RAS3U and RAS3L or CASU and CASL are output.

Commands for SDRAM are specified by RAS3L, RAS3U, CASL, CASU, RD/WR, and special address signals. The commands are NOP, auto-refresh (REF), self-refresh (SELF), precharge all banks (PALL), row address strobe bank active (ACTV), read (READ), read with precharge (READA), write (WRIT), write with precharge (WRITA), and mode register write (MRS).

Byte specification is performed by DQMUU, DQMUL, DQMLU, and DQMLL. A read/write is performed for the byte for which the corresponding DQM is low. In big-endian mode, DQMUU specifies an access to address 4n, and DQMLL specifies an access to address 4n + 3. In little-endian mode, DQMUU specifies an access to address 4n + 3, and DQMLL specifies an access to address 4n.

Figures 13.25 and 13.26 show examples of the connection of two 256k × 16-bit SDRAMs and one 1M × 16-bit SDRAM, respectively.

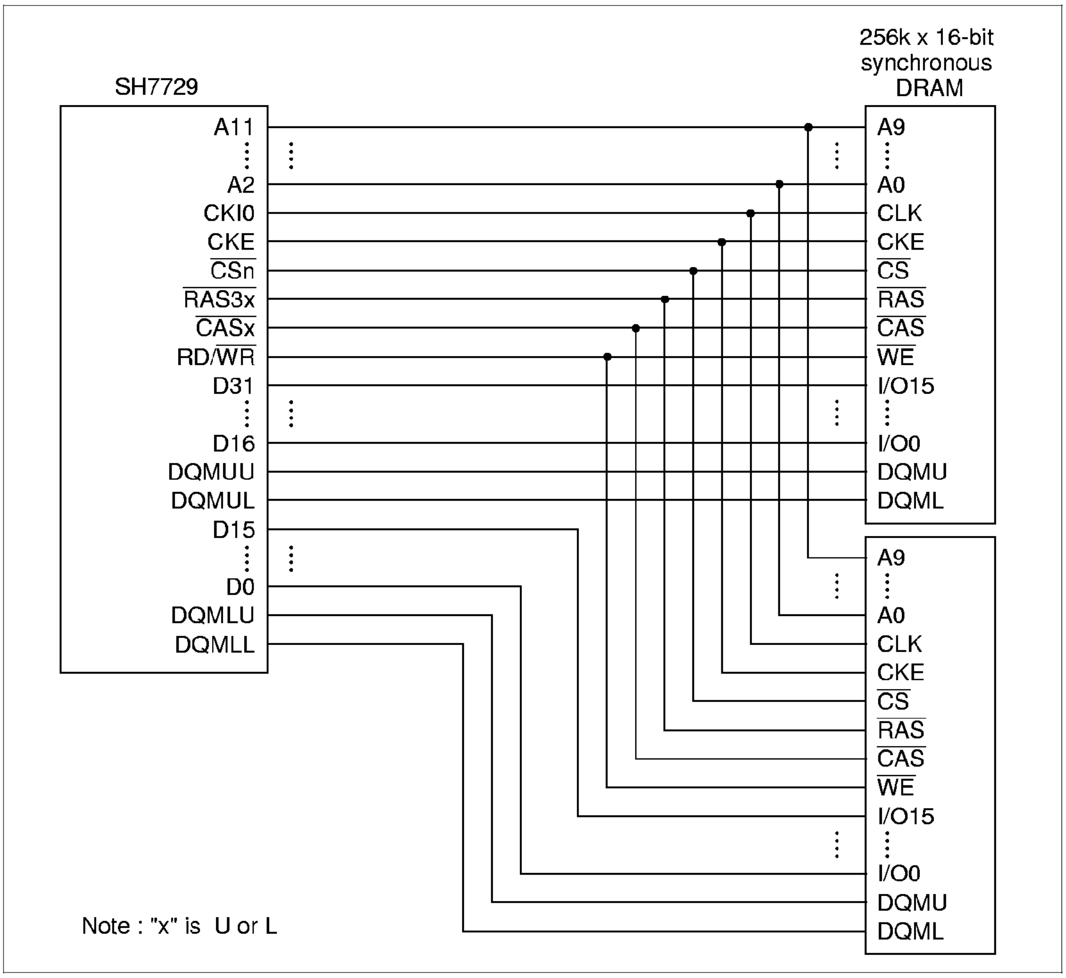


Figure 13.25 Example of SDRAM Connection (32-Bit Bus Width)

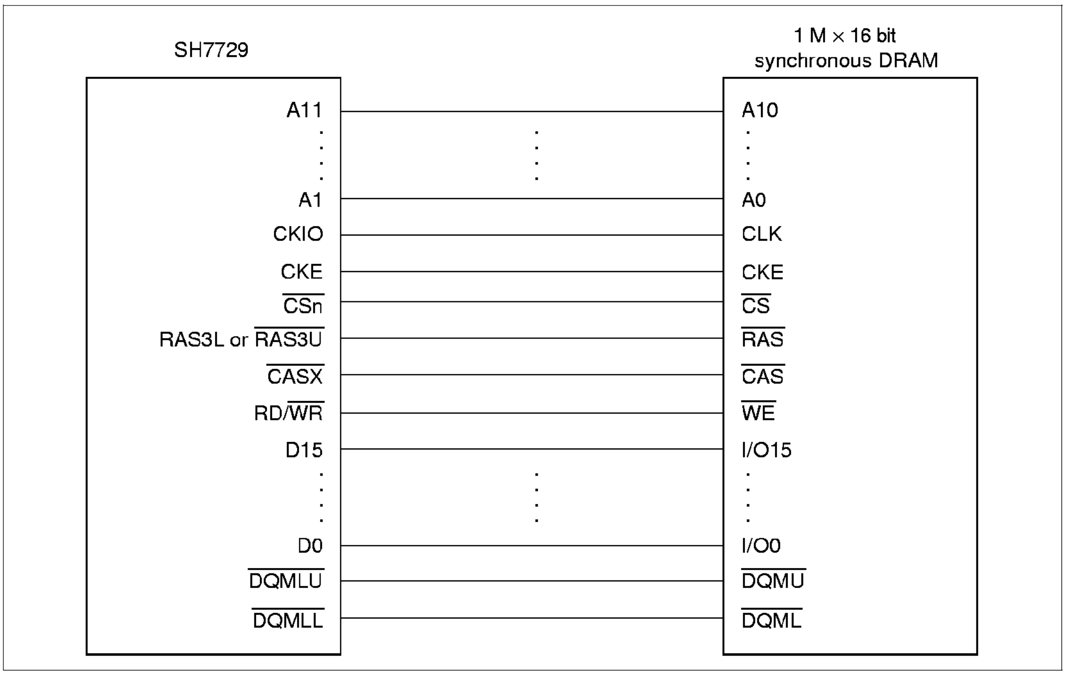


Figure 13.26 Example of SDRAM Connection (16-Bit Bus Width)

Address Multiplexing: SDRAM can be connected without external multiplexing circuitry in accordance with the address multiplex specification bits AMX2-AMX0 in MCR. Table 13.14 shows the relationship between the address multiplex specification bits and the bits output at the address pins.

A25–A16 and A0 are not multiplexed; the original values are always output at these pins.

When A0, the LSB of the SDRAM address, is connected to the SH7729, it performs longword address specification. Connection should therefore be made in the following order: with a 32-bit bus width, connect pin A0 of the SDRAM to pin A2 of the SH7729, then connect pin A1 to pin A3; with a 16-bit bus width, connect pin A0 of the SDRAM to pin A1 of the SH7729, then connect pin A2 to pin A2.

Table 13.14 Relationship between Bus Width, AMX, and Address Multiplex Output

Setting				External Address Pins							
AMX2	**AMX1**	**AMX0**	**Output Timing**	**A1 to A8**	**A9**	**A10**	**A11**	**A12**	**A13**	**A14**	**A15**
1	0	0	Column address	A1 to A8	A9	A10	A11	L/H*1	A13	A22*2	A23*2
			Row address	A9 to A16	A17	A18	A19	A20	A21	A22*2	A23*2
		1	Column address	A1 to A8	A9	A10	A11	L/H*1	A13	A23*2	A24*2
			Row address	A10 to A17	A18	A19	A20	A21	A22	A23*2	A24*2
	1	1	Column address	A1 to A8	A9	A20	A11	L/H*1	A21*2	A22*2	A15
			Row address	A9 to A16	A17	A18	A19	A20	A21*2	A22*2	A23
0	0	0	Column address	A1 to A8	A9	A10	A11	L/H*1	A21*2	A14	A15
			Row address	A9 to A16	A17	A18	A19	A20	A21*2	A22	A23
		1	Column address	A1 to A8	A9	A10	A11	L/H*1	A22*2	A14	A15
			Row address	A10 to A17	A18	A19	A20	A21	A22*2	A23	A24
	1	0	Column address	A1 to A8	A9	A20	A11	L/H*1	A23*2	A14	A15
			Row address	A11 to A18	A19	A20	A21	A22	A23*2	A24	A25
		1	Column address	A1 to A8	A9	L/H*1	A19*2	A12	A13	A14	A15
			Row address	A9 to A16	A17	A18	A19*2	A20	A21	A22	A23

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 13.15 Example of Correspondence between SH7729 and SDRAM Address Pins (AMX (2-0) = 011 (32-Bit Bus Width))

SH7729 Address Pin			SDRAM Address Pin	
	RAS Cycle	CAS Cycle		Function
A11	A19	A19	A9	BANK select bank address
A10	A18	L/H	A8	Address precharge setting
A9	A17	A9	A7	Address
A8	A16	A8	A6	
A7	A15	A7	A5	
A6	A14	A6	A4	
A5	A13	A5	A3	
A4	A12	A4	A2	
A3	A11	A3	A1	
A2	A10	A2	A0	
A1	A9	A1	Not used	
A0	A0	A0	Not used	

Burst Read: In the following example it is assumed that four 2M × 8-bit SDRAMs are connected and a 32-bit data width is used, and the burst length is 1. Following the Tr cycle in which ACTV command output is performed, a READ command is issued in the Tc1, Tc2, and Tc3 cycles, and a READA command in the Tc4 cycle, and the read data is accepted on the rising edge of the external command clock (CKIO) from cycle Td1 to cycle Td4. The Tpc cycle is used to wait for completion of auto-precharge based on the READA command inside the SDRAM; no new access command can be issued to the same bank during this cycle, but access to SDRAM for another area is possible. In the SH7729, the number of Tpc cycles is determined by the TPC bit specification in MCR, and commands cannot be issued for the same SDRAM during this interval.

The example in figure 13.26 shows the basic timing. To connect low-speed SDRAM, the cycle can be extended by setting WCR2 and MCR bits. The number of cycles from the ACTV command output cycle, Tr, to the READ command output cycle, Tc1, can be specified by the RCD bit in MCR, with a values of 0 to 3 specifying 1 to 4 cycles, respectively. In case of 2 or more cycles, a Trw cycle, in which an NOP command is issued for the SDRAM, is inserted between the Tr cycle and the Tc cycle. The number of cycles from READ and READA command output cycles Tc1-Tc4 to the first read data latch cycle, Td1, can be specified as 1 to 3 cycles independently for areas 2 and 3 by means of A2W1 and A2W0 or A3W1 and A3W0 in WCR2. This number of cycles corresponds to the number of SDRAM CAS latency cycles.

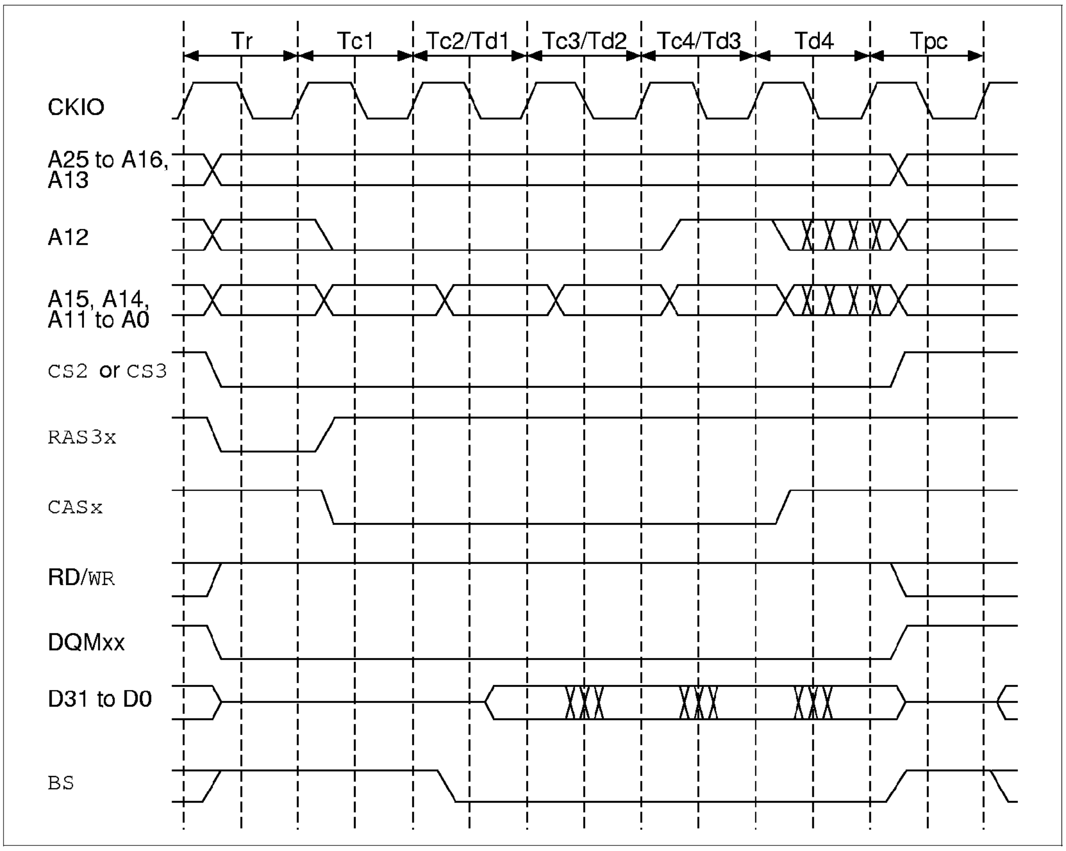


Figure 13.27 Basic Timing for SDRAM Burst Read

Figure 13.28 shows the burst read timing when RCD is set to 1, A3W1 and A3W0 are set to 10, and TPC is set to 1.

The BS cycle, which is asserted for one cycle at the start of a bus cycle for normal access space, is asserted in each of cycles Td1–Td4 in a SDRAM cycle. When a burst read is performed, the address is updated each time $\overline{\text{CAS}}$ is asserted. As the unit of burst transfer is 16 bytes, address updating is performed for A3 and A2 only. The order of access is as follows: in a fill operation in the event of a cache miss, the missed data is read first, then 16-byte boundary data including the missed data is read in wraparound mode.

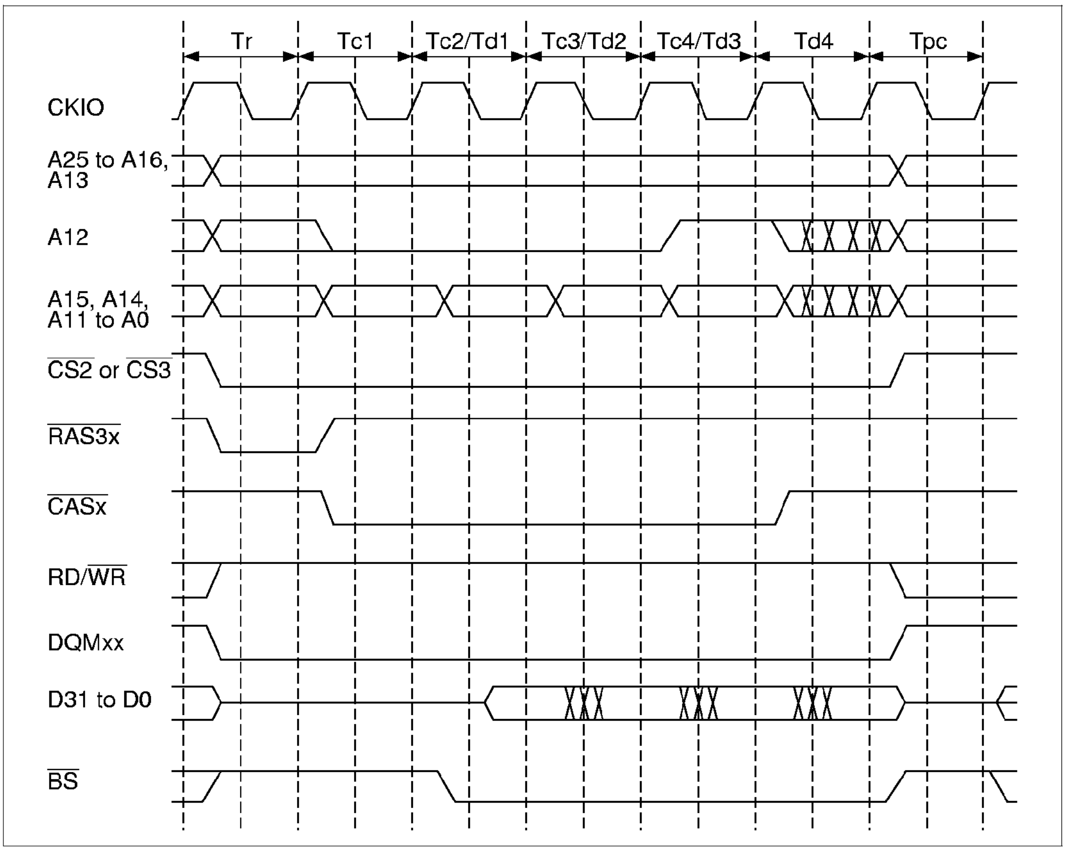


Figure 13.28 SDRAM Burst Read Wait Specification Timing

Single Read: Figure 13.29 shows the timing when a single address read is performed. As the burst length is set to 1 in 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.

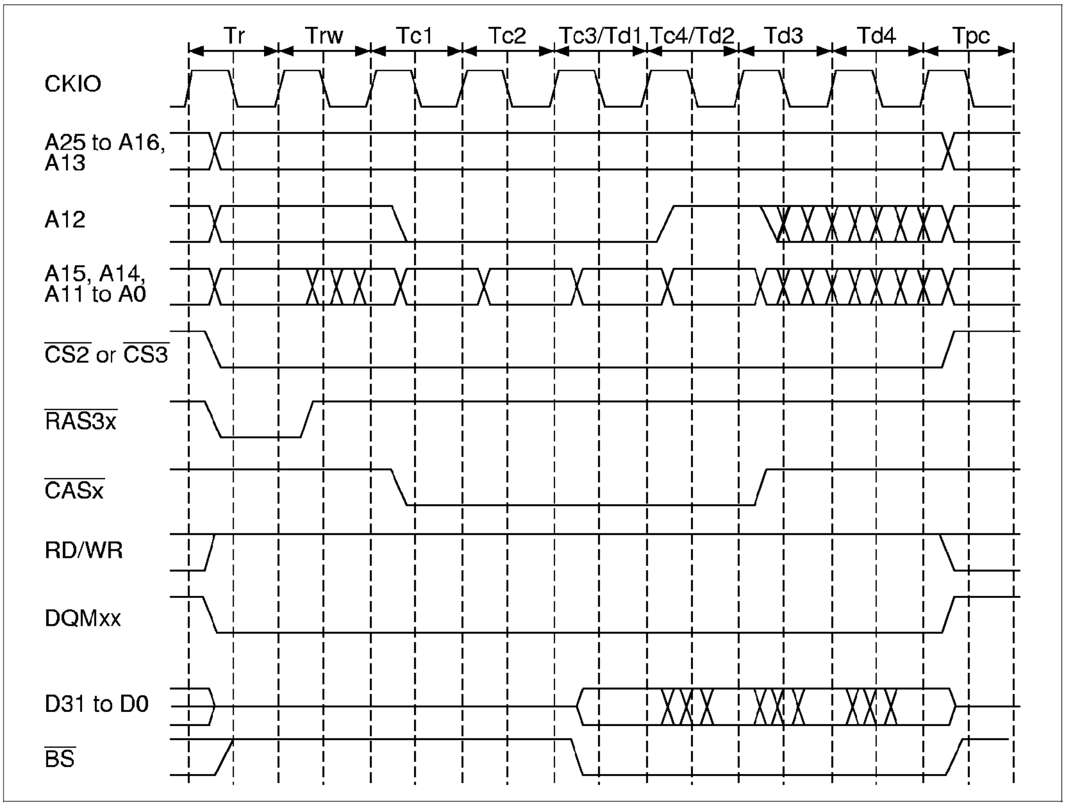


Figure 13.29 Basic Timing for SDRAM Single Read

Burst Write: The timing chart for a burst write is shown in figure 13.30. In the SH7729, a burst write occurs only in the event of cache write-back. In a burst write operation, following the Tr cycle in which ACTV command output is performed, a WRIT command is issued in the Tc1, Tc2, and Tc3 cycles, and a WRITA command that performs auto-precharge is issued in the Tc4 cycle. In the write cycle, the write data is output at the same time as the write command. In case of the write with auto-precharge command, precharging of the relevant bank is performed in the SDRAM after completion of the write command, and therefore no command can be issued for the same bank until precharging is completed. Consequently, in addition to the precharge wait cycle, Tpc, used in a read access, cycle Trwl is also added as a wait interval until precharging is started following the write command. Issuance of a new command for the same bank is postponed during this interval. The number of Trwl cycles can be specified by the TRWL bit in MCR.

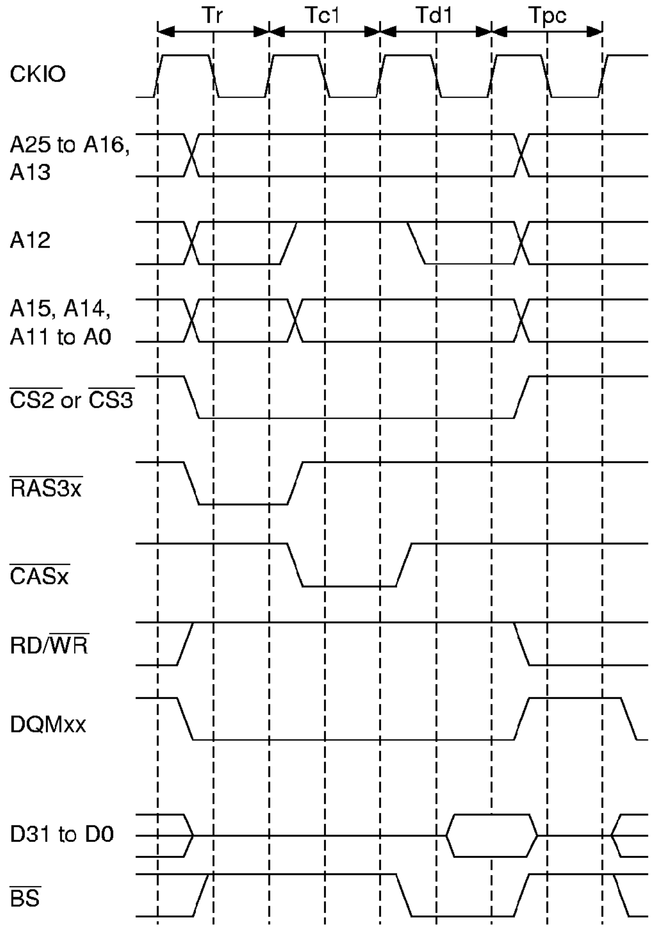


Figure 13.30 Basic Timing for SDRAM Burst Write

Single Write: The basic timing chart for write access is shown in figure 13.31. In a single write operation, following the Tr cycle in which ACTV command output is performed, a WRITA command that performs auto-precharge is issued in the Tc1 cycle. In the write cycle, the write data is output at the same time as the write command. In case of the write with auto-precharge command, precharging of the relevant bank is performed in the SDRAM after completion of the write command, and therefore no command can be issued for the same bank until precharging is completed. Consequently, in addition to the precharge wait cycle, Tpc, used in a read access, cycle Trwl is also added as a wait interval until precharging is started following the write command. Issuance of a new command for the same bank is postponed during this interval. The number of Trwl cycles can be specified by the TRWL bit in MCR.

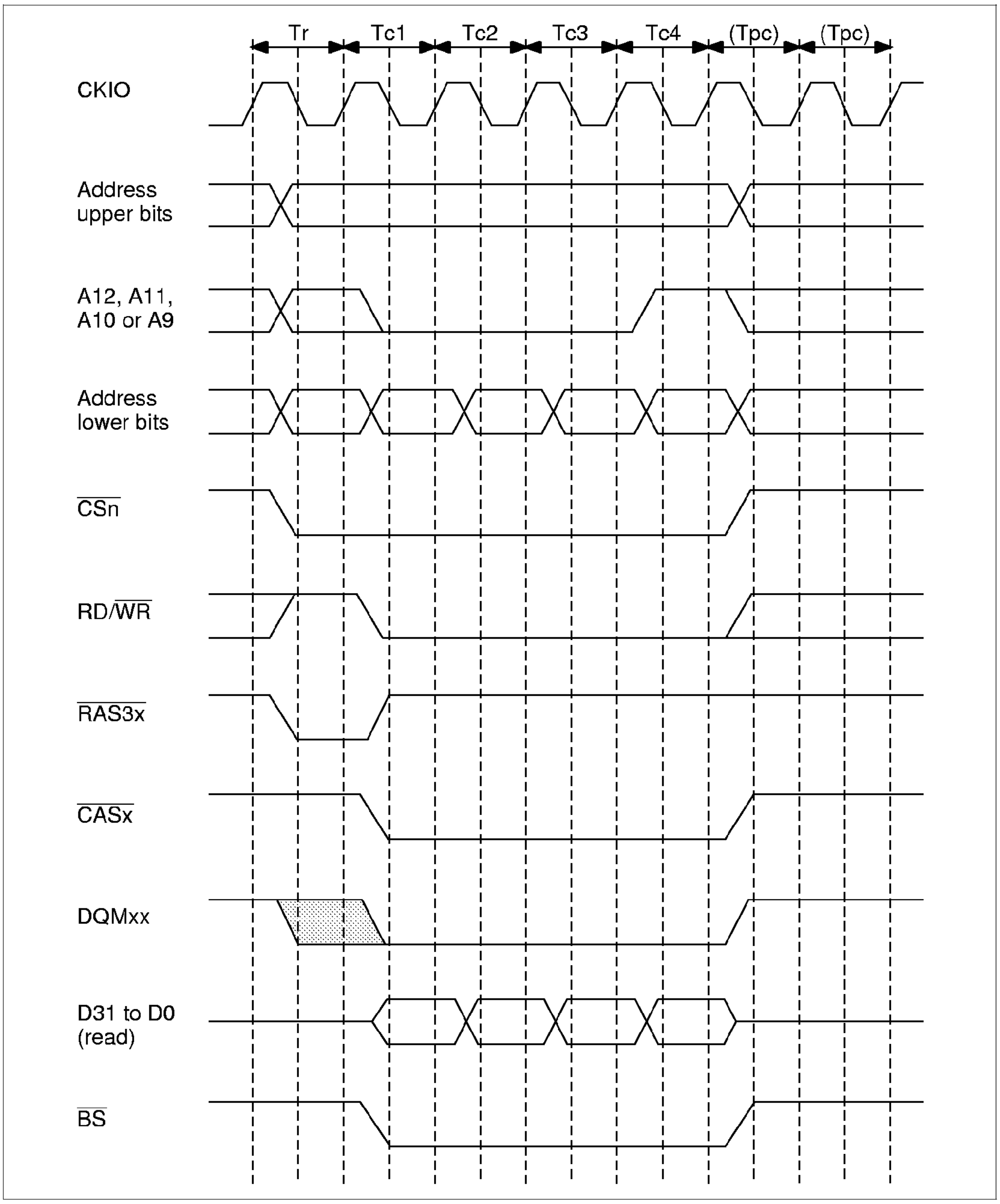


Figure 13.31 Basic Timing for SDRAM Single Write

Bank Active: The SDRAM bank function is used to support high-speed accesses to the same row address. When the RASD bit in MCR is 1, read/write command accesses are performed using commands without auto-precharge (READ, WRIT). In this case, 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, in the same way as in the RAS down state in DRAM fast page mode. As SDRAM is internally divided into two or four 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.

In a write, when auto-precharge is performed, a command cannot be issued for a period of Trwl + Tpc 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 + Tpc cycles for each write. The number of cycles between issuance of the precharge command and the row address strobe command is determined by the TPC bit in MCR.

Whether faster execution speed is achieved by use of bank active mode or by use of basic access is determined by the probability of accessing the same row address (P1), and the average number of cycles from completion of one access to the next access (Ta). If Ta is greater than Tpc, the delay due to the precharge wait when writing is imperceptible. In this case, the access speed for bank active mode and basic access is determined by the number of cycles from the start of access to issuance of the read/write command: (Tpc + Trcd) × (1 – P1) and Trcd, respectively.

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. In this way, it is possible to observe the restrictions on the maximum active state time for each bank. If auto-refresh is not used, measures must be taken in the program to ensure that the banks do not remain active for longer than the prescribed time.

A burst read cycle without auto-precharge is shown in figure 13.32, a burst read cycle for the same row address in figure 13.33, and a burst read cycle for different row addresses in figure 13.34. Similarly, a burst write cycle without auto-precharge is shown in figure 13.35, a burst write cycle for the same row address in figure 13.36, and a burst write cycle for different row addresses in figure 13.37.

A Tnop cycle, in which no operation is performed, is inserted before the Tc cycle in which the READ command is issued in figure 13.33, but when SDRAM is read, there is a two-cycle latency for the DQMxx signal that performs the byte specification. If the Tc cycle were performed immediately, without inserting a Tnop cycle, it would not be possible to perform the DQMxx signal specification for Td1 cycle data output. This is the reason for inserting the Tnop cycle. If the CAS latency is two cycles or longer, Tnop cycle insertion is not performed, since the timing requirements will be met even if the DQMxx signal is set after the Tc cycle.

When bank active mode is set, if only accesses to the respective banks in the area 3 space are considered, as long as accesses to the same row address continue, the operation starts with the cycle in figure 13.32 or 13.35, followed by repetition of the cycle in figure 13.33 or 13.36. An access to a different area 3 space 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 13.34 or 13.37 is executed instead of that in figure 13.33 or 13.36. 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.

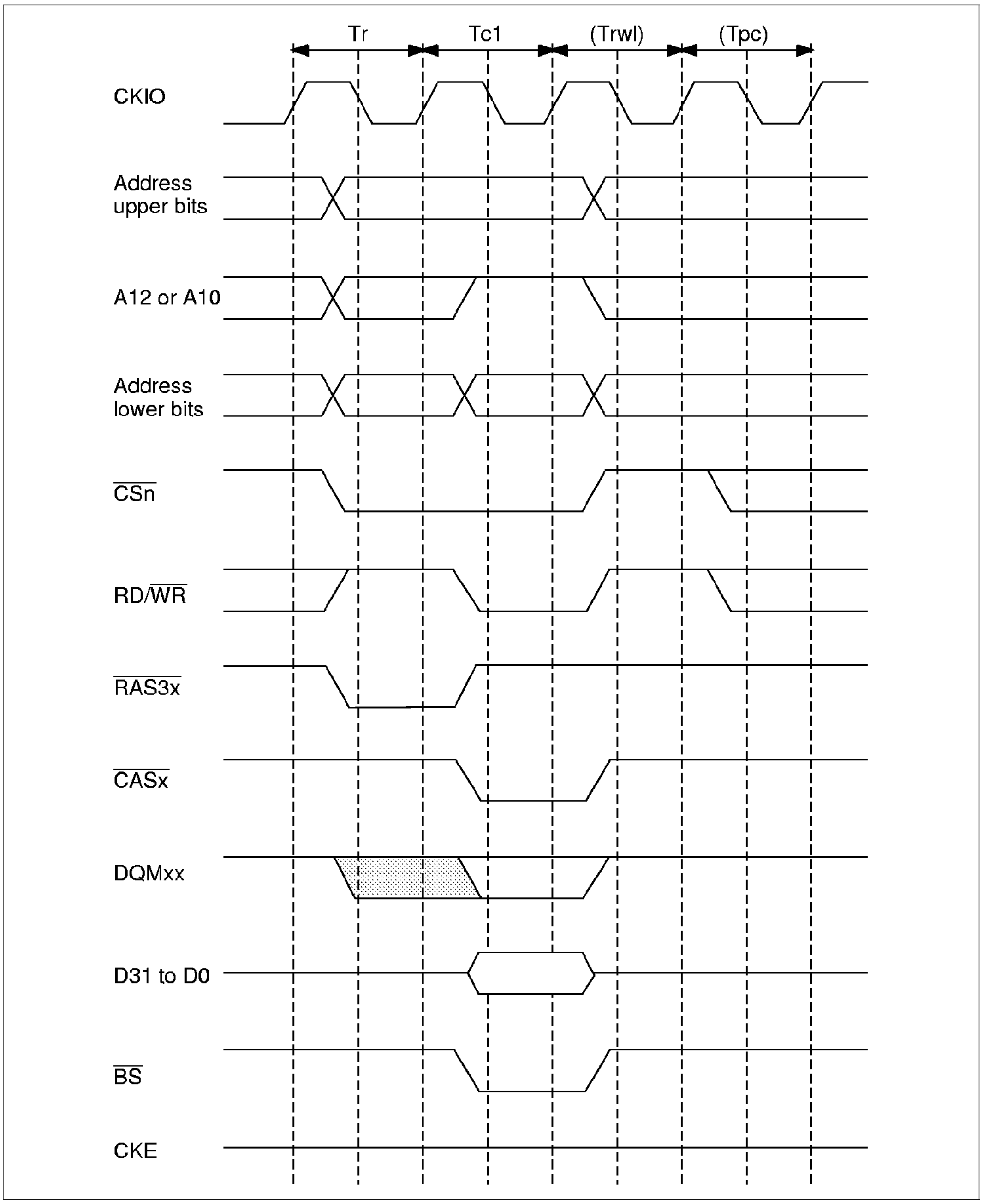


Figure 13.32 Burst Read Timing (No Precharge)

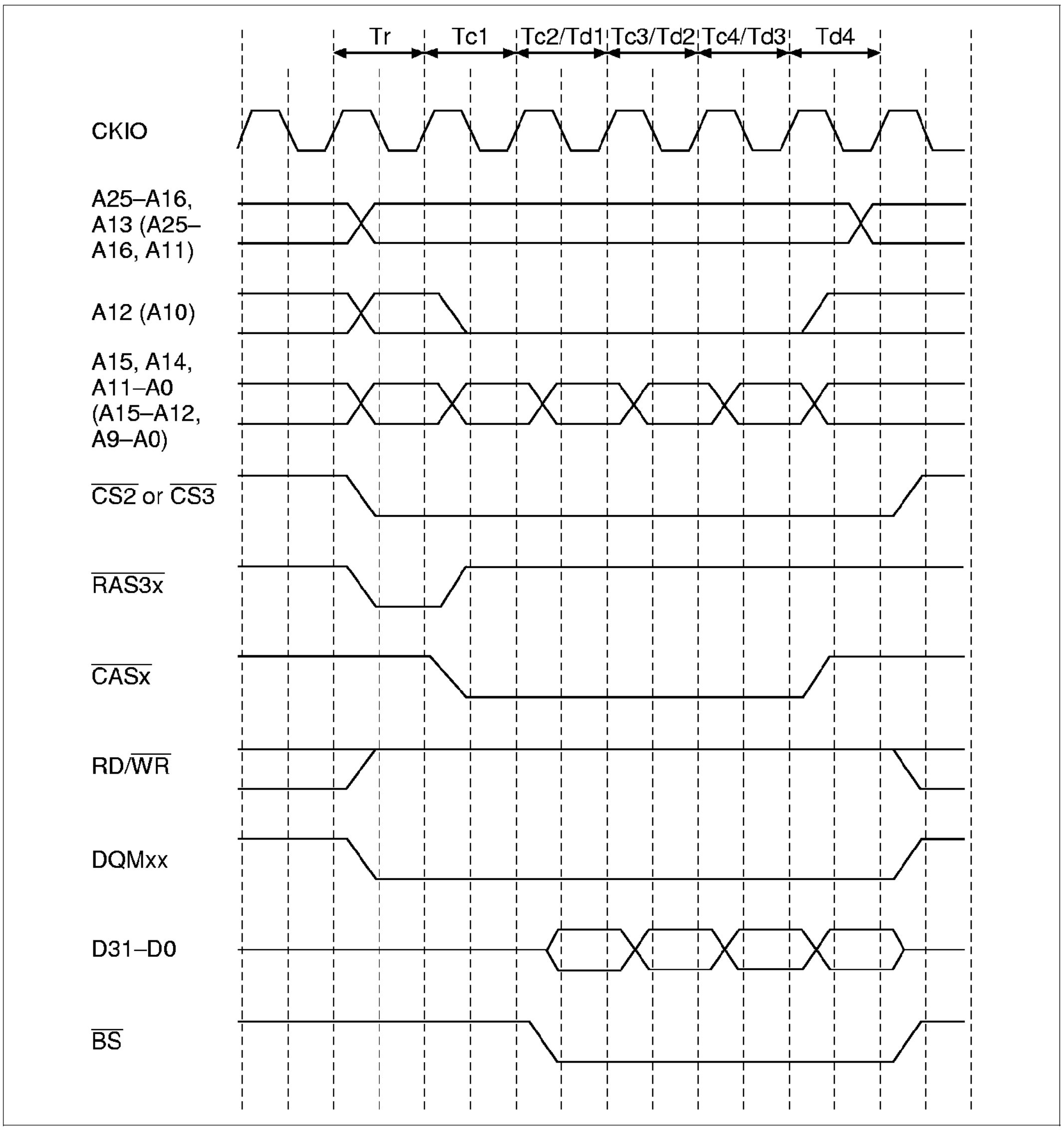

Tr
Tc1
Tc2/Td1
Tc3/Td2
Tc4/Td3
Td4
CKIO
A25–A16, A13 (A25–A16, A11)
A12 (A10)
A15, A14, A11–A0 (A15–A12, A9–A0)
$\overline{\text{CS2}}$ or $\overline{\text{CS3}}$
$\overline{\text{RAS3x}}$
$\overline{\text{CASx}}$
RD/$\overline{\text{WR}}$
DQMxx
D31–D0
$\overline{\text{BS}}$

Figure 13.33 Burst Read Timing (Same Row Address)

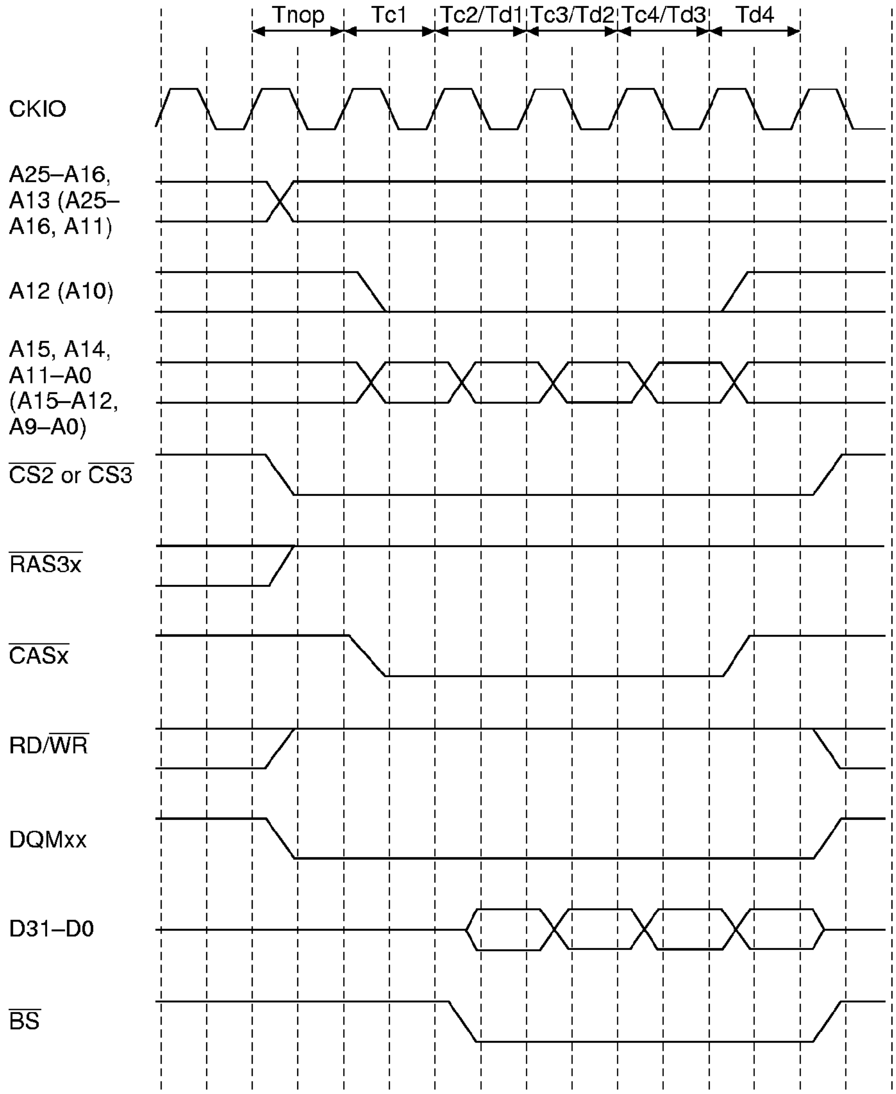


Figure 13.34 Burst Read Timing (Different Row Addresses)

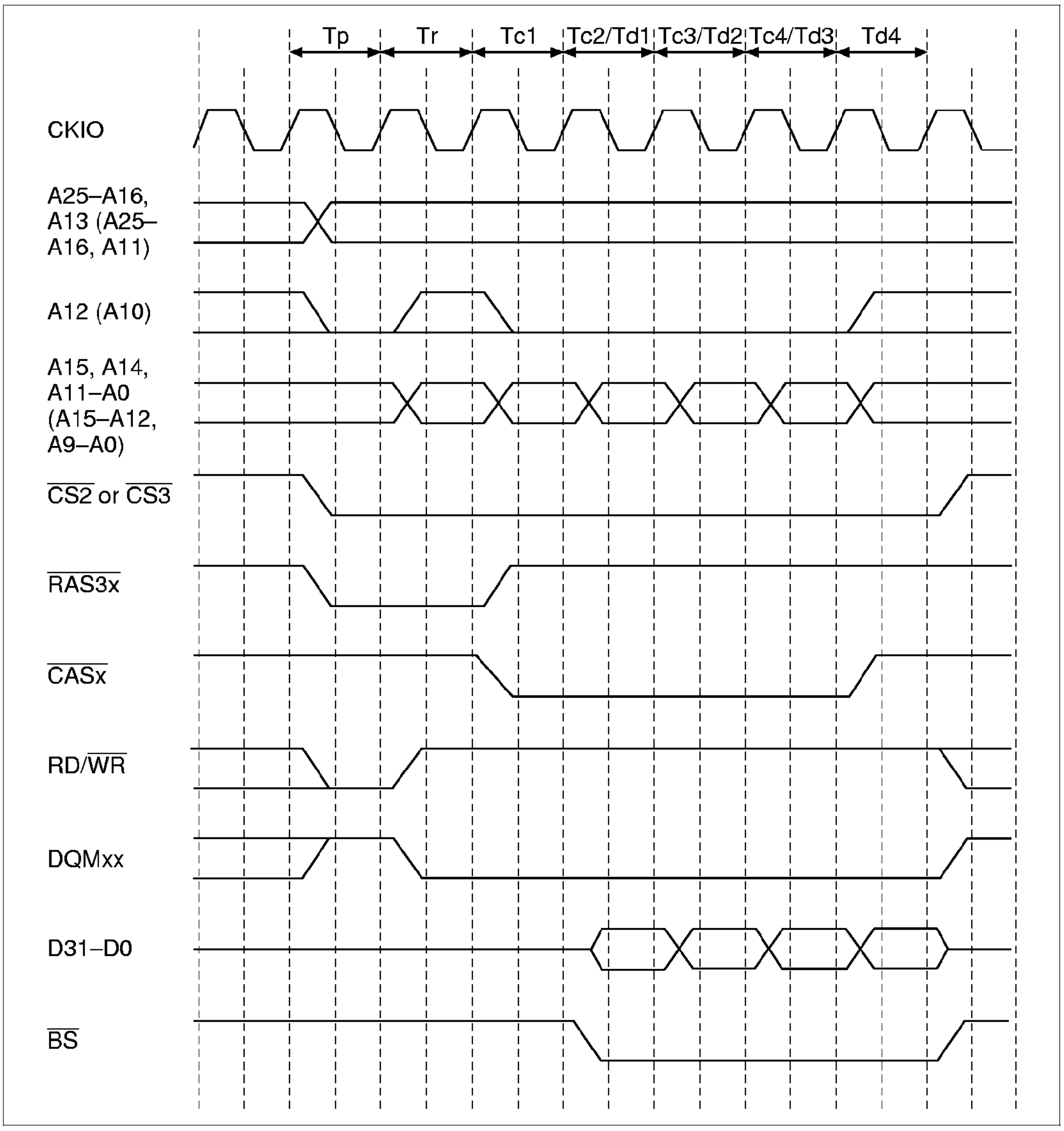


Figure 13.35 Burst Write Timing (No Precharge)

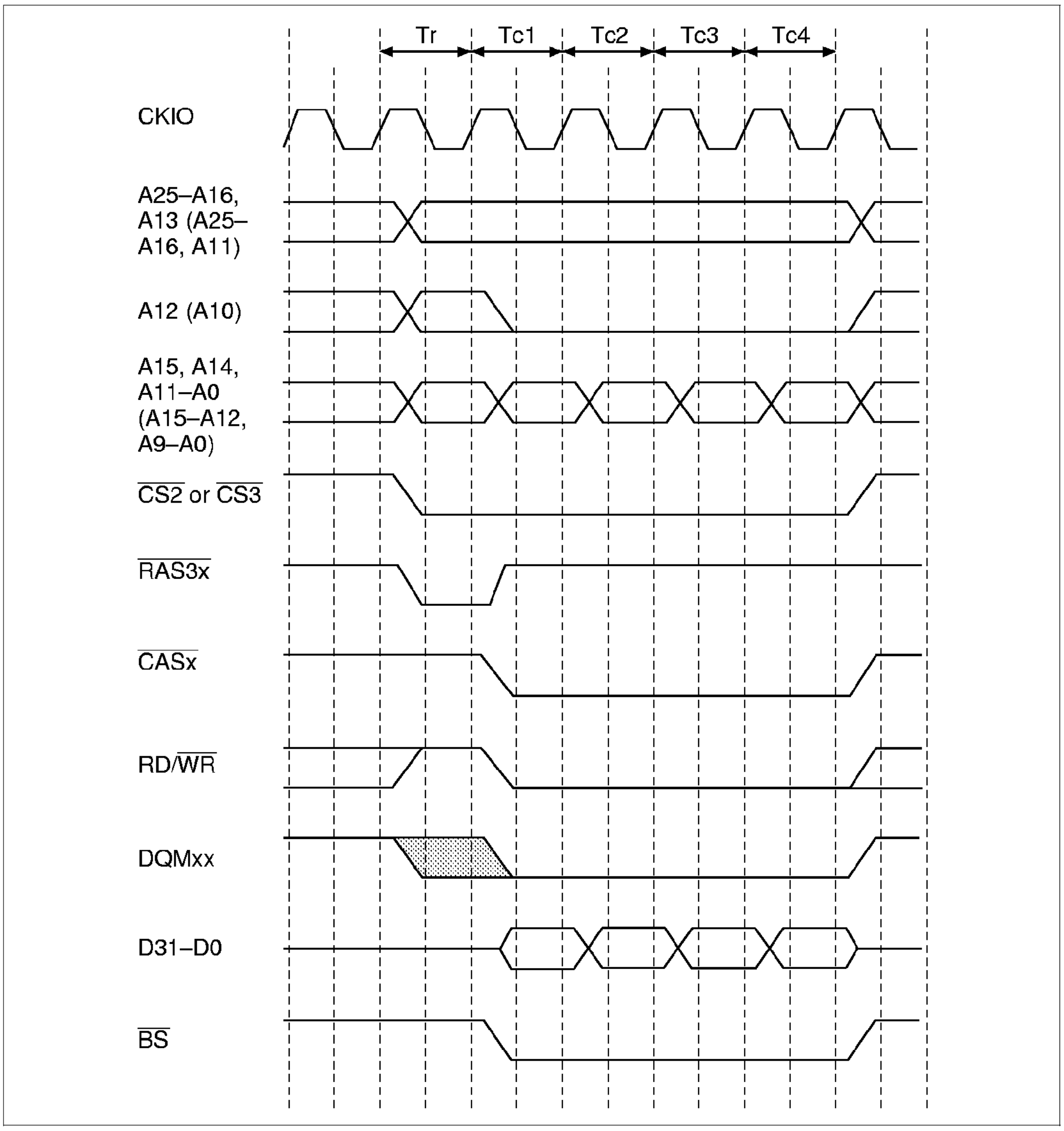


Figure 13.36 Burst Write Timing (Same Row Address)

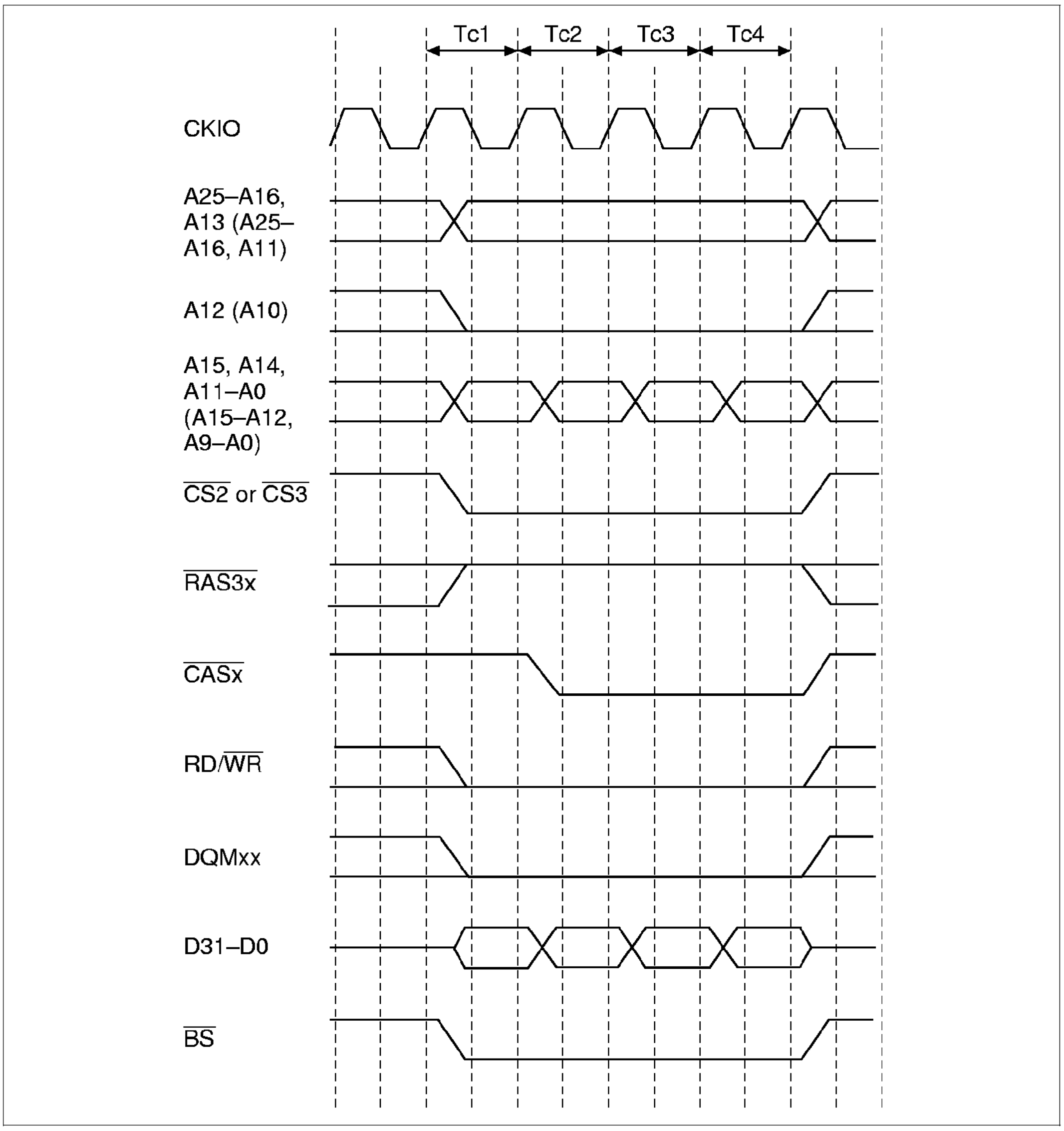


Figure 13.37 Burst Write Timing (Different Row Addresses)

Refreshing: The bus state controller is provided with a function for controlling SDRAM refreshing. Auto-refreshing can be performed by clearing the RMODE bit to 0 and setting the RFSH bit to 1 in MCR. 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 CKS2-0 in RTCSR, and the value set in RTCOR. The value of bits CKS2-0 in RTCOR 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 MCR, then make the CKS2-CKS0 setting. When the clock is selected by CKS2-CKS0, 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. At the same time, RTCNT is cleared to zero and the count-up is restarted. Figure 13.38 shows the auto-refresh cycle timing.

All-bank precharging is performed in the Tp cycle, then an REF command is issued in the TRr cycle following the interval specified by the TPC bits in MCR. After the TRr cycle, new command output cannot be performed for the duration of the number of cycles specified by the TRAS bits in MCR plus the number of cycles specified by the TPC bits in MCR. The TRAS and TPC bits must be set so as to satisfy the SDRAM refresh cycle time stipulation (active/active command delay time).

Auto-refreshing is performed in normal operation, in sleep mode, and in case of a manual reset.

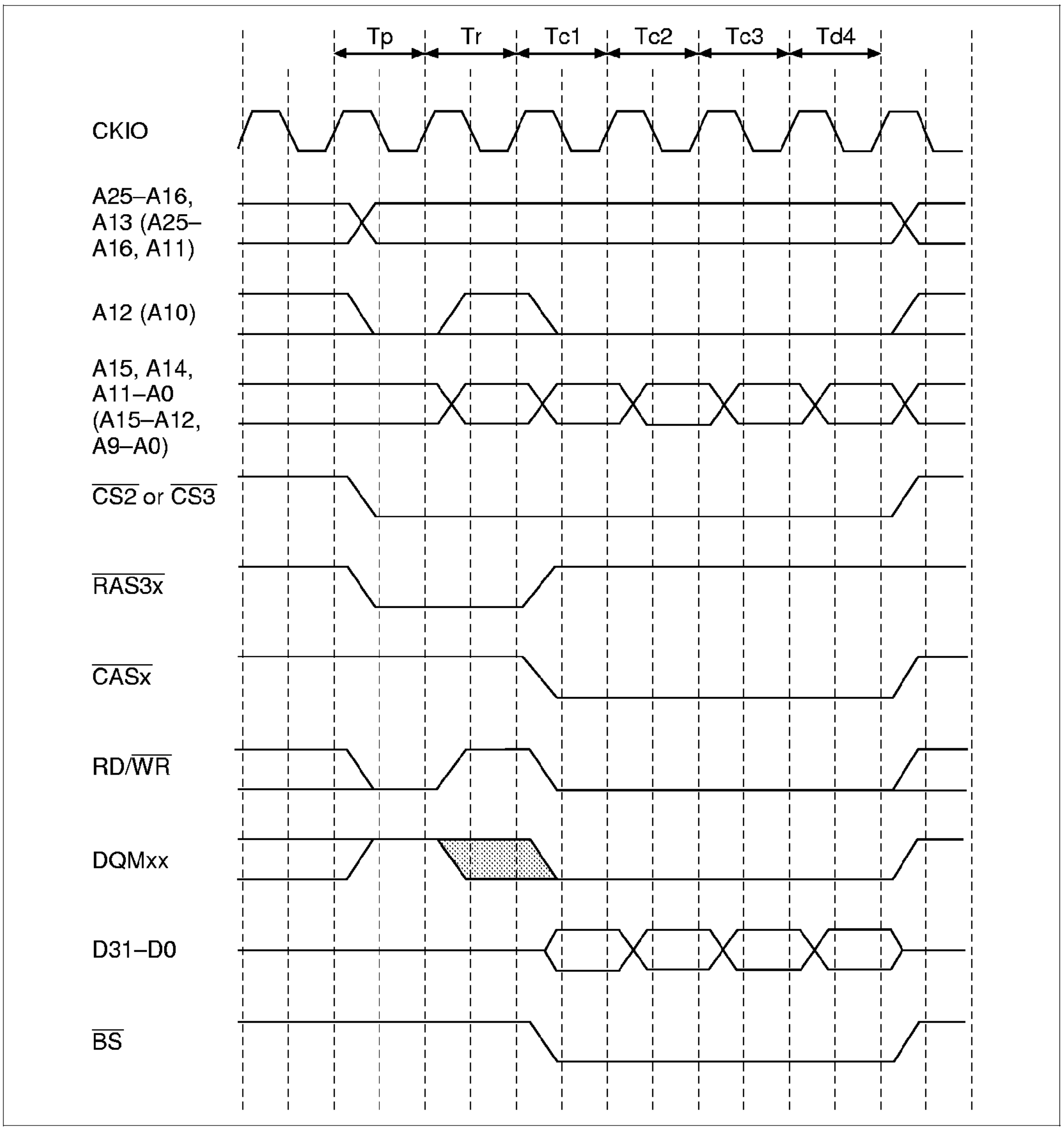


Figure 13.38 Auto-Refresh Operation

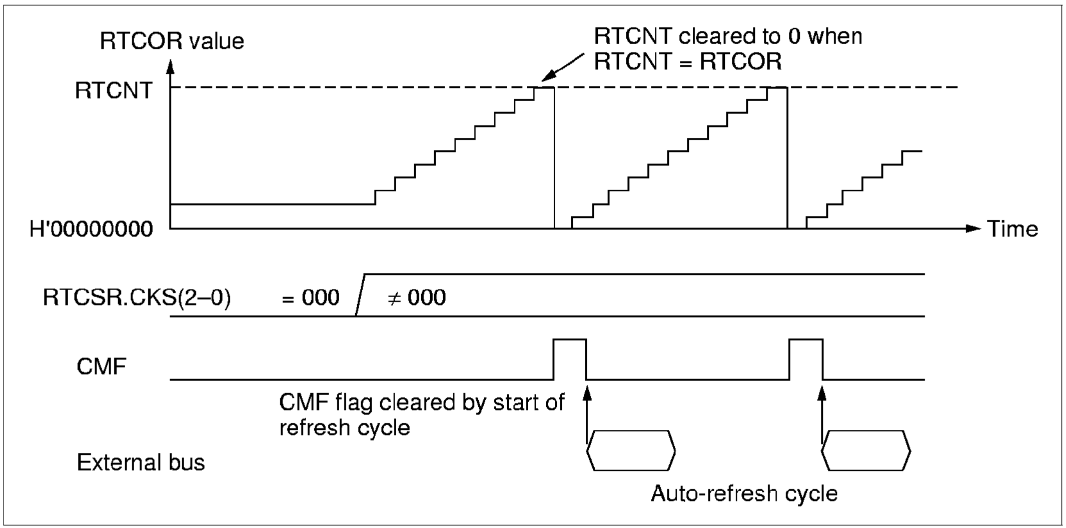


Figure 13.39 SDRAM Auto-Refresh Timing

2. Self-Refreshing

Self-refresh mode is a kind of standby 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 to 1. The self-refresh state is maintained while the CKE signal is low. 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 TPC bits in MCR. Self-refresh timing is shown in figure 13.40. 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 RFSH is set to 1 and RMODE 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 SH7729's 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.

Self-refreshing is performed in normal operation, in sleep mode, in standby mode, and in case of a manual reset.

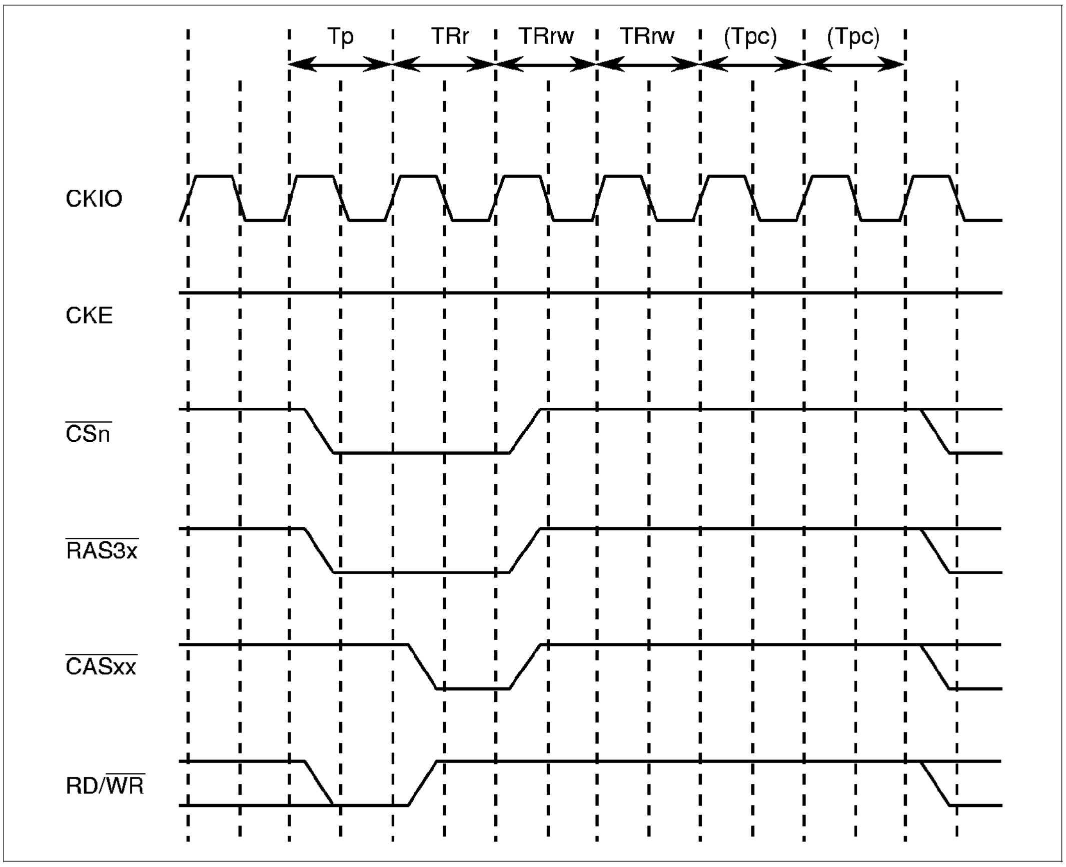


Figure 13.40 SDRAM Self-Refresh Timing

3. Relationship between Refresh Requests and Bus Cycle Requests

If a refresh request is generated during execution of a bus cycle, execution of the refresh is deferred until the bus cycle is completed. If a refresh request occurs when the bus has been released by the bus arbiter, refresh execution is deferred until the bus is acquired. If a match between RTCNT and RTCOR occurs while a refresh is waiting to be executed, so that a new refresh request is generated, the previous refresh request is eliminated. In order for refreshing to be performed normally, care must be taken to ensure that no bus cycle or bus mastership occurs that is longer than the refresh interval. When a refresh request is generated, the IRQOUT pin is asserted (driven low). Therefore, normal refreshing can be performed by having the$\overline{\text{IRQOUT}}$ pin monitored by a bus master other than the SH7729 requesting the bus, or the bus arbiter, and returning the bus to the SH7729. When refreshing is started, and if no other interrupt request has been generated, the$\overline{\text{IRQOUT}}$ pin is negated (driven high).

Power-On Sequence: In order to use 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. In SDRAM mode register setting, the address signal value at that time is latched by a combination of the $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and RD/$\overline{\text{WR}}$ 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'FFFFD000 + X for area 2 SDRAM, and to address H'FFFFE000 + X for area 3 SDRAM. In this operation the data is ignored, but the mode write is performed as a byte-size access. To set burst read/single write, CAS latency 1 to 3, wrap type = sequential, and burst length 1 supported by the SH7729, arbitrary data is written in a byte-size access to the following addresses.

With 32-bit bus width:

	Area 2	Area 3
CAS latency 1	FFFFD840	FFFFE840
CAS latency 2	FFFFD880	FFFFE880
CAS latency 3	FFFFD8C0	FFFFE8C0

With 16-bit bus width:

	Area 2	Area 3
CAS latency 1	FFFFD420	FFFFE420
CAS latency 2	FFFFD440	FFFFE440
CAS latency 3	FFFFD460	FFFFE460

Mode register setting timing is shown in figure 13.40.

As a result of the write to address H'FFFFD000 + X or H'FFFFE000 + X, a precharge all banks (PALL) command is first issued in the TRp1 cycle, then a mode register write command is issued in the TMw1 cycle.

Address signals, when the mode-register write command is issued, are as follows:

A15–A9 = 0000100 (burst read and single write)
A8–A6 = CAS latency
A5 = 0 (burst type = sequential)
A4–A2 = 000 (burst length 1)

Before mode register setting, a 100 µs idle time (depending on the memory manufacturer) must be guaranteed after powering on requested by the SDRAM. If the reset signal pulse width is greater than this idle time, there is no problem in performing mode register setting immediately. The number of dummy auto-refresh cycles specified by the manufacturer (usually 8) or more must be executed. This is usually achieved automatically while various kinds of initialization are being performed after auto-refresh setting, but a way of carrying this out more dependably is to set a short refresh request generation interval just while these dummy cycles are being executed. With simple read or write access, the address counter in the SDRAM used for auto-refreshing is not initialized, and so the cycle must always be an auto-refresh cycle.

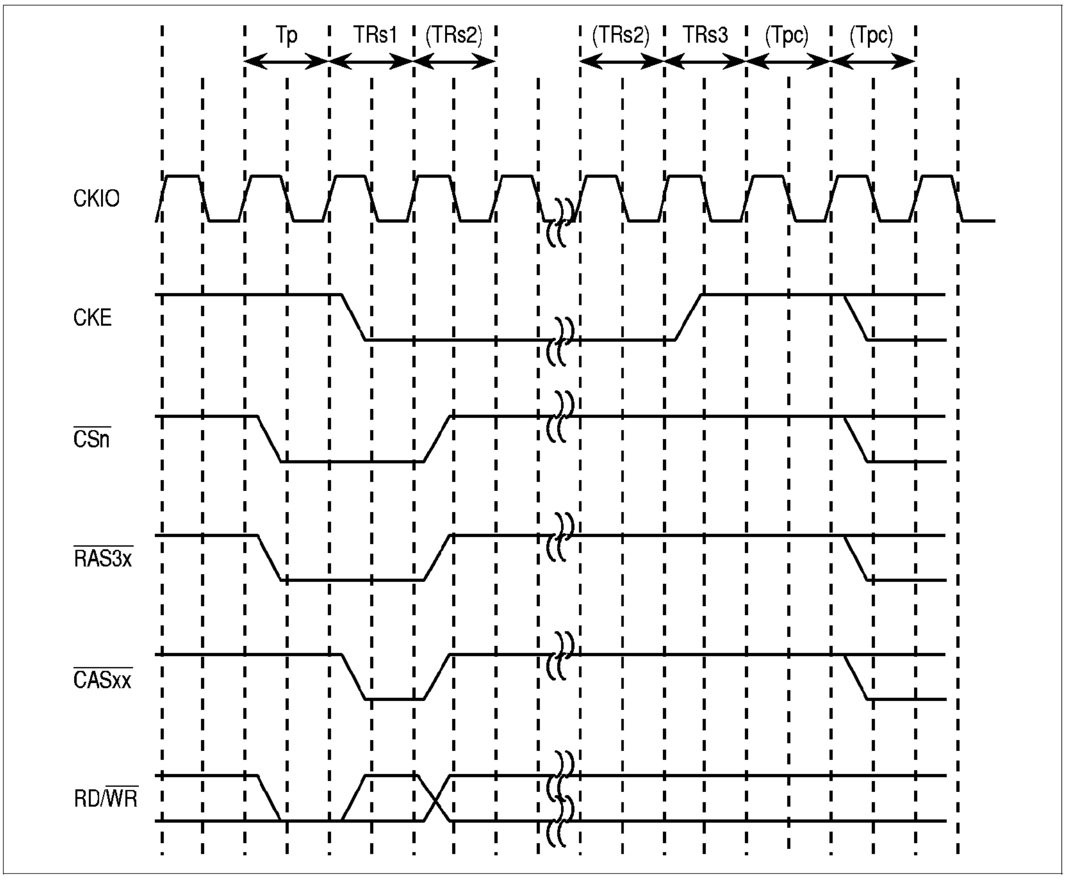

Tp
TRs1
(TRs2)
(TRs2)
TRs3
(Tpc)
(Tpc)
CKIO
CKE
$\overline{\text{CSn}}$
$\overline{\text{RAS3x}}$
$\overline{\text{CASxx}}$
RD/$\overline{\text{WR}}$

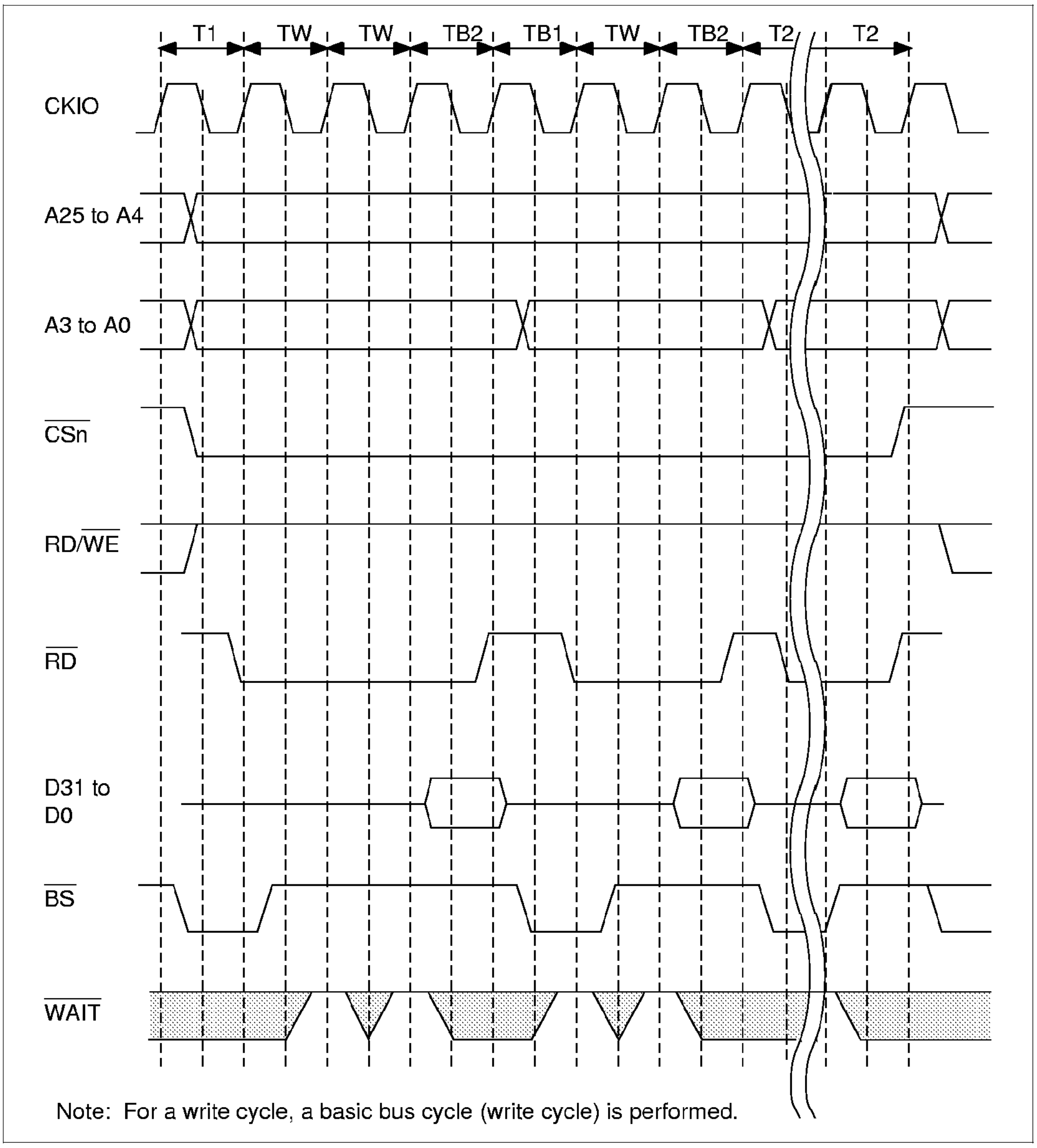
Figure 13.41 SDRAM Mode Write Timing

13.3.6 Burst ROM Interface

Setting bits A0BST (1–0), A5BST (1–0), and A6BST (1–0) in BCR1 to a non-zero value allows burst ROM to be connected to areas 0, 5, and 6. The burst ROM interface provides high-speed access to ROM that has a nibble access function. The timing for nibble access to burst ROM is shown in figure 13.42. Two wait cycles are set. Basically, access is performed in the same way as for normal space, but when the first cycle ends the $\overline{\text{CS0}}$ signal is not negated, and only the address is changed before the next access is executed. When 8-bit ROM is connected, the number of consecutive accesses can be set as 4, 8, or 16 by bits A0BST (1–0), A5BST (1–0), or A6BST (1–0). When 16-bit ROM is connected, 4 or 8 can be set in the same way. When 32-bit ROM is connected, only 4 can be set.

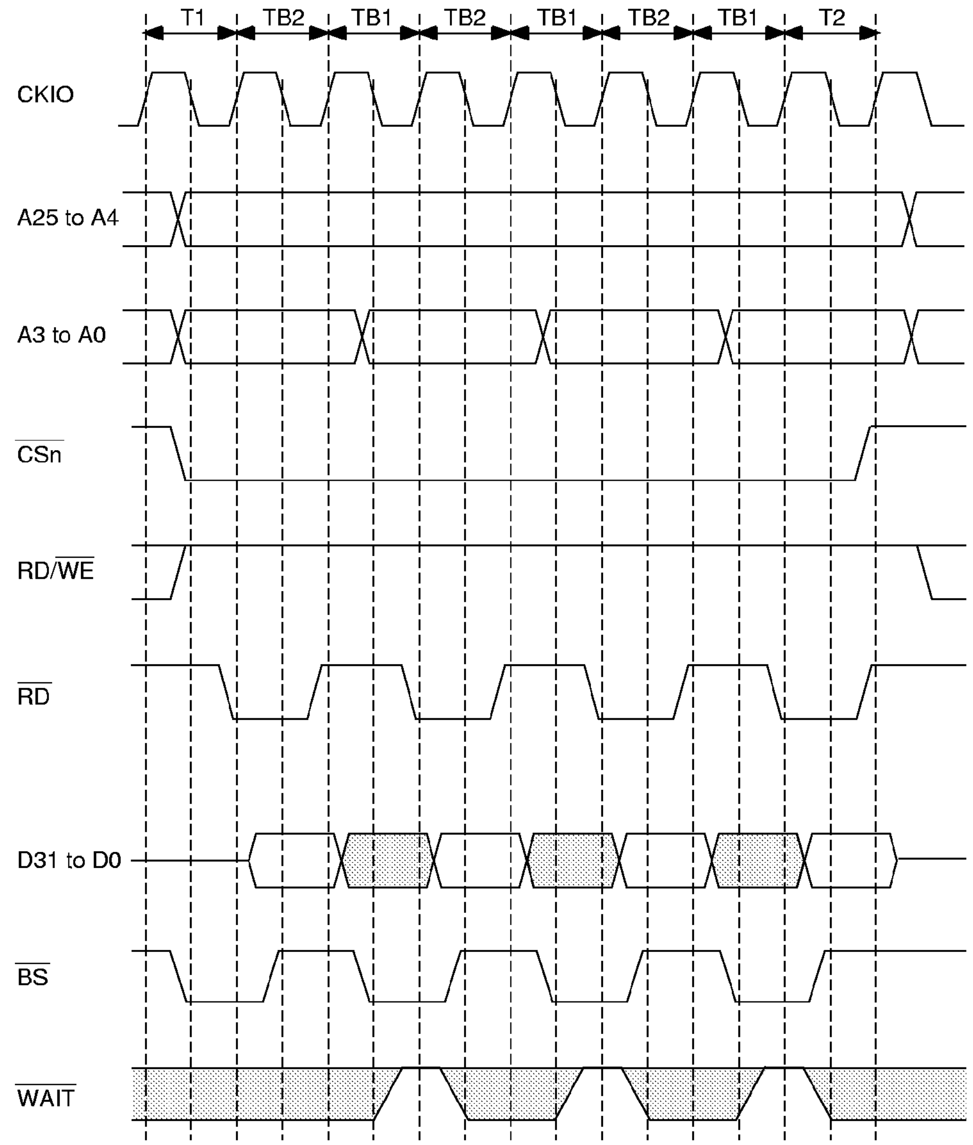
$\overline{\text{WAIT}}$ pin sampling is performed in the first access if one or more wait states are set, and is always performed in the second and subsequent accesses.

The second and subsequent access cycles also comprise two cycles when a burst ROM setting is made and the wait specification is 0. The timing in this case is shown in figure 13.43.



Note: For a write cycle, a basic bus cycle (write cycle) is performed.

Figure 13.42 Burst ROM Wait Access Timing



Note: For a write cycle, a basic bus cycle (write cycle) is performed.

Figure 13.43 Burst ROM Basic Access Timing

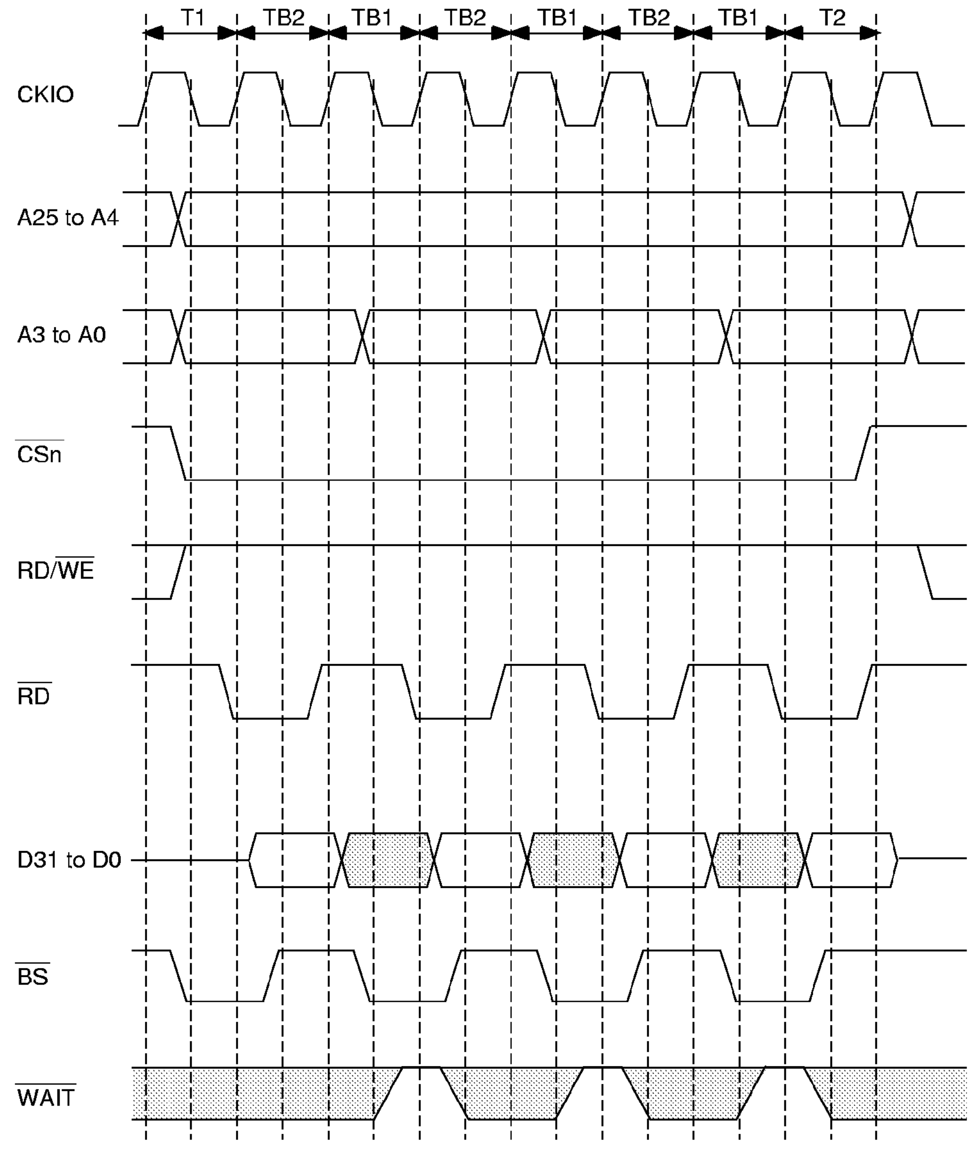
13.3.7 PCMCIA Interface

In the SH7729, setting the A5PCM bit in BCR1 to 1 makes the bus interface for physical space area 5 an IC memory card and I/O card interface as stipulated in JEIDA version 4.2 (PCMCIA2.1). Setting the A6PCM bit to 1 makes the bus interface for physical space area 6 an IC memory card and I/O card interface as stipulated in JEIDA version 4.2.

When the PCMCIA interface is used, a bus size of 8 or 16 bits can be set by bits A5SZ1 and A5SZ0, or A6SZ1 and A6SZ0, in BCR2.

Figure 13.43 shows an example of PCMCIA card connection to the SH7729. To enable active insertion of the PCMCIA cards (i.e. insertion or removal while system power is being supplied), a 3-state buffer must be connected between the SH7729's bus interface and the PCMCIA cards.

As operation in big-endian mode is not explicitly stipulated in the JEIDA/PCMCIA specifications, the PCMCIA interface for the SH7729 in big-endian mode is stipulated independently.



Note: For a write cycle, a basic bus cycle (write cycle) is performed.

Figure 13.44 Example of PCMCIA Interface

Memory Card Interface Basic Timing: Figure 13.45 shows the basic timing for the PCMCIA IC memory card interface. When physical space areas 5 and 6 are designated as PCMCIA interface areas, bus accesses are automatically performed as IC memory card interface accesses.

With a high external bus frequency (CKIO), the setup and hold times for the address (A24–A0), card enable ($\overline{\text{CS5}}$, $\overline{\text{CE2A}}$, $\overline{\text{CS6}}$, $\overline{\text{CE2B}}$), and write data (D15–D0) in a write cycle, become insufficient with respect to $\overline{\text{RD}}$ and $\overline{\text{WR}}$ (the $\overline{\text{WE}}$ pin in the SH7729). The SH7729 provides for this by enabling setup and hold times to be set for physical space areas 5 and 6 in the PCR register. Also, software waits by means of a WCR2 register setting and hardware waits by means of the $\overline{\text{WAIT}}$ pin can be inserted in the same way as for the basic interface. Figure 13.46 shows the PCMCIA memory bus wait timing.

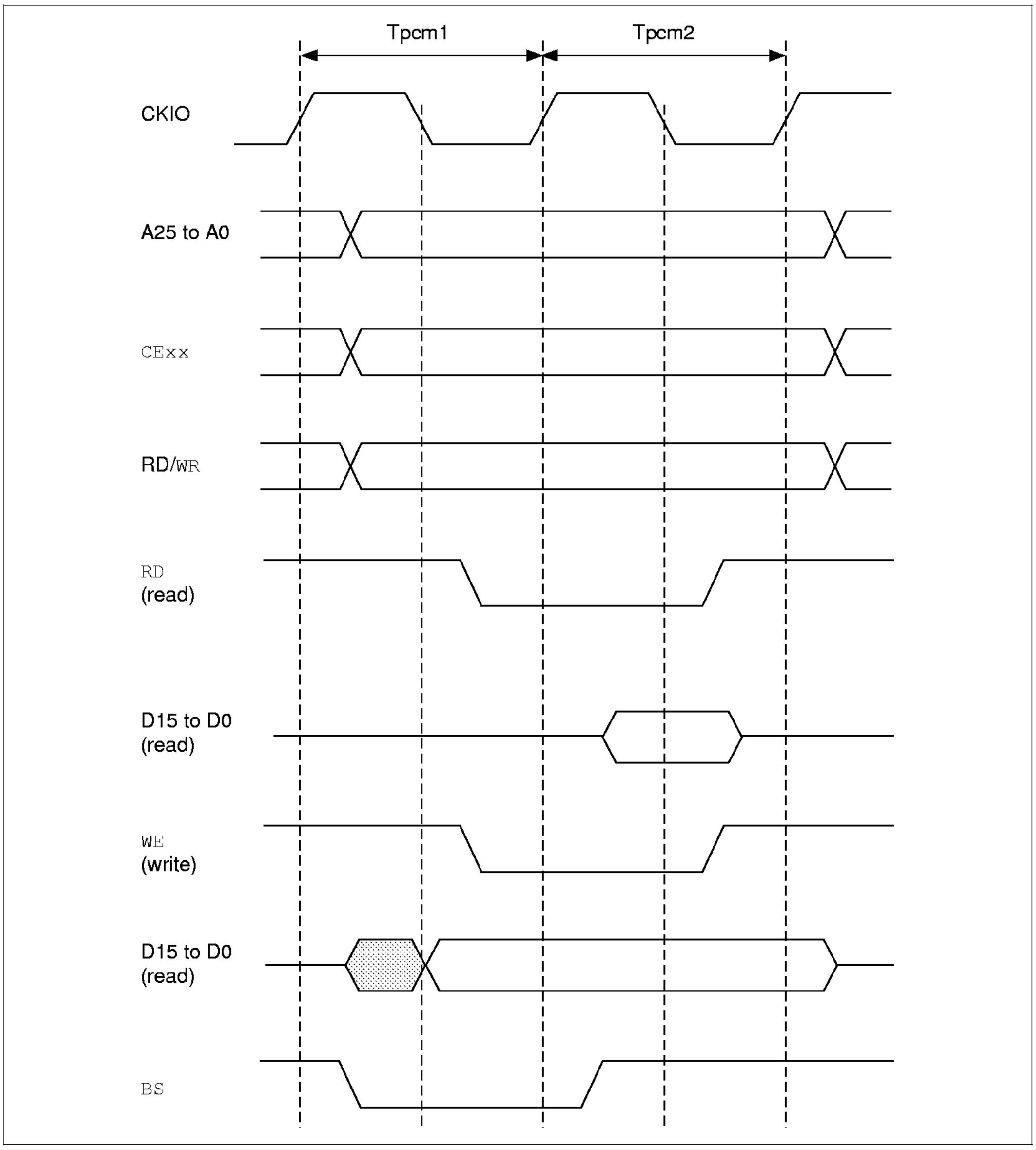


Figure 13.45 Basic Timing for PCMCIA Memory Card Interface

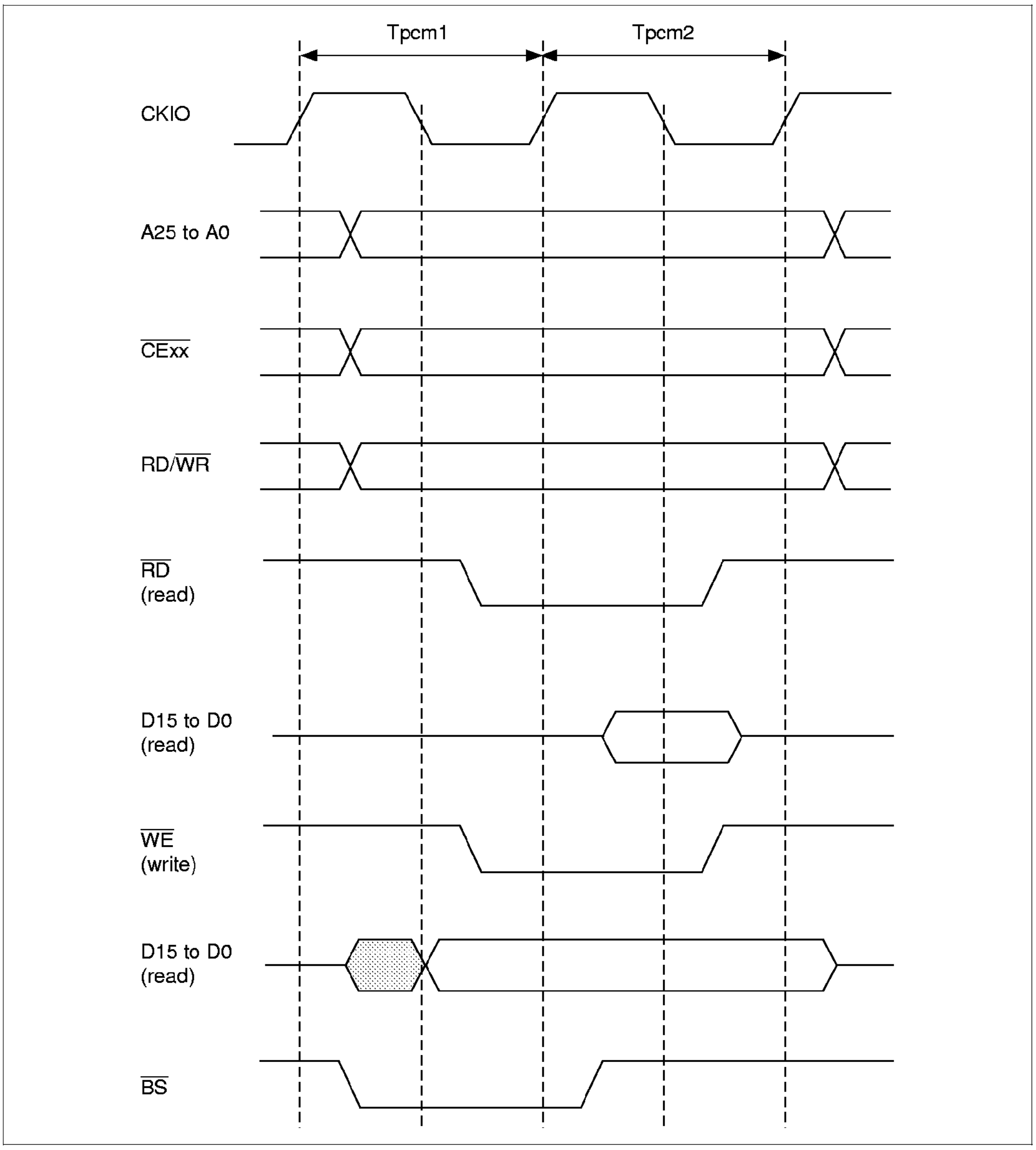


Figure 13.46 Wait Timing for PCMCIA Memory Card Interface

Memory Card Interface Burst Timing: In the SH7729, when the IC memory card interface is selected, page mode burst access mode can be used, for read access only, by setting bits A5BST1 and A5BST0 in BCR1 for physical space area 5, or bits A6BST1 and A6BST0 in BCR1 for area 6. This burst access mode is not stipulated in JEIDA version 4.2 (PCMCIA2.1), but allows high-speed data access using ROM provided with a burst mode, etc.

Burst access mode timing is shown in figures 13.47 and 13.48.

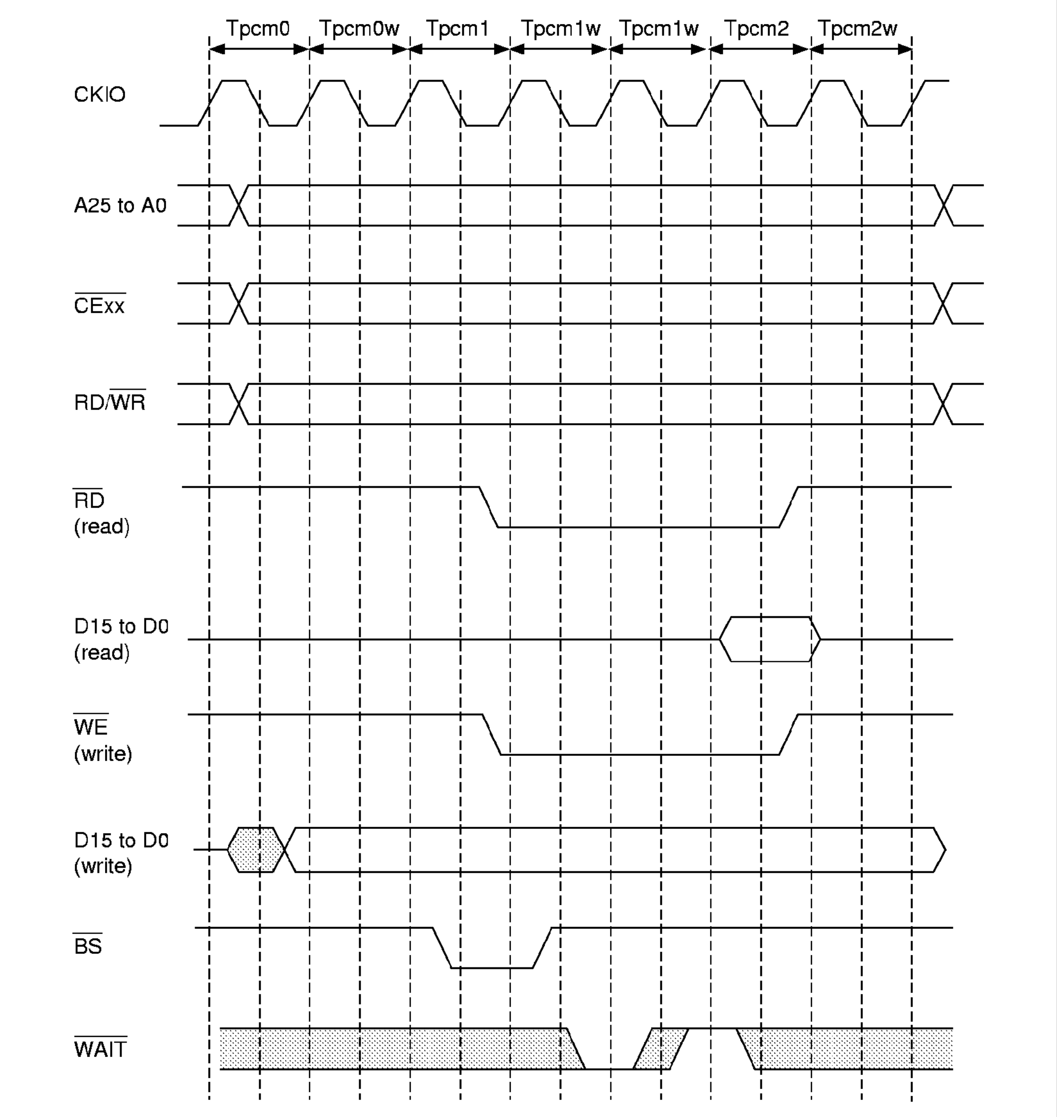


Figure 13.47 Basic Timing for PCMCIA Memory Card Interface Burst Access

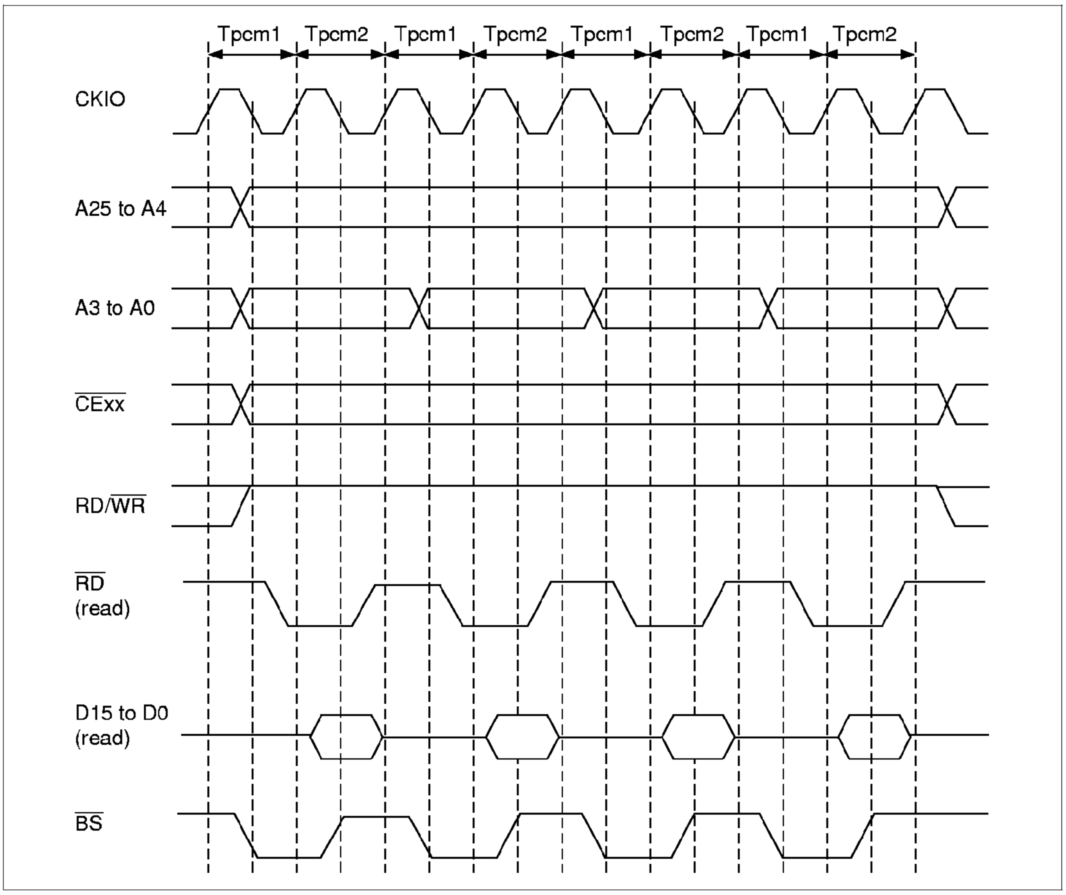


Figure 13.48 Wait Timing for PCMCIA Memory Card Interface Burst Access

When the entire 32-Mbyte memory space is used as IC memory card interface space, the common memory/attribute memory switching signal $\overline{\text{REG}}$ is generated using a port, etc. If 16-Mbytes or less of memory space is sufficient, using 16 Mbytes of memory space as common memory space and 16 Mbytes as attribute memory space enables the A24 pin to be used for the $\overline{\text{REG}}$ signal.

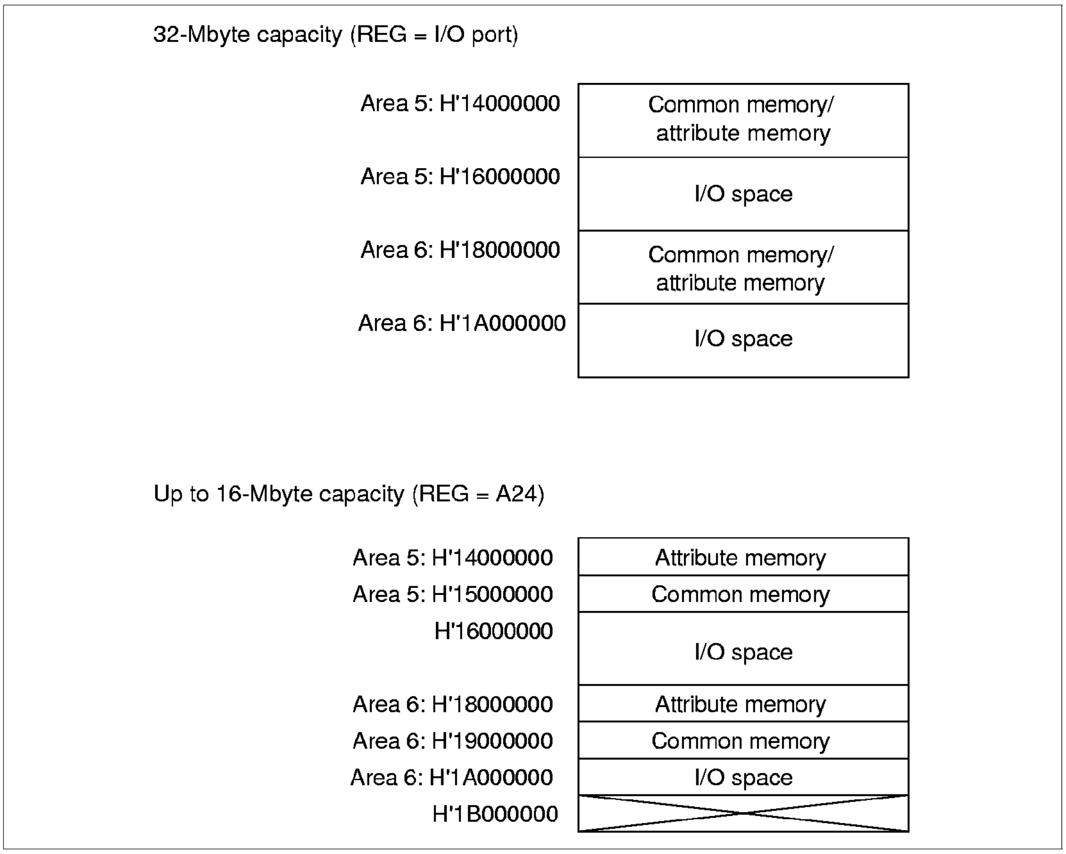


Figure 13.49 PCMCIA Space Allocation

I/O Card Interface Timing: Figures 13.50 and 13.51 show the timing for the PCMCIA I/O card interface.

Switching between the I/O card interface and the IC memory card interface is performed according to the accessed address. When PCMCIA is designed for physical space area 5, the bus access is automatically performed as an I/O card interface access when a physical address from H'16000000 to H'17FFFFFF is accessed. When PCMCIA is designated for physical space area 6, the bus access is automatically performed as an I/O card interface access when a physical address from H'1A000000 to H'1BFFFFFF is accessed.

When accessing a PCMCIA I/O card, the access should be performed using a non-cacheable area in virtual space (P2 or P3 space) or an area specified as non-cacheable by the MMU.

When an I/O card interface access is made to a PCMCIA card in little-endian mode, dynamic sizing of the I/O bus width is possible using the $\overline{\text{IOIS16}}$ pin. When a 16-bit bus width is set for area 6, if the $\overline{\text{IOIS16}}$ signal is high during a word-size I/O bus cycle, the I/O port is recognized as being 8 bits in width. In this case, a data access for only 8 bits is performed in the I/O bus cycle being executed, followed automatically by a data access for the remaining 8 bits.

Figure 13.51 shows the basic timing for dynamic bus sizing.

In big-endian mode, the $\overline{\text{IOIS16}}$ signal is not supported.

In big-endian mode, the $\overline{\text{IOIS16}}$ signal should be fixed low.

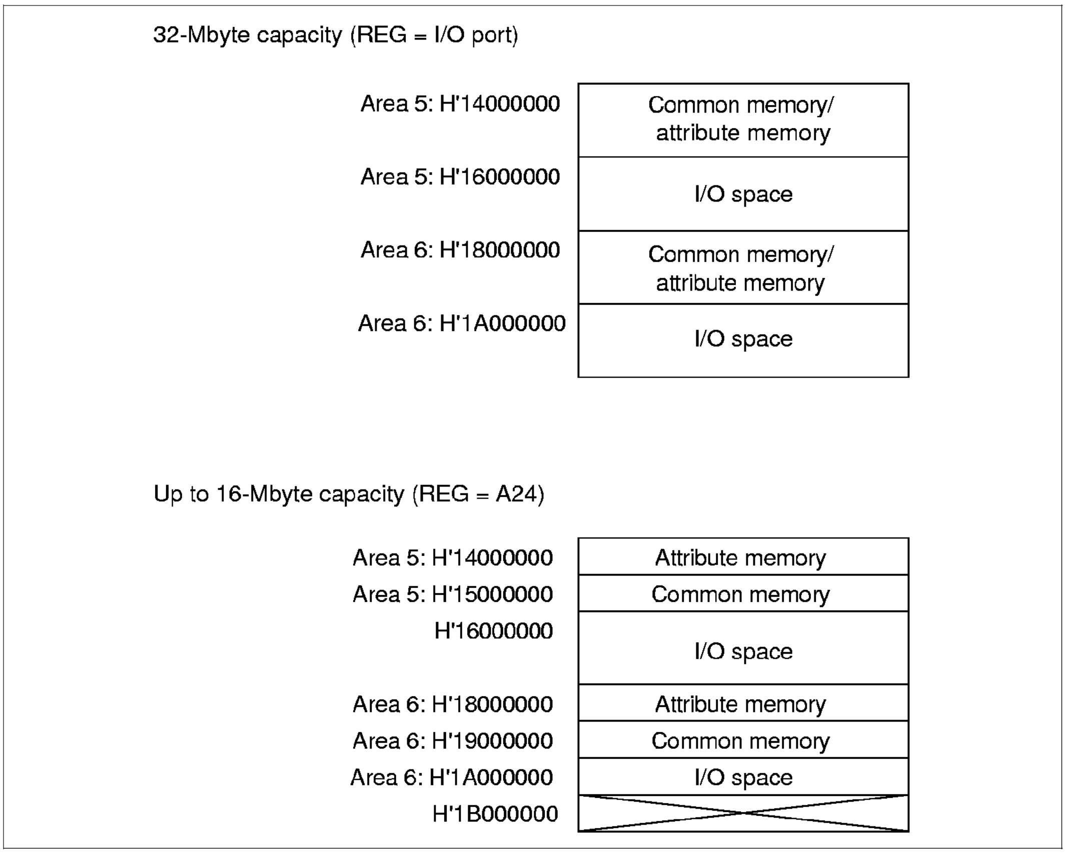


Figure 13.50 Basic Timing for PCMCIA I/O Card Interface

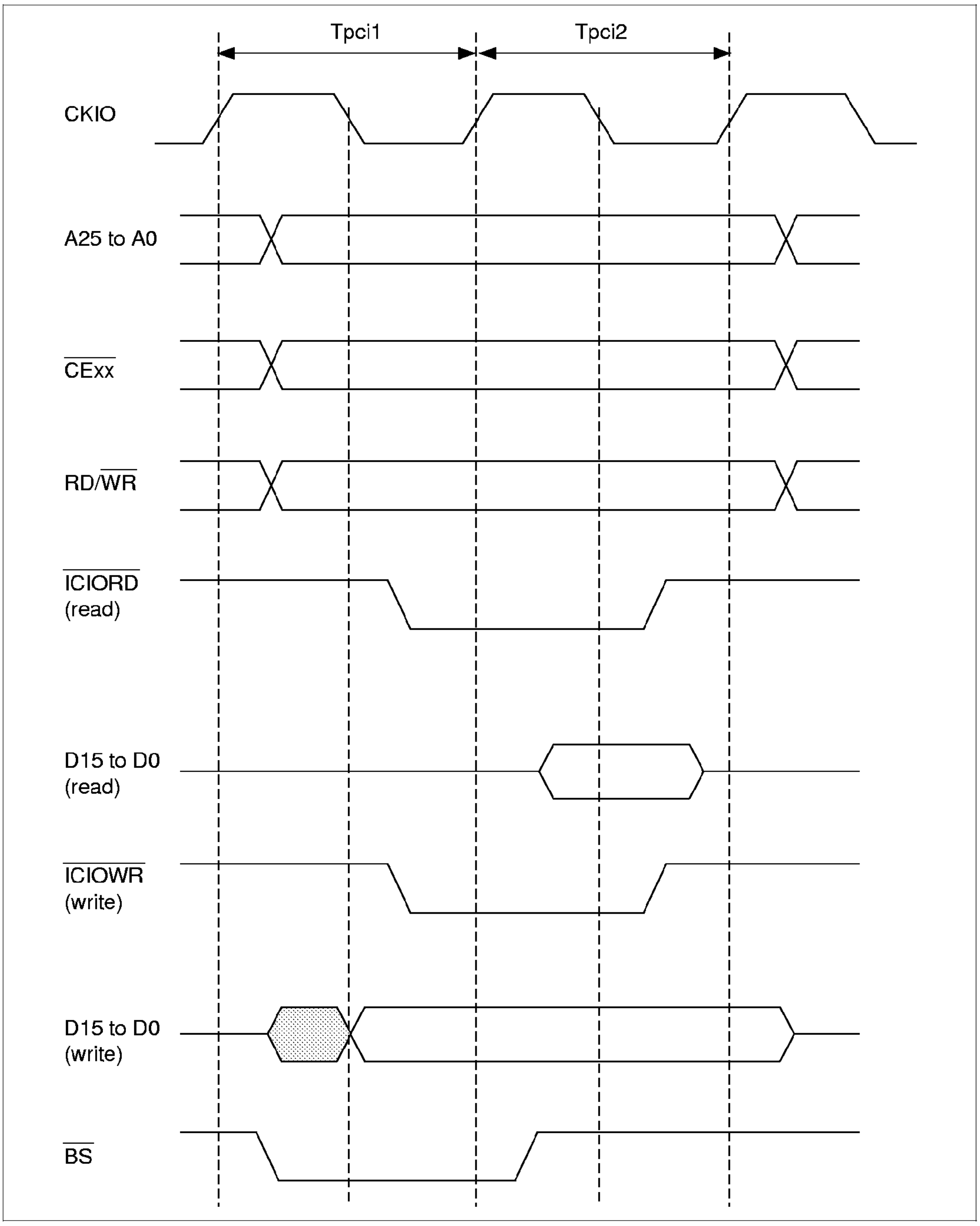


Figure 13.51 Wait Timing for PCMCIA I/O Card Interface

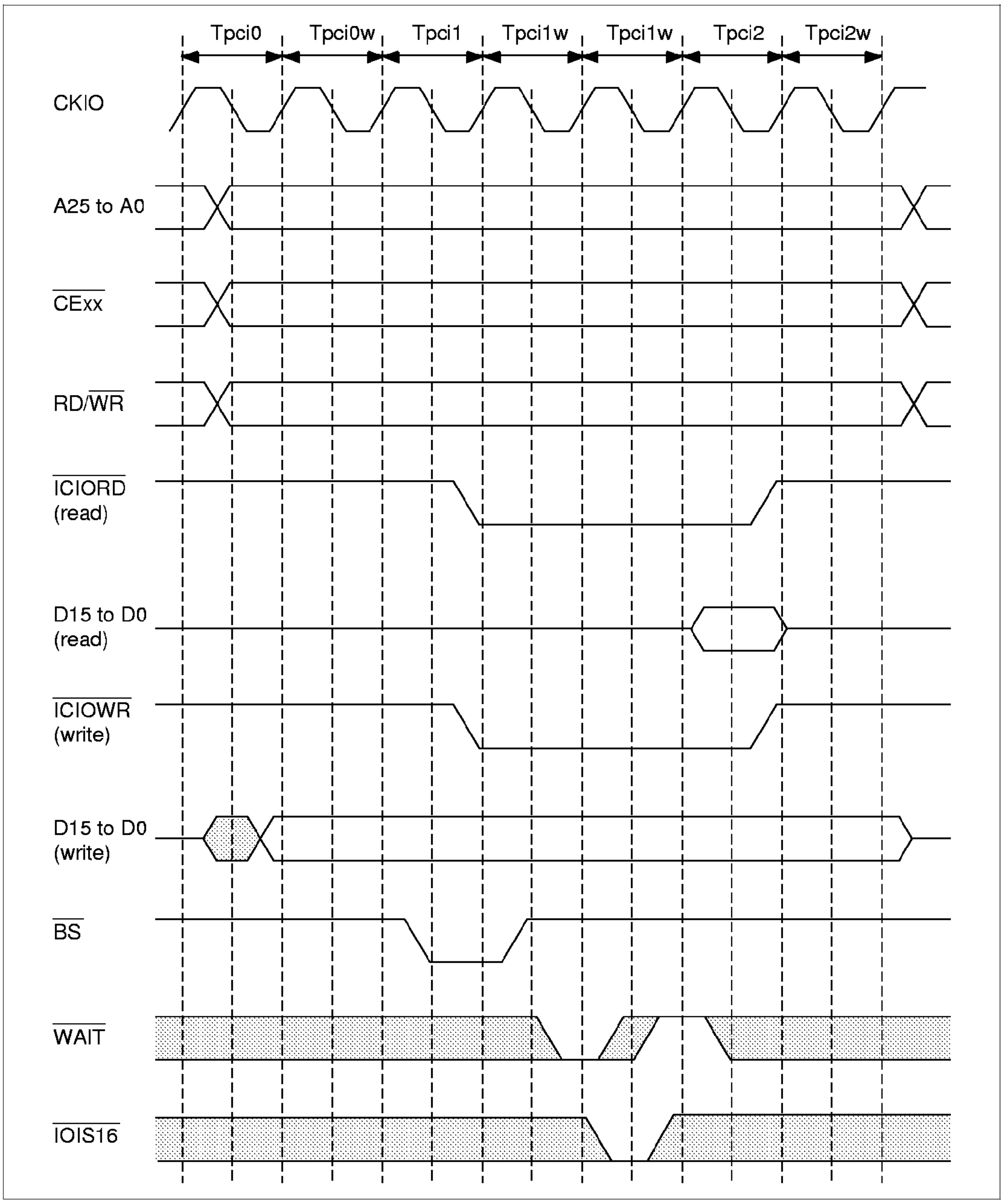


Figure 13.52 Dynamic Bus Sizing Timing for PCMCIA I/O Card Interface

13.3.8 Waits between Access Cycles

A problem associated with higher external memory bus operating frequencies is that data buffer turn-off on completion of a read from a low-speed device may be too slow, causing a collision with data in the next access. This results in lower reliability or incorrect operation. To avoid this problem, a data collision prevention feature has been provided. This memorizes the preceding access area and the kind of read/write. If there is a possibility of a bus collision when the next access is started, a wait cycle is inserted before the access cycle thus preventing a data collision. There are two cases in which a wait cycle is inserted: when an access is followed by an access to a different area, and when a read access is followed by a write access from the SH7729. When the SH7729 performs consecutive write cycles, the data transfer direction is fixed (from the SH7729 to other memory) and there is no problem. With read accesses to the same area, in principle, data is output from the same data buffer, and wait cycle insertion is not performed. Bits AnIW1 and AnIW0 (n = 0, 2–6) in WCR1 specify the number of idle cycles to be inserted between access cycles when a physical space area access is followed by an access to another area, or when the SH7729 performs a write access after a read access to physical space area n. If there is originally space between accesses, the number of idle cycles inserted is the specified number of idle cycles minus the number of empty cycles.

Waits are not inserted between accesses when bus arbitration is performed, since empty cycles are inserted for arbitration purposes.

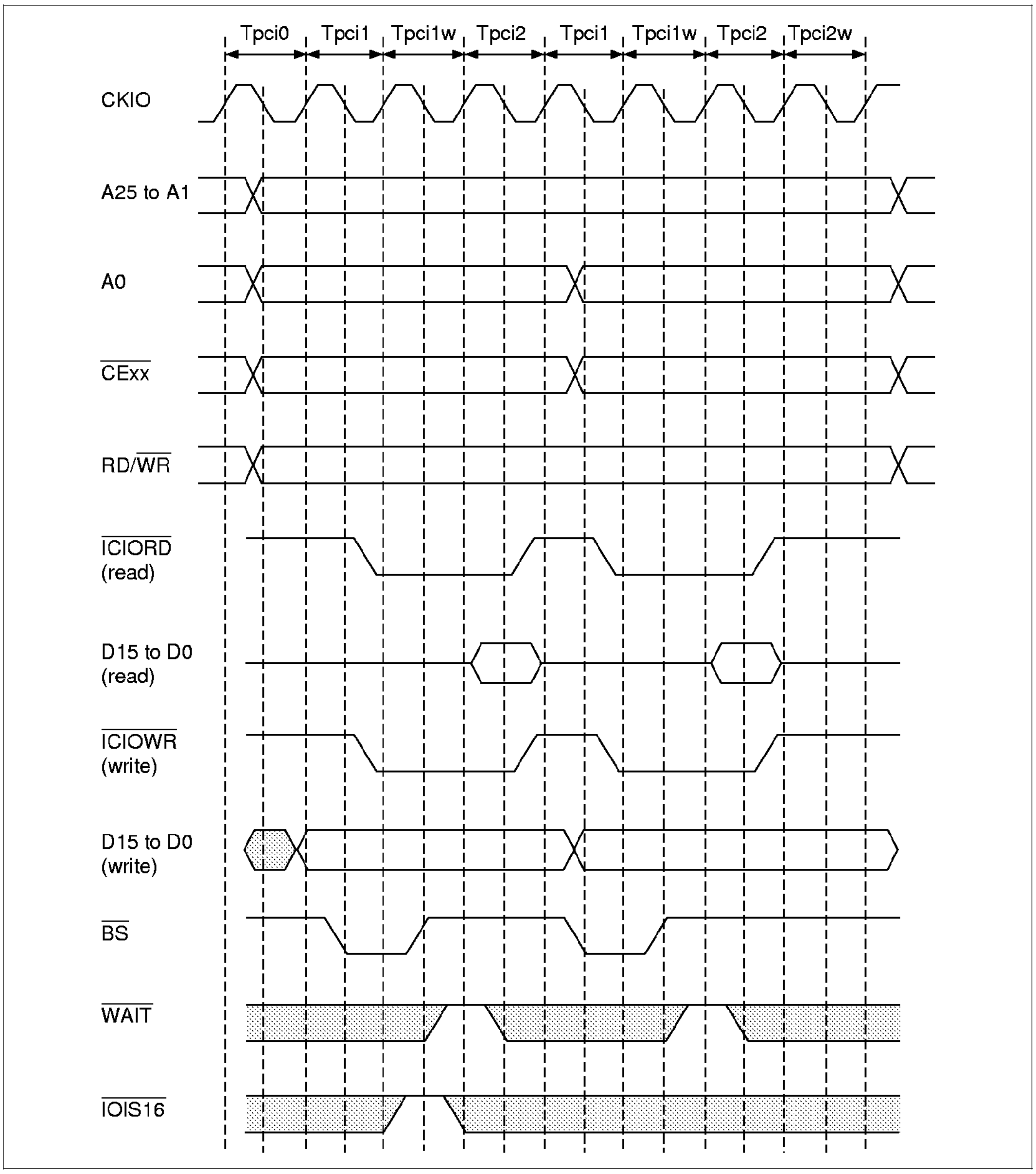


Figure 13.53 Waits between Access Cycles

13.3.9 Bus Arbitration

When a bus release request ($\overline{\text{BREQ}}$) is received from an external device, buses are released after the bus cycle being executed is completed and a bus grant signal ($\overline{\text{BACK}}$) is output. The bus is not released during burst transfers for cache fills or TAS instruction execution between the read cycle and write cycle. Bus arbitration is not executed in multiple bus cycles that are generated when the data bus width is shorter than the access size; i.e. in the bus cycles when longword access is executed for the 8-bit memory. At the negation of $\overline{\text{BREQ}}$, $\overline{\text{BACK}}$ is negated and bus use is restarted. See Appendix B, Pin States, for the pin state when the bus is released.

The SH7729 sometimes needs to retrieve a bus it has released. For example, when memory generates a refresh request or an interrupt request internally, the SH7729 must perform the appropriate processing. The SH7729 has a bus request signal ($\overline{\text{IRQOUT}}$) for this purpose. When it must retrieve the bus, it asserts the $\overline{\text{IRQOUT}}$ signal. Devices asserting an external bus release request receive the assertion of the $\overline{\text{IRQOUT}}$ signal and negate the $\overline{\text{BREQ}}$ signal to release the bus. The SH7729 retrieves the bus and carries out the processing.

$\overline{\text{IRQOUT}}$ Pin Assertion Conditions:

- When a memory refresh request has been generated but the refresh cycle has not yet begun
- When an interrupt is generated with an interrupt request level higher than the setting of the interrupt mask bits (I3–I0) in the status register (SR). (This does not depend on the SR.BL bit.)

13.3.10 Bus Pull-Up

With the SH7729, address pin pull-up can be performed when the bus is released by setting the PULA bit in BCR1 to 1. The address pins are pulled up for a 4-clock period after $\overline{\text{BACK}}$ is asserted. Figure 13.54 shows the address pin pull-up timing. Similarly, data pin pull-up can be performed by setting the PULD bit in BCR1 to 1. The data pins should be pulled up when the data bus is not in use. The data pin pull-up timing for a read cycle is shown in figure 13.55, and the timing for a write cycle in figure 13.56.

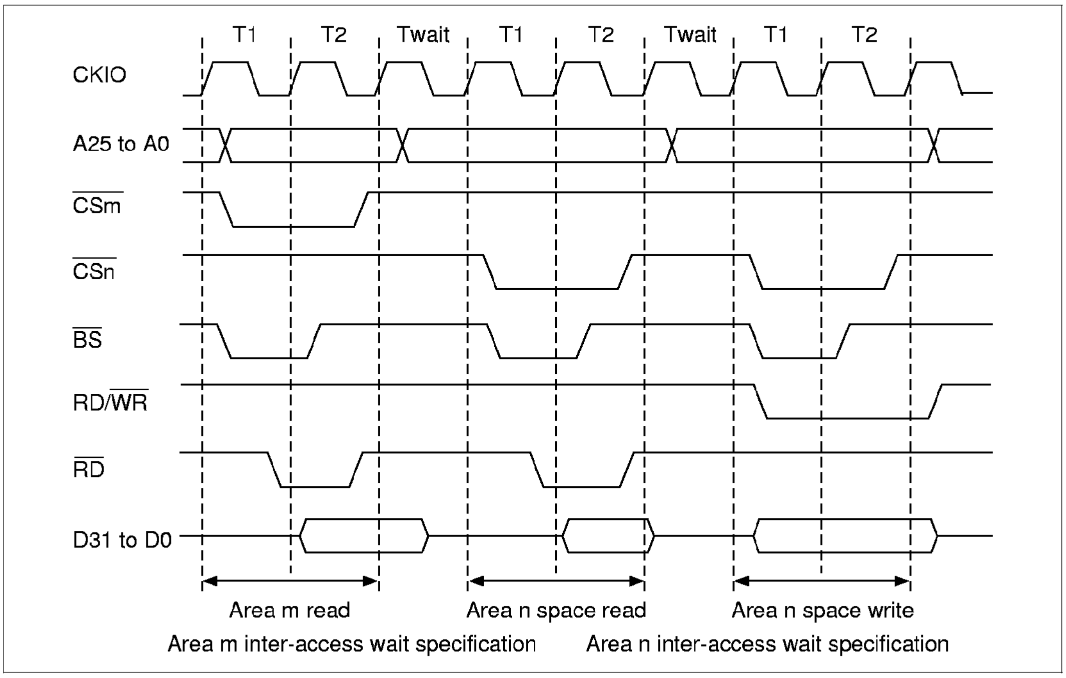


Figure 13.54 Pins A25 to A0 Pull-Up Timing

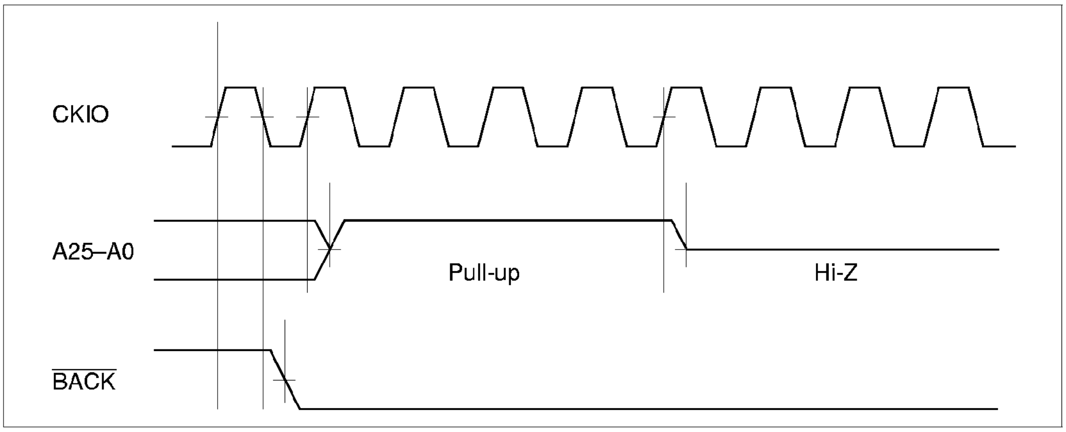


Figure 13.55 Pins D31 to D0 Pull-Up Timing (Read Cycle)

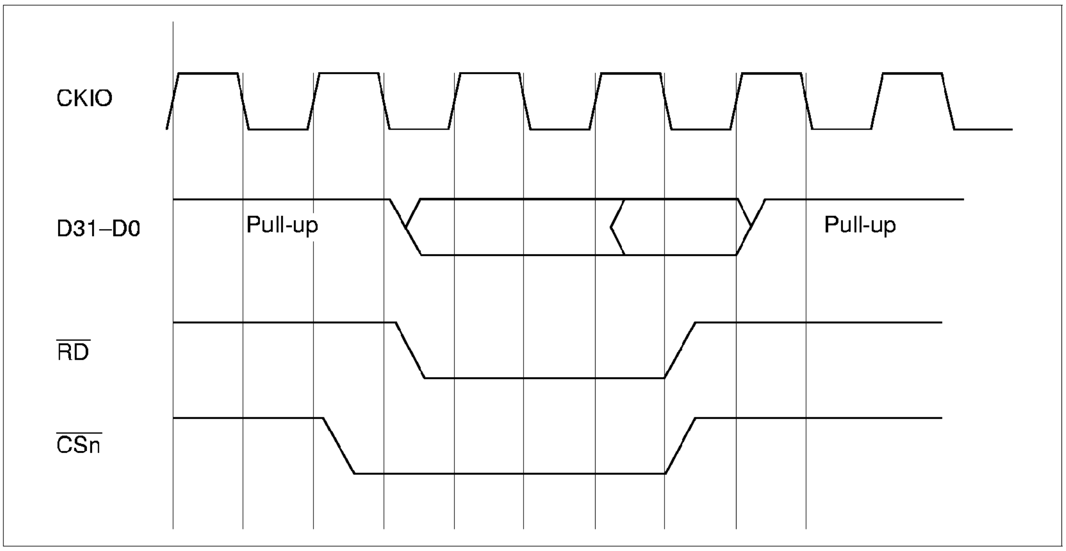


Figure 13.56 Pins D31 to D0 Pull-Up Timing (Write Cycle)

13.3.11 MCS0 to MCS7 Pin Control

The SH7729 is provided with pins MCS0–MCS7 as dedicated CS pins for the ROM connected to area 0 or 2. Assertion of MCS0–MCS7 is controlled by settings in MCSCR0–MCSCR7. This enables 32-, 64-, 128-, or 256-Mbit memory to be connected to area 0 or area 2. Table 13.16 shows MCSCR0–MCSCR7 settings and MCS0–MCS7 assertion conditions.

As the MCS0–MCS7 pins are multiplexed as the PTC0–PTC7 pins, when using these pins as MCS0–MCS7, the corresponding bits in the PCCR register should be set to “other function.”

When CS2/0 = 0 in the MCSCR0 and when the PTC0 pin is switched to MCS0 (when PCOMD1–PCOMD0 are set to “other function”), the CS0 pin is also switched to MCS0.

As port register writes operate on the peripheral clock, they take time compared with instruction execution by the CPU operating on the high-speed internal clock. Therefore, if an instruction that accesses MCS1 to MCS7 is located several instructions after an instruction that switches port C to MCS, the switch from PTC[n] to MCSn and from CS0 to MCS0 may not be performed correctly.

To prevent this problem, the following switching procedure should be used.

- When the program runs with cache on

(1) To switch port C to MCS, set the corresponding bits in the PCCR register to 00 ("other function").

(2) Read the PCCR register and check whether the set value is read. Repeat until the set value is read.

(3) Perform a dummy read from non-cacheable CS0 space (e.g. address H'A0000000). This will result in an access to the CS0 space, and immediately afterward, CS0 will be switched to $\overline{\text{MCS0}}$, and port C[n] will be switched to $\overline{\text{MCSn}}$.

(4) Access can now be made to the $\overline{\text{MCS1}}$ to $\overline{\text{MCS7}}$ spaces.

- When the program runs in $\overline{\text{MCS0}}$ space with cache off

(1) Sct thc PCCR rcgistcr as in 1. (1) abovc.

(2) Place at least three NOP instructions after the instruction in (1). As a result, when the PCCR register is rewritten, an access to the CS0 space will be generated, and immediately afterward, CS0 will be switched to $\overline{\text{MCS0}}$, and port C[n] will be switched to $\overline{\text{MCSn}}$.

(3) Access can now be made to the $\overline{\text{MCS1}}$ to $\overline{\text{MCS7}}$ spaces.

Table 13.16 MCSCRx Settings and $\overline{\text{MCS}}$x Assertion Conditions (X: 0–7)

MCSCRx Settings							$\overline{\text{MCS}}$x Assertion Conditions			
CS2/0	CAP1	CAP0	A25	A24	A23	A22	$\overline{\text{CS0}}$	$\overline{\text{CS2}}$	Address Bus A [25:0]	Notes
0	1	1	0	—	—	—	L	H	H'0000000 to H'1FFFFFF	256-Mbit ROM
			1	—	—	—	L	H	H'2000000 to H'3FFFFFF	
	1	0	0	0	—	—	L	H	H'0000000 to H'0FFFFFF	128-Mbit ROM
			0	1	—	—	L	H	H'1000000 to H'1FFFFFF	
			1	0	—	—	L	H	H'2000000 to H'2FFFFFF	
			1	1	—	—	L	H	H'3000000 to H'3FFFFFF	
	0	1	0	0	0	—	L	H	H'0000000 to H'07FFFFF	64-Mbit ROM
			0	0	1	—	L	H	H'0800000 to H'0FFFFFF	
			0	1	0	—	L	H	H'1000000 to H'17FFFFF	
			0	1	1	—	L	H	H'1800000 to H'1FFFFFF	
			1	0	0	—	L	H	H'2000000 to H'27FFFFF	
			1	0	1	—	L	H	H'2800000 to H'2FFFFFF	
			1	1	0	—	L	H	H'3000000 to H'37FFFFF	
			1	1	1	—	L	H	H'3800000 to H'3FFFFFF	
	0	0	0	0	0	0	L	H	H'0000000 to H'03FFFFF	32-Mbit ROM
			0	0	0	1	L	H	H'0400000 to H'07FFFFF	
			0	0	1	0	L	H	H'0800000 to H'0BFFFFF	
			0	0	1	1	L	H	H'0C00000 to H'0FFFFFF	
			0	1	0	0	L	H	H'1000000 to H'13FFFFF	
			0	1	0	1	L	H	H'1400000 to H'17FFFFF	
			0	1	1	0	L	H	H'1800000 to H'1BFFFFF	
			0	1	1	1	L	H	H'1C00000 to H'1FFFFFF	
			1	0	0	0	L	H	H'2000000 to H'23FFFFF	
			1	0	0	1	L	H	H'2400000 to H'27FFFFF	
			1	0	1	0	L	H	H'2800000 to H'2BFFFFF	
			1	0	1	1	L	H	H'2C00000 to H'2FFFFFF	
			1	1	0	0	L	H	H'3000000 to H'33FFFFF	
			1	1	0	1	L	H	H'3400000 to H'37FFFFF	
			1	1	1	0	L	H	H'3800000 to H'3BFFFFF	
			1	1	1	1	L	H	H'3C00000 to H'3FFFFFF	

Table 13.16 MCSCRx Settings and $\overline{\text{MCSx}}$ Assertion Conditions (X: 0–7) (cont)

MCSCRx Settings							$\overline{\text{MCSx}}$ Assertion Conditions			
CS2/0	CAP1	CAP0	A25	A24	A23	A22	$\overline{\text{CS0}}$	$\overline{\text{CS2}}$	Address Bus A[25:0]	Notes
1	1	1	0	—	—	—	H	L	H'0000000 to H'1FFFFFF	256-Mbit ROM
			1	—	—	—	H	L	H'2000000 to H'3FFFFFF	
	1	0	0	0	—	—	H	L	H'0000000 to H'0FFFFFF	128-Mbit ROM
			0	1	—	—	H	L	H'1000000 to H'1FFFFFF	
			1	0	—	—	H	L	H'2000000 to H'2FFFFFF	
			1	1	—	—	H	L	H'3000000 to H'3FFFFFF	
	0	1	0	0	0	—	H	L	H'0000000 to H'07FFFFF	64-Mbit ROM
			0	0	1	—	H	L	H'0800000 to H'0FFFFFF	
			0	1	0	—	H	L	H'1000000 to H'17FFFFF	
			0	1	1	—	H	L	H'1800000 to H'1FFFFFF	
			1	0	0	—	H	L	H'2000000 to H'27FFFFF	
			1	0	1	—	H	L	H'2800000 to H'2FFFFFF	
			1	1	0	—	H	L	H'3000000 to H'37FFFFF	
			1	1	1	—	H	L	H'3800000 to H'3FFFFFF	
	0	0	0	0	0	0	H	L	H'0000000 to H'03FFFFF	32-Mbit ROM
			0	0	0	1	H	L	H'0400000 to H'07FFFFF	
			0	0	1	0	H	L	H'0800000 to H'0BFFFFF	
			0	0	1	1	H	L	H'0C00000 to H'0FFFFFF	
			0	1	0	0	H	L	H'1000000 to H'13FFFFF	
			0	1	0	1	H	L	H'1400000 to H'17FFFFF	
			0	1	1	0	H	L	H'1800000 to H'1BFFFFF	
			0	1	1	1	H	L	H'1C00000 to H'1FFFFFF	
			1	0	0	0	H	L	H'2000000 to H'23FFFFF	
			1	0	0	1	H	L	H'2400000 to H'27FFFFF	
			1	0	1	0	H	L	H'2800000 to H'2BFFFFF	
			1	0	1	1	H	L	H'2C00000 to H'2FFFFFF	
			1	1	0	0	H	L	H'3000000 to H'33FFFFF	
			1	1	0	1	H	L	H'3400000 to H'37FFFFF	
			1	1	1	0	H	L	H'3800000 to H'3BFFFFF	
			1	1	1	1	H	L	H'3C00000 to H'3FFFFFF	

Section 14 Direct Memory Access Controller (DMAC)

14.1 Overview

This chip includes a four-channel 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, memory-mapped external devices, and on-chip peripheral modules (excluding the DMAC itself, BSC, and UBC). Using the DMAC reduces the burden on the CPU and increases overall operating efficiency.

14.1.1 Features

The DMAC has the following features.

- Four channels
- 4-GB physical address space
- 8-bit, 16-bit, 32-bit, or 16-byte transfer (In 16-byte transfer, four 32-bit reads are executed, followed by four 32-bit writes.)
- 16 Mbytes (16777216 transfers)
- Address mode: Dual address mode and single address mode are supported. In addition, direct address transfer mode or indirect address transfer mode can be selected.
  - Dual address mode transfer: Both the transfer source and transfer destination are accessed by address. Dual address mode has direct address transfer mode and indirect address transfer mode.

    Direct address transfer mode: The values specified in the DMAC registers indicates the transfer source and transfer destination. Two bus cycles are required for one data transfer.

    Indirect address transfer mode: Data is transferred with the address stored prior to the address specified in the transfer source address in the DMAC. Other operations are the same as those of direct address transfer mode. This function is only valid in channel 3. Four bus cycles are requested for one data transfer.
  - Single address mode transfer: 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. One transfer unit of data is transferred in one bus cycle.
- Channel functions: Transfer mode that can be specified is different in each channel.
  - Channel 0: External request can be accepted.
  - Channel 1: External request can be accepted.
  - Channel 2: This channel has a source address reload function, which reloads a source address for each 4 transfers.
  - Channel 3: In this channel, direct address mode or indirect address transfer mode can be specified.

- Reload function: The value that was specified in the source address register can be automatically reloaded every 4 DMA transfers. This function is only valid in channel 2.

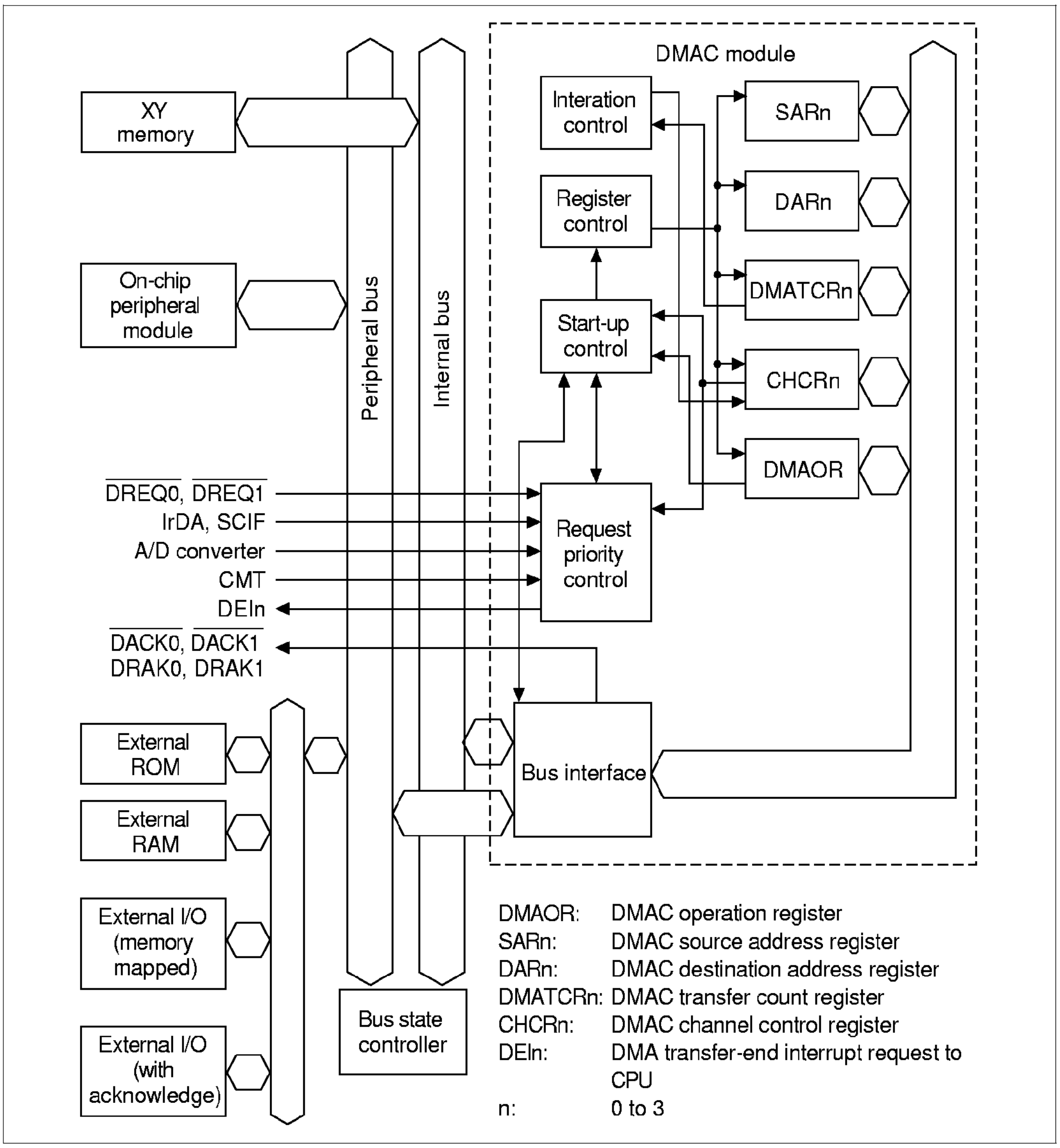
- Transfer requests
  - — External request (From two $\overline{\text{DREQ}}$ pins (channels 0 and 1 only). $\overline{\text{DREQ}}$ can be detected either by edge or by level)
  - — On-chip module request (Requests from on-chip peripheral modules such as serial communications interface (IRDA and SCIF), A/D converter (A/D) and a timer (CMT). This request can be accepted in all the channels)
  - — Auto request (the transfer request is generated automatically within the DMAC)
- Selectable bus modes: Cycle-steal mode or burst mode
- Selectable channel priority levels:

  Fixed mode: The channel priority is fixed.

  Round-robin mode: The priority of the channel in which the execution request was accepted is made the lowest.
- Interrupt request: An interrupt request can be generated to the CPU after transfers end by the specified counts.

14.1.2 Block Diagram

Figure 14.1 is a block diagram of the DMAC.



DMAOR:	DMAC operation register
SARn:	DMAC source address register
DARn:	DMAC destination address register
DMATCRn:	DMAC transfer count register
CHCRn:	DMAC channel control register
DEIn:	DMA transfer-end interrupt request to CPU
n:	0 to 3

Figure 14.1 DMAC Block Diagram

14.1.3 Pin Configuration

Table 14.1 shows the DMAC pins.

Table 14.1Pin Configuration

Channel	Name	Symbol	I/O	Function
0	DMA transfer request	$\overline{\text{DREQ0}}$	I	DMA transfer request input from external device to channel 0
	DREQ acknowledge	$\overline{\text{DACK0}}$	O	Strobe output to an external I/O at DMA transfer request from external device to channel 0
	DMA request acknowledge	DRAK0	O	Output showing that DREQ0 has been accepted
1	DMA transfer request	$\overline{\text{DREQ1}}$	I	DMA transfer request input from external device to channel 1
	DREQ acknowledge	$\overline{\text{DACK1}}$	O	Strobe output to an external I/O at DMA transfer request from external device to channel 1
	DMA request acknowledge	DRAK1	O	Output showing that DREQ1 has been accepted

14.1.4 Register Configuration

Table 14.2 summarizes the DMAC registers. DMAC has a total of 17 registers. Each channel has four control registers. One other control register is shared by all channels.

Table 14.2DMAC Registers

Channel	Name	Abbrevi-ation	R/W	Initial Value	Address	Register Size	Access Size
0	DMA source address register 0	SAR0	R/W	Undefined	H'4000020	32 bits	16, 32*2
	DMA destination address register 0	DAR0	R/W	Undefined	H'4000024	32 bits	16, 32*2
	DMA transfer count register 0	DMATCR0	R/W	Undefined	H'4000028	24 bits	16, 32*3
	DMA channel control register 0	CHCR0	R/W*1	H'00000000	H'400002C	32 bits	8, 16, 32*2
1	DMA source address register 1	SAR1	R/W	Undefined	H'4000030	32 bits	16, 32*2
	DMA destination address register 1	DAR1	R/W	Undefined	H'4000034	32 bits	16, 32*2
	DMA transfer count register 1	DMATCR1	R/W	Undefined	H'4000038	24 bits	16, 32*3
	DMA channel control register 1	CHCR1	R/W*1	H'00000000	H'400003C	32 bits	8, 16, 32*2
2	DMA source address register 2	SAR2	R/W	Undefined	H'4000040	32 bits	16, 32*2
	DMA destination address register 2	DAR2	R/W	Undefined	H'4000044	32 bits	16, 32*2
	DMA transfer count register 2	DMATCR2	R/W	Undefined	H'4000048	24 bits	16, 32*3
	DMA channel control register 2	CHCR2	R/W*1	H'00000000	H'400004C	32 bits	8, 16, 32*2
3	DMA source address register 3	SAR3	R/W	Undefined	H'4000050	32 bits	16, 32*2
	DMA destination address register 3	DAR3	R/W	Undefined	H'4000054	32 bits	16, 32*2
	DMA transfer count register 3	DMATCR3	R/W	Undefined	H'4000058	24 bits	16, 32*3
	DMA channel control register 3	CHCR3	R/W*1	H'00000000	H'400005C	32 bits	8, 16, 32*2
Shared	DMA operation register	DMAOR	R/W*1	H'0000	H'4000060	16 bits	8, 16*2

Notes: 1. Only 0s can be written to bits 1 of CHCR0 to CHCR3, and bits 1 and 2 of DMAOR to clear flag after 1 is read.

2. If SAR0 to SAR3, DAR0 to DAR3, and CHCR0 to CHCR3 are accessed in 16 bits, the value in 16 bits that were not accessed are held.

3. DMATCR has 24 bits from DMATCR0 to DMATCR23. Therefore, even if 1s are written to upper 24 to 31 bits, it is invalid; 0s are always read if these bits are read.

14.2 Register Descriptions

14.2.1 DMA Source Address Registers 0–3 (SAR0–SAR3)

DMA source address registers 0–3 (SAR0–SAR3) are 32-bit read/write registers that specify the source address of a DMA transfer. During a DMA transfer, these registers indicate the next source address.

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. Specifying other addresses does not guarantee operation.

The initial value is undefined by resets. The previous value is held in standby mode.

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

Bit:	23	22	21	20	...	0
Bit name:					...	
Initial value:	—	—	—	—	...	—
R/W:	R/W	R/W	R/W	R/W	...	R/W

14.2.2 DMA Destination Address Registers 0–3 (DAR0–DAR3)

DMA destination address registers 0–3 (DAR0–DAR3) are 32-bit read/write registers that specify the destination address of a DMA transfer. These registers include count functions, and during a DMA transfer, these registers indicate the next destination address.

To transfer data in 16 bits or in 32 bits, specify the address with 16-bit or 32-bit address boundary. Specifying other addresses does not guarantee operation.

The initial value is undefined by resets. The previous value is held in standby mode.

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

Bit:	23	22	21	20	...	0
Bit name:					...	
Initial value:	—	—	—	—	...	—
R/W:	R/W	R/W	R/W	R/W	...	R/W

14.2.3 DMA Transfer Count Registers 0–3 (DMATCR0–DMATCR3)

DMA transfer count registers 0–3 (DMATCR0–DMATCR3) are 24-bit read/write registers that specify the DMA transfer count (bytes, words, or longwords). The number of transfers is 1 when the setting is H'000001, and 16777216 (the maximum) when H'000000 is set. During a DMA transfer, these registers indicate the remaining transfer count.

To transfer data in 16 bytes, one 16-byte transfer (128 bits) counts one.

Writing to upper eight bits in DMATCR is invalid; 0s are read if these bits are read.

When using 16-byte transfer, an integral multiple of 4 (4n) must be set for the number of transfers to ensure normal operation.

The initial value is undefined by resets. The previous value is held in standby mode.

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

Bit:	23	22	21	20	...	0
Bit name:					...	
Initial value:	—	—	—	—	...	—
R/W:	R/W	R/W	R/W	R/W	...	R/W

14.2.4 DMA Channel Control Registers 0–3 (CHCR0–CHCR3)

DMA channel control registers 0–3 (CHCR0–CHCR3) are 32-bit read/write registers that specifies operation mode, transfer method, or others in each channel. Writing to bits 31 to 21 and 7 in this register is invalid; 0 s are read if these bits are read.

Bit 20 is only used in CHCR3. It is not used in CHCR0 to CHCR2. Consequently, writing to this bit is invalid in CHCR0 to CHCR2; 0 is read if this bit is read. Bit 19 is only used in CHCR2; it is not used in CHCR0, CHCR1, and CHCR3. Consequently, writing to this bit is invalid in CHCR0, CHCR1, and CHCR3; 0 is read if this bit is read. Bits 6 and 16 to 18 are only used in CHCR0 and CHCR1; they are not used in CHCR2 and CHCR3. Consequently, writing to these bits is invalid in CHCR2 and CHCR3; 0s are read if these bits are read.

These register values are initialized to 0s after power-on resets. The previous value is held in standby mode.

Bit:	31	...	21	20	19	18	17	16
Bit name:	—	...	—	DI	RO	RL	AM	AL
Initial value:	0	...	0	0	0	0	0	0
R/W:	R	...	R	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)

Bit:	15	14	13	12	11	10	9	8
Bit name:	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	DS	TM	TS1	TS0	IE	TE	DE
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Notes: 1. Only 0 can be written to the TE bit after 1 is read.

2. DI, RO, RL, AM, AL, and DS bits are not included in some channels.

Bit 20—Direct/Indirect Selection (DI): DI selects direct address mode or indirect address mode in channel 3.

This bit is only valid in CHCR3. Writing to this bit is invalid in CHCR0 to CHCR2; 0 is read if this bit is read. When using 16-byte transfer, direct address mode must be specified. Operation is not guaranteed if indirect address mode is specified.

Bit 20: DI	Description
0	Direct address mode (Initial value)
1	Indirect address mode

Bit 19—Source Address Reload Bit (RO): RO selects whether the source address initial value is reloaded in channel 2.

This bit is only valid in CHCR2. Writing to this bit is invalid in CHCR0, CHCR1, and CHCR3; 0 is read if this bit is read. When using 16-byte transfer, this bit must be cleared to 0, specifying non-reloading. Operation is not guaranteed if reloading is specified.

Bit 19: RO	Description
0	A source address is not reloaded (Initial value)
1	A source address is reloaded

Bit 18—Request Check Level Bit (RL): RL specifies the DRAK (acknowledge of $\overline{\text{DREQ}}$) signal output is high active or low active.

This bit is only valid in CHCR0 and CHCR1. Writing to this bit is invalid in CHCR2 and CHCR3; 0 is read if this bit is read.

Bit 18: RL	Description
0	Low-active output of DRAK (Initial value)
1	High-active output of DRAK

Bit 17—Acknowledge Mode Bit (AM): AM specifies whether $\overline{\text{DACK}}$ is output in data read cycle or in data write cycle in dual address mode.

This bit is only valid in CHCR0 and CHCR1. Writing to this bit is invalid in CHCR2 and CHCR3; 0 is read if this bit is read.

Bit 17: AM	Description
0	$\overline{\text{DACK}}$ output in read cycle (Initial value)
1	$\overline{\text{DACK}}$ output in write cycle

Bit 16—Acknowledge Level (AL): AL specifies the $\overline{\text{DACK}}$ (acknowledge) signal output is high active or low active.

This bit is only valid in CHCR0 and CHCR1. Writing to this bit is invalid in CHCR2 and CHCR3; 0 is read if this bit is read.

Bit 16: AL	Description
0	Low-active output of $\overline{\text{DACK}}$ (Initial value)
1	High-active output of $\overline{\text{DACK}}$

Bits 15 and 14—Destination Address Mode Bits 1, 0 (DM1 and DM0): DM1 and DM0 select whether the DMA destination address is incremented, decremented, or left fixed.

Bit 15: DM1	Bit 14: DM0	Description
0	0	Fixed destination address (Initial value)
0	1	Destination address is incremented (+1 in 8-bit transfer, +2 in 16-bit transfer, +4 in 32-bit transfer, +16 in 16-byte transfer)
1	0	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)
1	1	Illegal setting

Bits 13 and 12—Source Address Mode Bits 1, 0 (SM1 and SM0): SM1 and SM0 select whether the DMA source address is incremented, decremented, or left fixed.

Bit 13: SM1	Bit 12: SM0	Description
0	0	Fixed source address (Initial value)
0	1	Source address is incremented (+1 in 8-bit transfer, +2 in 16-bit transfer, +4 in 32-bit transfer, +16 in 16-byte transfer)
1	0	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)
1	1	Reserved (illegal setting)

If the transfer source is specified in indirect address, specify the address, in which the data to be transferred is stored and which is stored as data (indirect address), in source address register 3 (SAR3).

Specification of SAR3 increment or decrement in indirect address mode depends on SM1 and SM0 settings. In this case, however, the SAR3 increment or decrement value is +4, –4, or fixed to 0 regardless of the transfer data size specified in TS1 and TS0.

Bits 11–8—Resource Select Bits 3–0 (RS3–RS0): RS3–RS0 specify which transfer requests will be sent to the DMAC.

Bit 11: RS3	Bit 10: RS2	Bit 9: RS1	Bit 8: RS0	Description
0	0	0	0	External request*, dual address mode (Initial value)
0	0	0	1	Illegal setting
0	0	1	0	External request / Single address mode External address space → external device with DACK
0	0	1	1	External request / Single address mode External device with DACK → external address space
0	1	0	0	Auto request
0	1	0	1	Illegal setting
0	1	1	0	Illegal setting
0	1	1	1	Illegal setting
1	0	0	0	Illegal setting
1	0	0	1	Illegal setting
1	0	1	0	IrDA transmission
1	0	1	1	IrDA reception
1	1	0	0	SCIF transmission
1	1	0	1	SCIF reception
1	1	1	0	Internal A/D
1	1	1	1	CMT

*: External request specification is valid only in channels 0 and 1. None of the request sources can be selected in channels 2 and 3.

Note: When using 16-byte transfer, the following settings must not be made:

1010	IrDA transmission
1011	IrDA reception
1100	SCIF transmission
1101	SCIF reception
1110	A/D converter
1111	CMT

Operation is not guaranteed if these settings are made.

Bit 6—$\overline{\text{DREQ}}$ Select Bit (DS): DS selects the sampling method of the $\overline{\text{DREQ}}$ pin that is used in external request mode is detection in low level or at the falling edge.

This bit is only valid in CHCR0 and CHCR1. Writing to this bit is invalid in CHCR2 and CHCR3; 0 is read if this bit is read.

In channel 0 and 1, if an on-chip peripheral module is specified as a transfer request source or an auto request is specified, specification of this bit is ignored and detection at the falling edge is fixed except in an auto-request.

Bit 6: DS	Description
0	$\overline{\text{DREQ}}$ detected in low level (Initial value)
1	$\overline{\text{DREQ}}$ detected at falling edge

Bit 5—Transmit Mode (TM): TM specifies the bus mode when transferring data.

Bit 5: TM	Description
0	Cycle steal mode (Initial value)
1	Burst mode

Bits 4 and 3—Transmit Size Bits 1 and 0 (TS1, TS0): TS1 and TS0 specify the size of data to be transferred.

Bit 4: TS1	Bit 3: TS0	Description
0	0	Byte size (8 bits) (Initial value)
0	1	Word size (16 bits)
1	0	Longword size (32 bits)
1	1	16-byte unit (4 longword transfers)

Bit 2—Interrupt Enable Bit (IE): Setting this bit to 1 generates an interrupt request when data transfer end (TE = 1) by the count specified in DMATCR.

Bit 2: IE	Description
0	Interrupt request is not generated even if data transfer ends by the specified count (Initial value)
1	Interrupt request is generated if data transfer ends by the specified count

Bit 1—Transfer End Bit (TE): TE is set to 1 when data transfer ends by the count specified in DMATCR. At this time, if the IE bit is set to 1, an interrupt request is generated.

Before this bit is set to 1, if data transfer ends due to an NMI interrupt, a DMAC address error, or clearing the DE bit or the DME bit in DMAOR, this bit is not set to 1. Even if the DE bit is set to 1 while this bit is set to 1, transfer is not enabled.

Bit 1: TE	Description
0	Data transfer does not end by the count specified in DMATCR (Initial value) Clear condition: Writing 0 after TE = 1 read at power-on reset or manual reset
1	Data transfer ends by the specified count

Bit 0—DMAC Enable Bit (DE): DE enables channel operation.

Bit 0: DE	Description
0	Disables channel operation (Initial value)
1	Enables channel operation

If an auto request is specifies (specified in RS3 to RS0), transfer starts when this bit is set to 1. In an external request or an internal module request, transfer starts if transfer request is generated after this bit is set to 1. Clearing this bit during transfer can terminate transfer.

Even if the DE bit is set, transfer is not enabled if the TE bit is 1, the DME bit in DMAOR is 0, or the NMIF bit in DMAOR is 1.

14.2.5 DMA Operation Register (DMAOR)

The DMA operation register (DMAOR) is a 16-bit read/write register that controls the DMAC transfer mode. Writing to bits 15 to 10 and bits 7 to 3 is invalid in this register; 0 is always read if these bits are read.

It is initialized to 0 at power-on reset, or in hardware standby mode or software standby mode.

Bit:	15	14	13	12	11	10	9	8
Bit name:	—	—	—	—	—	—	PR1	PR0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	—	—	AE	NMIF	DME
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/(W)*	R/(W)*	R/W

Note: * Only 0 can be written to the AE and NMIF bits after 1 is read.

Bits 9 and 8—Priority Mode Bits 1 and 0 (PR1 and PR0): PR1 and PR0 select the priority level between channels when there are transfer requests for multiple channels simultaneously.

Bit 9: PR1	Bit 8: PR0	Description
0	0	CH0 > CH1 > CH2 > CH3 (Initial value)
0	1	CH0 > CH2 > CH3 > CH1
1	0	CH2 > CH0 > CH1 > CH3
1	1	Round-robin

Bit 2—Address Error Flag Bit (AE): AE indicates that an address error occurred during DMA transfer. If this bit is set during data transfer, transfers on all channels are suspended. The CPU cannot write 1 to this bit. This bit can only be cleared by writing 0 after reading 1.

Bit 2: AE	Description
0	No DMAC address error. DMA transfer is enabled. (Initial value) Clear conditions: Writing AE = 0 after AE = 1 read Power-on reset, manual reset
1	DMAC address error. DMA transfer is disabled. This bit is set by occurrence of a DMAC address error.

Bit 1—NMI Flag Bit (NMIF): NMIF indicates that an NMI interrupt occurred. This bit is set regardless of whether DMAC is in operating or halt state. The CPU cannot write 1 to this bit. Only 0 can be written to clear this bit after 1 is read.

Bit 1: NMIF	Description
0	No NMI input. DMA transfer is enabled. (Initial value) Clear condition: Writing NMIF = 0 after NMIF = 1 read at power-on reset or manual reset
1	NMI input. DMA transfer is disabled. This bit is set by occurrence of an NMI interrupt.

Bit 0—DMA Master Enable Bit (DME): 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 1s, transfer is enabled in the corresponding channel. If this bit is cleared during transfer, transfers in all the channels can be terminated.

Even if the DME bit is set, transfer is not enabled if the TE bit is 1 or the DE bit is 0 in CHCR, or the NMIF bit is 1 in DMAOR.

Bit 0: DME	Description
0	Disable DMA transfers on all channels (Initial value)
1	Enable DMA transfers on all channels

14.3 Operation

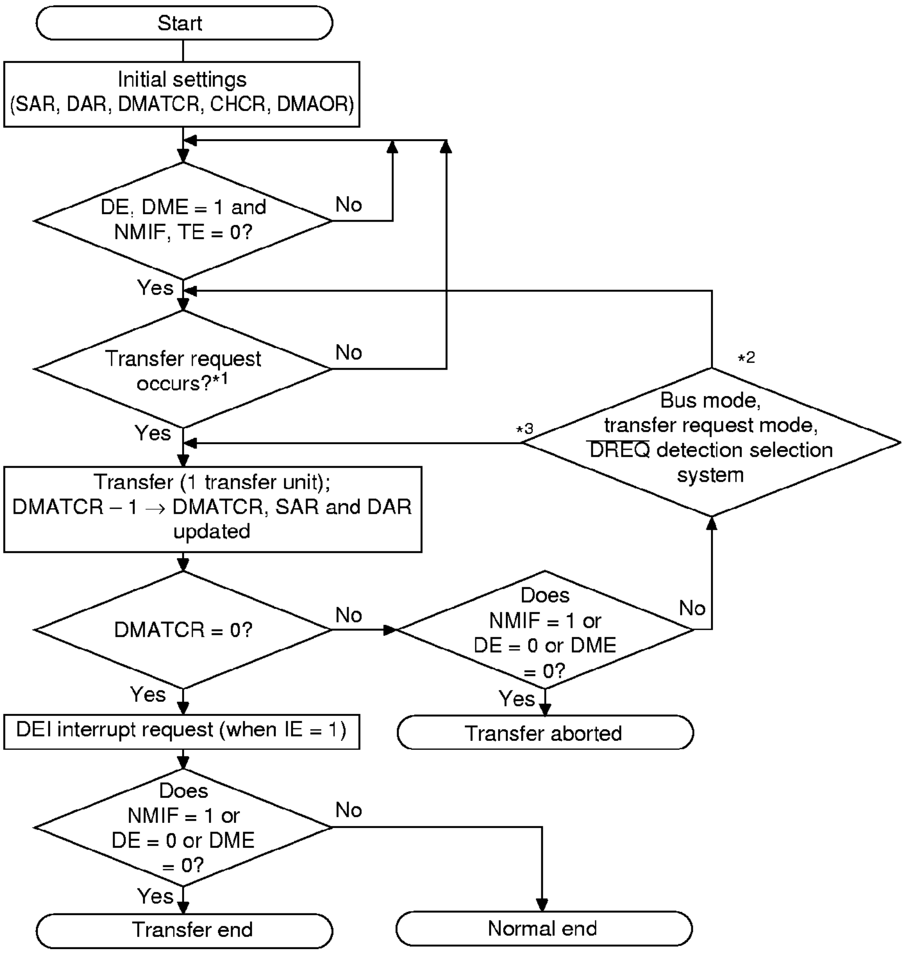
When there is a DMA transfer request, the DMAC starts the transfer according to the predetermined channel priority order; when the transfer end conditions are satisfied, it ends the transfer. Transfers can be requested in three modes: auto request, external request, and on-chip module request. The dual 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.

14.3.1 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), and DMA operation register (DMAOR) 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, 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.
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 14.2 is a flowchart of this procedure.



Notes: 1. In auto-request mode, transfer begins when NMIF and TE are all 0 and the DE and DME bits are set to 1.

2. $\overline{\text{DREQ}}$ = level detection in burst mode (external request) or cycle-steal mode.

3. $\overline{\text{DREQ}}$ = edge detection in burst mode (external request), or auto-request mode in burst mode.

Figure 14.2 DMAC Transfer Flowchart

14.3.2 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–RS0 bits of the DMA channel control registers 0–3 (CHCR0–CHCR3).

Auto-Request Mode: When there is no transfer request signal from an external source, as in a memory-to-memory transfer or a transfer between memory and an on-chip peripheral module unable to request a transfer, the auto-request mode allows the DMAC to automatically generate a transfer request signal internally. When the DE bits of CHCR0–CHCR3 and the DME bit of the DMAOR are set to 1, the transfer begins so long as the TE bits of CHCR0–CHCR3 and the NMIF bit of DMAOR are all 0.

External Request Mode: In this mode a transfer is performed at the request signal ($\overline{\text{DREQ}}$) of an external device. Choose one of the modes shown in table 14.3 according to the application system. When this mode is selected, if the DMA transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0), a transfer is performed upon a request at the $\overline{\text{DREQ}}$ input. Choose to detect $\overline{\text{DREQ}}$ by either the falling edge or low level of the signal input with the DS bit of CHCR0–CHCR3 (DS = 0 is level detection, DS = 1 is edge detection). The source of the transfer request does not have to be the data transfer source or destination.

Table 14.3Selecting External Request Modes with the RS Bits

RS3	RS2	RS1	RS0	Address Mode	Source	Destination
0	0	0	0	Dual address mode	Any*	Any*

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

On-Chip Module Request: In this mode a transfer is performed at the transfer request signal (interrupt request signal) of an on-chip module. This mode cannot be set in case of 16-byte transfer. The transfer request signals include 6 signals: the receive data full interrupts (RXI) and the transmit data empty interrupts (TXI) from two serial communication interfaces (IrDA, SCIF), the A/D conversion end interrupt (ADI) of the A/D converter and the compare match timer interrupt (CMI) of the CMT (table 14.4). When this mode is selected, if the DMA transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0), a transfer is performed upon the input of a transfer request signal. The source of the transfer request does not have to be the data transfer source or destination. When RXI is set as the transfer request, however, the transfer source must be the SCI's receive data register (RDR). Likewise, when TXI is set as the transfer request, the transfer source must be the SCI's transmit data register (TDR). And if the transfer requester is the A/D converter, the data transfer source must be the A/D data register.

Table 14.4Selecting On-Chip Peripheral Module Request Modes with the RS Bit

RS3	RS2	RS1	RS0	DMA Transfer Request Source	DMA Transfer Request Signal	Source	Destination	Bus Mode
1	0	1	0	IrDA transmitter	TXI1 (IrDA transmit data empty interrupt transfer request)	Any*	TDR1	Burst/ cycle steal
1	0	1	1	IrDA receiver	RXI1 (IrDA receive data full interrupt transfer request)	RDR1	Any*	Burst/ cycle steal
1	1	0	0	SCIF transmitter	TXI2 (SCIF transmit data empty interrupt transfer request)	Any*	TDR2	Burst/ cycle steal
1	1	0	1	SCIF receiver	RXI2 (SCIF receive data full interrupt transfer request)	RDR1	Any*	Burst/ cycle steal
1	1	1	0	A/D converter	ADI (A/D conversion end interrupt)	ADDR	Any*	Burst/ cycle steal
1	1	1	1	CMT	CMI (Compare match timer interrupt)	Any*	Any*	Burst/ cycle steal

ADDR: A/D data register of A/D converter

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

When outputting transfer requests from on-chip peripheral modules, the appropriate interrupt enable bits must be set to output the interrupt signals.

If the interrupt request signal of the on-chip peripheral module is used as a DMA transfer request signal, an interrupt is not generated to the CPU.

The DMA transfer request signals of table 14.4 are automatically withdrawn when the corresponding DMA transfer is performed. If the cycle steal mode is being employed, they are withdrawn at the first transfer; if the burst mode is being used, they are withdrawn at the last transfer.

14.3.3 Channel Priority

When the DMAC receives simultaneous transfer requests on two or more channels, it selects a channel according to a predetermined priority order. Two modes (fixed mode and round-robin mode) are selected by the priority bits PR1 and PR0 in the DMA operation register.

Fixed Mode: In these modes, the priority levels among the channels remain fixed. There are three kinds of fixed modes as follows:

CH0 > CH1 > CH2 > CH3
CH0 > CH2 > CH3 > CH1
CH2 > CH0 > CH1 > CH3

These are selected by the PR1 and the PR0 bits in the DMA operation register (DMAOR).

Round-Robin Mode: Each time one word, byte, or longword 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 14.3. The priority of the round-robin mode is CH0 > CH1 > CH2 > CH3 immediately after reset.

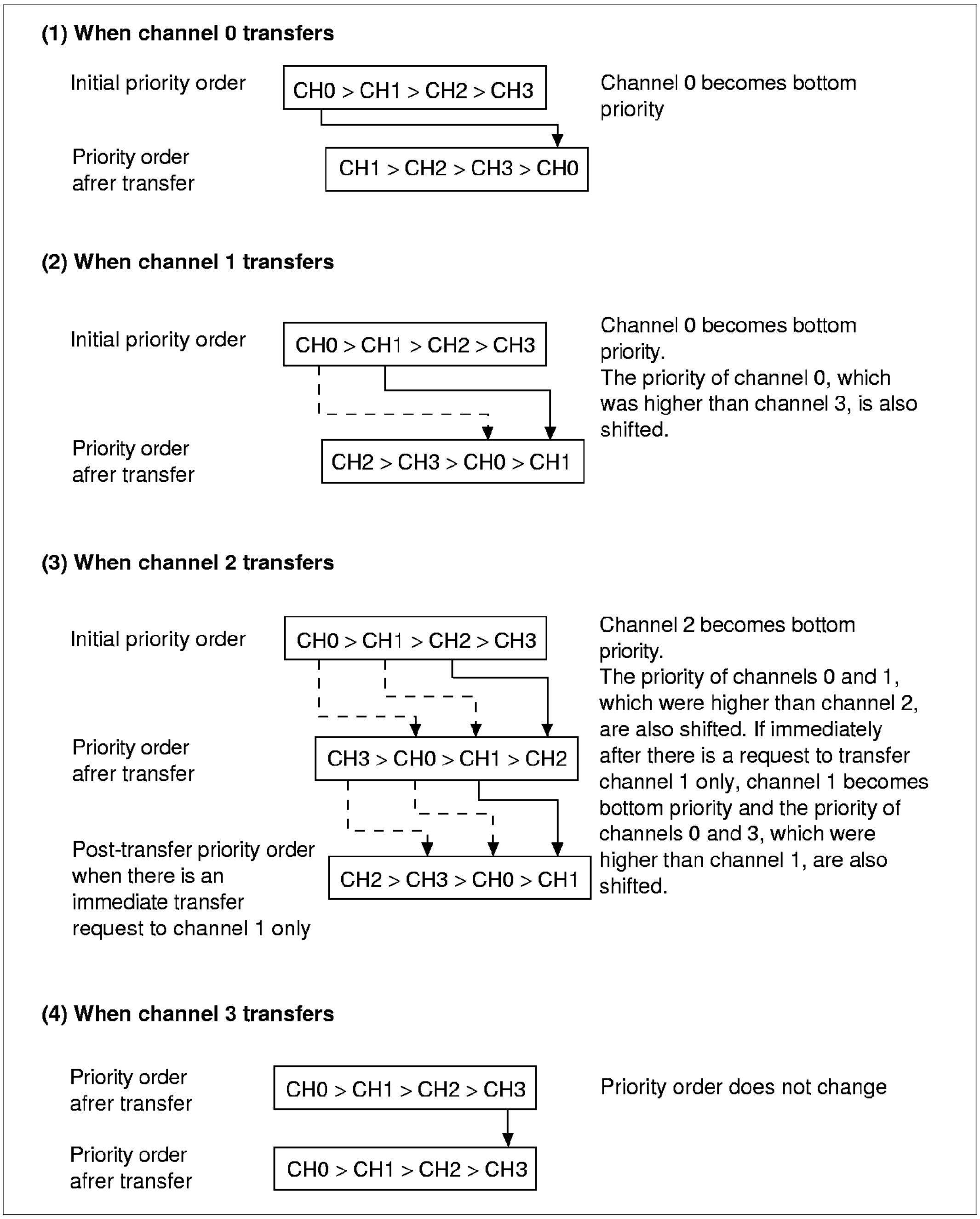

(1) When channel 0 transfers
Initial priority order
CH0 > CH1 > CH2 > CH3
Channel 0 becomes bottom priority
Priority order afrer transfer
CH1 > CH2 > CH3 > CH0
(2) When channel 1 transfers
Initial priority order
CH0 > CH1 > CH2 > CH3
Channel 0 becomes bottom priority.
The priority of channel 0, which was higher than channel 3, is also shifted.
Priority order afrer transfer
CH2 > CH3 > CH0 > CH1
(3) When channel 2 transfers
Initial priority order
CH0 > CH1 > CH2 > CH3
Channel 2 becomes bottom priority.
The priority of channels 0 and 1, which were higher than channel 2, are also shifted. If immediately after there is a request to transfer channel 1 only, channel 1 becomes bottom priority and the priority of channels 0 and 3, which were higher than channel 1, are also shifted.
Priority order afrer transfer
CH3 > CH0 > CH1 > CH2
Post-transfer priority order when there is an immediate transfer request to channel 1 only
CH2 > CH3 > CH0 > CH1
(4) When channel 3 transfers
Priority order afrer transfer
CH0 > CH1 > CH2 > CH3
Priority order does not change
Priority order afrer transfer
CH0 > CH1 > CH2 > CH3

Figure 14.3 Round-Robin Mode

Figure 14.4 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.

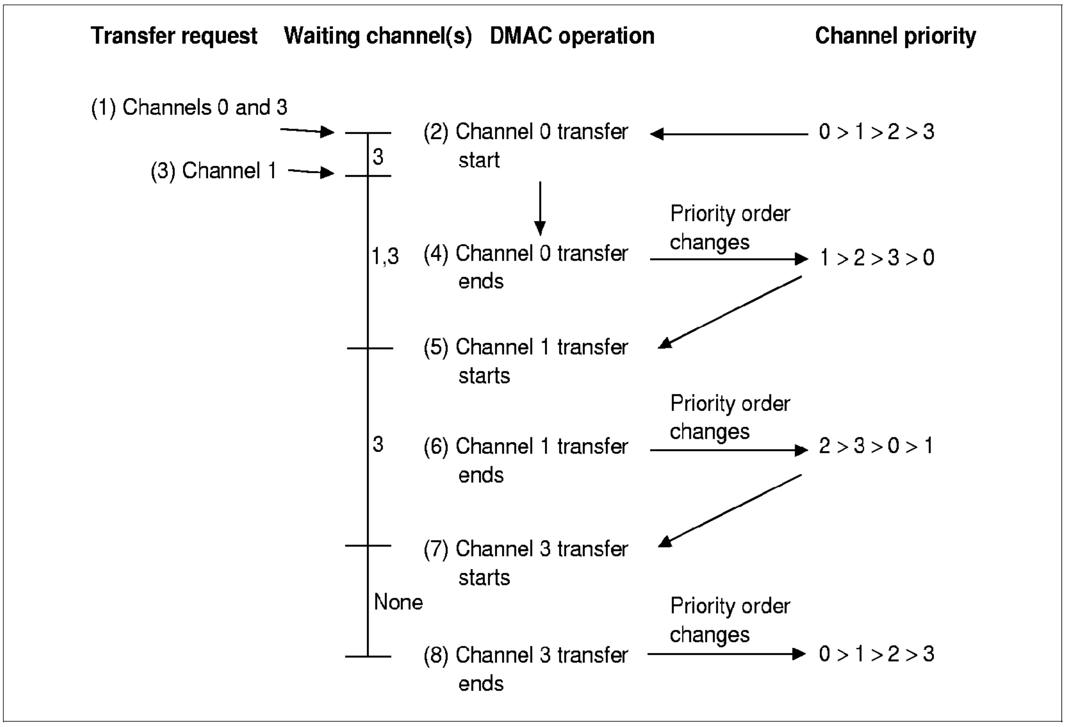


Figure 14.4 Changes in Channel Priority in Round-Robin Mode

14.3.4 DMA Transfer Types

The DMAC supports the transfers shown in table 14.5. In the dual address mode, both the transfer source address and the transfer destination address are output. The dual address mode has the direct address mode and the indirect address mode. In the direct address mode, an output address value is the data transfer target address; in the indirect address mode, the value stored in the output address, not the output address value itself, is the data transfer target address. A data transfer timing depends on the bus mode, which has cycle steal mode and burst mode.

Table 14.5Supported DMA Transfers

	Destination				
Source	**External Device with DACK**	**External Memory**	**Memory-Mapped External Device**	**On-Chip Peripheral Module**	**XY 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
XY memory	Not available	Dual	Dual	Dual	Dual

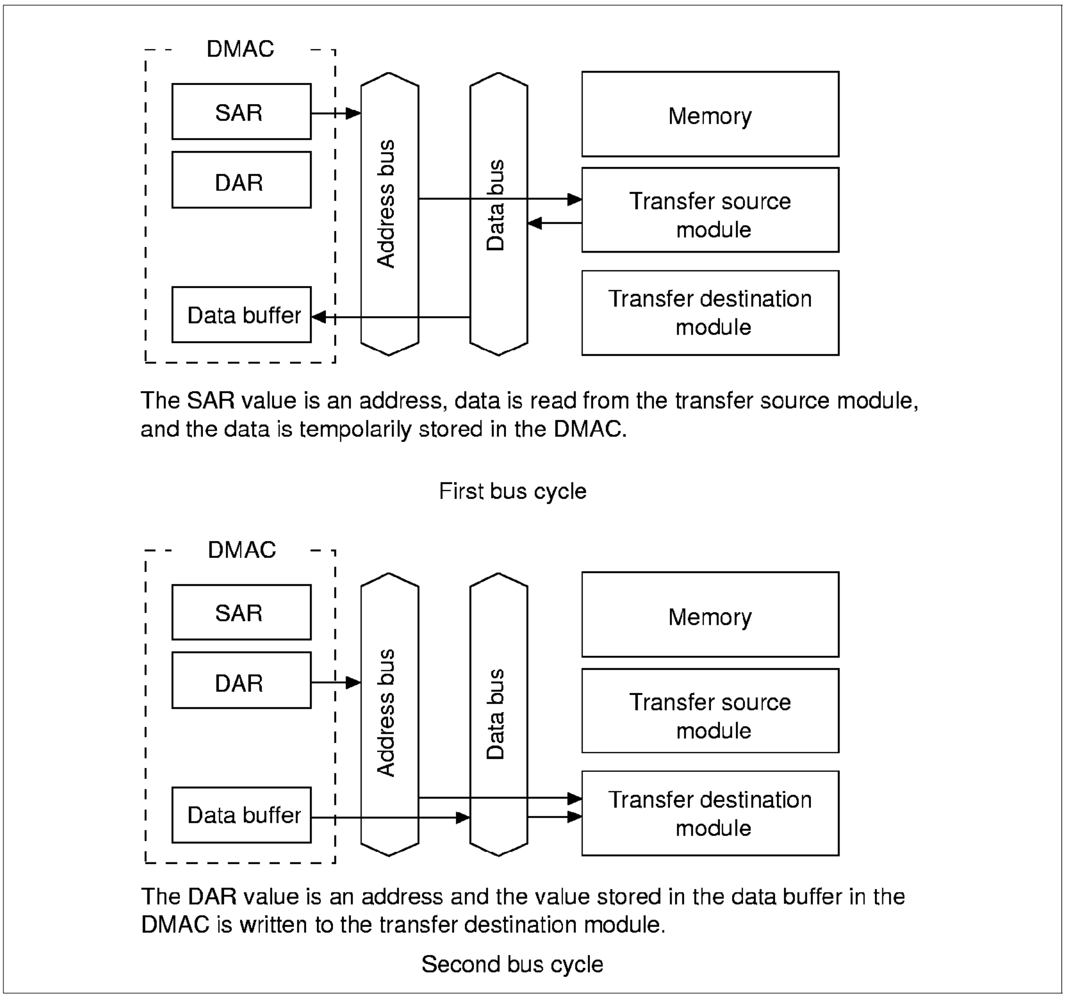
Notes: 1. Dual: Dual address mode
2. Single: Single address mode
3. The dual address mode includes the direct address mode and the indirect address mode.
4. 16-byte transfer is not available for on-chip peripheral modules.

Address Modes:

- Dual Address Mode

In the dual address mode, both the transfer source and destination are accessed (selectable) by an address. The source and destination can be located externally or internally. The dual address mode has (1) direct address transfer mode and (2) indirect address transfer mode.

(1) In the direct address transfer mode, 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 14.5, 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. Figures 14.6 to 14.8 show examples of the timing at this time.



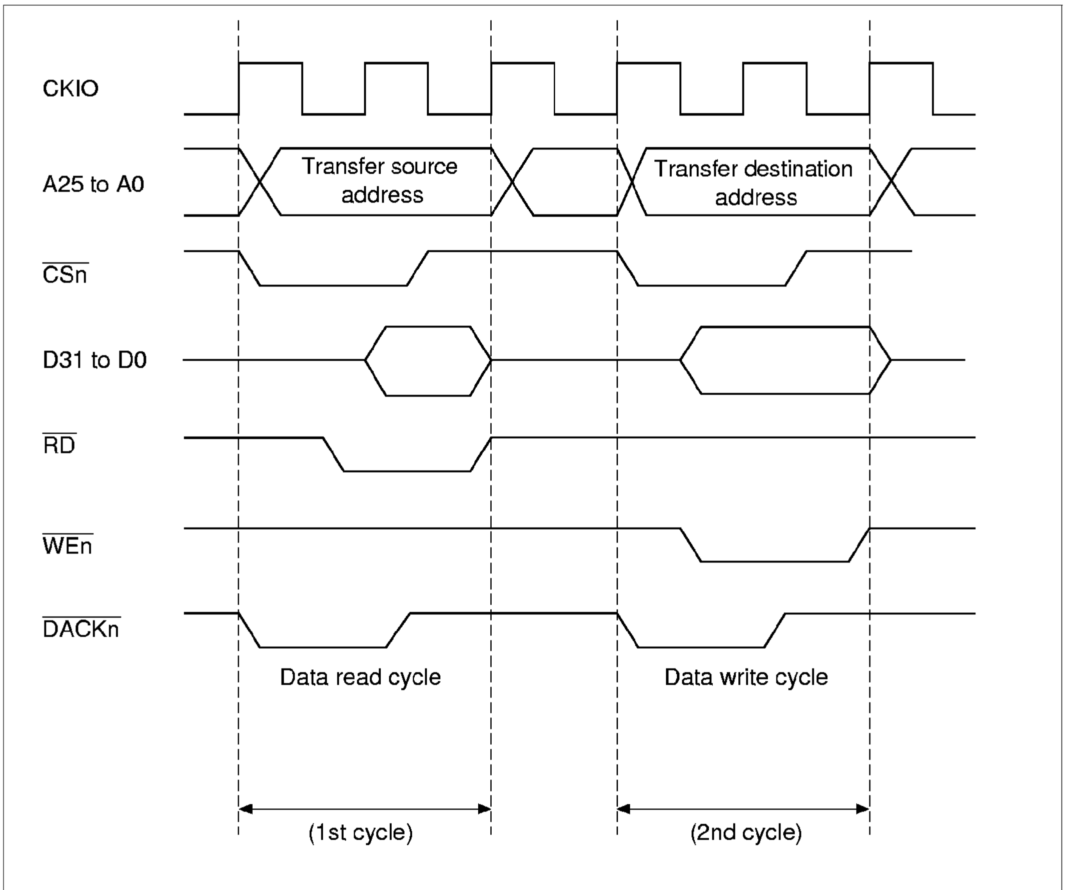
The SAR value is an address, data is read from the transfer source module, and the data is tempolarily stored in the DMAC.

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.

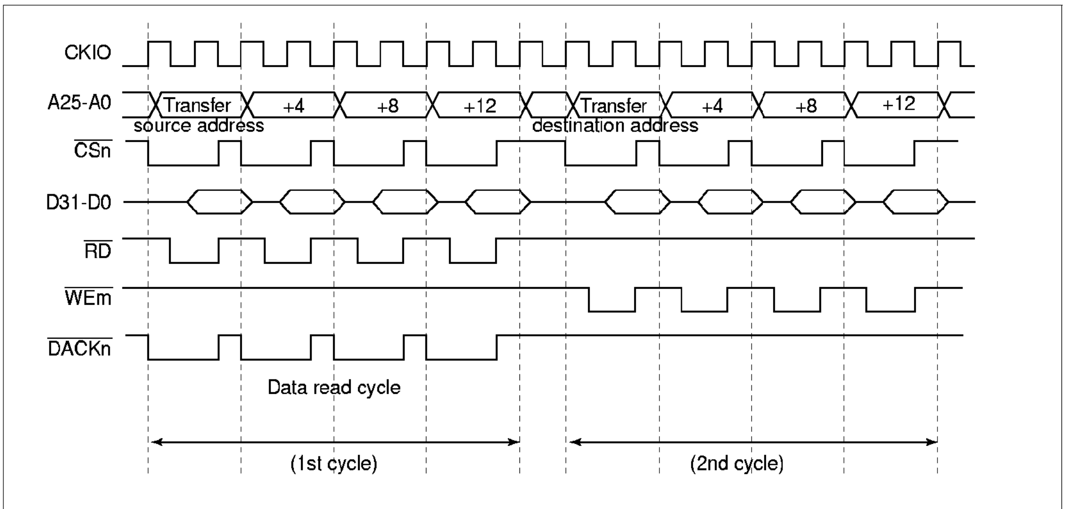
Second bus cycle

Figure 14.5 Operation of Direct Address Mode in Dual Address Mode



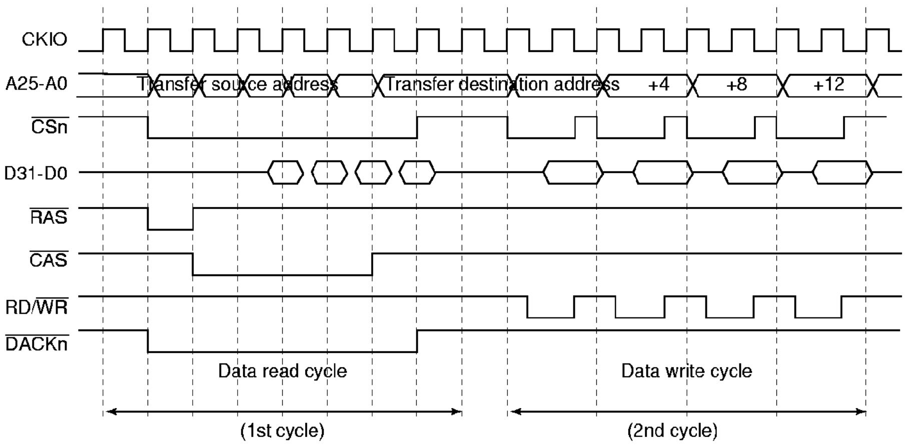
Note: Transfer between external memories, DACK output in a read cycle DACK output timing is the same as that of CSn.

Figure 14.6 Example of DMA Transfer Timing in the Direct Address Mode in the Dual Mode (Transfer Source: Ordinary Memory, Transfer Destination: Ordinary Memory)



Note: Transfer between external memories, DACK output in a read cycle DACK output timing is the same as that of $\overline{\text{CSn}}$.

Figure 14.7 Example of DMA Transfer Timing in the Direct Address Mode in the Dual Mode (16-byte Transfer, Transfer Source: Ordinary Memory, Transfer Destination: Ordinary Memory)



Note: Transfer between external memories, DACK output in a read cycle DACK output timing is the same as that of CSn.

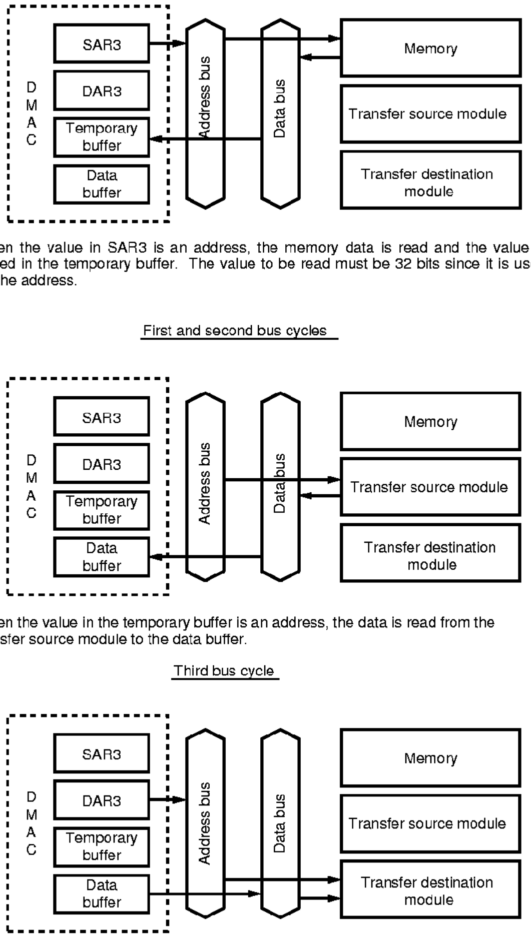
Figure 14.8 Example of DMA Transfer Timing in the Direct Address Mode in the Dual Mode (16-byte Transfer, Transfer Source: Synchronous DRAM, Transfer Destination: Ordinary Memory)

(2) In the indirect address transfer mode, the address of memory in which data to be transferred is stored is specified in the transfer source address register (SAR3) in the DMAC. 16-byte transfer is not possible. Consequently, in this mode, the address value specified in the transfer source address register in the DMAC is read first. This value is temporarily stored in the DMAC. Next, the read value is output as an address, and the value stored in that address is stored in the DMAC again. Then, the value read afterwards is written to the address specified in the transfer destination address; this completes one DMA transfer.

Figure 14.7 shows one example. In this example, the transfer destination, the transfer source, and the storage destination of the indirect address are external memories, and transfer data is 16 or 8 bits. Figure 14.8 shows an example of the transfer timing.

In this mode, one NOP cycle (CK1 cycle shown in figure 14.10) is required to output data read as an indirect address to an address bus.

If transfer data is 32 bits, third and fourth bus cycles shown in figure 14.10 is required twice for each; a total of six bus cycles and one NOP cycle are required.



When the value in SAR3 is an address, the memory data is read and the value is stored in the temporary buffer. The value to be read must be 32 bits since it is used for the address.

First and second bus cycles

When the value in the temporary buffer is an address, the data is read from the transfer source module to the data buffer.

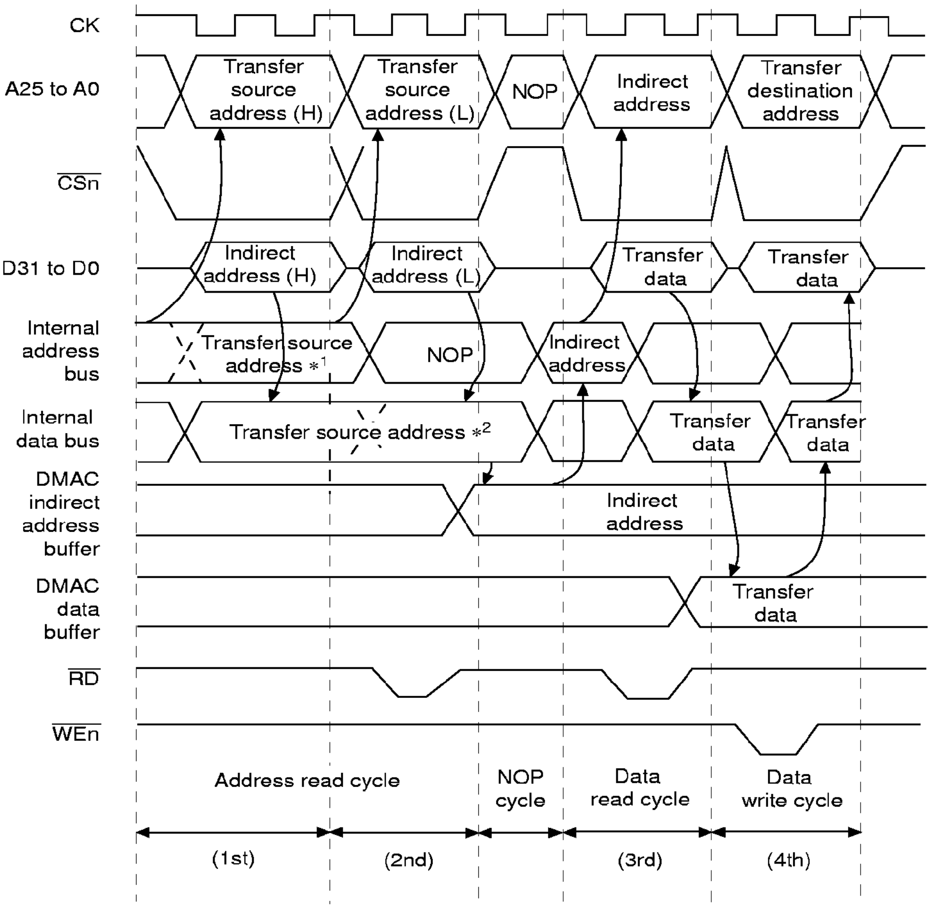
Third bus cycle

When the value in SAR3 is an address, the value in the data buffer is written to the transfer source module.

Fourth bus cycle

Note: The above description uses the memory, transfer source module, or transfer destination module; in practice, any module can be connected in the addressing space.

Figure 14.9 Operation of Indirect Address in the Dual Address Mode (When the External Memory Space has a 16-bit Width)



Transfer between external memories

Notes: 1. The internal address bus value does not change, and controlled by the port.
2. The DMAC does not fetch the value until 32-bit data is output to the internal data bus.

Figure 14.10 Example of Transfer Timing in the Indirect Address Mode in the Dual Address Mode

- 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, when the external device outputs data to the data bus, that data is written to the external memory in the same bus cycle.

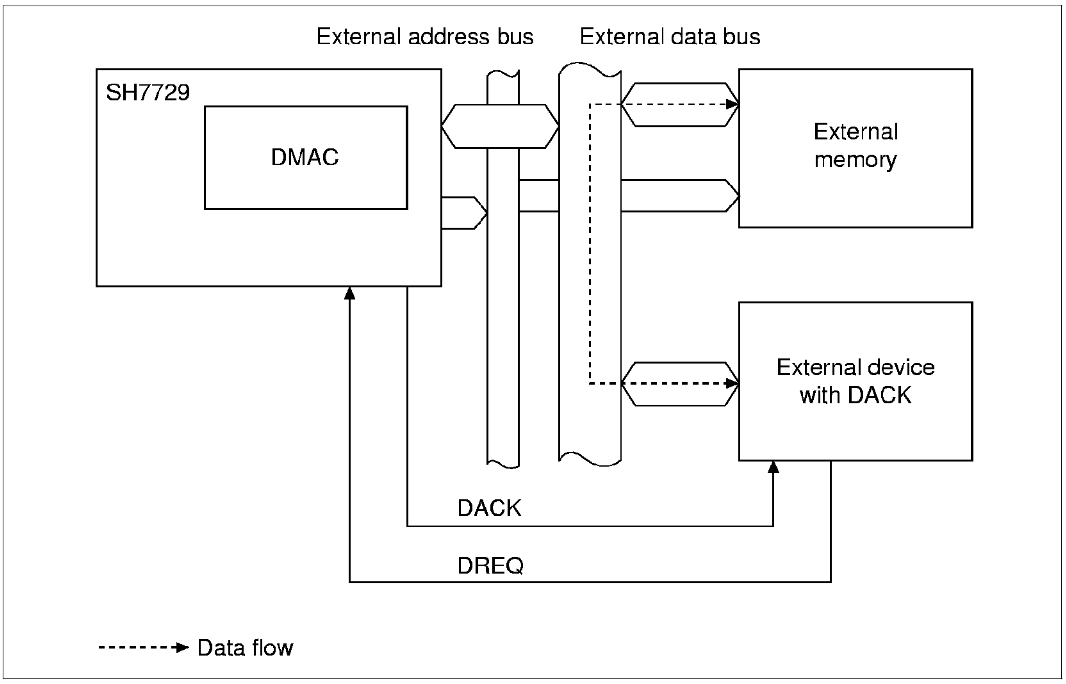
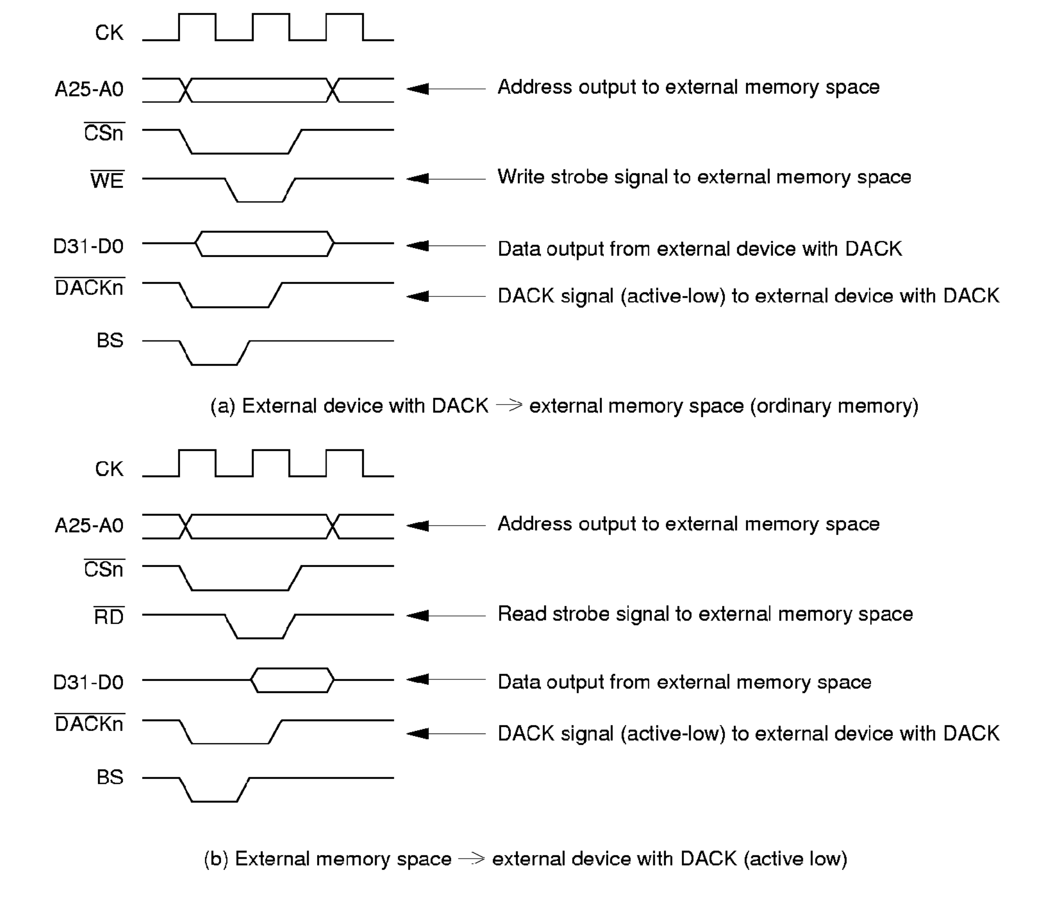


Figure 14.11 Data Flow in Single Address Mode

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 14.12 to 14.14 show examples of DMA transfer timing in single address mode.


CK
A25-A0
CSn
WE
D31-D0
DACKn
BS
Address output to external memory space
Write strobe signal to external memory space
Data output from external device with DACK
DACK signal (active-low) to external device with DACK
CK
A25-A0
CSn
RD
D31-D0
DACKn
BS
Address output to external memory space
Read strobe signal to external memory space
Data output from external memory space
DACK signal (active-low) to external device with DACK

(a) External device with DACK —> external memory space (ordinary memory)

(b) External memory space —> external device with DACK (active low)

Figure 14.12 Example of DMA Transfer Timing in Single Address Mode

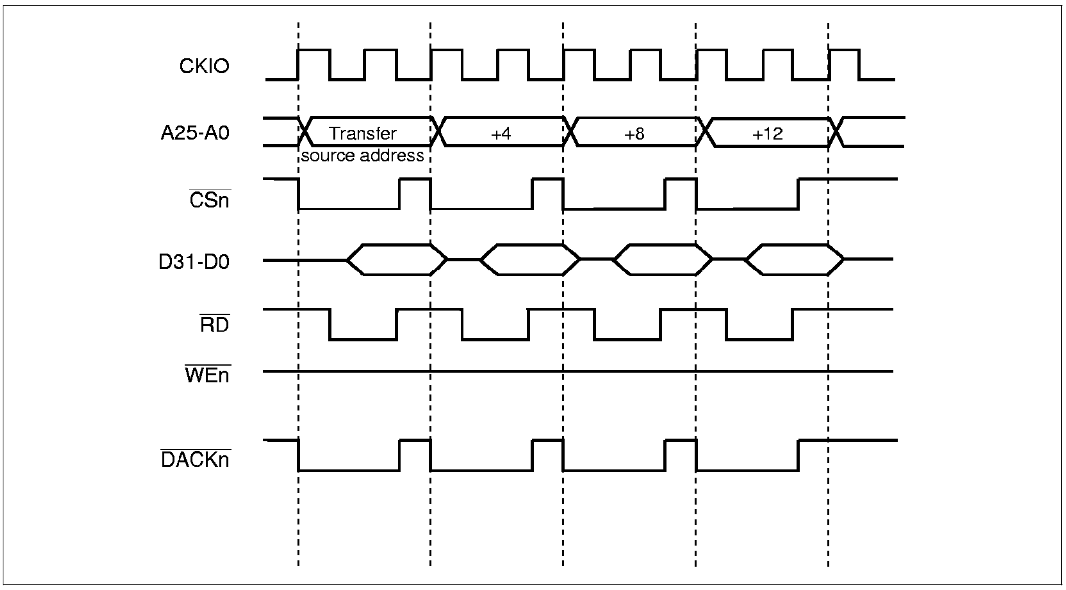


Figure 14.13 Example of DMA Transfer Timing in Single Address Mode (External Memory Space (Ordinary Memory) -> External Device with DACK)

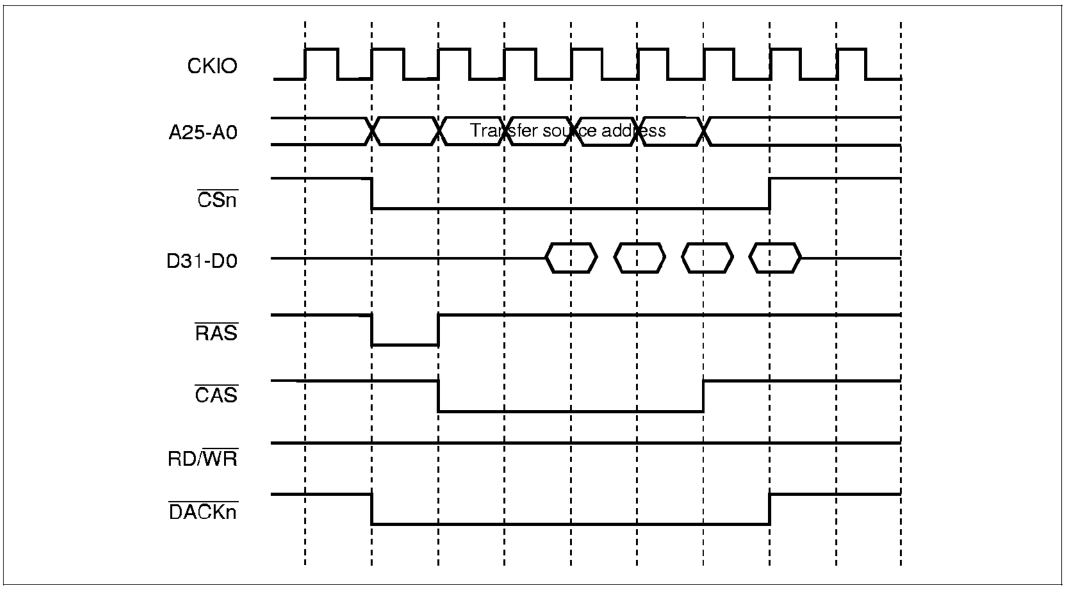


Figure 14.14 Example of DMA Transfer Timing in Single Address Mode (External Memory Space (SDRAM) -> External Device with DACK)

Bus Modes: There are two bus modes: cycle steal and burst. Select the mode in the TM bits of CHCR0–CHCR3.

- Cycle-Steal Mode

  In the cycle-steal mode, the bus right is given to another bus master after a one-transfer-unit (8-, 16-, or 32-bit unit) DMA transfer. When another transfer request occurs, the bus rights are obtained from the other bus master and a transfer is performed for one transfer unit. When that transfer ends, the bus right is passed to the other bus master. This is repeated until the transfer end conditions are satisfied.

  In the cycle-steal mode, transfer areas are not affected regardless of settings of the transfer request source, transfer source, and transfer destination. Figure 14.15 shows an example of DMA transfer timing in the cycle steal mode. Transfer conditions shown in the figure are:

  — Dual address mode
  — $\overline{\text{DREQ}}$ level detection

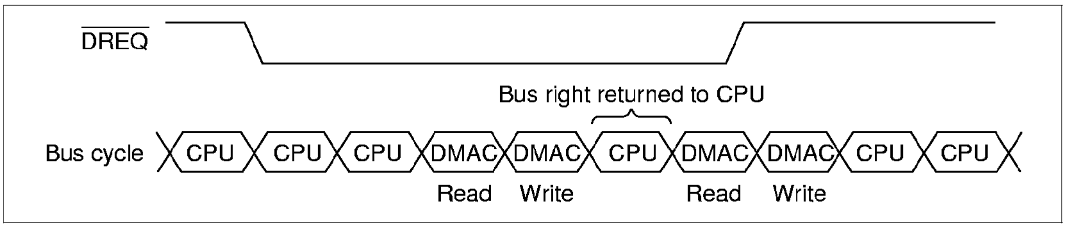


Figure 14.15 Transfer Example in the Cycle-Steal Mode

- Burst Mode

  Once the bus right is obtained, the transfer is performed continuously until the transfer end condition is satisfied. In the external request mode with low level detection of the $\overline{\text{DREQ}}$ pin, however, when the $\overline{\text{DREQ}}$ pin is driven high, the bus passes to the other bus master after the 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 when the serial communications interface (IRDA and SCI) is the transfer request source. Figure 14.16 shows a timing at this point.

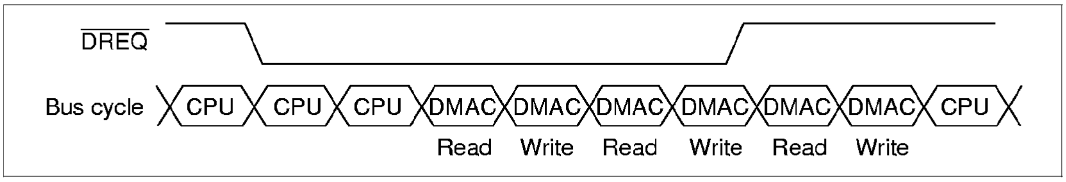


Figure 14.16 Transfer Example in the Burst Mode

Relationship between Request Modes and Bus Modes by DMA Transfer Category: Table 14.6 shows the relationship between request modes and bus modes by DMA transfer category.

Table 14.6Relationship 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	0,1
	External device with DACK and memory-mapped external device	External	B/C	8/16/32/128	0, 1
	External memory and external memory	All*1	B/C	8/16/32/128	0–3*5
	External memory and memory-mapped external device	All *1	B/C	8/16/32/128	0–3*5
	Memory-mapped external device and memory-mapped external device	All *1	B/C	8/16/32/128	0–3*5
	External memory and on-chip peripheral module	All *2	B/C*3	8/16/32*4	0–3*5
	Memory-mapped external device and on-chip peripheral module	All *2	B/C*3	8/16/32*4	0–3*5
	On-chip peripheral module and on-chip peripheral module	All *2	B/C*3	8/16/32*4	0–3*5
	X/Y memory and X/Y memory	All	B/C	8/16/32/128	0–3
	X/Y memory and memory-mapped external device	All *1	B/C	8/16/32/128	0–3
	X/Y memory and on-chip peripheral module	All *2	B/C*3	8/16/32	0–3
	X/Y memory and external memory	All	B/C	8/16/32/128	0–3
Single	External device with DACK and external memory	External	B/C	8/16/32	0, 1
	External device with DACK and memory-mapped external device	External	B/C	8/16/32	0, 1

B: Burst, C: Cycle steal

Notes: 1. External requests, auto requests and on-chip peripheral module requests are all available. For on-chip peripheral module requests, however, IrDA, SCI, and A/D converter cannot be specified as the transfer request source.

2. External requests, auto requests and on-chip peripheral module requests are all available. When the IrDA, SCI, or A/D converter is also the transfer request source, however, the transfer destination or transfer source must be the IrDA, SCI, or A/D converter, respectively.

3. If the transfer request source is the IrDA or SCI, the cycle-steal mode is only available.
4. The access size permitted when the transfer destination or source is an on-chip peripheral module register.
5. If the transfer request is an external request, channels 0 and 1 are only 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, the transfer of channel 0 will begin immediately.

At this time, if the priority is set in the fixed mode (CH0 > CH1), the channel 1 transfer will continue when the channel 0 transfer has completely finished, even if channel 0 is operating in the cycle steal mode or in the burst mode.

If the priority is set in the round-robin mode, channel 1 will begin operating again after channel 0 completes the transfer of one transfer unit, even if channel 0 is in the cycle steal mode or in the burst mode. The bus will then switch between the two in the order channel 1, channel 0, channel 1, channel 0.

Even if the priority is set in the fixed mode or in the round-robin mode, it will not give the bus to the CPU since channel 1 is in the burst mode. This example is illustrated in figure 14.17.

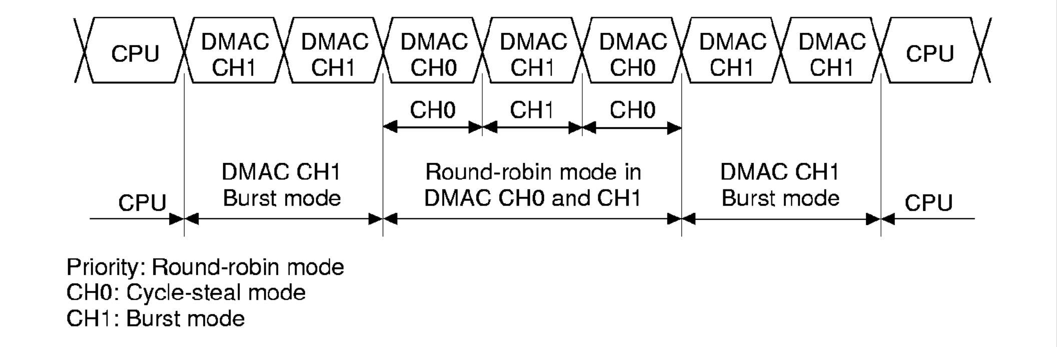


Figure 14.17 Bus State when Multiple Channels Are Operating

14.3.5 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 13, Bus State Controller.

DREQ Pin Sampling Timing: In external request mode, the DREQ pin is sampled by clock pulse (CKIO) falling edge or low level detection. When DREQ input is detected, a DMAC bus cycle is generated and DMA transfer performed, at the earliest, three states later.

The second and subsequent DREQ sampling operations are started two cycles after the first sample.

Operation

- Cycle-Steal Mode

  In cycle-steal mode, the DREQ sampling timing is the same regardless of whether level or edge detection is used.

  For example, in figure 14.18 (cycle-steal mode, level detection), DMAC transfer begins, at the earliest, three cycles after the first sampling is performed. The second sampling is started two cycles after the first. If DREQ is not detected at this time, sampling is performed in each subsequent cycle.

  Thus, DREQ sampling is performed one step in advance. The third sampling operation is not performed until the idle cycle following the end of the first DMA transfer.

  The above conditions are the same whatever the number of CPU transfer cycles, as shown in figure 14.19.

  DACK is output in a read in the example in figure 14.18, and in a write in the example in figure 14.19. In both cases, DACK is output for the same duration as CSn.

- Burst Mode, Level Detection

  In the case of burst mode with level detection, the DREQ sampling timing is the same as in the cycle-steal mode.

  For example, in figure 14.20, DMAC transfer begins, at the earliest, three cycles after the first sampling is performed. The second sampling is started two cycles after the first. Subsequent sampling operations are performed in the idle cycle following the end of the DMA transfer cycle.

  In the burst mode, also, the DACK output period is the same as in the cycle-steal mode.

- Burst Mode, Edge Detection

  In the case of burst mode with edge detection, $\overline{\text{DREQ}}$ sampling is only performed once.

  For example, in figure 14.21, DMAC transfer begins, at the earliest, three cycles after the first sampling is performed. After this, DMAC transfer is executed continuously until the number of data transfers set in the DMATCR register have been completed. $\overline{\text{DREQ}}$ is not sampled during this time.

  To restart DMA transfer after it has been suspended by an NMI, first clear NMIF, then input an edge request again.

  In the burst mode, also, the $\overline{\text{DACK}}$ output period is the same as in the cycle-steal mode.

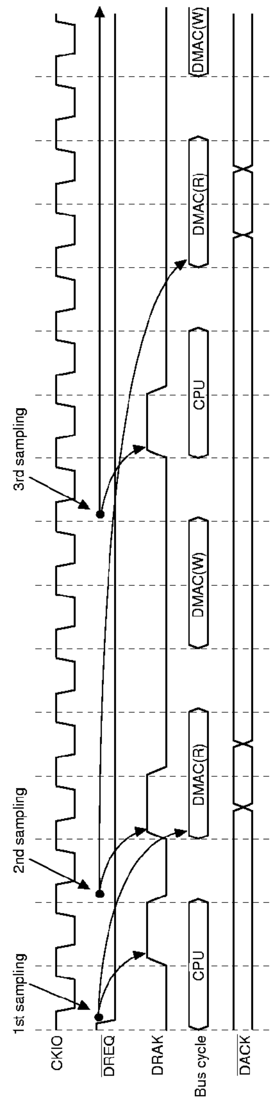


Figure 14.18 Cycle-Steal Mode, Level Input (CPU Access: 2 Cycles)

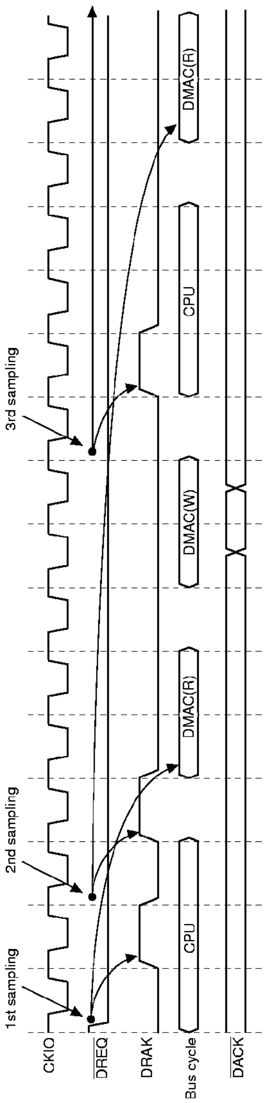


Figure 14.19 Cycle-Steal Mode, Level Input (CPU Access: 3 Cycles)

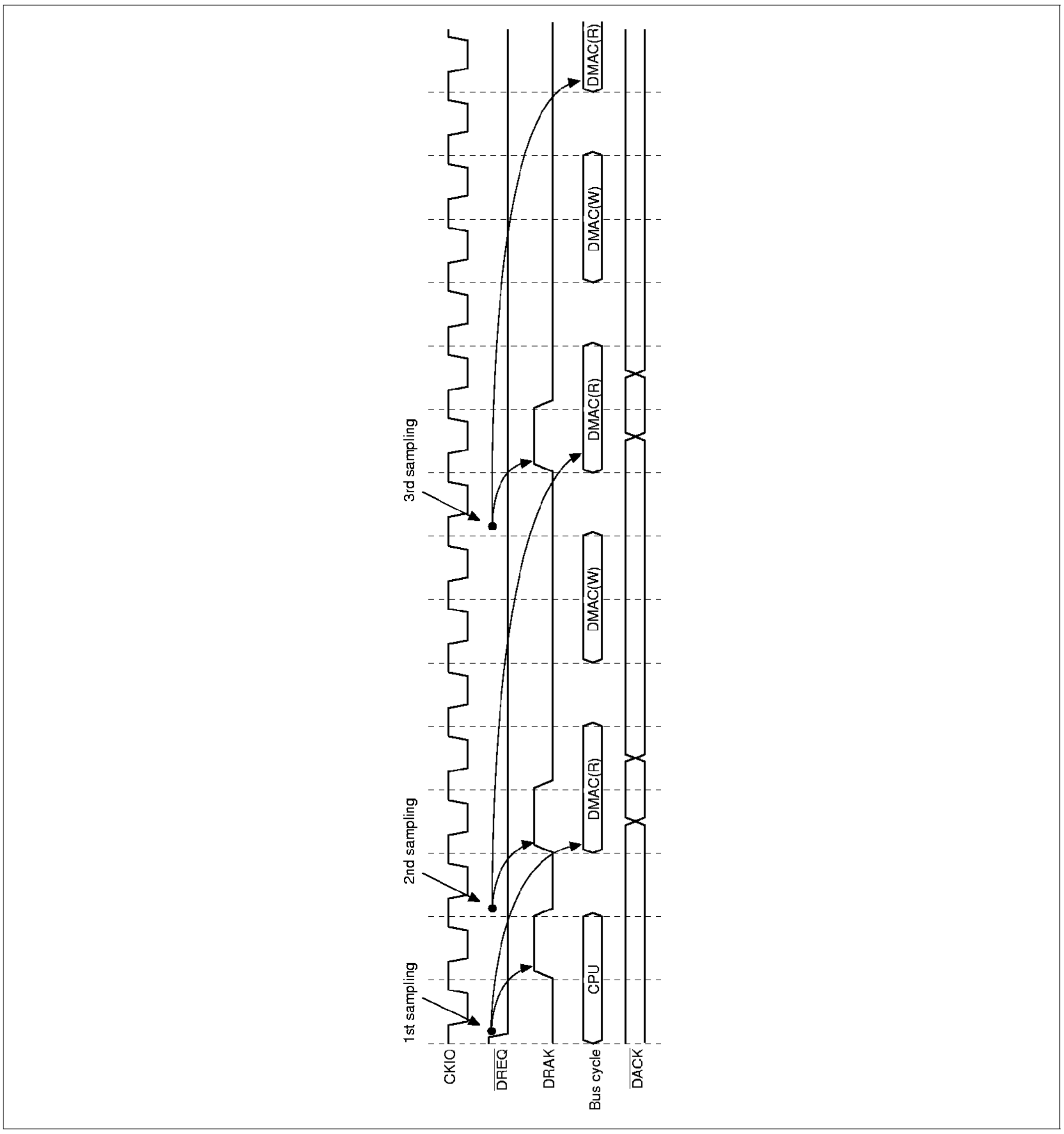


Figure 14.20 Burst Mode, Level Input

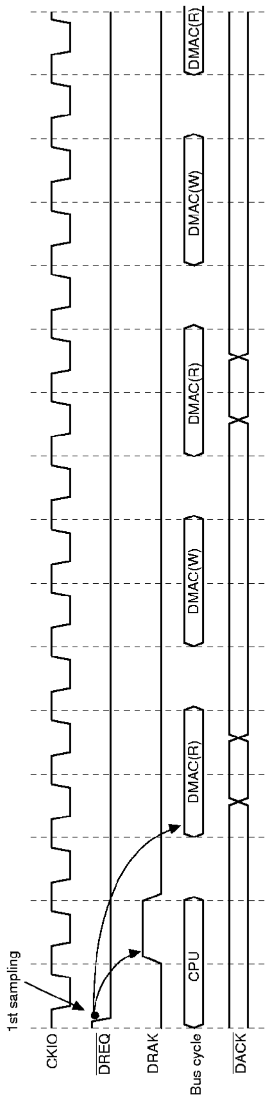


Figure 14.21 Burst Mode, Edge Input

14.3.6 Source Address Reload Function

Channel 2 includes a reload function, in which the value returns to the value set in the source address register (SAR2) for each four transfers by setting the RO bit in CHCR2. 16-byte transfer cannot be used. Figure 14.22 shows this operation. Figure 14.23 shows the timing chart of the source address reload function, which is under the following conditions: burst mode, auto request, 16-bit transfer data size, SAR2 count-up, DAR2 fixed, reload function on, and usage of only channel 2.

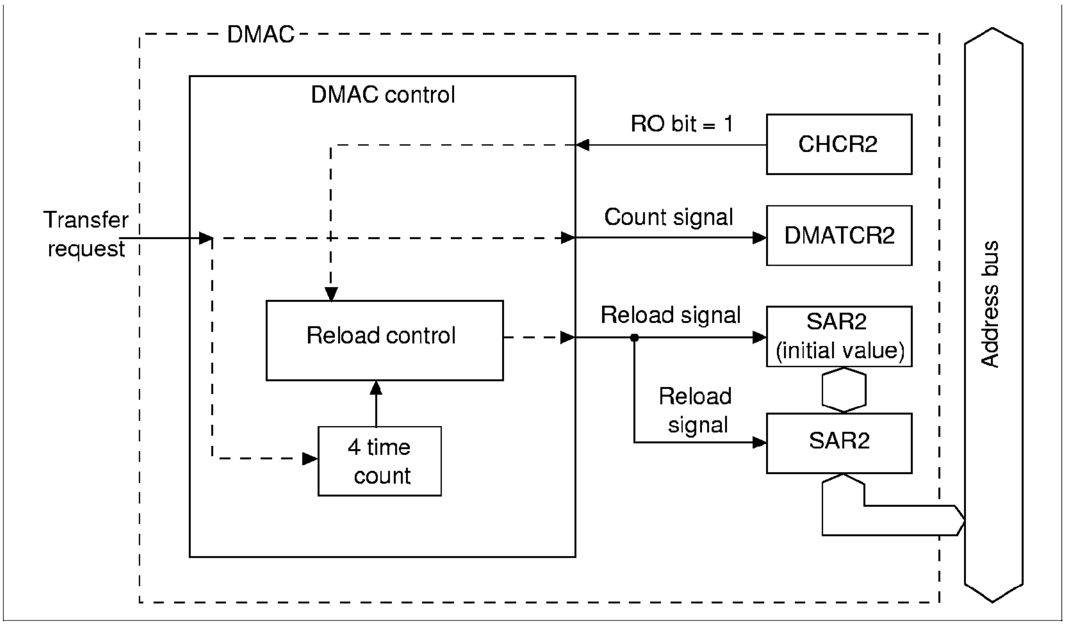


Figure 14.22 Source Address Reload Function Diagram

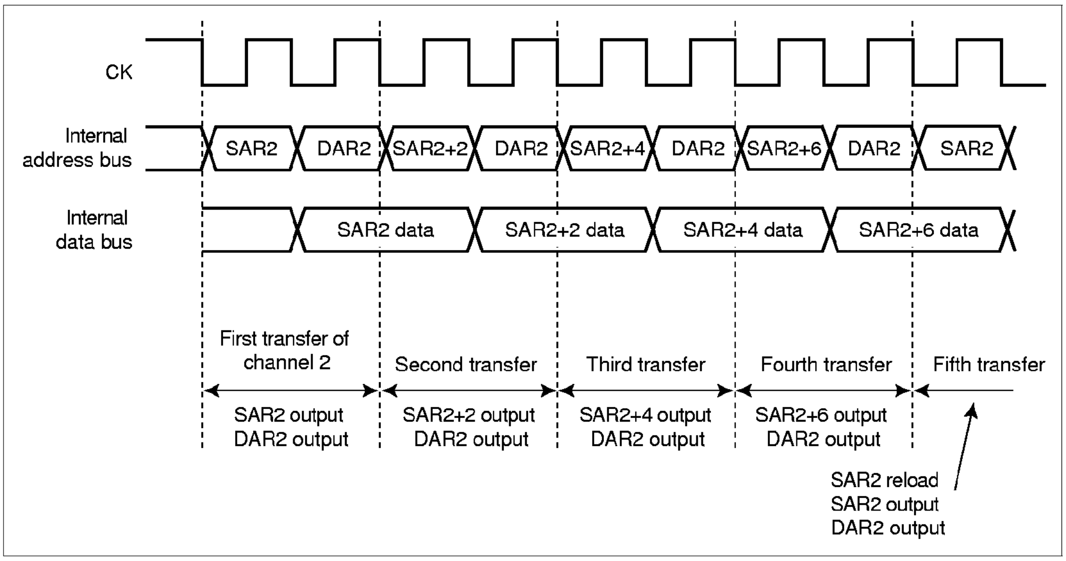


Figure 14.23 Timing Chart of Source Address Reload Function

Even if the transfer data size is 8, 16, or 32 bits, a reload function can be executed.

DMATCR2, which specifies a transfer count, increments 1 each time a transfer ends regardless of whether a reload function is on or off. Consequently, be sure to specify the value multiple of four in DMATCR2 when the reload function is on. Specifying other values does not guarantee the operation.

Though the counters that count transfers of four times for the reload function are reset by clearing the DME bit in DMAOR or the DE bit in CHCR2, by setting the transfer end flag (TE bit in CHCR2), by inputting NMI, besides by reset or standby, the SAR2, DAR2, DMATCR2 registers are not reset. Therefore, if these sources are generated, the counters that are initialized and are not initialized exist in the DMAC; malfunction will be caused by restarting the DMAC in that state. Consequently, if these sources occur except for setting the TE bit during the usage of the reload function, set SAR2, DAR2, and DMATCR2 again.

14.3.7 DMA Transfer Ending Conditions

The DMA transfer ending conditions vary for individual channels ending and all channels ending together. At transfer end, the following conditions are applied except the case where the value set in the DMA transfer count register (DMATCR) reaches 0.

(a) Cycle-steal mode (external request, internal request, and auto request)

When the transfer ending conditions are satisfied, DMAC transfer request acceptance is suspended. The DMAC stops operating after completing the number of transfers that it has accepted until the ending conditions are satisfied.

In the cycle-steal mode, the operation is the same regardless of whether the transfer request is detected by the level or at the edge.

(b) Burst mode, edge detection (external request, internal request, and auto request)

The timing from the point where the ending conditions are satisfied to the point where the DMAC stops operating does not differ from that in cycle steal mode. In the edge detection in the burst mode, though only one transfer request is generated to start up the DMAC, stop request sampling is performed in the same timing as transfer request sampling in the cycle-steal mode. As a result, the period when stop request is not sampled is regarded as the period when transfer request is generated, and after performing the DMA transfer for this period, the DMAC stops operating.

(c) Burst mode, level detection (external request)

Same as described in (a).

(d) Bus timing when transfers are suspended

The transfer is suspended when one transfer ends. Even if transfer ending conditions are satisfied during read in the direct address transfer in the dual address mode, the subsequent write process is executed, and after the transfer in (a) to (c) above has been executed, DMAC operation suspends.

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

- When DMATCR is 0: When the DMATCR value becomes 0 and the corresponding channel's DMA transfer ends, the transfer end flag bit (TE) is set in the CHCR. If the IE (interrupt enable) bit has been set, a DMAC interrupt (DEI) is requested to the CPU. This transfer ending does not apply to (a) to (d) described above.

- When DE of CHCR is 0: Software can halt a DMA transfer by clearing the DE bit in the channel's CHCR. The TE bit is not set when this happens. This transfer ending applies to (a) to (d) described above.

Conditions for Ending All Channels Simultaneously: Transfers on all channels end when 1) the NMIF (NMI flag) bit is set to 1 in the DMAOR, or 2) when the DME bit in the DMAOR is cleared to 0.

- Transfers ending when the NMIF bit is set to 1 in DMAOR: When an NMI interrupt occurs, the NMIF bit is set to 1 in the DMAOR and all channels stop their transfers according to the conditions in (a) to (d) described above, and pass the bus right to other bus masters. Consequently, even if the NMI bit is set to 1 during transfer, the SAR, DAR, DMATCR are updated. The TE bit is not set. To resume the transfers after NMI interrupt exception processing, clear the NMIF bit to 0. At this time, if there are channels that should not be restarted, clear the corresponding DE bit in the CHCR.

- Transfers ending when DME is cleared to 0 in DMAOR: Clearing the DME bit to 0 in the DMAOR forcibly aborts the transfers on all channels. The TE bit is not set. All channels aborts their transfers according to the conditions in (a) to (d) in 14.3.7, as in NMI interrupt generation. In this case, the values in SAR, DAR, and DMATCR are also updated.

14.4 Compare Match Timer (CMT)

14.4.1 Overview

DMAC has an on-chip compare match timer (CMT) to generate DMA transfer request. The CMT has 16-bit counter.

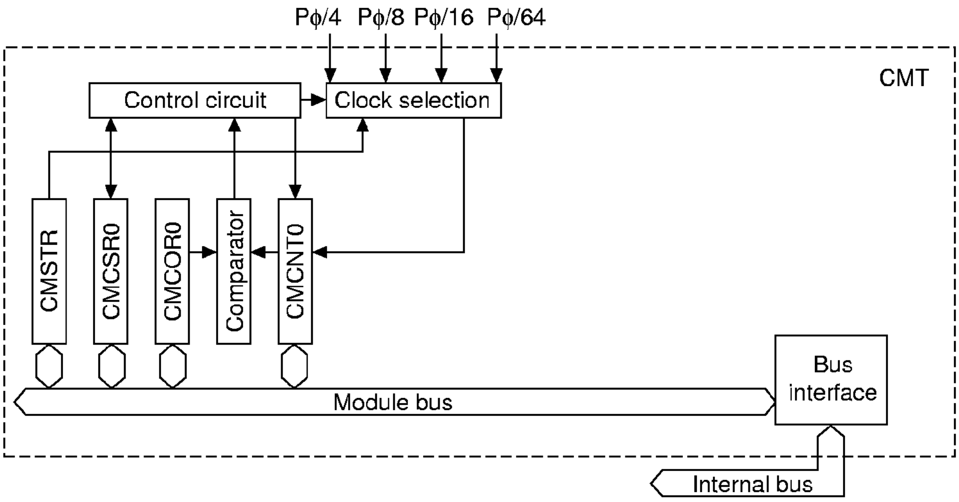
Features

The CMT has the following features:

- Four types of counter input clock can be selected
  — One of four internal clocks (Pϕ/4, Pϕ/8, Pϕ/16, Pϕ/64) can be selected.
- Generate DMA transfer request when compare match occurs.

Block Diagram

Figure 14.24 shows a CMT block diagram.



CMSTR: Compare match timer start register
CMCSR: Compare match timer control/status register
CMCOR: Compare match timer constant register
CMCNT: Compare match timer counter

Figure 14.24 CMT Block Diagram

Register Configuration

Table 14.7 summarizes the CMT register configuration.

Table 14.7Register Configuration

Name	Abbreviation	R/W	Initial Value	Address	Access Size (Bits)
Compare match timer start register	CMSTR	R/(W)	H'0000	H'4000070	8, 16, 32
Compare match timer control/status register 0	CMCSR0	R/(W)*	H'0000	H'4000072	8, 16, 32
Compare match counter 0	CMCNT0	R/W	H'0000	H'4000074	8, 16, 32
Compare match constant register 0	CMCOR0	R/W	H'FFFF	H'4000076	8, 16, 32

Note: The only value that can be written to CMF bits in CMCSR0 is a 0 to clear the flags.

14.4.2 Register Descriptions

Compare Match Timer Start Register (CMSTR)

The compare match timer start register (CMSTR) is a 16-bit register that selects whether to operate or halt the channel 0 and channel 1 counters (CMCNT). It is initialized to H'0000 by resets. It retains its previous value in standby mode.

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

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	STR0
Initial value:	0	0	0	0	0	0	0	0
R/W:	—	—	—	—	—	—	R/W	R/W

Bits 15–2—(Reserved): These bits always read as 0 and cannot be modified.

Bit 1 (Reserved): This bit can be read or written. Write 0 when writing.

Bit 0—(Count start 0 (STR0)): Selects whether to operate or halt compare match timer counter 0.

Bit 0: STR0	Description
0	CMCNT0 count operation halted (Initial value)
1	CMCNT0 count operation

Compare Match Timer Control/Status Register (CMCSR)

The compare match timer control/status register (CMCSR) is a 16-bit register that indicates the occurrence of compare matches, sets the enable/disable of interrupts, and establishes the clock used for incrementation. It is initialized to H'0000 by resets. It retains its previous value in standby mode.

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

Bit:	7	6	5	4	3	2	1	0
	CMF	—	—	—	—	—	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R/W	—	—	—	—	R/W	R/W

Note: The only value that can be written is 0 to clear the flag.

Bits 15–8 and 5–2—(Reserved): These bits always read as 0 and cannot be modified.

Bit 7—(Compare match flag (CMF)): This flag indicates whether or not the CMCNT and CMCOR values have matched.

Bit 7: CMF	Description
0	CMCNT and CMCOR values have not matched (Initial value) Clear condition: Write 0 to CMF after reading CMF = 1
1	CMCNT and CMCOR values have matched

Bit 6 (Reserved): This bit can be read or written. Write 0 when writing.

Bits 1, 0—(Clock select 1, 0 (CKS1, CKS0)): These bits select the clock input to the CMCNT from among the four internal clocks obtained by dividing the system clock (Pϕ). When the STR bit of the CMSTR is set to 1, the CMCNT begins incrementing with the clock selected by CKS1 and CKS0.

Bit 1: CKS1	Bit 0: CKS0	Description
0	0	P ϕ/4 (Initial value)
	1	P ϕ/8
1	0	P ϕ/16
	1	P ϕ/64

Compare Match Counter (CMCNT)

The compare match counter (CMCNT) is a 16-bit register used as an up-counter.

When an internal clock is selected with the CKS1 and CKS0 bits of the CMCSR register and the STR bit of the CMSTR is set to 1, the CMCNT begins incrementing with that clock. When the CMCNT value matches that of the compare match constant register (CMCOR), the CMCNT is cleared to H'0000 and the CMF flag of the CMCSR is set to 1.

The CMCNT is initialized to H'0000 by resets. It retains its previous value in standby mode.

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

Bit:	7	6	5	4	3	2	1	0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Compare Match Constant Register (CMCOR)

The compare match constant register (CMCOR) is a 16-bit register that sets the compare match period with the CMCNT.

The CMCOR is initialized to H'FFFF by resets. It retains its previous value in standby mode.

Bit:	15	14	13	12	11	10	9	8
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

14.4.3 Operation

Period Count Operation

When an internal clock is selected with the CKS1, CKS0 bits of the CMCSR register and the STR bit of the CMSTR is set to 1, the CMCNT begins incrementing with the selected clock. When the CMCNT counter value matches that of the compare match constant register (CMCOR), the CMCNT counter is cleared to H'0000 and the CMF flag of the CMCSR register is set to 1. The CMCNT counter begins counting up again from H'0000.

Figure 14.25 shows the compare match counter operation.

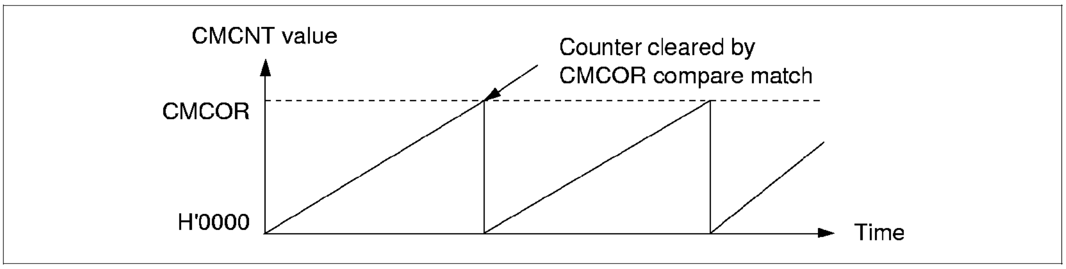


Figure 14.25 Counter Operation

CMCNT Count Timing

One of four clocks (Pϕ/4, Pϕ/8, Pϕ/16, Pϕ/64) obtained by dividing the clock (Pϕ) can be selected by the CKS1 and CKS0 bits of the CMCSR. Figure 14.26 shows the timing.

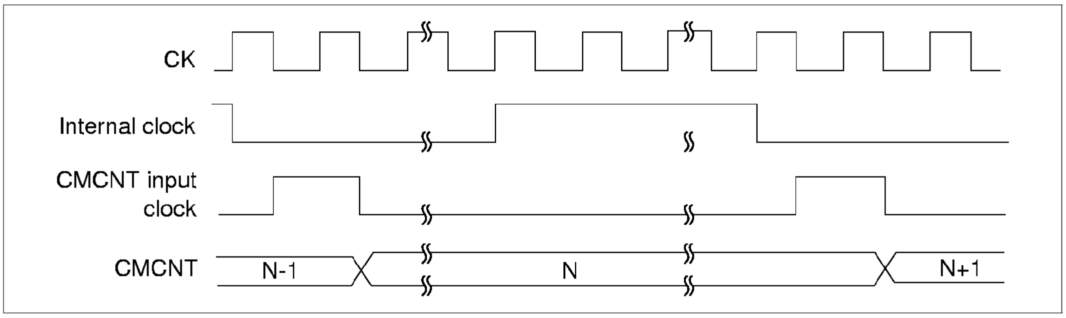


Figure 14.26 Count Timing

14.4.4 Compare Match

Compare Match Flag Set Timing

The CMF bit of the CMCSR register is set to 1 by the compare match signal generated when the CMCOR register and the CMCNT counter match. The compare match signal is generated upon the final state of the match (timing at which the CMCNT counter matching count value is updated). Consequently, after the CMCOR register and the CMCNT counter match, a compare match signal will not be generated until a CMCNT counter input clock occurs. Figure 14.27 shows the CMF bit set timing.

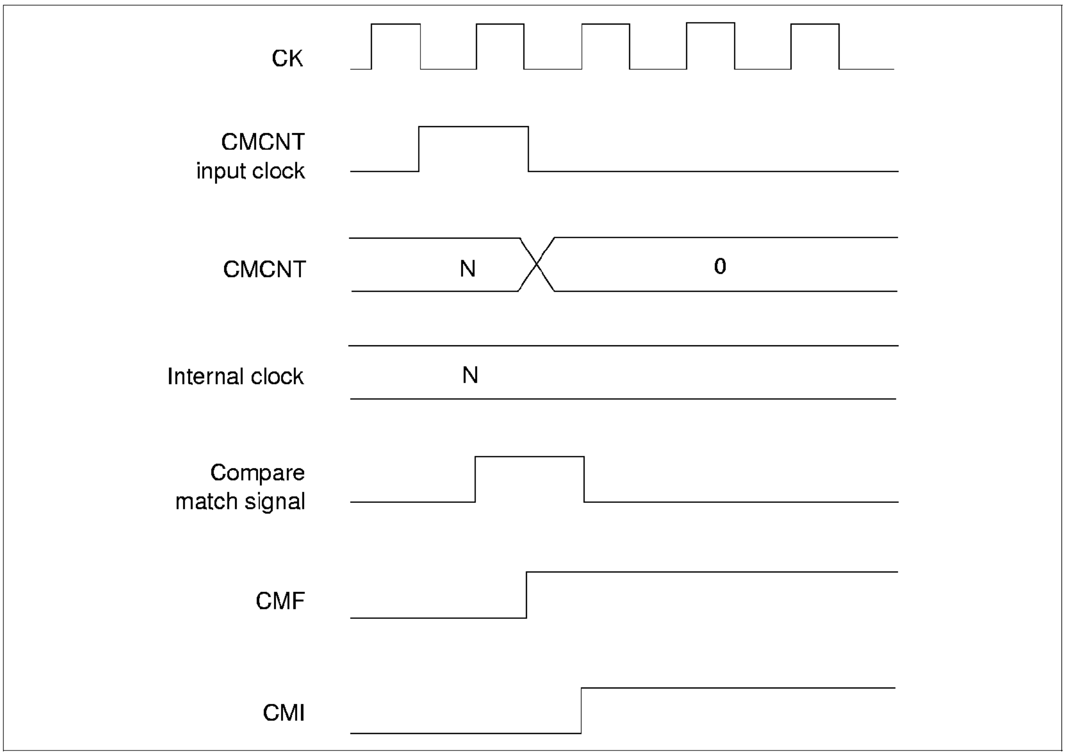


Figure 14.27 CMF Set Timing

Compare Match Flag Clear Timing

The CMF bit of the CMCSR register is cleared by writing 0 to it after reading 1. Figure 14.28 shows the timing when the CMF bit is cleared by the CPU.

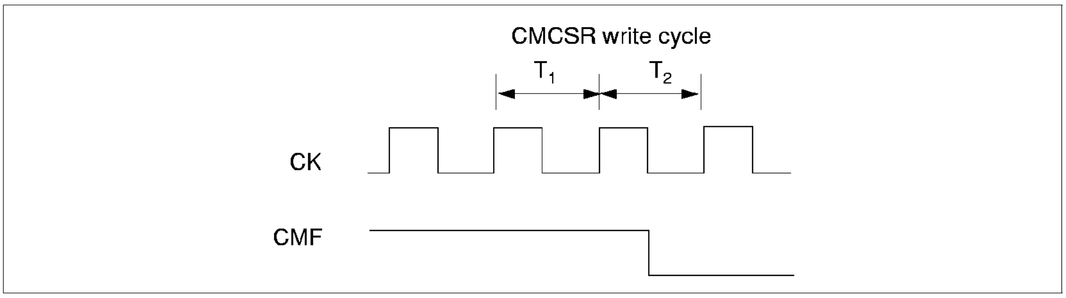


Figure 14.28 Timing of CMF Clear by the CPU

14.5 Examples of Use

14.5.1 Example of DMA Transfer between On-Chip IRDA and External Memory

In this example, receive data of on-chip IRDA is transferred to external memory using DMAC channel 3. Table 14.8 shows the transfer conditions and register settings. In addition, it is recommended that the trigger of the number of receive FIFO data in IRDA is set to 1 (RTRG1 = RTRG0 = 0 in SCFCR).

Table 14.8Transfer Conditions and Register Settings for Transfer between On-Chip SCI and External Memory

Transfer Conditions	Register	Setting
Transfer source: RDR1 of on-chip IRDA	SAR3	H'0400014A
Transfer destination: external memory	DAR3	H'00400000
Number of transfers: 64	DMATCR3	H'00000040
Transfer source address: fixed	CHCR3	H'00004B05
Transfer destination address: incremented		
Transfer request source: IRDA (RXI1)		
Bus mode: cycle steal		
Transfer unit: byte		
Interrupt request generated at end of transfer		
Channel priority order: 0 > 2 > 3 > 1	DMAOR	H'0101

14.5.2 Example of DMA Transfer between AD0 and External Memory (Address Reload on)

In this example, DMA transfer is performed between channel 0 of the on-chip AD converter (transfer source) and the external memory (transfer destination) with address reload function on. Table 14.9 shows the transfer conditions and register settings.

Table 14.9Transfer Conditions and Register Settings for Transfer between On-Chip Channel 2 of AD converter and External Memory

Transfer Conditions	Register	Setting
Transfer source: on-chip AD converter	SAR2	H'04000080
Transfer destination: external memory	DAR2	H'00400000
Number of transfers: 128 (reloading 32 times)	DMATCR2	H'00000080
Transfer source address: incremented	CHCR2	H'00089E35
Transfer destination address: decremented		
Transfer request source: AD1		
Bus mode: burst		
Transfer unit: long word		
Interrupt request generated at end of transfer		
Channel priority order: 0 > 2 > 3 > 1	DMAOR	H'0101

When the address reload function is on, the value set in SAR returns to the initially set value at each four transfers. In this example, when an interrupt request is generated from AD converter, byte data is read from the register in address H'04000080 in AD converter , and it is written to external memory address H'00400000. Since longword data has been transferred, the values in SAR and DAR are H'FFFF83E4 and H'00400004, respectively. The bus right is maintained and data transfers are successively performed because this transfer is in the burst mode.

After four transfers end, fifth and sixth transfers are performed if the address reload function is off, and the value in SAR is incremented from H'0400008C, H'04000090, H'04000094,.... If the address reload function is on, the DMA transfer stops after the fourth transfer ends, the bus request signal to the CPU is cleared. At this time, the value stored in SAR is not incremented from H'0400008C to H'04000090, but returns to the initially set value H'04000080. The value in DAR continues being incremented regardless of whether the address reload function is on or off.

As a result, the values in the DMAC are as shown in table 14.10 when the fourth transfer ends, depending on whether the address reload function is on or off.

Table 14.10 Values in the DMAC after the Fourth Transfer Ends

Items	Address reload on	Address reload off
SAR	H'04000080	H'04000090
DAR	H'003FFFFC	H'003FFFFC
DMATCR	H'0000007C	H'0000007C
Bus right	Released	Held
DMAC operation	Stops	Keeps operating
Interrupt	Not generated	Not generated
Transfer request source flag clear	Executed	Not executed

Notes: 1. An interrupt is generated regardless of whether the address reload function is on or off, if transfers are executed until the value in DMATCR reaches 0 and the IE bit in CHCR has been set to 1.

2. The transfer request source flag is cleared regardless of whether the address reload function is on or off, if transfers are executed until the value in DMATCR reaches 0.

3. Specify the burst mode to use the address reload function. This function may not be correctly executed in the cycle steal mode.

4. Set the value multiple of four in DMATCR to use the address reload function. This function may not be correctly executed if other values are specified.

14.5.3 Example of DMA Transfer between AD0 and External Memory (Address Reload on)

In this example, DMA transfer is performed between channel 0 of the on-chip A/D converter (transfer source) and the external memory (transfer destination) with address reload function on. Table 14.11 shows the transfer conditions and register settings.

Table 14.11 Transfer Conditions and Register Settings for Transfer between On-Chip Channel 2 of A/D converter and External Memory

Transfer Conditions	Register	Setting
Transfer source: on-chip A/D converter	SAR2	H'04000080
Transfer destination: on-chip memory	DAR2	H'00400000
Number of transfers: 128 (reloading 32 times)	DMATCR2	H'00000080
Transfer source address: incremented	CHCR2	H'00089E35
Transfer destination address: decremented		
Transfer request source: AD1		
Bus mode: burst		
Transfer unit: longword		
Interrupt request generated at end of transfer		
Channel priority order: 0 > 2 > 3 > 1	DMAOR	H'0101

When the address reload function is on, the value set in SAR returns to the initially set value at each four transfers. In this example, when an interrupt request is generated from A/D converter, byte data is read from the register in address H'04000080 in A/D converter , and it is written to external memory address H'00400000. Since longword data has been transferred, the values in SAR and DAR are H'04000084 and H'003FFFFC, respectively. The bus right is maintained and data transfers are successively performed because this transfer is in the burst mode.

After four transfers end, fifth and sixth transfers are performed if the address reload function is off, and the value in SAR is incremented from H'0400008C, H'04000090, H'04000094,.... If the address reload function is on, the DMA transfer stops after the fourth transfer ends, the bus request signal to the CPU is cleared. At this time, the value stored in SAR is not incremented from H'0400008C to H'04000090, but returns to the initially set value H'04000080. The value in DAR continues being incremented regardless of whether the address reload function is on or off.

As a result, the values in the DMAC are as shown in table 14.12 when the fourth transfer ends, depending on whether the address reload function is on or off.

Table 14.12 Values in the DMAC after the Fourth Transfer Ends

Items	Address reload on	Address reload off
SAR	H'04000080	H'04000090
DAR	H'003FFFFC	H'003FFFFC
DMATCR	H'0000007C	H'0000007C
Bus right	Released	Held
DMAC operation	Stops	Keeps operating
Interrupt	Not generated	Not generated
Transfer request source flag clear	Executed	Not executed

Notes: 1. An interrupt is generated regardless of whether the address reload function is on or off, if transfers are executed until the value in DMATCR reaches 0 and the IE bit in CHCR has been set to 1.

2. The transfer request source flag is cleared regardless of whether the address reload function is on or off, if transfers are executed until the value in DMATCR reaches 0.

3. Specify the burst mode to use the address reload function. This function may not be correctly executed in the cycle-steal mode.

4. Set the value multiple of four in DMATCR to use the address reload function. This function may not be correctly executed if other values are specified.

14.5.4 Example of DMA Transfer between External Memory and SCIF Transmitter (Indirect Address on)

In this example, DMA transfer is performed between the external memory specified with the indirect address (transfer source) and the SCIF transmitter (transfer destination) . Table 14.13 shows the transfer conditions and register settings. In addition, the trigger of the number of transmit FIFO data is set to 1 (TTRG1 = TTRG0 = 1 in SCFCR).

Table 14.13 Transfer Conditions and Register Settings for Transfer between External Memory and SCIF Transmitter

Transfer Conditions	Register	Setting
Transfer source: external memory	SAR3	H'00400000
Value stored in address H'00400000	—	H'00450000
Value stored in address H'04500000	—	H'55
Transfer destination: On-chip SCIF TDR2	DAR3	H'04000156
Number of transfers: 10	DMATCR3	H'0000000A
Transfer source address: incremented	CHCR3	H'00011C01
Transfer destination address: fixed		
Transfer request source: SCIF (TXI2)		
Bus mode: cycle steal		
Transfer unit: byte		
No interrupt request generated at end of transfer		
Channel priority order: 0 > 1 > 2 > 3	DMAOR	H'0001

If the indirect address is on, data stored in the address set in SAR is not used as transfer source data. In the indirect address, after the value stored in the address set in SAR is read, that read value is used as an address again, and the value stored in that address is read and stored in the address set in DAR.

In the example shown in table 14.11, when an SCIF transfer request is generated, the DMAC reads the value in address H'00400000 set in SAR3. Since the value H'00450000 is stored in that address, the DMAC reads the value H'00450000. Next, the DMAC uses that read value as an address again, and reads the value H'55 stored in that address. Then, the DMAC writes the value H'55 to address H'04000156 set in DAR3; this completes one indirect address transfer.

In the indirect address, when data is read first from the address set in SAR3, the data transfer size is always longword regardless of the settings of the TS0 and the TS1 bits that specify the transfer data size. However, whether the transfer source address is fixed, incremented, or decremented is specified according to the SM0 and the SM1 bits. Therefore, in this example, though the transfer data size is specified as byte, the value in SAR3 is H'00400004 when one transfer ends. Write operation is the same as that in the normal dual address transfer.

14.6 Cautions

1. The DMA channel control registers (CHCR0–CHCR3) can be accessed in any data size. The DMA operation register (DMAOR) must be accessed in byte (eight bits) or word (16 bits); other registers must be accessed in word (16 bits) or longword (32 bits).
2. Before rewriting the RS0–RS3 bits of CHCR0–CHCR3, first clear the DE bit to 0 (when rewriting CHCR with a byte address, be sure to set the DE bit to 0 in advance).
3. Even when the NMI interrupt is input when the DMAC is not operating, the NMIF bit of the DMAOR will be set.
4. When entering the standby mode, the DME bit in DMAOR must be cleared to 0 and the transfers accepted by the DMAC must end.
5. The on-chip peripherals which DMAC can access are IRDA, SCIF, A/D converter, D/A converter, and I/O ports. Do not access the other peripherals by DMAC.
6. When starting up the DMAC, set CHCR or DMAOR last. Specifying other registers last does not guarantee normal operation.
7. Even if the maximum number of transfers is performed in the same channel after the DMATCR count reaches 0 and the DMA transfer ends normally, write 0 to DMATCR. Otherwise, normal DMA transfer may not be performed.
8. When using the address reload function, specify the burst mode as a transfer mode. In the cycle-steal mode, normal DMA transfer may not be performed.
9. When using the address reload function, set the value multiple of four in DMATCR. Specifying other values does not guarantee normal operation.
10. When detecting an external request at the falling edge, keep the external request pin high when setting the DMAC.
11. Do not access the space ranging from H'4000062 to H'400006F, which is not used in the DMAC. Accessing that space may cause malfunctions.
12. In 16-byte transfer, set the SLPFRQ bit in the FRQCR in CPG to 0.

Section 15 Timer (TMU)

15.1 Overview

This LSI uses a three-channel 32-bit timer unit (TMU).

15.1.1 Features

The TMU has the following features:

- Each channel is provided with an auto-reload 32-bit down counter
- Channel 2 is provided with an input capture function
- All channels are provided with 32-bit constant registers and 32-bit down counters that can be read or written to at any time
- All channels generate interrupt requests when the 32-bit down counter underflows (H'00000000 → H'FFFFFFFF)
- Allows selection between 6 counter input clocks: External clock (TCLK), on-chip RTC output clock (16 kHz), Pϕ/4, Pϕ/16, Pϕ/64, Pϕ/256. (Pϕ is the internal clock for peripheral modules and can be selected as 1/4, 1/2, or the same frequency as that of the CPU operating clock ϕ.) See section 9, On-Chip Oscillation Circuits, for more information on the clock pulse generator.
- All channels can operate when the SH7729 is in standby mode: When the RTC output clock is being used as the counter input clock, the SH7729 is still able to count in standby mode.
- Synchronized read: TCNT is a sequentially changing 32-bit register. Since the peripheral module used has an internal bus width of 16 bits, a time lag can occur between the time when the upper 16 bits and lower 16 bits are read. To correct the discrepancy in the counter read value caused by this time lag, a synchronization circuit is built into the TCNT so that the entire 32-bit data in the TCNT can be read at once.
- The maximum operating frequency of the 32-bit counter is 2 MHz on all channels: Operate the SH7729 so that the clock input to the timer counters of each channel (obtained by dividing the external clock and internal clock with the prescaler) does not exceed the maximum operating frequency.

15.1.2 Block Diagram

Figure 15.1 shows a block diagram of the TMU.

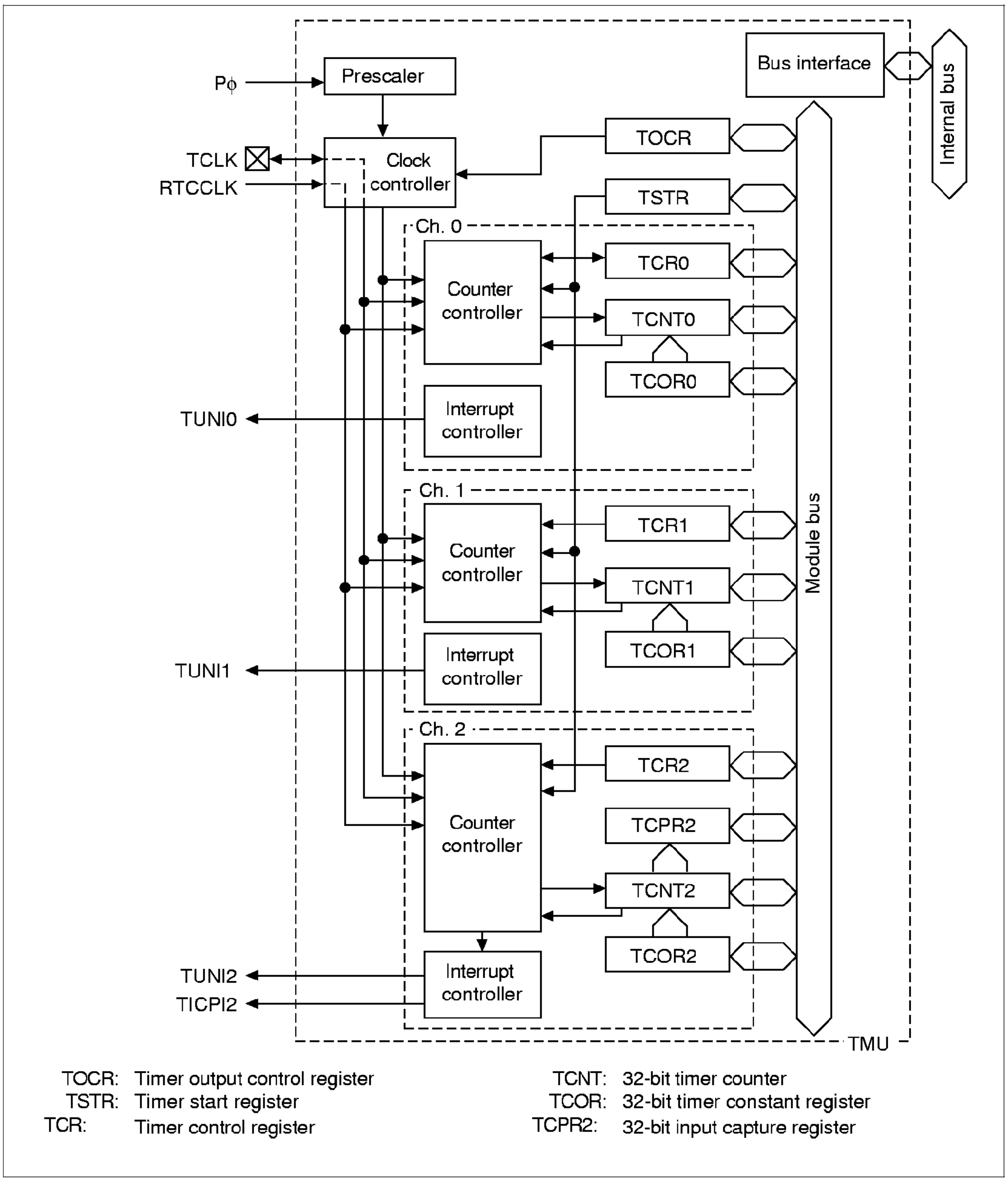


Figure 15.1 TMU Block Diagram

15.1.3 Pin Configuration

Table 15.1 shows the pin configuration of the TMU.

Table 15.1Pin Configuration

Channel	Pin	I/O	Description
Clock input/clock output	TCLK	I/O	External clock input pin/input capture control input pin/realtime clock (RTC) output pin

15.1.4 Register Configuration

Table 15.2 shows the TMU register configuration.

Table 15.2TMU Register Configuration

Channel	Register	Abbreviation	R/W	Initial Value*	Address	Access Size
Common	Timer output control register	TOCR	R/W	H'00	H'FFFFFE90	8
	Timer start register	TSTR	R/W	H'00	H'FFFFFE92	8
0	Timer constant register 0	TCOR0	R/W	H'FFFFFFFF	H'FFFFFE94	32
	Timer counter 0	TCNT0	R/W	H'FFFFFFFF	H'FFFFFE98	32
	Timer control register 0	TCR0	R/W	H'0000	H'FFFFFE9C	16
1	Timer constant register 1	TCOR1	R/W	H'FFFFFFFF	H'FFFFFEA0	32
	Timer counter 1	TCNT1	R/W	H'FFFFFFFF	H'FFFFFEA4	32
	Timer control register 1	TCR1	R/W	H'0000	H'FFFFFEA8	16
2	Timer constant register 2	TCOR2	R/W	H'FFFFFFFF	H'FFFFFEAC	32
	Timer counter 2	TCNT2	R/W	H'FFFFFFFF	H'FFFFFEB0	32
	Timer control register 2	TCR2	R/W	H'0000	H'FFFFFEB4	16
	Input capture register 2	TCPR2	R	Undefined	H'FFFFFEB8	32

Note: Initialized by power-on resets or manual resets.

15.2 TMU Registers

15.2.1 Timer Output Control Register (TOCR)

TOCR is an 8-bit read/write register that selects whether to use the external TCLK pin as an external clock or an input capture control usage input pin, or an output pin for the on-chip RTC output clock. TOCR is initialized to H'00 by a power-on reset or manual reset, but is not initialized in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	—	—	—	—	TCOE
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

Bits 7 to 1—Reserved: These bits always read 0. The write value should always be 0.

Bit 0—Timer Clock Pin Control (TCOE): Selects use of the timer clock pin (TCLK) as an external clock output pin or input pin for input capture control for the on-chip timer, or as an output pin for the on-chip RTC output clock.

Bit 0: TCOE	Description
0	Timer clock pin (TCLK) used as external clock input or input capture control input pin for the on-chip timer (Initial value)
1	Timer clock pin (TCLK) used as output pin for on-chip RTC output clock

15.2.2 Timer Start Register (TSTR)

TSTR is an 8-bit read/write register that selects whether to run or halt the timer counters (TCNT) for channels 0–2. TSTR is initialized to H'00 by a power-on reset or manual reset, but is not initialized in standby mode when the input clock selected for the channel is the on-chip RTC clock (RTCCLK). It is initialized in standby mode, changing the multiplying ratio of PLL circuit 1 or MSTP2 bit in STBCR is set to a logic one only when an external clock (TCLK) or the peripheral clock (Pϕ) is used as the input clock.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	—	—	STR2	STR1	STR0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

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

Bit 2—Counter Start 2 (STR2): Selects whether to run or halt timer counter 2 (TCNT2).

Bit 2: STR2	Description	
0	Halt TCNT2 count value)	(Initial
1	Start TCNT2 counting	

Bit 1—Counter Start 1 (STR1): Selects whether to run or halt timer counter 1 (TCNT1).

Bit 1: STR1	Description	
0	Halt TCNT1 count value)	(Initial
1	Start TCNT1 counting	

Bit 0—Counter Start 0 (STR0): Selects whether to run or halt timer counter 0 (TCNT0).

Bit 0: STR0	Description	
0	Halt TCNT0 count value)	(Initial
1	Start TCNT0 counting	

15.2.3 Timer Control Register (TCR)

The timer control registers (TCR) control the timer counters (TCNT) and interrupts. The TMU has three TCR registers for each channel.

The TCR registers are 16-bit read/write registers that 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. When the external clock has been selected, they also select its edge. Additionally, TCR2 controls the channel 2 input capture function and the issuance of interrupts during input capture. The TCRs are initialized to H'0000 by a power-on reset and manual reset. They are not initialized in standby mode.

Channel 0 and 1 TCR Bit Configuration:

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

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	UNIE	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Channel 2 TCR Bit Configuration:

Bit:	15	14	13	12	11	10	9	8
Bit name:	—	—	—	—	—	—	ICPF	UNF
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	ICPE1	ICPE0	UNIE	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 to 10, 9 (except TCR2), 7, and 6 (except TCR2)—Reserved: These bits always read 0. The write value should always be 0.

Bit 9—Input Capture Interrupt Flag (ICPF): A function of channel 2 only: the flag is set when input capture is requested via the TCLK pin.

Bit 9: ICPF	Description
0	No input capture request has been issued. Clearing condition: When 0 is written to ICPF (Initial value)
1	Input capture has been requested via the TCLK pin. Setting condition: When an input capture is requested via the TCLK pin*

Note: Contents do not change when 1 is written to ICPF.

Bit 8—Underflow Flag (UNF): Status flag that indicates occurrence of a TCNT underflow.

Bit 8: UNF	Description
0	TCNT has not underflowed. Clearing condition: When 0 is written to UNF (Initial value)
1	TCNT has underflowed (H'00000000 → H'FFFFFFFF). Setting condition: When TCNT underflows*

Note: Contents do not change when 1 is written to UNF.

Bits 7 and 6—Input Capture Control (ICPE1, ICPE0): A function of channel 2 only: determines whether the input capture function can be used, and when used, whether or not to enable interrupts.

When using this input capture function it is necessary to set the TCLK pin to input mode with the TCOE bit in the TOCR register. Additionally, use the CKEG bit to designate use of either the rising or falling edge of the TCLK pin to set the value in TCNT2 in the input capture register (TCPR2).

Bit 7: ICPE1	Bit 6: ICPE0	Description
0	0	Input capture function is not used. (Initial value)
	1	Reserved (Setting disabled)
1	0	Input capture function is used. Interrupts due to ICPF are not enabled.
	1	Input capture function is used. Interrupts due to ICPF are enabled.

Bit 5—Underflow Interrupt Control (UNIE): Controls enabling of interrupt generation when the status flag (UNF) indicating TCNT underflow has been set to 1.

Bit 5: UNIE	Description
0	Interrupts due to UNF are not enabled. (Initial value)
1	Interrupts due to UNF are enabled.

Bits 4 and 3—Clock Edge 1, 0 (CKEG1, CKEG0): These bits select the external clock edge when the external clock is selected, or when the input capture function is used.

Bit 4: CKEG1	Bit 3: CKEG0	Description
0	0	Count/capture register set on rising edge (Initial value)
	1	Count/capture register set on falling edge
1	X	Count/capture register set on both rising and falling edge

Note: X means 0, 1, or 'don't care'.

Bits 2 to 0—Timer Prescalers 2–0 (TPSC2–TPSC0): These bits select the TCNT count clock.

Bit 2: TPSC2	Bit 1: TPSC1	Bit 0: TPSC0	Description
0	0	0	Internal clock: count on Pϕ/4 (Initial value)
		1	Internal clock: count on Pϕ/16
	1	0	Internal clock: count on Pϕ/64
		1	Internal clock: count on Pϕ/256
1	0	0	Internal clock: count on clock output of on-chip RTC (RTCCLK)
		1	External clock: count on TCLK pin input
	1	0	Reserved (Setting disabled)
		1	Reserved (Setting disabled)

15.2.4 Timer Constant Register (TCOR)

The timer constant registers are 32-bit registers. The TMU has three TCOR registers, one for each of the three channels.

TCOR is a 32-bit read/write register. When a TCNT count-down results in an underflow, the TCOR value is set in TCNT and the count-down continues from that value. TCOR is initialized to H'FFFFFFFF by a power-on reset or manual reset; it is not initialized in standby mode, and retains its contents.

TCOR:

Bit:	31	30	29	28	27	26	25	24
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15.2.5 Timer Counters (TCNT)

The timer counters are 32-bit read/write registers. The TMU has three timer counters, one for each channel.

TCNT counts down upon input of a clock. The clock input is selected using the TPSC2–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.

Because the internal bus for the SH7729 on-chip supporting modules is 16 bits wide, a time lag can occur between the time when the upper 16 bits and lower 16 bits are read. Since TCNT counts sequentially, this time lag can create discrepancies between the data in the upper and lower halves. To correct the discrepancy, a buffer register is connected to TCNT so that upper and lower halves are not read separately. The entire 32-bit data in TCNT can thus be read at once.

TCNT is initialized to H'FFFFFFFF by a power-on reset or manual reset; it is not initialized in standby mode, and retains its contents.

TCNT:

Bit:	31	30	29	28	27	26	25	24
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15.2.6 Input Capture Register (TCPR2)

The input capture register (TCPR2) is a read-only 32-bit register built only into timer 2. Control of TCPR2 setting conditions due to the TCLK pin is affected by the input capture function bits (ICPE1/ICPE2 and CKEG1/CKEG0) in TCR2. When a TCPR2 setting indication due to the TCLK pin occurs, the value of TCNT2 is copied into TCPR2.

TCNT2 is not initialized by a power-on reset or manual reset, or in standby mode.

TCPR2:

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

Bit:	23	22	21	20	19	18	17	16
Bit name:								
Initial value:	—	—	—	—	—	—	—	—
R/W:	R	R	R	R	R	R	R	R

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

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	—	—	—	—	—	—	—	—
R/W:	R	R	R	R	R	R	R	R

15.3 TMU Operation

15.3.1 Overview

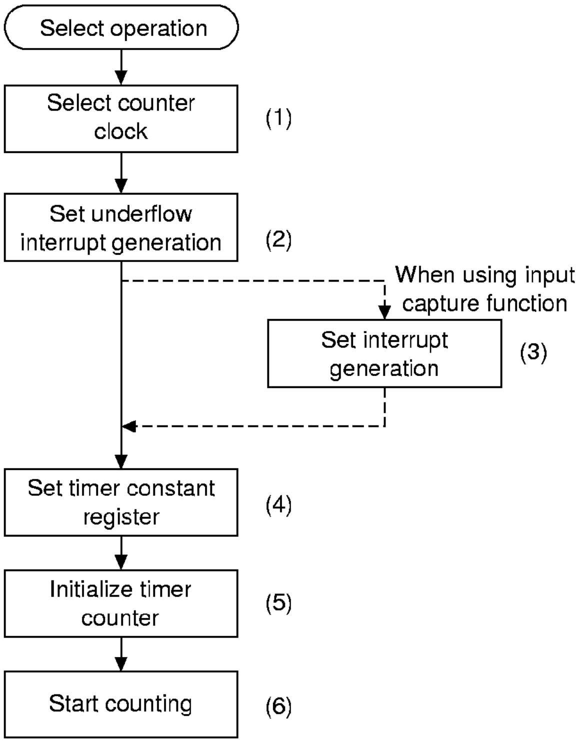
Each of three channels has a 32-bit timer counter (TCNT) and a 32-bit timer constant register (TCOR). The TCNT counts down. The auto-reload function enables synchronized counting and counting by external events. Channel 2 has an input capture function.

15.3.2 Basic Functions

Counter Operation: When the STR0–STR2 bits in the timer start register (TSTR) are set, the corresponding timer counter (TCNT) starts counting. When a TCNT underflows (H'00000000→ H'FFFFFFFF), 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.

The count operation is set as follows (figure 15.2):

1. Select the counter clock with the TPSC2–TPSC0 bits in the timer control register (TCR). If the external clock is selected, set the TCLK pin to input mode with the TOCE bit in TOCR, and select its edge with the CKEG1 and CKEG0 bits in TCR.
2. Use the UNIE bit in TCR to set whether to generate an interrupt when TCNT underflows.
3. When using the input capture function, set the ICPE bits in TCR, including the choice of whether or not to use the interrupt function (channel 2 only).
4. Set a value in the timer constant register (TCOR) (the cycle is the set value plus 1).
5. Set the initial value in the timer counter (TCNT).
6. Set the STR bit in the timer start register (TSTR) to 1 to start operation.



Note:

When an interrupt has been generated, clear the flag in the interrupt handler that caused it. If interrupts are enabled without clearing the flag, another interrupt will be generated.

Figure 15.2 Setting the Count Operation

Auto-Reload Count Operation: Figure 15.3 shows the TCNT auto-reload operation.

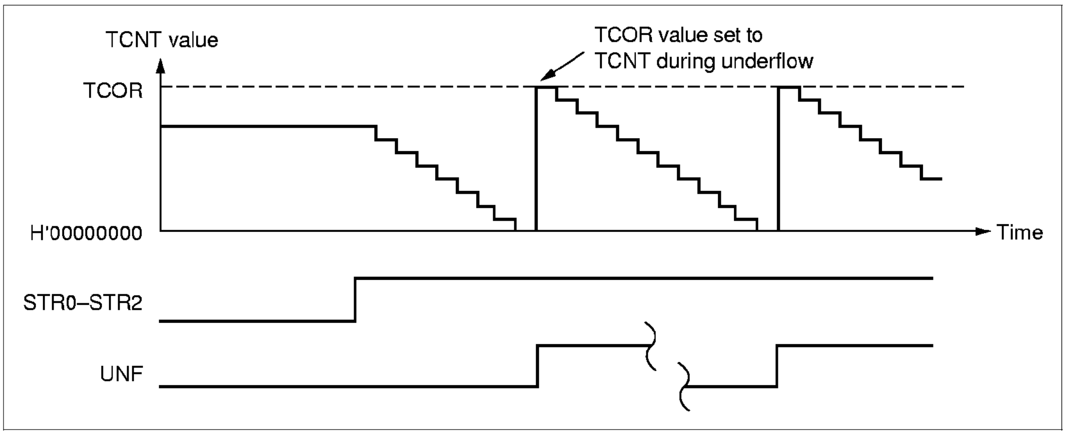


Figure 15.3 Auto-Reload Count Operation

TCNT Count Timing:

- Internal Clock Operation: Set the TPSC2–TPSC0 bits in TCR to select whether peripheral module clock Pϕ or one of the four internal clocks created by dividing it is used (Pϕ/4, Pϕ/16, Pϕ/64, Pϕ/256). Figure 15.4 shows the timing.

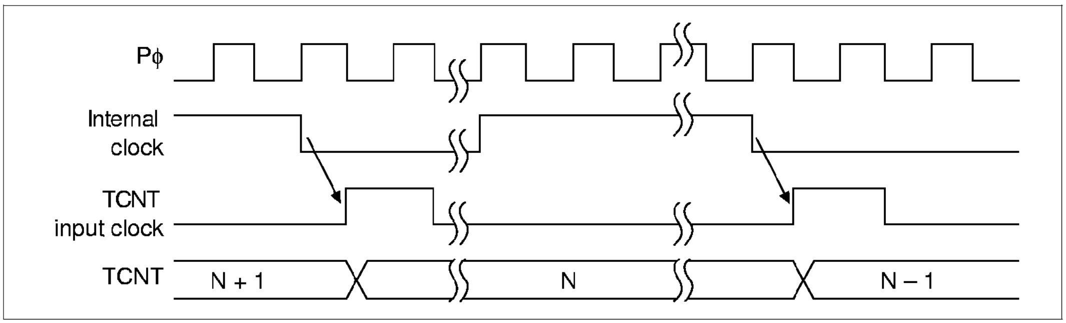


Figure 15.4 Count Timing when Internal Clock Is Operating

- External Clock Operation: Set the TPSC2–TPSC0 bits in TCR to select the external clock (TCLK) as the timer clock. Use the CKEG1 and CKEG0 bits in TCR to select the detection edge. Rise, fall or both may be selected. The pulse width of the external clock must be at least 1.5 peripheral module clock cycles for single edges or 2.5 peripheral module clock cycles for both edges. A shorter pulse width will result in accurate operation. Figure 15.5 shows the timing for both-edge detection.

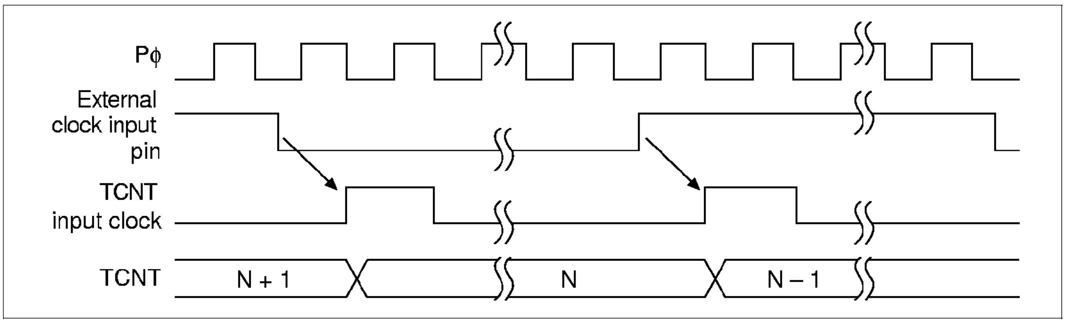


Figure 15.5 Count Timing when External Clock is Operating (Both Edges Detected)

- On-Chip RTC Clock Operation: Set the TPSC2–TPSC0 bits in TCR to select the on-chip RTC clock as the timer clock. Figure 15.6 shows the timing.

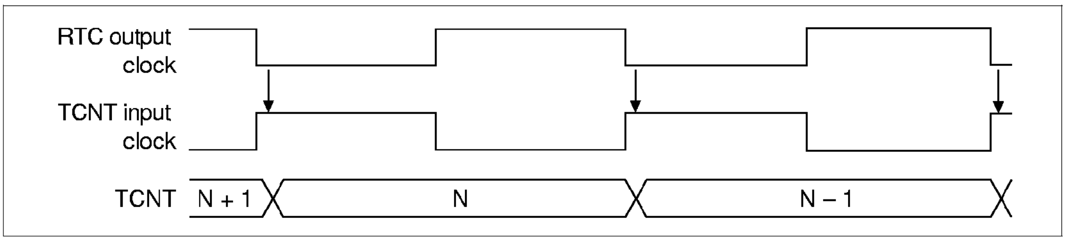


Figure 15.6 Count Timing when On-Chip RTC Clock Is Operating

Input Capture Function: Channel 2 has an input capture function (figure 15.7). When using the input capture function, set the TCLK pin to input mode with the TCOE bit in the timer output control register (TOCR) and set the timer operation clock to internal clock or on-chip RTC clock with the TPCS2–TPCS0 bits in the timer control register (TCR2). Also, designate use of the input capture function and whether to generate interrupts on using it with the IPCE1–IPCE0 bits in TCR2, and designate the use of either the rising or falling edge of the TCLK pin to set the timer counter (TCNT2) value into the input capture register (TCPR2) with the CKEG1–CKEG0 bits in TCR2.

The input capture function cannot be used in standby mode.

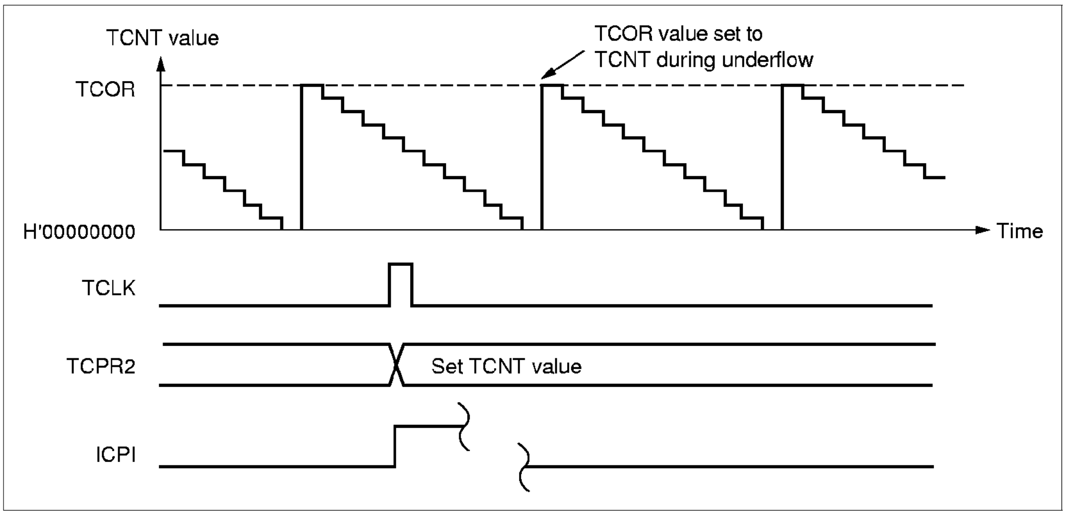


Figure 15.7 Operation Timing when Using the Input Capture Function (Using TCLK Rising Edge)

15.4 Interrupts

There are two sources of TMU interrupts: underflow interrupts (TUNI) and interrupts when using the input capture function (TICPI2).

15.4.1 Status Flag Set Timing

UNF is set to 1 when the TCNT underflows (H'00000000 → H'FFFFFFFF). Figure 15.8 shows the timing.

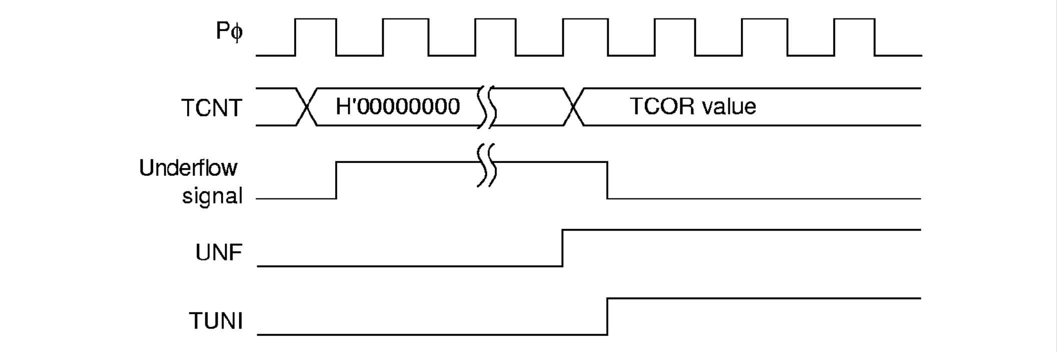


Figure 15.8 UNF Set Timing

15.4.2 Status Flag Clear Timing

The status flag can be cleared by writing 0 from the CPU. Figure 15.9 shows the timing.

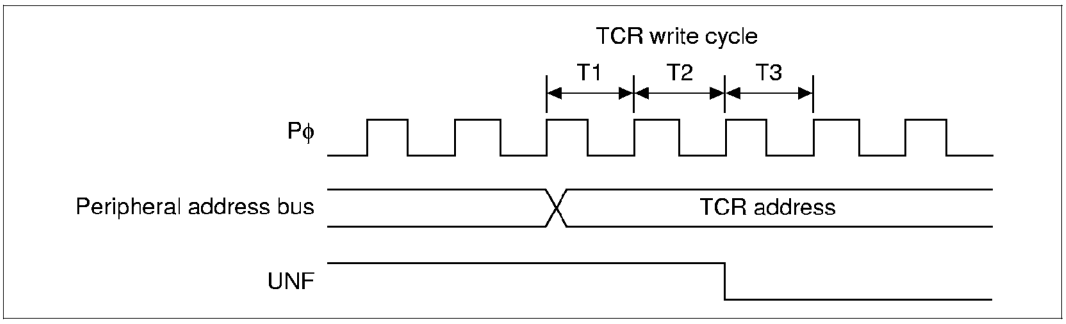


Figure 15.9 Status Flag Clear Timing

15.4.3 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 exception source register (INTEVT, 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 Processing, and section 6, Interrupt Controller). Table 15.3 lists TMU interrupt sources.

Table 15.3TMU Interrupt Sources

Channel	Interrupt Source	Description	Priority
0	TUNI0	Underflow interrupt 0	High
1	TUNI1	Underflow interrupt 1	↓
2	TUNI2	Underflow interrupt 2	
2	TICPI2	Input capture interrupt 2	Low

15.5 Usage Notes

15.5.1 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–STR0) in the timer start register (TSTR) to halt timer counting.

15.5.2 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 16 Realtime Clock (RTC)

16.1 Overview

This LSI has a realtime clock (RTC) with its own 32.768-kHz crystal oscillator.

16.1.1 Features

- Clock and calendar functions (BCD display): seconds, minutes, hours, date, day of the week, month, and year
- 1-Hz to 64-Hz timer (binary display)
- Start/stop function
- 30-second adjust function
- Alarm interrupt: frame comparison of seconds, minutes, hours, date, day of the week, and month can be used as conditions for the alarm interrupt
- Cyclic interrupts: the interrupt cycle may be 1/256 second, 1/64 second, 1/16 second, 1/4 second, 1/2 second, 1 second, or 2 seconds
- Carry interrupt: a carry interrupt indicates when a carry occurs during a counter read
- Automatic leap year correction

16.1.2 Block Diagram

The following abbreviations are used in the block diagram of the RTC (figure 16.1):

R64CNT:	64-Hz counter	RSECAR:	Second alarm register
RSECCNT:	Second counter	RMINAR:	Minute alarm register
RMINCNT:	Minute counter	RHRAR:	Hour alarm register
RHRCNT:	Hour counter	RWKAR:	Day of the week alarm register
RWKCNT:	Day of the week counter	RDAYAR:	Date alarm register
RDAYCNT:	Date counter	RMONAR:	Month alarm register
RMONCNT:	Month counter	RCR1:	RTC control register 1
RYRCNT:	Year counter	RCR2:	RTC control register 2

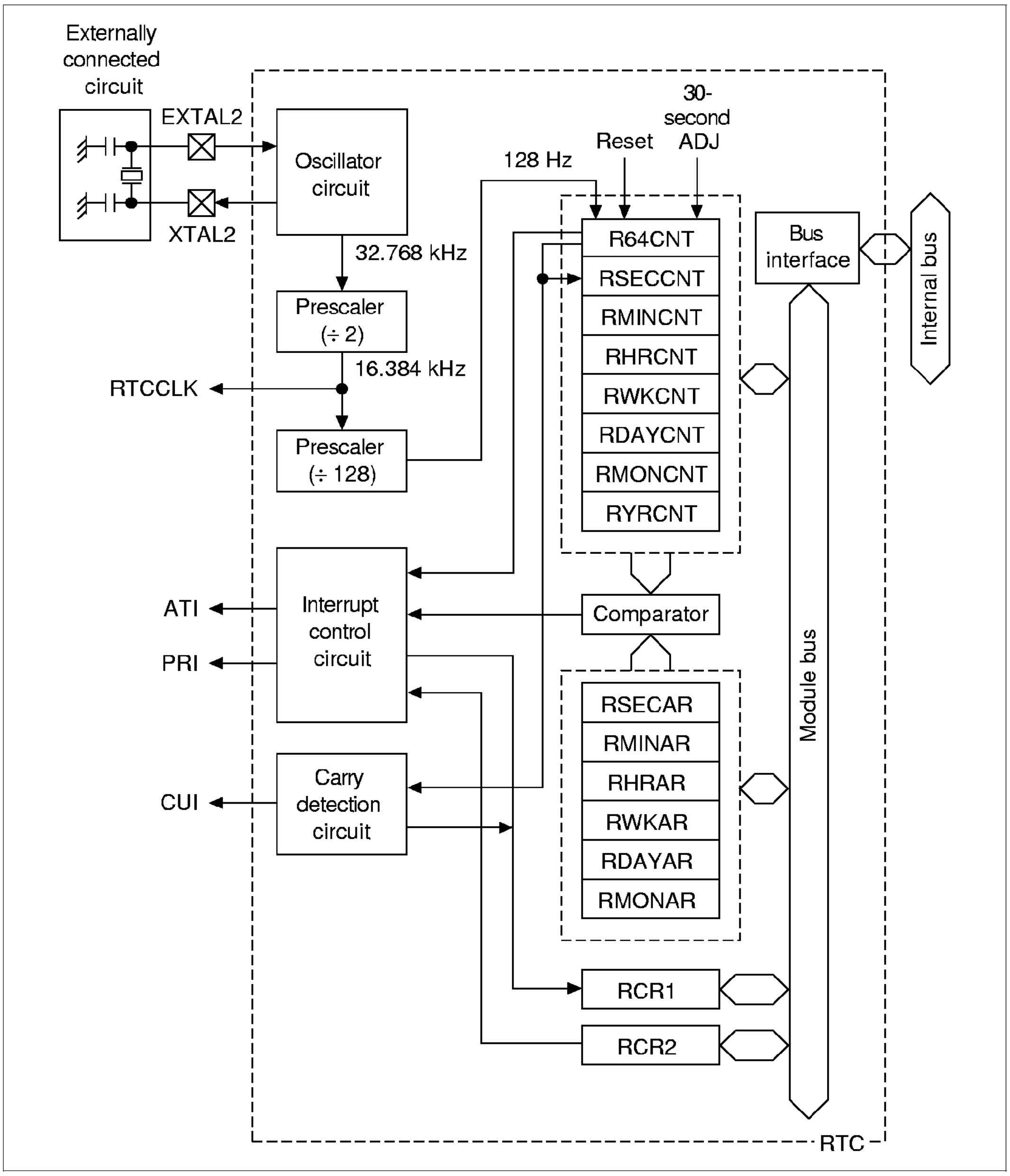


Figure 16.1 RTC Block Diagram

16.1.3 Pin Configuration

Table 16.1 shows the RTC pin configuration.

Table 16.1RTC Pin Configuration

Pin	Abbreviation	I/O	Description
RTC oscillator crystal pin	EXTAL2	I	Connects crystal to RTC oscillator
RTC oscillator crystal pin	XTAL2	O	Connects crystal to RTC oscillator
Clock input/clock output	TCLK	I/O	External clock input pin/input capture control input pin/realtime clock (RTC) output pin (shared by TMU)
Dedicated power-supply pin for RTC	VCC(RTC)	—	Dedicated power-supply pin for RTC*
Dedicated GND pin for RTC	GND(RTC)	—	Dedicated GND pin for RTC*

Note: Power must be supplied to the RTC power supply pins even when the RTC is not used. Even if only the RTC is used, power must be supplied to all power supply pins, including these pins.

In standby mode, also, power must be supplied to all power supply pins, including these pins.

16.1.4 RTC Register Configuration

Table 16.2 shows the RTC register configuration.

Table 16.2RTC Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
64-Hz counter	R64CNT	R	Undefined	H'FFFFFEC0	8
Second counter	RSECCNT	R/W	Undefined	H'FFFFFEC2	8
Minute counter	RMINCNT	R/W	Undefined	H'FFFFFEC4	8
Hour counter	RHRCNT	R/W	Undefined	H'FFFFFEC6	8
Day of week counter	RWKCNT	R/W	Undefined	H'FFFFFEC8	8
Date counter	RDAYCNT	R/W	Undefined	H'FFFFFECA	8
Month counter	RMONCNT	R/W	Undefined	H'FFFFFECC	8
Year counter	RYRCNT	R/W	Undefined	H'FFFFFECE	8
Second alarm register	RSECAR	R/W	Undefined*	H'FFFFFED0	8
Minute alarm register	RMINAR	R/W	Undefined*	H'FFFFFED2	8
Hour alarm register	RHRAR	R/W	Undefined*	H'FFFFFED4	8
Day of week alarm register	RWKAR	R/W	Undefined*	H'FFFFFED6	8
Date alarm register	RDAYAR	R/W	Undefined*	H'FFFFFED8	8
Month alarm register	RMONAR	R/W	Undefined*	H'FFFFFEDA	8
RTC control register 1	RCR1	R/W	H'00	H'FFFFFEDC	8
RTC control register 2	RCR2	R/W	H'09	H'FFFFFEDE	8

Note: Only the ENB bits of each register are initialized.

16.2 RTC Registers

16.2.1 64-Hz Counter (R64CNT)

Thc 64-Hz countcr (R64CNT) is an 8-bit rcad-only rcgistcr that indicatcs thc statc of thc RTC divider circuit between 64 Hz and 1 Hz.

R64CNT is reset to H'00 by setting the RESET bit in RTC control register 2 (RCR2) or the ADJ bit in RCR2 to 1.

R64CNT is not initialized by a power-on reset or manual reset, or in standby mode.

Bit 7 always reads 0.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	1Hz	2Hz	4Hz	8Hz	16Hz	32Hz	64Hz
Initial value:	0	—	—	—	—	—	—	—
R/W:	R	R	R	R	R	R	R	R

16.2.2 Second Counter (RSECCNT)

The second counter (RSECCNT) is an 8-bit read/write register used for setting/counting in the BCD-coded second section of the RTC. The count operation is performed by a carry for each second of the 64-Hz counter.

The range that can be set is 00–59 (decimal). Errant operation will result if any other value is set. Carry out write processing after halting the count operation with the START bit in RCR2.

RSECCNT is not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	10 seconds			1 second			
Initial value:	0	—	—	—	—	—	—	—
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

16.2.3 Minutc Countcr (RMINCNT)

The minute counter (RMINCNT) is an 8-bit read/write register used for setting/counting in the BCD-coded minute section of the RTC. The count operation is performed by a carry for each minute of the second counter.

The range that can be set is 00–59 (decimal). Errant operation will result if any other value is set. Carry out writc proccssing aftcr halting thc count opcration with thc START bit in RCR2.

RMINCNT is not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	10 minutes			1 minute			
Initial value:	0	—	—	—	—	—	—	—
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

16.2.4 Hour Counter (RHRCNT)

The hour counter (RHRCNT) is an 8-bit read/write register used for setting/counting in the BCD-coded hour section of the RTC. The count operation is performed by a carry for each 1 hour of the minute counter.

The range that can be set is 00–23 (decimal). Errant operation will result if any other value is set. Carry out write processing after halting the count operation with the START bit in RCR2.

RHRCNT is not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	10 hours		1 hour			
Initial value:	0	0	—	—	—	—	—	—
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

16.2.5 Day of the Week Counter (RWKCNT)

The day of the week counter (RWKCNT) is an 8-bit read/write register used for setting/counting in the BCD-coded day of week section of the RTC. The count operation is performed by a carry for each day of the date counter.

The range that can be set is 0–6 (decimal). Errant operation will result if any other value is set. Carry out write processing after halting the count operation with the START bit in RCR2.

RWKCNT is not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	—	—	Day of week		
Initial value:	0	0	0	0	0	—	—	—
R/W:	R	R	R	R	R	R/W	R/W	R/W

Days of the week are coded as shown in table 16.3.

Table 16.3Day-of-Week Codes (RWKCNT)

Day of Week	Code
Sunday	0
Monday	1
Tuesday	2
Wednesday	3
Thursday	4
Friday	5
Saturday	6

16.2.6 Date Counter (RDAYCNT)

The date counter (RDAYCNT) is an 8-bit read/write register used for setting/counting in the BCD-coded date section of the RTC. The count operation is performed by a carry for each day of the hour counter.

The range that can be set is 01–31 (decimal). Errant operation will result if any other value is set. Carry out write processing after halting the count operation with the START bit in RCR2.

RDAYCNT is not initialized by a power-on reset or manual reset, or in standby mode.

The RDAYCNT range that can be set changes with each month and in leap years. Please confirm the correct setting.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	10 days		1 day			
Initial value:	0	0	—	—	—	—	—	—
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

16.2.7 Month Counter (RMONCNT)

The month counter (RMONCNT) is an 8-bit read/write register used for setting/counting in the BCD-coded month section of the RTC. The count operation is performed by a carry for each month of the date counter.

The range that can be set is 00–12 (decimal). Errant operation will result if any other value is set. Carry out write processing after halting the count operation with the START bit in RCR2.

RMONCNT is not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	10 months	1 month			
Initial value:	0	0	0	—	—	—	—	—
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W

16.2.8 Year Counter (RYRCNT)

The year counter (RYRCNT) is an 8-bit read/write register used for setting/counting in the BCD-coded year section of the RTC. The least significant 2 digits of the western calendar year are displayed. The count operation is performed by a carry for each year of the month counter.

The range that can be set is 00–99 (decimal). Errant operation will result if any other value is set. Carry out write processing after halting the count operation with the START bit in RCR2 or using a carry flag as shown in figure 16.2.

RYRCNT is not initialized by a power-on reset or manual reset, or in standby mode.

Leap years are recognized by dividing the year counter value by 4 and obtaining a fractional result of 0.

Bit:	7	6	5	4	3	2	1	0
Bit name:	10 years				1 year			
Initial value:	—	—	—	—	—	—	—	—
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

16.2.9 Second Alarm Register (RSECAR)

The second alarm register (RSECAR) is an 8-bit read/write register, and an alarm register corresponding to the BCD-coded second section counter RSECCNT of the RTC. When the ENB bit is set to 1, a comparison with the RSECCNT value is performed. From among the RSECAR/RMINAR/RHRAR/RWKAR/RDAYAR/RMONAR registers, the counter and alarm register comparison is performed only on those with ENB bits set to 1, and if each of those coincide, an RTC alarm interrupt is generated.

The range that can be set is 00–59 (decimal) + ENB bit. Errant operation will result if any other value is set.

The ENB bit in RSECAR is initialized to 0 by a power-on reset. The remaining RSECAR fields are not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	ENB	10 seconds			1 second			
Initial value:	0	—	—	—	—	—	—	—
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

16.2.10Minute Alarm Register (RMINAR)

The minute alarm register (RMINAR) is an 8-bit read/write register, and an alarm register corresponding to the BCD-coded minute section counter RMINCNT of the RTC. When the ENB bit is set to 1, a comparison with the RMINCNT value is performed. From among the RSECAR/RMINAR/RHRAR/RWKAR/RDAYAR/RMONAR registers, the counter and alarm register comparison is performed only on those with ENB bits set to 1, and if each of those coincide, an RTC alarm interrupt is generated.

The range that can be set is 00–59 (decimal) + ENB bit. Errant operation will result if any other value is set.

The ENB bit in RMINAR is initialized by a power-on reset. The remaining RMINAR fields are not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	ENB	10 minutes			1 minute			
Initial value:	0	—	—	—	—	—	—	—
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

16.2.11Hour Alarm Register (RHRAR)

The hour alarm register (RHRAR) is an 8-bit read/write register, and an alarm register corresponding to the BCD-coded hour section counter RHRCNT of the RTC. When the ENB bit is set to 1, a comparison with the RHRCNT value is performed. From among the RSECAR/RMINAR/RHRAR/RWKAR/RDAYAR/RMONAR registers, the counter and alarm register comparison is performed only on those with ENB bits set to 1, and if each of those coincide, an RTC alarm interrupt is generated.

The range that can be set is 00–23 (decimal) + ENB bit. Errant operation will result if any other value is set.

The ENB bit in RHRAR is initialized by a power-on reset. The remaining RHRAR fields are not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	ENB	—	10 hours		1 hour			
Initial value:	0	0	—	—	—	—	—	—
R/W:	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

16.2.12Day of the Week Alarm Register (RWKAR)

The day of the week alarm register (RWKAR) is an 8-bit read/write register, and an alarm register corresponding to the BCD-coded day of week section counter RWKCNT of the RTC. When the ENB bit is set to 1, a comparison with the RWKCNT value is performed. From among the RSECAR/RMINAR/RHRAR/RWKAR/RDAYAR/RMONAR registers, the counter and alarm register comparison is performed only on those with ENB bits set to 1, and if each of those coincide, an RTC alarm interrupt is generated.

The range that can be set is 0–6 (decimal) + ENB bit. Errant operation will result if any other value is set.

The ENB bit in RWKAR is initialized by a power-on reset. The remaining RWKAR fields are not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	ENB	—	—	—	—	Day of week		
Initial value:	0	0	0	0	0	—	—	—
R/W:	R/W	R	R	R	R	R/W	R/W	R/W

Days of the week are coded as shown in table 16.4.

Table 16.4Day-of-Week Codes (RWKAR)

Day of Week	Code
Sunday	0
Monday	1
Tuesday	2
Wednesday	3
Thursday	4
Friday	5
Saturday	6

16.2.13Datc Alarm Rcgistcr (RDAYAR)

The date alarm register (RDAYAR) is an 8-bit read/write register, and an alarm register corresponding to the BCD-coded date section counter RDAYCNT of the RTC. When the ENB bit is set to 1, a comparison with the RDAYCNT value is performed. From among the registers RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, RMONAR, the counter and alarm register comparison is performed only on those with ENB bits set to 1, and if each of those coincide, an RTC alarm interrupt is generated.

The range that can be set is 01–31 (decimal) + ENB bit. Errant operation will result if any other value is set. The RDAYCNT range that can be set changes with some months and in leap years. Please confirm the correct setting.

The ENB bit in RDAYAR is initialized by a power-on reset. The remaining RDAYAR fields are not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	ENB	—	10 days		1 day			
Initial value:	0	0	—	—	—	—	—	—
R/W:	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

16.2.14Month Alarm Register (RMONAR)

The month alarm register (RMONAR) is an 8-bit read/write register, and an alarm register corresponding to the BCD-coded month section counter RMONCNT of the RTC. When the ENB bit is set to 1, a comparison with the RMONCNT value is performed. From among the registers RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, RMONAR, the counter and alarm register comparison is performed only on those with ENB bits set to 1, and if each of those coincide, an RTC alarm interrupt is generated.

The range that can be set is 01–12 (decimal) + ENB bit. Errant operation will result if any other value is set.

The ENB bit in RMONAR is initialized by a power-on reset. The remaining RMONAR fields are not initialized by a power-on reset or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	ENB	—	—	10 months	1 month			
Initial value:	0	0	0	—	—	—	—	—
R/W:	R/W	R	R	R/W	R/W	R/W	R/W	R/W

16.2.15 RTC Control Register 1 (RCR1)

The RTC control register 1 (RCR1) is an 8-bit read/write register that affects carry flags and alarm flags. It also selects whether to generate interrupts for each flag. Because flags are sometimes set after an operand read, do not use this register in read-modify-write processing.

RCR1 is initialized to H'00 by a power-on reset. In a manual reset, all bits are initialized to 0 except for the CF flag, which is undefined. When using the CF flag, it must be initialized beforehand. This register is not initialized in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	CF	—	—	CIE	AIE	—	—	AF
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R	R/W	R/W	R	R	R/W

Bit 7—Carry Flag (CF): Status flag that indicates that a carry has occurred. CF is set to 1 when a count-up to R64CNT or RSECCNT occurs. A count register value read at this time cannot be guaranteed; another read is required.

Bit 7: CF	Description
0	No count up of R64CNT or RSECCNT. Clearing condition: When 0 is written to CF (Initial value)
1	Count up of R64CNT or RSECCNT. Setting condition: When 1 is written to CF

Bits 6, 5, 2, and 1—Reserved: These bits always read 0. The write value should always be 0.

Bit 4—Carry Interrupt Enable Flag (CIE): When the carry flag (CF) is set to 1, the CIE bit enables interrupts.

Bit 4: CIE	Description
0	A carry interrupt is not generated when the CF flag is set to 1 (Initial value)
1	A carry interrupt is generated when the CF flag is set to 1

Bit 3—Alarm Interrupt Enable Flag (AIE): When the alarm flag (AF) is set to 1, the AIE bit allows interrupts.

Bit 3: AIE	Description
0	An alarm interrupt is not generated when the AF flag is set to 1 (Initial value)
1	An alarm interrupt is generated when the AF flag is set to 1

Bit 0—Alarm Flag (AF): The AF flag is set to 1 when the alarm time set in an alarm register (only registers with ENB bit set to 1) matches the clock and calendar time. This flag is cleared to 0 when 0 is written, but holds the previous value when 1 is to be written.

Bit 0: AF	Description
0	Clock/calendar and alarm register have not matched since last reset to 0. Clearing condition: When 0 is written to AF (Initial value)
1	Setting condition: Clock/calendar and alarm register have matched (only registers with ENB set)*

Note: Contents do not change when 1 is written to AF.

16.2.16RTC Control Register 2 (RCR2)

The RTC control register 2 (RCR2) is an 8-bit read/write register for periodic interrupt control, 30-second adjustment ADJ, divider circuit RESET, and RTC count start/stop control. It is initialized to H'09 by a power-on reset. It is initialized except for RTCEN and START by a manual reset. It is not initialized in standby mode, and retains its contents.

Bit:	7	6	5	4	3	2	1	0
Bit name:	PEF	PES2	PES1	PES0	RTCEN	ADJ	RESET	START
Initial value:	0	0	0	0	1	0	0	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7—Periodic Interrupt Flag (PEF): Indicates interrupt generation with the period designated by the PES bits. When set to 1, PEF generates periodic interrupts.

Bit 7: PEF	Description
0	Interrupts not generated with the period designated by the PES bits. Clearing condition: When 0 is written to PEF (Initial value)
1	Interrupts generated with the period designated by the PES bits. Setting condition: When 1 is written to PEF

Bits 6–4—Periodic Interrupt Flags (PES2-PES0): These bits specify the periodic interrupt.

Bit 6: PES2	Bit 5: PES1	Bit 4: PES0	Description
0	0	0	No periodic interrupts generated (Initial value)
		1	Periodic interrupt generated every 1/256 second
	1	0	Periodic interrupt generated every 1/64 second
		1	Periodic interrupt generated every 1/16 second
1	0	0	Periodic interrupt generated every 1/4 second
		1	Periodic interrupt generated every 1/2 second
	1	0	Periodic interrupt generated every 1 second
		1	Periodic interrupt generated every 2 seconds

Bit 3—RTCEN: Controls the operation of the crystal oscillator for the RTC.

Bit 3: RTCEN	Description
0	Halts the crystal oscillator for the RTC.
1	Runs the crystal oscillator for the RTC. (Initial value)

Bit 2–30—Second Adjustment (ADJ): When 1 is written to the ADJ bit, times of 29 seconds or less will be rounded to 00 seconds and 30 seconds or more to 1 minute. The divider circuit will be simultaneously reset. This bit always reads 0.

Bit 2: ADJ	Description
0	Runs normally. (Initial value)
1 (write)	30-second adjustment.

Bit 1—Reset (RESET): When 1 is written, initializes the divider circuit (RTC prescaler and R64CNT). This bit always reads 0.

Bit 1: RESET	Description
0	Runs normally. (Initial value)
1 (Write)	Divider circuit is reset.

Bit 0—Start Bit (START): Halts and restarts the counter (clock).

Bit 0: START	Description
0	Second/minute/hour/day/week/month/year counter halts.
1	Second/minute/hour/day/week/month/year counter runs normally. (Initial value)

Note: The 64-Hz counter always runs unless stopped with the RTCEN bit.

16.3 RTC Operation

16.3.1 Initial Settings of Registers after Power-On

All the registers should be set after the power is turned on.

16.3.2 Setting the Time

Part (a) in figure 16.2 shows how to set the time when the clock is stopped. This works when the entire calendar or clock is to be set.

Part (b) in figure 16.2 describes how to set the clock when the clock is running. This works when only part of the calendar or clock needs to be reset (e.g., changing only the seconds or only the hour). The write state is checked using the carry flags. When there is a carry during the writing of new data, the new data is automatically updated. Since this causes errors in the data, the data must be rewritten if the carry flag is set to 1.

The interrupt function can be used to determine the state of the carry flag.

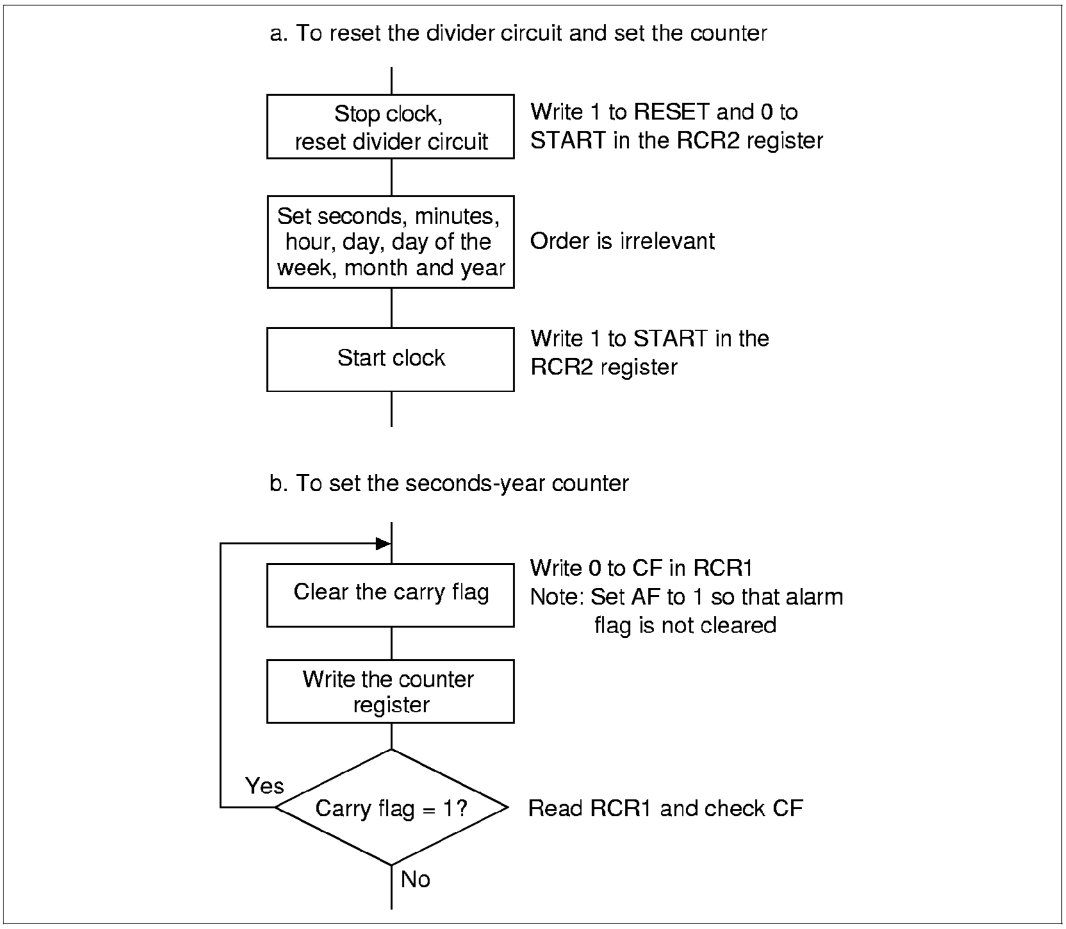

a. To reset the divider circuit and set the counter
Stop clock, reset divider circuit
Write 1 to RESET and 0 to START in the RCR2 register
Set seconds, minutes, hour, day, day of the week, month and year
Order is irrelevant
Start clock
Write 1 to START in the RCR2 register
b. To set the seconds-year counter
Clear the carry flag
Write 0 to CF in RCR1
Note: Set AF to 1 so that alarm flag is not cleared
Write the counter register
Carry flag = 1?
Yes
No
Read RCR1 and check CF

Figure 16.2 Setting the Time

16.3.3 Reading the Time

Figure 16.3 shows how to read the time. If a carry occurs while reading the time, the correct time will not be obtained, so it must be read again. Part (a) in figure 16.3 shows the method of reading the time without using interrupts; part (b) in figure 16.3 shows the method using carry interrupts. To keep programming simple, method (a) should normally be used.

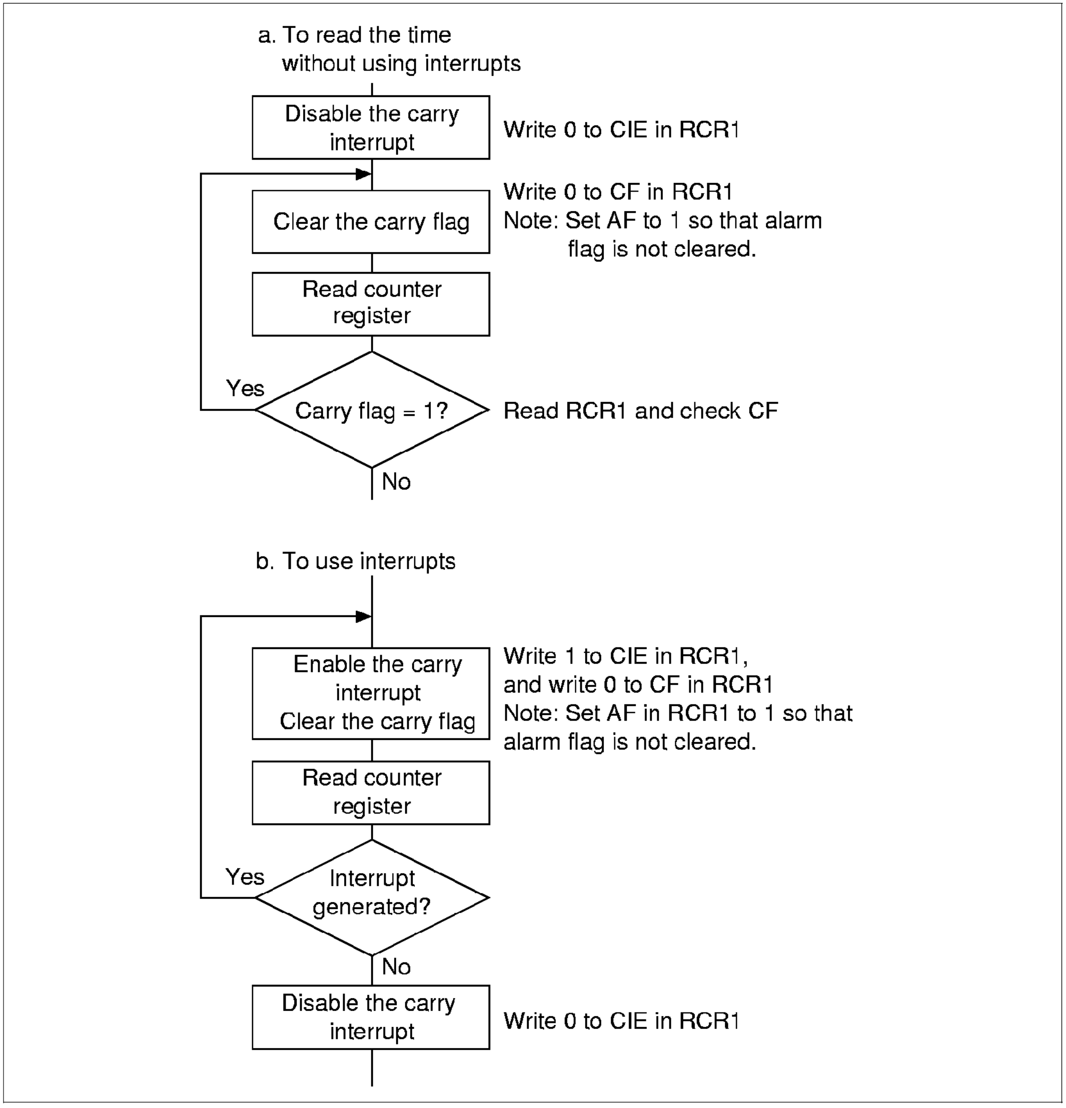


Figure 16.3 Reading the Time

16.3.4 Alarm Function

Figure 16.4 shows how to use the alarm function.

Alarms can be generated using seconds, minutes, hours, day of the week, date, month, or any combination of these. Set the ENB bit (bit 7) in the register on which the alarm is placed to 1, and then set the alarm time in the lower bits. Clear the ENB bit in the register on which the alarm is placed to 0.

When the clock and alarm times match, 1 is set in the AF bit (bit 0) in RCR1. Alarm detection can be checked by reading this bit, but normally it is done by interrupt. If 1 is placed in the AIE bit (bit 3) in RCR1, an interrupt is generated when an alarm occurs.

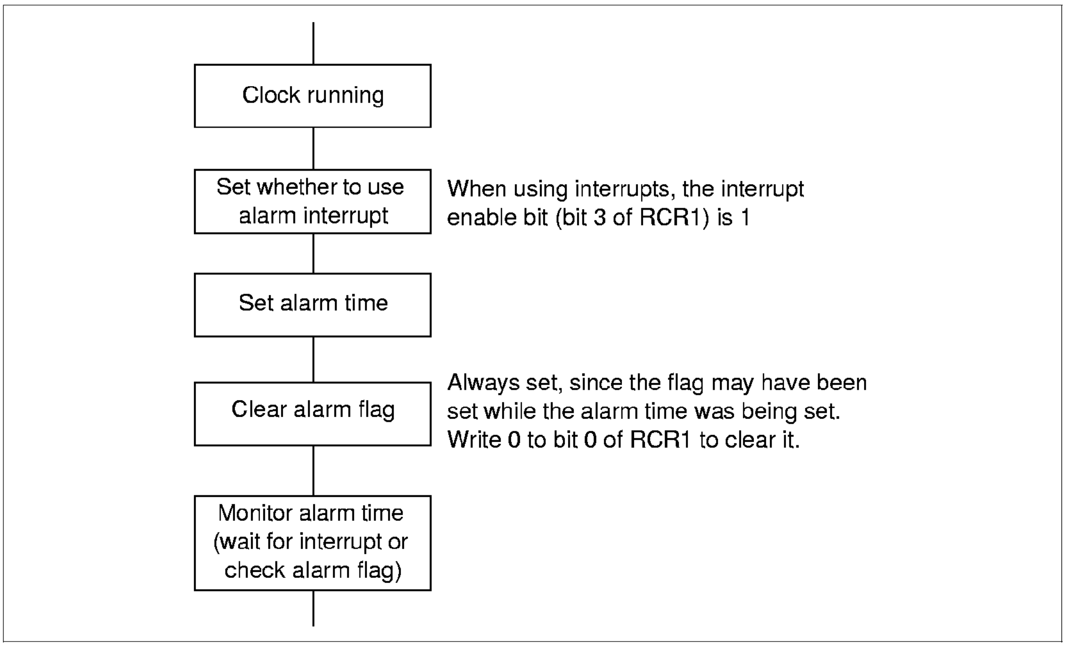

Clock running
Set whether to use alarm interrupt
When using interrupts, the interrupt enable bit (bit 3 of RCR1) is 1
Set alarm time
Clear alarm flag
Always set, since the flag may have been set while the alarm time was being set. Write 0 to bit 0 of RCR1 to clear it.
Monitor alarm time (wait for interrupt or check alarm flag)

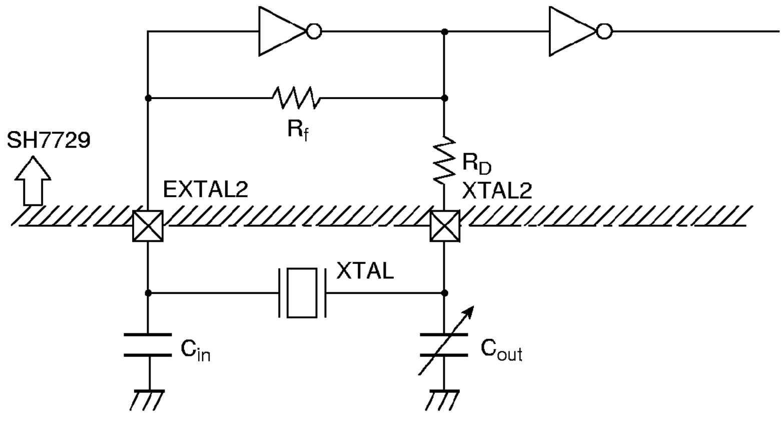
Figure 16.4 Using the Alarm Function

16.3.5 Crystal Oscillator Circuit

Crystal oscillator circuit constants (recommended values) are shown in table 16.5, and the RTC crystal oscillator circuit in figure 16.5.

Table 16.5Recommended Oscillator Circuit Constants (Recommended Values)

fosc	Cin	Cout
32.768 kHz	10 to 22 pF	10 to 22 pF



Notes: 1. Select either the C_{in} or C_{out} side for frequency adjustment variable capacitor according to requirements such as frequency range, degree of stability, etc.

2. Built-in resistance value R_f (Typ value) = 10 MΩ, R_D (Typ value) = 400 kΩ

3. C_{in} and C_{out} values include floating capacitance due to the wiring. Take care when using a ground plane.

4. The crystal oscillation settling time depends on the mounted circuit constants, floating capacitance, etc., and should be decided after consultation with the crystal resonator manufacturer.

5. Place the crystal resonator and load capacitors C_{in} and C_{out} as close as possible to the chip.

(Correct oscillation may not be possible if there is externally induced noise in the EXTAL2 and XTAL2 pins.)

6. Ensure that the crystal resonator connection pin (EXTAL2, XTAL2) wiring is routed as far away as possible from other power lines (except GND) and signal lines.

Figure 16.5 Example of Crystal Oscillator Circuit Connection

Section 17 Serial Communication Interface (SCI)

17.1 Overview

This LSI has an on-chip serial communication interface (SCI) that supports both asynchronous and clock synchronous serial communication. It also has a multiprocessor communication function for serial communication among two or more processors. The SCI supports a smart card interface, which is a serial communications feature for IC card interfaces that conforms to the ISO/IEC standard 7816-3 for identification cards. See section 18, Smart Card Interface, for more information.

17.1.1 Features

Select asynchronous or clock synchronous as the serial communications mode.

- Asynchronous mode:
  - Serial data communications are synched by start-stop in character units. The SCI can communicate with a universal asynchronous receiver/transmitter (UART), an asynchronous communication interface adapter (ACIA), or any other communications chip that employs a standard asynchronous serial system. It can also communicate with two or more other processors using the multiprocessor communication function. There are 12 selectable serial data communication formats.
  - Data length: Seven or eight bits
  - Stop bit length: One or two bits
  - Parity: Even, odd, or none
  - Multiprocessor bit: 1 or 0
  - Receive error detection: Parity, overrun, and framing errors
  - Break detection: By reading the RxD level directly from the port SC data register (SCSPTR) when a framing error occurs
- Clock synchronous mode:
  - Serial data communication is synchronized with a clock signal. The SCI can communicate with other chips having a clock synchronous communication function. There is one serial data communication format.
  - Data length: Eight bits
  - Receive error detection: Overrun errors
- Full duplex communication: The transmitting and receiving sections are independent, so the SCI can transmit and receive simultaneously. Both sections use double buffering, so continuous data transfer is possible in both the transmit and receive directions.
- On-chip baud rate generator with selectable bit rates

- Internal or external transmit/receive clock source: From either baud rate generator (internal) or SCK pin (external)
- Four types of interrupts: Transmit-data-empty, transmit-end, receive-data-full, and receive-error interrupts are requested independently.
- When the SCI is not in use, it can be stopped by halting the clock supplied to it, saving power.

17.1.2 Block Diagram

Figure 17.1 shows a SCI block diagram.

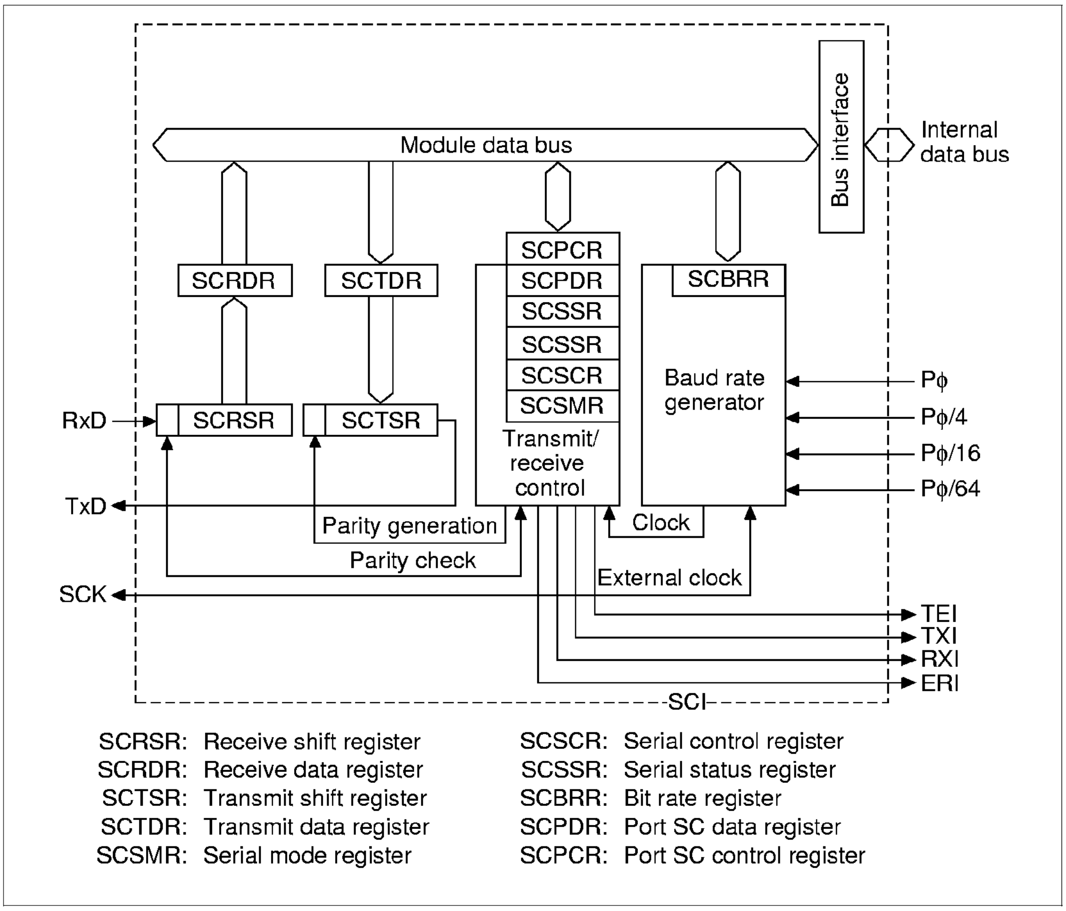
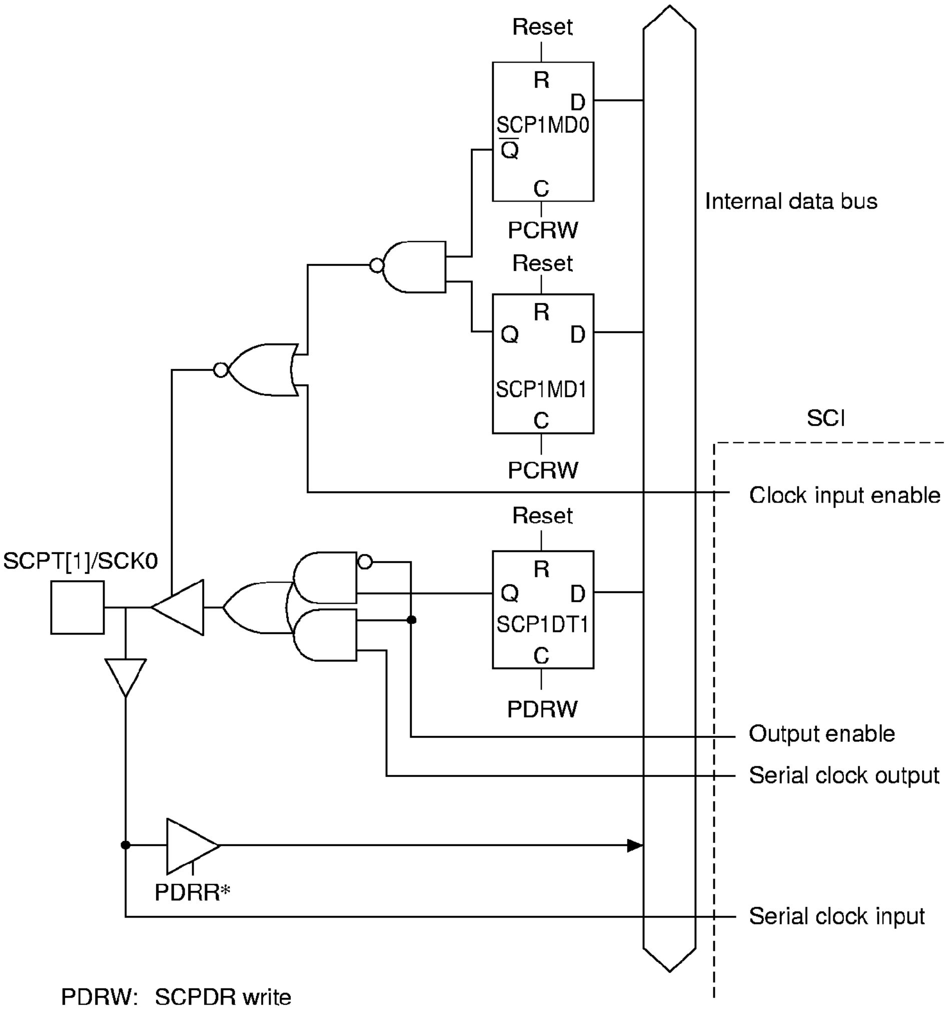

Module data bus
Bus interface
Internal data bus
SCRDR
SCTDR
SCPCR
SCPDR
SCSSR
SCSSR
SCSCR
SCSMR
SCBRR
Baud rate generator
Transmit/ receive control
RxD
TxD
SCK
SCRSR
SCTSR
Parity generation
Parity check
Clock
External clock
Pϕ
Pϕ/4
Pϕ/16
Pϕ/64
TEI
TXI
RXI
ERI
SCI
SCRSR: Receive shift register
SCRDR: Receive data register
SCTSR: Transmit shift register
SCTDR: Transmit data register
SCSMR: Serial mode register
SCSCR: Serial control register
SCSSR: Serial status register
SCBRR: Bit rate register
SCPDR: Port SC data register
SCPCR: Port SC control register

Figure 17.1 SCI Block Diagram

Figures 17.2, 17.3, and 17.4 show the block diagrams of the SCI I/O port.



PDRW: SCPDR write
PDRR: SCPDR read
PCRW: SCPCR write

Note: * When reading the SCK0 pin, clear the C/$\overline{A}$ bit in SCSMR and the CKE1 and CKE0 bits in SCSCR to 0, and set the SCP1MD1 bit in SCPCR to 1 (see section 17.2.8).

Figure 17.2 SCPT[1]/SCK0 Pin

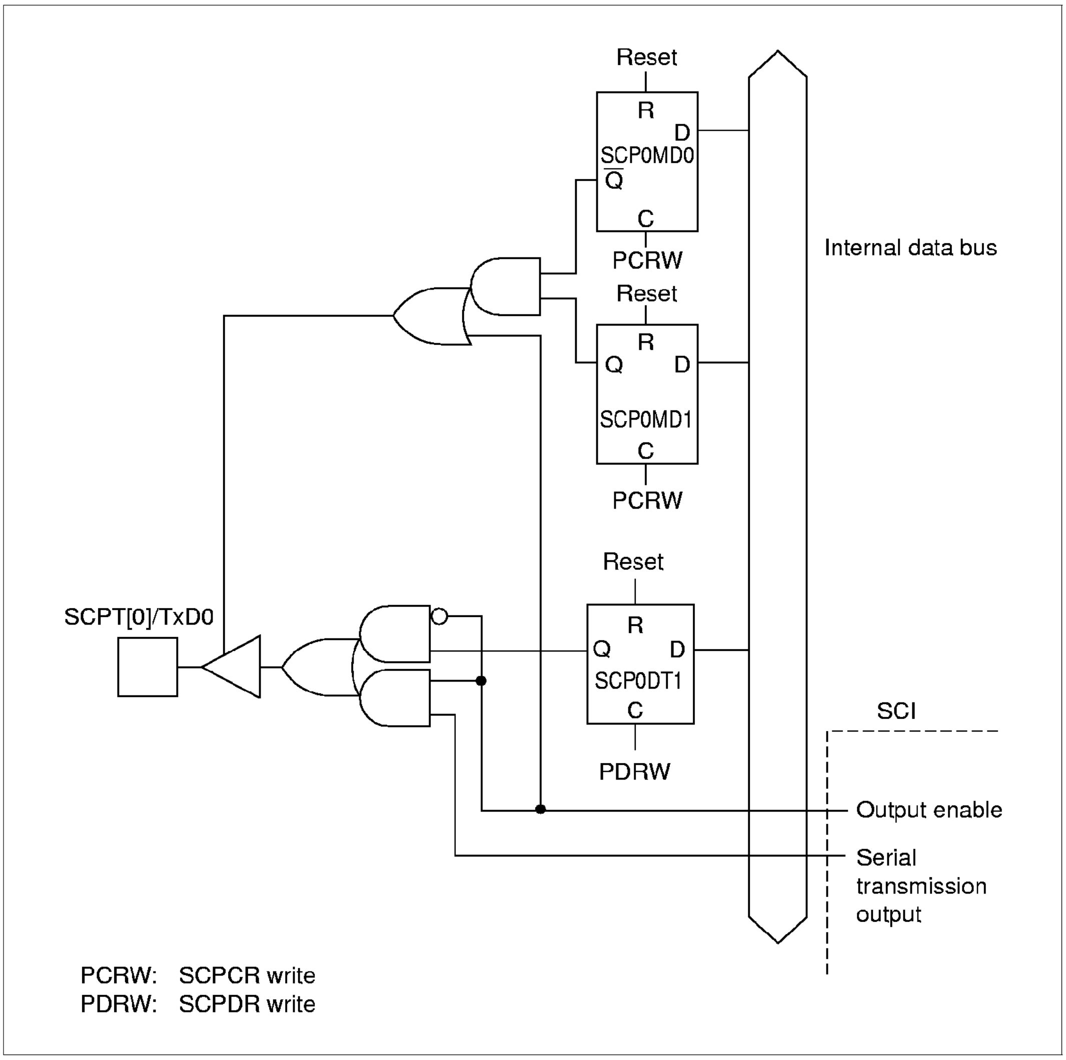
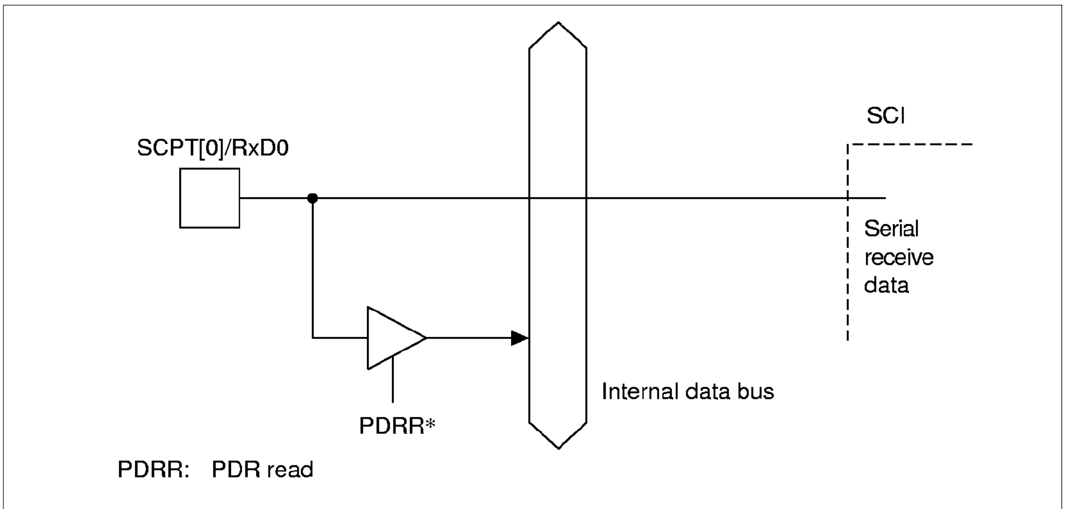


Figure 17.3 SCPT[0]/TxD0 Pin



Note: * When reading the RxD0 pin, set the RE bit in SCSCR to 1.

Figure 17.4 SCPT[0]/RxD0 Pin

17.1.3 Pin Configuration

The SCI has the serial pins summarized in table 17.1.

Table 17.1SCI Pins

Pin Name	Abbreviation	I/O	Function
Serial clock pin	SCK0	I/O	Clock I/O
Receive data pin	RxD0	Input	Receive data input
Transmit data pin	TxD0	Output	Transmit data output

Note: They are made to function as serial pins by performing SCI operation settings with the TE, RE, CKEI, and CKEO bits in SCSCR and the C/$\overline{\text{A}}$ bit in SCSMR. Break state transmission and detection can be performed by means of the SCI's SCSPTR register.

17.1.4 Register Configuration

Table 17.2 summarizes the SCI internal registers. These registers select the communication mode (asynchronous or clock synchronous), specify the data format and bit rate, and control the transmitter and receiver sections.

Table 17.2Registers

Address	Name	Abbreviation	R/W	Initial Value	Access size
H'FFFFFE80	Serial mode register	SCSMR	R/W	H'00	8
H'FFFFFE82	Bit rate register	SCBRR	R/W	H'FF	8
H'FFFFFE84	Serial control register	SCSCR	R/W	H'00	8
H'FFFFFE86	Transmit data register	SCTDR	R/W	H'FF	8
H'FFFFFE88	Serial status register	SCSSR	R/(W)*	H'84	8
H'FFFFFE8A	Receive data register	SCRDR	R	H'00	8
H'4000136	Port SC data register	SCPDR	R/W	H'00	8
H'4000116	Port SC control register	SCPCR	R/W	H'A888	16

Note: The only value that can be written is 0 to clear the flags.

17.2 Register Descriptions

17.2.1 Receive Shift Register

The receive shift register (SCRSR) receives serial data. Data input at the RxD pin is loaded into the SCRSR in the order received, LSB (bit 0) first, converting the data to parallel form. When one byte has been received, it is automatically transferred to the SCRDR. The CPU cannot read or write the SCRSR directly.

Bit:	7	6	5	4	3	2	1	0
Bit name:								
R/W:	—	—	—	—	—	—	—	—

17.2.2 Receive Data Register

The receive data register (SCRDR) stores serial receive data. The SCI completes the reception of one byte of serial data by moving the received data from the receive shift register (SCRSR) into the SCRDR for storage. The SCRSR is then ready to receive the next data. This double buffering allows the SCI to receive data continuously.

The CPU can read but not write the SCRDR. The SCRDR is initialized to H'00 by a reset or in standby or module standby modes.

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

17.2.3 Transmit Shift Register

The transmit shift register (SCTSR) transmits serial data. The SCI loads transmit data from the transmit data register (SCTDR) into the SCTSR, then transmits the data serially from the TxD pin, LSB (bit 0) first. After transmitting one-byte data, the SCI automatically loads the next transmit data from the SCTDR into the SCTSR and starts transmitting again. If the TDRE bit of the SCSSR is 1, however, the SCI does not load the SCTDR contents into the SCTSR. The CPU cannot read or write the SCTSR directly.

Bit:	7	6	5	4	3	2	1	0
Bit name:								
R/W:	—	—	—	—	—	—	—	—

17.2.4 Transmit Data Register

The transmit data register (SCTDR) is an eight-bit register that stores data for serial transmission. When the SCI detects that the transmit shift register (SCTSR) is empty, it moves transmit data written in the SCTDR into the SCTSR and starts serial transmission. Continuous serial transmission is possible by writing the next transmit data in the SCTDR during serial transmission from the SCTSR.

The CPU can always read and write the SCTDR. The SCTDR is initialized to H'FF by a reset or in standby and module standby modes.

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

17.2.5 Serial Mode Register

The serial mode register (SCSMR) is an eight-bit register that specifies the SCI serial communication format and selects the clock source for the baud rate generator.

The CPU can always read and write the SCSMR. The SCSMR is initialized to H'00 by a reset or in standby and module standby modes.

Bit:	7	6	5	4	3	2	1	0
Bit name:	C/A	CHR	PE	O/E	STOP	MP	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7—Communication Mode (C/$\overline{\text{A}}$): Selects whether the SCI operates in the asynchronous or clock synchronous mode.

Bit 7: C/A	Description	
0	Asynchronous mode	(Initial value)
1	Clock synchronous mode	

Bit 6—Character Length (CHR): Selects seven-bit or eight-bit data in the asynchronous mode. In the clock synchronous mode, the data length is always eight bits, regardless of the CHR setting.

Bit 6: CHR	Description	
0	Eight-bit data	(Initial value)
1	Seven-bit data. (When seven-bit data is selected, the MSB (bit 7) of the transmit data register is not transmitted.)	

Bit 5—Parity Enable (PE): Selects whether to add a parity bit to transmit data and to check the parity of receive data, in the asynchronous mode. In the clock synchronous mode, a parity bit is neither added nor checked, regardless of the PE setting.

Bit 5: PE	Description	
0	Parity bit not added or checked	(Initial value)
1	Parity bit added and checked. When PE is set to 1, an even or odd parity bit is added to transmit data, depending on the parity mode (O/E) setting. Receive data parity is checked according to the even/odd (O/E) mode setting.	

Bit 4—Parity Mode (O/$\overline{E}$): Selects even or odd parity when parity bits are added and checked. The O/$\overline{E}$ setting is used only in asynchronous mode and only when the parity enable bit (PE) is set to 1 to enable parity addition and check. The O/$\overline{E}$ setting is ignored in the clock synchronous mode, or in the asynchronous mode when parity addition and check is disabled.

Bit 4: O/E	Description	
0	Even parity. If even parity is selected, the parity bit is added to transmit data to make an even number of 1s in the transmitted character and parity bit combined. Receive data is checked to see if it has an even number of 1s in the received character and parity bit combined.	(Initial value)
1	Odd parity. If odd parity is selected, the parity bit is added to transmit data to make an odd number of 1s in the transmitted character and parity bit combined. Receive data is checked to see if it has an odd number of 1s in the received character and parity bit combined.	

Bit 3—Stop Bit Length (STOP): Selects one or two bits as the stop bit length in the asynchronous mode. This setting is used only in the asynchronous mode. It is ignored in the clock synchronous mode because no stop bits are added.

In receiving, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, it is treated as a stop bit, but if the second stop bit is 0, it is treated as the start bit of the next incoming character.

Bit 3: STOP	Description	
0	One stop bit In transmitting, a single bit of 1 is added at the end of each transmitted character.	(Initial value)
1	Two stop bits. In transmitting, two bits of 1 are added at the end of each transmitted character.	

Bit 2—Multiprocessor Mode (MP): Selects multiprocessor format. When multiprocessor format is selected, settings of the parity enable (PE) and parity mode (O/$\overline{E}$) bits are ignored. The MP bit setting is used only in the asynchronous mode; it is ignored in the clock synchronous mode. For the multiprocessor communication function, see section 17.3.3, Multiprocessor Communication.

Bit 2: MP	Description	
0	Multiprocessor function disabled	(Initial value)
1	Multiprocessor format selected	

Bits 1 and 0—Clock Select 1 and 0 (CKS1 and CKS0): These bits select the internal clock source of the on-chip baud rate generator. Four clock sources are available. Pϕ, Pϕ/4, Pϕ/16 and Pϕ/64. For further information on the clock source, bit rate register settings, and baud rate, see section 17.2.9, Bit Rate Register.

Bit 1: CKS1	Bit 0: CKS0	Description	
0	0	Pϕ	(Initial value)
	1	Pϕ/4	
1	0	Pϕ/16	
	1	Pϕ/64	

Note: Pϕ: Peripheral clock

17.2.6 Serial Control Register

The serial control register (SCSCR) operates the SCI transmitter/receiver, selects the serial clock output in the asynchronous mode, enables/disables interrupt requests, and selects the transmit/receive clock source. The CPU can always read and write the SCSCR. The SCSCR is initialized to H'00 by a reset or in standby and module standby modes.

Bit:	7	6	5	4	3	2	1	0
Bit name:	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7—Transmit Interrupt Enable (TIE): Enables or disables the transmit-data-empty interrupt (TXI) requested when the transmit data register empty bit (TDRE) in the serial status register (SCSSR) is set to 1 due to transfer of serial transmit data from the SCTDR to the SCTSR.

Bit 7: TIE	Description	
0	Transmit-data-empty interrupt request (TXI) is disabled. The TXI interrupt request can be cleared by reading TDRE after it has been set to 1, then clearing TDRE to 0, or by clearing TIE to 0.	(Initial value)
1	Transmit-data-empty interrupt request (TXI) is enabled.	

Bit 6—Receive Interrupt Enable (RIE): Enables or disables the receive-data-full interrupt (RXI) requested when the receive data register full bit (RDRF) in the serial status register (SCSSR) is set to 1 due to transfer of serial receive data from the SCRSR to the SCRDR. It also enables or disables receive-error interrupt (ERI) requests.

Bit 6: RIE	Description
0	Receive-data-full interrupt (RXI) and receive-error interrupt (ERI) requests are disabled. (Initial value) RXI and ERI interrupt requests can be cleared by reading the RDRF flag or error flag (FER, PER, or ORER) after it has been set to 1, then clearing the flag to 0, or by clearing RIE to 0.
1	Receive-data-full interrupt (RXI) and receive-error interrupt (ERI) requests are enabled.

Bit 5—Transmit Enable (TE): Enables or disables the SCI serial transmitter.

Bit 5: TE	Description
0	Transmitter disabled. (Initial value). The transmit data register empty bit (TDRE) in the serial status register (SCSSR) is fixed to 1.
1	Transmitter enabled. Serial transmission starts when the transmit data register empty (TDRE) bit in the serial status register (SCSSR) is cleared to 0 after writing of transmit data into the SCTDR. Select the transmit format in the SCSMR before setting TE to 1.

Bit 4—Receive Enable (RE): Enables or disables the SCI serial receiver.

Bit 4: RE	Description
0	Receiver disabled. (Initial value) Clearing RE to 0 does not affect the receive flags (RDRF, FER, PER, ORER). These flags retain their previous values.
1	Receiver enabled. Serial reception starts when a start bit is detected in the asynchronous mode, or synchronous clock input is detected in the clock synchronous mode. Select the receive format in the SCSMR before setting RE to 1.

Bit 3—Multiprocessor Interrupt Enable (MPIE): Enables or disables multiprocessor interrupts. The MPIE setting is used only in the asynchronous mode, and only if the multiprocessor mode bit (MP) in the serial mode register (SCSMR) is set to 1 during reception. The MPIE setting is ignored in the clock synchronous mode or when the MP bit is cleared to 0.

Bit 3: MPIE	Description
0	Multiprocessor interrupts are disabled (normal receive operation). (Initial value) MPE is cleared to 0 when MPIE is cleared to 0, or the multiprocessor bit (MPB) is set to 1 in receive data.
1	Multiprocessor interrupts are enabled. Receive-data-full interrupt requests (RXI), receive-error interrupt requests (ERI), and setting of the RDRF, FER, and ORER status flags in the serial status register (SCSSR) are disabled until data with a multiprocessor bit of 1 is received. The SCI does not transfer receive data from the SCRSR to the SCRDR, does not detect receive errors, and does not set the RDRF, FER, and ORER flags in the serial status register (SCSSR). When it receives data that includes MPB = 1, the SCSSR's MPB flag is set to 1, and the SCI automatically clears MPIE to 0, generates RXI and ERI interrupts (if the TIE and RIE bits in the SCSCR are set to 1), and allows the FER and ORER bits to be set.

Bit 2—Transmit-End Interrupt Enable (TEIE): Enables or disables the transmit-end interrupt (TEI) requested if SCTDR does not contain new transmit data when the MSB is transmitted.

Bit 2: TEIE	Description	
0	Transmit-end interrupt (TEI) requests are disabled*	(Initial value)
1	Transmit-end interrupt (TEI) requests are enabled.*	

Note: The TEI request can be cleared by reading the TDRE bit in the serial status register (SCSSR) after it has been set to 1, then clearing TDRE to 0 and clearing the transmit end (TEND) bit to 0, or by clearing the TEIE bit to 0.

Bits 1 and 0—Clock Enable 1 and 0 (CKE1 and CKE0): These bits select the SCI clock source and enable or disable clock output from the SCK pin. Depending on the combination of CKE1 and CKE0, the SCK pin can be used for serial clock output or serial clock input.

The CKE0 setting is valid only in the asynchronous mode, and only when the SCI is internally clock (CKE1 = 0). The CKE0 setting is ignored in the clock synchronous mode, or when an external clock source is selected (CKE1 = 1). Before selecting the SCI operating mode in the serial mode register (SCSMR), set CKE1 and CKE0. For further details on selection of the SCI clock source, see table 17.9 in section 17.3, Operation.

Bit 1: CKE1	Bit 0: CKE0	Description	
0	0	Asynchronous mode	Internal clock, SCK pin used for input pin (input signal is ignored) (Initial value)
		Clock synchronous mode	Internal clock, SCK pin used for synchronous clock output (Initial value)
	1	Asynchronous mode	Internal clock, SCK pin used for clock output*1
		Clock synchronous mode	Internal clock, SCK pin used for synchronous clock output
1	0	Asynchronous mode	External clock, SCK pin used for clock input*2
		Clock synchronous mode	External clock, SCK pin used for synchronous clock input
	1	Asynchronous mode	External clock, SCK pin used for clock input*2
		Clock synchronous mode	External clock, SCK pin used for synchronous clock input

Notes: 1. The output clock frequency is the same as the bit rate.
2 The input clock frequency is 16 times the bit rate.

17.2.7 Serial Status Register

The serial status register (SCSSR) is an 8-bit register containing multiprocessor bit values, and status flags that indicate SCI operating state.

The CPU can always read and write the SCSSR, but cannot write 1 in the status flags (TDRE, RDRF, ORER, PER, and FER). These flags can be cleared to 0 only if they have first been read (after being set to 1). Bits 2 (TEND) and 1 (MPB) are read-only bits that cannot be written. The SCSSR is initialized to H'84 by a reset or in standby and module standby modes.

Bit:	7	6	5	4	3	2	1	0
Bit name:	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial value:	1	0	0	0	0	1	0	0
R/W:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Note: The only value that can be written is a 0 to clear the flag.

Bit 7—Transmit Data Register Empty (TDRE): Indicates that the SCI has loaded transmit data from the SCTDR into the SCTSR and new serial transmit data can be written in the SCTDR.

Bit 7: TDRE	Description
0	SCTDR contains valid transmit data. TDRE is cleared to 0 when software reads TDRE after it has been set to 1, then writes 0 in TDRE or data is written in SCTDR.
1	SCTDR does not contain valid transmit data. (Initial value) TDRE is set to 1 when the chip is reset or enters standby mode, the TE bit in the serial control register (SCSCR) is cleared to 0, or SCTDR contents are loaded into SCTSR, so new data can be written in SCTDR.

Bit 6—Receive Data Register Full (RDRF): Indicates that SCRDR contains received data.

Bit 6: RDRF	Description
0	SCRDR does not contain valid received data. (Initial value) RDRF is cleared to 0 when the chip is reset or enters standby mode, or software reads RDRF after it has been set to 1, then writes 0 in RDRF.
1	SCRDR contains valid received data. RDRF is set to 1 when serial data is received normally and transferred from SCRSR to SCRDR.

Note: The SCRDR and RDRF are not affected by detection of receive errors or by clearing of the RE bit to 0 in the serial control register. They retain their previous contents. If RDRF is still set to 1 when reception of the next data ends, an overrun error (ORER) occurs and the received data is lost.

Bit 5—Overrun Error (ORER): Indicates that data reception aborted due to an overrun error.

Bit 5: ORER	Description
0	Receiving is in progress or has ended normally.*1 (Initial value) ORER is cleared to 0 when the chip is reset or enters standby mode or software reads ORER after it has been set to 1, then writes 0 in ORER.
1	A receive overrun error occurred.*2 ORER is set to 1 if reception of the next serial data ends when RDRF is set to 1.

Notes: 1. Clearing the RE bit to 0 in the serial control register does not affect the ORER bit, which retains its previous value.

2. SCRDR continues to hold the data received before the overrun error, so subsequent receive data is lost. Serial receiving cannot continue while ORER is set to 1. In the clock synchronous mode, serial transmitting is also disabled.

Bit 4—Framing Error (FER): Indicates that data reception aborted due to a framing error in the asynchronous mode.

Bit 4: FER	Description
0	Receiving is in progress or has ended normally. (Initial value) Clearing the RE bit to 0 in the serial control register does not affect the FER bit, which retains its previous value. FER is cleared to 0 when the chip is reset or enters standby mode or software reads FER after it has been set to 1, then writes 0 in FER.
1	A receive framing error occurred. When the stop bit length is two bits, only the first bit is checked. The second stop bit is not checked. When a framing error occurs, the SCI transfers the receive data into the SCRDR but does not set RDRF. Serial receiving cannot continue while FER is set to 1. In the clock synchronous mode, serial transmitting is also disabled. FER is set to 1 if the stop bit at the end of receive data is checked and found to be 0.

Bit 3—Parity Error (PER): Indicates that data reception (with parity) aborted due to a parity error in the asynchronous mode.

Bit 3: PER	Description
0	Receiving is in progress or has ended normally. (Initial value) Clearing the RE bit to 0 in the serial control register does not affect the PER bit, which retains its previous value. PER is cleared to 0 when the chip is reset or enters standby mode or software reads PER after it has been set to 1, then writes 0 in PER.
1	A receive parity error occurred. When a parity error occurs, the SCI transfers the receive data into the SCRDR but does not set RDRF. Serial receiving cannot continue while PER is set to 1. In the clock synchronous mode, serial transmitting is also disabled. PER is set to 1 if the number of 1s in receive data, including the parity bit, does not match the even or odd parity setting of the parity mode bit (O/$\overline{\text{E}}$) in the serial mode register (SCSMR).

Bit 2—Transmit End (TEND): Indicates that when the last bit of a serial character was transmitted, the SCTDR did not contain valid data, so transmission has ended. TEND is a read-only bit and cannot be written.

Bit 2: TEND	Description	
0	Transmission is in progress. TEND is cleared to 0 when software reads TDRE after it has been set to 1, then writes 0 in TDRE.	
1	End of transmission. TEND is set to 1 when the chip is reset or enters standby mode. TE is cleared to 0 in the serial control register (SCSCR), or TDRE is 1 when the last bit of a one-byte serial character is transmitted.	(Initial value)

Bit 1—Multiprocessor Bit (MPB): Stores the value of the multiprocessor bit in receive data when a multiprocessor format is selected for receiving in the asynchronous mode. The MPB is a read-only bit and cannot be written.

Bit 1: MPB	Description	
0	Multiprocessor bit value in receive data is 0. If RE is cleared to 0 when a multiprocessor format is selected, the MPB retains its previous value.	(Initial value)
1	Multiprocessor bit value in receive data is 1.	

Bit 0—Multiprocessor Bit Transfer (MPBT): Stores the value of the multiprocessor bit added to transmit data when a multiprocessor format is selected for transmitting in the asynchronous mode. The MPBT setting is ignored in the clock synchronous mode, when a multiprocessor format is not selected, or when the SCI is not transmitting.

Bit 0: MPBT	Description	
0	Multiprocessor bit value in transmit data is 0	(Initial value)
1	Multiprocessor bit value in transmit data is 1	

17.2.8 Port SC Control Register (SCPCR)/Port SC Data Register (SCPDR)

The port SC control register (SCPCR) and port SC data register (SCPDR) control I/O and data for the port multiplexed with the serial communication interface (SCI) pins.

SCPCR settings are used to perform I/O control, to enable data written in SCPDR to be output to the TxD pin, and input data to be read from the RxD pin, and to control serial transmission/reception breaks.

It is also possible to read data on the SCK pin, and write output data.

SCPCR

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	SCP7 MD1	SCP7 MD0	SCP6 MD1	SCP6 MD0	SCP5 MD1	SCP5 MD0	SCP4 MD1	SCP4 MD0	SCP3 MD1	SCP3 MD0	SCP2 MD1	SCP2 MD0	SCP1 MD1	SCP1 MD0	SCP0 MD1	SCP0 MD0
Initial value:	1	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SCPDR

Bit:	7	6	5	4	3	2	1	0
Bit name:	SCP7DT	SCP6DT	SCP5DT	SCP4DT	SCP3DT	SCP2DT	SCP1DT	SCP0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SCI pin I/O and data control are performed by bits 3–0 of SCPCR and bits 1 and 0 of SCPDR.

SCPCR bits 3 and 2—Serial Clock Port I/O (SCP1MD1, SCP1MD0): These bits specify serial port SCK pin I/O. When the SCK pin is actually used as a port I/O pin, clear the $C/\overline{A}$ bit of SCSMR and bits CKE1 and CKE0 of SCSCR to 0.

Bit 3: SCP1MD1	Bit 2: SCP1MD0	Description
0	0	SCP1DT bit value is not output to SCK pin
0	1	SCP1DT bit value is output to SCK pin
1	0	SCK pin value is read from SCP1DT bit (Initial values: 1 and 0)
1	1	

SCPCR bit 1—Serial Clock Port Data (SCP1DT): Specifies the serial port SCK pin I/O data. Input or output is specified by the SCP1MD0 and SCP1MD1 bits. In output mode, the value of the SCP1DT bit is output to the SCK pin. In output mode, the SCK pin value is read from the SCP1DT bit.

Bit 1: SCP1DT	Description	
0	I/O data is low	(Initial value)
1	I/O data is high	

SCPCR bits 1 and 0—Serial Port Break I/O (SCP0MD1, SCP0MD0): These bits specify the serial port TxD pin output condition. When the TxD pin is actually used as a port output pin and outputs the value set with the SCP0DT bit, clear the TE bit of SCSCR to 0.

Bit 1: SCP0MD1	Bit 0: SCP0MD0	Description	
0	0	SCP0DT bit value is not output to TxD pin	(Initial value)
0	1	SCP0DT bit value is output to TxD pin	

SCPCR bit 0—Serial Port Break Data (SCP0DT): Specifies the serial port RxD pin input data and TxD pin output data. The TxD pin output condition is specified by the SCP0MD0 and SCP0MD1 bits. When the TxD pin is set to output mode, the value of the SCP0DT bit is output to the TxD pin. The RxD pin value is read from the SCP0DT bit regardless of the values of the SCP0MD0 and SCP0MD1 bits, if RE in the SCSCR is set to 1. The initial value of this bit after a power-on reset is undefined.

Bit 0: SCP0DT	Description	
0	I/O data is low	(Initial value)
1	I/O data is high	

Block diagrams of the SCI I/O ports are shown in figures 17.2, 17.3, and 17.4.

17.2.9 Bit Rate Register (SCBRR)

The bit rate register (SCBRR) is an eight-bit register that, together with the baud rate generator clock source selected by the CKS1 and CKS0 bits in the serial mode register (SCSMR), determines the serial transmit/receive bit rate.

The CPU can always read and write the SCBRR. The SCBRR is initialized to H'FF by a reset or in module standby or standby mode. Each channel has independent baud rate generator control, so different values can be set in two channels.

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The SCBRR setting is calculated as follows:

Asynchronous mode: $N = [P\phi/(64 \times 2^{2n-1} \times B)] \times 10^6 - 1$

Clock synchronous mode: $N = [P\phi/(8 \times 2^{2n-1} \times B)] \times 10^6 - 1$

B: Bit rate (bit/s)
N: SCBRR setting for baud rate generator ($0 \leq N \leq 255$)
Pϕ: Operating frequency for peripheral modules (MHz)
n: Baud rate generator clock source (n = 0, 1, 2, 3) (for the clock sources and values of n, see table 17.3.)

Table 17.3SCSMR Settings

		SCSMR Settings	
n	**Clock Source**	**CKS1**	**CKS0**
0	Pϕ	0	0
1	Pϕ/4	0	1
2	Pϕ/16	1	0
3	Pϕ/64	1	1

Note: Find the bit rate error for the asynchronous mode by the following formula:
Error (%) = $\{P(\phi \times 10^6)/[(N + 1) \times B \times 64 \times 2^{2n-1}] - 1\} \times 100$

Table 17.4 lists examples of SCBRR settings in the asynchronous mode; table 17.5 lists examples of SCBRR settings in the clock synchronous mode.

Table 17.4Bit Rates and SCBRR Settings in Asynchronous Mode

	Pϕ (MHz)								
	2			2.097152			2.4576		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	141	0.03	1	148	–0.04	1	174	–0.26
150	1	103	0.16	1	108	0.21	1	127	0.00
300	0	207	0.16	0	217	0.21	0	255	0.00
600	0	103	0.16	0	108	0.21	0	127	0.00
1200	0	51	0.16	0	54	–0.70	0	63	0.00
2400	0	25	0.16	0	26	1.14	0	31	0.00
4800	0	12	0.16	0	13	–2.48	0	15	0.00
9600	0	6	–6.99	0	6	–2.48	0	7	0.00
19200	0	2	8.51	0	2	13.78	0	3	0.00
31250	0	1	0.00	0	1	4.86	0	1	22.88
38400	0	1	–18.62	0	1	–14.67	0	1	0.00

	Pϕ (MHz)								
	3			3.6864			4		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	212	0.03	2	64	0.70	2	70	0.03
150	1	155	0.16	1	191	0.00	1	207	0.16
300	1	77	0.16	1	95	0.00	1	103	0.16
600	0	155	0.16	0	191	0.00	0	207	0.16
1200	0	77	0.16	0	95	0.00	0	103	0.16
2400	0	38	0.16	0	47	0.00	0	51	0.16
4800	0	19	–2.34	0	23	0.00	0	25	0.16
9600	0	9	–2.34	0	11	0.00	0	12	0.16
19200	0	4	–2.34	0	5	0.00	0	6	–6.99
31250	0	2	0.00	—	—	—	0	3	0.00
38400	—	—	—	0	2	0.00	0	2	8.51

Table 17.4Bit Rates and SCBRR Settings in Asynchronous Mode (cont)

	Pφ (MHz)								
	4.9152			5			6		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	86	0.31	2	88	–0.25	2	106	–0.44
150	1	255	0.00	2	64	0.16	2	77	0.16
300	1	127	0.00	1	129	0.16	1	155	0.16
600	0	255	0.00	1	64	0.16	1	77	0.16
1200	0	127	0.00	0	129	0.16	0	155	0.16
2400	0	63	0.00	0	64	0.16	0	77	0.16
4800	0	31	0.00	0	32	–1.36	0	38	0.16
9600	0	15	0.00	0	15	1.73	0	19	–2.34
19200	0	7	0.00	0	7	1.73	0	9	–2.34
31250	0	4	–1.70	0	4	0.00	0	5	0.00
38400	0	3	0.00	0	3	1.73	0	4	–2.34

	Pφ (MHz)								
	6.144			7.3728			8		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	108	0.08	2	130	–0.07	2	141	0.03
150	2	79	0.00	2	95	0.00	2	103	0.16
300	1	159	0.00	1	191	0.00	1	207	0.16
600	1	79	0.00	1	95	0.00	1	103	0.16
1200	0	159	0.00	0	191	0.00	0	207	0.16
2400	0	79	0.00	0	95	0.00	0	103	0.16
4800	0	39	0.00	0	47	0.00	0	51	0.16
9600	0	19	0.00	0	23	0.00	0	25	0.16
19200	0	9	0.00	0	11	0.00	0	12	0.16
31250	0	5	2.40	0	6	5.33	0	7	0.00
38400	0	4	0.00	0	5	0.00	0	6	–6.99

Table 17.4Bit Rates and SCBRR Settings in Asynchronous Mode (cont)

	Pϕ (MHz)											
	9.8304			10			12			12.288		
Bit Rate			Error			Error			Error			Error
(bits/s)	n	N	(%)	n	N	(%)	n	N	(%)	n	N	(%)
110	2	174	–0.26	2	177	–0.25	2	212	0.03	2	217	0.08
150	2	127	0.00	2	129	0.16	2	155	0.16	2	159	0.00
300	1	255	0.00	2	64	0.16	2	77	0.16	2	79	0.00
600	1	127	0.00	1	129	0.16	1	155	0.16	1	159	0.00
1200	0	255	0.00	1	64	0.16	1	77	0.16	1	79	0.00
2400	0	127	0.00	0	129	0.16	0	155	0.16	0	159	0.00
4800	0	63	0.00	0	64	0.16	0	77	0.16	0	79	0.00
9600	0	31	0.00	0	32	–1.36	0	38	0.16	0	39	0.00
19200	0	15	0.00	0	15	1.73	0	19	0.16	0	19	0.00
31250	0	9	–1.70	0	9	0.00	0	11	0.00	0	11	2.40
38400	0	7	0.00	0	7	1.73	0	9	–2.34	0	9	0.00

	Pϕ (MHz)											
	14.7456			16			19.6608			20		
Bit Rate			Error			Error			Error			Error
(bits/s)	n	N	(%)	n	N	(%)	n	N	(%)	n	N	(%)
110	3	64	0.70	3	70	0.03	3	86	0.31	3	88	–0.25
150	2	191	0.00	2	207	0.16	2	255	0.00	3	64	0.16
300	2	95	0.00	2	103	0.16	2	127	0.00	2	129	0.16
600	1	191	0.00	1	207	0.16	1	255	0.00	2	64	0.16
1200	1	95	0.00	1	103	0.16	1	127	0.00	1	129	0.16
2400	0	191	0.00	0	207	0.16	0	255	0.00	1	64	0.16
4800	0	95	0.00	0	103	0.16	0	127	0.00	0	129	0.16
9600	0	47	0.00	0	51	0.16	0	63	0.00	0	64	0.16
19200	0	23	0.00	0	25	0.16	0	31	0.00	0	32	–1.36
31250	0	14	–1.70	0	15	0.00	0	19	–1.70	0	19	0.00
38400	0	11	0.00	0	12	0.16	0	15	0.00	0	15	1.73

Table 17.4Bit Rates and SCBRR Settings in Asynchronous Mode (cont)

	Pφ (MHz)											
	24			24.576			28.7			30		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	106	–0.44	3	108	0.08	3	126	0.31	3	132	0.13
150	3	77	0.16	3	79	0.00	3	92	0.46	3	97	–0.35
300	2	155	0.16	2	159	0.00	2	186	–0.08	2	194	0.16
600	2	77	0.16	2	79	0.00	2	92	0.46	2	97	–0.35
1200	1	155	0.16	1	159	0.00	1	186	–0.08	1	194	0.16
2400	1	77	0.16	1	79	0.00	1	92	0.46	1	97	–0.35
4800	0	155	0.16	0	159	0.00	0	186	–0.08	0	194	–1.36
9600	0	77	0.16	0	79	0.00	0	92	0.46	0	97	–0.35
19200	0	38	0.16	0	39	0.00	0	46	–0.61	0	48	–0.35
31250	0	23	0.00	0	24	–1.70	0	28	–1.03	0	29	0.00
38400	0	19	–2.34	0	19	0.00	0	22	1.55	0	23	1.73

Table 17.5Bit Rates and SCBRR Settings in Clock Synchronous Mode

Bit Rate (bits/s)	Pφ (MHz) 4 n	N	8 n	N	16 n	N	28.7 n	N	30 n	N
110	—	—	—	—	—	—	—	—	—	—
250	2	249	3	124	3	249	—	—	—	—
500	2	124	2	249	3	124	3	223	3	233
1k	1	249	2	124	2	249	3	111	3	116
2.5k	1	99	1	199	2	99	2	178	2	187
5k	0	199	1	99	1	199	2	89	2	93
10k	0	99	0	199	1	99	1	178	1	187
25k	0	39	0	79	0	159	1	71	1	74
50k	0	19	0	39	0	79	0	143	0	149
100k	0	9	0	19	0	39	0	71	0	74
250k	0	3	0	7	0	15	—	—	0	29
500k	0	1	0	3	0	7	—	—	0	14
1M	0	0*	0	1	0	3	—	—	—	—
2M			0	0*	0	1	—	—	—	—

Note: Settings with an error of 1% or less are recommended.

Blank : No setting possible

— : Setting possible, but error occurs

* : Continuous transmit/receive not possible

Table 17.6 indicates the maximum bit rates in the asynchronous mode when the baud rate generator is being used. Tables 17.7 and 17.8 list the maximum rates for external clock input.

Table 17.6Maximum Bit Rates for Various Frequencies with Baud Rate Generator (Asynchronous Mode)

		Settings	
Pϕ (MHz)	Maximum Bit Rate (bits/s)	n	N
2	62500	0	0
2.097152	65536	0	0
2.4576	76800	0	0
3	93750	0	0
3.6864	115200	0	0
4	125000	0	0
4.9152	153600	0	0
8	250000	0	0
9.8304	307200	0	0
12	375000	0	0
14.7456	460800	0	0
16	500000	0	0
19.6608	614400	0	0
20	625000	0	0
24	750000	0	0
24.576	768000	0	0
28.7	896875	0	0
30	937500	0	0

Table 17.7 Maximum Bit Rates during External Clock Input (Asynchronous Mode)

Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bits/s)
2	0.5000	31250
2.097152	0.5243	32768
2.4576	0.6144	38400
3	0.7500	46875
3.6864	0.9216	57600
4	1.0000	62500
4.9152	1.2288	76800
8	2.0000	125000
9.8304	2.4576	153600
12	3.0000	187500
14.7456	3.6864	230400
16	4.0000	250000
19.6608	4.9152	307200
20	5.0000	312500
24	6.0000	375000
24.576	6.1440	384000
28.7	7.1750	448436
30	7.5000	468750

Table 17.8 Maximum Bit Rates during External Clock Input (Clock Synchronous Mode)

Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bits/s)
8	1.3333	1333333.3
16	2.6667	2666666.7
24	4.0000	4000000.0
28.7	4.7833	4783333.3
30	5.0000	5000000.0

17.3 Operation

17.3.1 Overview

For serial communication, the SCI has an asynchronous mode in which characters are synchronized individually, and a clock synchronous mode in which communication is synchronized with clock pulses. Asynchronous/clock synchronous mode and the transmission format are selected in the serial mode register (SCSMR), as listed in table 17.9. The SCI clock source is selected by the combination of the C/$\overline{\text{A}}$ bit in the serial mode register (SCSMR) and the CKE1 and CKE0 bits in the serial control register (SCSCR), as listed in table 17.10.

Asynchronous Mode:

- Data length is selectable: seven or eight bits.
- Parity and multiprocessor bits are selectable. So is the stop bit length (one or two bits). The combination of the preceding selections constitutes the communication format and character length.
- In receiving, it is possible to detect framing errors (FER), parity errors (PER), overrun errors (ORER) and breaks.
- An internal or external clock can be selected as the SCI clock source.
  - — When an internal clock is selected, the SCI operates using the on-chip baud rate generator, and can output a serial clock signal with a frequency matching the bit rate.
  - — When an external clock is selected, the external clock input must have a frequency 16 times the bit rate. (The on-chip baud rate generator is not used.)

Clock Synchronous Mode:

- The transmission/reception format has a fixed eight-bit data length.
- In receiving, it is possible to detect overrun errors (ORER).
- An internal or external clock can be selected as the SCI clock source.
  - — When an internal clock is selected, the SCI operates using the on-chip baud rate generator, and outputs a synchronous clock signal to external devices.
  - — When an external clock is selected, the SCI operates on the input synchronous clock. The on-chip baud rate generator is not used.

Table 17.9Serial Mode Register Settings and SCI Communication Formats

	SCSMR Settings					SCI Communication Format			
Mode	**Bit 7 C/A**	**Bit 6 CHR**	**Bit 5 PE**	**Bit 2 MP**	**Bit 3 STOP**	**Data Length**	**Parity Bit**	**Multiprocessor Bit**	**Stop Bit Length**
Asynchronous	0	0	0	0	0	8-bit	Not set	Not set	1 bit
					1				2 bits
			1		0		Set		1 bit
					1				2 bits
		1	0		0	7-bit	Not set		1 bit
					1				2 bits
			1		0		Set		1 bit
					1				2 bits
Asynchronous (multiprocessor format)		0	*	1	0	8-bit	Not set	Set	1 bit
			*		1				2 bits
		1	*		0	7-bit			1 bit
			*		1				2 bits
Clock synchronous	1	*	*	*	*	8-bit		Not set	None

Note: Asterisks (*) indicate don't-care bits.

Table 17.10 SCSMR and SCSCR Settings and SCI Clock Source Selection

	SCSMR	SCSCR Settings		SCI Transmit/Receive Clock	
Mode	**Bit 7 C/A**	**Bit 1 CKE1**	**Bit 0 CKE0**	**Clock Source**	**SCK Pin Function**
Asynchronous mode	0	0	0	Internal	SCI does not use the SCK pin
			1		Outputs a clock with frequency matching the bit rate
		1	0	External	Inputs a clock with frequency 16 times the bit rate
			1		
Clock synchronous mode	1	0	0	Internal	Outputs the synchronous clock
			1		
		1	0	External	Inputs the synchronous clock
			1		

17.3.2 Operation in Asynchronous Mode

In the asynchronous mode, each transmitted or received character begins with a start bit and ends with a stop bit. Serial communication is synchronized one character at a time.

The transmitting and receiving sections of the SCI are independent, so full duplex communication is possible. The transmitter and receiver are both double buffered, so data can be written and read while transmitting and receiving are in progress, enabling continuous transmitting and receiving.

Figure 17.5 shows the general format of asynchronous serial communication. In asynchronous serial communication, the communication line is normally held in the mark (high) state. The SCI monitors the line and starts serial communication when the line goes to the space (low) state, indicating a start bit. One serial character consists of a start bit (low), data (LSB first), parity bit (high or low), and stop bit (high), in that order.

When receiving in the asynchronous mode, the SCI synchronizes on the falling edge of the start bit. The SCI samples each data bit on the eighth pulse of a clock with a frequency 16 times the bit rate. Receive data is latched at the center of each bit.

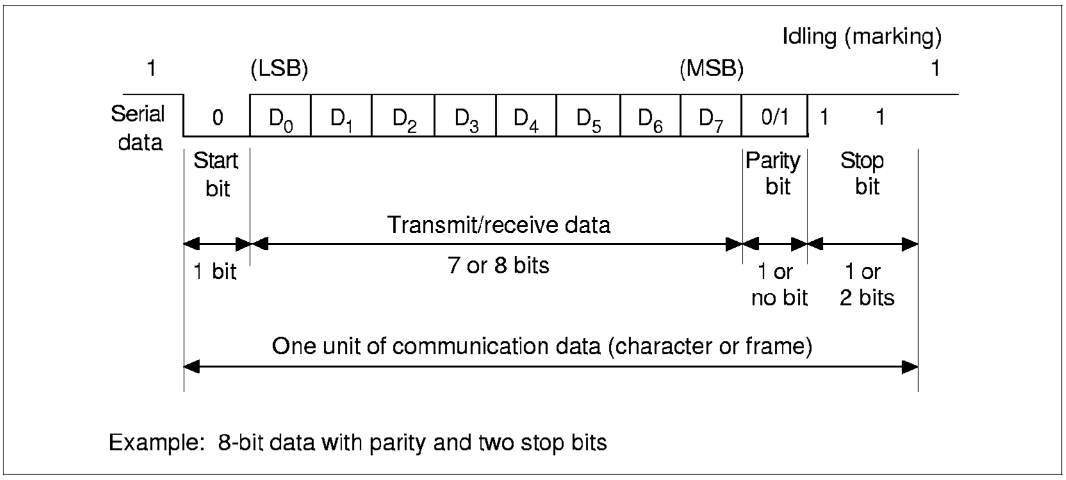


Figure 17.5 Data Format in Asynchronous Communication

Transmit/Receive Formats: Table 17.11 lists the 12 communication formats that can be selected in the asynchronous mode. The format is selected by settings in the serial mode register (SCSMR).

Table 17.11 Serial Communication Formats (Asynchronous Mode)

SCSMR Bits				Serial Transmit/Receive Format and Frame Length											
CHR	PE	MP	STOP	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	START	8-Bit data								STOP		
0	0	0	1	START	8-Bit data								STOP	STOP	
0	1	0	0	START	8-Bit data								P	STOP	
0	1	0	1	START	8-Bit data								P	STOP	STOP
1	0	0	0	START	7-Bit data							STOP			
1	0	0	1	START	7-Bit data							STOP	STOP		
1	1	0	0	START	7-Bit data							P	STOP		
1	1	0	1	START	7-Bit data							P	STOP	STOP	
0	—	1	0	START	8-Bit data								MPB	STOP	
0	—	1	1	START	8-Bit data								MPB	STOP	STOP
1	—	1	0	START	7-Bit data							MPB	STOP		
1	—	1	1	START	7-Bit data							MPB	STOP	STOP	

Notes: — : Don't-care bits
START: Start bit
STOP: Stop bit
P: Parity bit
MPB: Multiprocessor bit

Clock: An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected as the SCI transmit/receive clock. The clock source is selected by the $C/\overline{A}$ bit in the serial mode register (SCSMR) and bits CKE1 and CKE0 in the serial control register (SCSCR) (table 17.10).

When an external clock is input at the SCK pin, it must have a frequency equal to 16 times the desired bit rate.

When the SCI operates on an internal clock, it can output a clock signal at the SCK pin. The frequency of this output clock is equal to the bit rate. The phase is aligned as in figure 17.6 so that the rising edge of the clock occurs at the center of each transmit data bit.

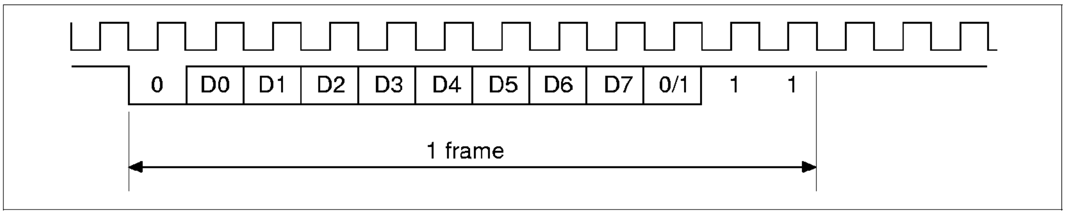


Figure 17.6 Output Clock and Serial Data Timing (Asynchronous Mode)

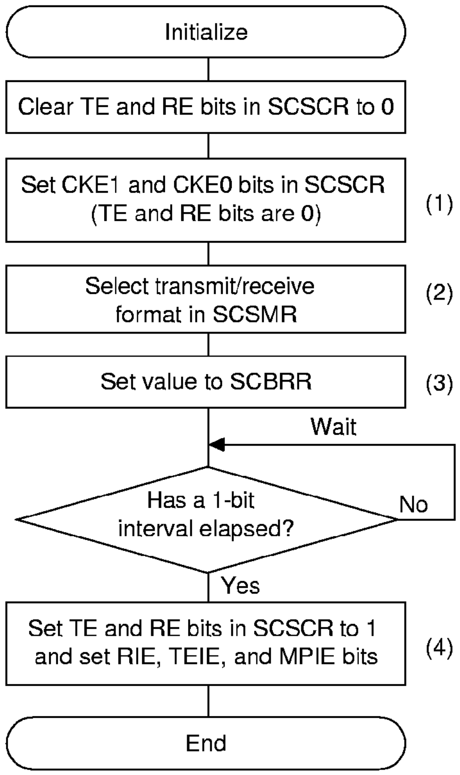
Transmitting and Receiving Data (SCI Initialization (Asynchronous Mode)): Before transmitting or receiving, clear the TE and RE bits to 0 in the serial control register (SCSCR), then initialize the SCI as follows.

When changing the operation mode or communication format, always clear the TE and RE bits to 0 before following the procedure given below. Clearing TE to 0 sets TDRE to 1 and initializes the transmit shift register (SCTSR). Clearing RE to 0, however, does not initialize the RDRF, PER, FER, and ORER flags or receive data register (SCRDR), which retain their previous contents.

When an external clock is used, the clock should not be stopped during initialization or subsequent operation. SCI operation becomes unreliable if the clock is stopped.

Figure 17.7 is a sample flowchart for initializing the SCI. The procedure for initializing the SCI is:

1. Select the clock source in the serial control register (SCSCR). Leave RIE, TIE, TEIE, MPIE, TE, and RE cleared to 0. If clock output is selected in asynchronous mode, clock output starts immediately after the setting is made to SCSCR.
2. Select the communication format in the serial mode register (SCSMR).
3. Write the value corresponding to the bit rate in the bit rate register (SCBRR) unless an external clock is used.
4. Wait for at least the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCSCR) to 1. Also set RIE, TIE, TEIE, and MPIE as necessary. Setting TE or RE enables the SCI to use the TxD or RxD pin. The initial states are the mark transmit state, and the idle receive state (waiting for a start bit).

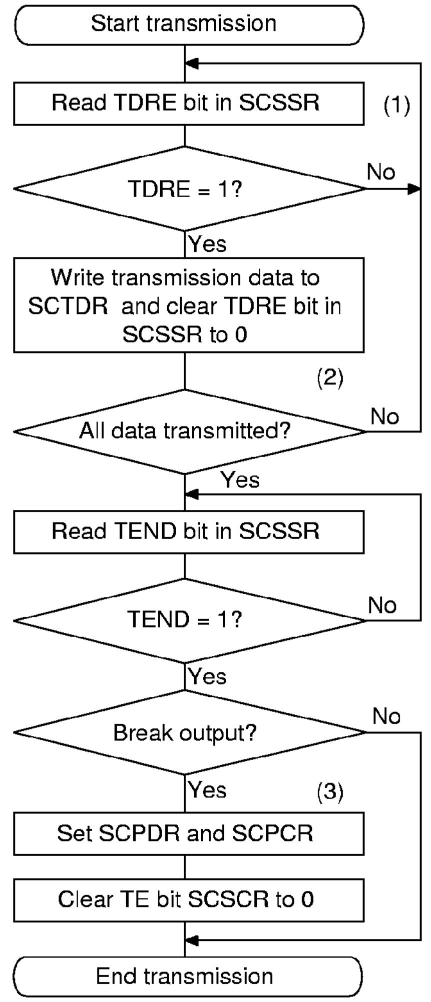

Initialize
Clear TE and RE bits in SCSCR to 0
Set CKE1 and CKE0 bits in SCSCR (TE and RE bits are 0)
(1)
Select transmit/receive format in SCSMR
(2)
Set value to SCBRR
(3)
Wait
Has a 1-bit interval elapsed?
No
Yes
Set TE and RE bits in SCSCR to 1 and set RIE, TEIE, and MPIE bits
(4)
End

Note: Circled numbers refer to the preceding procedure.

Figure 17.7 Sample Flowchart for SCI Initialization

Transmitting Serial Data (Asynchronous Mode): Figure 17.8 shows a sample flowchart for transmitting serial data. The procedure for transmitting serial data is:

1. SCI status check and transmit data write: read the serial status register (SCSSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (SCTDR) and clear TDRE to 0.
2. To continue transmitting serial data: Read the TDRE bit to check whether it is safe to write (if it reads 1); if so, write data in SCTDR, then clear TDRE to 0.
3. To output a break at the end of serial transmission: Set the port SC data register (SCPDR) and port SC control register (SCPCR), then clear the TE bit to 0 in the serial control register (SCSCR). For SCPCR and SCPDR settings, see section 17.2.8.



Note: Circled numbers refer to the preceding procedure.

Figure 17.8 Sample Flowchart for Transmitting Serial Data

In transmitting serial data, the SCI operates as follows:

1. The SCI monitors the TDRE bit in the SCSSR. When TDRE is cleared to 0, the SCI recognizes that the transmit data register (SCTDR) contains new data, and loads this data from the SCTDR into the transmit shift register (SCTSR).
2. After loading the data from the SCTDR into the SCTSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) is set to 1 in the SCSCR, the SCI requests a transmit-data-empty interrupt (TXI) at this time. Serial transmit data is transmitted in the following order from the TxD pin:
   a. Start bit: One 0 bit is output.
   b. Transmit data: Seven or eight bits of data are output, LSB first.
   c. Parity bit or multiprocessor bit: One parity bit (even or odd parity) or one multiprocessor bit is output. Formats in which neither a parity bit nor a multiprocessor bit is output can also be selected.
   d. Stop bit: One or two 1 bits (stop bits) are output.
   e. Marking: Output of 1 bits continues until the start bit of the next transmit data.
3. The SCI checks the TDRE bit when it outputs the stop bit. If TDRE is 0, the SCI loads new data from the SCTDR into the SCTSR, outputs the stop bit, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit to 1 in the SCSSR, outputs the stop bit, then continues output of 1 bits (marking). If the transmit-end interrupt enable bit (TEIE) in the SCSCR is set to 1, a transmit-end interrupt (TEI) is requested.

Figure 17.9 shows an example of SCI transmit operation in the asynchronous mode.

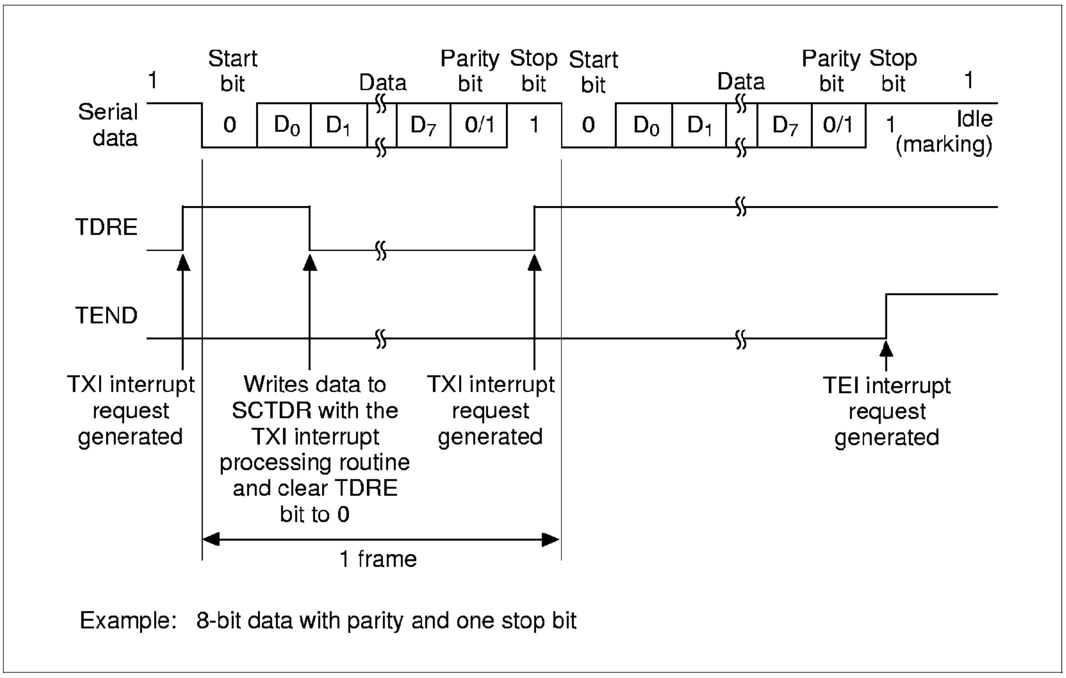
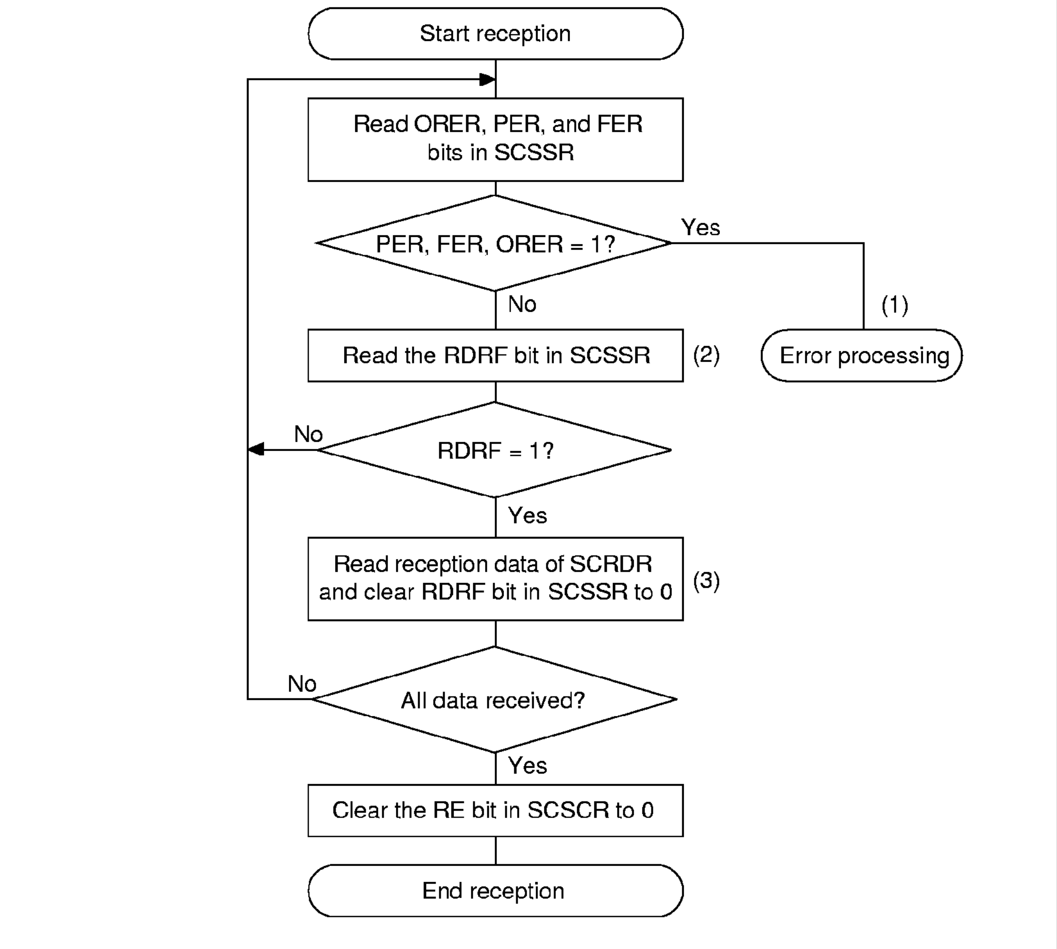

Start bit
Parity bit
Stop bit
Start bit
Data
Parity bit
Stop bit
1
Serial data
0
D0
D1
D7
0/1
1
Idle (marking)
TDRE
TEND
TXI interrupt request generated
Writes data to SCTDR with the TXI interrupt processing routine and clear TDRE bit to 0
TXI interrupt request generated
TEI interrupt request generated
1 frame
Example: 8-bit data with parity and one stop bit

Figure 17.9 SCI Transmit Operation in Asynchronous Mode

Receiving Serial Data (Asynchronous Mode): Figure 17.10 shows a sample flowchart for receiving serial data. The procedure for receiving serial data after enabling the SCI for reception is:

1. Receive error processing and break detection: If a receive error occurs, read the ORER, PER and FER bits of the SCSSR to identify the error. After executing the necessary error processing, clear ORER, PER and FER all to 0. Receiving cannot resume if ORER, PER or FER remain set to 1. When a framing error occurs, the RxD pin can be read to detect the break state.
2. SCI status check and receive-data read: Read the serial status register (SCSSR), check that RDRF is set to 1, then read receive data from the receive data register (SCRDR) and clear RDRF to 0. The RXI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
3. To continue receiving serial data: Read the RDRF and SCRDR bits and clear RDRF to 0 before the stop bit of the current frame is received.


Start reception
Read ORER, PER, and FER bits in SCSSR
PER, FER, ORER = 1?
Yes
No
(1)
Read the RDRF bit in SCSSR
(2)
Error processing
No
RDRF = 1?
Yes
Read reception data of SCRDR and clear RDRF bit in SCSSR to 0
(3)
No
All data received?
Yes
Clear the RE bit in SCSCR to 0
End reception

Note: Circled numbers refer to the preceding procedure.

Figure 17.10 Sample Flowchart for Receiving Serial Data

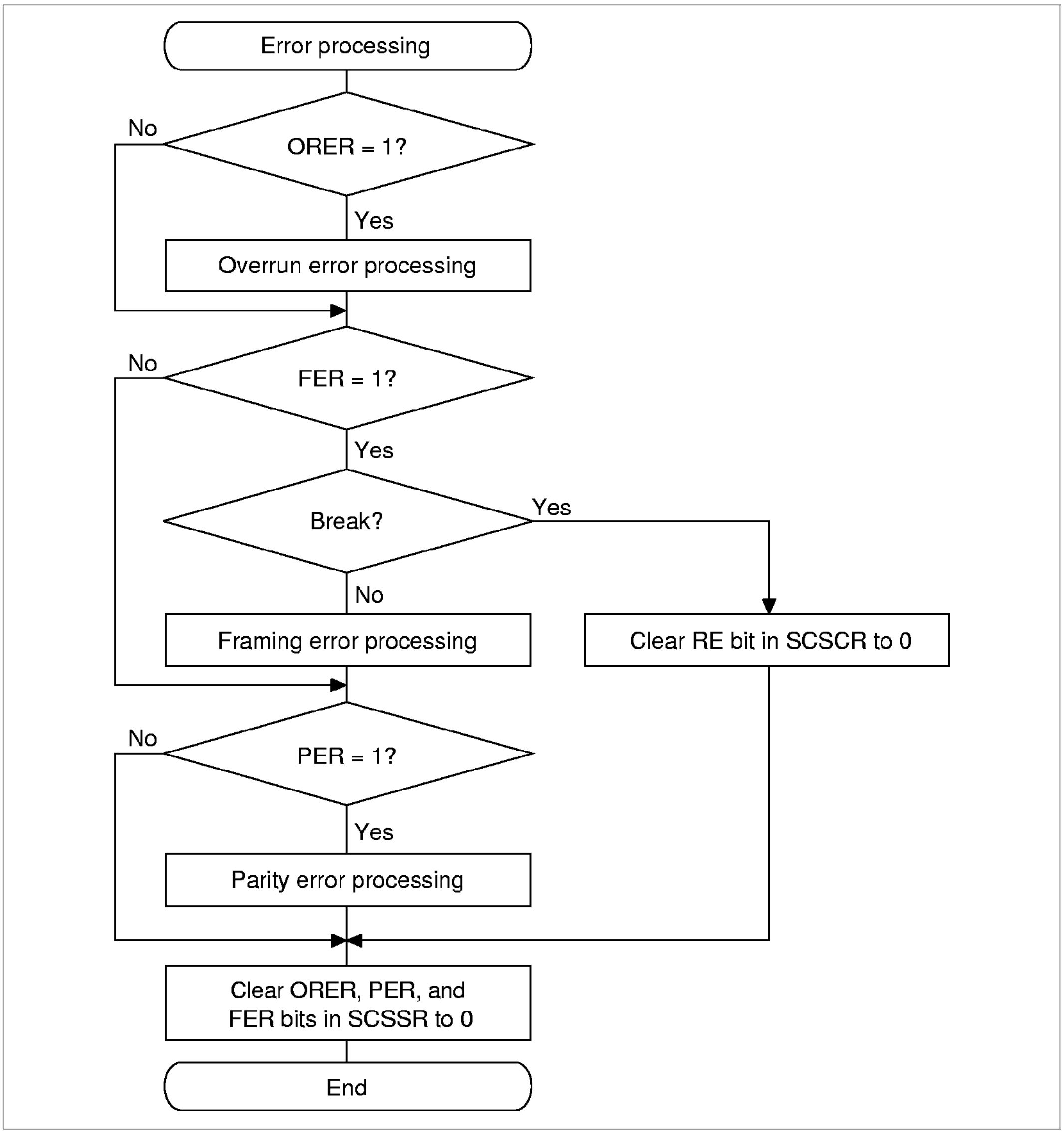


Figure 17.10 Sample Flowchart for Receiving Serial Data (cont)

In receiving, the SCI operates as follows:

1. The SCI monitors the communication line. When it detects a start bit (0), the SCI synchronizes internally and starts receiving.
2. Receive data is shifted into the SCRSR in order from the LSB to the MSB.
3. The parity bit and stop bit are received. After receiving these bits, the SCI makes the following checks:
   a. Parity check: The number of 1s in the receive data must match the even or odd parity setting of the O/$\overline{\text{E}}$ bit in the SCSMR.
   b. Stop bit check: The stop bit value must be 1. If there are two stop bits, only the first stop bit is checked.
   c. Status check: RDRF must be 0 so that receive data can be loaded from the SCRSR into the SCRDR.

   If these checks all pass, the SCI sets RDRF to 1 and stores the received data in the SCRDR. If one of the checks fails (receive error), the SCI operates as indicated in table 17.12.

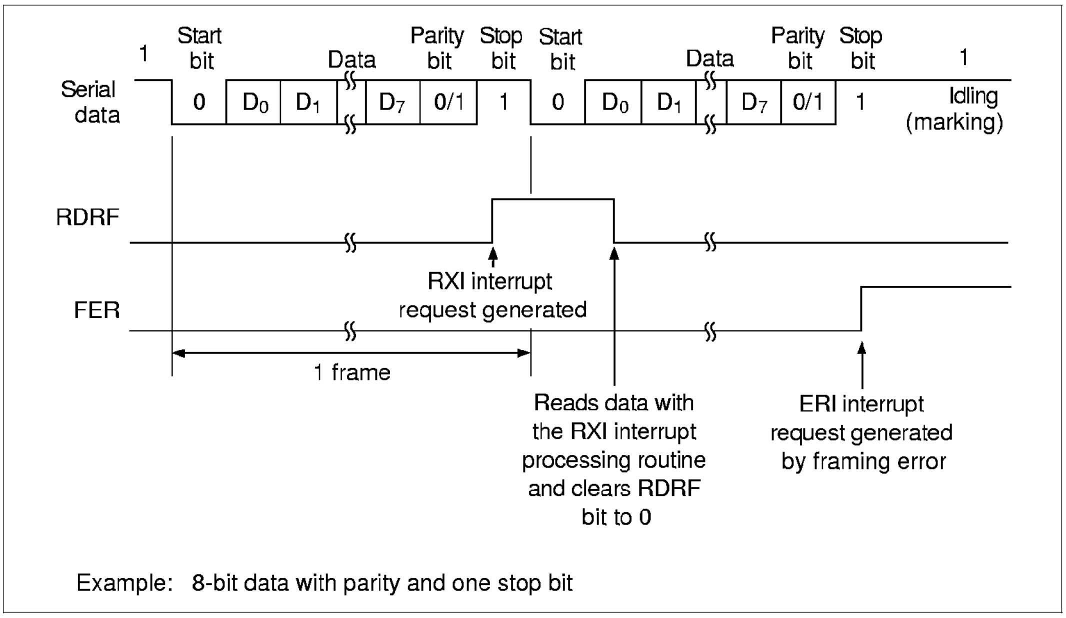
Note: When a receive error flag is set, further receiving is disabled. The RDRF bit is not set to 1. Be sure to clear the error flags.

4. After setting RDRF to 1, if the receive-data-full interrupt enable bit (RIE) is set to 1 in the SCSCR, the SCI requests a receive-data-full interrupt (RXI). If one of the error flags (ORER, PER, or FER) is set to 1 and the receive-data-full interrupt enable bit (RIE) in the SCSCR is also set to 1, the SCI requests a receive-error interrupt (ERI).

Table 17.12 Receive Error Conditions and SCI Operation

Receive Error	Abbreviation	Condition	Data Transfer
Overrun error	ORER	Receiving of next data ends while RDRF is still set to 1 in SCSSR	Receive data not loaded from SCRSR into SCRDR
Framing error	FER	Stop bit is 0	Receive data loaded from SCRSR into SCRDR
Parity error	PER	Parity of receive data differs from even/odd parity setting in SCSMR	Receive data loaded from SCRSR into SCRDR

Figure 17.11 shows an example of SCI receive operation in the asynchronous mode.



Start bit
Data
Parity bit
Stop bit
Start bit
Data
Parity bit
Stop bit
1
1
Serial data
0
D0
D1
D7
0/1
1
0
D0
D1
D7
0/1
1
Idling (marking)
RDRF
FER
RXI interrupt request generated
1 frame
Reads data with the RXI interrupt processing routine and clears RDRF bit to 0
ERI interrupt request generated by framing error
Example: 8-bit data with parity and one stop bit

Figure 17.11 SCI Receive Operation

17.3.3 Multiprocessor Communication

The multiprocessor communication function enables several processors to share a single serial communication line. The processors communicate in the asynchronous mode using a format with an additional multiprocessor bit (multiprocessor format).

In multiprocessor communication, each receiving processor is addressed by a unique ID. A serial communication cycle consists of an ID-sending cycle that identifies the receiving processor, and a data-sending cycle. The multiprocessor bit distinguishes ID-sending cycles from data-sending cycles. The transmitting processor starts by sending the ID of the receiving processor with which it wants to communicate as data with the multiprocessor bit set to 1. Next the transmitting processor sends transmit data with the multiprocessor bit cleared to 0.

Receiving processors skip incoming data until they receive data with the multiprocessor bit set to 1. When they receive data with the multiprocessor bit set to 1, receiving processors compare the data with their IDs. The receiving processor with a matching ID continues to receive further incoming data. Processors with IDs not matching the received data skip further incoming data until they again receive data with the multiprocessor bit set to 1. Multiple processors can send and receive data in this way.

Figure 17.12 shows an example of communication among processors using the multiprocessor format.

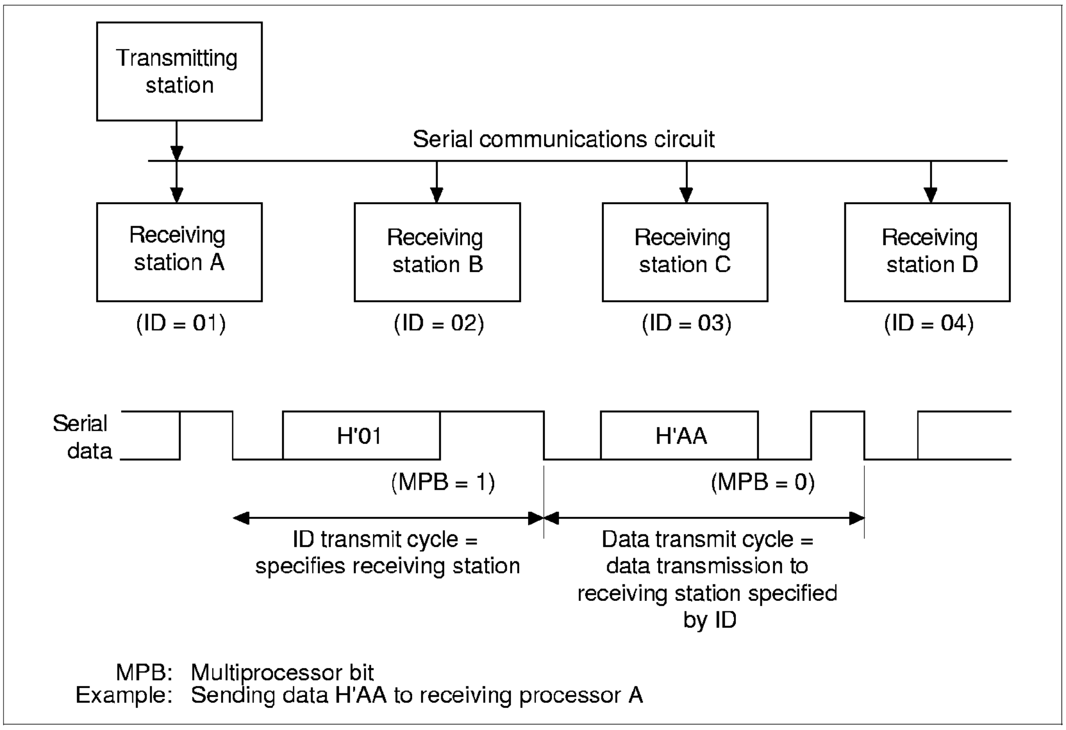

Transmitting station
Serial communications circuit
Receiving station A
Receiving station B
Receiving station C
Receiving station D
(ID = 01)
(ID = 02)
(ID = 03)
(ID = 04)
Serial data
H'01
H'AA
(MPB = 1)
(MPB = 0)
ID transmit cycle = specifies receiving station
Data transmit cycle = data transmission to receiving station specified by ID
MPB: Multiprocessor bit
Example: Sending data H'AA to receiving processor A

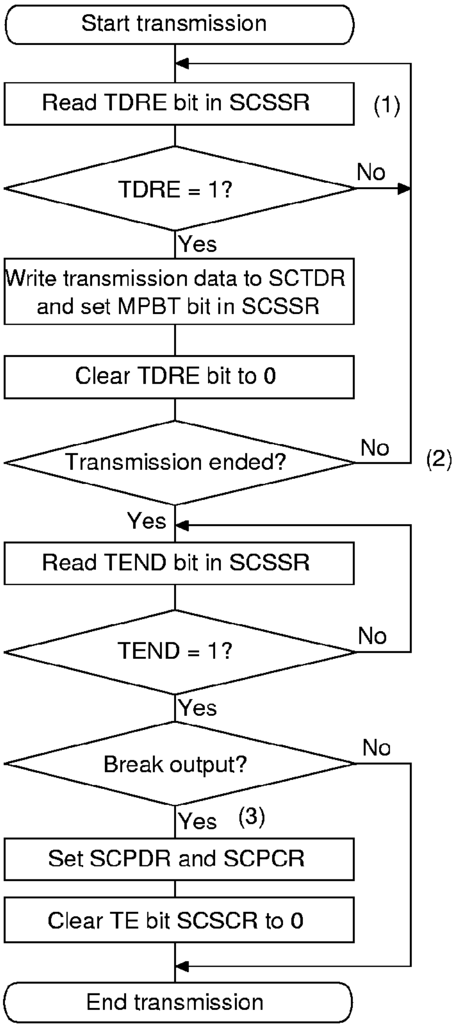
Figure 17.12 Communication Among Processors Using Multiprocessor Format

Communication Formats: Four formats are available. Parity-bit settings are ignored when the multiprocessor format is selected. For details see table 17.11.

Clock: See the description in the asynchronous mode section.

Transmitting Multiprocessor Serial Data: Figure 17.13 shows a sample flowchart for transmitting multiprocessor serial data. The procedure for transmitting multiprocessor serial data is:

1. SCI status check and transmit data write: Read the serial status register (SCSSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (SCTDR). Also set MPBT (multiprocessor bit transfer) to 0 or 1 in SCSSR. Finally, clear TDRE to 0.
2. To continue transmitting serial data: Read the TDRE bit to check whether it is safe to write (if it reads 1); if so, write data in SCTDR, then clear TDRE to 0.
3. To output a break at the end of serial transmission: Set the port SC data register (SCPDR) and port SC control register (SCPCR), then clear the TE bit to 0 in the serial control register (SCSCR). For SCPCR and SCPDR settings, see section 17.2.8.



Note: Circled numbers refer to the preceding procedure.

Figure 17.13 Sample Flowchart for Transmitting Multiprocessor Serial Data

In transmitting serial data, the SCI operates as follows:

1. The SCI monitors the TDRE bit in the SCSSR. When TDRE is cleared to 0 the SCI recognizes that the transmit data register (SCTDR) contains new data, and loads this data from the SCTDR into the transmit shift register (SCTSR).
2. After loading the data from the SCTDR into the SCTSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) in the SCSCR is set to 1, the SCI requests a transmit-data-empty interrupt (TXI) at this time. Serial transmit data is transmitted in the following order from the TxD pin:
   a. Start bit: One 0 bit is output.
   b. Transmit data: Seven or eight bits are output, LSB first.
   c. Multiprocessor bit: One multiprocessor bit (MPBT value) is output.
   d. Stop bit: One or two 1 bits (stop bits) are output.
   e. Marking: Output of 1 bits continues until the start bit of the next transmit data.
3. The SCI checks the TDRE bit when it outputs the stop bit. If TDRE is 0, the SCI loads data from the SCTDR into the SCTSR, outputs the stop bit, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit in the SCSSR to 1, outputs the stop bit, then continues output of 1 bits in the marking state. If the transmit-end interrupt enable bit (TEIE) in the SCSCR is set to 1, a transmit-end interrupt (TEI) is requested at this time.

Figure 17.14 shows SCI transmission in the multiprocessor format.

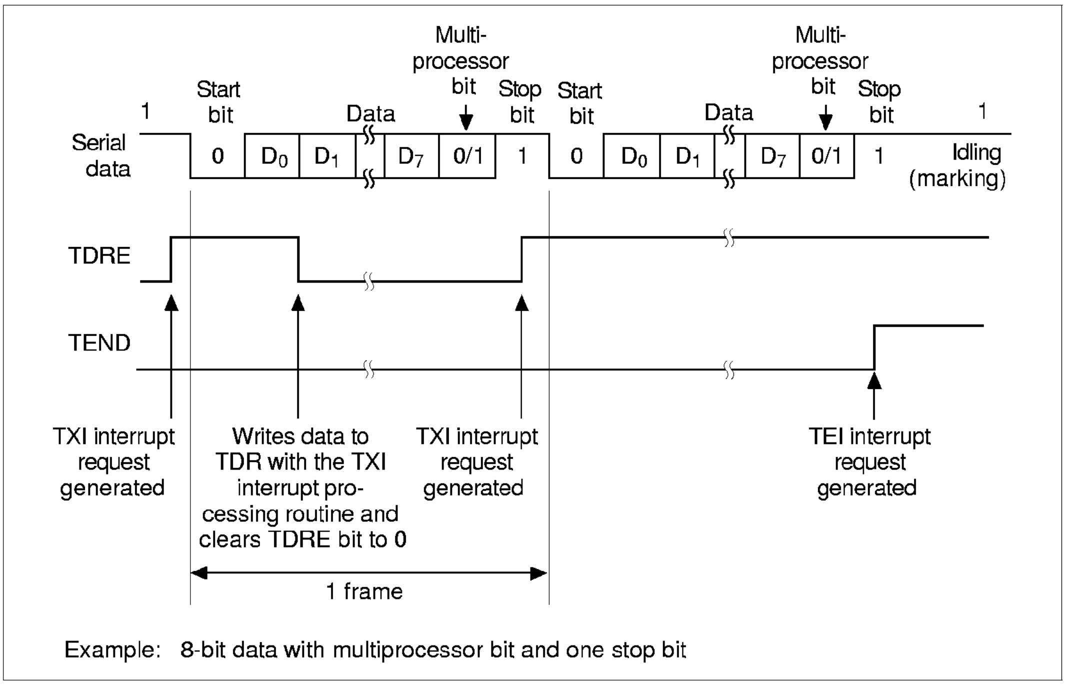

Multi-processor bit
Multi-processor bit
Start bit
Data
Stop bit
Start bit
Data
Stop bit
1
1
Serial data
0
D0
D1
D7
0/1
1
0
D0
D1
D7
0/1
1
Idling (marking)
TDRE
TEND
TXI interrupt request generated
Writes data to TDR with the TXI interrupt processing routine and clears TDRE bit to 0
TXI interrupt request generated
TEI interrupt request generated
1 frame
Example: 8-bit data with multiprocessor bit and one stop bit

Figure 17.14 SCI Multiprocessor Transmit Operation

Receiving Multiprocessor Serial Data: Figure 17.15 shows a sample flowchart for receiving multiprocessor serial data. The procedure for receiving multiprocessor serial data is:

1. ID receive cycle: Set the MPIE bit in the serial control register (SCSCR) to 1.
2. SCI status check and compare to ID reception: Read the serial status register (SCSSR), check that RDRF is set to 1, then read data from the receive data register (SCRDR) and compare with the processor's own ID. If the ID does not match the receive data, set MPIE to 1 again and clear RDRF to 0. If the ID matches the receive data, clear RDRF to 0.
3. SCI status check and data receiving: Read SCSSR, check that RDRF is set to 1, then read data from the receive data register (SCRDR).
4. Receive error processing and break detection: If a receive error occurs, read the ORER and FER bits in SCSSR to identify the error. After executing the necessary error processing, clear both ORER and FER to 0. Receiving cannot resume if ORER or FER remain set to 1. When a framing error occurs, the RxD pin can be read to detect the break state.

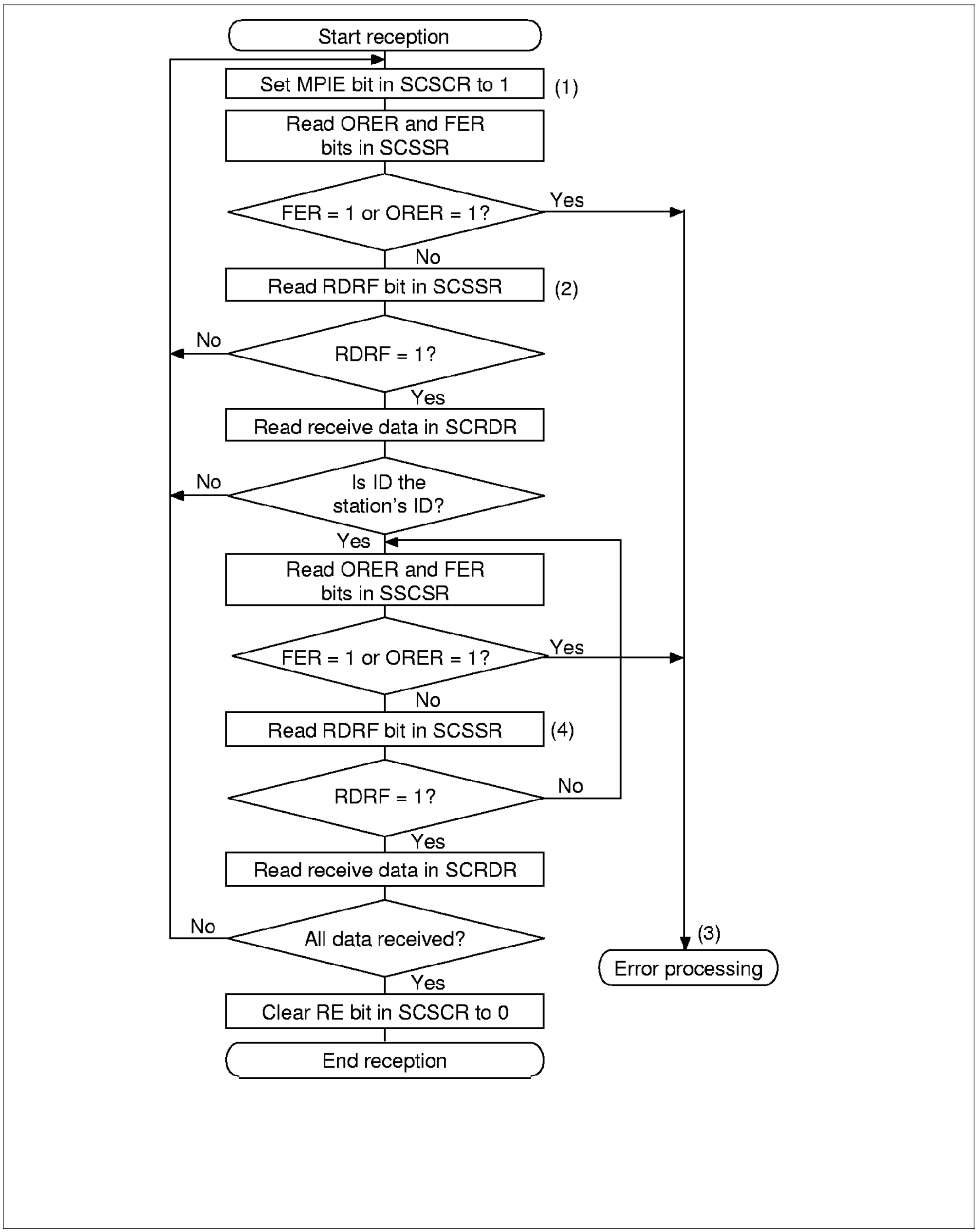


Figure 17.15 Sample Flowchart for Receiving Multiprocessor Serial Data

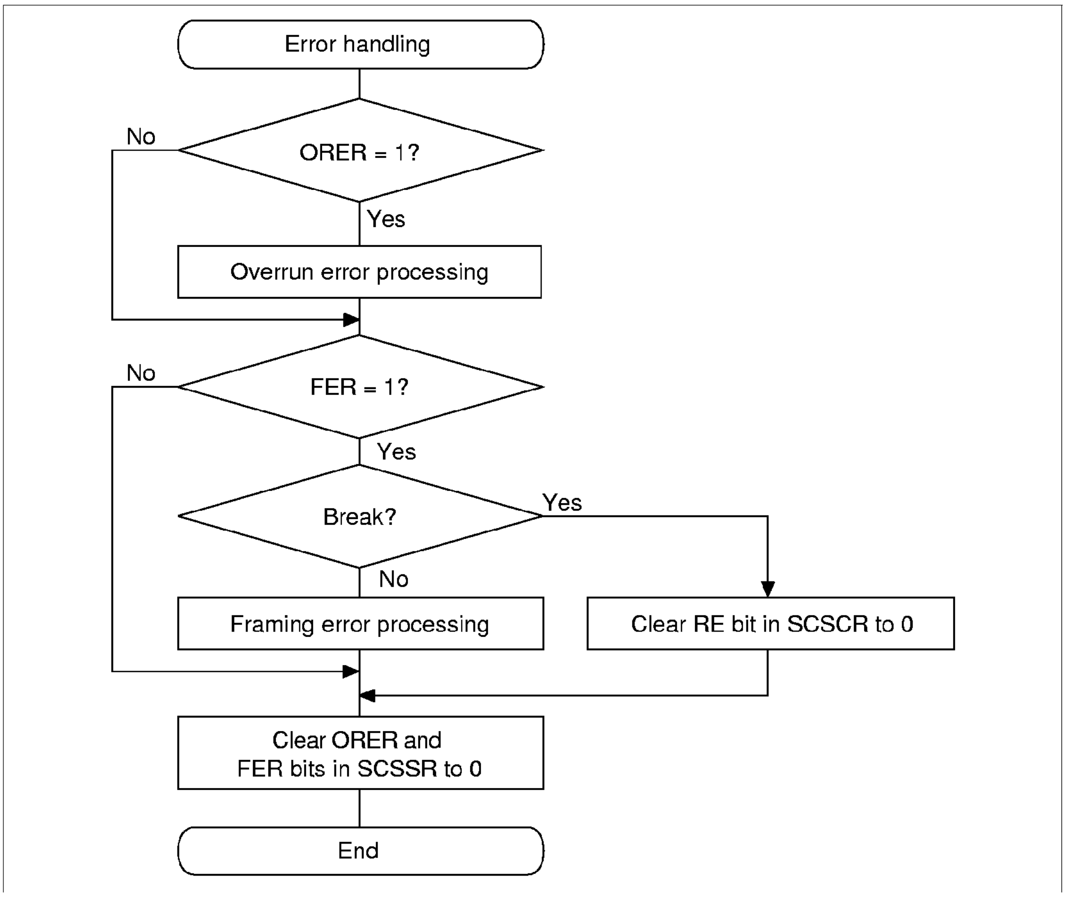


Figure 17.15 Sample Flowchart for Receiving Multiprocessor Serial Data (cont)

Figure 17.16 shows an example of SCI receive operation using a multiprocessor format.

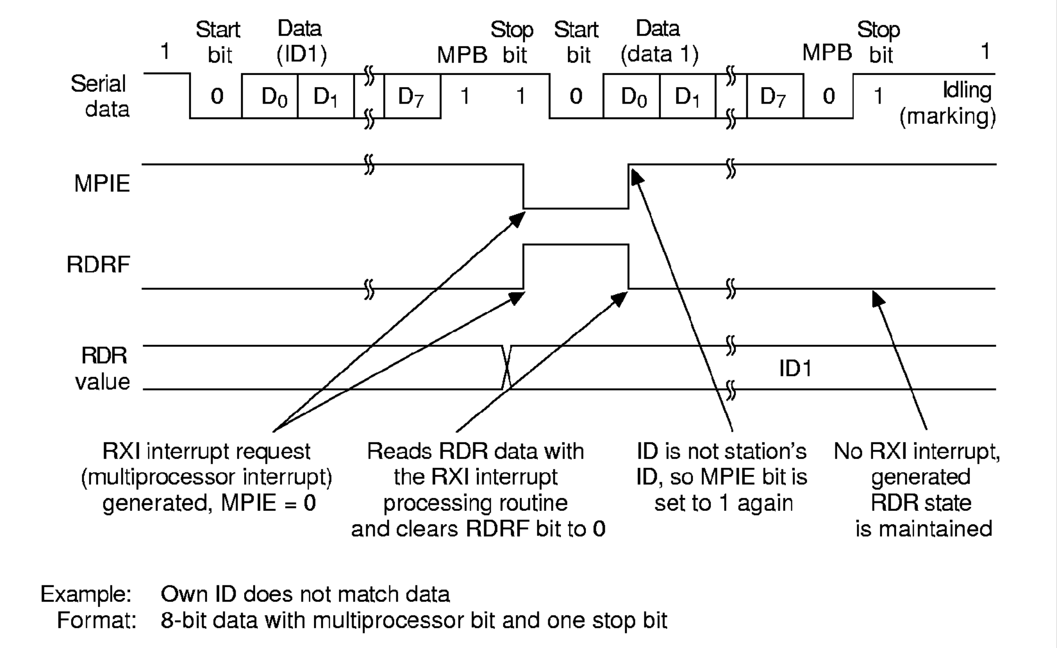
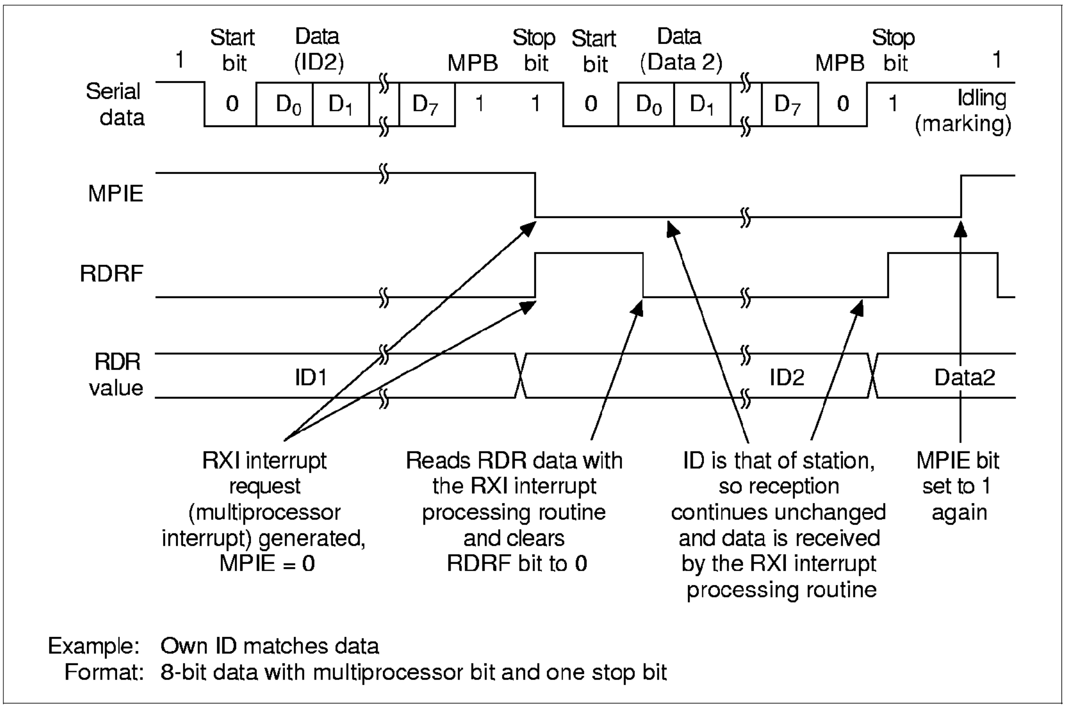

Start bit
Data (ID1)
MPB
Stop bit
Start bit
Data (data 1)
MPB
Stop bit
1
Serial data
0
D_0
D_1
D_7
1
1
0
D_0
D_1
D_7
0
1
1
Idling (marking)
MPIE
RDRF
RDR value
ID1
RXI interrupt request (multiprocessor interrupt) generated, MPIE = 0
Reads RDR data with the RXI interrupt processing routine and clears RDRF bit to 0
ID is not station's ID, so MPIE bit is set to 1 again
No RXI interrupt, generated RDR state is maintained
Example: Own ID does not match data
Format: 8-bit data with multiprocessor bit and one stop bit

Figure 17.16 Example of SCI Receive Operation



Start bit
Data (ID2)
MPB
Stop bit
Start bit
Data (Data 2)
MPB
Stop bit
1
1
Serial data
0
D_0
D_1
D_7
1
1
0
D_0
D_1
D_7
0
1
Idling (marking)
MPIE
RDRF
RDR value
ID1
ID2
Data2
RXI interrupt request (multiprocessor interrupt) generated, MPIE = 0
Reads RDR data with the RXI interrupt processing routine and clears RDRF bit to 0
ID is that of station, so reception continues unchanged and data is received by the RXI interrupt processing routine
MPIE bit set to 1 again
Example: Own ID matches data
Format: 8-bit data with multiprocessor bit and one stop bit

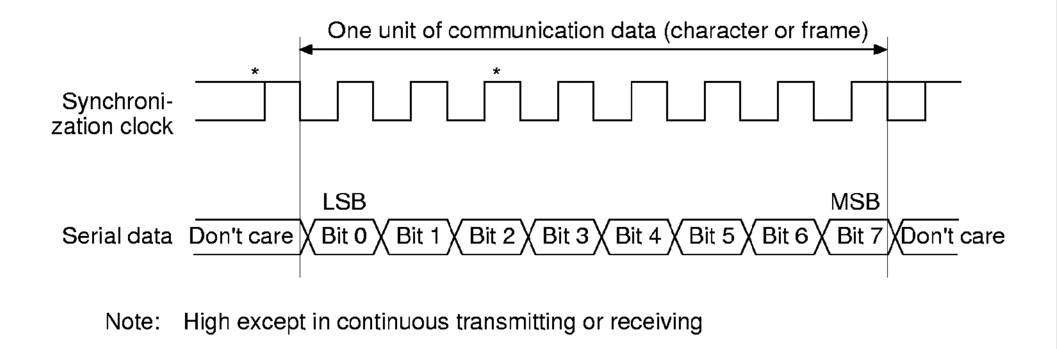
Figure 17.16 Example of SCI Receive Operation (cont)

17.3.4 Clock Synchronous Operation

In the clock synchronous mode, the SCI transmits and receives data in synchronization with clock pulses. This mode is suitable for high-speed serial communication.

The SCI transmitter and receiver are independent, so full-duplex communication is possible while sharing the same clock. The transmitter and receiver are also double buffered, so continuous transmitting or receiving is possible by reading or writing data while transmitting or receiving is in progress.

Figure 17.17 shows the general format in clock synchronous serial communication.



Note: High except in continuous transmitting or receiving

Figure 17.17 Data Format in Clock Synchronous Communication

In clock synchronous serial communication, each data bit is output on the communication line from one falling edge of the serial clock to the next. Data are guaranteed valid at the rising edge of the serial clock. In each character, the serial data bits are transmitted in order from the LSB (first) to the MSB (last). After output of the MSB, the communication line remains in the state of the MSB. In the clock synchronous mode, the SCI transmits or receives data by synchronizing with the falling edge of the serial clock.

Communication Format: The data length is fixed at eight bits. No parity bit or multiprocessor bit can be added.

Clock: An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected as the SCI transmit/receive clock. The clock source is selected by the C/$\overline{\text{A}}$ bit in the serial mode register (SCSMR) and bits CKE1 and CKE0 in the serial control register (SCSCR). See table 17.10.

When the SCI operates on an internal clock, it outputs the clock signal at the SCK pin. Eight clock pulses are output per transmitted or received character. When the SCI is not transmitting or receiving, the clock signal remains in the high state. When only receiving, the SCI receives in 2-character units, so a 16 pulse synchronization clock is output. To receive in 1-character units, select an external clock source.

Transmitting and Receiving Data: SCI Initialization (clock synchronous mode). Before transmitting, receiving, or changing the mode or communication format, the software must clear the TE and RE bits to 0 in the serial control register (SCSCR), then initialize the SCI. Clearing TE to 0 sets TDRE to 1 and initializes the transmit shift register (SCTSR). Clearing RE to 0, however, does not initialize the RDRF, PER, FER, and ORER flags and receive data register (SCRDR), which retain their previous contents.

Figure 17.18 is a sample flowchart for initializing the SCI. The procedure for initializing the SCI is:

1. Select the clock source in the serial control register (SCSCR). Leave RIE, TIE, TEIE, MPIE, TE and RE cleared to 0.
2. Select the communication format in the serial mode register (SCSMR).
3. Write the value corresponding to the bit rate in the bit rate register (SCBRR) unless an external clock is used.
4. Wait for at least the interval required to transmit or receive one bit, then set TE or RE in the serial control register (SCSCR) to 1. Also set RIE, TIE, TEIE and MPIE. Setting TE and RE allows use of the TxD and RxD pins.

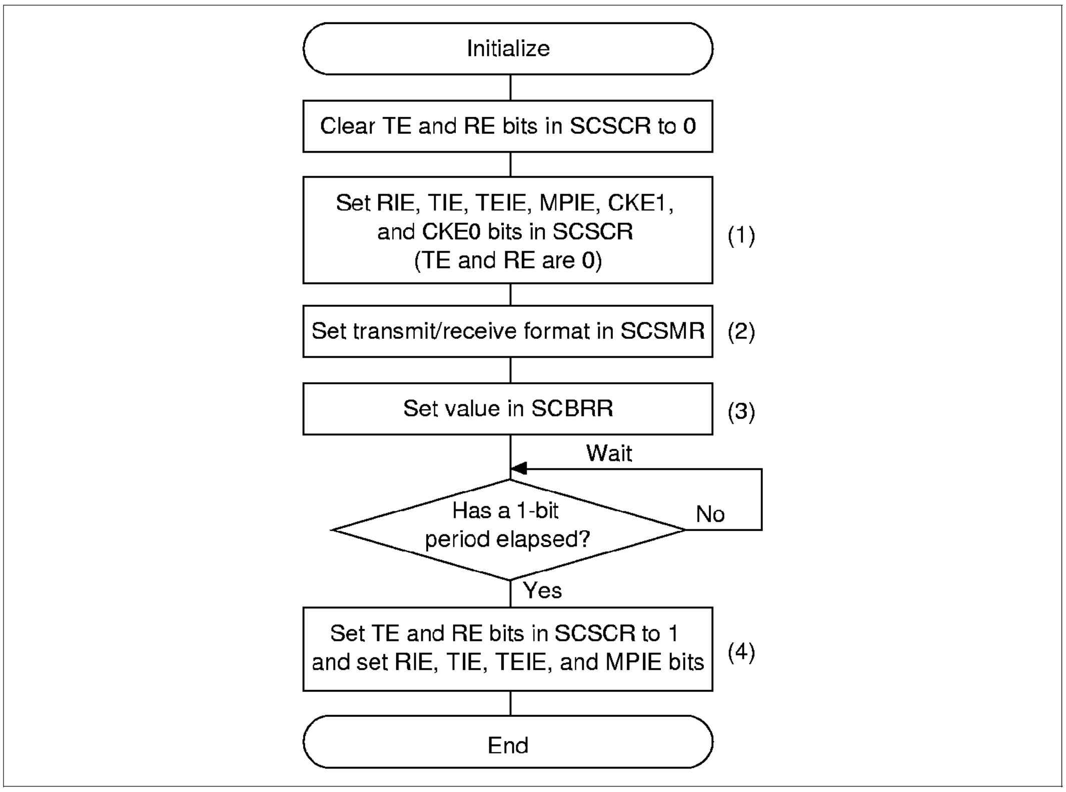

Initialize
Clear TE and RE bits in SCSCR to 0
Set RIE, TIE, TEIE, MPIE, CKE1, and CKE0 bits in SCSCR (TE and RE are 0)
(1)
Set transmit/receive format in SCSMR
(2)
Set value in SCBRR
(3)
Wait
Has a 1-bit period elapsed?
No
Yes
Set TE and RE bits in SCSCR to 1 and set RIE, TIE, TEIE, and MPIE bits
(4)
End

Figure 17.18 Sample Flowchart for SCI Initialization

Transmitting Serial Data (Clock Synchronous Mode): Figure 17.19 shows a sample flowchart for transmitting serial data. The procedure for transmitting serial data is:

1. SCI status check and transmit data write: Read the serial status register (SCSSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (SCTDR) and clear TDRE to 0.
2. To continue transmitting serial data: Read the TDRE bit to check whether it is safe to write (if it reads 1); if so, write data in SCTDR, then clear TDRE to 0.

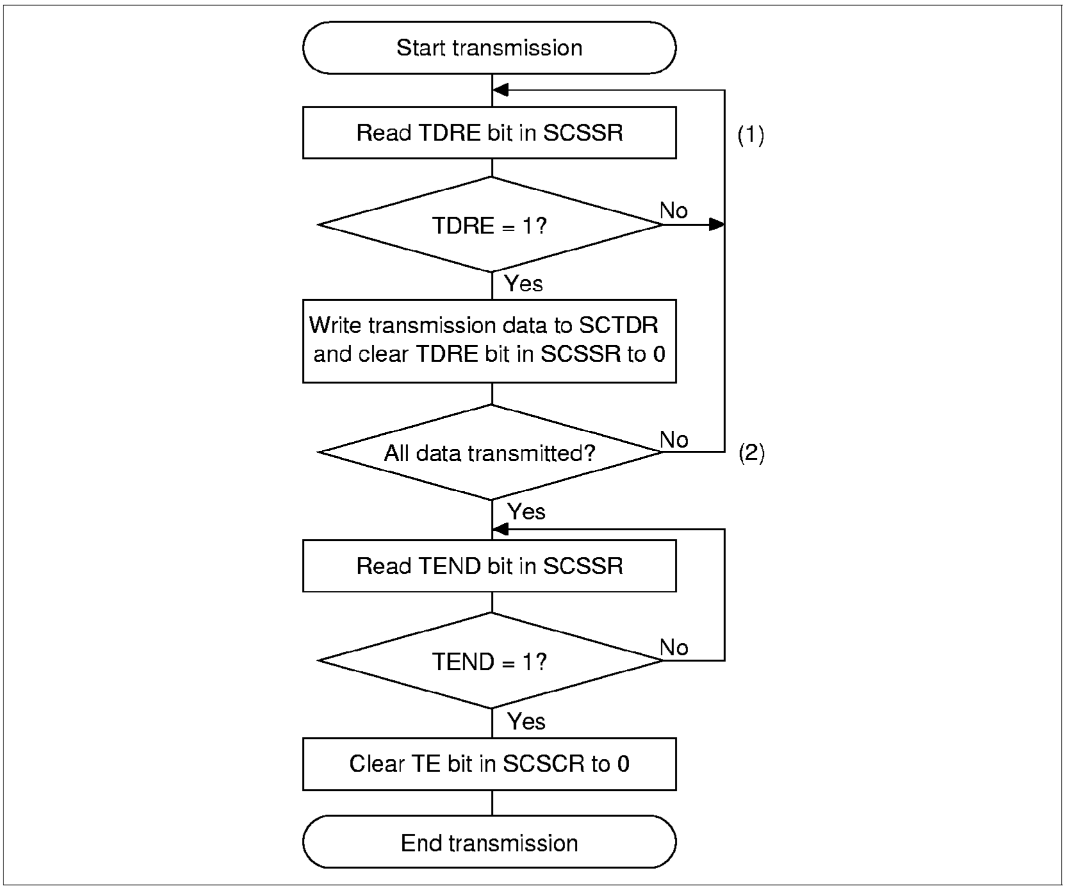


Figure 17.19 Sample Flowchart for Serial Transmitting

In transmitting serial data, the SCI operates as follows:

1. The SCI monitors the TDRE bit in the SCSSR. When TDRE is cleared to 0 the SCI recognizes that the transmit data register (SCTDR) contains new data and loads this data from the SCTDR into the transmit shift register (SCTSR).
2. After loading the data from the SCTDR into the SCTSR, the SCI sets the TDRE bit to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE) in the SCSCR is set to 1, the SCI requests a transmit-data-empty interrupt (TXI) at this time.

   If clock output mode is selected, the SCI outputs eight synchronous clock pulses. If an external clock source is selected, the SCI outputs data in synchronization with the input clock. Data are output from the TxD pin in order from the LSB (bit 0) to the MSB (bit 7).
3. The SCI checks the TDRE bit when it outputs the MSB (bit 7). If TDRE is 0, the SCI loads data from the SCTDR into the SCTSR, then begins serial transmission of the next frame. If TDRE is 1, the SCI sets the TEND bit in the SCSSR to 1, transmits the MSB, then holds the transmit data pin (TxD) in the MSB state. If the transmit-end interrupt enable bit (TEIE) in the SCSCR is set to 1, a transmit-end interrupt (TEI) is requested at this time.
4. After the end of serial transmission, the SCK pin is held in the high state.

Figure 17.20 shows an example of SCI transmit operation.

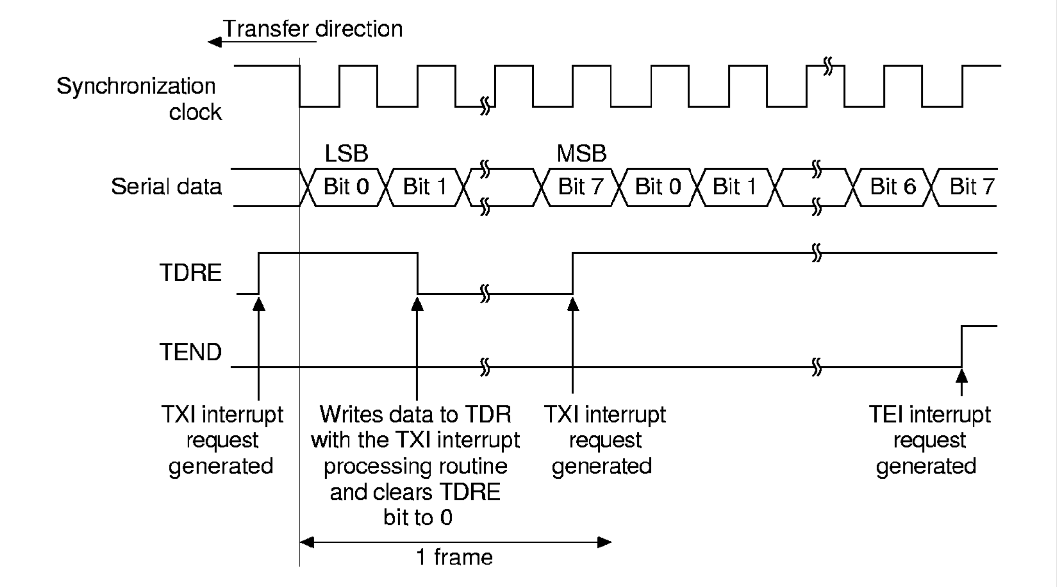

Transfer direction
Synchronization clock
LSB
MSB
Serial data
Bit 0
Bit 1
Bit 7
Bit 0
Bit 1
Bit 6
Bit 7
TDRE
TEND
TXI interrupt request generated
Writes data to TDR with the TXI interrupt processing routine and clears TDRE bit to 0
TXI interrupt request generated
TEI interrupt request generated
1 frame

Figure 17.20 Example of SCI Transmit Operation

Receiving Serial Data (Clock Synchronous Mode): Figure 17.21 shows a sample flowchart for receiving serial data. When switching from the asynchronous mode to the clock synchronous mode, make sure that ORER, PER, and FER are cleared to 0. If PER or FER is set to 1, the RDRF bit will not be set and both transmitting and receiving will be disabled.

The procedure for receiving serial data is:

1. Receive error processing: If a receive error occurs, read the ORER bit in SCSSR to identify the error. After executing the necessary error processing, clear ORER to 0. Transmitting/receiving cannot resume if ORER remains set to 1.
2. SCI status check and receive data read: Read the serial status register (SCSSR), check that RDRF is set to 1, then read receive data from the receive data register (SCRDR) and clear RDRF to 0. The RXI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
3. To continuc rcceiving scrial data: Rcad SCRDR, and clcar RDRF to 0 bcforc thc framc MSB (bit 7) of the current frame is received.

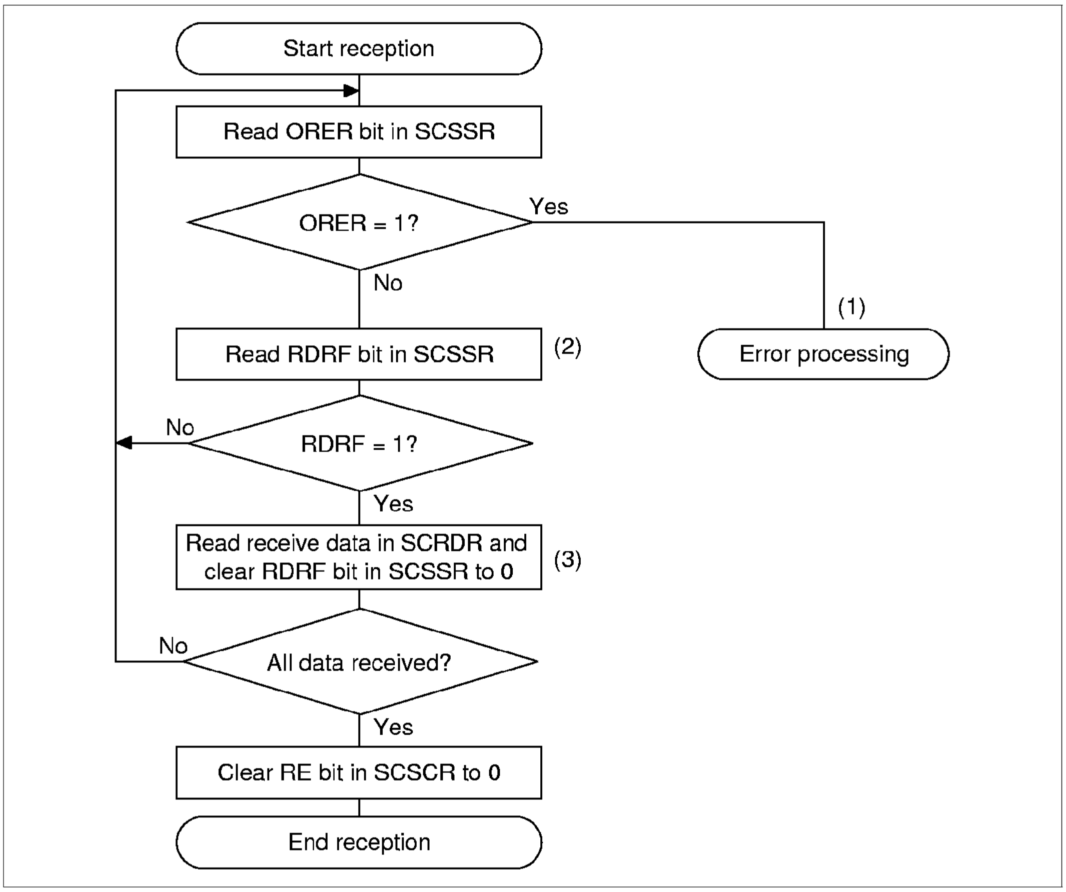


Figure 17.21 Sample Flowchart for Serial Data Receiving

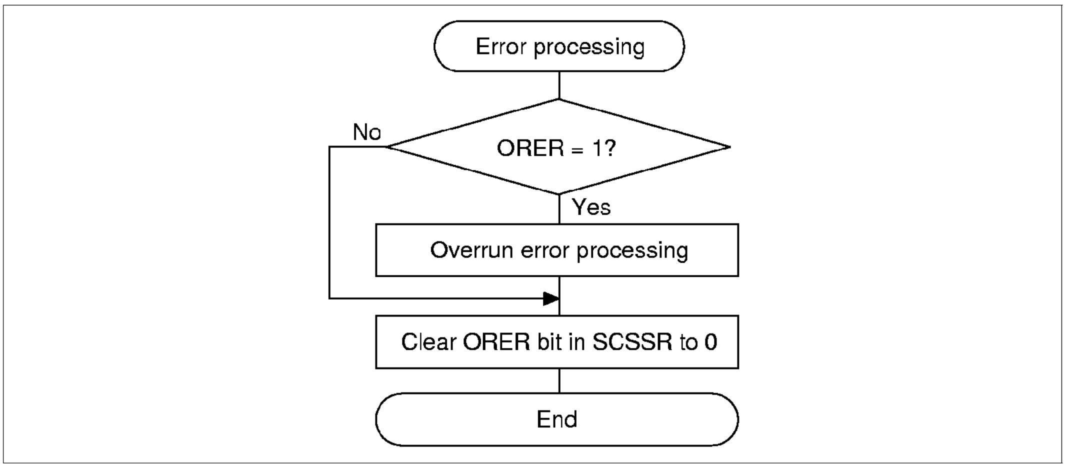


Figure 17.21 Sample Flowchart for Serial Data Receiving (cont)

In receiving, the SCI operates as follows:

1. The SCI synchronizes with serial clock input or output and initializes internally.
2. Receive data is shifted into the SCRSR in order from the LSB to the MSB. After receiving the data, the SCI checks that RDRF is 0 so that receive data can be loaded from the SCRSR into the SCRDR. If this check is passed, the SCI sets RDRF to 1 and stores the received data in the SCRDR. If the check is not passed (receive error), the SCI operates as indicated in table 17.12. This state prevents further transmission or reception. While receiving, the RDRF bit is not set to 1. Be sure to clear the error flag.
3. After setting RDRF to 1, if the receive-data-full interrupt enable bit (RIE) is set to 1 in the SCSCR, the SCI requests a receive-data-full interrupt (RXI). If the ORER bit is set to 1 and the receive-data-full interrupt enable bit (RIE) in the SCSCR is also set to 1, the SCI requests a receive-error interrupt (ERI).

Figure 17.22 shows an example of the SCI receive operation.

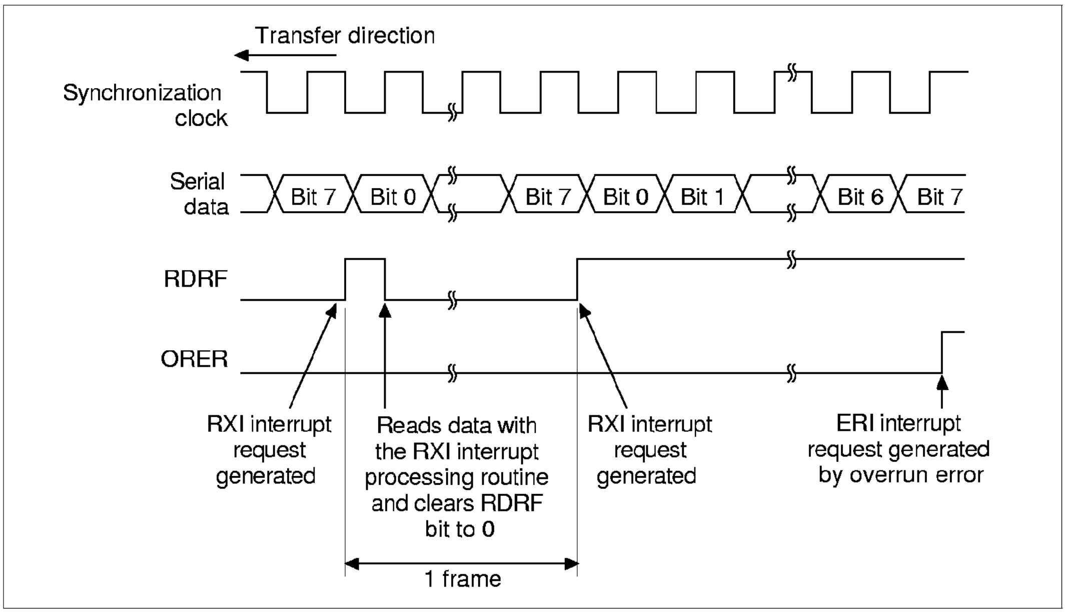
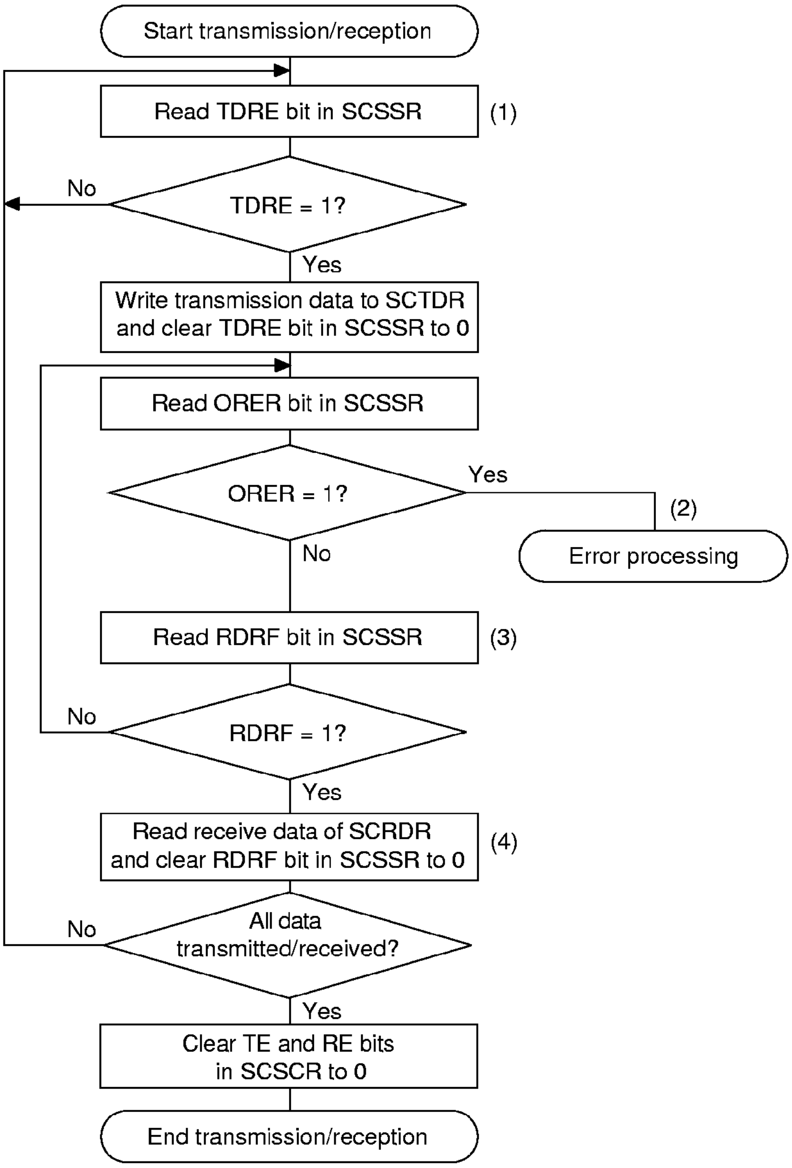

Transfer direction
Synchronization clock
Serial data
Bit 7
Bit 0
Bit 7
Bit 0
Bit 1
Bit 6
Bit 7
RDRF
ORER
RXI interrupt request generated
Reads data with the RXI interrupt processing routine and clears RDRF bit to 0
RXI interrupt request generated
ERI interrupt request generated by overrun error
1 frame

Figure 17.22 Example of SCI Receive Operation

Transmitting and Receiving Serial Data Simultaneously (Clock Synchronous Mode): Figure 17.23 shows a sample flowchart for transmitting and receiving serial data simultaneously. The procedure for setting the SCI to transmit and receive serial data simultaneously is:

1. SCI status check and transmit data write: Read the serial status register (SCSSR), check that the TDRE bit is 1, then write transmit data in the transmit data register (SCTDR) and clear TDRE to 0. The TXI interrupt can also be used to determine if the TDRE bit has changed from 0 to 1.
2. Receive error processing: If a receive error occurs, read the ORER bit in SCSSR to identify the error. After executing the necessary error processing, clear ORER to 0. Transmitting/receiving cannot resume if ORER remains set to 1.
3. SCI status check and receive data read: Read the serial status register (SCSSR), check that RDRF is set to 1, then read receive data from the receive data register (SCRDR) and clear RDRF to 0. The RXI interrupt can also be used to determine if the RDRF bit has changed from 0 to 1.
4. To continue transmitting and receiving serial data: Read the RDRF bit and SCRDR, and clear RDRF to 0 before the frame MSB (bit 7) of the current frame is received. Also read the TDRE bit to check whether it is safe to write (if it reads 1); if so, write data in SCTDR, then clear TDRE to 0 before the MSB (bit 7) of the current frame is transmitted.


Start transmission/reception
Read TDRE bit in SCSSR
(1)
No
TDRE = 1?
Yes
Write transmission data to SCTDR and clear TDRE bit in SCSSR to 0
Read ORER bit in SCSSR
ORER = 1?
Yes
(2)
Error processing
No
Read RDRF bit in SCSSR
(3)
No
RDRF = 1?
Yes
Read receive data of SCRDR and clear RDRF bit in SCSSR to 0
(4)
No
All data transmitted/received?
Yes
Clear TE and RE bits in SCSCR to 0
End transmission/reception

Note: In switching from transmitting or receiving to simultaneous transmitting and receiving, clear both TE and RE to 0, then set both TE and RE to 1.

Figure 17.23 Sample Flowchart for Serial Data Transmitting/Receiving

17.4 SCI Interrupt Sources

The SCI has four interrupt sources in each channel: Transmit-end (TEI), receive-error (ERI), receive-data-full (RXI), and transmit-data-empty (TXI). Table 17.13 lists the interrupt sources and indicates their priority. These interrupts can be enabled and disabled by the TIE, RIE, and TEIE bits in the serial control register (SCSCR). Each interrupt request is sent separately to the interrupt controller.

TXI is requested when the TDRE bit in the SCSSR is set to 1. TDRE is automatically cleared to 0 when data is written in the transmit data register (SCTDR).

RXI is requested when the RDRF bit in the SCSSR is set to 1. RDRF is automatically cleared to 0 when the receive data register (SCRDR) is read.

ERI is requested when the ORER, PER, or FER bit in the SCSSR is set to 1.

TEI is requested when the TEND bit in the SCSSR is set to 1. Where the TXI interrupt indicates that transmit data writing is enabled, the TEI interrupt indicates that the transmit operation is complete.

Table 17.13 SCI Interrupt Sources

Interrupt Source	Description	Priority When Reset Is Cleared
ERI	Receive error (ORER, PER, or FER)	High
RXI	Receive data full (RDRF)	
TXI	Transmit data empty (TDRE)	↓
TEI	Transmit end (TEND)	Low

See section 4, Exception Processing, for information on the priority order and relationship to non-SCI interrupts.

17.5 Notes on Use

Note the following points when using the SCI.

SCTDR Write and TDRE Flags: The TDRE bit in the serial status register (SCSSR) is a status flag indicating loading of transmit data from the SCTDR into the SCTSR. The SCI sets TDRE to 1 when it transfers data from the SCTDR to the SCTSR. Data can be written to the SCTDR regardless of the TDRE bit state. If new data is written in the SCTDR when TDRE is 0, however, the old data stored in the SCTDR will be lost because the data has not yet been transferred to the SCTSR. Before writing transmit data to the SCTDR, be sure to check that TDRE is set to 1.

Simultaneous Multiple Receive Errors: Table 17.14 indicates the state of the SCSSR status flags when multiple receive errors occur simultaneously. When an overrun error occurs, the SCRSR contents cannot be transferred to the SCRDR, so receive data is lost.

Table 17.14 SCSSR Status Flags and Transfer of Receive Data

	SCSSR Status Flags				Receive Data Transfer
Receive Error Status	**RDRF**	**ORER**	**FER**	**PER**	**SCRSR → SCRDR**
Overrun error	1	1	0	0	X*1
Framing error	0	0	1	0	O*2
Parity error	0	0	0	1	O
Overrun error + framing error	1	1	1	0	X
Overrun error + parity error	1	1	0	1	X
Framing error + parity error	0	0	1	1	O
Overrun error + framing error + parity error	1	1	1	1	X

Notes: 1. Receive data is not transferred from SCRSR to SCRDR.
2. Receive data is transferred from SCRSR to SCRDR.

Break Detection and Processing: Break signals can be detected by reading the RxD pin directly when a framing error (FER) is detected. In the break state, the input from the RxD pin consists of all 0s, so FER is set and the parity error flag (PER) may also be set. In the break state, the SCI receiver continues to operate, so if the FER bit is cleared to 0, it will be set to 1 again.

Sending a Break Signal: The TxD pin I/O condition and level can be determined by means of the SCP0DT bit of the port SC data register (SCPDR) and bits SCP0MD0 and SCP0MD1 of the port SC control register (SCPCR). These bits can be used to send breaks. To send a break during serial transmission, clear the SCP0DT bit to 0 (designating low level), then clear the TE bit to 0 (halting transmission). When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, and 0 is output from the TxD pin.

TEND Flag and TE Bit Processing: The TEND flag is set to 1 during transmission of the stop bit of the last data. Consequently, if the TE bit is cleared to 0 immediately after setting of the TEND flag has been confirmed, the stop bit will be in the process of transmission and will not be transmitted normally. Therefore, the TE bit should not be cleared to 0 for at least 0.5 serial clock cycles (or 1.5 cycles if two stop bits are used) after setting of the TEND flag setting is confirmed.

Receive Error Flags and Transmitter Operation (Clock Synchronous Mode Only): When a receive error flag (ORER, PER, or FER) is set to 1, the SCI will not start transmitting even if TDRE is set to 1. Be sure to clear the receive error flags to 0 before starting to transmit. Note that clearing RE to 0 does not clear the receive error flags.

Receive Data Sampling Timing and Receive Margin in the Asynchronous Mode: In the asynchronous mode, the SCI operates on a base clock of 16 times the transfer rate frequency. In receiving, the SCI synchronizes internally with the falling edge of the start bit, which it samples on the base clock. Receive data is latched on the rising edge of the eighth base clock pulse (figure 17.24).

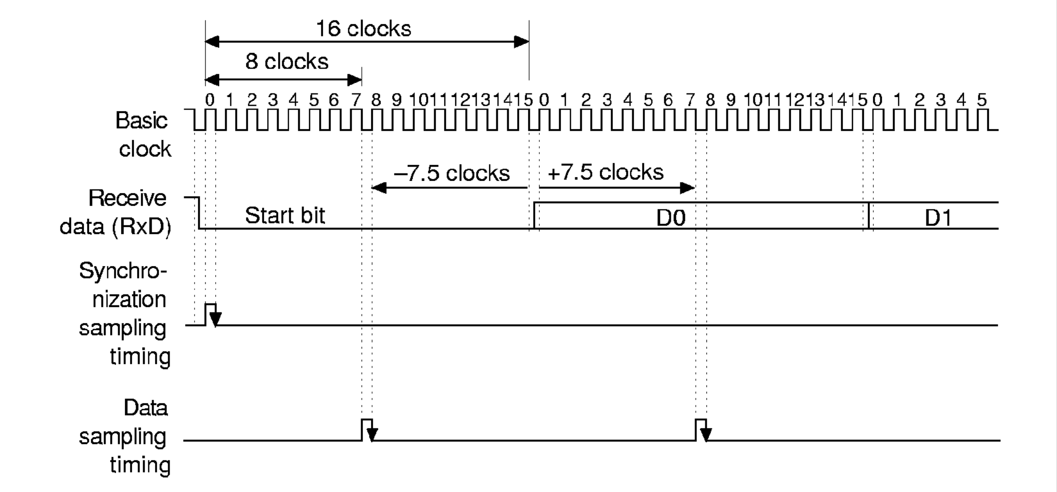


Figure 17.24 Receive Data Sampling Timing in Asynchronous Mode

The receive margin in the asynchronous mode can therefore be expressed as in equation 1.

Equation 1:

$$M = \left| \left(0.5 - \frac{1}{2N}\right) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100\%$$

Where:

M = Receive margin (%)
N = Ratio of clock frequency to bit rate (N = 16)
D = Clock duty cycle (D = 0–1.0)
L = Frame length (L = 9–12)
F = Absolute deviation of clock frequency

From equation (1), if F = 0 and D = 0.5, the receive margin is 46.875%, as in equation 2.

Equation 2:

$$\begin{aligned} M &= (0.5 - 1/(2 \times 16)) \times 100\% \\ &= 46.875\% \end{aligned}$$

This is a theoretical value. A reasonable margin to allow in system designs is 20–30%.

Cautions for Clock Synchronous External Clock Mode:

- Set TE = RE = 1 only when the external clock SCK is 1.
- Do not set TE = RE = 1 until at least four clocks after the external clock SCK has changed from 0 to 1.
- When receiving, RDRF is 1 when RE is set to zero 2.5–3.5 clocks after the rising edge of the SCK input of the D7 bit in RxD, but it cannot be copied to SCRDR.

Caution for Clock Synchronous Internal Clock Mode: When receiving, RDRF is 1 when RE is set to zero 1.5 clocks after the rising edge of the SCK output of the D7 bit in RxD, but it cannot be copied to SCRDR.

Section 18 Smart Card Interface

18.1 Overview

As an added serial communications interface function, the SCI supports an IC card (smart card) interface that conforms to the ISO/IEC standard 7816-3 for identification of cards. Register settings are used to switch between the ordinary serial communication interface and the smart card interface.

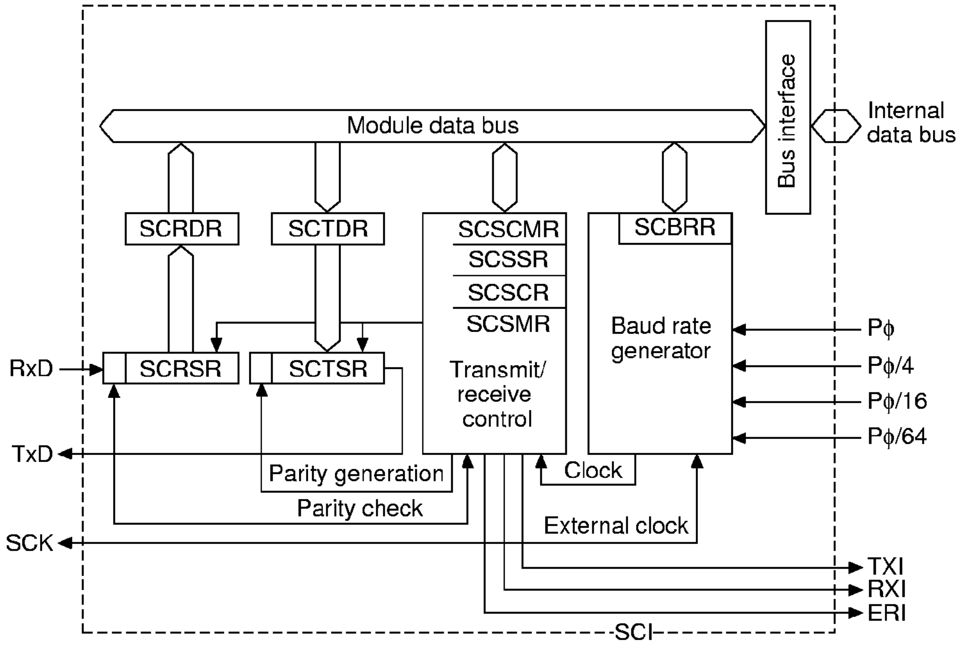
18.1.1 Features

The smart card interface has the following features:

- Asynchronous mode
  - Data length: Eight bits
  - Parity bit generation and check
  - Receive mode error signal detection (parity error)
  - Transmit mode error signal detection and automatic re-transmission of data
  - Supports both direct convention and inverse convention
- Bit rate can be selected using on-chip baud rate generator.
- Three types of interrupts: Transmit-data-empty, receive-data-full, and communication-error interrupts are requested independently.

18.1.2 Block Diagram

Figure 18.1 shows a block diagram of the smart card interface.



SCSCMR:	Smart card mode register
SCRSR:	Receive shift register
SCRDR:	Receive data register
SCTSR:	Transmit shift register
SCTDR:	Transmit data register
SCSMR:	Serial mode register
SCSCR:	Serial control register
SCSSR:	Serial status register
SCBRR:	Bit rate register

Figure 18.1 Smart Card Interface Block Diagram

18.1.3 Pin Configuration

Table 18.1 summarizes the smart card interface pins.

Table 18.1SCI Pins

Pin Name	Abbreviation	I/O	Function
Serial clock pin	SCK0	Output	Clock output
Receive data pin	RxD0	Input	Receive data input
Transmit data pin	TxD0	Output	Transmit data output

18.1.4 Register Configuration

Table 18.2 summarizes the registers used by the smart card interface. The SCSMR, SCBRR, SCSCR, SCTDR, and SCRDR registers are the same as in the ordinary SCI function. They are described in section 14, Serial Communication Interface.

Table 18.2Registers

Name	Abbreviation	R/W	Initial Value*3	Address	Access Size
Serial mode register	SCSMR	R/W	H'00	H'FFFFFE80	8
Bit rate register	SCBRR	R/W	H'FF	H'FFFFFE82	8
Serial control register	SCSCR	R/W	H'00	H'FFFFFE84	8
Transmit data register	SCTDR	R/W	H'FF	H'FFFFFE86	8
Serial status register	SCSSR	R/(W)*1	H'84	H'FFFFFE88	8
Receive data register	SCRDR	R	H'00	H'FFFFFE8A	8
Smart card mode register	SCSCMR	R/W	H'00*2	H'FFFFFE8C	8

Notes: 1. Only 0 can be written, to clear the flags.

2. Bits 0, 2, and 3 are cleared. The value of the other bits is undefined.

3. Initialized by a power-on or manual reset.

18.2 Register Descriptions

This section describes the registers added for the smart card interface and the bits whose functions are changed.

18.2.1 Smart Card Mode Register (SCSCMR)

The smart card mode register (SCSCMR) is an 8-bit read/write register that selects smart card interface functions. SCSCMR bits 0, 2, and 3 are initialized to H'00 by a reset and in standby mode.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	—	SDIR	SINV	—	SMIF
Initial value:	—	—	—	—	0	0	—	0
R/W:	R	R	R	R	R/W	R/W	R	R/W

Bits 7 to 4 and 1—Reserved: An undefined value will be returned if these bits are read.

Bit 3—Smart Card Data Transfer Direction (SDIR): Selects the serial/parallel conversion format.

Bit 3: SDIR	Description
0	Contents of SCTDR are transferred as LSB first, receive data is stored in SCRDR as LSB first. (Initial value)
1	Contents of SCTDR are transferred as MSB first, receive data is stored in SCRDR as MSB first.

Bit 2—Smart Card Data Inversion (SINV): Specifies whether to invert the logic level of the data. This function is used in combination with bit 3 for transmitting and receiving with an inverse convention card. SINV does not affect the logic level of the parity bit. See section 18.3.4, Register Settings, for information on how parity is set.

Bit 2: SINV	Description
0	Contents of SCTDR are transferred unchanged, receive data is stored in SCRDR unchanged. (Initial value)
1	Contents of SCTDR are inverted before transfer, receive data is inverted before storage in SCRDR.

Bit 0—Smart Card Interface Mode Select (SMIF): Enables the smart card interface function.

Bit 0 : SMIF	Description
0	Smart card interface function disabled (Initial value)
1	Smart card interface function enabled

18.2.2 Serial Status Register (SCSSR)

In the smart card interface mode, the function of SCSSR bit 4 is changed. The setting conditions for bit 2, the TEND bit, are also changed.

Bit:	7	6	5	4	3	2	1	0
Bit name:	TDRE	RDRF	ORER	FER/ERS	PER	TEND	MPB	MPBT
Initial value:	1	0	0	0	0	1	0	0
R/W:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

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

Bits 7 to 5: These bits have the same function as in the ordinary SCI. See section 17, Serial Communication Interface, for more information.

Bit 4—Error Signal Status (ERS): In the smart card interface mode, bit 4 indicates the state of the error signal returned from the receiving side during transmission. The smart card interface cannot detect framing errors.

Bit 4: ERS	Description
0	Receiving ended normally with no error signal. (Initial value) ERS is cleared to 0 when the chip is reset or enters standby mode, or when software reads ERS after it has been set to 1, then writes 0 in ERS.
1	An error signal indicating a parity error was transmitted from the receiving side. ERS is set to 1 if the error signal sampled is low.

Note: The ERS flag maintains its state even when the TE bit in SCSCR is cleared to 0.

Bits 3 to 0: These bits have the same function as in the ordinary SCI. See section 17, Serial Communication Interface, for more information. The setting conditions for bit 2, the transmit end bit (TEND), are changed as follows.

Bit 2: TEND	Description
0	Transmission is in progress. TEND is cleared to 0 when software reads TDRE after it has been set to 1, then writes 0 in TDRE.
1	End of transmission. (Initial value) TEND is set to 1 when: • the chip is reset or enters standby mode, • the TE bit in SCSCR is 0 and the FER/ERS bit is also 0, • the C/$\overline{A}$ bit in SCSMR is 0, and TDRE = 1 and FER/ERS = 0 (normal transmission) 2.5 etu after a one-byte serial character is transmitted, or • the C/$\overline{A}$ bit in SCSMR is 1, and TDRE = 1 and FER/ERS = 0 (normal transmission) 1.0 etu after a one-byte serial character is transmitted.

Note: etu is an abbreviation of elementary time unit, which is the period for the transfer of 1 bit.

18.3 Operation

18.3.1 Overview

The primary functions of the smart card interface are described below.

1. Each frame consists of 8-bit data and 1 parity bit.
2. During transmission, the card leaves a guard time of at least 2 etu (elementary time units: the period for 1 bit to transfer) from the end of the parity bit to the start of the next frame.
3. During reception, the card outputs an error signal low level for 1 etu after 10.5 etu has elapsed from the start bit if a parity error was detected.
4. During transmission, it automatically transmits the same data after allowing at least 2 etu from the time the error signal is sampled.
5. Only start-stop type asynchronous communication functions are supported; no synchronous communication functions are available.

18.3.2 Pin Connections

Figure 18.2 shows the pin connection diagram for the smart card interface. During communication with an IC card, transmission and reception are both carried out over the same data transfer line, so connect the TxD and RxD pins on the chip. Pull up the data transfer line to the power supply V_{CC} side with a resistor.

When using the clock generated by the smart card interface on an IC card, input the SCK pin output to the IC card's CLK pin. This connection is not necessary when the internal clock is used on the IC card.

Use the chip's port output as the reset signal. Apart from these pins, the power and ground pin connections are usually also required.

Note: When the IC card is not connected and both RE and TE are set to 1, closed communication is possible and auto-diagnosis can be performed.

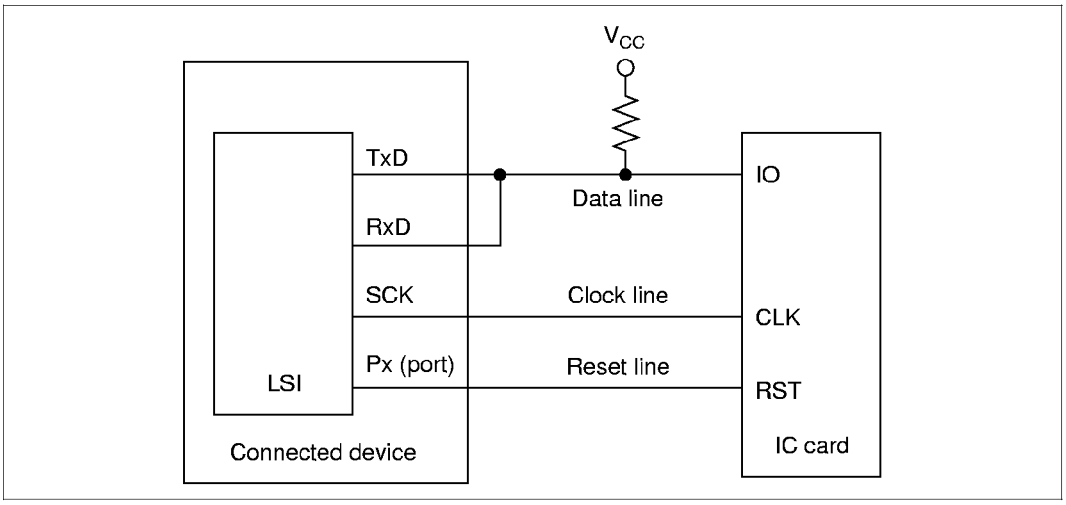


Figure 18.2 Pin Connection Diagram for the Smart Card Interface

18.3.3 Data Format

Figure 18.3 shows the data format for the smart card interface. In this mode, parity is checked every frame while receiving and error signals sent to the transmitting side whenever an error is detected so that data can be re-transmitted. During transmission, error signals are sampled and data re-transmitted whenever an error signal is detected.

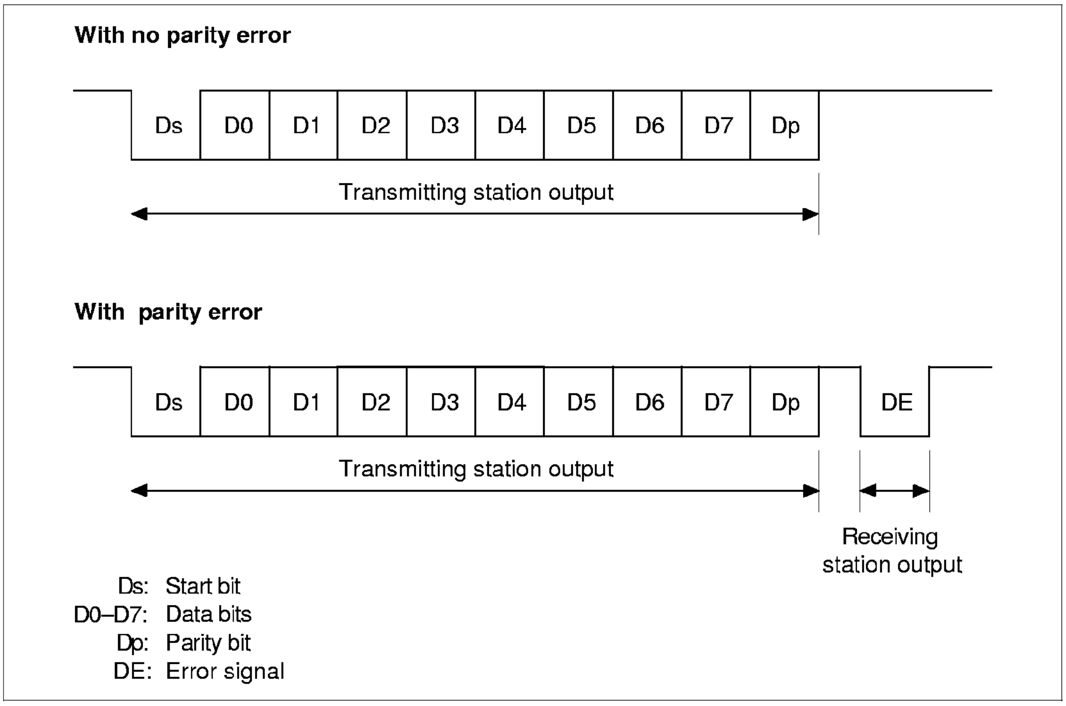


Figure 18.3 Data Format for Smart Card Interface

The operating sequence is:

1. The data line is high impedance when not in use and is fixed high with a pull-up resistor.
2. The transmitting side starts one frame of data transmission. The data frame starts with a start bit (Ds, low level). The start bit is followed by eight data bits (D0–D7) and a parity bit (Dp).
3. On the smart card interface, the data line returns to high impedance after this. The data line is pulled high with a pull-up resistor.
4. The receiving side checks parity. When the data is received normally with no parity errors, the receiving side then waits to receive the next data. When a parity error occurs, the receiving side outputs an error signal (DE, low level) and requests re-transfer of data. The receiving station returns the signal line to high impedance after outputting the error signal for a specified period. The signal line is pulled high with a pull-up resistor.

5. The transmitting side transmits the next frame of data unless it receives an error signal. If it does receive an error signal, it returns to step 2 to re-transmit the erroneous data.

18.3.4 Register Settings

Table 18.3 shows the bit map of the registers that the smart card interface uses. Bits shown as 1 or 0 must be set to the indicated value. The settings for the other bits are described below.

Table 18.3Register Settings for the Smart Card Interface

Register	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SCSMR	H'FFFFFE80	C/$\overline{\text{A}}$	0	1	O/$\overline{\text{E}}$	1	0	CKS1	CKS0
SCBRR	H'FFFFFE82	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0
SCSCR	H'FFFFFE84	TIE	RIE	TE	RE	0	0	CKE1	CKE0
SCTDR	H'FFFFFE86	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0
SCSSR	H'FFFFFE88	TDRE	RDRF	ORER	FER/ ERS	PER	TEND	0	0
SCRDR	H'FFFFFE8A	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0
SCSCMR	H'FFFFFE8C	—	—	—	—	SDIR	SINV	—	SMIF

Note: Dashes indicate unused bits.

1. Setting the serial mode register (SCSMR): The C/$\overline{\text{A}}$ bit selects the set timing of the TEND flag, and selects the clock output state with the combination of bits CKE1 and CKE0 in the serial control register (SCSCR). Set the O/$\overline{\text{E}}$ bit to 0 when the IC card uses the direct convention or to 1 when it uses the inverse convention. Select the on-chip baud rate generator clock source with the CKS1 and CKS0 bits (see section 18.3.5, Clock).
2. Setting the bit rate register (SCBRR): Set the bit rate. See section 18.3.5, Clock, to see how to calculate the set value.
3. Setting the serial control register (SCSCR): The TIE, RIE, TE and RE bits function as they do for the ordinary SCI. See section 17, Serial Communication Interface, for more information. The CKE0 bit specifies the clock output. When no clock is output, set 0; when a clock is output, set 1.
4. Setting the smart card mode register (SCSCMR): The SDIR and SINV bits are both set to 0 for IC cards that use the direct convention and both to 1 when the inverse convention is used. The SMIF bit is set to 1 for the smart card interface.

Figure 18.4 shows sample waveforms for register settings of the two types of IC cards (direct convention and inverse convention) and their start characters.

In the direct convention type, the logical 1 level is state Z, the logical 0 level is state A, and communication is LSB first. The start character data is H'3B. The parity bit is even (from the smart card standards), and thus 1.

In the inverse convention type, the logical 1 level is state A, the logical 0 level is state Z, and communication is MSB first. The start character data is H'3F. The parity bit is even (from the smart card standards), and thus 0, which corresponds to state Z.

Only data bits D7–D0 are inverted by the SINV bit. To invert the parity bit, set the O/$\overline{\text{E}}$ bit in SCSMR to odd parity mode. This applies to both transmission and reception.

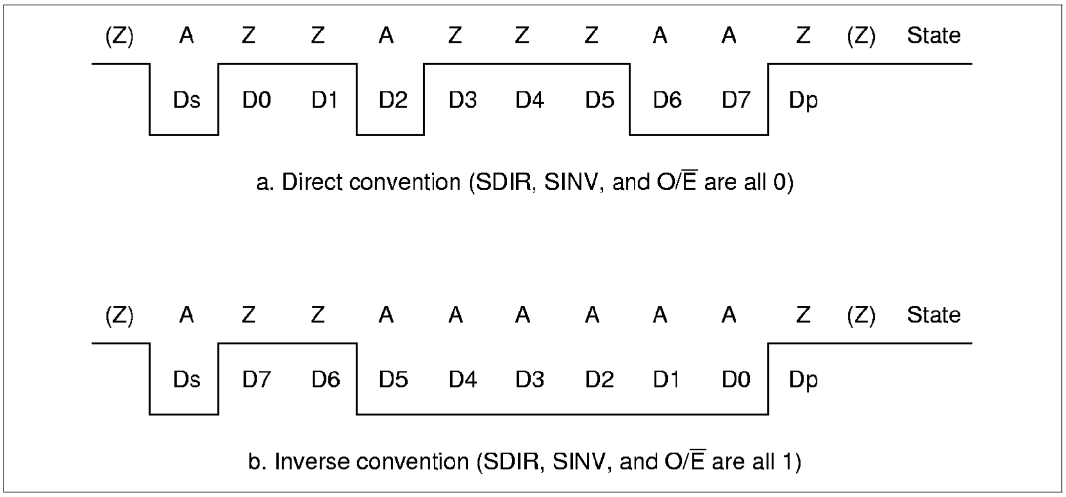

(Z) A Z Z A Z Z Z A A Z (Z) State
Ds D0 D1 D2 D3 D4 D5 D6 D7 Dp
a. Direct convention (SDIR, SINV, and O/Ē are all 0)
(Z) A Z Z A A A A A A Z (Z) State
Ds D7 D6 D5 D4 D3 D2 D1 D0 Dp
b. Inverse convention (SDIR, SINV, and O/Ē are all 1)

Figure 18.4 Waveform of Start Character

18.3.5 Clock

Only the internal clock generated by the on-chip baud rate generator can be used as the communication clock in the smart card interface. The bit rate for the clock is set by the bit rate register (SCBRR) and the CKS1 and CKS0 bits in the serial mode register (SCSMR), and is calculated using the equation below. Table 18.5 shows sample bit rates. If clock output is then selected by setting CKE0 to 1, a clock with a frequency 372 times the bit rate is output from the SCK0 pin.

$$B = \frac{P\phi}{1488 \times 2^{2n-1} \times (N + 1)} \times 10^6$$

Where:

N = Value set in SCBRR ($0 \leq N \leq 255$)
B = Bit rate (bit/s)

Pϕ = Peripheral module operating frequency (MHz)
n = 0–3 (table 18.4)

Table 18.4Relationship of n to CKS1 and CKS0

n	CKS1	CKS0
0	0	0
1	0	1
2	1	0
3	1	1

Table 18.5Examples of Bit Rate B (Bit/s) for SCBRR Settings (n = 0)

	Pϕ (MHz)						
N	7.1424	10.00	10.7136	13.00	14.2848	16.00	18.00
0	9600.0	13440.9	14400.0	17473.1	19200.0	21505.4	24193.5
1	4800.0	6720.4	7200.0	8736.6	9600.0	10752.7	12096.8
2	3200.0	4480.3	4800.0	5824.4	6400.0	7168.5	8064.5

Note: The bit rate is rounded to two decimal places.

Calculate the value to be set in the bit rate register (SCBRR) from the operating frequency and the bit rate. N is an integer in the range 0 ≤ N ≤ 255, specifying a smallish error.

$$N = \frac{P\phi}{1488 \times 2^{2n-1} \times B} \times 10^{6} - 1$$

Table 18.6Examples of SCBRR Settings for Bit Rate B (Bit/s) (n = 0)

ϕ (MHz) (9600 Bits/s)

7.1424		10.00		10.7136		13.00		14.2848		16.00		18.00	
N	Error	N	Error	N	Error	N	Error	N	Error	N	Error	N	Error
0	0.00	1	30.00	1	25.00	1	8.99	1	0.00	1	12.01	2	15.99

Table 18.7Maximum Bit Rates for Frequencies (Smart Card Interface Mode)

Pφ (MHz)	Maximum Bit Rate (Bit/s)	N	n
7.1424	9600	0	0
10.00	13441	0	0
10.7136	14400	0	0
13.00	17473	0	0
14.2848	19200	0	0
16.00	21505	0	0
18.00	24194	0	0

The bit rate error is found as follows:

$$\text{Error (\%)} = \left(\frac{P\phi}{1488 \times 2^{2n-1} \times B \times (N + 1)} \times 10^6 - 1\right) \times 100$$

Table 18.8 shows the relationship between transmit/receive clock register set values and output states on the smart card interface.

Table 18.8Register Set Values and SCK Pin

	Reglster Value				SCK Pln	
Setting	SMIF	C/$\overline{\text{A}}$	CKE1	CKE0	Output	State
1*1	1	0	0	0	Port	Determined by setting of port register SCP1MD1 and SCP1MD0 bits
	1	0	0	1	⎍⎍	SCK (serial clock) output state
2*2	1	1	0	0	Low output	Low output state
	1	1	0	1	⎍⎍	SCK (serial clock) output state
3*2	1	1	1	0	High output	High output state
	1	1	1	1	⎍⎍	SCK (serial clock) output state

Notes: 1. The SCK output state changes as soon as the CKE0 bit is modified. The CKE1 bit should be cleared to 0.

2. The clock duty remains constant despite stopping and starting of the clock by modification of the CKE0 bit.

18.3.6 Data Transmission and Reccption

Initialization: Initialize the SCI using the following procedure before sending or receiving data. Initialization is also required for switching from transmit mode to receive mode or from receive mode to transmit mode. Figure 18.5 shows a flowchart of the initialization process.

1. Clear TE and RE in the serial control register (SCSCR) to 0.
2. Clear error flags FER/ERS, PER, and ORER to 0 in the serial status register (SCSSR).
3. Set the C/$\overline{\text{A}}$ bit, parity bit (O/$\overline{\text{E}}$ bit), and baud rate generator select bits (CKS1 and CKS0 bits) in the serial mode register (SCSMR). At this time also clear the CHR and MP bits to 0 and set the STOP and PE bits to 1.
4. Set the SMIF, SDIR, and SINV bits in the smart card mode register (SCSCMR). When the SMIF bit is set to 1, the TxD and RxD pins both switch from ports to SCI pins and become high impedance.
5. Set the value corresponding to the bit rate in the bit rate register (SCBRR).
6. Set the clock source select bits (CKE1 and CKE0 bits) in the serial control register (SCSCR). Clear the TIE, RIE, TE, RE, MPIE, and TEIE bits to 0. When the CKE0 bit is set to 1, a clock is output from the SCK pin.
7. Aftcr waiting at lcast 1 bit, sct thc TIE, RIE, TE, and RE bits in SCSCR. Do not sct thc TE and RE bits simultaneously unless performing auto-diagnosis.

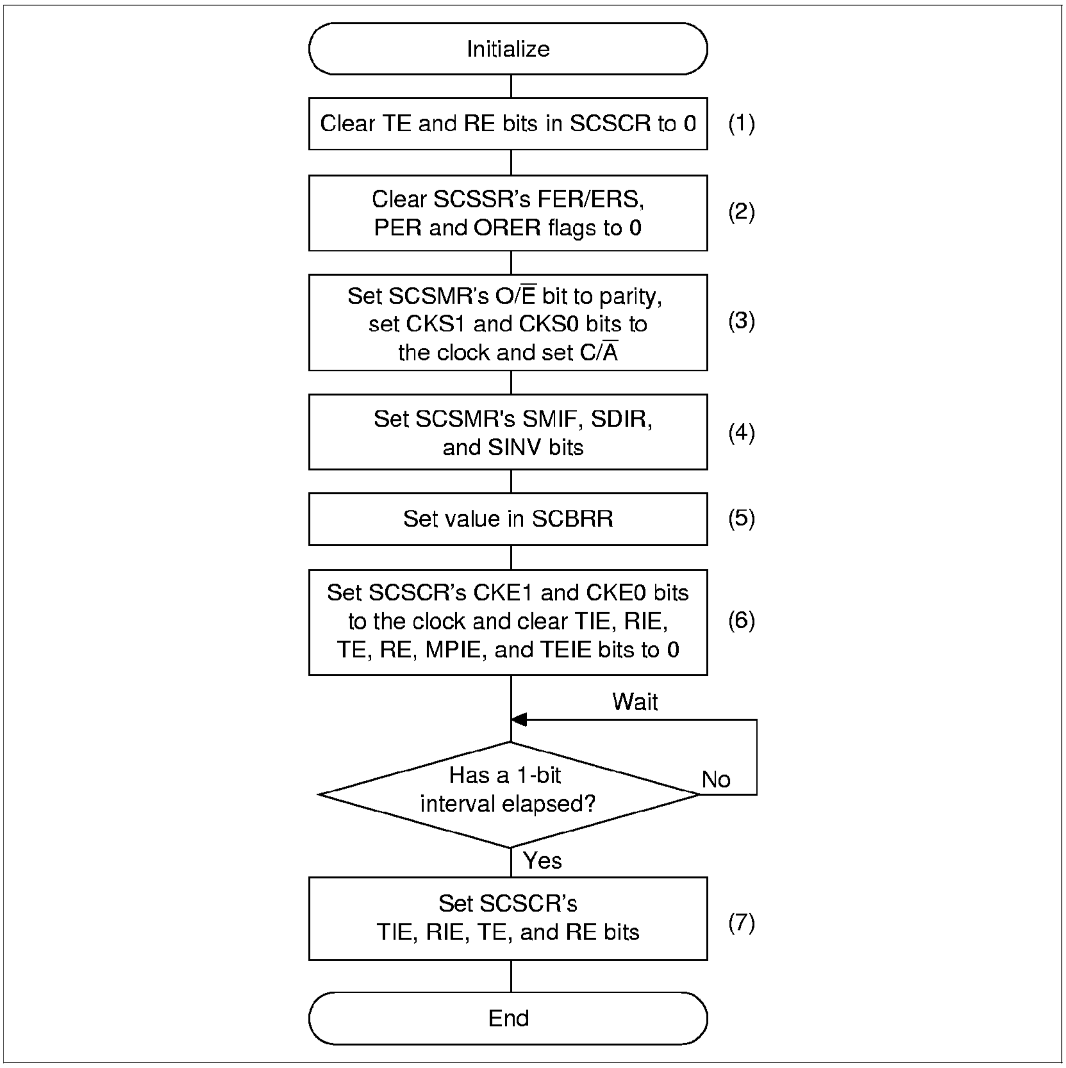

Initialize
Clear TE and RE bits in SCSCR to 0
(1)
Clear SCSSR's FER/ERS, PER and ORER flags to 0
(2)
Set SCSMR's O/$\overline{E}$ bit to parity, set CKS1 and CKS0 bits to the clock and set C/$\overline{A}$
(3)
Set SCSMR's SMIF, SDIR, and SINV bits
(4)
Set value in SCBRR
(5)
Set SCSCR's CKE1 and CKE0 bits to the clock and clear TIE, RIE, TE, RE, MPIE, and TEIE bits to 0
(6)
Wait
Has a 1-bit interval elapsed?
No
Yes
Set SCSCR's TIE, RIE, TE, and RE bits
(7)
End

Figure 18.5 Initialization Flowchart (Example)

Serial Data Transmission: The processing procedures in the smart card mode differ from ordinary SCI processing because data is retransmitted when an error signal is sampled during a data transmission. This results in the transmission processing flowchart shown in figure 18.6.

1. Initialize the smart card interface mode as described in initialization above.
2. Check that the FER/ERS bit in SCSSR is cleared to 0.
3. Repeat steps 2 and 3 until the TEND flag in SCSSR is set to 1.
4. Write the transmit data into SCTDR, clear the TDRE flag to 0 and start transmitting. The TEND flag will be cleared to 0.
5. To transmit more data, return to step 2.
6. To end transmission, clear the TE bit to 0.

This processing can be interrupted. When the TIE bit is set to 1 and interrupt requests are enabled, a transmit-data-empty interrupt (TXI) will be requested when the TEND flag is set to 1 at the end of the transmission. When the RIE bit is set to 1 and interrupt requests are enabled, a communication error interrupt (ERI) will be requested when the ERS flag is set to 1 when an error occurs in transmission. See Interrupt Operation below for more information.

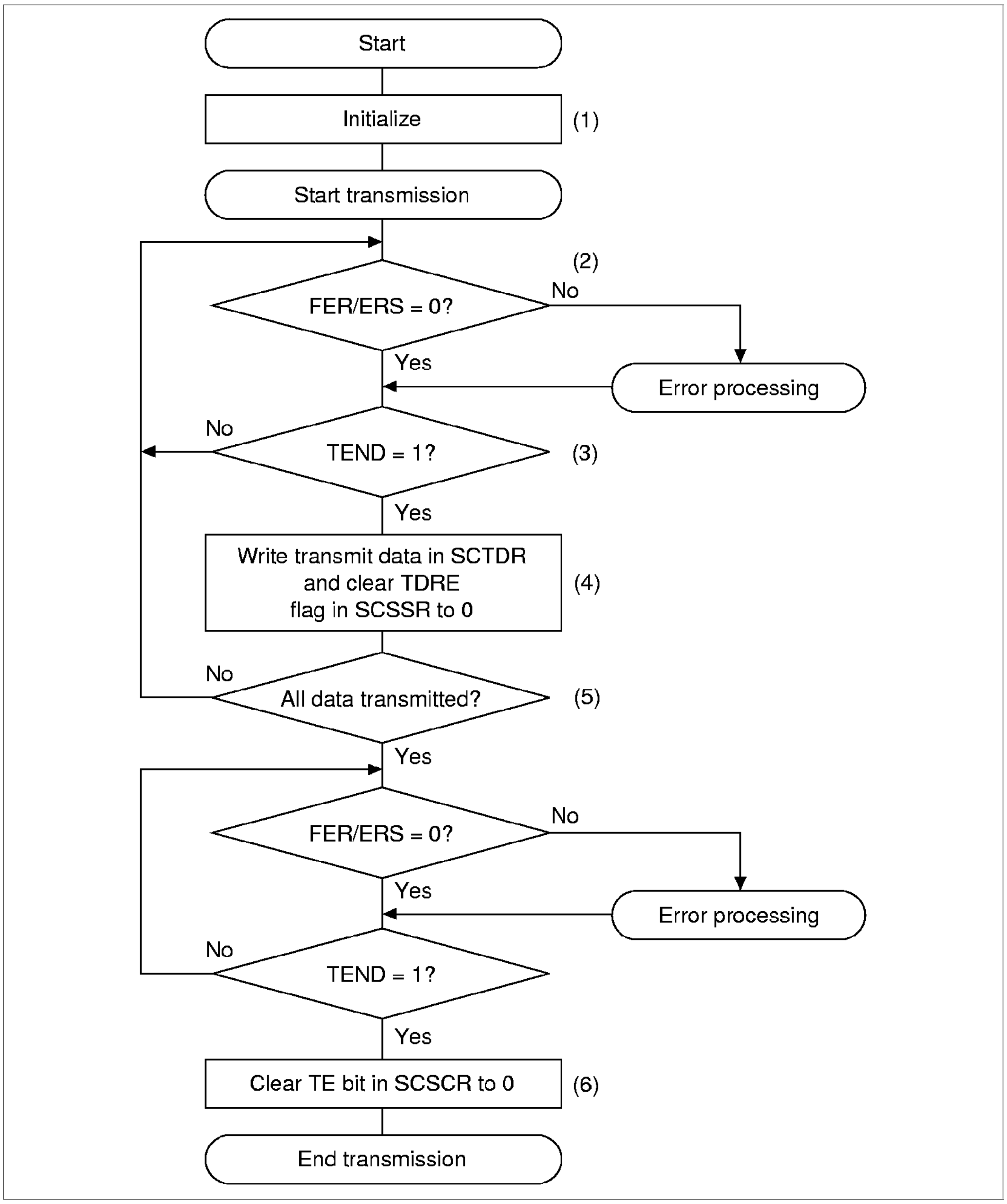


Figure 18.6 Transmission Flowchart

Serial Data Reception: The processing procedures in the smart card mode are the same as in ordinary SCI processing. The reception processing flowchart is shown in figure 18.7.

1. Initialize the smart card interface mode as described above in Initialization and in figure 18.5.
2. Check that the ORER and PER flags in SCSSR are cleared to 0. If either flag is set, clear both to 0 after performing the appropriate error processing procedures.
3. Repeat steps 2 and 3 until the RDRF flag is set to 1.
4. Read the receive data from SCRDR.
5. To receive more data, clear the RDRF flag to 0 and return to step 2.
6. To end reception, clear the RE bit to 0.

This processing can be interrupted. When the RIE bit is set to 1 and interrupt requests are enabled, a receive-data-full interrupt (RXI) will be requested when the RDRF flag is set to 1 at the end of the reception. When an error occurs during reception and either the ORER or PER flag is set to 1, a communication error interrupt (ERI) will be requested. See Interrupt Operation below for more information.

The received data will be transferred to SCRDR even when a parity error occurs during reception and PER is set to 1, so this data can still be read.

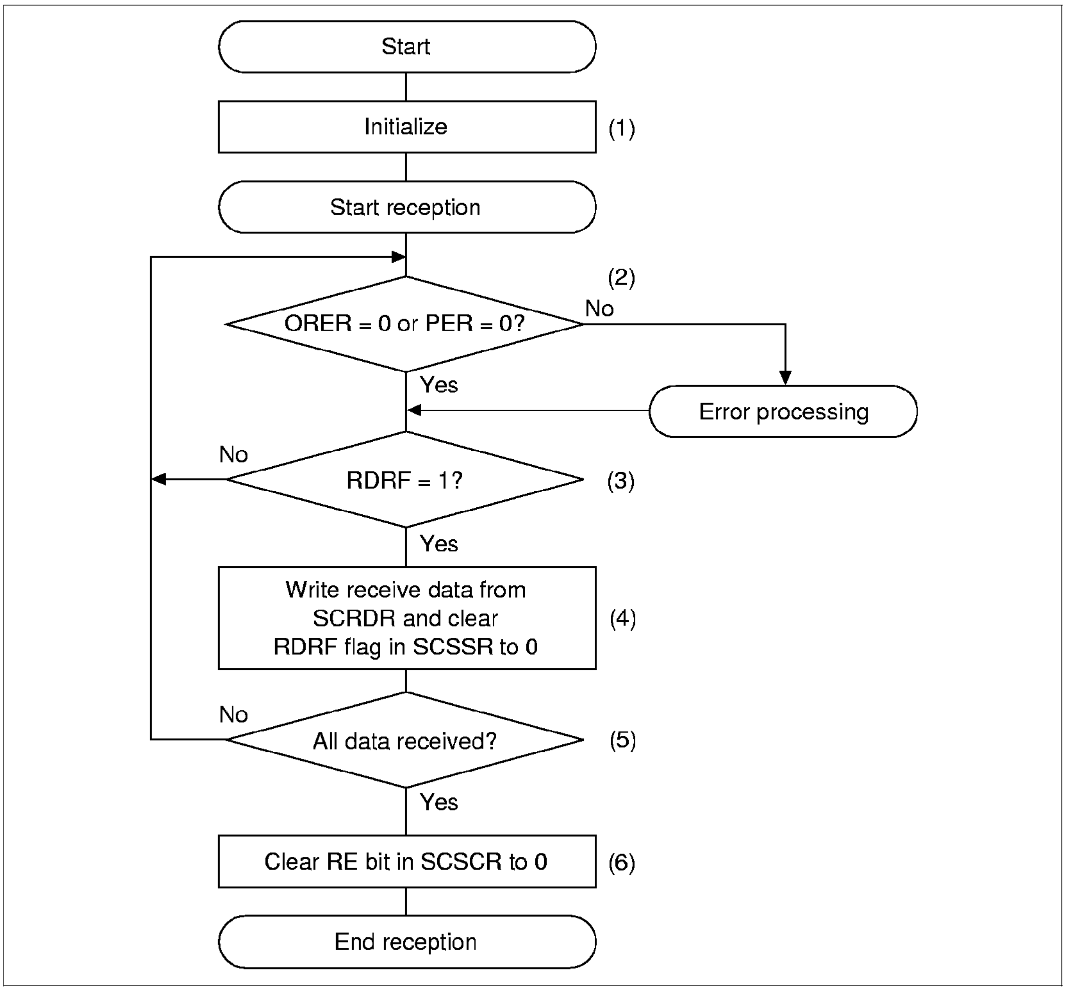


Figure 18.7 Reception Flowchart (Example)

Switching Modes: When switching from receive mode to transmit mode, check that the receive operation is completed before starting initialization and setting RE to 0 and TE to 1. The RDRF, PER, and ORER flags can be used to check if reception is completed. When switching from transmit mode to receive mode, check that the transmit operation is completed before starting initialization and setting TE to 0 and RE to 1. The TEND flag can be used to check if transmission is completed.

Interrupt Operation: In the smart card interface mode, there are three types of interrupts: transmit-data-empty (TXI), communication error (ERI) and receive-data-full (RXI). In this mode, the transmit-end interrupt (TEI) cannot be requested.

Set the TEND flag in SCSSR to 1 to request a TXI interrupt. Set the RDRF flag in SCSSR to 1 to request an RXI interrupt. Set the ORER, PER, or FER/ERS flag in SCSSR to 1 to request an ERI interrupt (table 18.9).

Table 18.9Smart Card Mode Operating State and Interrupt Sources

Mode	State	Flag	Mask Bit	Interrupt Source
Transmit mode	Normal	TEND	TIE	TXI
	Error	FER/ERS	RIE	ERI
Receive mode	Normal	RDRF	RIE	RXI
	Error	PER, ORER	RIE	ERI

18.4 Usage Notes

When the SCI is used as a smart card interface, be sure that all criteria in sections 18.4.1 and 18.4.2 are applied.

18.4.1 Receive Data Timing and Receive Margin in Asynchronous Mode

In asynchronous mode, the SCI runs on a basic clock with a frequency of 372 times the transfer rate. During reception, the SCI samples the falling of the start bit using the base clock to achieve internal synchronization. Receive data is latched internally on the rising edge of the 186th basic clock cycle (figure 18.8).

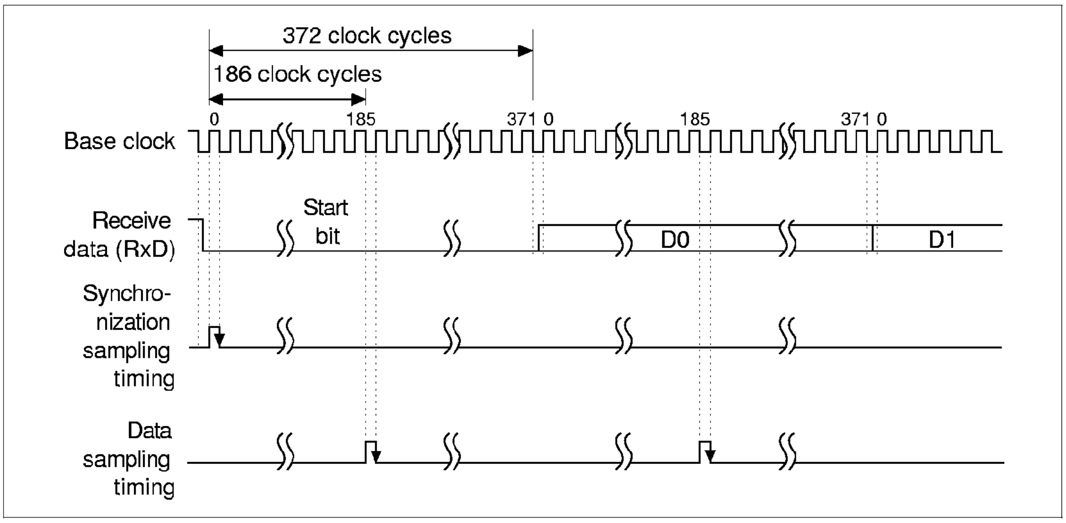


Figure 18.8 Receive Data Sampling Timing in Smart Card Mode

The receive margin is found from the following equation:

For smart card mode:

$$M = \left| (0.5 - \frac{1}{2N}) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100\%$$

Where:

M = Receive margin (%)
N = Ratio of bit rate to clock (N = 372)
D = Clock duty (D = 0 to 1.0)
L = Frame length (L = 10)
F = Absolute value of clock frequency deviation

Using this equation, the receive margin when F = 0 and D = 0.5 is as follows:

$$M = (0.5 - 1/2 \times 372) \times 100\% = 49.866\%$$

18.4.2 Retransmission (Receive and Transmit Modes)

Retransmission by the SCI in Receive Mode: Figure 18.9 shows the retransmission operation in the SCI receive mode.

1. When the received parity bit is checked and an error is found, the PER bit in SCSSR is automatically set to 1. If the RIE bit in SCSCR is enabled at this time, an ERI interrupt is requested. Be sure to clear the PER bit before the next parity bit is sampled.
2. The RDRF bit in SCSSR is not set in the frame that caused the error.
3. When the received parity bit is checked and no error is found, the PER bit in SCSSR is not set.
4. When the received parity bit is checked and no error is found, reception is considered to have been completed normally and the RDRF bit in SCSSR is automatically set to 1. If the RIE bit in SCSCR is enabled at this time, an RXI interrupt is requested.
5. When a normal frame is received, the pin maintains a three-state state when it transmits the error signal.

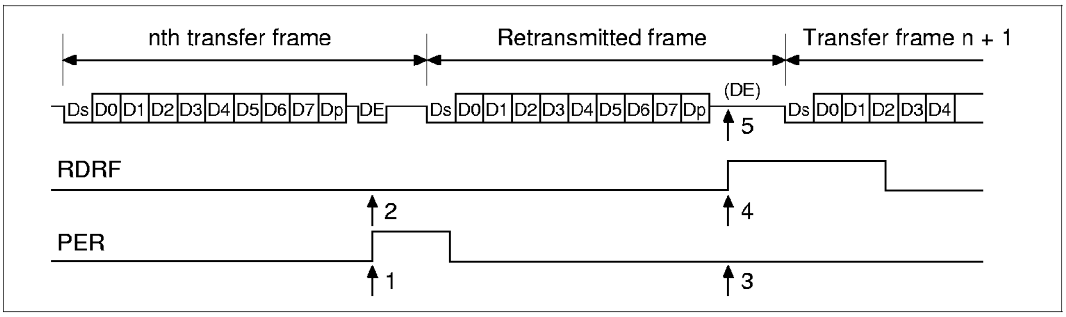


Figure 18.9 Retransmission in SCI Receive Mode

Retransmission by the SCI in Transmit Mode: Figure 18.10 shows the retransmission operation in the SCI transmit mode.

1. After transmission of one frame is completed, the FER/ERS bit in SCSSR is set to 1 when a error signal is returned from the receiving side. If the RIE bit in SCSCR is enabled at this time, an ERI interrupt is requested. Be sure to clear the FER/ERS bit before the next parity bit is sampled.
2. The TEND bit in SCSSR is not set in the frame that received the error signal that indicated the error.
3. The FER/ERS bit in SCSSR is not set when no error signal is returned from the receiving side.
4. When no error signal is returned from the receiving side, the TEND bit in SCSSR is set to 1 when the transmission of the frame that includes the retransmission is considered completed. If the TIE bit in SCSCR is enabled at this time, a TXI interrupt will be requested.

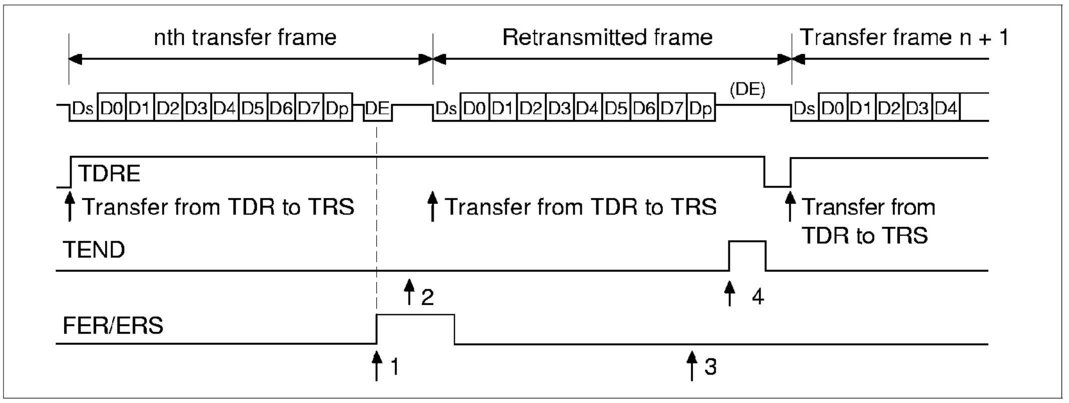


Figure 18.10 Retransmission in SCI Transmit Mode

Section 19 Serial Communication Interface with FIFO (SCIF)

19.1 Overview

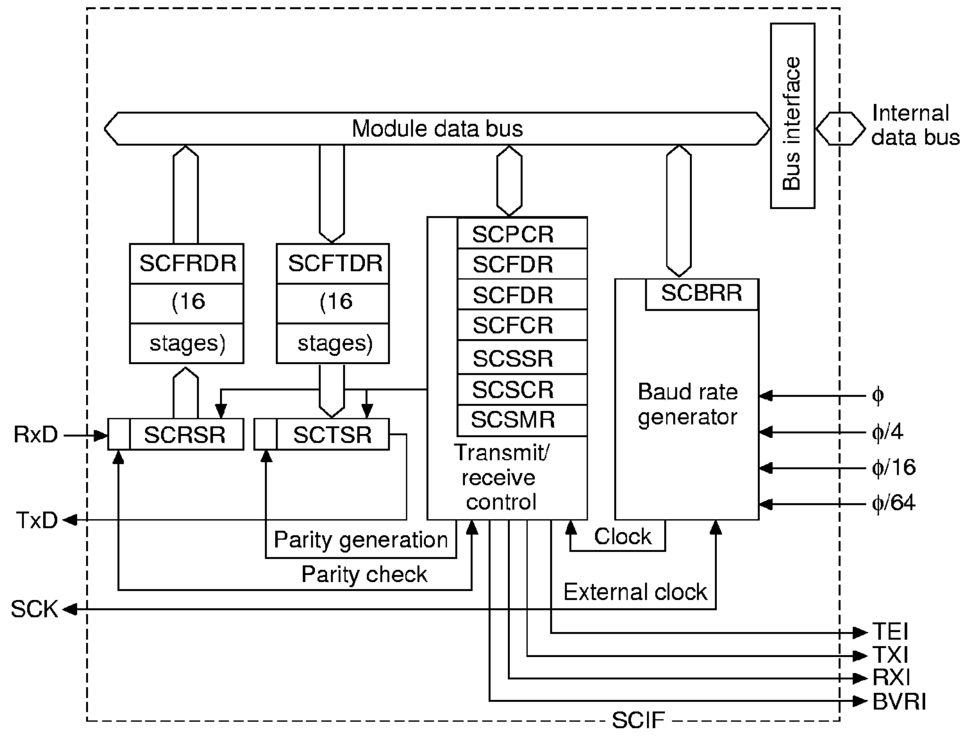
This LSI has two-channel serial communication interface with FIFO (SCIF) that supports asynchronous serial communication. It also has 16-stage FIFO registers for both transfer and receive that enables this LSI efficient high-speed continuous communication.

19.1.1 Features

- Asynchronous serial communication:
  - Serial data communications are performed by start-stop in character units. The SCI can communicate with a universal asynchronous receiver/transmitter (UART), an asynchronous communication interface adapter (ACIA), or any other communications chip that employs a standard asynchronous serial system. There are eight selectable serial data communication formats.
  - Data length: Seven or eight bits
  - Stop bit length: One or two bits
  - Parity: Even, odd, or none
  - Receive error detection: Parity and framing errors
  - Break detection: Break is detected when the receive data next the generated framing error is the space ɸ level and has the framing error. It is also detected by reading the RxD level directly from the port SC data register (SCPDR) when a framing error occurs
- Full duplex communication: The transmitting and receiving sections are independent, so the SCI can transmit and receive simultaneously. Both sections use 16-stage FIFO buffering, so high-speed continuous data transfer is possible in both the transmit and receive directions.
- On-chip baud rate generator with selectable bit rates
- Internal or external transmit/receive clock source: From either baud rate generator (internal) or SCK pin (external)
- Four types of interrupts: Transmit-FIFO-data-empty, break, receive-FIFO-data-full, and receive-error interrupts are requested independently. The direct memory access controller (DMAC) can be activated to execute a data transfer by a transmit-FIFO-data-empty or receive-FIFO-data-full interrupt.
- When the SCIF is not in use, it can be stopped by halting the clock supplied to it, saving power.
- On-chip modem control functions (RTS and CTS)
- The quantity of data in the transmit and receive FIFO registers and the number of receive errors of the receive data in the receive FIFO register can be known.
- The time-out error (DR) can be detected in receiving.

19.1.2 Block Diagram

Figure 19.1 shows a block diagram of the SCI.

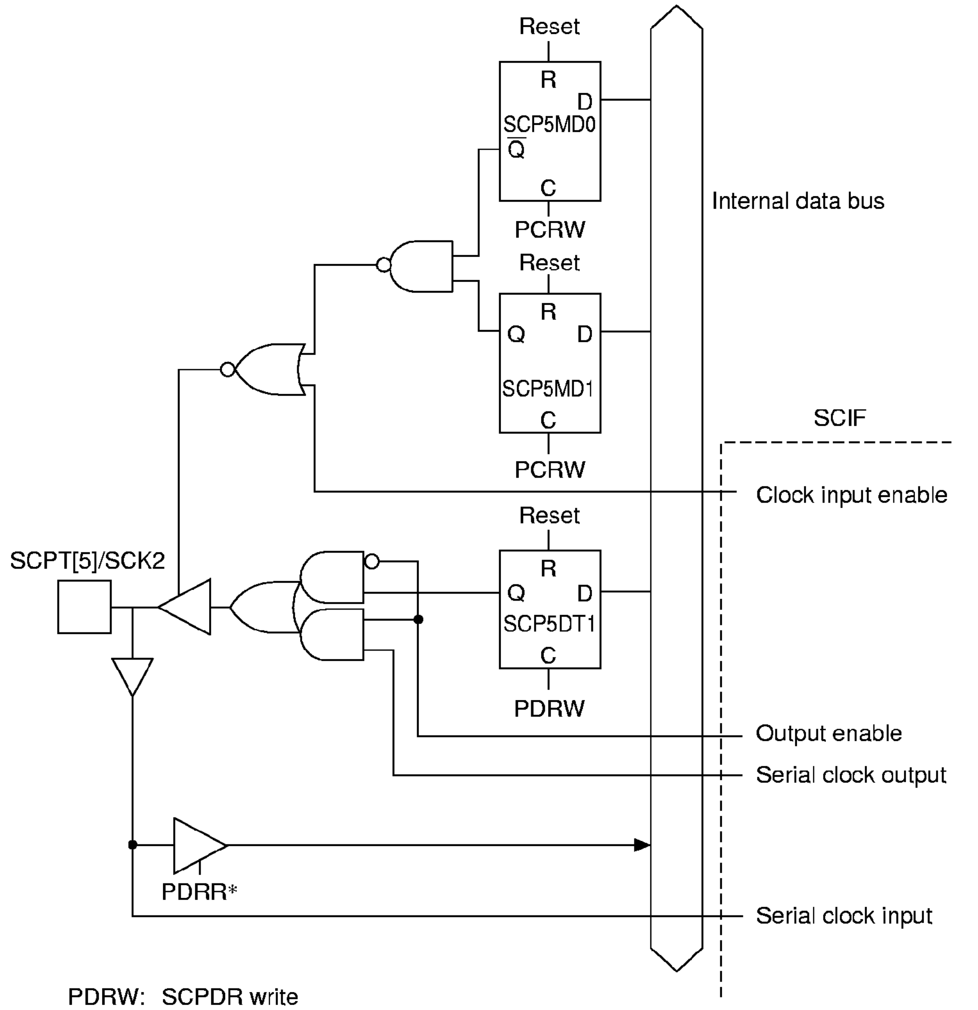


SCRSR:	Receive shift register	SCSSR:	Serial status register
SCFRDR:	Receive FIFO data register	SCBRR:	Bit rate register
SCTSR:	Transmit shift register	SCFCR:	FIFO control register
SCFTDR:	Transmit FIFO data register	SCFDR:	Number of FIFO data registers
SCSMR:	Serial mode register	SCPDR:	Port SC data register
SCSCR:	Serial control register	SCPCR:	Port SC control register

Figure 19.1 SCIF Block Diagram

Figures 19.2 to 19.4 show the SCIF I/O ports.

SCIF pin I/O and data control is performed by bits 11–8 of SCPCR and bits 5 and 4 of SCPDR. For details, see section 14.2.8.



PDRW: SCPDR write
PDRR: SCPDR read
PCRW: SCPCR write

Note: * When reading the SCK2 pin, clear the C/$\overline{A}$ bit in SCSMR and the CKE1 and CKE0 bits in SCSCR to 0, and set the SCP5MD1 bit in SCSPR to 1 (see section 19.2.8).

Figure 19.2 SCPT[5]/SCK2 Pin

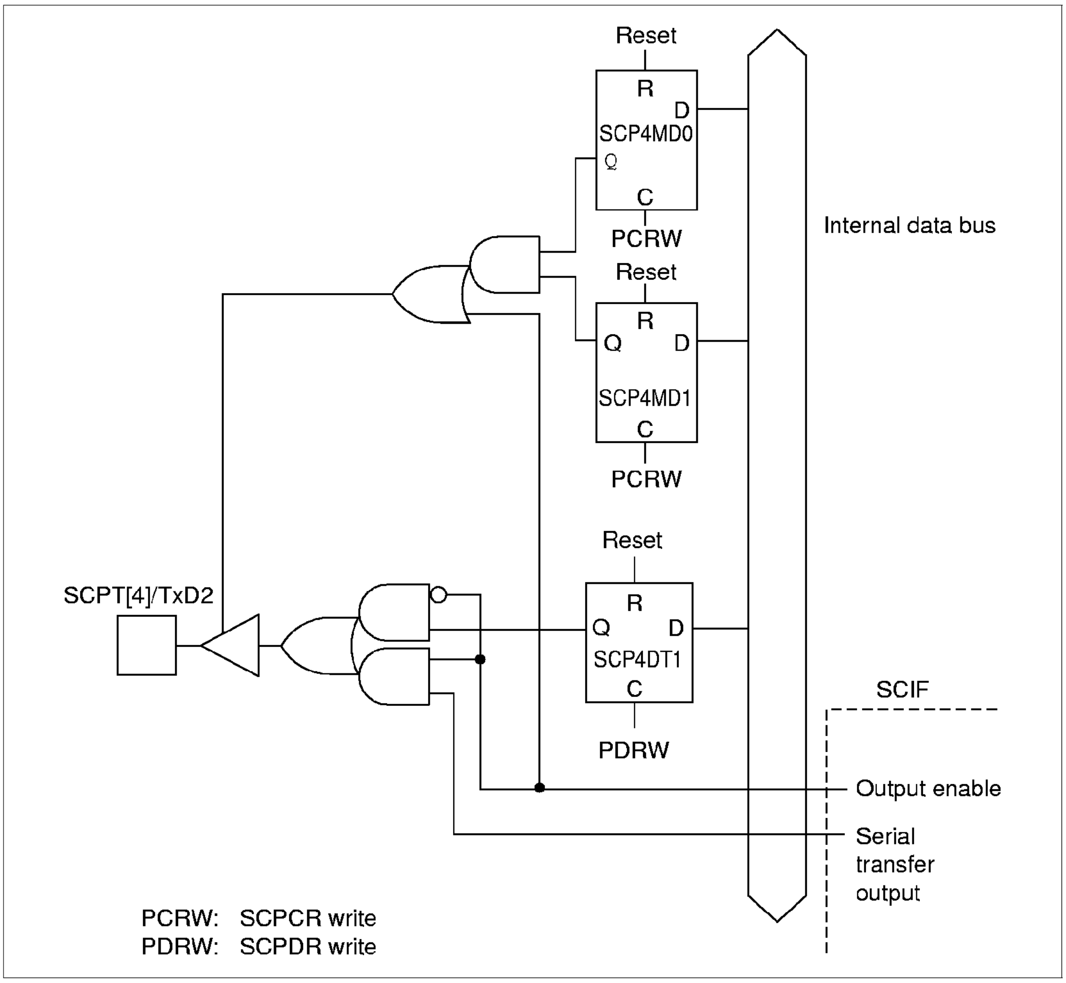
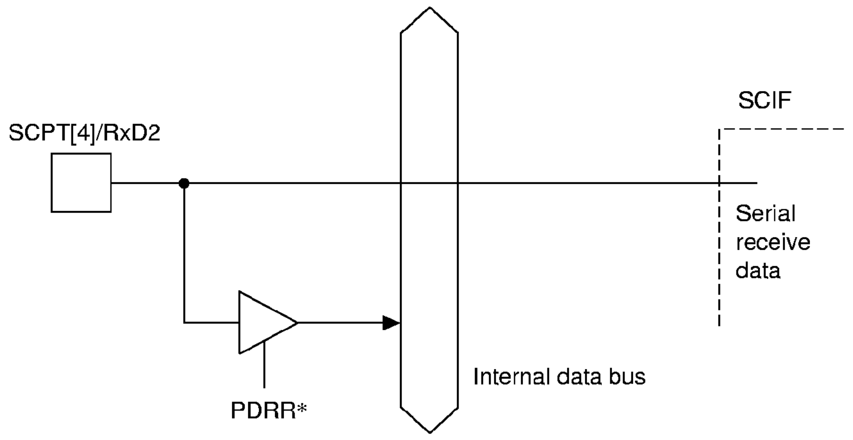


Figure 19.3 SCPT[4]/TxD2 Pin



PDRR: SCPDR read

Note: * When reading the RxD2 pin, set the RE bit in SCSCR to 1.

Figure 19.4 SCPT[4]/RxD2 Pin

19.1.3 Pin Configuration

The SCIF has the serial pins summarized in table 19.1.

Table 19.1SCIF Pins

Pin Name	Abbreviation	I/O	Function
Serial clock pin	SCK2	I/O	Clock I/O
Receive data pin	RxD2	Input	Receive data input
Transmit data pin	TxD2	Output	Transmit data output
Request to send pin	$\overline{\text{RTS2}}$	Output	Request to send
Clear to send pin	$\overline{\text{CTS2}}$	Input	Clear to send

19.1.4 Register Configuration

Table 19.2 summarizes the SCIF internal registers. These registers specify the data format and bit rate, and control the transmitter and receiver sections.

Table 19.2Registers

Register Name	Abbreviation	R/W	Initial Value	Address	Access size
Serial mode register 2	SCSMR2	R/W	H'00	H'4000150	8 bits
Bit rate register 2	SCBRR2	R/W	H'FF	H'4000152	8 bits
Serial control register 2	SCSCR2	R/W	H'00	H'4000154	8 bits
Transmit FIFO data register 2	SCFTDR2	W	—	H'4000156	8 bits
Serial status register 2	SCSSR2	R/(W)*1	H'0060	H'4000158	16 bits
Receive data FIFO register 2	SCFRDR2	R	Undefined	H'400015A	8 bits
FIFO control register 2	SCFCR2	R/W	H'00	H'400015C	8 bits
FIFO data count set register 2	SCFDR2	R	H'0000	H'400015E	16 bits

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

19.2 Register Descriptions

19.2.1 Receive Shift Register

The receive shift register (SCRSR) receives serial data. Data input at the RxD pin is loaded into the SCRSR in the order received, LSB (bit 0) first, converting the data to parallel form. When one byte has been received, it is automatically transferred to the SCFRDR, which is a receive FIFO register. The CPU cannot read or write the SCRSR directly.

Bit:	7	6	5	4	3	2	1	0
Bit name:								
R/W:	—	—	—	—	—	—	—	—

19.2.2 Receive FIFO Data Register

The 16-byte receive FIFO data register (SCFRDR) stores serial receive data. The SCIF completes the reception of one byte of serial data by moving the received data from the receive shift register (SCRSR) into the SCFRDR for storage. Continuous receive is enabled until 16 bytes are stored.

The CPU can read but not write the SCFRDR. When data is read without received data in the receive FIFO data register, the value is undefined. When the received data in this register becomes full, the subsequent serial data is lost.

Bit:	7	6	5	4	3	2	1	0
Bit name:								
R/W:	R	R	R	R	R	R	R	R

19.2.3 Transmit Shift Register

The transmit shift register (SCTSR) transmits serial data. The SCI loads transmit data from the transmit FIFO data register (SCFTDR) into the SCTSR, then transmits the data serially from the TxD pin, LSB (bit 0) first. After transmitting one data byte, the SCI automatically loads the next transmit data from the SCFTDR into the SCTSR and starts transmitting again. The CPU cannot read or write the SCTSR directly.

Bit:	7	6	5	4	3	2	1	0
Bit name:								
R/W:	—	—	—	—	—	—	—	—

19.2.4 Transmit FIFO Data Register

The transmit FIFO data register (SCFTDR) is a 16-byte 8-bit-length FIFO register that stores data for serial transmission. When the SCIF detects that the transmit shift register (SCTSR) is empty, it moves transmit data written in the SCFTDR into the SCTSR and starts serial transmission. Continuous serial transmission is performed until the transmit data in the SCFTDR becomes empty. The CPU can always write to the SCFTDR.

When the transmit data in the SCFTDR is full (16 bytes), next data cannot be written. If attempted to write, the data is ignored.

Bit:	7	6	5	4	3	2	1	0
Bit name:								
R/W:	W	W	W	W	W	W	W	W

19.2.5 Serial Mode Register

The serial mode register (SCSMR) is an eight-bit register that specifies the SCIF serial communication format and selects the clock source for the baud rate generator.

The CPU can always read and write the SCSMR. The SCSMR is initialized to H'00 by a reset or in standby and module standby modes.

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	CHR	PE	O/$\overline{E}$	STOP	—	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R	R/W	R/W

Bit 6—Character Length (CHR): Selects seven-bit or eight-bit data in the asynchronous mode.

Bit 6: CHR	Description	
0	Eight-bit data.	(Initial value)
1	Seven-bit data. (When seven-bit data is selected, the MSB (bit 7) of the transmit FIFO data register is not transmitted.)	

Bit 5—Parity Enable (PE): Selects whether to add a parity bit to transmit data and to check the parity of receive data.

Bit 5: PE	Description	
0	Parity bit not added or checked.	(Initial value)
1	Parity bit added and checked. When PE is set to 1, an even or odd parity bit is added to transmit data, depending on the parity mode (O/$\overline{E}$) setting. Receive data parity is checked according to the even/odd (O/$\overline{E}$) mode setting.	

Bit 4—Parity Mode (O/$\overline{E}$): Selects even or odd parity when parity bits are added and checked. The O/$\overline{E}$ setting is used only when the parity enable bit (PE) is set to 1 to enable parity addition and check. The O/$\overline{E}$ setting is ignored when parity addition and check is disabled.

Bit 4: O/$\overline{\text{E}}$	Description
0	Even parity. (Initial value). If even parity is selected, the parity bit is added to transmit data to make an even number of 1s in the transmitted character and parity bit combined. Receive data is checked to see if it has an even number of 1s in the received character and parity bit combined.
1	Odd parity. If odd parity is selected, the parity bit is added to transmit data to make an odd number of 1s in the transmitted character and parity bit combined. Receive data is checked to see if it has an odd number of 1s in the received character and parity bit combined.

Bit 3—Stop Bit Length (STOP): Selects one or two bits as the stop bit length.

In receiving, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, it is treated as a stop bit, but if the second stop bit is 0, it is treated as the start bit of the next incoming character.

Bit 3: STOP	Description
0	One stop bit. (Initial value) In transmitting, a single bit of 1 is added at the end of each transmitted character.
1	Two stop bits. In transmitting, two bits of 1 are added at the end of each transmitted character.

Bits 1 and 0—Clock Select 1 and 0 (CKS1 and CKS0): These bits select the internal clock source of the on-chip baud rate generator. Four clock sources are available. Pϕ, Pϕ/4, Pϕ/16 and Pϕ/64. For further information on the clock source, bit rate register settings, and baud rate, see section 19.2.8, Bit Rate Register.

Bit 1: CKS1	Bit 0: CKS0	Description
0	0	Pϕ (Initial value)
	1	Pϕ/4
1	0	Pϕ/16
	1	Pϕ/64

Note: Pϕ: Peripheral clock

19.2.6 Serial Control Register

The serial control register (SCSCR) operates the SCI transmitter/receiver, selects the serial clock output in the asynchronous mode, enables/disables interrupt requests, and selects the transmit/receive clock source. The CPU can always read and write the SCSCR. The SCSCR is initialized to H'00 by a reset or in standby and module standby modes.

Bit:	7	6	5	4	3	2	1	0
Bit name:	TIE	RIE	TE	RE	0	0	CKE1	CKE0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7—Transmit Interrupt Enable (TIE): Enables or disables the transmit-FIFO-data-empty interrupt (TXI) requested when the serial transmit data is transferred from the transmit FIFO data register (SCFTDR) to transmit shift register (SCTSR), when the quantity of data in the transmit FIFO register becomes less than the specified number of transmission triggers, and when the TDFE flag of the serial FIFO status register (SCFSR) is set to1.

Bit 7: TIE	Description
0	Transmit-FIFO-data-empty interrupt request (TXI) is disabled. (Initial value) The TXI interrupt request can be cleared by writing the greater quantity of transmit data than the specified number of transmission triggers to SCFTDR and by clearing TDFE to 0 after reading 1 from TDFE, or can be cleared by clearing TIE to 0.
1	Transmit-FIFO-data-empty interrupt request (TXI) is enabled

Bit 6—Receive Interrupt Enable (RIE): Enables or disables the receive-data-full (RXI) and receive-error (ERI) interrupts requested when the serial receive data is transferred from the receive shift register (SCRSR) to receive FIFO data register (SCFRDR), when the quantity of data in the receive FIFO register becomes more than the specified number of receive triggers, and when the RDRF flag of SCSSR is set to1.

Bit 6: RIE	Description
0	Receive-data-full interrupt (RXI), receive-error interrupt (ERI), and receive break interrupt (BRI) requests are disabled. (Initial value) RXI and ERI interrupt requests can be cleared by reading the DR, ER, or RDF flag after it has been set to 1, then clearing the flag to 0, or by clearing RIE to 0. At RDF, read 1 from the RDF flag and clear it to 0, after reading the received data from SCRDR until the quantity of received data becomes less than the specified number of the receive triggers.
1	Receive-data-full interrupt (RXI) and receive-error interrupt (ERI) requests are enabled.

Bit 5—Transmit Enable (TE): Enables or disables the SCIF serial transmitter.

Bit 5: TE	Description
0	Transmitter disabled. (Initial value)
1	Transmitter enabled. Serial transmission starts after writing of transmit data into the SCFTDR. Select the transmit format in the SCSMR and SCFCR and reset the TFIFO before setting TE to 1.

Bit 4—Receive Enable (RE): Enables or disables the SCIF serial receiver.

Bit 4: RE	Description
0	Receiver disabled. (Initial value) Clearing RE to 0 does not affect the receive flags (DR, ER, BRK, FER, PER, and ORER). These flags retain their previous values.
1	Receiver enabled. Serial reception starts when a start bit is detected. Select the receive format in the SCSMR before setting RE to 1.

Bits 1 and 0—Clock Enable 1 and 0 (CKE1 and CKE0): These bits select the SCI clock source and enable or disable clock output from the SCK pin. Depending on the combination of CKE1 and CKE0, the SCK pin can be used for serial clock output or serial clock input.

The CKE0 setting is valid only when the SCI is operating with the internal clock (CKE1 = 0). The CKE0 setting is ignored when an external clock source is selected (CKE1= 1). Before selecting the SCIF operating mode in the serial mode register (SCSMR), set CKE1 and CKE0. For further details on selection of the SCI clock source, see table 19.8 in section 19.3, Operation.

Bit 1: CKE1	Bit 0: CKE0	Description
0	0	Internal clock, SCK pin used for input pin (input signal is ignored) (Initial value)
	1	Internal clock, SCK pin used for clock output*[1]
1	0	External clock, SCK pin used for clock input*[2]
	1	External clock, SCK pin used for clock input*[2]

Notes: 1. The output clock frequency is 16 times the bit rate.

2. The input clock frequency is 16 times the bit rate.

19.2.7 Serial Status Register

The serial status register (SCSSR) is a 16-bit register. The upper 8 bits indicate the number of receive errors in the data of the receive FIFO register, and the lower 8 bits indicate SCI operating state.

The CPU can always read and write the SCSSR, but cannot write 1 in the status flags (ER, TEND, TDFE, BRK, OPER, and DR). These flags can be cleared to 0 only if they have first been read (after being set to 1). Bits 3 (FER) and 2 (PER) are read-only bits that cannot be written. The SCSSR is initialized to H'0060 by a reset or in standby and module standby modes.

Lower 8 bits:	7	6	5	4	3	2	1	0
Bit name:	ER	TEND	TDFE	BRK	FER	PER	RDF	DR
Initial value:	0	1	1	0	0	0	0	0
R/W:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/(W)*	R/(W)*

Note: The only value that can be written is 0 to clear the flag.

Bit 7—Receive Error (ER): Indicates that a parity error has occurred when received data includes a framing error or a parity. *[1]

Bit 7: ER	Description
0	Receive is in progress, or receive is normally completed. (Initial value) ER is cleared to 0 when the chip is reset or enters standby mode, or when 0 is written after 1 is read from ER.
1	A framing error or a parity error has occurred. ER is set to 1 when the stop bit is 0 after checking whether or not the last stop bit of the received data is 1 at the end of one-data receive*[2], or when the total number of 1's in the received data and in the parity bit does not match the even/odd parity specification specified by the O/$\overline{E}$ bit of the SCSMR.

Notes: 1. Clearing the RE bit to 0 in SCSCR does not affect the ER bit, which retains its previous value. Even if a receive error occurs, the received data is transferred to SCFRDR and the receive operation is continued. Whether or not the data read from SCRDR includes a receive error can be detected by the FER and PER bits of SCSSR.

2. In the stop mode, only the first stop bit is checked; the second stop bit is not checked.

Bit 6—Transmit End (TEND): Indicates that when the last bit of a serial character was transmitted, the SCFTDR did not contain valid data, so transmission has ended.

Bit 6: TEND	Description
0	Transmission is in progress. TEND is cleared to 0 when data is written in SCTDR.
1	End of transmission. (Initial value) TEND is set to 1 when the chip is reset or enters standby mode, TE is cleared to 0 in the serial control register (SCSCR), or when SCFTDR does not contain received data when the last bit of a one-byte serial character is transmitted.

Bit 5—Transmit FIFO Data Empty (TDFE): Indicates that data is transferred from transmit FIFO data register (SCFTDR) to transmit shift register (SCTSR), the quantity of data in SCFTDR becomes less than the number of transmission triggers specified by the TTRG1 and TTRG0 bits in FIFO control register (SCFCR), and writing the transmit data to SCFTDR is enabled.

Bit 5: TDFE	Description
0	The quantity of transmit data written to SCFTDR is greater than the specified number of transmission triggers. (Initial value) TDFE is cleared to 0 when the data exceeding the specified number of transmission triggers is written to SCFTDR, software reads TDFE after it has been set to 1, then writes 0 in TDFE.
1	The quantity of transmit data in SCFTDR is less than the specified number of transmission triggers. * TDFE is set to 1 at reset or at standby mode, or when the quantity of transmission data in SCFTDR becomes less than the specified number of transmission triggers as a result of transmission.

Note: Since SCFTDR is a 16-byte FIFO register, the maximum quantity of data which can be written when TDFE is 1 is "16 minus the specified number of transmission triggers". If attempted to write additional data, the data is ignored. The quantity of data in SCFTDR is indicated by the upper 8 bits of SCFTDR.

Bit 4—Break Detection (BRK): Indicates that a break signal is detected in received data.

Bit 4: BRK	Description
0	No break signal is being received. (Initial value) BRK is cleared to 0 when the chip is reset or enters standby mode, or software reads BRK after it has been set to 1, then writes 0 in BRK.
1	The break signal is received. BRK is set to 1 when data including a framing error is received and a framing error occurs with space 0 in the subsequent received data.

Note: When a break is detected, transfer of the received data (H'00) to SCFRDR stops after detection. When the break ends and the receive signal becomes mark 1, the transfer of the received data resumes. The received data of a frame in which a break signal is detected is transferred to SCFRDR. After this, however, no received data is transferred until a break ends with the received signal being mark 1 and the next data is received.

Bit 3—Framing Error (FER): Indicates a framing error in the data read from the receive FIFO data register (SCFRDR).

Bit 3: FER	Description
0	No receive framing error occurred in the data read from SCFRDR. (Initial value) FER is cleared to 0 when the chip is power-on reset or enters standby mode, or when no framing error is present in the data read from SCFRDR.
1	A receive framing error occurred in the data read from SCFRDR. FER is set to 1 when a framing error is present in the data read from SCFRDR.

Bit 2—Parity Error (PER): Indicates a parity error in the data read from the receive FIFO data register (SCFRDR).

Bit 2: PER	Description
0	No receive parity error occurred in the data read from SCFRDR. (Initial value) PER is cleared to 0 when the chip is power-on reset or enters standby mode, or when no parity error is present in the data read from SCFRDR.
1	A receive framing error occurred in the data read from SCFRDR. PER is set to 1 when a parity error is present in the data read from SCFRDR.

Bit 1—Receive FIFO Data Full (RDF): Indicates that received data is transferred to the receive FIFO data register (SCFRDR), the quantity of data in SCFRDR becomes more than the number of receive triggers specified by the RTRG1 and RTRG0 bits in FIFO control register (SCFCR).

Bit 1: RDF	Description
0	The quantity of transmit data written to SCFRDR is less than the specified number of receive triggers. (Initial value) RDF is cleared to 0 at power-on reset or at standby mode, or cleared when the SCFRDR is read until the quantity of receive data in SCFRDR becomes less than the specified number of receive triggers, when 1 is read from RDF, and 0 is then written.
1	The quantity of receive data in SCFRDR is more than the specified number of receive triggers. RDF is set to 1 when the quantity of receive data which is greater than the specified number of receive triggers is stored in SCFRDR.*

Note: Since SCFTDR is a 16-byte FIFO register, the maximum quantity of data which can be read when RDF is 1 is the specified number of receive triggers. If attempted to read after all data in the SCFRDR have been read, the data is undefined. The quantity of receive data in SCFRDR is indicated by the lower 8 bits of SCFTDR.

Bit 0—Receive Data Ready (DR): Indicates that the receive FIFO data register (SCFRDR) stores the data which is less than the specified number of receive triggers, and that next data is not yet received after 15 etu has elapsed from the last stop bit.

Bit 0: DR	Description
0	Receive is in progress, or no received data remains in SCFRDR after completing receive normally. (Initial value) DR is cleared to 0 when the chip is power-on reset or enters standby mode, or software reads DR after it has been set to 1, then writes 0 in DR.
1	Next receive data is not received. DR is set to 1 when SCFRDR stores the data which is less than the specified number of receive triggers, and that next data is not yet received after 15 etu has elapsed from the last stop bit.*

Note: This is equivalent to 1.5 frames with the 8-bit 1-stop-bit format. (ETU: Element Time Unit)

Upper 8 bits:	15	14	13	12	11	10	9	8
Bit name:	PER3	PER2	PER1	PER0	FER3	FER2	FER1	FER0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bits 15–12—Number of Parity Errors (PER): Indicates the quantity of data including a parity error in the received data stored in the receive FIFO data register (SCFRDR). The value indicated by the bits 15 to 12 represents the number of parity errors in SCFRDR.

Bits 11–8—Number of Framing Errors (FER): Indicates the quantity of data including a framing error in the received data stored in SCFRDR. The value indicated by bits 11 to 8 represents the number of framing errors in SCFRDR.

19.2.8 Bit Rate Register (SCBRR)

The bit rate register (SCBRR) is an eight-bit register that, together with the baud rate generator clock source selected by the CKS1 and CKS0 bits in the serial mode register (SCSMR), determines the serial transmit/receive bit rate.

The CPU can always read and write the SCBRR. The SCBRR is initialized to H'FF by a reset or in module standby or standby mode. Each channel has independent baud rate generator control, so different values can be set in two channels.

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The SCBRR setting is calculated as follows:

Asynchronous mode:

$$N = \frac{P\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

B: Bit rate (bit/s)
N: SCBRR setting for baud rate generator ($0 \leq N \leq 255$)
Pϕ: Operating frequency for peripheral modules (MHz)
n: Baud rate generator clock source (n = 0, 1, 2, 3) (for the clock sources and values of n, see table 19.3.)

Table 19.3SCSMR Settings

		SCSMR Settings	
n	Clock Source	CKS1	CKS0
0	Pϕ	0	0
1	Pϕ/4	0	1
2	Pϕ/16	1	0
3	Pϕ/64	1	1

Note: Find the bit rate error by the following formula:

$$\text{Error (\%)} = \left\{ \frac{P\text{ø}}{(N+1) \times 64 \times 2^{2n-1} \times B} \times 10^{6} - 1 \right\} \times 100$$

Table 19.4 lists examples of SCBRR settings.

Table 19.4Bit Rates and SCBRR Settings

	Pϕ (MHz)								
	2			2.097152			2.4576		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	141	0.03	1	148	–0.04	1	174	–0.26
150	1	103	0.16	1	108	0.21	1	127	0.00
300	0	207	0.16	0	217	0.21	0	255	0.00
600	0	103	0.16	0	108	0.21	0	127	0.00
1200	0	51	0.16	0	54	–0.70	0	63	0.00
2400	0	25	0.16	0	26	1.14	0	31	0.00
4800	0	12	0.16	0	13	–2.48	0	15	0.00
9600	0	6	–6.99	0	6	–2.48	0	7	0.00
19200	0	2	8.51	0	2	13.78	0	3	0.00
31250	0	1	0.00	0	1	4.86	0	1	22.88
38400	0	1	–18.62	0	0	–14.67	0	1	0.00

	Pϕ (MHz)								
	3			3.6864			4		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	212	0.03	2	64	0.70	2	70	0.03
150	1	155	0.16	1	191	0.00	1	207	0.16
300	1	77	0.16	1	95	0.00	1	103	0.16
600	0	155	0.16	0	191	0.00	0	207	0.16
1200	0	77	0.16	0	95	0.00	0	103	0.16
2400	0	38	0.16	0	47	0.00	0	51	0.16
4800	0	19	–2.34	0	23	0.00	0	25	0.16
9600	0	9	–2.34	0	11	0.00	0	12	0.16
19200	0	4	–2.34	0	5	0.00	0	6	–6.99
31250	0	2	0.00	0	3	–7.84	0	3	0.00
38400	—	—	—	0	2	0.00	0	2	8.51

Table 19.4Bit Rates and SCBRR Settings (cont)

	Pϕ (MHz)								
	4.9152			5			6		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	86	0.31	2	88	–0.25	2	106	–0.44
150	1	255	0.00	2	64	0.16	2	77	0.16
300	1	127	0.00	1	129	0.16	1	155	0.16
600	0	255	0.00	1	64	0.16	1	77	0.16
1200	0	127	0.00	0	129	0.16	0	155	0.16
2400	0	63	0.00	0	64	0.16	0	77	0.16
4800	0	31	0.00	0	32	–1.36	0	38	0.16
9600	0	15	0.00	0	15	1.73	0	19	–2.34
19200	0	7	0.00	0	7	1.73	0	9	–2.34
31250	0	4	–1.70	0	4	0.00	0	5	0.00
38400	0	3	0.00	0	3	1.73	0	4	–2.34

	Pϕ (MHz)								
	6.144			7.3728			8		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	108	0.08	2	130	–0.07	2	141	0.03
150	2	79	0.00	2	95	0.00	2	103	0.16
300	1	159	0.00	1	191	0.00	1	207	0.16
600	1	79	0.00	1	95	0.00	1	103	0.16
1200	0	159	0.00	0	191	0.00	0	207	0.16
2400	0	79	0.00	0	95	0.00	0	103	0.16
4800	0	39	0.00	0	47	0.00	0	51	0.16
9600	0	19	0.00	0	23	0.00	0	25	0.16
19200	0	9	0.00	0	11	0.00	0	12	0.16
31250	0	5	2.40	0	6	5.33	0	7	0.00
38400	0	4	0.00	0	5	0.00	0	6	–6.99

Table 19.4Bit Rates and SCBRR Settings (cont)

	Pϕ (MHz)											
	9.8304			10			12			12.288		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	1	174	–0.26	2	177	–0.25	1	212	0.03	2	217	0.08
150	1	127	0.00	2	129	0.16	1	155	0.16	2	159	0.00
300	0	255	0.00	2	64	0.16	1	77	0.16	2	79	0.00
600	0	127	0.00	1	129	0.16	0	155	0.16	1	159	0.00
1200	0	255	0.00	1	64	0.16	0	77	0.16	1	79	0.00
2400	0	127	0.00	0	129	0.16	0	38	0.16	0	159	0.00
4800	0	63	0.00	0	64	0.16	0	19	0.16	0	79	0.00
9600	0	31	0.00	0	32	–1.36	0	9	0.16	0	39	0.00
19200	0	15	0.00	0	15	1.73	0	4	0.16	0	19	0.00
31250	0	9	–1.70	0	9	0.00	0	2	0.00	0	11	2.40
38400	0	1	0.00	0	7	1.73	0	9	–2.34	0	9	0.00

	Pφ (MHz)											
	14.7456			16			19.6608			20		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	64	0.70	3	70	0.03	3	86	0.31	3	88	–0.25
150	2	191	0.00	2	207	0.16	2	255	0.00	2	64	0.16
300	2	95	0.00	2	103	0.16	2	127	0.00	2	129	0.16
600	1	191	0.00	1	207	0.16	1	255	0.00	1	64	0.16
1200	1	95	0.00	1	103	0.16	1	127	0.00	1	129	0.16
2400	0	191	0.00	0	207	0.16	0	255	0.00	0	64	0.16
4800	0	95	0.00	0	103	0.16	0	127	0.00	0	129	0.16
9600	0	47	0.00	0	51	0.16	0	63	0.00	0	64	0.16
19200	0	23	0.00	0	25	0.16	0	31	0.00	0	32	–1.36
31250	0	14	–1.70	0	15	0.00	0	19	–1.70	0	19	0.00
38400	0	11	0.00	0	12	0.16	0	15	0.00	0	15	1.73
115200	0	3	0.00	0	3	8.51	0	4	6.67	0	4	8.51
500000	0	0	–7.84	0	0	0.00	0	0	22.9	0	0	25.0

Table 19.4Bit Rates and SCBRR Settings (cont)

	Pϕ (MHz)											
	24			24.576			28.7			30		
Bit Rate (bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	106	–0.44	3	108	0.08	3	126	0.31	3	132	0.13
150	3	77	0.16	3	79	0.00	3	92	0.46	3	97	–0.35
300	2	155	0.16	2	159	0.00	2	186	–0.08	2	194	0.16
600	2	77	0.16	2	79	0.00	2	92	0.46	2	97	–0.35
1200	1	155	0.16	1	159	0.00	1	186	–0.08	1	194	0.16
2400	1	77	0.16	1	79	0.00	1	92	0.46	1	97	–0.35
4800	0	155	0.16	0	159	0.00	0	186	–0.08	0	194	−1.36
9600	0	77	0.16	0	79	0.00	0	92	0.46	0	97	–0.35
19200	0	38	0.16	0	39	0.00	0	46	–0.61	0	48	–0.35
31250	0	23	0.00	0	24	−1.70	0	28	−1.03	0	29	0.00
38400	0	19	−2.34	0	19	0.00	0	22	1.55	0	23	1.73
115200	0	6	–6.99	0	6	−4.76	0	7	−2.68	0	7	1.73
500000	0	1	–25.0	0	1	–23.2	0	1	−10.3	0	1	–6.25

Table 19.5 indicates the maximum bit rates in the asynchronous mode when the baud rate generator is being used. Table 19.6 list the maximum rates for external clock input.

Table 19.5Maximum Bit Rates for Various Frequencies with Baud Rate Generator (Asynchronous Mode)

		Settings	
Pϕ (MHz)	Maximum Bit Rate (bits/s)	n	N
2	62500	0	0
2.097152	65536	0	0
2.4576	76800	0	0
3	93750	0	0
3.6864	115200	0	0
4	125000	0	0
4.9152	153600	0	0
8	250000	0	0
9.8304	307200	0	0
12	375000	0	0
14.7456	460800	0	0
16	500000	0	0
19.6608	614400	0	0
20	625000	0	0
24	750000	0	0
24.576	768000	0	0
28.7	896875	0	0
30	937500	0	0

Table 19.6Maximum Bit Rates during External Clock Input (Asynchronous Mode)

Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bits/s)
2	0.5000	31250
2.097152	0.5243	32768
2.4576	0.6144	38400
3	0.7500	46875
3.6864	0.9216	57600
4	1.0000	62500
4.9152	1.2288	76800
8	2.0000	125000
9.8304	2.4576	153600
12	3.0000	187500
14.7456	3.6864	230400
16	4.0000	250000
19.6608	4.9152	307200
20	5.0000	312500
24	6.0000	375000
24.576	6.1440	384000
28.7	7.1750	448436
30	7.5000	468750

19.2.9 FIFO Control Register (SCFCR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	RTRG1	RTRG0	TTRG1	TTRG0	MCE	TFRST	RFRST	LOOP
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The FIFO control register (SCFCR) resets the number of data in the transmit and receive FIFO registers, sets the number of trigger data, and contains the permit bit for the loop back test. The SCFCR is always read and written by the CPU. It is initialized to H'00 by the reset, the module standby function, or in the standby mode.

Bits 7 and 6—Trigger of the Number of Receive FIFO Data (RTRG1 and RTRG0): Set the number of receive data which sets the receive data full (RDF) flag in the serial status register (SCSSR). These bits set the RDF flag when the number of receive data stored in the receive FIFO register (SCFRDR) is increased more than the number of setting triggers listed below.

Bit 7: RTRG1	Bit 6: RTRG0	Number of Received Triggers
0	0	1*
0	1	4
1	0	8
1	1	14

Note: Initial state.

Bits 5 and 4—Trigger of the Number of Transmit FIFO Data (TTRG1 and TTRG0): Set the number of remaining transmit data which sets the transmit FIFO data register empty (TDFE) flag in the serial status register (SCSSR). These bits set the TDFE flag when the number of transmit data in the transmit FIFO data register (SCFTDR) is decreased less than the number of setting triggers listed below.

Bit 5: TTRG1	Bit 4: TTRG0	Number of Transmitted Triggers
0	0	8 (8)*
0	1	4 (12)
1	0	2 (14)
1	1	1 (15)

Note: Initial state. Values in brackets mean the number of empty SCFTDR when a flag occurs.

Bit 3—Modem Control Enable (MCE): Enables the modem control signals CTS and RTS.

Bit 3: MCE	Description	
0	Disables the modem signal*	(Initial state)
1	Enables the modem signal	

Note: The CTS is fixed to active 0 regardless of the input value, and the RTS is also fixed to 0.

Bit 2—Transmit FIFO Data Register Reset (TFRST): Disables the transmit data in the transmit FIFO data register and resets the data to the empty state.

Bit 2: TFRST	Description	
0	Disables reset operation*	(Initial state)
1	Enables reset operation	

Note: Reset is operated in resets or the standby mode.

Bit 1—Receive FIFO Data Register Reset (RFRST): Disables the receive data in the receive FIFO data register and resets the data to the empty state.

Bit 1: RFRST	Description	
0	Disables reset operation*	(Initial state)
1	Enables reset operation	

Note: Reset is operated in resets or the standby mode.

Bit 0—Loop Back Test (LOOP): Internally connects the transmit output pin (TXD) and receive input pin (RXD) and enables the loop back test.

Bit 0: LOOP	Description	
0	Disables the loop back test	(Initial state)
1	Enables the loop back test	

19.2.10 Register of the Number of FIFO Data (SCFDR)

The SCFDR is a 16-bit register which indicates the number of data stored in the transmit FIFO data register (SCFTDR) and the receive FIFO data register (SCFRDR). It indicates the number of transmit data in the SCFTDR with the upper eight bits, and the number of receive data in the SCFRDR with the lower eight bits. The SCFDR is always read from the CPU.

Lower 8 Bits:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	R4	R3	R2	R1	R0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

The SCFDR indicates the number of receive data stored in the SCFRDR. The H'00 means no receive data, and the H'10 means that the full of receive data are stored in the SCFRDR.

Upper 8 Bits:	15	14	13	12	11	10	9	8
Bit name:	—	—	—	T4	T3	T2	T1	T0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

The SCFDR indicates the number of non-transmitted data stored in the SCFTDR. The H'00 means no transmit data, and the H'10 means that the full of transmit data are stored in the SCFTDR.

19.3 Operation

19.3.1 Overview

For serial communication, the SCIF has an asynchronous mode in which characters are synchronized individually. Refer to section 17.3.2, SCI, Operation in Asynchronous Mode. The SCIF has the 16-byte FIFO buffer for both transmit and receive, reduces an overhead of the CPU, and enables continuous high-speed communication. Moreover, it has the $\overline{\text{RTS}}$ and $\overline{\text{CTS}}$ signals as the modem control signals. The transmission format is selected in the serial mode register (SCSMR), as listed in table 19.6. The SCI clock source is selected by the combination of the CKE1 and CKE0 bits in the serial control register (SCSCR), as listed in table 19.7.

- Data length is selectable: seven or eight bits.
- Parity and multiprocessor bits are selectable. So is the stop bit length (one or two bits). The combination of the preceding selections constitutes the communication format and character length.
- In receiving, it is possible to detect framing errors (FER), parity errors (PER), receive FIFO data full, receive data ready, and breaks.
- In transmitting, it is possible to detect transmit FIFO data empty.
- The number of stored data for both the transmit and receive FIFO registers is displayed.
- An internal or external clock can be selected as the SCIF clock source.
  - When an internal clock is selected, the SCIF operates using the on-chip baud rate generator, and can output a serial clock signal with a frequency 16 times the bit rate.
  - When an external clock is selected, the external clock input must have a frequency 16 times the bit rate. (The on-chip baud rate generator is not used.)

Table 19.7Serial Mode Register Settings and SCIF Communication Formats

	SCSMR Settings					SCIF Communication Format
Mode	Bit 6 CHR	Bit 5 PE	Bit 3 STOP	Data Length	Parity Bit	Stop Bit Length
Asynchronous	0	0	0	8-bit	Not set	1 bit
			1			2 bits
		1	0		Set	1 bit
			1			2 bits
	1	0	0	7-bit	Not set	1 bit
			1			2 bits
		1	0		Set	1 bit
			1			2 bits

Table 19.8SCSMR and SCSCR Settings and SCIF Clock Source Selection

	SCSCR Settings		SCIF Transmit/Receive Clock	
Mode	Bit 1 CKE1	Bit 0 CKE0	Clock Source	SCK Pin Function
Asynchronous mode	0	0	Internal	SCIF does not use the SCK pin
		1		Outputs a clock with a frequency 16 times the bit rate
	1	0	External	Inputs a clock with frequency 16 times the bit rate
		1		

19.3.2 Serial Operation

Transmit/Receive Formats: Table 19.8 lists eight communication formats that can be selected. The format is selected by settings in the serial mode register (SCSMR).

Table 19.9Serial Communication Formats

SCSMR Bits			Serial Transmit/Receive Format and Frame Length											
CHR	**PE**	**STOP**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
0	0	0	START	8-Bit data								STOP		
0	0	1	START	8-Bit data								STOP	STOP	
0	1	0	START	8-Bit data								P	STOP	
0	1	1	START	8-Bit data								P	STOP	STOP
1	0	0	START	7-Bit data							STOP			
1	0	1	START	7-Bit data							STOP	STOP		
1	1	0	START	7-Bit data							P	STOP		
1	1	1	START	7-Bit data							P	STOP	STOP	

Notes: START: Start bit
STOP: Stop bit
P: Parity bit

Clock: An internal clock generated by the on-chip baud rate generator or an external clock input from the SCK pin can be selected as the SCIF transmit/receive clock. The clock source is selected by bits CKE1 and CKE0 in the serial control register (SCSCR) (table 19.7).

When an external clock is input at the SCK pin, it must have a frequency equal to 16 times the desired bit rate.

When the SCIF operates on an internal clock, it can output a clock signal at the SCK pin. The frequency of this output clock is 16 times the bit rate.

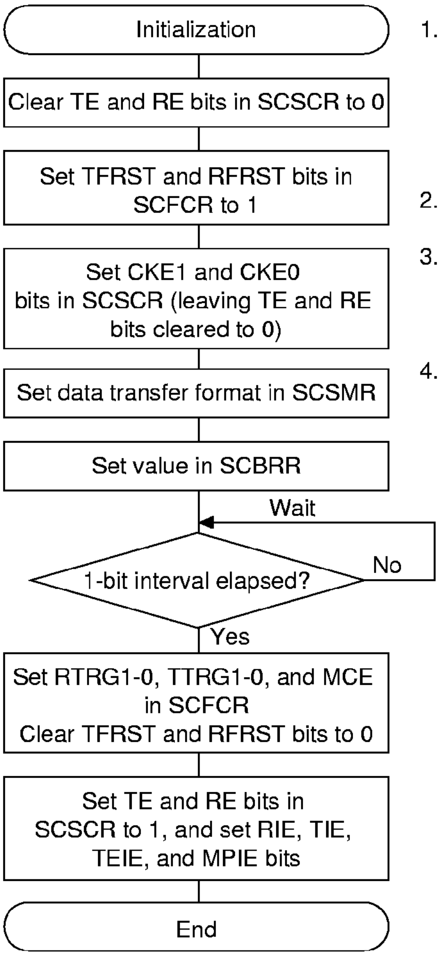
Transmitting and Receiving Data (SCIF Initialization): Before transmitting or receiving, clear the TE and RE bits to 0 in the serial control register (SCSCR), then initialize the SCIF as follows.

When changing the communication format, always clear the TE and RE bits to 0 before following the procedure given below. Clearing TE to 0 initializes the transmit shift register (SCTSR).

Clearing TE and RE to 0, however, does not initialize the serial status register (SCSSR), transmit FIFO data register (SCFTDR), or receive FIFO data register (SCFRDR), which retain their previous contents. Clear TE to 0 after all transmit data are transmitted and the TEND flag in the SCSSR is set. The transmitting data enters the high impedance state after clearing to 0 although the bit can be cleared to 0 in transmitting. Set the TFRST bit in the SCFCR to 1 and reset the SCFTDR before TE is set again to start transmission.

When an external clock is used, the clock should not be stopped during initialization or subsequent operation. SCIF operation becomes unreliable if the clock is stopped.

Figure 19.5 is a sample flowchart for initializing the SCIF. The procedure for initializing the SCIF is:


Initialization
Clear TE and RE bits in SCSCR to 0
Set TFRST and RFRST bits in SCFCR to 1
Set CKE1 and CKE0 bits in SCSCR (leaving TE and RE bits cleared to 0)
Set data transfer format in SCSMR
Set value in SCBRR
Wait
1-bit interval elapsed?
No
Yes
Set RTRG1-0, TTRG1-0, and MCE in SCFCR
Clear TFRST and RFRST bits to 0
Set TE and RE bits in SCSCR to 1, and set RIE, TIE, TEIE, and MPIE bits
End

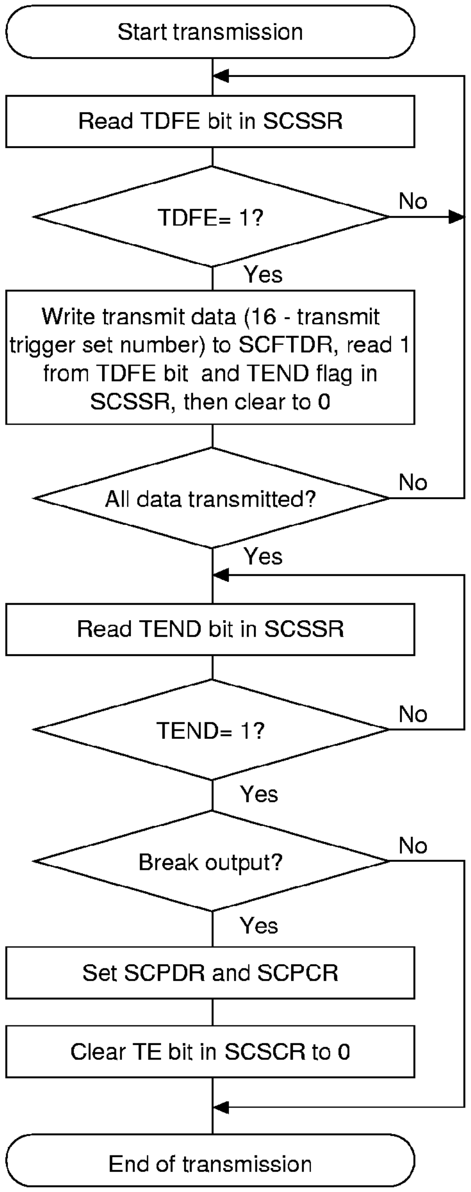
1. Set the clock selection in SCSCR.
   Be sure to clear bits RIE TIE, TE, and RE to 0.
   When clock output is selected, it is output immediately after SCSCR settings are made.
2. Set the data transfer format in SCSMR.
3. Write a value corresponding to the bit rate into the bit rate register (SCBRR).
   (Not necessary if an external clock is used.)
4. Wait at least one bit interval, then set the TE bit or RE bit in SCSCR to 1. Also set the RIE and TIE bits.
   Setting the TE and RE bits enables the TxD and RxD pins to be used. When transmitting, the SCIF will go to the mark state; when receiving, it will go to the idle state, waiting for a start bit.

Figure 19.5 Sample SCIF Initialization Flowchart

- Serial data transmission

Figure 19.6 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 serial status register (SCSSR) and check that the TDFE flag is set to 1, then write transmit data to the transmit FIFO data register (SCFTDR), read 1 from the TDFE and TEND flags, then clear these flags to 0.

The number of transmit data bytes that can be written is 16 - (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 flag 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 to 0 in the serial control register (SCSCR). For information on SCPDR and SCPCR, see section 14.2.8.

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 the FIFO data register (SCFDR).

Figure 19.6 Sample Serial Transmission Flowchart

In serial transmission, the SCIF operates as described below.

1. When data is written into the transmit FIFO data register (SCFTDR), the SCIF transfers the data from SCFTDR to the transmit shift register (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 (16 – transmit trigger setting).

2. 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 below the transmit trigger number set in the FIFO control register (SCFCR), the TDFE flag is set. If the TIE bit in the serial control register (SCSR) is set to 1 at this time, a transmit-FIFO-data-empty interrupt (TXI) request is generated.

   The serial transmit data is sent from the TxD pin in the following order.

   a. Start bit: One-bit 0 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-bit 1s (stop bits) are output.
   e. Mark state: 1 is output continuously until the start bit that starts the next transmission is sent.

3. 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 continuously.

   Figure 19.7 shows an example of the operation for transmission.

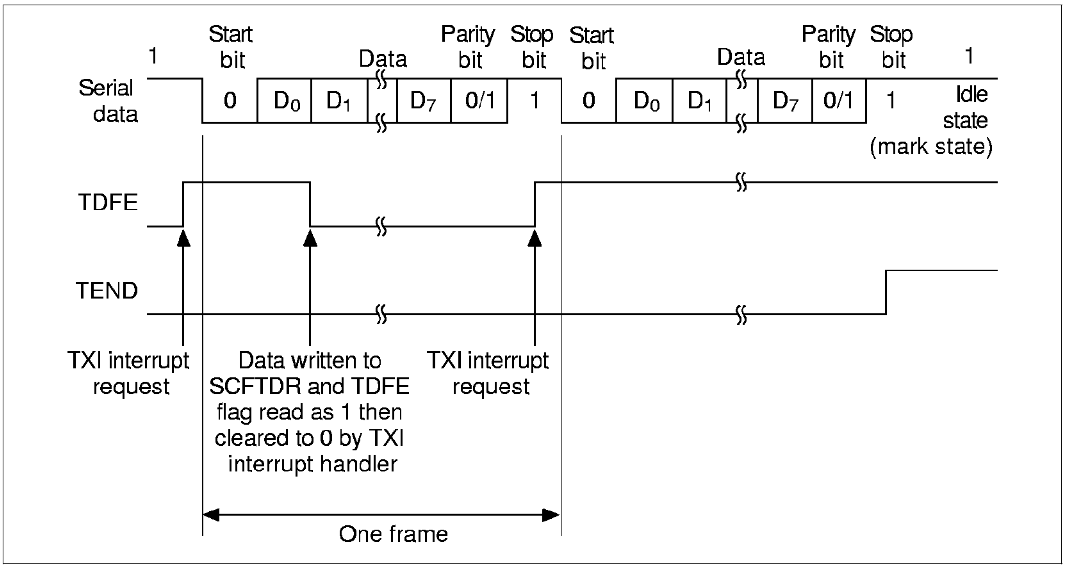


Figure 19.7 Example of Transmit Operation (Example with 8-Bit Data, Parity, One Stop Bit)

4. When modem control is enabled, transmission can be stopped and restarted in accordance with the $\overline{\text{CTS}}$ input value. When $\overline{\text{CTS}}$ is set to 1, if transmission is in progress, the line goes to the mark state after transmission of one frame. When $\overline{\text{CTS}}$ is set to 0, the next transmit data is output starting from the start bit.

   Figure 19.8 shows an example of the operation when modem control is used.

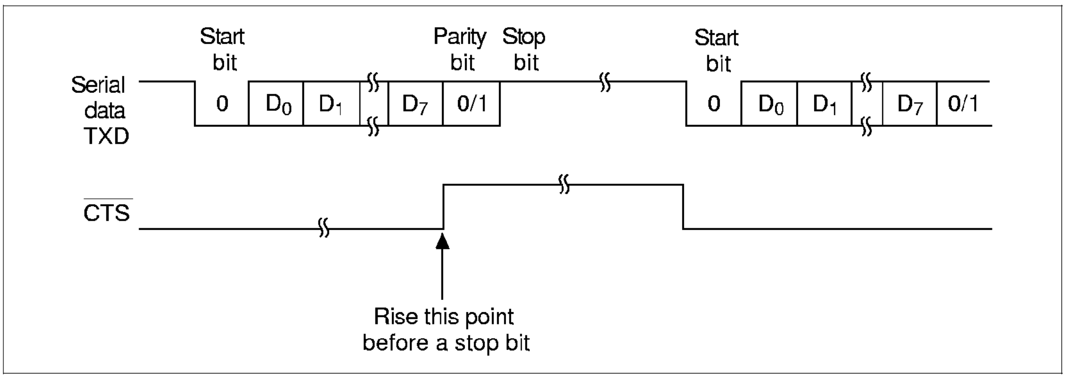
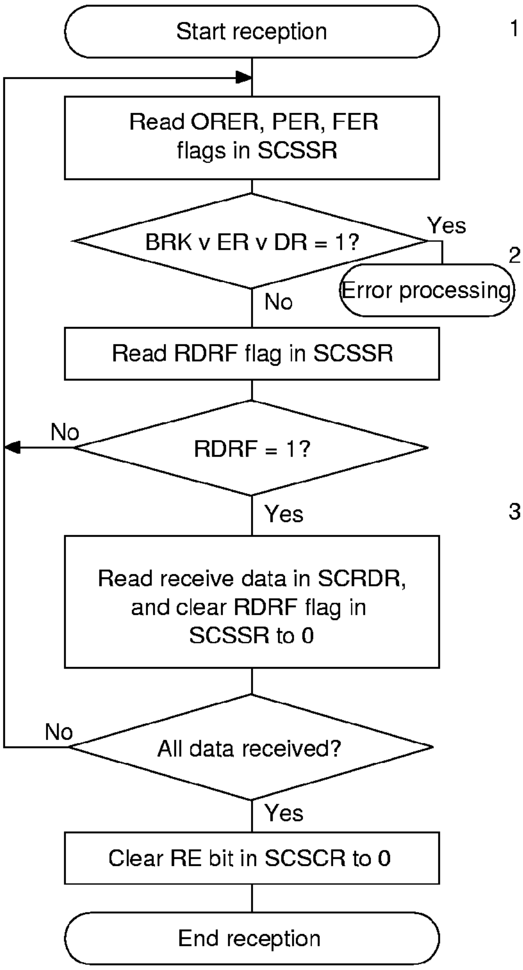


Figure 19.8 Example of Operation Using Modem Control ($\overline{\text{CTS}}$)

- Serial data reception

Figure 19.9 shows a sample flowchart for serial reception.

Use the following procedure for serial data reception after enabling the SCIF for reception.

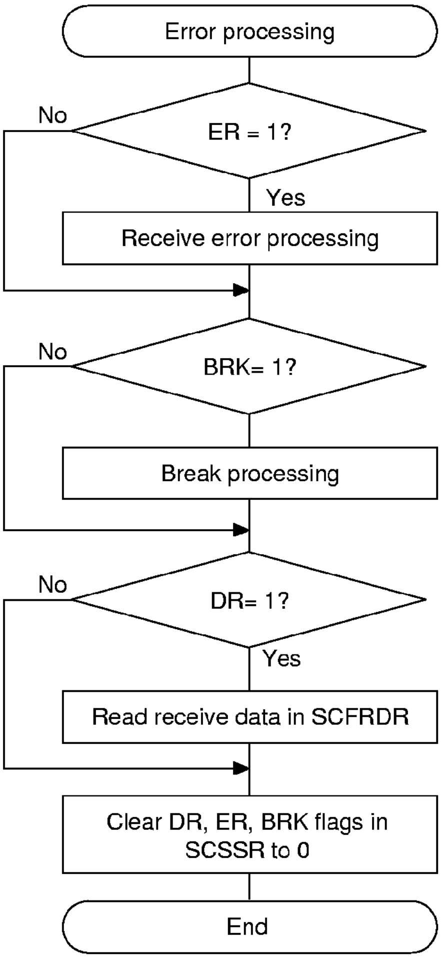


1. Receive error handling and break detection: Read the DR, ER, and BRK flags in SCSSR to identify any error, perform the appropriate error handling, then clear the DR, ER, and BRK flags to 0. In the case of a framing error, a break can also be detected by reading the value of the RxD pin.

2. SCIF status check and receive data read : Read the serial status register (SCSSR) and check that RDF = 1, then read the receive data in the receive FIFO data register (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 be identified by an RXI interrupt.

3. Serial reception continuation procedure: To continue serial reception, read at least the receive trigger set number of receive data bytes from SCFRDR, read 1 from the RDR flag, then clear the RDR flag to 0. The number of receive data bytes in SCFRDR can be ascertained by reading the lower bits of SCFDR.

Figure 19.9 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 19.10 Sample Serial Reception Flowchart (2)

In serial reception, the SCIF operates as described below.

1. The SCIF monitors the transmission line, and if a 0 start bit is detected, performs internal synchronization and starts reception.
2. The received data is stored in SCRSR in LSB-to-MSB order.
3. 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 is not suspended when a receive error occurs.

4. If the RIE bit in SCSR is set to 1 when the RDF or DR flag changes to 1, a receive-FIFO-data-full interrupt (RXI) request is generated.
If the RIE bit in SCSR is set to 1 when the ER flag changes to 1, a receive-error interrupt (ERI) request is generated.

If the RIE bit in SCSR is set to 1 when the BRK flag changes to 1, a break reception interrupt (BRI) request is generated.

Figurc 19.11 shows an cxamplc of thc opcration for rcccption.

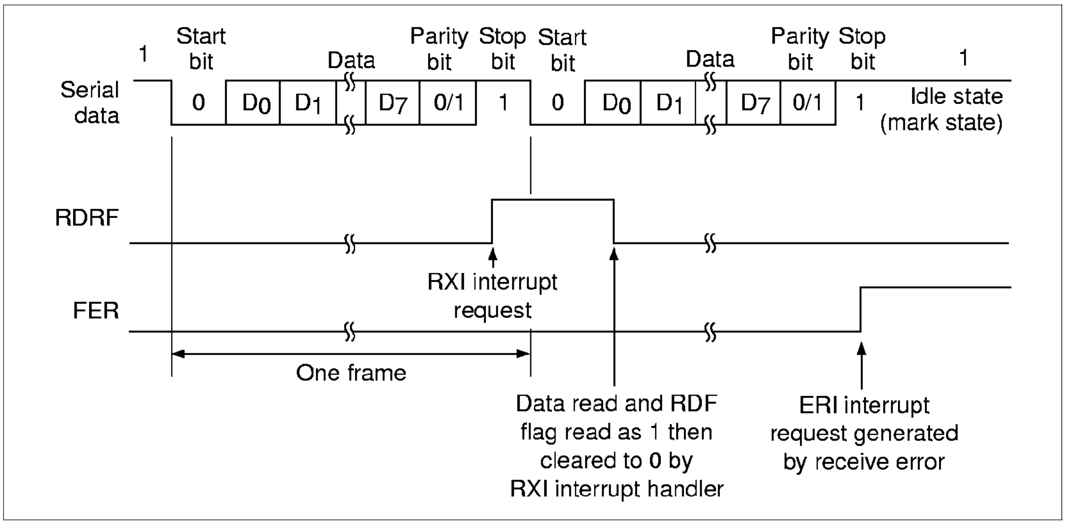


Figure 19.11 Example of SCIF Receive Operation (Example with 8-Bit Data, Parity, One Stop Bit)

5. When modem control is enabled, the $\overline{\text{RTS}}$ signal is output when SCFRDR is empty. When $\overline{\text{RTS}}$ is 0, reception is possible. When $\overline{\text{RTS}}$ is 1, this indicates that SCFRDR is full and reception is not possible.

Figure 19.12 shows an example of the operation when modem control is used.

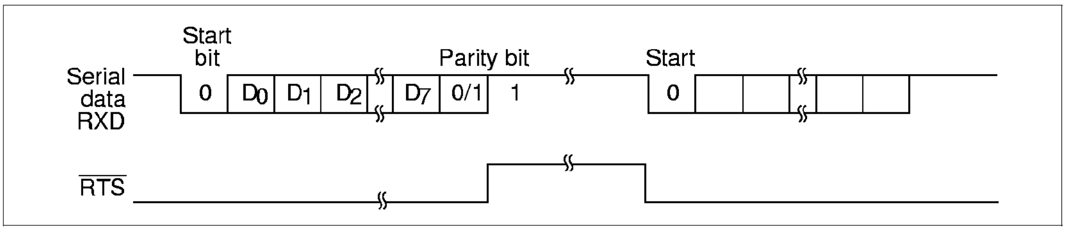


Figure 19.12 Example of Operation Using Modem Control ($\overline{\text{RTS}}$)

19.4 SCIF Interrupts

The SCIF has four interrupt sources: transmit-FIFO-data-empty (TXI), receive-error (ERI), receive-data-full (RXI), and break (BRI).

Table 19.9 shows the interrupt sources and their order of priority. The interrupt sources are enabled or disabled by means of the TIE and RIE bits in SCSCR. A separate interrupt request is sent to the interrupt controller for each of these interrupt sources.

When the TDFE flag in the serial status register (SCSSR) is set to 1, a TXI interrupt request is generated. The DMAC can be activated and data transfer performed when this interrupt is generated. The TDFE flag is automatically cleared to 0 when data is written to the transmit data register (SCFTDR) by the DMAC.

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 when the RDF flag in SCSSR is set to 1. The RDF flag is automatically cleared to 0 when data is read from the receive data register (SCFRDR) by the DMAC.

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.

The TXI interrupt indicates that transmit data can be written, and the RXI interrupt indicates that there is receive data in SCFRDR.

Table 19.9SCIF Interrupt Sources

Interrupt Source	Description	DMAC Activation	Priority on Reset Release
ERI	Interrupt initiated by receive error flag (ER)	Not possible	High
RXI	Interrupt initiated by receive data FIFO full flag (RDF) or data ready flag (DR)	Possible (RDF only)	↓
BRI	Interrupt initiated by break flag (BRK)	Not possible	
TXI	Interrupt initiated by transmit FIFO data empty flag (TDFE)	Possible	Low

See section 4, Exception Processing, for priorities and the relationship with non-SCIF interrupts.

19.5 Notes on Use

Note the following when using the SCIF.

1. 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 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 upper 8 bits of the FIFO data count register (SCFDR).
2. 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 equal to or greater than the trigger number, 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 the receive data has been read.
The number of receive data bytes in SCFRDR can be found from the lower 8 bits of the FIFO data count register (SCFDR).
3. Break Detection and Processing: Break signals can be detected by reading the RxD pin directly when a framing error (FER) is detected. In the break state the input from the RxD pin consists of all 0s, so the FER flag is set and the parity error flag (PER) may also be set. Note that,

although transfer of receive data to SCFRDR is halted in the break state, the SCIF receiver continues to operate, so if the BRK flag is cleared to 0 it will be set to 1 again.

4. Sending a Break Signal: The I/O condition and level of the TxD pin are determined by the SCP4DT bit in the port SC data register (SCPDR) and bits SCP4MD0 and SCP4MD1 in the port SC control register (SCPCR). This feature can be used to send a break signal.
   To send a break signal during serial transmission, clear the CP4DT bit to 0 (designating low level), then set the SCP4MD0 and SCP4MD1 bits to 0 and 1, respectively, and finally clear the TE bit to 0 (halting transmission). When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, and 0 is output from the TxD pin.
5. TEND Flag and TE Bit Processing: The TEND flag is set to 1 during transmission of the stop bit of the last data. Consequently, if the TE bit is cleared to 0 immediately after setting of the TEND flag has been confirmed, the stop bit will be in the process of transmission and will not be transmitted normally. Therefore, the TE bit should not be cleared to 0 for at least 0.5 serial clock cycles (or 1.5 cycles if two stop bits are used) after setting of the TEND flag setting is confirmed.
6. Receive Data Sampling Timing and Receive Margin: The SCIF operates on a base clock with a frequency of 16 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. The timing is shown in figure 19.13.

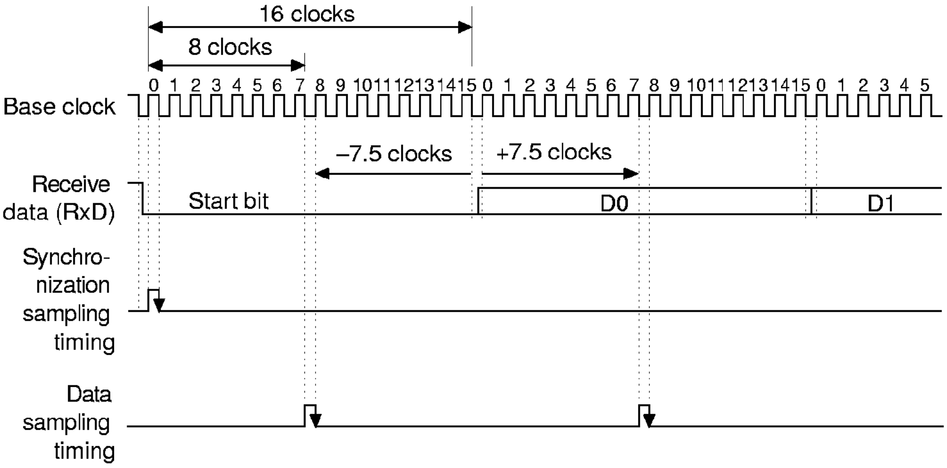


Figure 19.13 Receive Data Sampling Timing in Asynchronous Mode

The receive margin in asynchronous mode can therefore be expressed as shown in equation (1).

Equation 1:

$$M = \left| \left(0.5 - \frac{1}{2N}\right) - (L - 0.5)\,F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100\% \quad \text{....................} \quad (1)$$

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:

$$\begin{aligned} M &= (0.5 - 1/(2 \times 16)) \times 100\% \\ &= 46.875\% \quad \text{..} \quad (2) \end{aligned}$$

This is a theoretical value. A reasonable margin to allow in system designs is 20% to 30%.

Section 20 IrDA

20.1 Overview

This LSI has an on-chip infrared data association (IrDA) interface which is based on the IrDA 1.0 system and can perform infrared communication. It also can be used as the SCIF by setting registers.

20.1.1 Features

- Based on the IrDA 1.0 system
- Asynchronous serial communication
  - — Data length: Eight bits
  - — Stop bit length: One bit
  - — Parity bit: None
- On-chip 16-stage FIFO buffers for both transmit and receive
- On-chip baud rate generator with selectable bit rates
- Guard functions not to affect the receiver in transmitting
- Clock supply halted to reduce power consumption in using no IrDA

20.1.2 Block Diagram

Figure 20.1 shows a block diagram of the IrDA.

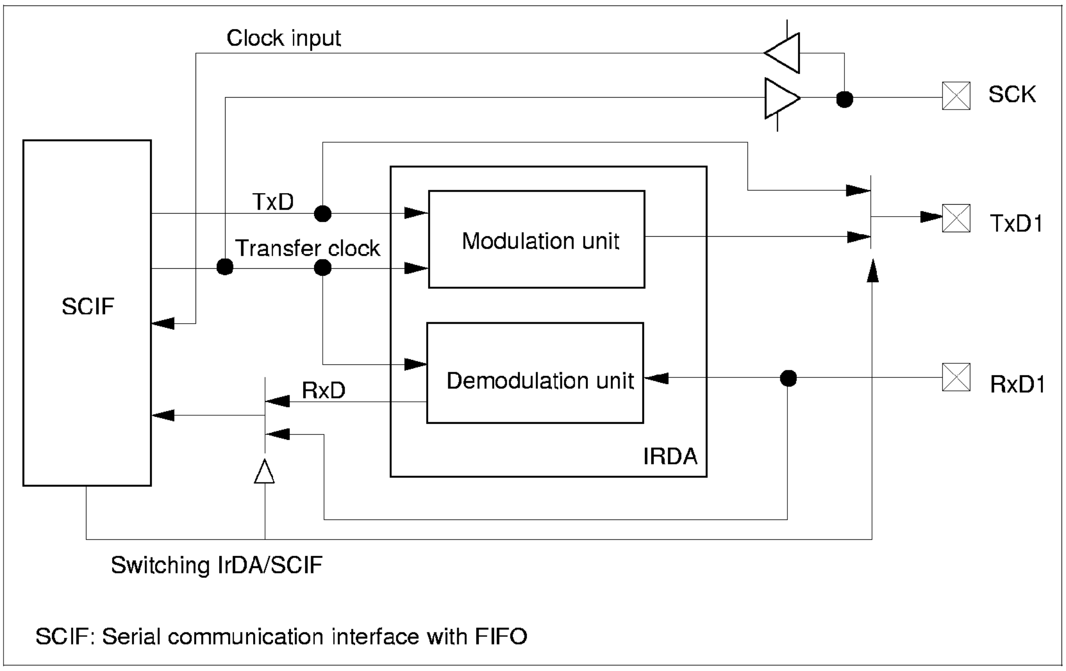
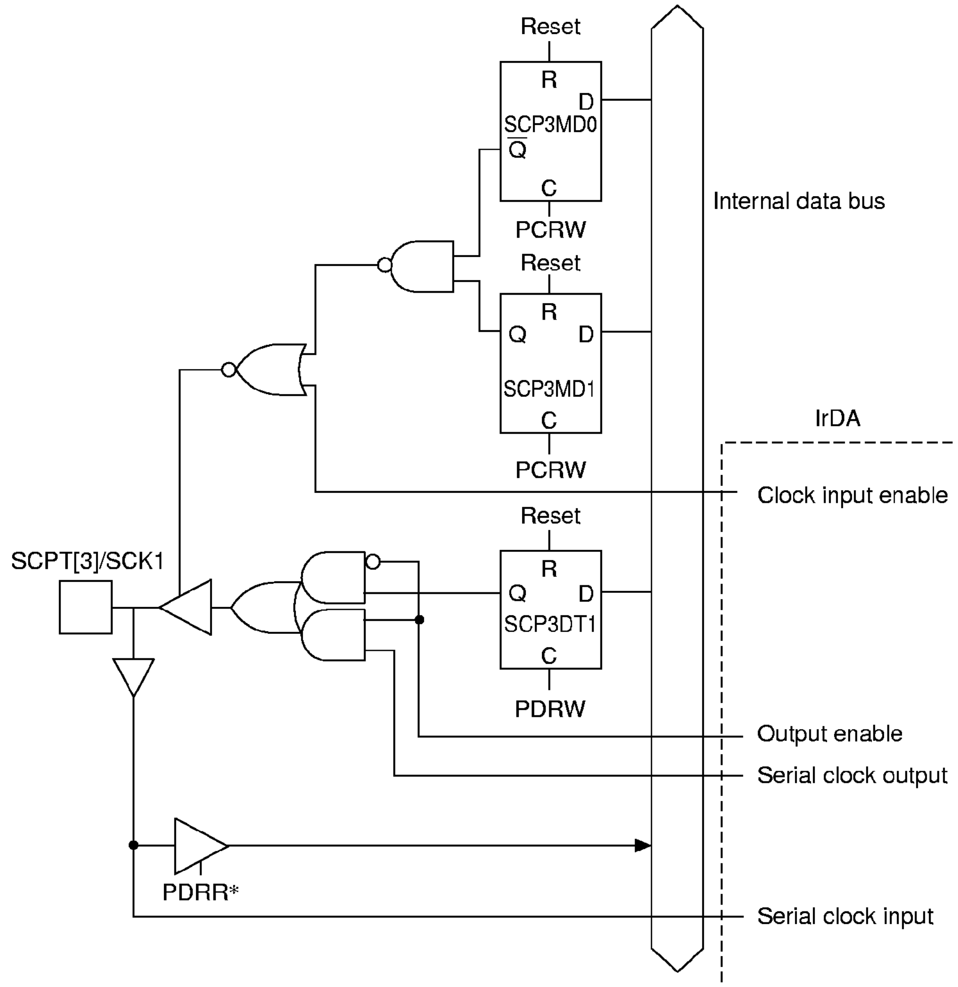


Figure 20.1 IrDA Block Diagram

Figures 20.2 to 20.4 show the IrDA I/O port pins.

SCIF pin I/O and data control is performed by bits 7–4 of SCPCR and bits 3 and 2 of SCPDR. For details, see section 17.2.8.



PDRW:	SCPDR write
PDRR:	SCPDR read
PCRW:	SCPCR write

Note: * When reading the SCK1 pin, the CKE1 and CKE0 bits in SCSCR to 0, and set the SCP3MD1 bit in SCSPR to 1 (see section 17.2.8).

Figure 20.2 SCPT[3]/SCK1 Pin

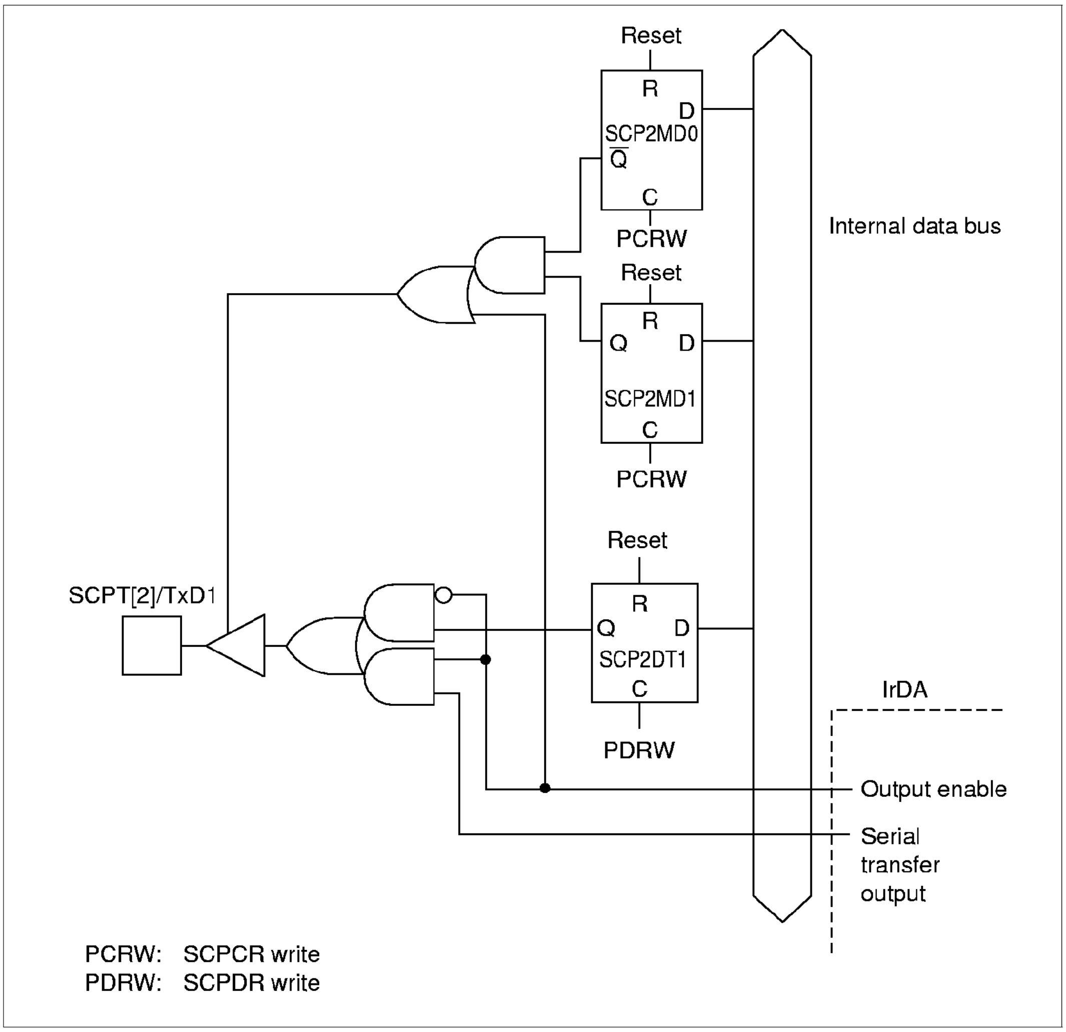
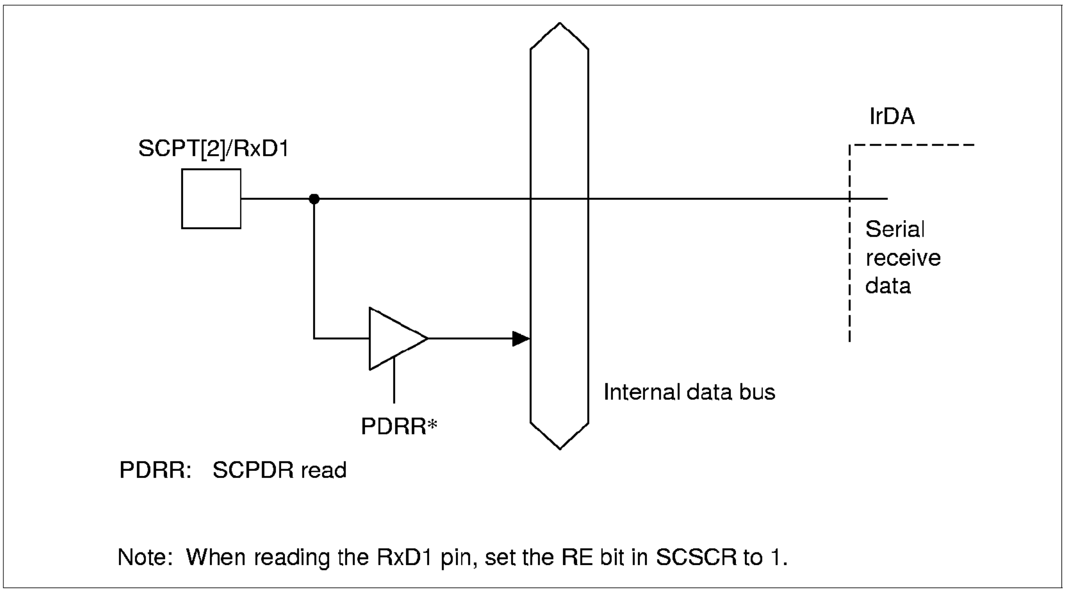


Figure 20.3 SCPT[2]/TxD1 Pin



Note: When reading the RxD1 pin, set the RE bit in SCSCR to 1.

Figure 20.4 SCPT[2]/RxD1 Pin

20.1.3 Pin Configuration

Table 20.1Pin Configuration

Pin Name	Signal Name	I/O	Function
Serial clock pin	SCK1	I/O	Clock I/O
Receive data pin	RxD1	Input	Receive data input
Transmit data pin	TxD1	Output	Transmit data output

Note: Clock input from the serial clock pin cannot be set in IrDA mode.

20.1.4 Register Configuration

The IrDA has internal registers shown in table 20.2. By using these registers, an IrDA or an SCIF mode, a data format, and a bit rate can be specified, and a transmit and a receive unit can be controlled.

Table 20.2Register Configuration

Register Name	Abbreviation	R/W	Initial Value	Address	Access Size
Serial mode register 1	SCSMR1	R/W	H'00	H'4000140	8 bits
Bit rate register 1	SCBRR1	R/W	H'FF	H'4000142	8 bits
Serial control register 1	SCSCR1	R/W	H'00	H'4000144	8 bits
Transmit FIFO data register 1	SCFTDR1	W	—	H'4000146	8 bits
Serial status register 1	SCSSR1	R/(W)*	H'0060	H'4000148	16 bits
Receive data FIFO register 1	SCFRDR1	R	Undefined	H'400014A	8 bits
FIFO control register 1	SCFCR1	R/W	H'00	H'400014C	8 bits
FIFO data count set register 1	SCFDR1	R	H'0000	H'400014E	16 bits

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

20.2 Register Description

Specifications of the registers in the IrDA are the same as those in the SCIF except for the serial mode register described below. Therefore, refer to section 19, Serial Communication Interface with FIFO, for these registers.

20.2.1 Serial Mode Register (SCSMR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	IRMOD	ICK3	ICK2	ICK1	ICK0	PSEL	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SCSMR is an 8-bit register, which can select an IrDA or an SCIF mode, specify an SCIF serial communication format, select an output pulse width of IrDA and select a baud rate generator clock source.

This module operates as IrDA by setting the IRMOD bit to 1. At this time, bits 3 to 6 are fixed to 0. This register functions in the same way as the SCSMR register in SCIF by setting the IRMOD bit to 0; therefore, this module can also operates as SCIF.

SCSMR is initialized to H'00 at power-on reset, manual reset, stop by module standby function, or standby mode.

Bit 7—IrDA Mode (IRMOD): This bit selects whether this module operates as an IrDA serial communication interface or as an SCIF.

Bit 7: IRMOD	Description	
0	Operates as an SCIF	(Initial value)
1	Operates as an IrDA	

Bits 6 to 3—Ir Cck Select Bits (ICK3–ICK0): Bit 2—Output pulse width select (PSEL)

Output pulse width select bit (PSEL) selects an output pulse width of IrDA that is 3/16 of bit length for 115 kbps or 3/16 of a bit length for selected baud rate.

Ir clock select bits should be set properly to fix an output pulse width to 3/16 of bit length for 115 kbps by setting PSEL bit to 1.

Bit 6: ICK3	Bit 5: ICK2	Bit 4: ICK1	Bit 3: ICK0	Bit 2: PSEL	Description
ICK3	ICK2	ICK1	ICK0	1	Pulse width: 3/16 of 115 kbps bit length
Don't care	Don't care	Don't care	Don't care	0	Pulse width: 3/16 of a bit length

It is necessary to generate a fixed clock pulse, IRCLK, by dividing Pϕ clock to 1/2N + 2 with a value N determined by setting of ICK3–ICK0.

Example;

Pϕ clock: 14.7456 MHz

IRCLK: 921.6 kHz (fixed)

N: setting of ICK3–ICK0 ($0 \le N \le 15$)

$$N \geq \frac{P\phi}{2XIRCLK} - 1 \geq 7$$

Accordingly, N is 7.

Bits 1 and 0—Clock Select 1 and 0 (CKS1 and CKS0): This bit selects an internal baud rate generator clock source. Pϕ, Pϕ/4, Pϕ/16, or Pϕ/64 can be selected by setting the CKS1 and CKS0 bits.

Refer to section 17.2.9, Bit Rate Register, for relationships among the clock source, the bit rate register set value, and the baud rate.

Bit 1: CKS1	Bit 0: CKS0	Description	
0	0	Pϕ clock	(Initial value)
0	1	Pϕ/4 clock	
1	0	Pϕ/16	
1	1	Pϕ/64	

Note: Peripheral clock

20.3 Operation Description

The IrDA module can perform a infrared communication conforming to IrDA 1.0 by connecting a infrared transmit/receive unit. A serial communication interface unit includes a 16-stage FIFO buffer in a transmit unit and a receive unit, and therefore CPU overhead can be reduced and a high-speed communication can be successively performed. This module also supports DMAC data transfer. The IrDA module differs from the SCIF in section 19 in that it does not include modem control signals RTS and CTS.

Refer to section 19.3, Operation, for SCIF mode operation.

20.3.1 Overview

The IrDA module modifies TxD/RxD transmit/receive data waveforms to satisfy the IrDA 1.0 specification of the infrared communication.

In the IrDA 1.0 specification, communication is first performed in the rate of 9600 bps, and the communication rate is changed. However, the communication rate cannot be automatically changed in this module, and therefore, perform communication by checking the communication rate and setting the appropriate rate in this module with software.

Note: In IrDA mode, reception cannot be performed when the TE bit in the serial control register (SCSCR) is set to 1 (enabling transmission). When performing reception, clear the TE bit in SCSCR to 0.
As the SH7729's RxD1 pin is active-high in IrDA mode, a (Schmidt) inverter must be inserted when connecting an active-low IrDA module.
The RxD1 pin is active-low in SCIF mode.

20.3.2 Transmit

As for the serial output signal (UART frame) from the SCIF, its waveforms are modified and the signal is converted into the IR frame serial output signal by the IrDA module, as shown in figure 20.5.

When serial data is 0, the 3/16-bit width pulse of the IR frame is generated and output. When serial data is 1, a pulse is not output.

The infrared LED is driven with this signal that was demodulated into 3/16 width.

20.3.3 Receive

The 3/16-bit width pulse of the IR frame that was received is demodulated and converted into the UART frame, as shown in figure 20.5.

Demodulation to 0 is performed for pulse output, and demodulation to 1 is performed for no pulse output.

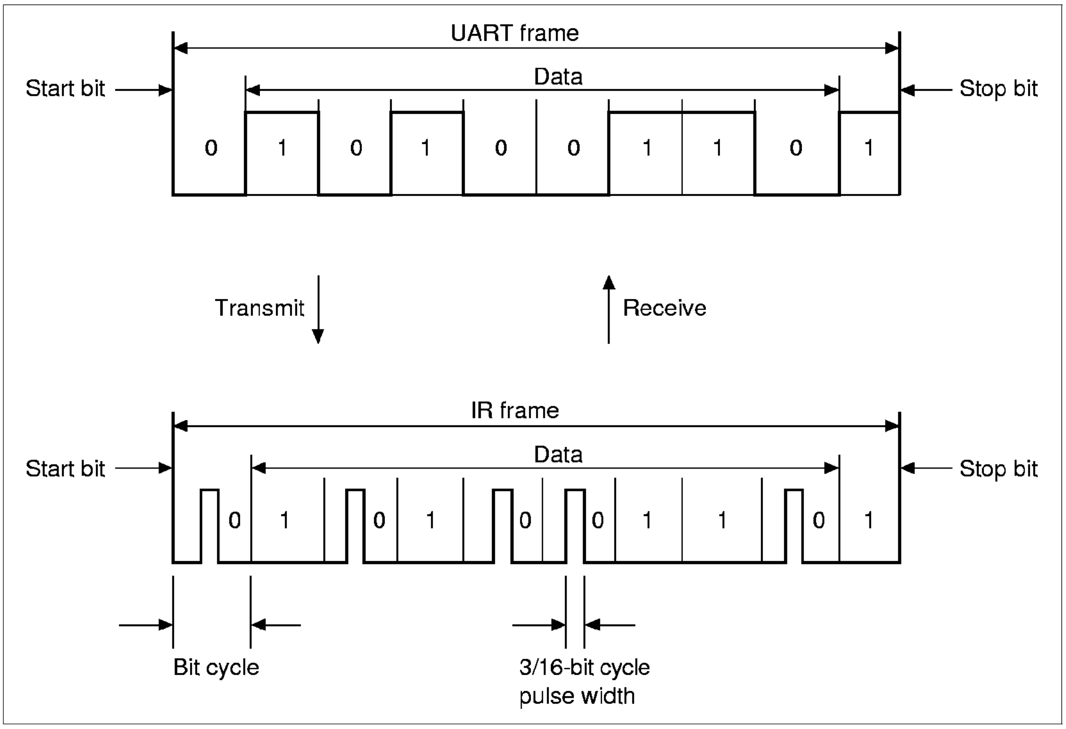


Figure 20.5 Transmit/Receive Operation

Section 21 Pin Function Controller

21.1 Overview

The pin function controller (PFC) is composed of registers for selecting the function of multiplexed pins and the direction of input/output. The pin function and I/O direction can be selected for each pin individually without regard to the operating mode of the LSI. Table 21.1 lists the multiplexed pins.

Table 21.1List of Multiplexed Pins

	Function 1	Function 2
Port	(Related Module)	(Related Module)
A	PTA7 I/O (port)	D23 I/O (data bus)
A	PTA6 I/O (port)	D22 I/O (data bus)
A	PTA5 I/O (port)	D21 I/O (data bus)
A	PTA4 I/O (port)	D20 I/O (data bus)
A	PTA3 I/O (port)	D19 I/O (data bus)
A	PTA2 I/O (port)	D18 I/O (data bus)
A	PTA1 I/O (port)	D17 I/O (data bus)
A	PTA0 I/O (port)	D16 I/O (data bus)
B	PTB7 I/O (port)	D31 I/O (data bus)
B	PTB6 I/O (port)	D30 I/O (data bus)
B	PTB5 I/O (port)	D29 I/O (data bus)
B	PTB4 I/O (port)	D28 I/O (data bus)
B	PTB3 I/O (port)	D27 I/O (data bus)
B	PTB2 I/O (port)	D26 I/O (data bus)
B	PTB1 I/O (port)	D25 I/O (data bus)
B	PTB0 I/O (port)	D24 I/O (data bus)
C	PTC7 I/O (port)/PINT7 input (INTC)	$\overline{\text{MCS7}}$ (BSC)
C	PTC6 I/O (port)/PINT6 input (INTC)	$\overline{\text{MCS6}}$ (BSC)
C	PTC5 I/O (port)/PINT5 input (INTC)	$\overline{\text{MCS5}}$ (BSC)
C	PTC4 I/O (port)/PINT4 input (INTC)	$\overline{\text{MCS4}}$ (BSC)
C	PTC3 I/O (port)/PINT3 input (INTC)	$\overline{\text{MCS3}}$ (BSC)
C	PTC2 I/O (port)/PINT2 input (INTC)	$\overline{\text{MCS2}}$ (BSC)
C	PTC1 I/O (port)/PINT1 input (INTC)	$\overline{\text{MCS1}}$ (BSC)

Table 21.1List of Multiplexed Pins (cont)

Port	Function 1 (Related Module)	Function 2 (Related Module)
C	PTC0 I/O (port)/PINT0 input (INTC)	$\overline{\text{MCS0}}$ (BSC)
D	PTD7 I/O (port)	$\overline{\text{DACK1}}$ output (DMAC)
D	PTD6 input (port)	$\overline{\text{DREQ1}}$ input (DMAC)
D	PTD5 I/O (port)	$\overline{\text{DACK0}}$ output (DMAC)
D	PTD4 input (port)	$\overline{\text{DREQ0}}$ input (DMAC)
D	PTD3 I/O (port)	$\overline{\text{WAKEUP}}$ output (WTC)
D	PTD2 I/O (port)	$\overline{\text{RESETOUT}}$
D	PTD1 I/O (port)	DRAK0 output (DMAC)
D	PTD0 I/O (port)	DRAK1 output (DMAC)
E	PTE7 I/O (port)	AUDSYNC (AUD)
E	PTE6 I/O (port)	$\overline{\text{CAS2L}}$ output (BSC)
E	PTE5 I/O (port)	$\overline{\text{CE2B}}$ output (PCMCIA)
E	PTE4 I/O (port)	$\overline{\text{CE2A}}$ output (PCMCIA)
E	PTE3 I/O (port)	$\overline{\text{CAS2H}}$ output (BSC)
E	PTE2 I/O (port)	$\overline{\text{RAS3U}}$ output (BSC)
E	PTE1 I/O (port)	$\overline{\text{RAS2U}}$ output (BSC)
E	PTE0 I/O (port)	TD0 (H-UDI)
F	PTF7 input (port)/PINT15 input (INTC)	TRST (AUD, H-UDI)
F	PTF6 input (port)/PINT14 input (INTC)	TMS (H-UDI)
F	PTF5 input (port)/PINT13 input (INTC)	TD1 (H-UDI)
F	PTF4 input (port)/PINT12 input (INTC)	TCK (H-UDI)
F	PTF3 input (port)/PINT11 input (INTC)	$\overline{\text{IRLS3}}$ (INTC)
F	PTF2 input (port)/PINT10 input (INTC)	$\overline{\text{IRLS2}}$ (INTC)
F	PTF1 input (port)/PINT9 input (INTC)	$\overline{\text{IRLS1}}$ (INTC)
F	PTF0 input (port)/PINT8 input (INTC)	$\overline{\text{IRLS0}}$ (INTC)
G	PTG7 input (port)	$\overline{\text{IOIS16}}$ input (PCMCIA)
G	PTG6 input (port)	$\overline{\text{ASEMD0}}$ (AUD, H-UDI)
G	PTG5 input (port)	$\overline{\text{ASEBRKAK}}$ (AUD)
G	PTG4 input (port)	—
G	PTG3 input (port)	AUDATA3 (AUD)
G	PTG2 input (port)	AUDATA2 (AUD)

Table 21.1List of Multiplexed Pins (cont)

Port	Function 1 (Related Module)	Function 2 (Related Module)
G	PTG1 input (port)	AUDATA1 (AUD)
G	PTG0 input (port)	AUDATA0 (AUD)
H	PTH7 I/O (port)	TCLK I/O (TMU)
H	PTH6 input (port)	AUDCK (AUD)
H	PTH5 input (port)	$\overline{\text{ADTRG}}$ input (ADC)
H	PTH4 input (port)/IRQ4 input (INTC)	IRQ4 input (INTC)
H	PTH3 input (port)/IRQ3 input (INTC)	IRQ3 input (INTC)
H	PTH2 input (port)/IRQ2 input (INTC)	IRQ2 input (INTC)
H	PTH1 input (port)/IRQ1 input (INTC)	IRQ1 input (INTC)
H	PTH0 input (port)/IRQ0 input (INTC)	IRQ0 input (INTC)
J	PTJ7 I/O (port)	STATUS1 output (CPG)
J	PTJ6 I/O (port)	STATUS0 output (CPG)
J	PTJ5 I/O (port)	$\overline{\text{CASHH}}$ output (BSC)
J	PTJ4 I/O (port)	$\overline{\text{CASHL}}$ output (BSC)
J	PTJ3 I/O (port)	$\overline{\text{CASLH}}$ output (BSC)/$\overline{\text{CASU}}$ output (BSC)
J	PTJ2 I/O (port)	$\overline{\text{CASLL}}$ output (BSC)/$\overline{\text{CASL}}$ output (BSC)
J	PTJ1 I/O (port)	$\overline{\text{RAS2L}}$ output (BSC)
J	PTJ0 I/O (port)	$\overline{\text{RAS3L}}$ output (BSC)
K	PTK7 I/O (port)	$\overline{\text{WE3}}$ output (BSC)/DQMUU output (BSC)/$\overline{\text{ICIOWR}}$ output (BSC)
K	PTK6 I/O (port)	$\overline{\text{WE2}}$ output (BSC)/DQMUL output (BSC)/$\overline{\text{ICIORD}}$ output (BSC)
K	PTK5 I/O (port)	CKE output (BSC)
K	PTK4 I/O (port)	$\overline{\text{BS}}$ output (BSC)
K	PTK3 I/O (port)	$\overline{\text{CS5}}$ output (BSC)/$\overline{\text{CE1A}}$ output (BSC)
K	PTK2 I/O (port)	$\overline{\text{CS4}}$ output (BSC)
K	PTK1 I/O (port)	$\overline{\text{CS3}}$ output (BSC)
K	PTK0 I/O (port)	$\overline{\text{CS2}}$ output (BSC)
L	PTL7 input (port)	AN7 input (ADC)/DA0 output (DAC)
L	PTL6 input (port)	AN6 input (ADC)/DA1 output (DAC)
L	PTL5 input (port)	AN5 input (ADC)

Table 21.1 List of Multiplexed Pins (cont)

Port	Function 1 (Related Module)	Function 2 (Related Module)
L	PTL4 input (port)	AN4 input (ADC)
L	PTL3 input (port)	AN3 input (ADC)
L	PTL2 input (port)	AN2 input (ADC)
L	PTL1 input (port)	AN1 input (ADC)
L	PTL0 input (port)	AN0 input (ADC)
SCPT	SCPT7 input (port)/IRQ5 input (INTC)	$\overline{\text{CTS2}}$ input (UART ch 3)/IRQ5 input (INTC)
SCPT	SCPT6 I/O (port)	$\overline{\text{RTS2}}$ input (UART ch 3)
SCPT	SCPT5 I/O (port)	SCK2 I/O (UART ch 3)
SCPT	SCPT4 input (port)	RxD2 input (UART ch 3)
	SCPT4 output (port)	TxD2 output (UART ch 3)
SCPT	SCPT3 I/O (port)	SCK1 I/O (UART ch 2)
SCPT	SCPT2 input (port)	RxD1 input (UART ch 2)
	SCPT2 output (port)	TxD1 output (UART ch 2)
SCPT	SCPT1 I/O (port)	SCK0 I/O (UART ch 1)
SCPT	SCPT0 input (port)	RxD0 input (UART ch 1)
	SCPT0 output (port)	TxD0 output (UART ch 1)

Note: SCPT0, SCPT2, and SCPT4 have the same data register to be accessed although they have different input pins and output pins.

21.2 Register Configuration

Table 21.2 summarizes the registers of the pin function controller.

Table 21.2Pin Function Controller Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port A control register	PACR	R/W	H'0000	H'4000100	16
Port B control register	PBCR	R/W	H'0000	H'4000102	16
Port C control register	PCCR	R/W	H'AAAA	H'4000104	16
Port D control register	PDCR	R/W	H'AA8A	H'4000106	16
Port E control register	PECR	R/W	H'AAAA/H'2AA8	H'4000108	16
Port F control register	PFCR	R/W	H'AAAA/H'00AA	H'400010A	16
Port G control register	PGCR	R/W	H'AAAA/H'8200	H'400010C	16
Port H control register	PHCR	R/W	H'AAAA/H'8AAA	H'400010E	16
Port J control register	PJCR	R/W	H'0000	H'4000110	16
Port K control register	PKCR	R/W	H'0000	H'4000112	16
Port L control register	PLCR	R/W	H'0000	H'4000114	16
SC port control register	SCPCR	R/W	H'A888	H'4000116	16

Note: The initial values of port E, F, G, and H control registers depend on ASEMD0 pin state.

21.3 Register Descriptions

21.3.1 Port A Control Register (PACR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PA7 MD1	PA7 MD0	PA6 MD1	PA6 MD0	PA5 MD1	PA5 MD0	PA4 MD1	PA4 MD0	PA3 MD1	PA3 MD0	PA2 MD1	PA2 MD0	PA1 MD1	PA1 MD0	PA0 MD1	PA0 MD0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port A Control Register (PACR) is a 16-bit read/write register that selects the pin functions. PACR is initialized to H'0000 by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode.

Bits 15, 14: PA7 Mode 1, 0 (PA7MD1, PA7MD0)
Bits 13, 12: PA6 Mode 1, 0 (PA6MD1, PA6MD0)
Bits 11, 10: PA5 Mode 1, 0 (PA5MD1, PA5MD0)
Bits 9, 8: PA4 Mode 1, 0 (PA4MD1, PA4MD0)
Bits 7, 6: PA3 Mode 1, 0 (PA3MD1, PA3MD0)
Bits 5, 4: PA2 Mode 1, 0 (PA2MD1, PA2MD0)
Bits 3, 2: PA1 Mode 1, 0 (PA1MD1, PA1MD0)
Bits 1, 0: PA0 Mode 1, 0 (PA0MD1, PA0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n		
PAnMD1	**PAnMD0**	**Pin Function**	
0	0	Other function	(Initial value)
0	1	Port output	
1	0	Port input (Pullup MOS: on)	
1	1	Port input (Pullup MOS: off)	

(n = 0–7)

21.3.2 Port B Control Register (PBCR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PB7 MD1	PB7 MD0	PB6 MD1	PB6 MD0	PB5 MD1	PB5 MD0	PB4 MD1	PB4 MD0	PB3 MD1	PB3 MD0	PB2 MD1	PB2 MD0	PB1 MD1	PB1 MD0	PB0 MD1	PB0 MD0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port B Control Register (PBCR) is a 16-bit read/write register that selects the pin functions. PBCR is initialized to H'0000 by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode.

Bits 15, 14: PB7 Mode 1, 0 (PB7MD1, PB7MD0)
Bits 13, 12: PB6 Mode 1, 0 (PB6MD1, PB6MD0)
Bits 11, 10: PB5 Mode 1, 0 (PB5MD1, PB5MD0)
Bits 9, 8: PB4 Mode 1, 0 (PB4MD1, PB4MD0)
Bits 7, 6: PB3 Mode 1, 0 (PB3MD1, PB3MD0)
Bits 5, 4: PB2 Mode 1, 0 (PB2MD1, PB2MD0)
Bits 3, 2: PB1 Mode 1, 0 (PB1MD1, PB1MD0)
Bits 1, 0: PB0 Mode 1, 0 (PB0MD1, PB0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	**Bit 2n**	
PBnMD1	**PBnMD0**	**Pin Function**
0	0	Other function (Initial value)
0	1	Port output
1	0	Port input (Pullup MOS: on)
1	1	Port input (Pullup MOS: off)

(n = 0–7)

21.3.3 Port C Control Register (PCCR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PC7 MD1	PC7 MD0	PC6 MD1	PC6 MD0	PC5 MD1	PC5 MD0	PC4 MD1	PC4 MD0	PC3 MD1	PC3 MD0	PC2 MD1	PC2 MD0	PC1 MD1	PC1 MD0	PC0 MD1	PC0 MD0
Initial value:	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port C Control Register (PCCR) is a 16-bit read/write register that selects the pin functions. PCCR is initialized to H'0000 by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode.

Bits 15, 14: PC7 Mode 1, 0 (PC7MD1, PC7MD0)
Bits 13, 12: PB6 Mode 1, 0 (PC6MD1, PC6MD0)
Bits 11, 10: PC5 Mode 1, 0 (PC5MD1, PC5MD0)
Bits 9, 8: PC4 Mode 1, 0 (PC4MD1, PC4MD0)
Bits 7, 6: PC3 Mode 1, 0 (PC3MD1, PC3MD0)
Bits 5, 4: PC2 Mode 1, 0 (PC2MD1, PC2MD0)
Bits 3, 2: PC1 Mode 1, 0 (PC1MD1, PC1MD0)
Bits 1, 0: PC0 Mode 1, 0 (PC0MD1, PC0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n		
PCnMD1	**PCnMD0**	**Pin Function**	
0	0	Other function	
0	1	Port output	
1	0	Port input (Pullup MOS: on)	(Initial value)
1	1	Port input (Pullup MOS: off)	

(n = 0–7)

21.3.4 Port D Control Register (PDCR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PD7 MD1	PD7 MD0	PD6 MD1	PD6 MD0	PD5 MD1	PD5 MD0	PD4 MD1	PD4 MD0	PD3 MD1	PD3 MD0	PD2 MD1	PD2 MD0	PD1 MD1	PD1 MD0	PD0 MD1	PD0 MD0
Initial value:	1	0	1	0	1	0	1	0	1	0	0	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port D Control Register (PDCR) is a 16-bit read/write register that selects the pin functions. PDCR is initialized to H'AA8A by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode.

Bits 15, 14: PD7 Mode 1, 0 (PD7MD1, PD7MD0)
Bits 11, 10: PD5 Mode 1, 0 (PD5MD1, PD5MD0)
Bits 7, 6: PD3 Mode 1, 0 (PD3MD1, PD3MD0)
Bits 5, 4: PD2 Mode 1, 0 (PD2MD1, PD2MD0)
Bits 3, 2: PD1 Mode 1, 0 (PD1MD1, PD1MD0)
Bits 1, 0: PD0 Mode 1, 0 (PD0MD1, PD0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n	
PDnMD1	**PDnMD0**	**Pin Function**
0	0	Other function (Initial value) (n = 2)
0	1	Port output
1	0	Port input (Pullup MOS: on) (Initial value) (n = 0, 1, 3, 5, 7)
1	1	Port input (Pullup MOS: off)

(n = 0–3, 5, 7)

Bits 13, 12: PD6 Mode 1, 0 (PD6MD1, PD6MD0)
Bits 9, 8: PD4 Mode 1, 0 (PD4MD1, PD4MD0)

These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n	
PDnMD1	**PDnMD0**	**Pin Function**
0	0	Other function
0	1	Reserved
1	0	Port input (Pullup MOS: on) (Initial value)
1	1	Port input (Pullup MOS: off)

(n = 4, 6)

21.3.5 Port E Control Register (PECR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PE7 MD1	PE7 MD0	PE6 MD1	PE6 MD0	PE5 MD1	PE5 MD0	PE4 MD1	PE4 MD0	PE3 MD1	PE3 MD0	PE2 MD1	PE2 MD0	PE1 MD1	PE1 MD0	PE0 MD1	PE0 MD0
Initial value:	1/0	0	1	0	1	0	1	0	1	0	1	0	1	0	1/0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port E Control Register (PECR) is a 16-bit read/write register that selects the pin functions. PECR is initialized to H'AAAA (ASEMD0 = 1) or H'2AA8 (ASEMD0 = 0) by power-on resets; however, it is not initialized by manual resets, in software standby mode, or in sleep mode.

Bits 15, 14: PE7 Mode 1, 0 (PE7MD1, PE7MD0)
Bits 13, 12: PE6 Mode 1, 0 (PE6MD1, PE6MD0)
Bits 11, 10: PE5 Mode 1, 0 (PE5MD1, PE5MD0)
Bits 9, 8: PE4 Mode 1, 0 (PE4MD1, PE4MD0)
Bits 7, 6: PE3 Mode 1, 0 (PE3MD1, PE3MD0)
Bits 5, 4: PE2 Mode 1, 0 (PE2MD1, PE2MD0)
Bits 3, 2: PE1 Mode 1, 0 (PE1MD1, PE1MD0)
Bits 1, 0: PE0 Mode 1, 0 (PE0MD1, PE0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n		
PEnMD1	**PEnMD0**	**Pin Function**	
0	0	Reserved (n = 0, 7)	(Initial value) (ASEMD0 = 0)
0	1	Port output	
1	0	Port input (Pullup MOS: on)	(Initial value) (ASEMD0 = 1)
1	1	Port input (Pullup MOS: off)	

(n = 0, 7)

Bit (2n + 1)	Bit 2n		
PEnMD1	**PEnMD0**	**Pin Function**	
0	0	Other function	
0	1	Port output	
1	0	Port input (Pullup MOS: on)	(Initial value)
1	1	Port input (Pullup MOS: off)	

(n = 1–6)

21.3.6 Port F Control Register (PFCR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PF7 MD1	PF7 MD0	PF6 MD1	PF6 MD0	PF5 MD1	PF5 MD0	PF4 MD1	PF4 MD0	PF3 MD1	PF3 MD0	PF2 MD1	PF2 MD0	PF1 MD1	PF1 MD0	PF0 MD1	PF0 MD0
Initial value:	1/0	0	1/0	0	1/0	0	1/0	0	1	0	1	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port F Control Register (PFCR) is a 16-bit read/write register that selects the pin functions. PFCR is initialized to H'AAAA (ASEMD0 = 1) or H'00AA (ASEMD0 = 0) by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode.

Bits 15, 14: PF7 Mode 1, 0 (PF7MD1, PF7MD0)
Bits 13, 12: PF6 Mode 1, 0 (PF6MD1, PF6MD0)
Bits 11, 10: PF5 Mode 1, 0 (PF5MD1, PF5MD0)
Bits 9, 8: PF4 Mode 1, 0 (PF4MD1, PF4MD0)
Bits 7, 6: PF3 Mode 1, 0 (PF3MD1, PF3MD0)
Bits 5, 4: PF2 Mode 1, 0 (PF2MD1, PF2MD0)
Bits 3, 2: PF1 Mode 1, 0 (PF1MD1, PF1MD0)
Bits 1, 0: PF0 Mode 1, 0 (PF0MD1, PF0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n		
PFnMD1	**PFnMD0**	**Pin Function**	
0	0	Other function	(Initial value) (ASEMD0 = 0)
0	1	Reserved	
1	0	Port input (Pullup MOS: on)	(Initial value) (ASEMD0 = 1)
1	1	Port input (Pullup MOS: off)	

(n = 4–7)

Bit (2n + 1)	Bit 2n		
PFnMD1	**PFnMD0**	**Pin Function**	
0	0	Other function	
0	1	Reserved	
1	0	Port input (Pullup MOS: on)	(Initial value)
1	1	Port input (Pullup MOS: off)	

(n = 0–3)

21.3.7 Port G Control Register (PGCR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PG7 MD1	PG7 MD0	PG6 MD1	PG6 MD0	PG5 MD1	PG5 MD0	PG4 MD1	PG4 MD0	PG3 MD1	PG3 MD0	PG2 MD1	PG2 MD0	PG1 MD1	PG1 MD0	PG0 MD1	PG0 MD0
Initial value:	1	0	1/0	0	1/0	0	1	0	1/0	0	1/0	0	1/0	0	1/0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port G Control Register (PGCR) is a 16-bit read/write register that selects the pin functions. PGCR is initialized to H'AAAA (ASEMD0 = 1) or H'8200 (ASEMD0 = 0) by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode.

Bits 15, 14: PG7 Mode 1, 0 (PG7MD1, PG7MD0)
Bits 13, 12: PG6 Mode 1, 0 (PG6MD1, PG6MD0)
Bits 11, 10: PG5 Mode 1, 0 (PG5MD1, PG5MD0)
Bits 9, 8: PG4 Mode 1, 0 (PG4MD1, PG4MD0)
Bits 7, 6: PG3 Mode 1, 0 (PG3MD1, PG3MD0)
Bits 5, 4: PG2 Mode 1, 0 (PG2MD1, PG2MD0)
Bits 3, 2: PG1 Mode 1, 0 (PG1MD1, PG1MD0)
Bits 1, 0: PG0 Mode 1, 0 (PG0MD1, PG0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n	
PGnMD1	**PGnMD0**	**Pin Function**
0	0	Other function (n = 7), Reserved (n = 0–6) (Initial value) (ASEMD0 = 0)
0	1	Reserved
1	0	Port input (Pullup MOS: on) (Initial value) (ASEMD0 = 1)
1	1	Port input (Pullup MOS: off)

(n = 0–3, 5, 6)

Bit (2n + 1)	Bit 2n	
PGnMD1	**PGnMD0**	**Pin Function**
0	0	Other function (n = 7), Reserved (n = 0–6)
0	1	Reserved
1	0	Port input (Pullup MOS: on) (Initial value)
1	1	Port input (Pullup MOS: off)

(n = 4, 7)

21.3.8 Port H Control Register (PHCR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PH7 MD1	PH7 MD0	PH6 MD1	PH6 MD0	PH5 MD1	PH5 MD0	PH4 MD1	PH4 MD0	PH3 MD1	PH3 MD0	PH2 MD1	PH2 MD0	PH1 MD1	PH1 MD0	PH0 MD1	PH0 MD0
Initial value:	1	0	1/0	0	1	0	1	0	1	0	1	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port H Control Register (PHCR) is a 16-bit read/write register that selects the pin functions. PHCR is initialized to H'AAAA (ASEMD0 = 1) or H'8AAA (ASEMD0 = 0) by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode.

Bits 15, 14: PH7 Mode 1, 0 (PH7MD1, PH7MD0): These bits select the pin functions and the input pullup MOS control.

Bit 15	Bit 14		
PH7MD1	**PH7MD0**	**Pin Function**	
0	0	Other function	
0	1	Port output	
1	0	Port input (Pullup MOS: on)	(Initial value)
1	1	Port input (Pullup MOS: off)	

Bits 13, 12: PH6 Mode 1, 0 (PH6MD1, PH6MD0)
Bits 11, 10: PH5 Mode 1, 0 (PH5MD1, PH5MD0)
Bits 9, 8: PH4 Mode 1, 0 (PH4MD1, PH4MD0)
Bits 7, 6: PH3 Mode 1, 0 (PH3MD1, PH3MD0)
Bits 5, 4: PH2 Mode 1, 0 (PH2MD1, PH2MD0)
Bits 3, 2: PH1 Mode 1, 0 (PH1MD1, PH1MD0)
Bits 1, 0: PH0 Mode 1, 0 (PH0MD1, PH0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n		
PH6MD1	PH6MD0	Pin Function	
0	0	Other function	(Initial value) (ASEMD0 = 0)
0	1	Reserved	
1	0	Port input (Pullup MOS: on)	(Initial value) (ASEMD0 = 1)
1	1	Port input (Pullup MOS: off)	

Bit (2n + 1)	Bit 2n		
PHnMD1	PHnMD0	Pin Function	
0	0	Other function	
0	1	Reserved	
1	0	Port input (Pullup MOS: on)	(Initial value)
1	1	Port input (Pullup MOS: off)	

(n = 0–5)

21.3.9 Port J Control Register (PJCR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PJ7 MD1	PJ7 MD0	PJ6 MD1	PJ6 MD0	PJ5 MD1	PJ5 MD0	PJ4 MD1	PJ4 MD0	PJ3 MD1	PJ3 MD0	PJ2 MD1	PJ2 MD0	PJ1 MD1	PJ1 MD0	PJ0 MD1	PJ0 MD0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port J Control Register (PJCR) is a 16-bit read/write register that selects the pin functions. PJCR is initialized to H'0000 by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode.

Bits 15, 14: PJ7 Mode 1, 0 (PJ7MD1, PJ7MD0)
Bits 13, 12: PJ6 Mode 1, 0 (PJ6MD1, PJ6MD0)
Bits 11, 10: PJ5 Mode 1, 0 (PJ5MD1, PJ5MD0)
Bits 9, 8: PJ4 Mode 1, 0 (PJ4MD1, PJ4MD0)
Bits 7, 6: PJ3 Mode 1, 0 (PJ3MD1, PJ3MD0)
Bits 5, 4: PJ2 Mode 1, 0 (PJ2MD1, PJ2MD0)
Bits 3, 2: PJ1 Mode 1, 0 (PJ1MD1, PJ1MD0)
Bits 1, 0: PJ0 Mode 1, 0 (PJ0MD1, PJ0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n		
PJnMD1	**PJnMD0**	**Pin Function**	
0	0	Other function	(Initial value)
0	1	Port output	
1	0	Port input (Pullup MOS: on)	
1	1	Port input (Pullup MOS: off)	

(n = 0–7)

21.3.10 Port K Control Register (PKCR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PK7 MD1	PK7 MD0	PK6 MD1	PK6 MD0	PK5 MD1	PK5 MD0	PK4 MD1	PK4 MD0	PK3 MD1	PK3 MD0	PK2 MD1	PK2 MD0	PK1 MD1	PK1 MD0	PK0 MD1	PK0 MD0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port K Control Register (PKCR) is a 16-bit read/write register that selects the pin functions. PKCR is initialized to H'0000 by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode.

Bits 15, 14: PK7 Mode 1, 0 (PK7MD1, PK7MD0)
Bits 13, 12: PK6 Mode 1, 0 (PK6MD1, PK6MD0)
Bits 11, 10: PK5 Mode 1, 0 (PK5MD1, PK5MD0)
Bits 9, 8: PK4 Mode 1, 0 (PK4MD1, PK4MD0)
Bits 7, 6: PK3 Mode 1, 0 (PK3MD1, PK3MD0)
Bits 5, 4: PK2 Mode 1, 0 (PK2MD1, PK2MD0)
Bits 3, 2: PK1 Mode 1, 0 (PK1MD1, PK1MD0)
Bits 1, 0: PK0 Mode 1, 0 (PK0MD1, PK0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n	
PKnMD1	**PKnMD0**	**Pin Function**
0	0	Other function (Initial value)
0	1	Port output
1	0	Port input (Pullup MOS: on)
1	1	Port input (Pullup MOS: off)

(n = 0–7)

21.3.11 Port L Control Register (PLCR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	PL7 MD1	PL7 MD0	PL6 MD1	PL6 MD0	PL5 MD1	PL5 MD0	PL4 MD1	PL4 MD0	PL3 MD1	PL3 MD0	PL2 MD1	PL2 MD0	PL1 MD1	PL1 MD0	PL0 MD1	PL0 MD0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port L Control Register (PLCR) is a 16-bit read/write register that selects the pin functions. PLCR is initialized to H'0000 by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode.

Bits 15, 14: PL7 Mode 1, 0 (PL7MD1, PL7MD0)
Bits 13, 12: PL6 Mode 1, 0 (PL6MD1, PL6MD0)
Bits 11, 10: PL5 Mode 1, 0 (PL5MD1, PL5MD0)
Bits 9, 8: PL4 Mode 1, 0 (PL4MD1, PL4MD0)
Bits 7, 6: PL3 Mode 1, 0 (PL3MD1, PL3MD0)
Bits 5, 4: PL2 Mode 1, 0 (PL2MD1, PL2MD0)
Bits 3, 2: PL1 Mode 1, 0 (PL1MD1, PL1MD0)
Bits 1, 0: PL0 Mode 1, 0 (PL0MD1, PL0MD0)
These bits select the pin functions and the input pullup MOS control.

Bit (2n + 1)	Bit 2n		
PLnMD1	**PLnMD0**	**Pin Function**	
0	0	Other function	(Initial value)
0	1	Reserved	
1	0	Port input (Pullup MOS: on)	
1	1	Port input (Pullup MOS: off)	

(n = 0–7)

When the DA0 and DA1 pins are used as the D/A converter outputs or when PTL7 and PTL6 are used as the other function states, PLCR should remain at its initial value.

21.3.12SC Port Control Register (SCPCR)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit name:	SCP7 MD1	SCP7 MD0	SCP6 MD1	SCP6 MD0	SCP5 MD1	SCP5 MD0	SCP4 MD1	SCP4 MD0	SCP3 MD1	SCP3 MD0	SCP2 MD1	SCP2 MD0	SCP1 MD1	SCP1 MD0	SCP0 MD1	SCP0 MD0
Initial value:	1	0	1	0	1	0	0	0	1	0	0	0	1	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SC Port Control Register (SCPCR) is a 16-bit read/write register that selects the pin functions. The setting of SCPCR is valid only when the transmit/receive operation is disabled in the setting of the SCSCR register. SCPCR is initialized to H'A888 by power-on resets; however, it is not initialized by manual resets, in standby mode, or in sleep mode. When the TE bit in SCSCR is set to 1, the other function output state has a higher priority than the SCPCR setting of the TxD[2:0] pin. When the RE bit in SCSCR is set to 1, the input state has a higher priority than the SCPCR setting of the RxD[2:0] pin.

Bits 15, 14: SCP7 Mode 1, 0 (SCP7MD1, SCP7MD0): These bits select the pin functions and the input pullup MOS control.

Bit 15	Bit 14		
SCP7MD1	SCP7MD0	Pin Function	
0	0	Other function	
0	1	Reserved	
1	0	Port input (Pullup MOS: on)	(Initial value)
1	1	Port input (Pullup MOS: off)	

Bits 13, 12: SCP6 Mode 1, 0 (SCP6MD1, SCP6MD0): These bits select the pin functions and the input pullup MOS control.

Bit 13	Bit 12		
SCP6MD1	SCP6MD0	Pin Function	
0	0	Other function	
0	1	Port output	
1	0	Port input (Pullup MOS: on)	(Initial value)
1	1	Port input (Pullup MOS: off)	

Bits 11, 10: SCP5 Mode 1, 0 (SCP5MD1, SCP5MD0): These bits select the pin functions and the input pullup MOS control.

Bit 11	Bit 10		
SCP5MD1	**SCP5MD0**	**Pin Function**	
0	0	Other function	
0	1	Port output	
1	0	Port input (Pullup MOS: on)	(Initial value)
1	1	Port input (Pullup MOS: off)	

Bits 9, 8: SCP4 Mode 1, 0 (SCP4MD1, SCP4MD0): These bits select the pin functions and the input pullup MOS control.

Bit 9	Bit 8		
SCP4MD1	**SCP4MD0**	**Pin Function**	
0	0	Transmit data output 2 (TxD2) Receive data input 2 (RxD2)	(Initial value)
0	1	General output (SCPT[4] output pin) Receive data input 2 (RxD2)	
1	0	SCPT[4] input pin pullup (input pin) Transmit data output 2 (TxD2)	
1	1	General input (SCPT[4] input pin) Transmit data output 2 (TxD2)	

Note: There is no combination of simultaneous I/O of SCPT[4] because one bit (SCP4DT) is accessed using two pins of TxD2 and RxD2.

When the port input is set (bit SCPnMD1 is set to 1) and when the TE bit in SCSCR is set to 1, the TxD2 pin is in the output state. When the TE bit is cleared to 0, the TxD2 pin is in the high-impedance state.

Bits 7, 6: SCP3 Mode 1, 0 (SCP3MD1, SCP3MD0): These bits select the pin functions and the input pullup MOS control.

Bit 7	Bit 6		
SCP3MD1	**SCP3MD0**	**Pin Function**	
0	0	Other function	
0	1	Port output	
1	0	Port input (Pullup MOS: on)	(Initial value)
1	1	Port input (Pullup MOS: off)	

Bits 5, 4: SCP2 Mode 1, 0 (SCP2MD1, SCP2MD0): These bits select the pin functions and the input pullup MOS control.

Bit 5	Bit 4		
SCP2MD1	**SCP2MD0**	**Pin Function**	
0	0	Transmit data output 1 (TxD1) Receive data input 1 (RxD1)	(Initial value)
0	1	General output (SCPT[2] output pin) Receive data input 1 (RxD1)	
1	0	SCPT[2] input pin pullup (input pin) Transmit data output 1 (TxD1)	
1	1	General input (SCPT[2] input pin) Transmit data output 1 (TxD1)	

Note: There is no combination of simultaneous I/O of SCPT[2] because one bit (SCP2DT) is accessed using two pins of TxD1 and RxD1.

When the port input is set (bit SCPnMD1 is set to 1) and when the TE bit in SCSCR is set to 1, the TxD1 pin is in the output state. When the TE bit is cleared to 0, the TxD1 pin is in the high-impedance state.

Bits 3, 2: SCP1 Mode 1, 0 (SCP1MD1, SCP1MD0): These bits select the pin functions and the input pullup MOS control.

Bit 3	Bit 2		
SCP1MD1	**SCP1MD0**	**Pin Function**	
0	0	Other function	
0	1	Port output	
1	0	Port input (Pullup MOS: on)	(Initial value)
1	1	Port input (Pullup MOS: off)	

Bits 1, 0: SCP0 Mode 1, 0 (SCP0MD1, SCP0MD0): These bits select the pin functions and the input pullup MOS control.

Bit 1	Bit 0	
SCP0MD1	**SCP0MD0**	**Pin Function**
0	0	Transmit data output 0 (TxD0) Receive data input 0 (RxD0) (Initial value)
0	1	General output (SCPT[0] output pin) Receive data input 0 (RxD0)
1	0	SCPT[0] input pin pullup (input pin) Transmit data output 0 (TxD0)
1	1	General input (SCPT[0] input pin) Transmit data output 0 (TxD0)

Note: There is no combination of simultaneous I/O of SCPT[0] because one bit (SCP0DT) is accessed using two pins of TxD0 and RxD0.

When the port input is set (bit SCPnMD1 is set to 1) and when the TE bit in SCSCR is set to 1, the TxD0 pin is in the output state. When the TE bit is cleared to 0, the TxD0 pin is in the high-impedance state.

Section 22 I/O Ports

22.1 Overview

This LSI has twelve 8-bit ports (ports A to L and SC). All port pins are multiplexed with other pin functions (Pin Function Controller (PFC) maintains the selection of the pin functions and pullup MOS control). Each port has a data register which stores the data to the pins.

22.2 Port A

Port A is an 8-bit I/O port with the pin configuration shown in figure 22.1. Each pin has an input pullup MOS, which is controlled by Port A Control Register (PACR) in PFC.

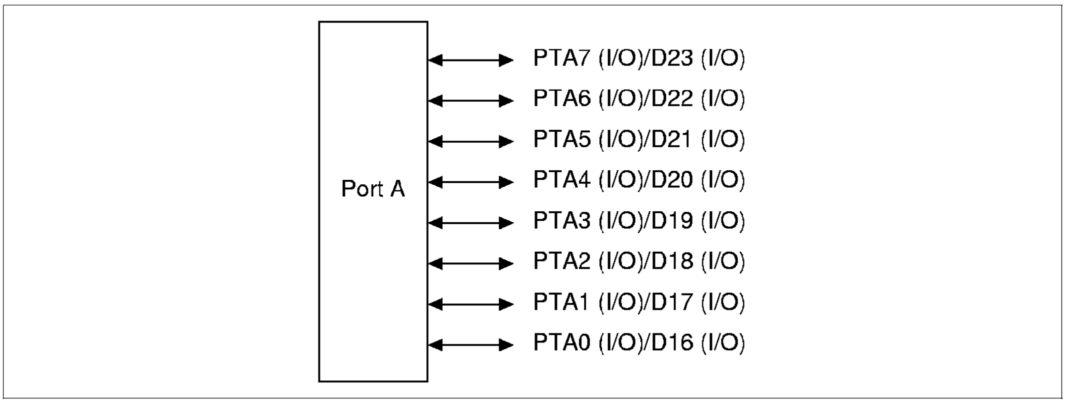


Figure 22.1 Port A

22.2.1 Register Descriptions

Table 22.1 summarizes the register of port A.

Table 22.1Register Descriptions

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port A data register	PADR	R/W	H'0000	H'4000120	8

22.2.2 Port A Data Register (PADR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PA7DT	PA6DT	PA5DT	PA4DT	PA3DT	PA2DT	PA1DT	PA0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port A data Register (PADR) is an 8-bit read/write register that stores data for pins PTA7 to PTA0. PA7DT to PA0DT bit corresponds to PTA7 to PTA0 pin. 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. Table 22.2 shows the function of PADR.

PADR is initialized to H'00 by a power-on reset. It retains its previous value in standby mode and sleep mode, and in a manual reset.

Table 22.2Read/Write Operation of the Port A Data Register (PADR)

PAnMD1	PAnMD0	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 (Pullup MOS on)	Pin state	Value is written to PADR, but does not affect pin state.
	1	Input (Pullup MOS off)	Pin state	Value is written to PADR, but does not affect pin state.

(n = 7 to 0)

22.3 Port B

Port B is an 8-bit I/O port with the pin configuration shown in figure 22.2. Each pin has an input pullup MOS, which is controlled by Port B Control Register (PBCR) in PFC.

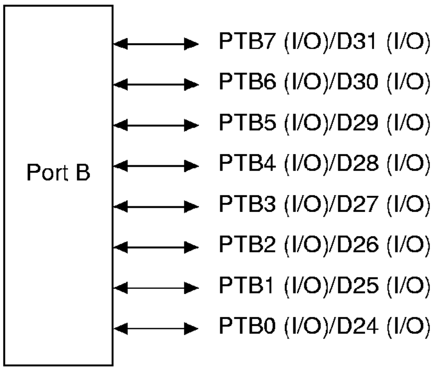


Figure 22.2 Port B

22.3.1 Register Descriptions

Table 22.3 summarizes the register of port B.

Table 22.3Register Descriptions

Name	Abbreviation	R/W	Initial	ValueAddress	Access Size
Port B data register	PBDR	R/W	H'0000	H'4000122	8

22.3.2 Port B Data Register (PBDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PB7DT	PB6DT	PB5DT	PB4DT	PB3DT	PB2DT	PB1DT	PB0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port B data register (PBDR) is an 8-bit read/write register that stores data for pins PTB7 to PTB0. PB7DT to PB0DT bit corresponds to PTB7 to PTB0 pin. 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. Table 22.4 shows the function of PBDR.

PBDR is initialized to H'00 by a power-on reset. It retains its previous value in standby mode and sleep mode, and in a manual reset.

Table 22.4 Read/Write Operation of the Port B Data Register (PBDR)

PBnMD1	PBnMD0	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 (Pullup MOS on)	Pin state	Value is written to PBDR, but does not affect pin state.
	1	Input (Pullup MOS off)	Pin state	Value is written to PBDR, but does not affect pin state.

(n = 7 to 0)

22.4 Port C

Port C is an 8-bit I/O port with the pin configuration shown in figure 22.3. Each pin has an input pullup MOS, which is controlled by Port C Control Register (PCCR) in PFC.

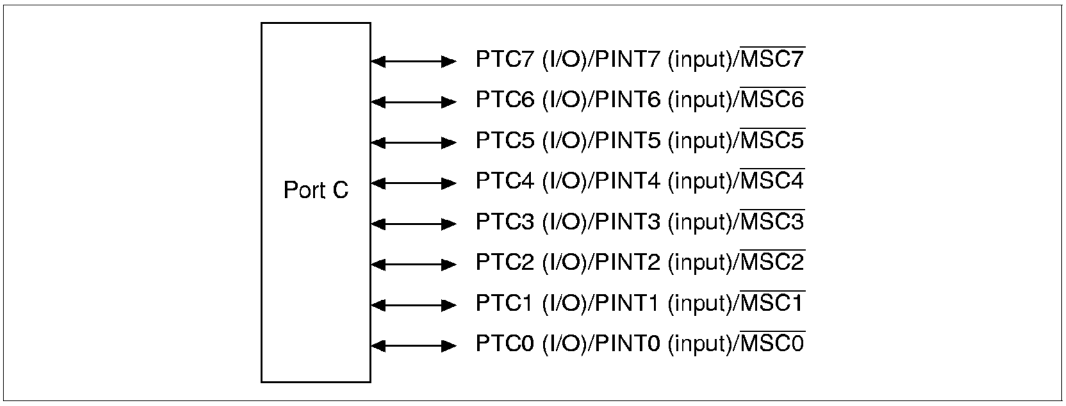

Port C
PTC7 (I/O)/PINT7 (input)/$\overline{\text{MSC7}}$
PTC6 (I/O)/PINT6 (input)/$\overline{\text{MSC6}}$
PTC5 (I/O)/PINT5 (input)/$\overline{\text{MSC5}}$
PTC4 (I/O)/PINT4 (input)/$\overline{\text{MSC4}}$
PTC3 (I/O)/PINT3 (input)/$\overline{\text{MSC3}}$
PTC2 (I/O)/PINT2 (input)/$\overline{\text{MSC2}}$
PTC1 (I/O)/PINT1 (input)/$\overline{\text{MSC1}}$
PTC0 (I/O)/PINT0 (input)/$\overline{\text{MSC0}}$

Figure 22.3 Port C

22.4.1 Register Descriptions

Table 22.5 summarizes the register of port C.

Table 22.5Register Description

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port C data register	PCDR	R/W	H'00	H'4000124	8

22.4.2 Port C Data Register (PCDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PC7DT	PC6DT	PC5DT	PC4DT	PC3DT	PC2DT	PC1DT	PC0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port C data register (PCDR) is an 8-bit read/write register that stores data for pins PTC7 to PTC0. PC7DT to PC0DT bit corresponds to PTC7 to PTC0 pin. When the pin function is general output port, if the port is read, the value of the corresponding PCDR bit is returned directly. When the function is general input port, if the port is read, the corresponding pin level is read. Table 22.6 shows the function of PCDR.

PCDR is initialized to H'00 by a power-on reset, after which the general input port function (pullup MOS on) is set as the initial pin function, and the corresponding pin levels are read.

Table 22.6Read/Write Operation of the Port C Data Register (PCDR)

PCnMD1	PCnMD0	Pin State	Read	Write
0	0	Other function	PCDR value	Value is written to PCDR, but does not affect pin state.
	1	Output	PCDR value	Write value is output from pin.
1	0	Input (Pullup MOS on)	Pin state	Value is written to PCDR, but does not affect pin state.
	1	Input (Pullup MOS off)	Pin state	Value is written to PCDR, but does not affect pin state.

(n = 7 to 0)

22.5 Port D

Port D is a 2-bit I/O and 2-bit input port with the pin configuration shown in figure 22.3. Each pin has an input pullup MOS, which is controlled by Port D Control Register (PDCR) in PFC.

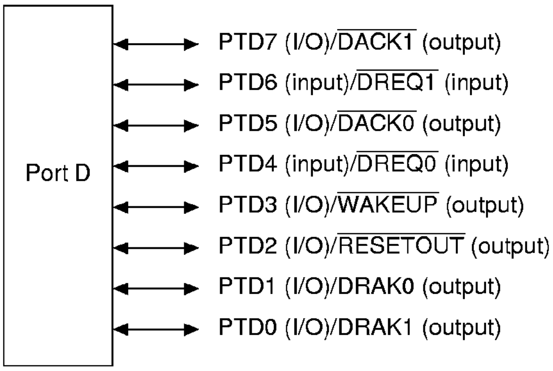

Port D
PTD7 (I/O)/DACK1 (output)
PTD6 (input)/DREQ1 (input)
PTD5 (I/O)/DACK0 (output)
PTD4 (input)/DREQ0 (input)
PTD3 (I/O)/WAKEUP (output)
PTD2 (I/O)/RESETOUT (output)
PTD1 (I/O)/DRAK0 (output)
PTD0 (I/O)/DRAK1 (output)

Figure 22.4 Port D

22.5.1 Register Descriptions

Table 22.7 summarizes the register of port D.

Table 22.7Register Descriptions

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port D data register	PDDR	R/W or R	B'0*0*0000	H'4000126	8

Note: * Means no value.

22.5.2 Port D Data Register (PDDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PD7DT	PD6DT	PD5DT	PD4DT	PD3DT	PD2DT	PD1DT	PD0DT
Initial value:	0	*	0	*	0	0	0	0
R/W:	R/W	R	R/W	R	R/W	R/W	R/W	R/W

Note: • Undefined

Port D data register (PDDR) is a 6-bit read/write and 2-bit read register that stores data for pins PTD7 to PTD0. PD7DT to PD0DT bit corresponds to PTD7 to PTD0 pin. 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. Table 22.8 shows the function of PDDR.

PDDR is initialized to B'0*0*0000 by a power-on reset. After initialization, the general input port function (pullup MOS on) is set as the initial pin function, and the corresponding pin levels are read from bits PD7DT—PD3DT, PD1DT, and PD0DT. It retains its previous value in standby mode and sleep mode, and in a manual reset.

Note that the low level is read if bits 6 and 4 are read except in general-purpose input.

Table 22.8 Read/Write Operation of the Port D Data Register (PDDR)

PDnMD1	PDnMD0	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 (Pullup MOS on)	Pin state	Value is written to PDDR, but does not affect pin state.
	1	Input (Pullup MOS off)	Pin state	Value is written to PDDR, but does not affect pin state.

(n = 0, 1, 2, 3, 5, 7)

PDnMD1	PDnMD0	Pin State	Read	Write
0	0	Other function	Low level	Ignored (no affect on pin state)
	1	Reserved	Low level	Ignored (no affect on pin state)
1	0	Input (Pullup MOS on)	Pin state	Ignored (no affect on pin state)
	1	Input (Pullup MOS off)	Pin state	Ignored (no affect on pin state)

(n = 4, 6)

Bits 3 to 0—Reserved: These bits are always read as 0, and should only be written with 0.

22.6 Port E

Port E is an 8-bit I/O port with the pin configuration shown in figure 22.5. Each pin has an input pullup MOS, which is controlled by Port E Control Register (PECR) in PFC.

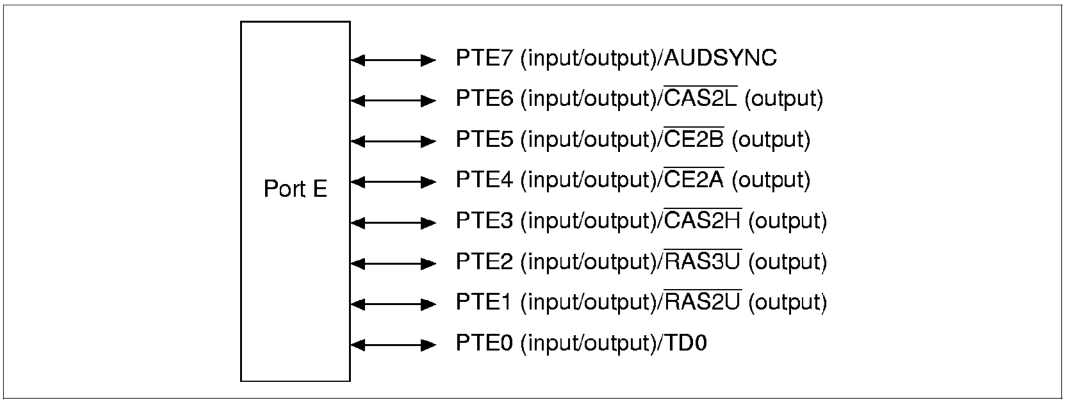

Port E
PTE7 (input/output)/AUDSYNC
PTE6 (input/output)/$\overline{\text{CAS2L}}$ (output)
PTE5 (input/output)/$\overline{\text{CE2B}}$ (output)
PTE4 (input/output)/$\overline{\text{CE2A}}$ (output)
PTE3 (input/output)/$\overline{\text{CAS2H}}$ (output)
PTE2 (input/output)/$\overline{\text{RAS3U}}$ (output)
PTE1 (input/output)/$\overline{\text{RAS2U}}$ (output)
PTE0 (input/output)/TD0

Figure 22.5 Port E

22.6.1 Register Descriptions

Table 22.9 summarizes the register of port E.

Table 22.9Register Descriptions

Name	Abbreviation	R/W	Initial	ValueAddress	Access Size
Port E data register	PEDR	R/W	H'00	H'4000128	8

22.6.2 Port E Data Register (PEDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PE7DT	PE6DT	PE5DT	PE4DT	PE3DT	PE2DT	PE1DT	PE0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port E data register (PEDR) is an 8-bit read/write register that stores data for pins PTE7 to PTE0. PE7DT to PE0DT bit corresponds to PTE7 to PTE0 pin. 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. Table 22.10 shows the function of PEDR.

PEDR is initialized to H'00 by a power-on reset, after which the general input port function (pullup MOS on) is set as the initial pin function, and the corresponding pin levels are read. It retains its previous value in standby mode and sleep mode, and in a manual reset.

Table 22.10 Read/Write Operation of the Port E Data Register (PEDR)

PEnMD1	PEnMD0	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 (Pullup MOS on)	Pin state	Value is written to PEDR, but does not affect pin state.
	1	Input (Pullup MOS off)	Pin state	Value is written to PEDR, but does not affect pin state.

(n = 0 to 7)

22.7 Port F

Port F is an 8-bit input port with the pin configuration shown in figure 22.6. Each pin has an input pullup MOS, which is controlled by Port F Control Register (PFCR) in PFC.

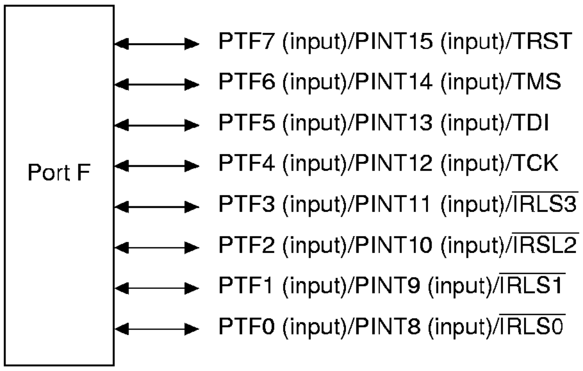

Port F
PTF7 (input)/PINT15 (input)/TRST
PTF6 (input)/PINT14 (input)/TMS
PTF5 (input)/PINT13 (input)/TDI
PTF4 (input)/PINT12 (input)/TCK
PTF3 (input)/PINT11 (input)/IRLS3
PTF2 (input)/PINT10 (input)/IRSL2
PTF1 (input)/PINT9 (input)/IRLS1
PTF0 (input)/PINT8 (input)/IRLS0

Figure 22.6 Port F

22.7.1 Register Descriptions

Table 22.11 summarizes the register of port F.

Table 22.11 Register Descriptions

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port F data register	PFDR	R	H'**	H'400012A	8

Note: * Means no value.

22.7.2 Port F Data Register (PFDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PF7DT	PF6DT	PF5DT	PF4DT	PF3DT	PF2DT	PF1DT	PF0DT
Initial value*:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Note: * Undefined

Port F data register (PFDR) is an 8-bit read register that stores data for pins PTF7 to PTF0. PF7DT to PF0DT bit corresponds to PTF7 to PTF0 pin. When the function is general input port, if the port is read the corresponding pin level is read. Table 22.12 shows the function of PFDR.

PFDR is initialized by a power-on reset, after which the general input port function (pullup MOS on) is set as the initial pin function, and the corresponding pin levels are read.

Table 22.12 Read/Write Operation of the Port F Data Register (PFDR)

PFnMD1	PFnMD0	Pin State	Read	Write
0	0	Other function	H'00	Ignored (no affect on pin state)
	1	Reserved	H'00	Ignored (no affect on pin state)
1	0	Input (Pullup MOS on)	Pin state	Ignored (no affect on pin state)
	1	Input (Pullup MOS off)	Pin state	Ignored (no affect on pin state)

(n = 0 to 7)

22.8 Port G

Port G is an 8-bit input port with the pin configuration shown in figure 22.7. Each pin has an input pullup MOS, which is controlled by Port G Control Register (PGCR) in PFC.

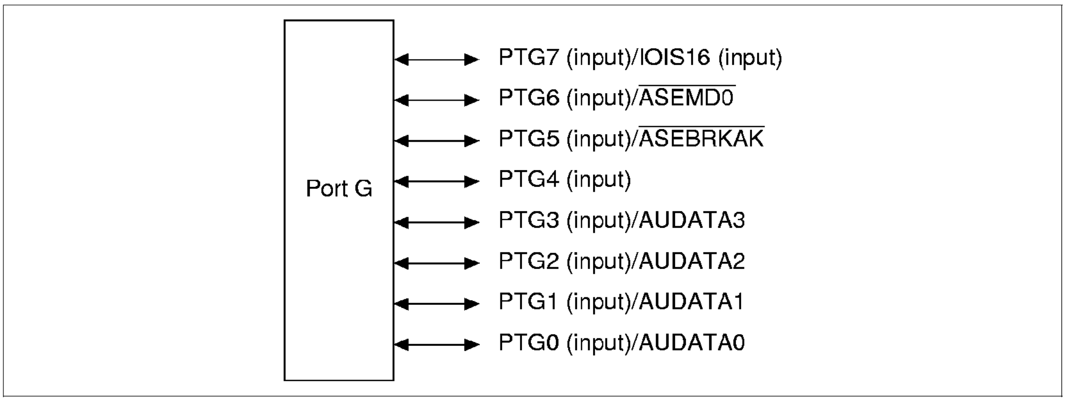

Port G
PTG7 (input)/IOIS16 (input)
PTG6 (input)/$\overline{\text{ASEMD0}}$
PTG5 (input)/$\overline{\text{ASEBRKAK}}$
PTG4 (input)
PTG3 (input)/AUDATA3
PTG2 (input)/AUDATA2
PTG1 (input)/AUDATA1
PTG0 (input)/AUDATA0

Figure 22.7 Port G

22.8.1 Register Descriptions

Table 22.13 summarizes the register of port G.

Table 22.13 Register Descriptions

Name	Abbreviation	R/W	Initial	ValueAddress	Access Size
Port G data register	PGDR	R/W	H'**	H'400012C	8

Note: * Means no value.

22.8.2 Port G Data Register (PGDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PG7DT	PG6DT	PG5DT	PG4DT	PG3DT	PG2DT	PG1DT	PG0DT
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Note: * Undefined

Port G data register (PGDR) is an 8-bit read register that stores data for pins PTG7 to PTG0. PG7DT to PG0DT bit corresponds to PTG7 to PTG0 pin. When the function is general input port, if the port is read the corresponding pin level is read. Table 22.14 shows the function of PGDR.

PGDR is initialized by a power-on reset, after which the general input port function (pullup MOS on) is set as the initial pin function, and the corresponding pin levels are read.

Table 22.14 Read/Write Operation of the Port G Data Register (PGDR)

PGnMD1	PGnMD0	Pin State	Read	Write
0	0	Other function	H'00	Ignored (no affect on pin state)
	1	Reserved	H'00	Ignored (no affect on pin state)
1	0	Input (Pullup MOS on)	Pin state	Ignored (no affect on pin state)
	1	Input (Pullup MOS off)	Pin state	Ignored (no affect on pin state)

(n = 0 to 7)

22.9 Port H

Port H is a 1-bit I/O and 5-bit input port with the pin configuration shown in figure 22.8. Each pin has an input pullup MOS, which is controlled by Port H Control Register (PHCR) in PFC.

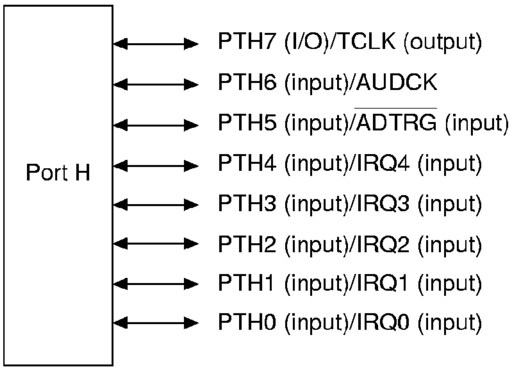

Port H
PTH7 (I/O)/TCLK (output)
PTH6 (input)/AUDCK
PTH5 (input)/$\overline{\text{ADTRG}}$ (input)
PTH4 (input)/IRQ4 (input)
PTH3 (input)/IRQ3 (input)
PTH2 (input)/IRQ2 (input)
PTH1 (input)/IRQ1 (input)
PTH0 (input)/IRQ0 (input)

Figure 22.8 Port H

22.9.1 Register Descriptions

Table 22.15 summarizes the register of port H.

Table 22.15 Register Descriptions

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port H data register	PHDR	R/W or R	B'0*******	H'400012E	8

Note: * Means no value.

22.9.2 Port H Data Register (PHDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PH7DT	PH6DT	PH5DT	PH4DT	PH3DT	PH2DT	PH1DT	PH0DT
Initial value:	0	*	*	*	*	*	*	*
R/W:	R/W	R	R	R	R	R	R	R

Note: * Undefined

Port H data register (PHDR) is a 1-bit read/write and 7-bit read register that stores data for pins PTH7 to PTH0. PH7DT to PH0DT bit corresponds to PTH7 to PTH0 pin. When the pin 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. Table 22.16 shows the function of PHDR.

PHDR is initialized to B'0******* by a power-on reset, after which the general input port function (pullup MOS on) is set as the initial pin function, and the corresponding pin levels are read. It retains its previous value in standby mode and sleep mode, and in a manual reset.

Note that the low level is read if bits 6 to 0 are read except in general-purpose input.

Table 22.16 Read/Write Operation of the Port H Data Register (PHDR)

PHnMD1	PHnMD0	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 (Pullup MOS on)	Pin state	Value is written to PHDR, but does not affect pin state.
	1	Input (Pullup MOS off)	Pin state	Value is written to PHDR, but does not affect pin state.

(n = 7)

PHnMD1	PHnMD0	Pin State	Read	Write
0	0	Other function	Low level	Ignored (no affect on pin state)
	1	Reserved	Low level	Ignored (no affect on pin state)
1	0	Input (Pullup MOS on)	Pin state	Ignored (no affect on pin state)
	1	Input (Pullup MOS off)	Pin state	Ignored (no affect on pin state)

(n = 0 to 6)

22.10 Port J

Port J is an 8-bit I/O port with the pin configuration shown in figure 22.9. Each pin has an input pullup MOS, which is controlled by Port J Control Register (PJCR) in PFC.

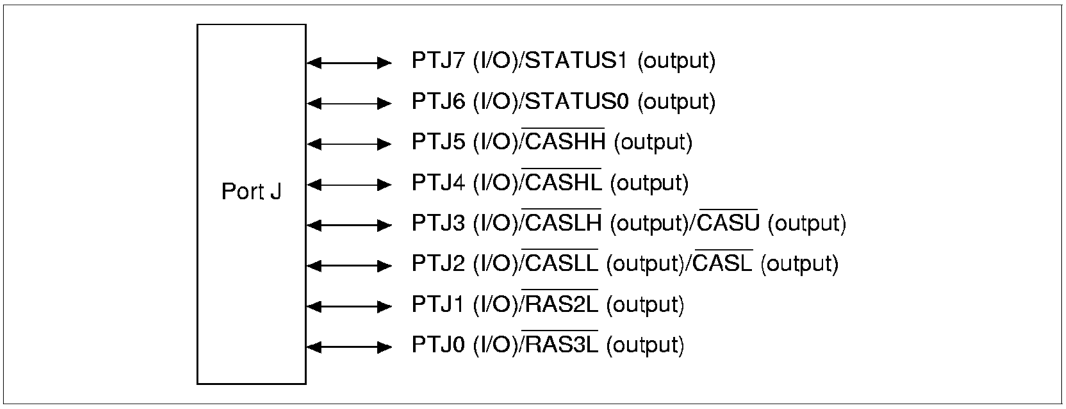

Port J
PTJ7 (I/O)/STATUS1 (output)
PTJ6 (I/O)/STATUS0 (output)
PTJ5 (I/O)/$\overline{CASHH}$ (output)
PTJ4 (I/O)/$\overline{CASHL}$ (output)
PTJ3 (I/O)/$\overline{CASLH}$ (output)/$\overline{CASU}$ (output)
PTJ2 (I/O)/$\overline{CASLL}$ (output)/$\overline{CASL}$ (output)
PTJ1 (I/O)/$\overline{RAS2L}$ (output)
PTJ0 (I/O)/$\overline{RAS3L}$ (output)

Figure 22.9 Port J

22.10.1 Register Descriptions

Table 22.17 summarizes the register of port J.

Table 22.17 Register Descriptions

Name	Abbreviation	R/W	Initial	ValueAddress	Access	Size
Port J data register	PJDR	R/W	H'00	H'4000130	8	

22.10.2 Port J Data Register (PJDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PJ7DT	PJ6DT	PJ5DT	PJ4DT	PJ3DT	PJ2DT	PJ1DT	PJ0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port J data register (PJDR) is an 8-bit read/write register that stores data for pins PTJ7 to PTJ0. PJ7DT to PJ0DT bit corresponds to PTJ7 to PTJ0 pin. When the pin function is general output port, if the port is read the value of the corresponding PJDR bit is returned directly. When the function is general input port, if the port is read, the corresponding pin level is read. Table 22.18 shows the function of PJDR.

PJDR is initialized to H'00 by a power-on reset. It retains its previous value in software standby mode and sleep mode, and in a manual reset.

Table 22.18 Read/Write Operation of the Port J Data Register (PJDR)

PJnMD1	PJnMD0	Pin State	Read	Write
0	0	Other function	PJDR value	Value is written to PJDR, but does not affect pin state.
	1	Output	PJDR value	Write value is output from pin.
1	0	Input (Pullup MOS on)	Pin state	Value is written to PJDR, but does not affect pin state.
	1	Input (Pullup MOS off)	Pin state	Value is written to PJDR, but does not affect pin state.

(n = 0 to 7)

22.11 Port K

Port K is an 8-bit I/O port with the pin configuration shown in figure 22.10. Each pin has an input pullup MOS, which is controlled by Port K Control Register (PKCR) in PFC.

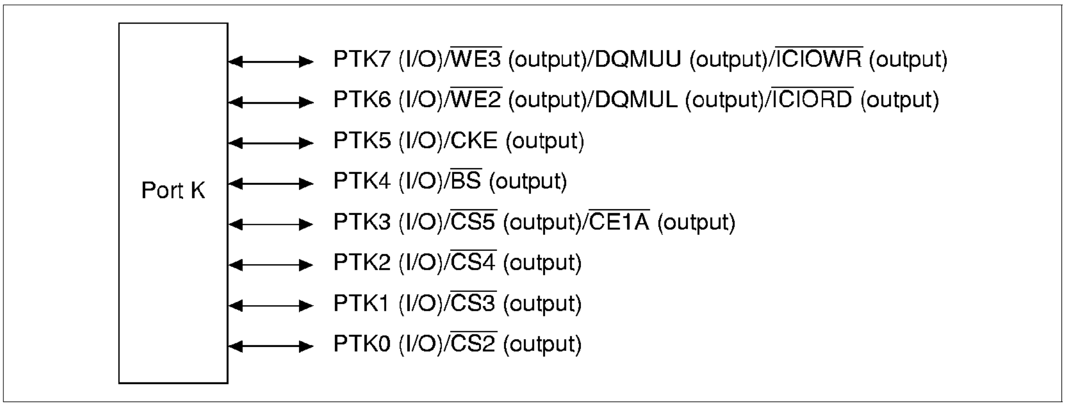

Port K
PTK7 (I/O)/$\overline{\text{WE3}}$ (output)/DQMUU (output)/$\overline{\text{ICIOWR}}$ (output)
PTK6 (I/O)/$\overline{\text{WE2}}$ (output)/DQMUL (output)/$\overline{\text{ICIORD}}$ (output)
PTK5 (I/O)/CKE (output)
PTK4 (I/O)/$\overline{\text{BS}}$ (output)
PTK3 (I/O)/$\overline{\text{CS5}}$ (output)/$\overline{\text{CE1A}}$ (output)
PTK2 (I/O)/$\overline{\text{CS4}}$ (output)
PTK1 (I/O)/$\overline{\text{CS3}}$ (output)
PTK0 (I/O)/$\overline{\text{CS2}}$ (output)

Figure 22.10 Port K

22.11.1 Register Descriptions

Table 22.19 summarizes the register of port K.

Table 22.19 Register Descriptions

Name	Abbreviation	R/W	Initial	ValueAddress	Access Size
Port K data register	PKDR	R/W	H'00	H'4000132	8

22.11.2 Port K Data Register (PKDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PK7DT	PK6DT	PK5DT	PK4DT	PK3DT	PK2DT	PK1DT	PK0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Port K data register (PKDR) is an 8-bit read/write register that stores data for pins PTK7 to PTK0. PK7DT to PK0DT bit corresponds to PTK7 to PTK0 pin. When the pin function is general output port, if the port is read, the value of the corresponding PKDR bit is returned directly. When the function is general input port, if the port is read, the corresponding pin level is read. Table 22.20 shows the function of PKDR.

PKDR is initialized to H'00 by a power-on reset. It retains its previous value in standby mode and sleep mode, and in a manual reset.

Table 22.20 Read/Write Operation of the Port K Data Register (PKDR)

PKnMD1	PKnMD0	Pin State	Read	Write
0	0	Other function	PKDR value	Value is written to PKDR, but does not affect pin state.
	1	Output	PKDR value	Write value is output from pin.
1	0	Input (Pullup MOS on)	Pin state	Value is written to PKDR, but does not affect pin state.
	1	Input (Pullup MOS off)	Pin state	Value is written to PKDR, but does not affect pin state.

(n = 0 to 7)

22.12 Port L

Port L is an 8-bit input port with the pin configuration shown in figure 22.11.

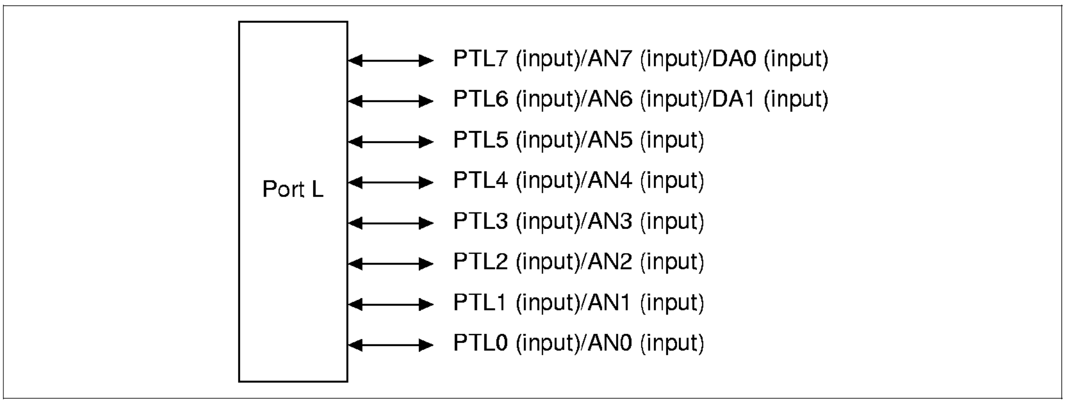


Figure 22.11 Port L

22.12.1 Register Descriptions

Table 22.21 summarizes the register of port L.

Table 22.21 Register Descriptions

Name	Abbreviation	R/W	Initial	ValueAddress	Access Size
Port L data register	PLDR	R	H'00	H'4000134	8

22.12.2 Port L Data Register (PLDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	PL7DT	PL6DT	PL5DT	PL4DT	PL3DT	PL2DT	PL1DT	PL0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Port L data register (PLDR) is an 8-bit read register that stores data for pins PTL7 to PTL0. PL7DT to PL0DT bit corresponds to PTL7 to PTL0 pin. When the function is general input port, if the port is read, the corresponding pin level is read. Table 22.22 shows the function of PLDR.

PKDR is initialized to H'00 by power-on reset. It retains its previous value in software standby mode, sleep mode and by manual reset.

Table 22.22 Read/Write Operation of the Port L Data Register (PLDR)

PLnMD1	PLnMD0	Pin State	Read	Write
0	0	Other function	H'00	Ignored (no affect on pin state)
	1	Reserved	H'00	Ignored (no affect on pin state)
1	0	Input (Pullup MOS on)	Pin state	Ignored (no affect on pin state)
	1	Input (Pullup MOS on)	Pin state	Ignored (no affect on pin state)

(n = 0 to 7)

22.13 SC Port

SC port is a 4-bit I/O, 3-bit output and 4-bit input port with the pin configuration shown in figure 22.12. Each pin has an input pullup MOS, which is controlled by SC port Control Register (SCPCR) in PFC.

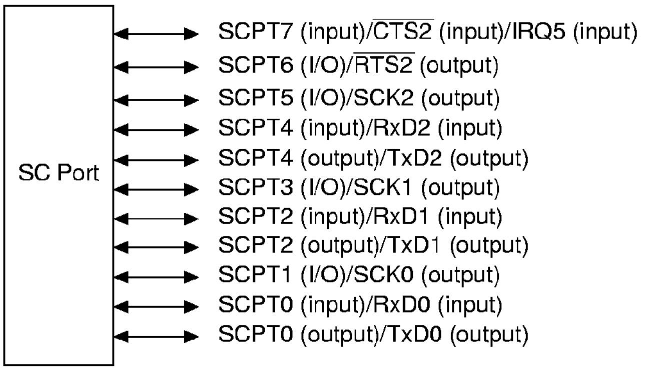

SC Port
SCPT7 (input)/$\overline{\text{CTS2}}$ (input)/IRQ5 (input)
SCPT6 (I/O)/$\overline{\text{RTS2}}$ (output)
SCPT5 (I/O)/SCK2 (output)
SCPT4 (input)/RxD2 (input)
SCPT4 (output)/TxD2 (output)
SCPT3 (I/O)/SCK1 (output)
SCPT2 (input)/RxD1 (input)
SCPT2 (output)/TxD1 (output)
SCPT1 (I/O)/SCK0 (output)
SCPT0 (input)/RxD0 (input)
SCPT0 (output)/TxD0 (output)

Figure 22.12 SC Port

22.13.1 Register Descriptions

Table 22.23 summarizes the register of SC port.

Table 22.23 Register Descriptions

Name	Abbreviation	R/W	Initial Value	Address	Access Size
SC Port data register	SCPDR	R/W or R	B'*0000000	H'4000136	8

Note: * Means no value.

22.13.2 Port SC Data Register (SCPDR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	SCP7DT	SCP6DT	SCP5DT	SCP4DT	SCP3DT	SCP2DT	SCP1DT	SCP0DT
Initial value:	*	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: * Undefined

Port SC data register (SCPDR) is a 7-bit read/write and 1-bit read register that stores data for pins SCPT7 to SCPT0. SCP7DT to SCP0DT bit corresponds to SCPT7 to SCPT0 pin. When the pin function is general output port, if the port is read, the value of the corresponding SCPDR bit is returned directly. When the function is general input port, if the port is read, the corresponding pin level is read. Table 22.24 shows the function of SCPDR.

SCPDR is initialized to B'*0000000 by a power-on reset. After initialization, the general input port function (pullup MOS on) is set as the initial pin function, and the corresponding pin levels are read from bits SCP7DT—SCP5DT, SCP3DT, and SCP1DT. SCPDR retains its previous value in standby mode and sleep mode, and in a manual reset.

Note that the low level is read if bit 7 is read except in general-purpose input.

Table 22.24 Read/Write Operation of the SC Port Data Register (SCPDR)

SCPnMD 1	SCPnMD 0	Pin State	Read	Write
0	0	Other function	SCPDR value	Value is written to SCPDR, but does not affect pin state.
	1	Output	SCPDR value	Write value is output from pin.
1	0	Input (Pullup MOS on)	Pin state	Value is written to SCPDR, but does not affect pin state.
	1	Input (Pullup MOS off)	Pin state	Value is written to SCPDR, but does not affect pin state.

(n = 0 to 6)

SCPnMD 1	SCPnMD 0	Pin State	Read	Write
0	0	Other function	Low level	Ignored (no affect on pin state)
	1	Output	Low level	Ignored (no affect on pin state)
1	0	Input (Pullup MOS on)	Pin state	Ignored (no affect on pin state)
	1	Input (Pullup MOS off)	Pin state	Ignored (no affect on pin state)

(n = 7)

Section 23 A/D Converter

23.1 Overview

This LSI includes a 10-bit successive-approximation A/D converter with a selection of up to eight analog input channels.

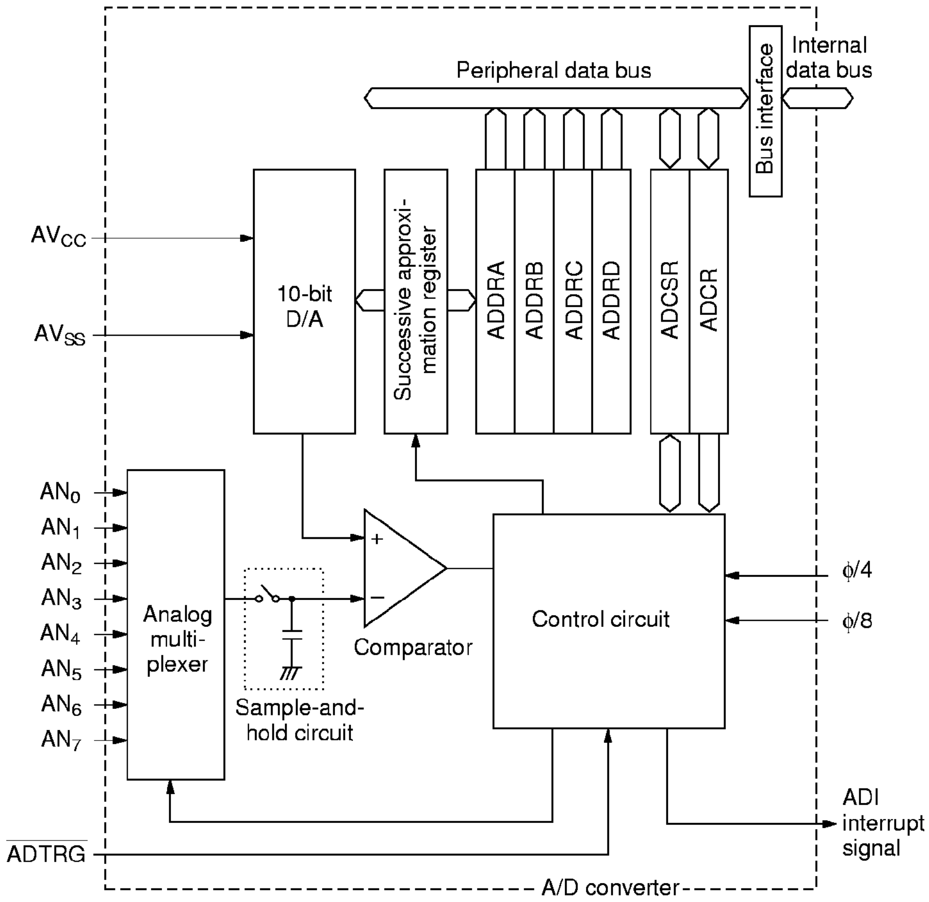
23.1.1 Features

A/D converter features are listed below.

- 10-bit resolution
- Eight input channels
- High-speed conversion
  - — Conversion time: maximum 8.9 µs per channel (with 15-MHz peripheral clock)
- Three conversion modes
  - — Single mode: A/D conversion of one channel
  - — Multi mode: A/D conversion on one to four channels
  - — Scan mode: Continuous A/D conversion on one to four channels
- Four 16-bit data registers
  - — A/D conversion results are transferred for storage into data registers corresponding to the channels.
- Sample-and-hold function
- A/D conversion can be externally triggered
- A/D interrupt requested at the end of conversion
  - — At the end of A/D conversion, an A/D end interrupt (ADI) can be requested.

23.1.2 Block Diagram

Figure 23.1 shows a block diagram of the A/D converter.



ADCR: A/D control register

ADCSR: A/D control/status register

ADDRA: A/D data register A

ADDRB: A/D data register B

ADDRC: A/D data register C

ADDRD: A/D data register D

Figure 23.1 A/D Converter Block Diagram

23.1.3 Input Pins

Table 23.1 summarizes the A/D converter's input pins. The eight analog input pins are divided into two groups: group 0 (AN_0 to AN_3), and group 1 (AN_4 to AN_7). AV_{CC} and AV_{SS} are the power supply for the analog circuits in the A/D converter. AVcc also functions as the A/D converter reference voltage.

Table 23.1 A/D Converter Pins

Pin Name	Abbreviation	I/O	Function
Analog power-supply pin	AVcc	Input	Analog power supply
Analog ground pin	AVss	Input	Analog ground and reference voltage
Analog input pin 0	AN0	Input	Group 0 analog inputs
Analog input pin 1	AN1	Input	
Analog input pin 2	AN2	Input	
Analog input pin3	AN3	Input	
Analog input pin 4	AN4	Input	Group1 analog inputs
Analog input pin 5	AN5	Input	
Analog input pin6	AN6	Input	
Analog input pin7	AN7	Input	
A/D external trigger input pin	$\overline{\text{ADTRG}}$	Input	External trigger input for starting A/D conversion

23.1.4 Register Configuration

Table 23.2 summarizes the A/D converter's registers.

Table 23.2A/D Converter Registers

Address	Name	Abbreviation	R/W	Initial Value	Access size
H'04000080	A/D data register A (high)	ADDRAH	R	H'00	16, 8
H'04000082	A/D data register A (low)	ADDRAL	R	H'00	8
H'04000084	A/D data register B (high)	ADDRBH	R	H'00	16, 8
H'04000086	A/D data register B (low)	ADDRBL	R	H'00	8
H'04000088	A/D data register C (high)	ADDRCH	R	H'00	16, 8
H'0400008A	A/D data register C (low)	ADDRCL	R	H'00	8
H'0400008C	A/D data register D (high)	ADDRDH	R	H'00	16, 8
H'0400008E	A/D data register D (low)	ADDRDL	R	H'00	8
H'04000090	A/D control/status register	ADCSR	R/(W) *	H'00	8
H'04000092	A/D control register	ADCR	R/W	H'3F	8

Note: Only 0 can be written in bit 7, to clear the flag.

23.2 Register Descriptions

23.2.1 A/D Data Registers A to D (ADDRA to ADDRD)

Bit:	15	14	13	12	11	10	9	8
ADDRn:	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
ADDRn:	AD1	AD0	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

n = A–D

The four A/D data registers (ADDRA to ADDRD) are 16-bit read-only registers that store the results of A/D conversion.

An A/D conversion produces 10-bit data, which is transferred for storage into the A/D data register corresponding to the selected channel. The upper 8 bits of the result are stored in the upper byte (bits 15 to 8) of the A/D data register. The lower 2 bits are stored in the lower byte (bits 7 and 6). Bits 5 to 0 of an A/D data register are reserved bits that always read 0. Table 23.3 indicates the pairings of analog input channels and A/D data registers.

The A/D data registers are initialized to H'0000 by a reset and in standby mode.

Table 23.3Analog Input Channels and A/D Data Registers

Analog Input Channel		
Group 0	**Group 1**	**A/D Data Register**
AN0	AN4	ADDRA
AN1	AN5	ADDRB
AN2	AN6	ADDRC
AN3	AN7	ADDRD

23.2.2 A/D Control/Status Register (ADCSR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	ADF	ADIE	ADST	MULTI	CKS	CH2	CH1	CH0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: Write 0 to clear the flag.

ADCSR is an 8-bit read/write register that selects the mode and controls the A/D converter. ADCSR is initialized to H'00 by a reset and in standby mode.

Bit 7—A/D End Flag (ADF): Indicates the end of A/D conversion.

Bit 7: ADF	Description
0	[Clear condition] (Initial value) (1) Cleared by reading ADF while ADF = 1, then writing 0 in ADF (2) Cleared when DMAC is activated by ADI interrupt and ADDR is read
1	[Set conditions] Single mode: A/D conversion ends Multi mode: A/D conversion ends in all selected channels

Bit 6—A/D Interrupt Enable (ADIE): Enables or disables the interrupt (ADI) requested at the end of A/D conversion.

Bit 6: ADIE	Description
0	A/D end interrupt request (ADI) is disabled (Initial value)
1	A/D end interrupt request (ADI) is enabled

Bit 5—A/D Start (ADST): Starts or stops A/D conversion. The ADST bit remains set to 1 during A/D conversion. It can also be set to 1 by external trigger input at the $\overline{\text{ADTRG}}$ pin.

Bit 5: ADST	Description
0	A/D conversion is stopped (Initial value)
1	Single mode: A/D conversion starts; ADST is automatically cleared to 0 when conversion ends. Scan mode: A/D conversion starts and continues, cycling among the selected channels, until ADST is cleared to 0 by software, by a reset, or by a transition to standby mode.

Bit 4—Multi Mode (MULTI): Selects single mode or multi mode. For further information on operation in these modes, see section 23.4, Operation.

Bit 4:SCAN	Description	
0	Single mode	(Initial value)
1	Multi mode	

Bit 3—Clock Select (CKS): Selects the A/D conversion time. Clear the ADST bit to 0 before switching the conversion time.

Bit 3:CKS	Description	
0	Conversion time = 266 states (maximum)	(Initial value)
1	Conversion time = 134 states (maximum)	

Bits 2 to 0—Channel Select 2 to 0 (CH2 to CH0): These bits and the MULTI bit select the analog input channels. Clear the ADST bit to 0 before changing the channel selection.

Channel Selection			Description	
CH2	**CH1**	**CH0**	**Single Mode (MULTI = 0)**	**Multi Mode (MULTI = 1)**
0	0	0	AN0 (Initial value)	AN0
		1	AN1	AN0, AN1
	1	0	AN2	AN0 to AN2
		1	AN3	AN0 to AN3
1	0	0	AN4	AN4
		1	AN5	AN4, AN5
	1	0	AN6	AN4 to AN6
		1	AN7	AN4 to AN7

23.2.3 A/D Control Register (ADCR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	TRGE1	TRGE0	SCN	RESVD1	RESVD2	—	—	—
Initial value:	0	0	0	0	0	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R	R	R

ADCR is an 8-bit read/write register that enables or disables external triggering of A/D conversion. ADCR is initialized to H'07 by a reset and in standby mode.

Bit 7, 6—Trigger Enable (TRGE1, TRGE0): Enables or disables external triggering of A/D conversion.

The TRGE1 and TRGE0 bits should only be set when conversion is not in progress.

Bit 7: TRGE1	Bit 6: TRGE0	Description
0	0	When an external trigger is input, the A/D conversion does not start (Initial value)
0	1	
1	0	
1	1	The A/D conversion starts at the falling edge of an input signal from the external trigger pin ($\overline{\text{ADTRG}}$).

Bit 5—Scan Mode (SCN): Selects multi mode or scan mode when the MULTI bit is set to 1. See the description of bit 4 in section 23.2.2.

Bits 4 and 3—Reserved (RESVD1, RESVD2): These bits are always read as 0, and should only be written with 0.

23.3 Bus Master Interface

ADDRA to ADDRD are 16-bit registers, but they are connected to the bus master by the upper 8 bits of the 16-bit peripheral data bus. Therefore, although the upper byte can be accessed directly by the bus master, the lower byte is read through an 8-bit temporary register (TEMP).

An A/D data register is read as follows. When the upper byte is read, the upper-byte value is transferred directly to the bus master and the lower-byte value is transferred into TEMP. Next, when the lower byte is read, the TEMP contents are transferred to the bus master.

When reading an A/D data register, always read the upper byte before the lower byte. It is possible to read only the upper byte, but if only the lower byte is read, the read value is not guaranteed.

Figure 23.2 shows the data flow for access to an A/D data register.

See section 23.7.3, Access Size and Read Data.

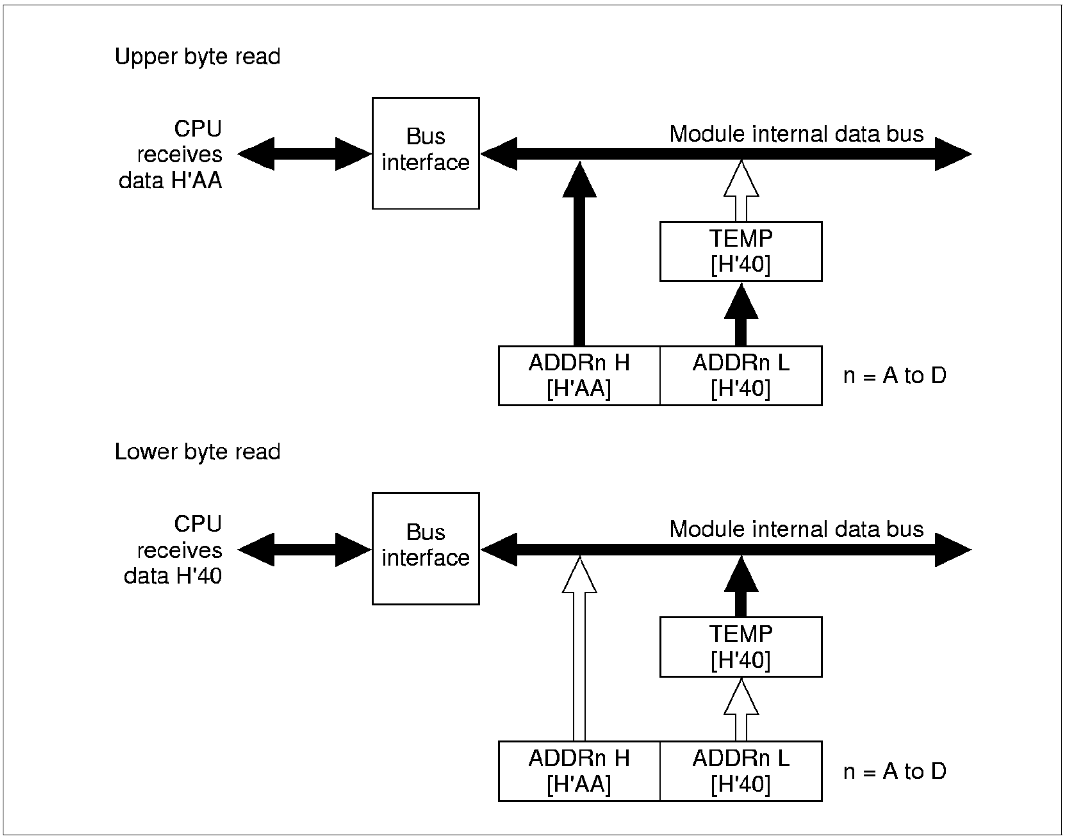


Figure 23.2 A/D Data Register Access Operation (Reading H'AA40)

23.4 Operation

The A/D converter operates by successive approximations with 10-bit resolution. It has two operating modes: single mode and scan mode.

23.4.1 Single Mode (MULTI = 0)

Single mode should be selected when only one A/D conversion on one channel is required. A/D conversion starts when the ADST bit is set to 1 by software, or by external trigger input. The ADST bit remains set to 1 during A/D conversion and is automatically cleared to 0 when conversion ends.

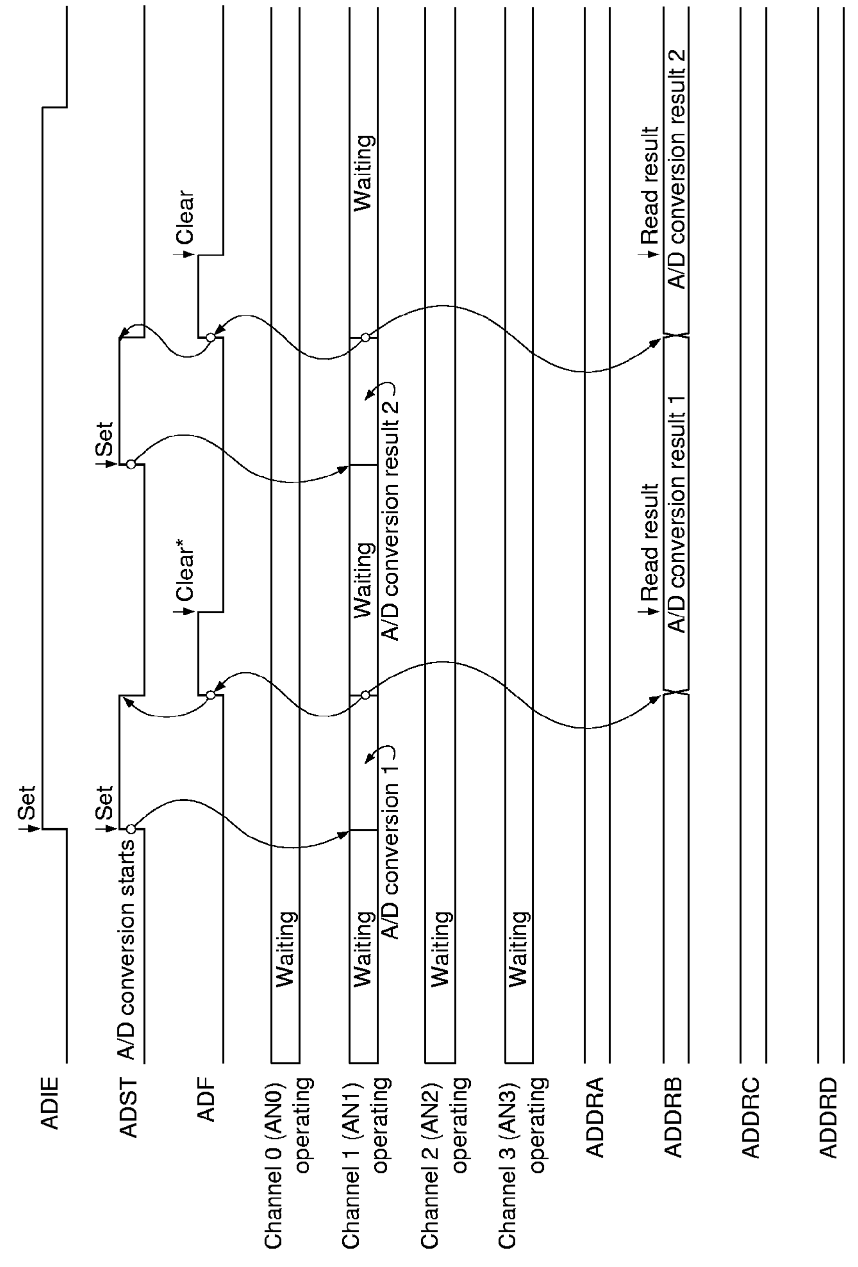
When conversion ends the ADF bit is set to 1. If the ADIE bit is also set to 1, an ADI interrupt is requested at this time. To clear the ADF flag to 0, first read ADCSR, then write 0 in ADF.

When the mode or analog input channel must be switched during A/D conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1 to start A/D conversion again. The ADST bit can be set at the same time as the mode or channel is changed.

Typical operations when channel 1 (AN1) is selected in single mode are described next.

Figure 23.3 shows a timing diagram for this example.

1. Single mode is selected (MULTI = 0), input channel AN1 is selected (CH2 = CH1 = 0, CH0 = 1), the A/D interrupt is enabled (ADIE = 1), and A/D conversion is started (ADST = 1).
2. When A/D conversion is completed, the result is transferred into ADDRB. At the same time the ADF flag is set to 1, the ADST bit is cleared to 0, and the A/D converter becomes idle.
3. Since ADF = 1 and ADIE = 1, an ADI interrupt is requested.
4. The A/D interrupt processing routine starts.
5. The routine reads ADCSR, then writes 0 in the ADF flag.
6. The routine reads and processes the conversion result (ADDRB = 0).
7. Execution of the A/D interrupt processing routine ends. Then, when the ADST bit is set to 1, A/D conversion starts to execute 2 to 7 above.



Note: Downward arrows (↓) indicate instruction execution.

Figure 23.3 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)

23.4.2 Multi Modc (MULTI = 1)

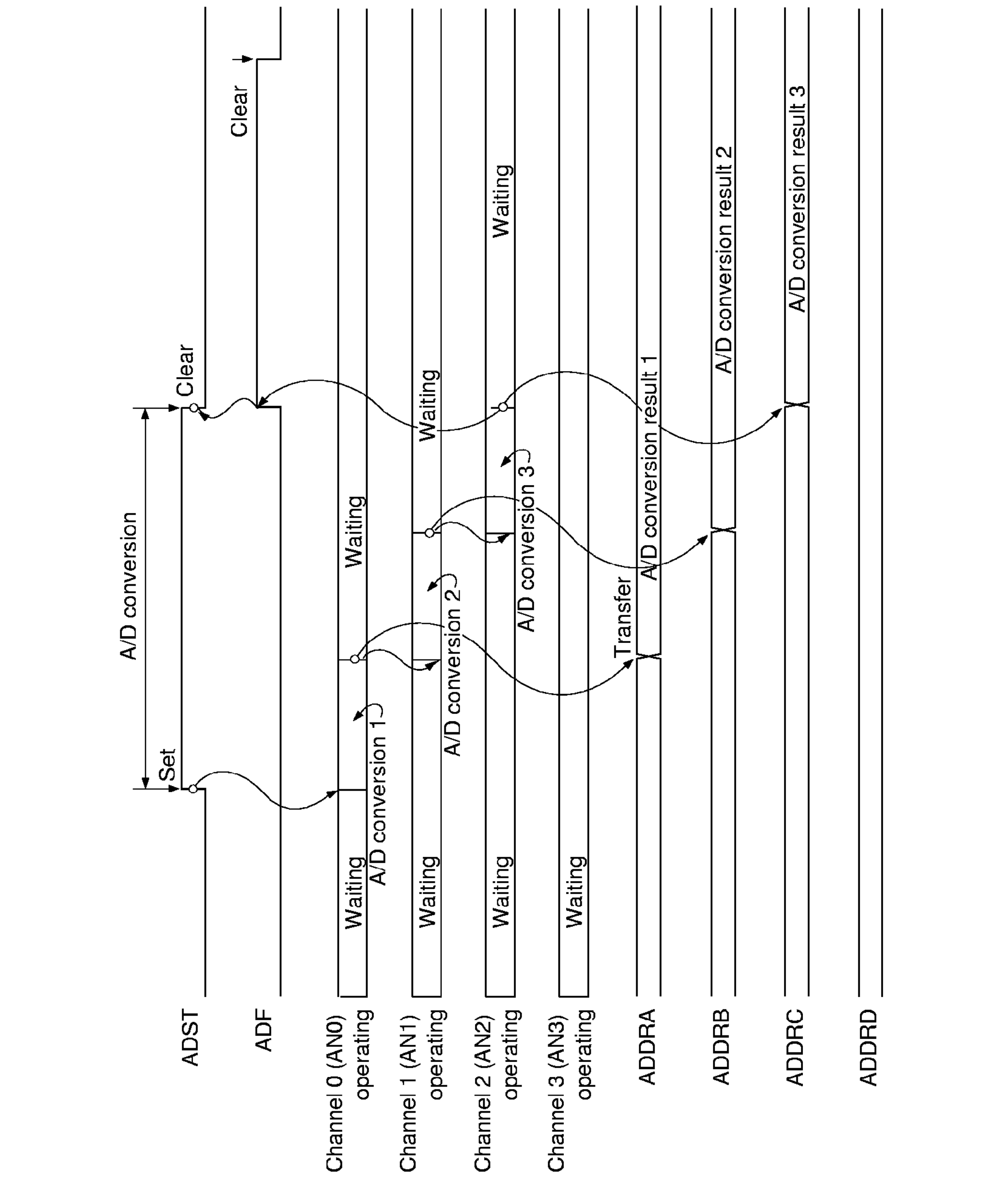
Multi mode should be selected when performing multi channel A/D conversions on one or more channels. When the ADST bit is set to 1 by software or external trigger input, A/D conversion starts on the first channel in the group (AN0 when CH2 = 0, AN4 when CH2 = 1). When two or more channels are selected, after conversion of the first channel ends, conversion of the second channel (AN1 or AN5) starts immediately. When A/D conversions end on the selected channels, the ADST bit is cleared to 0. The conversion results are transferred for storage into the A/D data registers corresponding to the channels.

When the mode or analog input channel selection must be changed during A/D conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1. A/D conversion will start again from the first channel in the group. The ADST bit can be set at the same time as the mode or channel selection is changed.

Typical operations when three channels in group 0 (AN0 to AN2) are selected in scan mode are described next. Figure 23.4 shows a timing diagram for this example.

1. Multi mode is selected (MULTI = 1), channel group 0 is selected (CH2 = 0), analog input channels AN0 to AN2 are selected (CH1 = 1, CH0 = 0), and A/D conversion is started (ADST = 1).
2. When A/D conversion of the first channel (AN0) is completed, the result is transferred into ADDRA. Next, conversion of the second channel (AN1) starts automatically.
3. Conversion proceeds in the same way through the third channel (AN2).
4. When conversion of all selected channels (AN0 to AN2) is completed, the ADF flag is set to 1 and ADST bit is cleared to 0. If the ADIE bit is set to 1, an ADI interrupt is requested at this time.

When the ADST bit is cleared to 0, A/D conversion stops. After that, if the ADST bit is set to 1, A/D conversion starts again from the first channel (AN0).



Note: Downward arrows (↓) indicate instruction executed by software.

Figure 23.4 Example of A/D Converter Operation (Multi Mode, Channels AN0 to AN2 Selected)

23.4.3 Scan Mode (MULTI = 1, SCN = 1)

Scan mode is useful for monitoring analog inputs in a group of one or more channels. When the ADST bit in the A/D control/status register (ADCSR) is set to 1 by software or external trigger input, A/D conversion starts on the first channel in the group (AN0 when CH2 = 0, AN4 when CH2 = 1)). When two or more channels are selected, after conversion of the first channel ends, conversion of the second channel (AN1 or AN5) starts immediately. A/D conversion continues cyclically on the selected channels until the ADST bit is cleared to 0. The conversion results are transferred for storage into the A/D data registers corresponding to the channels.

When the mode or analog input channel must be changed during analog conversion, to prevent incorrect operation, first clear the ADST bit to 0 to halt A/D conversion. After making the necessary changes, set the ADST bit to 1. A/D conversion will start again from the first channel in the group. The ADST bit can be set at the same time as the mode or channel selection is changed.

Typical operations when three channels (AN0 to AN2) are selected in scan mode are described next. Figure 23.5 shows a timing diagram for this example.

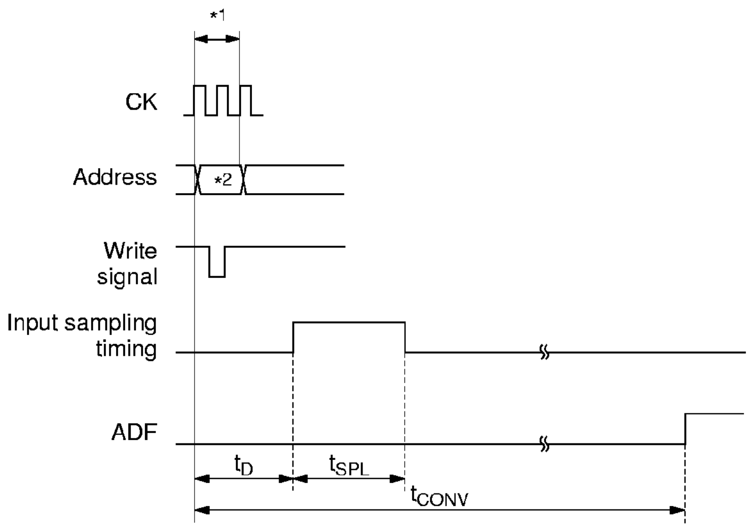
1. Scan mode is selected (MULTI = 1, SCN = 1), channel group 2 is selected (CH2 = 0), analog input channels AN0 to AN2 are selected (CH1 = 1, CH0 = 0), and A/D conversion is started (ADST = 1).
2. When A/D conversion of the first channel (AN0) is completed, the result is transferred into ADDRA. Next, conversion of the second channel (AN1) starts automatically.
3. Conversion proceeds in the same way through the third channel (AN2).
4. When conversion of all the selected channels (AN0 to AN2) is completed, the ADF flag is set to 1 and conversion of the first channel (AN0) starts again. If the ADIE bit is set to 1, an ADI interrupt is requested at this time.
5. Steps 2 to 4 are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops. After that, if the ADST bit is set to 1, A/D conversion starts again from the first channel (AN0).

23.4.4 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit. The A/D converter samples the analog input at a time t_D after the ADST bit is set to 1, then starts conversion. Figure 23.6 shows the A/D conversion timing. Table 23.4 indicates the A/D conversion time.

As indicated in figure 23.6, the A/D conversion time includes t_D and the input sampling time. The length of t_D varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 23.4.

In multi mode and scan mode, the values given in table 23.4 apply to the first conversion. In the second and subsequent conversions the conversion time is fixed at 256 states when CKS = 0 or 128 states when CKS = 1.



t_D A/D conversion start delay

t_{SPL} Input sampling time

t_{CONV} A/D conversion time

Notes: 1. ADCSR write cycle

2. ADCSR address

Figure 23.6 A/D Conversion Timing

Table 23.4A/D Conversion Time (Single Mode)

		CKS = 0			CKS = 1		
	Symbol	**Min**	**Typ**	**Max**	**Min**	**Typ**	**Max**
A/D conversion start delay	t_D	10	—	17	6	—	9
Input sampling time	t_{SPL}	—	65	—	—	32	—
A/D conversion time	t_{CONV}	259	—	266	131	—	134

Note: Values in the table are numbers of states (t_{cyc}).

23.4.5 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGE1, TRGE0 bits are set to 1 in ADCR, external trigger input is enabled at the $\overline{\text{ADTRG}}$ pin. A high-to-low transition at the $\overline{\text{ADTRG}}$ pin sets the ADST bit to 1 in ADCSR, starting A/D conversion. Other operations, regardless of the conversion mode, are the same as if the ADST bit had been set to 1 by software. Figure 23.6 shows the timing.

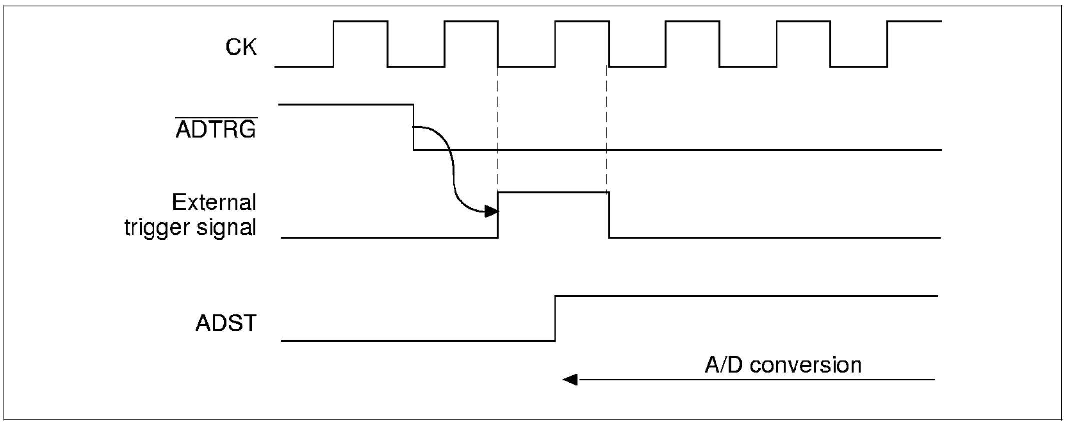


Figure 23.7 External Trigger Input Timing

23.5 Interrupts

The A/D converter generates an interrupt (ADI) at the end of A/D conversion. The ADI interrupt request can be enabled or disabled by the ADIE bit in ADCSR.

23.6 Definitions of A/D Conversion Accuracy

The A/D converter compares an analog value input from an analog input channel to its analog reference value and converts it into 10-bit digital data. The absolute accuracy of this A/D conversion is the deviation between the input analog value and the output digital value. It includes the following errors:

- Offset error
- Full-scale error
- Quantization error
- Nonlinearity error

These four error quantities are explained below using figure 23.8. In the figure, the 10 bits of the A/D converter have been simplified to 3 bits.

Offset error is the deviation between actual and ideal A/D conversion characteristics when the digital output value changes from the minimum (zero voltage) 0000000000 (000 in the figure) to 000000001 (001 in the figure)(figure 23.8, item (1)). Full-scale error is the deviation between actual and ideal A/D conversion characteristics when the digital output value changes from the 1111111110 (110 in the figure) to the maximum 1111111111 (111 in the figure)(figure 23.8, item (2)). Quantization error is the intrinsic error of the A/D converter and is expressed as 1/2 LSB (figure 23.8, item (3)). Nonlinearity error is the deviation between actual and ideal A/D conversion characteristics between zero voltage and full-scale voltage (figure 23.8, item (4)). Note that it does not include offset, full-scale or quantization error.

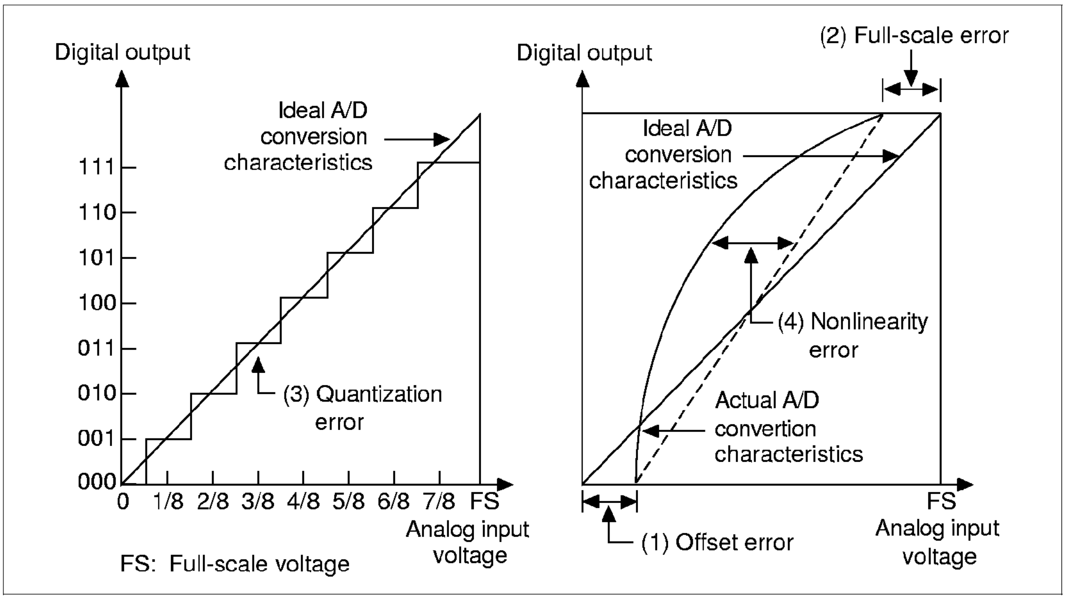


Figure 23.8 Definitions of A/D Conversion Accuracy

23.7 A/D Converter Usage Notes

When using the A/D converter, note the points listed in section 23.7.1 below.

23.7.1 Setting Analog Input Voltage

- Analog Input Voltage Range: During A/D conversion, the voltages input to the analog input pins ANn should be in the range $AV_{SS} \leq ANn \leq AV_{CC}$ (n = 0–7).
- Relationships of AV_{CC} and AV_{SS} to V_{CC} and V_{SS}: AV_{CC}, AV_{SS}, V_{CC} and V_{SS} should be related as follows: $AV_{CC} = V_{CC} \pm 0.3$ V and $AV_{SS} = V_{SS}$.

23.7.2 Processing of Analog Input Pins

To prevent damage from voltage surges at the analog input pins (AN0–AN7), connect an input protection circuit like the one shown in figure 23.9. The circuit shown also includes an RC filter to suppress noise. This circuit is shown as an example; The circuit constants should be selected according to actual application conditions. Table 23.5 lists the analog input pin specifications and figure 23.10 shows an equivalent circuit diagram of the analog input ports.

23.7.3 Access Size and Read Data

Table 23.6 shows the relationship between access size and read data. Note the read data obtained with different access sizes, bus widths, and endian modes.

The case is shown here in which H'3FF is obtained when AV_{CC} is input as an analog input. FF is the data containing the upper 8 bits of the conversion result, and C0 is the data containing the lower 2 bits.

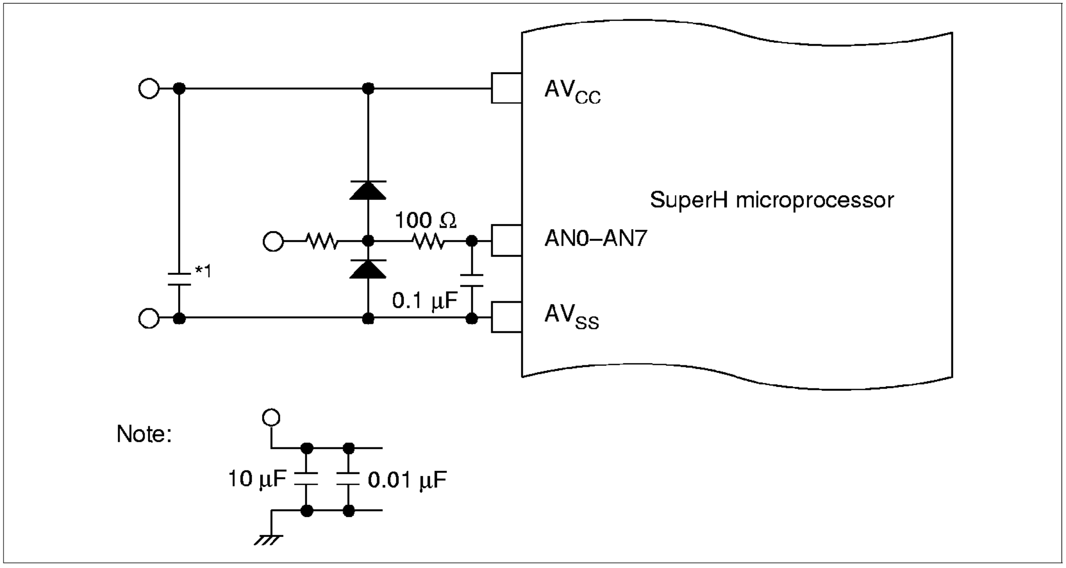


Figure 23.9 Example of Analog Input Protection Circuit

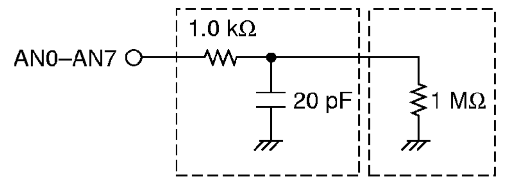


Figure 23.10 Analog Input Pin Equivalent Circuit

Table 23.5 Analog Input Pin Ratings

Item	Min	Max	Unit
Analog input capacitance	—	20	pF
Allowable signal-source impedance	—	5	kΩ

Table 23.6 Relationship between Access Size and Read Data

Access		Bus Width	32 Bits (D31—D0)		16 Bits (D15—D0)		8 Bits (D7—D0)	
Size	Command	Endian	Big	Little	Big	Little	Big	Little
Byte access	MOV.L	#ADDRAH,R9						
	MOV.B	@R9,R8	FFFFFFFF	FFFFFFFF	FFFF	FFFF	FF	FF
	MOV.L	#ADDRAL,R9						
	MOV.B	@R9,R8	C0C0C0C0	C0C0C0C0	C0C0	C0C0	C0	C0
Word access	MOV.L	#ADDRAH,R9						
	MOV.W	@R9,R8	FFxxFFxx	FFxxFFxx	FFxx	FFxx	FFxx	xxFF
	MOV.L	#ADDRAL,R9						
	MOV.W	@R9,R8	C0xxC0xx	C0xxC0xx	C0xx	C0xx	C0xx	xxC0
Longword access	MOV.L	#ADDRAH,R9						
	MOV.L	@R9,R8	FFxxC0xx	FFxxC0xx	FFxxC0xx	C0xxFFxx	FFxxC0xx	xxC0xxFF

In this table: #ADDRAH .EQU H'04000080

#ADDRAL .EQU H'04000082

Values are shown in hexadecimal for the case where read data is output to an external device via R8.

Section 24 D/A Converter

24.1 Overview

This LSI includes a D/A converter with two channels.

24.1.1 Features

D/A converter features are listed below.

- Eight-bit resolution
- Two output channels
- Conversion time: maximum 10 µs (with 20-pF capacitive load)
- Output voltage: 0 V to AVcc

24.1.2 Block Diagram

Figure 24.1 shows a block diagram of the D/A converter.

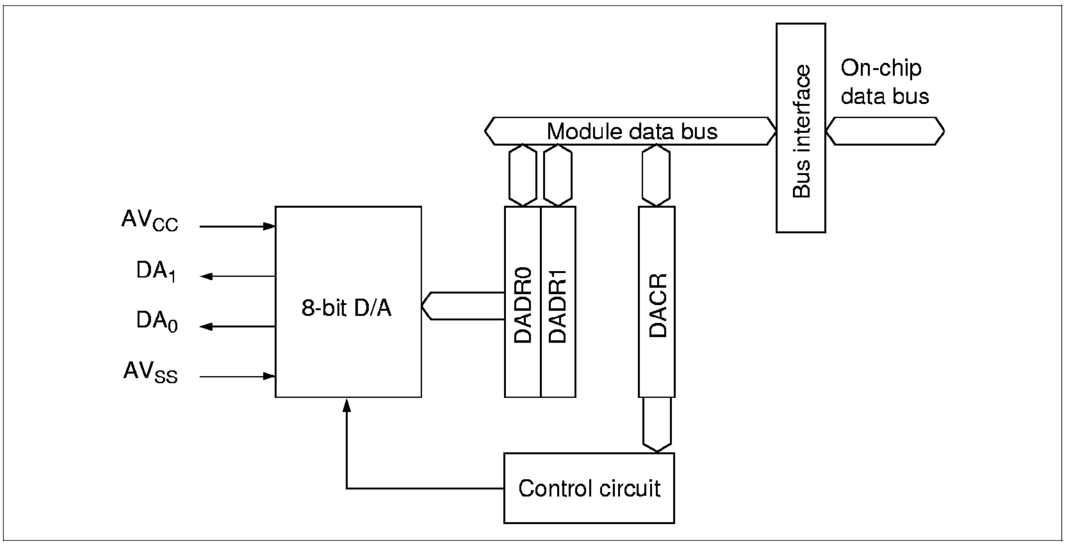


Figure 24.1 D/A Converter Block Diagram

24.1.3 I/O Pins

Table 24.1 summarizes the D/A converter's input and output pins.

Table 24.1D/A Converter Pins

Pin Name	Abbreviation	I/O	Function
Analog power-supply pin	AVcc	Input	Analog power supply
Analog ground pin	AVss	Input	Analog ground and reference voltage
Analog output pin 0	DA0	Output	Analog output, channel 0
Analog output pin 1	DA1	Output	Analog output, channel 1

24.1.4 Register Configuration

Table 24.2 summarizes the D/A converter's registers.

Table 24.2D/A Converter Registers

Address*	Name	Abbreviation	R/W	Initial Value
H'040000A0	D/A data register 0	DADR0	R/W	H'00
H'040000A2	D/A data register 1	DADR1	R/W	H'00
H'040000A4	D/A control register	DACR	R/W	H'1F

Note: Lower 16 bits of the address

24.2 Register Descriptions

24.2.1 D/A Data Registers 0 and 1 (DADR0/1)

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The D/A data registers (DADR0 and DADR1) are 8-bit read/write registers that store the data to be converted. When analog output is enabled, the D/A data register values are constantly converted and output at the analog output pins.

The D/A data registers are initialized to H'00 by a reset.

24.2.2 D/A Control Register (DACR)

Bit:	7	6	5	4	3	2	1	0
Bit name:	DAOE1	DAOE0	DAE	—	—	—	—	—
Initial value:	0	0	0	1	1	1	1	1
R/W:	R/W	R/W	R/W	R	R	R	R	R

DACR is an 8-bit read/write register that controls the operation of the D/A converter. DACR is initialized to H'1F by a reset.

Bit 7—D/A Output Enable 1 (DAOE1): Controls D/A conversion and analog output.

Bit 7: DAOE1	Description	
0	DA1 analog output is disabled	(Initial value)
1	Channel-1 D/A conversion and DA1 analog output are enabled	

Bit 6—D/A Output Enable 0 (DAOE0): Controls D/A conversion and analog output.

Bit 6: DAOE0	Description	
0	DA0 analog output is disabled	(Initial value)
1	Channel-0 D/A conversion and DA0 analog output are enabled	

Bit 5—D/A Enable (DAE): Controls D/A conversion, together with bits DAOE0 and DAOE1. When the DAE bit is cleared to 0, D/A conversion is controlled independently in channels 0 and 1. When this LSI enters standby mode while D/A conversion is enabled, the D/A output is held and the analog power-supply current is equivalent to that during D/A conversion. To reduce the analog power-supply current in standby mode, clear the DAOE0 and DAOE1 bits and disable the D/A output.

Bit 7: DAOE1	Bit 6: DAOE0	Bit 5: DAE	Description
0	0	—	D/A conversion is disabled in channels 0 and 1 (Initial value)
0	1	0	D/A conversion is enabled in channel 0 D/A conversion is disabled in channel 1
0	1	1	D/A conversion is enabled in channels 0 and 1
1	0	0	D/A conversion is disabled in channel 0 D/A conversion is enabled in channel 1
1	0	1	D/A conversion is enabled in channels 0 and 1
1	1	—	D/A conversion is enabled in channels 0 and 1

When the DAE bit is set to 1, even if bits DAOE0 and DAOE1 in DACR and the ADST bit in ADCSR are cleared to 0, the same current is drawn from the analog power supply as during A/D and D/A conversion.

Bits 4 to 0—Reserved: Read-only bits, always read as 1.

24.3 Operation

The D/A converter has two built-in D/A conversion circuits that can perform conversion independently.

D/A conversion is performed constantly while enabled in DACR. If the DADR0 or DADR1 value is modified, conversion of the new data begins immediately. The conversion results are output when bits DAOE0 and DAOE1 are set to 1.

An example of D/A conversion on channel 0 is given next. Timing is indicated in figure 24.2.

1. Data to be converted is written in DADR0.
2. Bit DAOE0 is set to 1 in DACR. D/A conversion starts and DA0 becomes an output pin. The converted result is output after the conversion time. The output value is (DADR0 contents/256) x AVcc. Output of this conversion result continues until the value in DADR0 is modified or the DAOE0 bit is cleared to 0.
3. If the DADR0 value is modified, conversion starts immediately, and the result is output after the conversion time.
4. When the DAOE0 bit is cleared to 0, DA0 becomes an input pin.

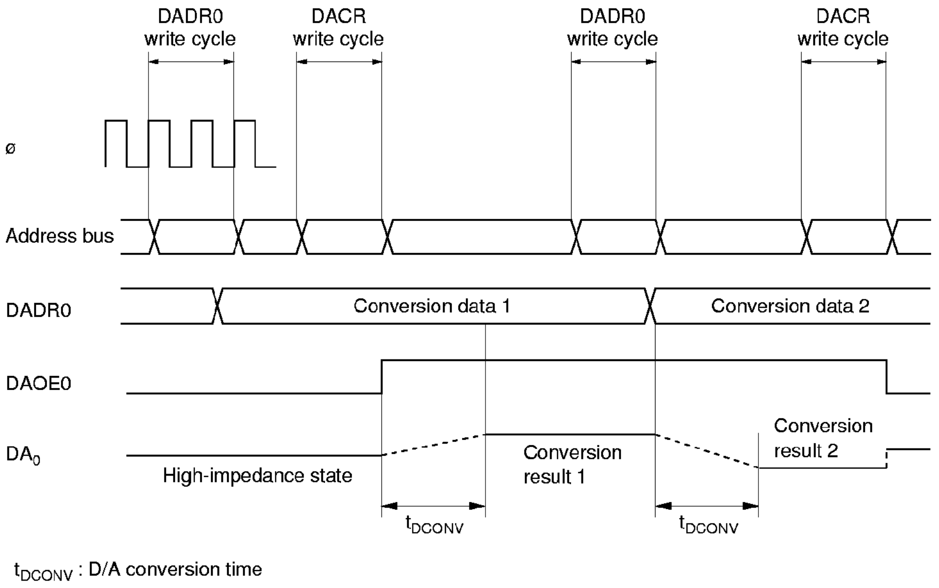


Figure 24.2 Example of D/A Converter Operation

Section 25 Hitachi User-Debugging Interface and Advanced User Debugger

25.1 Overview

The SH7729 incorporates a Hitachi user-debugging interface (H-UDI) and advanced user debugger (AUD) for program debugging.

25.2 Hitachi User Debugging Interface (H-UDI)

The H-UDI (user-debugging interface) performs on-chip debugging which is supported by the SH7729. The H-UDI described here is a serial interface which is pi-compatible with JTAG (Joint Test Action Group, IEEE Standard 1149.1 and IEEE Standard Test Access Port and Boundary-Scan Architecture) specifications and has the following emulation functions:

- 10-bit resolution
- H-UDI boot
- H-UDI rcsct
- H-UDI break
- H-UDI interrupt
- ASERAM write

The ASERAM in the SH7729 is provided for storage of the emulator's firmware program and is a 1-kbyte RAM which can be accessed only in the ASE mode.

Note: There is no dedicated evaluation chip for the SH7729. The SH7729 production chip can be used also as an evaluation chip.

25.2.1 Hardware Resources and Pins

The hardware resources are listed in table 25.1. Because there is no dedicated evaluation chip, all functions and pins are supported by the SH7729. Of the pins listed in the table, TCK, TMS, $\overline{\text{TRST}}$, TDI, and TDO conform with the JTAG standard (IEEE Standard 1149.1).

Table 25.1Hardware Resources for Emulation

Name	Description
H-UDI	5 pins (TRST, TMS, TCK, TDI, TDO)
$\overline{\text{ASEMD0}}$	1 pin
ASEBRKAK	1 pin
ASERAM	1-kbyte firmware RAM

25.2.2 Pin Description

TCK: Clock pin for serial data input/output in the H-UDI. Data is delivered to the H-UDI serially from the data input pin (TDI) and output from the data output pin (TDO) synchronized with this clock.

TMS: Mode select input pin. Changes in this signal, in sync with TCK, determine the state of the TAP controller. Protocol conforms with JTAG (IEEE Std. 1140.1).

$\overline{\text{TRST}}$: Reset input pin for the H-UDI. Receives input asynchronously with TCK, and resets the H-UDI when low. Please refer to 25.4.2, reset configuration, for more details.

TDI: H-UDI serial data input pin. Data transferring to the H-UDI is performed by changes of this signal synchronized with TCK.

TDO: H-UDI serial data output pin. The data is output from the H-UDI by reading this signal in sync with TCK.

$\overline{\text{ASEMD0}}$: ASE mode select pin, 0/1 for ASE mode/Main mode.

$\overline{\text{ASEBRKAK}}$: ASE break acknowledge output pin; output from INTC to external pin. This pin is invoked by three possible bread resources: UBC break, H-UDI break and software break (BRK instruction).

25.2.3 Block Diagram

Figure 25.1 shows the block diagram of the H-UDI.

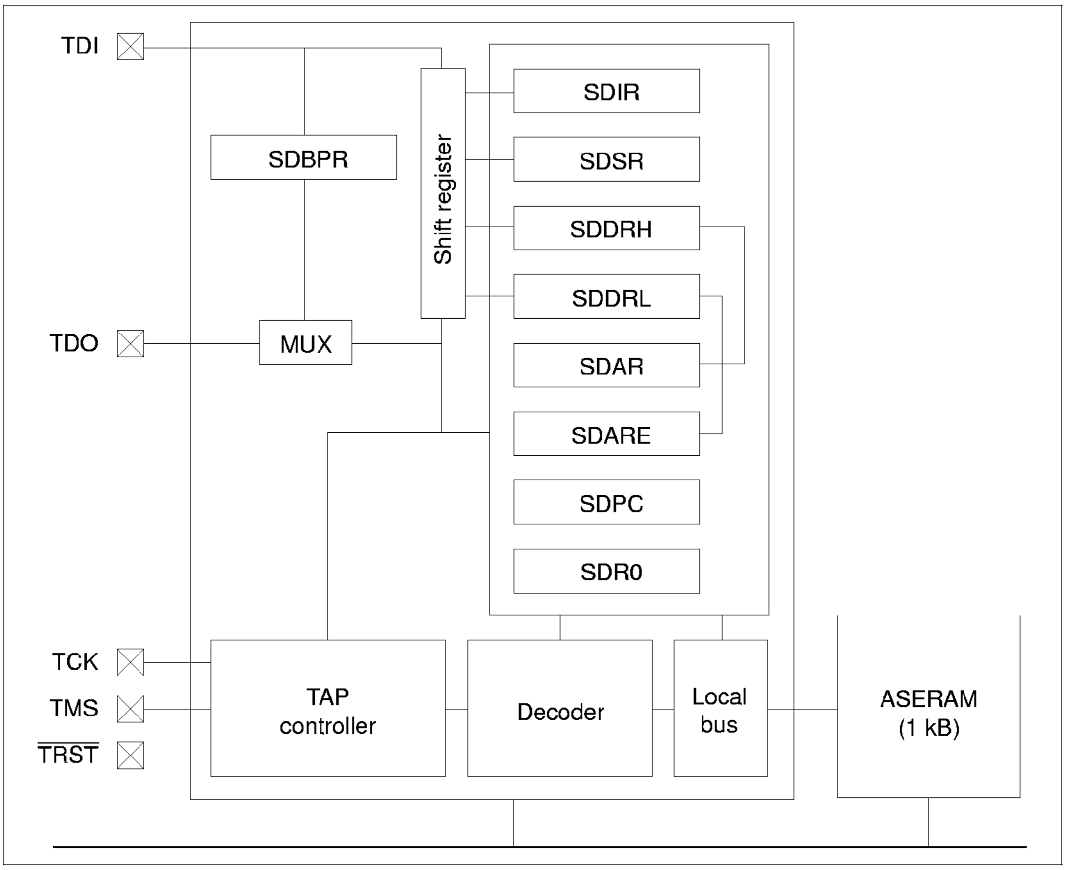


Figure 25.1 H-UDI Block Diagram

25.3 Control Registers

The H-UDI has the following registers.

- SDBPR: bypass register
- SDIR: instruction register
- SDSR: status register
- SDDRH and SDDRL: data registers high/low
- SDAR and SDARE: ASERAM write address and end address

In the SH7729 all registers are memory-mapped. Table 25.2 shows H-UDI register configuration. Control register spaces other than SDBPR are addressed by the CPU.

Table 25.2H-UDI Registers

		CPU Side			H-UDI Side		Initial
Name	**Abbreviation**	**R/W**	**Size**	**Address**	**R/W**	**Size**	**Value *1**
Bypass register	SDBPR	–	–	–	R/W	1	Undefined
Instruction register	SDIR	R	16	4000200	R/W	16	H'FFFF
Status register	SDSR	R/W *2	16	4000204	R	16	H'1101
Data register H	SDDR/ SDDRH	R/W	16/ 32	4000208	R/W	32	Undefined
Data register L	SDDRL	R/W	16	400020A	R/W	–	Undefined
Address register	SDAR	R*3	16	400020C	W*4	–	H'03FC
End address register	SDARE	R*3	16	4000210	W*4	–	H'0000
PC save register	SDPC	R/W	32	FD000000	–	–	Undefined
R0 save register	SDR0	R/W	32	FD000004	–	–	Undefined

Note: 1. Initialized when $\overline{\text{TRST}}$ pin is low or when TAP is in the test-logic-reset state.

2. Accessible only in the ASE mode; when write, only 1 can be written to bit 0 (SDTRF) by the CPU.

3. Accessible only the ASE mode.

4. Rewritten by the ASERAM write command.

25.3.1 Bypass Register (SDBPR)

The bypass register is a 1-bit register that cannot be accessed by the CPU. When the SDIR is set to the bypass mode, the SDBPR is connected between H-UDI pins TDI and TDO.

25.3.2 Instruction Register (SDIR)

The instruction register (SDIR) is a 16-bit read-only register. The register is in bypass mode in its initial state. It is initialized by $\overline{\text{TRST}}$ or in the TAP test-logic-reset state, and can be written by the H-UDI irrespective of the CPU mode. Operation is not guaranteed when a reserved command is set to this register

Bit:	15	14	13	12	11	10	9	8
Bit name:	TI3	TI2	TI1	TI0	—	—	—	—
Initial value:	1	1	1	1	1	1	1	1

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	—	—	—	—	—
Initial value:	1	1	1	1	1	1	1	1

Bits 15 to 12—Test Instruction Bits (TI3–TI0): Cannot be written by the CPU.

Bits 11 to 0—Reserved: Always read 1.

Table 25.3H-UDI Commands

TI3	TI2	TI1	TI0	Description
0	0	0	0	EXTEST
0	1	0	0	SAMPLE/PRELOAD
0	1	0	1	ASEM write
0	1	1	0	H-UDI reset negate
0	1	1	1	H-UDI reset assert
1	0	0	—	H-UDI boot
1	0	1	—	H-UDI interrupt
1	1	0	—	H-UDI break
1	1	1	0	Reserved
1	1	1	1	Bypass mode (initial value)
0	0	0	1	Recovery from sleep

25.3.3 Status Register (SDSR)

The status register (SDSR) is a 16-bit register that can be accessed by the CPU only in the ASE mode. This register's SDTRF bit can be re-written by the CPU as 1.

When the SDTRF bit is 1, the register can be read irrespective of the CPU state, though it cannot be written directly. The SDTRF is automatically set to 0 with an SDDR data write from the H-UDI (update DR). The register goes into initial state (1) with $\overline{\text{TRST}}$ asserted or in the TAP test-logic-reset state.

Bit:	15	14	13	12	11	10	9	8
Bit name:	PR3	PR2	PR1	PR0	VR3	VR2	VR1	VR0
Initial value:	1	1	1	1	1	1	1	1

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	—	—	—	ASEMW	BRKAF	SDTRF
Initial value:	1	1	1	1	1	1	1	1

Bits 15 to 12—Processor ID (PR3–PR0): 0001 for the SH7729.

Bits 11 to 8—Version ID (VR3–VR0): 0001 for the first cut.

Bits 7 to 3—Reserved: Cannot be written by the CPU, and always read 0.

Bit 2—ASERAM Write Status Flag (ASEMW): Indicating whether ASERAM write is under execution.

ASEMW	Description
0	ASERAM write not executed
1	ASERAM write executed

Bit 1—Break Acknowledgment Flag (BRAKF): Indicating whether the CPU is in the ASE break mode.

BRAKF	Description
0	CPU not in ASE break mode
1	CPU in ASE break mode

Bit 0—Serial Data Transfer Control Flag (SDTRF): Indicating SDDR access state.

SDTRF	Description
0	End of serial data transfer; SDDR can be accessed by the CPU
1	SDDR access disabled

25.3.4 Data Register (SDDR)

The data register is a 32-bit register comprised of two 16-bit linked registers (SDDRH and SDDRL) that can be written and read by the CPU. The SDDR is read/write enabled when the SDTRF bit is set to 0. The SDDR can be accessed in modes other than ASE mode. The SDTRF is automatically set to 0 when the data is written to the SDDR from the H-UDI (update DR state). This register value is not initialized by a $\overline{\text{TRST}}$ or CPU reset.

Bit:	31	30	29	28	27	26	25	24
Bit name:								
Initial value:	*	*	*	*	*	*	*	*

Bit:	23	22	21	20	19	18	17	16
Bit name:								
Initial value:	*	*	*	*	*	*	*	*

Bit:	15	14	13	12	11	10	9	8
Bit name:								
Initial value:	*	*	*	*	*	*	*	*

Bit:	7	6	5	4	3	2	1	0
Bit name:								
Initial value:	*	*	*	*	*	*	*	*

Bits 31 to 0—H-UDI data: Storing the SDDR value.

25.3.5 Address Register (SDAR and SDARE)

The address register (SDAR and SDARE) is an 8-bit register that is accessible by the CPU in the ASE mode. After issue of the ASERAM write command from the H-UDI, this register stores the first received SDDR value (SDDR[25:18] to SDAR and SDDR[9:2] to SDARE). The address register value is initialized by $\overline{\text{TRST}}$ or in the TAP test-logic-reset state.

SDAR (from SDDR[25:18])

Bit:	15	14	13	12	11	10	9	8
Bit name:	—	—	—	—	—	—	AR9	AR8
Initial value:	0	0	0	0	0	0	1	1

Bit:	7	6	5	4	3	2	1	0
Bit name:	AR7	AR6	AR5	AR4	AR3	AR2	—	—
Initial value:	1	1	1	1	1	1	0	0

Bits 9 to 2—Address Data (AR9–AR2): Storing address of ASERAM write.

Bits 15 to 10, 1, 0—Reserved: Always read 0.

SDARE (from SDDR[9:2])

Bit:	15	14	13	12	11	10	9	8
Bit name:	—	—	—	—	—	—	ARE9	ARE8
Initial value:	0	0	0	0	0	0	1	1

Bit:	7	6	5	4	3	2	1	0
Bit name:	ARE7	ARE6	ARE5	ARE4	ARE3	ARE2	—	—
Initial value:	1	1	1	1	1	1	0	0

Bits 9 to 2—End Address Data (ARE9–ARE2): Storing end address of ASERAM write.

Bits 15 to 10, 1, 0—Reserved: Always read 0.

25.3.6 SDPC and SDR0

SDPC and SDR0 are used to store the CPU PC and R0 values in the ASE mode. Because the PC and R0 are both 32-bit registers, only longword save will be accepted. When an ASE break interrupt occurs, $\overline{\text{ASEBRKAK}}$ will be asserted and the break source will be set in ASEEVT (address: H'400001E). At this time, the PC and R0 will be saved into H'FD000000 and H"FD000004, respectively. Refer to 25.5 for more details.

25.4 H-UDI Operations

Figure 25.2 shows the internal states of TAP controller. State transitions basically conform with the JTAG standard. However, the SDSR access control block has been expanded to facilitate handshake with the CPU and external devices.

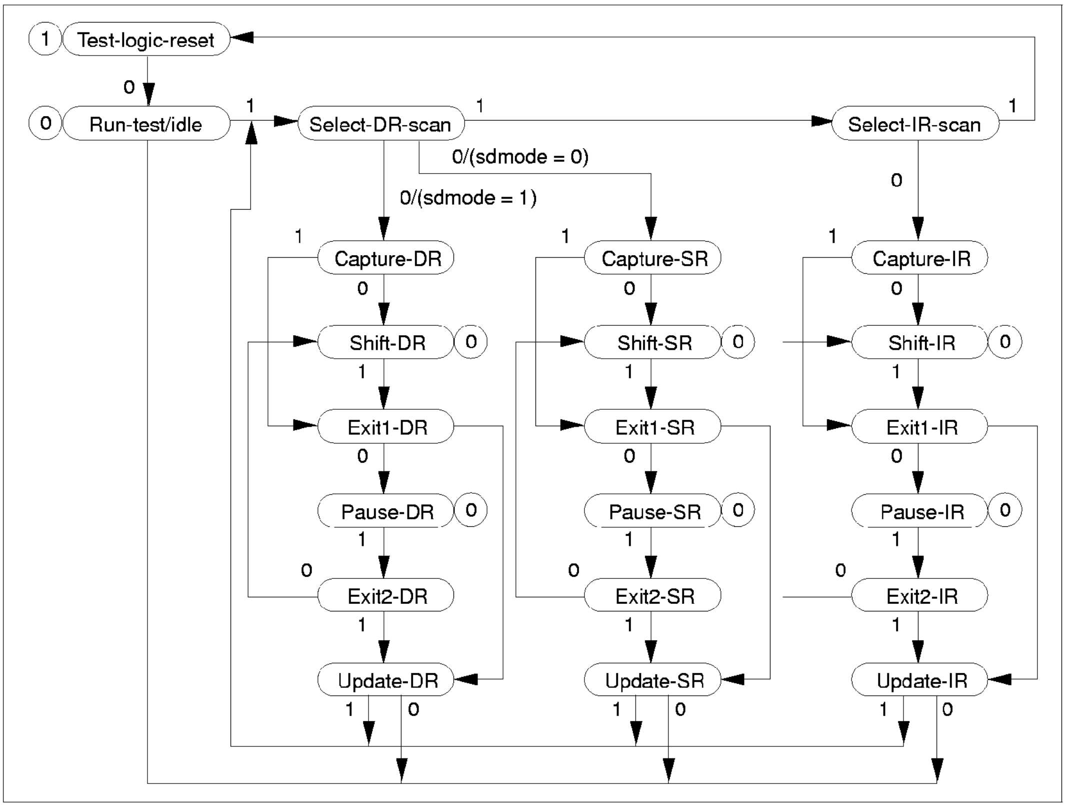


Figure 25.2 TAP Controller State Transitions

Note: The transition condition is the TMS value on the rising edge of TCK. The TDI value is sampled on the rising edge of TCK; shifting occurs on the falling edge of TCK. The TDO value changes on the TCK falling edge. The TDO is at high impedance, except with shift-DR (shift-SR) and shift-IR states. During the change to TRSTN = 0, there is a transition to test-logic-reset asynchronously with TCK. The SDTRF and sdmode are both initiated at 1.

25.4.1 Example of Data Transfer from Emulator to CPU

Figure 25.3 shows an example of data transfer from the emulator to the CPU.

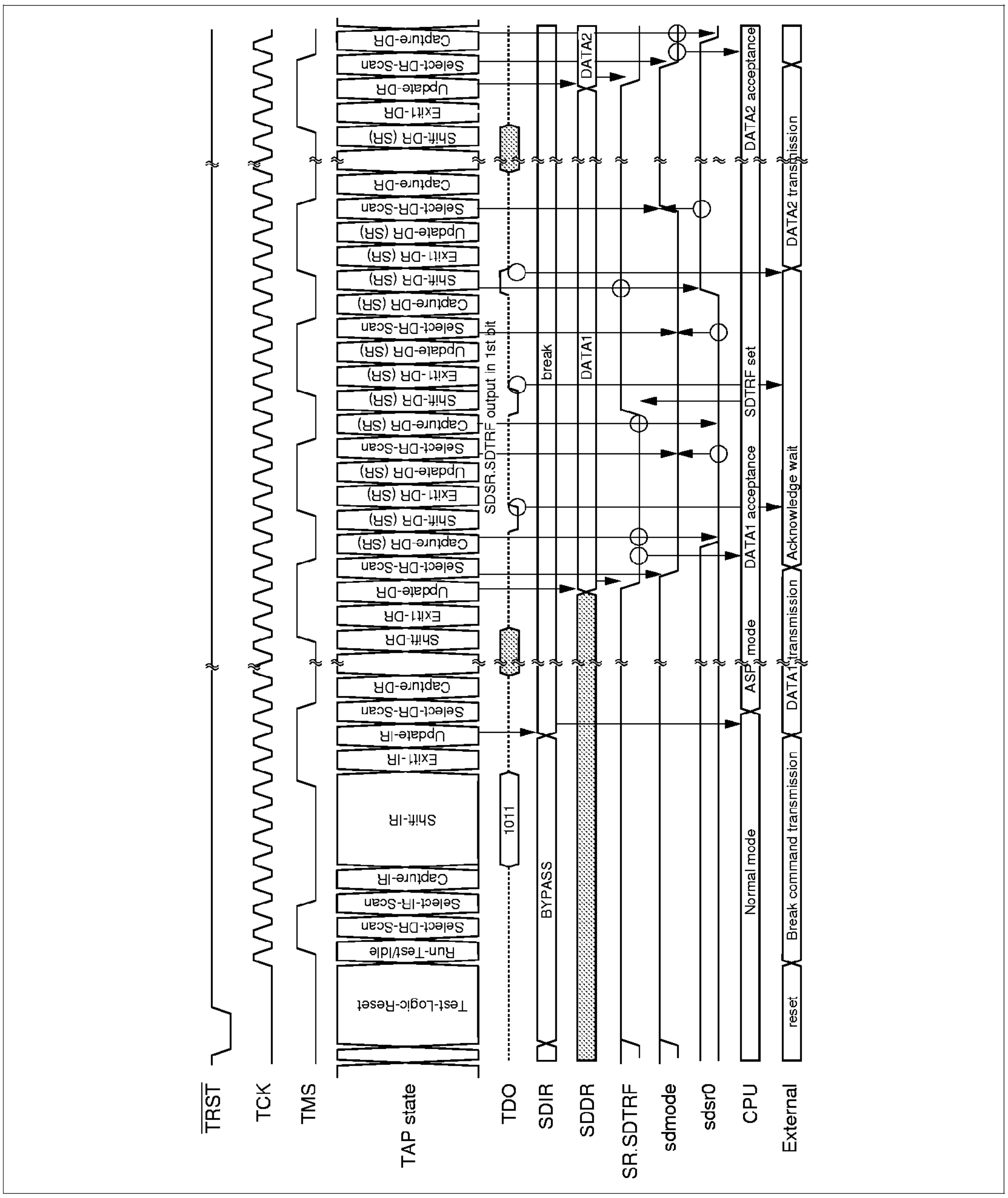


Figure 25.3 Example of Data Transfer from Emulator to CPU

1. On reset (TRST low), the TAP state initiates at test-logic-reset state; the SDIR is initiated at the bypass mode (SDIR = 1111); SDTRF and internal signal sdmode are both initiated at 1.
2. In shift-IR state (4-bit serial shift, with MSB last), instruction is serially shifted in.
3. Instruction gets update into SDIR in update-IR state.
4. Data is similarly shifted (32-bit serial shift, with MSB last) in the shift-DR state by inputting appropriate TMS (SDTRF and sdmode remain 1 at this time).
5. Update DR; SDTRF is automatically set to 0; the sdmode changes to 0.
6. When SDTRF bit is set to 0, H-UDI registers can be accessed by the CPU. The data can be transferred from SDDR. After register access is completed, 1 will be written to SDSR's SDTRF bit.
7. Monitoring SDSR by shifting out the SDTRF bit to the external controller. When the SDTRF set state (SDTRF = 1) is detected, internal register selector switches (SDSR -> SDDR, by sdmode set to 1), and serial transfer to SDDR is, again, enabled.
8. Steps 4 to 7 are repeated until all data transfer is finished.

Note: Please refer to figure 25.2 for state transitions as well as corresponding TMS input.

25.4.2 Reset Configuration

Table 25.4Reset Configuration

ASDMD0*[1]	RESETP	TRST	Chip State
1	0	0	Normal reset*[2] and H-UDI reset
		1	Normal reset
	1	0	H-UDI reset only
		1	Normal operation
0	0	0	Reset hold*[3]
		1	Normal reset
	1	0	H-UDI reset only
		1	Normal operation

Notes: 1. Performs main chip mode and ASE mode settings
ASEMD0 = 1, main chip mode
ASEMD0 = 0, ASE mode
2. Normal reset becomes main chip reset
3. During ASE mode, reset hold is enabled by setting RESETP and TRST pins at low level for a constant cycle. In this state, the CPU does not start up, even if RESETP is set to high level. When TRST is set to high level, H-UDI operation is enabled, but the CPU does not start up. The reset hold state is canceled by the following:
- Boot request from H-UDI (boot sequence)
- Another RESETP assert (power-on reset)

25.4.3 H-UDI Reset

An H-UDI reset is executed by setting an H-UDI reset assert command in SDIR. An H-UDI reset is of the same kind as a power-on reset. An H-UDI reset is released by inputting an H-UDI reset negate command.

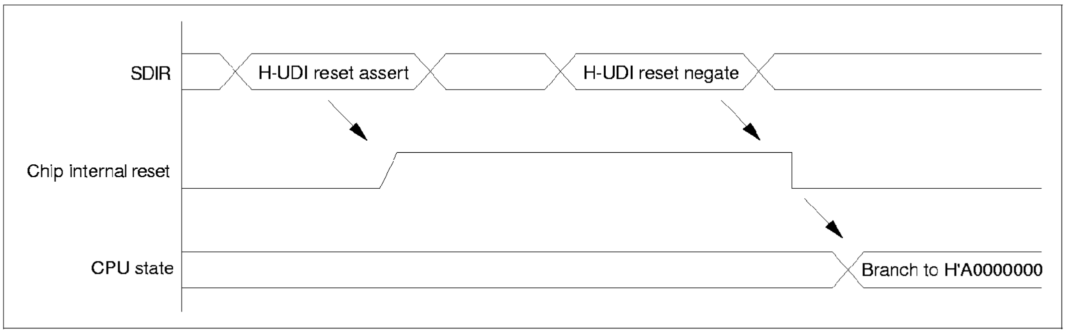


Figure 25.4 H-UDI Reset

25.4.4 H-UDI Break

An H-UDI break is an ASE break generated by setting a command from the H-UDI in the SDIR. As with other ASE breaks, the PC value is stored in H'FD000000 followed by a branch to H'FC0000000. An H-UDI break is distinguished from other ASE breaks by analyzing ASEEVT register. Please refer to 25.5 for more details.

H-UDI breaks are accepted in the ASE user mode and break mode.

Notes: 1. The ASE break request from the UBC is not accepted in the ASE break mode.

2. Break from the H-UDI, however, is accepted during break mode. In this case, the SDPC and SDR0 will be stored again; previously saved SDPC/SDR0 will be lost. Same thing applies to the BRK instruction in the break mode. Users have to be very careful in using H-UDI breaks in the ASE break mode.

25.4.5 H-UDI Interrupt

The H-UDI interrupt function generates an interrupt by setting a command from the H-UDI in the SDIR. An H-UDI interrupt is a general exception/interrupt operation, resulting in a branch to an address based on the VBR value plus offset, and return by the RTE instruction. This interrupt request is sent to interrupt controller (INTC), then CPU, with fixed priority level 15.

H-UDI interrupts are accepted in the main chip mode and ASE user mode.

25.4.6 Bypass

The JTAG-based bypass mode for the H-UDI pins can be selected by setting a command from the H-UDI in the SDIR. The chip goes into bypass mode when the H-UDI is initialized ($\overline{\text{TRST}}$ is asserted or TAP test-logic-reset state).

25.4.7 ASERAM Write

Continuous write to the ASERAM is enabled by the following procedure due to H-UDI command in the SDIR (4'b0101).

After issuing of the H-UDI ASERAM write command, the first-received SDDRH and SDDRL are stored as data-transfer start address and end address (the data-transfer start address is stored in the SDAR and end address in the SDARE). The SDDRH and SDDRL specify the lower 10 bits of the ASERAM (1 kB). Thc transfcr data is in longword units (4 bytcs).

During transfer, when the 32-bit serial data transfer ends, the SDTRF bit of the SDSR is automatically set to 1.

The SDIR command is automatically changed to bypass command after the write to the ASERAM ends.

The ASERAM can only be accessed in the ASE mode; an address error occurs in other modes, and the data at such time is unrelated to the ASERAM. Physically, connection is via (I-bus), and access is enabled by address (byte, word, or longword).

When the CPU and H-UDI access the ASERAM at the same time, the CPU has priority.

25.4.8 H-UDI Boot

The SH7729 has 1-kB on-chip emulator memory (ASERAM) and is necessary to transfer the data from an emulator to the ASERAM through the H-UDI. H-UDI boot is used to initialize emulation functions. Because of this, H-UDI boot branches to a different address (H'FC0003D0) from an ASE break (H'FC000000). Furthermore, as with a reset, it is not possible to return to the state just prior to the boot with an RTB instruction. A boot is only accepted in the ASE-reset hold mode.

25.4.9 Boot Sequence

The boot sequence can be performed by the following procedure:

1. Reset hold

   In the ASE mode, the chip can be put in the reset hold mode by driving $\overline{\text{RESETP}}$ and $\overline{\text{TRST}}$ pins low for a few cycles. The H-UDI is activated when the $\overline{\text{TRST}}$ pin is driven high. At this time, the CPU remains inactive on account of reset hold, even if the $\overline{\text{RESETP}}$ pin is driven high.
2. ASERAM write

   Loading firmware program into the ASERAM by issuing the ASERAM write command (SDIR, 0101). See above, ASERAM write for details.
3. H-UDI boot

   The boot command executes serial transfer to the SDIR. See section 25.4.8, H-UDI Boot, for details.

25.4.10 Using H-UDI to Recover from Sleep Mode

It is possible to recover from sleep mode by setting a command (0001) from the H-UDI in SDIR.

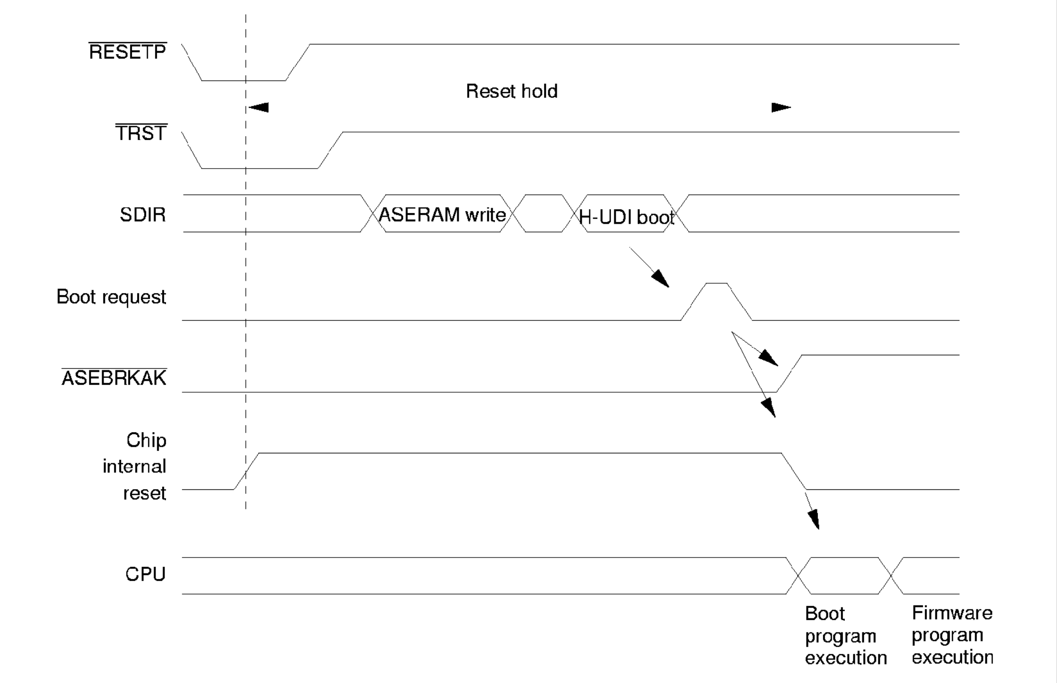


Figure 25.5 Boot Sequence

25.5 CPU Operations when ASE Break Occurs

The following interrupt process is for hardware and software breaks alike.

1. $\overline{\text{ASEBRKAK}}$ is asscrtcd and thc brcak sourcc is sct in ASEEVT.
2. PC is saved to H'FD000000.
3. R0 is saved to H'FD000004.
4. Process jumps to ASE routine (H'FC000000).

The ASEEVT is an 8-bit register and is readable and writable only in the break mode. In the ASE user mode, writes are ignored. The configuration of the ASEEVT is shown below:

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	—	FJBT	FTER	FTMS	FBRS	FJBR	FBRU
Initial value:	0	0	*	*	*	*	*	*
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

25.5.1 ASEEVT Configuration

FBRU: Break by user break controller (UBC)
FJBR: H-UDI brcak
FBRS: Software break (BRK instruction)
FTMS: TLB miss while in ASE break mode
FTER: TLB error while in ASE break mode
FJBT: H-UDI boot

Notes: 1. The ASEEVT is located in interrupt controller with address H'400001E.

2. If multiple ASE breaks occur (e.g. H-UDI break and UBC break occur at the same time), all corresponding flags will be set in the ASEEVT.

3. H'FD000000 (SDPC) and H'FD000004 (SDR0) are located in the H-UDI and can only be accessed by the CPU in longword format.

4. It is responsibility of software (or emulator program) to clear the ASEEVT flags after reading.

25.6 Notes on Use

1. ASERAM write by H-UDI

   The SH7729 allows continuous write from the ASERAM through the H-UDI without disrupting CPU control, even when CPU operation is suspended. However, when the H-UDI and CPU try to access the ASERAM at the same time, the CPU has priority. Reading ASERAM from the H-UDI is not permitted in the SH7729.

2. CPU access of H-UDI registers when SDTRF = 1

   In the SH7729, the CPU can access H-UDI registers without wait, even when SDTRF = 1. Owing to this, it is necessary and very important to check the SDTRF bit (in SDSR) to see whether SDDR load has finished.

3. ASERAM write command

   Operation is not guaranteed if an H-UDI command is issued during execution of an ASERAM write command.

4. H-UDI commands in standby mode

   Because chip operations are suspended during standby mode, H-UDI commands are not accepted. However, TAP controller (in H-UDI) remains in operation at this time.

5. H-UDI commands

   An H-UDI command other than ASERAM write or H-UDI interrupt, once set, will not be modified, as long as another command is not re-issued from the H-UDI. ASERAM write and H-UDI interrupt commands, however, will be changed to bypass command once set.

25.7 Advanced User Debugger (AUD)

D has features as follows:

Seven Input/Output Pins

- Test reset ($\overline{\text{TRST}}$, also used in H-UDI)
- AUD clock (AUDCK)
- Trace data bus (AUDATA3 to 0)
- Trace data synchronous signal ($\overline{\text{AUDSYNC}}$)
- Two modes: One of the following tow modes can be selected by inputting a value to AUDATA1 and 0 while $\overline{\text{TRST}}$ is asserted.
  - — Realtime trace mode and full trace mode
  - — No trace output during ASE break mode

Realtime Trace Mode: When branch occurs in program due to execution of a branch instruction or interrupt occurrence, AUD detects this and outputs a branch destination address data and branch source address calculation data (branch source trace data) from AUDATA3 to 0. Branch source trace data consists of the address at which the last instruction was fetched before branch occurrence, and data required for calculating an actual branch source address using the last instruction fetched address.

A branch destination address and branch source data are compared with each previous output addresses, and 4-, 8-, 16-, or 32-bit output is selected according to how many bits of an upper address match. The CPU continues operating even while the AUD is outputting a branch destination address or branch source trace data.

If the next branch occurs while trace data is output, the next branch trace data is output and its previous trace data output is suspended.

Full Trace Mode: The output data format is the same as that in realtime trace mode.

This mode differs from realtime trace mode in that CPU stops operating while the AUD is outputting trace data and trace data output is not suspended.

25.7.1 Block Diagram

Figure 25.6 shows the AUD block diagram.

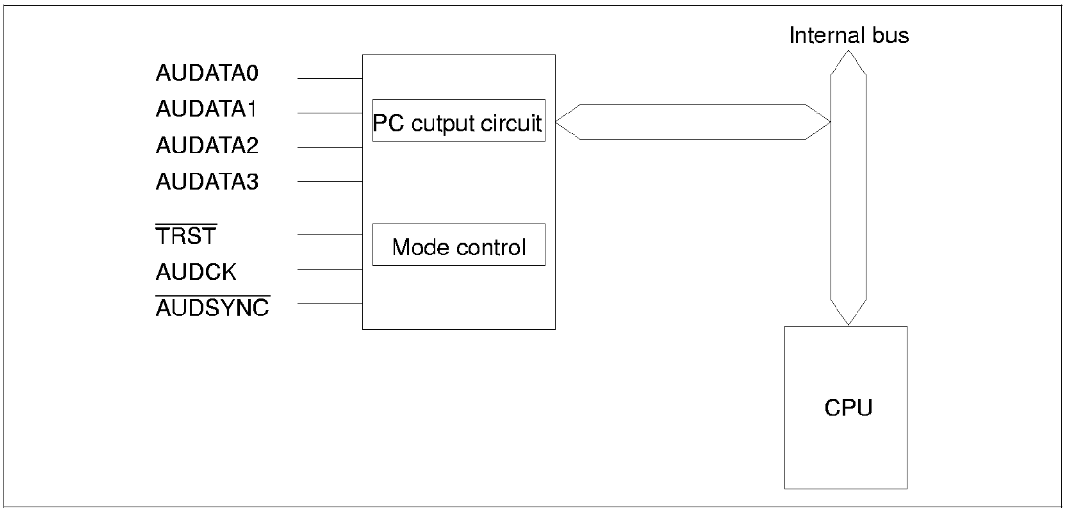


Figure 25.6 AUD Block Diagram

25.7.2 Pin Configuration

The AUD has I/O pins shown in table 25.5.

Table 25.5Pin Configuration

Name	Input/Output	Function
TRST	Input	Test reset input (also used in H-UDI)
AUDCK	Input	Test clock input
AUDATA 3 to 0	I/O	Trace mode selection input/trace data output
AUDSYNC	Output	Indicating whether trace data is valid or invalid

25.7.3 Pin Description

Name	Description
$\overline{\text{TRST}}$	The AUD, its buffer, and its logic can be reset by inputting a low level to this pin. After that, when a high level is input to this pin again after levels of AUDATA1 and 0 are defined, the AUD operates in the selected mode. This pin is also used as an H-UDI test reset input pin.
AUDCK	A clock that is used for debugging should be input to this pin. Then, trace data is output synchronously with this clock.
AUDATA3 to 0	AUDATA1 and 0 are used as trace mode selection input pins when $\overline{\text{TRST}}$ is asserted. Trace mode can be selected by inputting a value to AUDATA1 and 0 as shown below. At this time, AUDATA3 and 2 must not be driven. • AUDATA1,0 11: Trace data is not output 01: Full trace mode 10: Realtime trace mode 00: Reserved. Must not be set to this value. Realtime trace mode and full trace mode 1. $\overline{\text{AUDSYNC}}$ = Low When program branch or interrupt branch occurs, AUD asserts $\overline{\text{AUDSYNC}}$ and outputs a branch destination address and branch source trace data. A branch destination address is output in the order of A4 to A1, A8 to A5, A12 to A9, A16 to A13, A20 to A17, A24 to A21, A28 to A25, and A31 to A29. Branch source trace data is output in the order of A4 to A1, A8 to A5, A12 to A9, A16 to A13, A20 to A17, A24 to A21, A28 to A25, A31 to A29, and PID1 to PID0. A31 to A1 of branch source trace data are the address at which the last instruction was fetched before branch occurrence. PID2 to PID0 are used to calculate and actual branch source address with these output A31 to A1. Refer to the SH7410 hardware manual for more details. At this time, since the instruction is 16 or 32 bits, A0 of an address is always 0 and A0 is not output. A31 to A29 and PID2 to PID0 are output to AUDATA2 to 0; an undefined value is output to AUDATA3.

Name	Description
	2. $\overline{\text{AUDSYNC}}$ = High 0011 is always output in the waiting state for output of a branch destination address or branch source trace data. When branch occurs, 1 and 0 are output to AUDATA3 and 2, respectively. A value is output to AUDATA1 and 0 that indicates how many bits of an address are output according to the result of comparison of the current output destination address with the previous full-output branch destination address (see below). After a branch destination address is output, 1s are output to AUDATA3 and 2 before output of branch-source trace data. A value is output to AUDATA1 and 0 that indicates how many bits of an address are output according to the result of comparison of the current output branch-source trace data with the previous full-output branch-source trace data (see below). The meaning of value output to AUDATA1 and 0 is as follows: 00: Indicates that A31 to A5 of an address match, and that four bits of A4 to A1 are output. Therefore, output count is 1. 01: Indicates that A31 to a9 of an address match, and that eight bits of A4 to A1 and A8 to A5 are output. Therefore, output count is 2. 10: Indicates that A31 to A17 of an address match, and that 16 bits of A4 to A1, A8 to A5, A12 to A19, and A16 to A13 are output. Therefore, output count is 4. 11: Indicates that less than A31 to A17 of and address match, and that 31 bits of A4 to A1, A8 to A5, A12 to A9, A16 to A13, A20 to A17, A24 to A21, A28 to A25, and A31 to A29 are output. Therefore, output count is 8.
$\overline{\text{AUDSYNC}}$	This pin indicates whether a branch destination address or branch-source trace data is being output. High: A branch destination address or branch-source trace data is not being output. Low: A branch destination address or branch source trace data is being output.

25.8 Realtime Trace Mode

25.8.1 Overview

In this modc, a branch dcstination addrcss and branch-sourcc tracc data arc output whcn branch occurs in user program due to execution of a branch instruction or interrupt occurrence. The CPU continues operating even while the AUD is outputting trace data.

25.8.2 Description on Operation

When $\overline{\text{TRST}}$ is asserted, AUDATA1 and 0 are set to 1 and 0, respectively, and then $\overline{\text{TRST}}$ is negated, AUD starts operating in realtime trace mode.

Figurc 25.7 shows thc cxamplc of data output.

The AUDATA always outputs 0011 synchronously with AUDCK as long as user program is executed without branch.

When branch occurs, operation starts as follows. First 1000 (4-bit output), 1001 (8-bit output), 1010 (16-bit output), or 1011 (31-bit output) is output for one clock according to the result of comparison of the current output branch destination address with the previous full-output branch destination address (that is output without being suspended due to next branch occurrence). Then, $\overline{\text{AUDSYNC}}$ is asserted and the branch destination address is output. The comparison address is H'00000000 in the initial status.

After a branch destination address is output, $\overline{\text{AUDSYNC}}$ is negated and 1100 (4-bit output), 1101 (8-bit output), 1110 (16-bit output), or 1111 (31-bit output) is output for one clock according to the result of comparison of the current branch-source trace data with the previous full-output branch-source trace data. The comparison address is H'00000000 in the initial state.

When the output cycle for branch-source trace data ends, $\overline{\text{AUDSYNC}}$ is negated and 0011 is output from the AUDATA.

If the next branch occurs while the trace data is output, the next branch trace data is output. Before this output, $\overline{\text{AUDSYNC}}$ is negated and 10** is output from the AUDATA (see figure 25.8, Example of Data Output at Successive Branch Occurrence). At this time, the current output address is compared with the previous full-output address, not the output-suspended address. This is because an upper address of the output-suspended address is unclear.

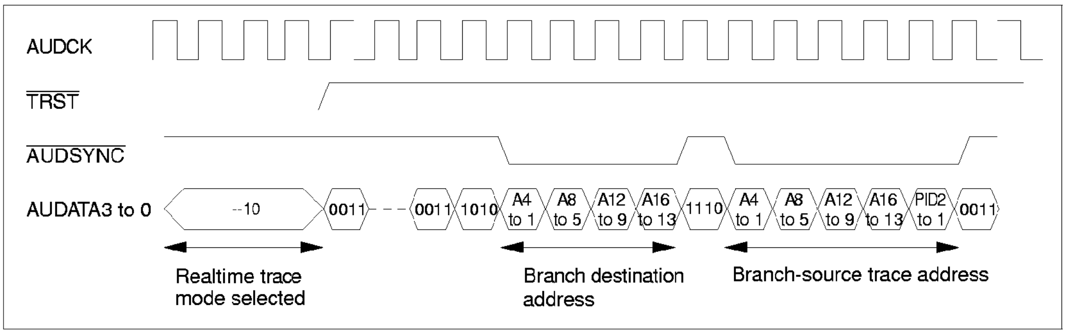


Figure 25.7 Example of Data Output

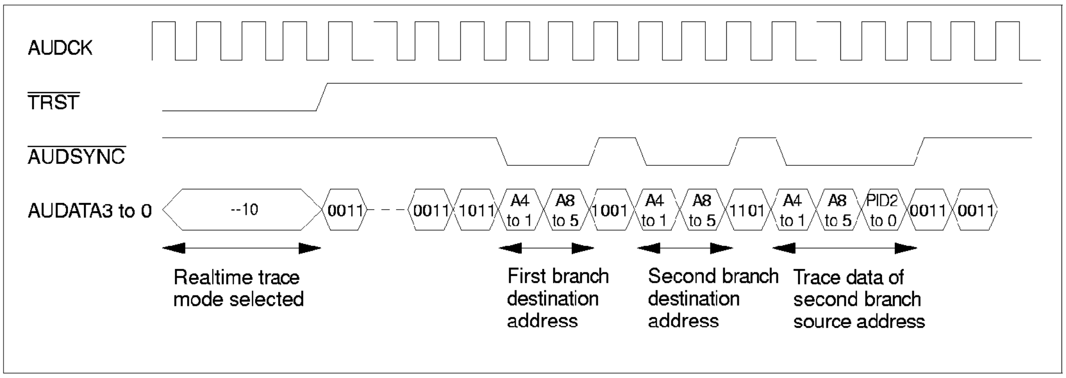

AUDCK
TRST
AUDSYNC
AUDATA3 to 0
--10
0011
0011
1011
A4 to 1
A8 to 5
1001
A4 to 1
A8 to 5
1101
A4 to 1
A8 to 5
PID2 to 0
0011
0011
Realtime trace mode selected
First branch destination address
Second branch destination address
Trace data of second branch source address

Figure 25.8 Example of Data Output at Successive Branch Occurrence

25.9 Full Trace Mode

25.9.1 Overview

In this modc, a branch dcstination addrcss and branch-sourcc tracc data arc output whcn branch occurs in user program due to execution of a branch instruction or interrupt occurrence. The CPU stops operating while the AUD is outputting trace data.

25.9.2 Description on Operation

When $\overline{\text{TRST}}$ is asserted, AUDATA1 and 0 are set to 0 and 1, respectively, and the AUD starts operating in the full trace mode.

Figurc 25.9 shows thc cxamplc of data output.

The output data format is the same as that in the realtime trace mode.

This mode differs from realtime trace mode in that the CPU stops operating while AUD is outputting trace data and trace data output is not suspended.

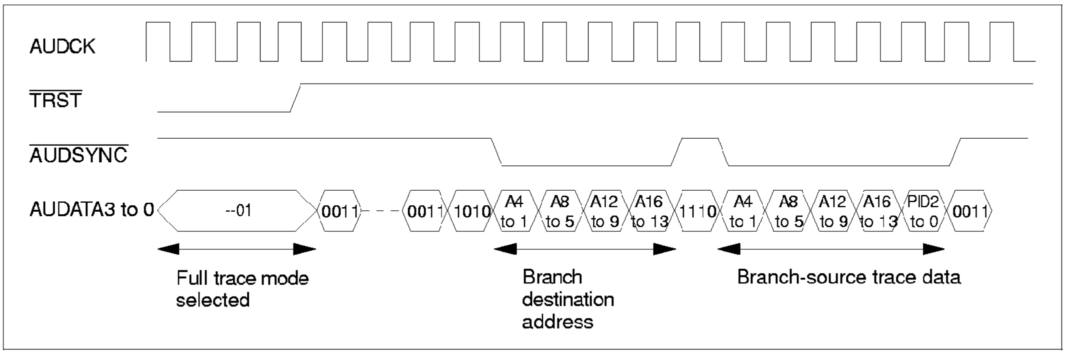


Figure 25.9 Example of Data Output

25.10 Notes on Use

Buffers and processing states in the AUD are initialized by one of the following conditions:

- Powcr-on rcsct
- Hardware standby
- Low level input to $\overline{\text{TRST}}$

During the power-on reset, the PC will jump to reset handle routine, of which address is H'A0000000. The AUD will output branch destination address as H'A0000000 and branch source address as H'00000000. Since the branch source is initialized as 0, and the branch source address and previous branch source address is the same, the AUD will treat it as 4-bit output plus PID-bit output (i.e. 2 AUD clock cycles). Due to the same reason, if branch destination is the same as the previous branch destination address, there will be 4-bit output also.

The trace mode is determined around the negated $\overline{\text{TRST}}$ area. AUD data pins will be tri-stated one or two cycles, then go into 0011 as idle state (i.e. waiting the branch occurs(.

The AUD logic is designed to be run at greater than 100 MHz internally. But, the external pins loading (AUDATA pins and $\overline{\text{AUDSYNC}}$ pin) could affect the AUD clock speed. The recommended maximum AUD clock frequency is less or equal to the CPU clock, and the minimum AUD clock frequency is one quarter of the CPU frequency.

In both realtime and full trace modes, if the TLB miss or TLB error occurs right after branch instruction fetch, the AUDATA will not output the branch source and destination address because this branch instruction is not executed.

If this occurs in the full trace mode, the A1_AUDSTALL signal will be negated right after the TLB miss or TLB error occurs, to the CPU can jump to the TLB invalid service routine.

Section 26 Electrical Characteristics

26.1 Absolute Maximum Ratings

Table 26.1 shows the absolute maximum ratings.

Table 26.1Absolute Maximum Ratings **— Preliminary —**

Item	Symbol	Rating	Unit
Power supply voltage (I/O)	VccQ	–0.3 to 4.2	V
Power supply voltage (internal)	Vcc Vcc – PLL1 Vcc – PLL2 Vcc – RTC	–0.3 to 2.5	V
Input voltage (except port L)	Vin	–0.3 to VccQ + 0.3	V
Input voltage (port L)	Vin	–0.3 to AVcc + 0.3	V
Analog power-supply voltage	AVcc	–0.3 to 4.6	V
Analog input voltage	V_{AN}	–0.3 to AVcc + 0.3	V
Operating temperature	Topr	–20 to 75	°C
Storage temperature	Tstr	–55 to 125	°C

Caution: Operating the chip in excess of the absolute maximum rating may result in permanent damage.

26.2 DC Characteristics

Tables 26.2 and 26.3 list DC characteristics.

Table 26.2DC Characteristics (Ta = –20 to 75°C) **— Preliminary —**

Item	Symbol	Min	Typ	Max	Unit	Measurement Conditions
Power supply voltage	VccQ	3.0	3.3	3.6	V	
	Vcc	1.6	1.8	2.0		

Table 26.2DC Characteristics (Ta = –20 to 75°C) (cont) **— Preliminary —**

Item		Symbol	Min	Typ	Max	Unit	Measurement Conditions
Current dissipation	Normal operation	Icc	—	TBD*1	TBD*1	mA	VccQ = 3.3 V, Vcc = 1.8 V
			—	TBD*2	TBD*2		*1: 133 MHz (Bø: 66 MHz)
			—	TBD*3	TBD*3		*2: 40 MHz
	In sleep mode*4		—	TBD*1	TBD*1		*3: 20 MHz
			—	TBD*2	TBD*2		*4: No external bus cycles
			—	TBD*3	TBD*3		except refresh cycles
	In standby mode		—	TBD	TBD	µA	Ta = 25°C (RTC on)
			—	TBD	TBD		Ta > 50°C (RTC on)
			—	TBD	TBD		Ta = 25°C (RTC off)
			—	—	TBD		Ta > 50°C (RTC off)
Input high voltage	$\overline{\text{RESETP}}$, $\overline{\text{RESETM}}$, NMI	VIH	VccQ × 0.9	—	VccQ + 0.3	V	
	$\overline{\text{BREQ}}$, $\overline{\text{IRQ5}}$–		VccQ – 0.5	—	VccQ + 0.3		In standby mode
	$\overline{\text{IRQ0}}$, MD5–MD0		VccQ – 0.7	—	VccQ + 0.3		Normal operation
	EXTAL, CKIO		VccQ – 0.7	—	VccQ + 0.3		
	Port L		2.0	—	AVcc + 0.3		
	Other input pins		2.0	—	VccQ + 0.3		

Table 26.2DC Characteristics (Ta = –20 to 75°C) (cont) **— Preliminary —**

Item		Symbol	Min	Typ	Max	Unit	Measurement Conditions
Input low voltage	$\overline{\text{RESETP}}$, $\overline{\text{RESETM}}$, NMI	VIL	–0.3	—	VccQ × 0.1	V	
	$\overline{\text{BREQ}}$, IRQ5–		–0.3	—	0.5		In standby mode
	IRQ0, MD5–MD0		–0.3	—	VccQ × 0.2		Normal operation
	Port L		–0.3	—	AVcc × 0.2		
	Other input pins		–0.3	—	VccQ × 0.2		
Input leak current	All input pins	I Iin I	—	—	1.0	µA	Vin = 0.5 to VccQ–0.5 V
Three-state leak current	I/O, all output pins (off condition)	I Isti I	—	—	1.0	µA	Vin = 0.5 to VccQ–0.5 V
Output high voltage	All output pins	VOH	2.4	—	—	V	VccQ = 3.0 V, IOH = –200 µA
			2.0	—	—		VccQ = 3.0 V, IOH = –2 mA
Output low voltage	All output pins	VOL	—	—	0.55		VccQ = 3.6 V, IOL = 1.6 mA
Pull-up resistance	Port pin	Ppull	30	60	120	kΩ	
Pin capacity	All pins	C	—	—	20	PF	
Analog power-supply voltage		AVcc	3.0	3.3	3.6	V	

Table 26.2DC Characteristics (Ta = –20 to 75°C) (cont) — Preliminary —

Item		Symbol	Min	Typ	Max	Unit	Measurement Conditions
Analog power-supply	During A/D conversion	AIcc	—	0.8	2	mA	
current	During A/D and D/A conversion		—	2.4	6	mA	
	Idle		—	0.01	5.0	µA	
RAM standby voltage		VRAM	2.0	—	—	V	

Notes: 1. Regardless of whether PLL or RTC is used, connect Vcc – PLL and Vcc – RTC to Vcc, and Vss – PLL and Vss – RTC to Vss.

2. AVcc must be under condition of VccQ – 0.3 V ≤ AVcc ≤ VccQ + 0.3 V. If the A/D and D/A converters are not used, do not leave the AVcc and AVss pins open. Connect AVcc to VccQ, and connect AVss to VssQ.

3. Current dissipation values shown are the values at which all output pins are without load under conditions of VIHmin = VccQ – 0.5 V, VILmax = 0.5 V.

Table 26.3Permitted Output Current Values — Preliminary —
(VccQ = 3.3 ± 0.3 V, Vcc = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

Item	Symbol	Min	Typ	Max	Unit
Output low-level permissible current (per pin)	IOL	—	—	2.0	mA
Output low-level permissible current (total)	∑ IOL	—	—	120	mA
Output high-level permissible current (per pin)	–IOH	—	—	2.0	mA
Output high-level permissible current (total)	∑ (–IOH)	—	—	40	mA

Caution: To ensure LSI reliability, do not exceed the value for output current given in table 26.3.

26.3 AC Characteristics

In general, inputting for this LSI should be clock synchronous. Keep the setup and hold times for each input signal unless otherwise specified.

Table 26.4Maximum Operating Frequencies — Preliminary — (VccQ = 3.3 ± 0.3 V, VccQ = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

Item		Symbol	Min	Typ	Max	Unit	Remarks
Operating frequency	CPU, cache, TLB	f	16	—	133	MHz	
	External bus		16	—	66		
	Peripheral module		4	—	33		

26.3.1 Clock Timing

Table 26.5Clock Timing (1) **— Preliminary —**

(VccQ = 3.3 ± 0.3 V, VccQ = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C, Maximum External Bus Operating Frequency: 20 MHz)

Item	Symbol	Min	Max	Unit	Figure
EXTAL clock input frequency	f_{EX}	5	20	MHz	26.1
EXTAL clock input cycle time	t_{EXcyc}	50	200	ns	
EXTAL clock input low pulse width	t_{EXL}	8	—	ns	
EXTAL clock input high pulse width	t_{EXH}	8	—	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	20	MHz	26.2
CKIO clock input cycle time	t_{CKIcyc}	50	50	ns	
CKIO clock input low pulse width	t_{CKIL}	8	—	ns	
CKIO clock input high pulse width	t_{CKIH}	8	—	ns	
CKIO clock input rise time	t_{CKIR}	—	4	ns	
CKIO clock input fall time	t_{CKIF}	—	4	ns	
CKIO clock output frequency	f_{OP}	16	20	MHz	26.3
CKIO clock output cycle time	t_{cyc}	50	62.5	ns	
CKIO clock output low pulse width	t_{CKOL}	20	—	ns	
CKIO clock output high pulse width	t_{CKOH}	20	—	ns	
CKIO clock output rise time	t_{CKOR}	—	7	ns	
CKIO clock output fall time	t_{CKOF}	—	—	ns	
Power-on oscillation settling time	t_{OSC1}	10	—	ms	26.4
$\overline{\text{RESETP}}$ setup time	t_{RESPS}	20	—	ns	26.4, 26.5
$\overline{\text{RESETM}}$ setup time	t_{RESMS}	0	—	ns	
$\overline{\text{RESETP}}$ assert time	t_{RESPW}	20	—	tcyc	
$\overline{\text{RESETM}}$ assert time	t_{RESMW}	20	—	tcyc	
Standby return oscillation settling time 1	t_{OSC2}	—	10	ms	26.5
Standby return oscillation settling time 2	t_{OSC3}	—	10	ms	26.6
Standby return oscillation settling time 3	t_{OSC4}	—	10	ms	26.7

Table 26.5Clock Timing (1) (cont) — Preliminary —
(VccQ = 3.3 ± 0.3 V, Vcc = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C, Maximum External Bus Operating Frequency: 20 MHz)

Item	Symbol	Min	Max	Unit	Figure
PLL synchronization settling time 1 (PLL1)	t_{PLL1}	100	—	µs	26.8, 26.10
PLL synchronization settling time 2 (PLL2)	t_{PLL2}	100	—	µs	26.9, 26.11
IRQ interrupt determination time (RTC used and standby mode)	t_{IRQSTB}	100	—	µs	26.10

Table 26.5Clock Timing (2) **— Preliminary —**

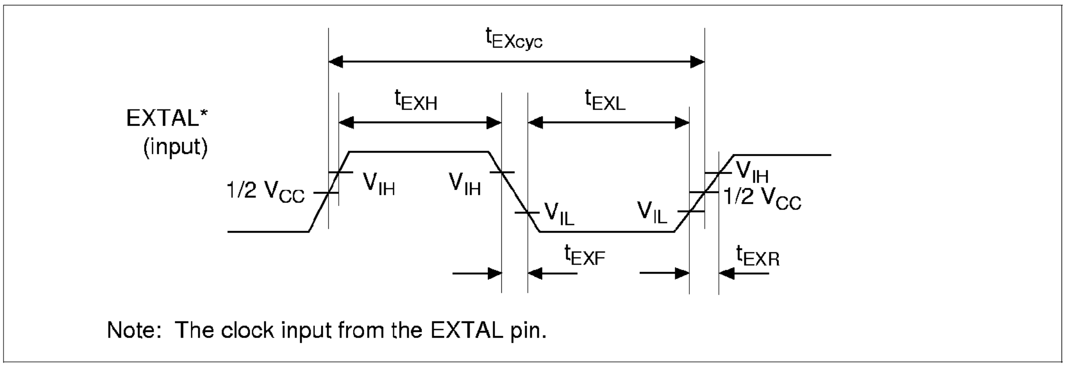
(VccQ = 3.3 ± 0.3 V, Vcc = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C, Maximum External Bus Operating Frequency: 33 MHz)

Item	Symbol	Min	Max	Unit	Figure
EXTAL clock input frequency	f_{EX}	5	33	MHz	26.1
EXTAL clock input cycle time	t_{EXcyc}	30.3	200	ns	
EXTAL clock input low pulse width	t_{EXL}	7	—	ns	
EXTAL clock input high pulse width	t_{EXH}	7	—	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	33	MHz	26.2
CKIO clock input cycle time	t_{CKIcyc}	30.3	50	ns	
CKIO clock input low pulse width	t_{CKIL}	7	—	ns	
CKIO clock input high pulse width	t_{CKIH}	7	—	ns	
CKIO clock input rise time	t_{CKIR}	—	3	ns	
CKIO clock input fall time	t_{CKIF}	—	3	ns	
CKIO clock output frequency	f_{OP}	16	33	MHz	26.3
CKIO clock output cycle time	t_{cyc}	30.3	50	ns	
CKIO clock output low pulse width	t_{CKOL}	8	—	ns	
CKIO clock output high pulse width	t_{CKOH}	8	—	ns	
CKIO clock output rise time	t_{CKOR}	—	6	ns	
CKIO clock output fall time	t_{CKOF}	—	6	ns	
Power-on oscillation settling time	t_{OSC1}	10	—	ms	26.4
$\overline{\text{RESETP}}$ setup time	t_{RESPS}	20	—	ns	26.4, 26.5
$\overline{\text{RESETM}}$ setup time	t_{RESMS}	0	—	ns	
$\overline{\text{RESETP}}$ assert time	t_{RESPW}	20	—	tcyc	
$\overline{\text{RESETM}}$ assert time	t_{RESMW}	20	—	tcyc	
Standby return oscillation settling time 1	t_{OSC2}	10	—	ms	26.5
Standby return oscillation settling time 2	t_{OSC3}	10	—	ms	26.6
Standby return oscillation settling time 3	t_{OSC4}	11	—	ms	26.7
PLL synchronization settling time	t_{PLL}	100	—	µs	26.8, 26.9, 26.10
IRQ interrupt determination time (RTC used and standby mode)	t_{IRQSTB}	100	—	µs	26.10

Table 26.5Clock Timing (3) **— Preliminary —**

(VccQ = 3.3 ± 0.3 V, Vcc = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C, Maximum External Bus Operating Frequency: 66 MHz)

Item	Symbol	Min	Max	Unit	Figure
EXTAL clock input frequency	f_{EX}	5	66	MHz	26.1
EXTAL clock input cycle time	t_{EXcyc}	15.2	200	ns	
EXTAL clock input low pulse width	t_{EXL}	4	—	ns	
EXTAL clock input high pulse width	t_{EXH}	4	—	ns	
EXTAL clock input rise time	t_{EXR}	—	2	ns	
EXTAL clock input fall time	t_{EXF}	—	2	ns	
CKIO clock input frequency	f_{CKI}	20	66	MHz	26.2
CKIO clock input cycle time	t_{CKIcyc}	15.2	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}	16	66	MHz	26.3
CKIO clock output cycle time	t_{cyc}	15.2	62.5	ns	
CKIO clock output low pulse width	t_{CKOL}	3	—	ns	
CKIO clock output high pulse width	t_{CKOH}	3	—	ns	
CKIO clock output rise time	t_{CKOR}	—	5	ns	
CKIO clock output fall time	t_{CKOF}	—	5	ns	
Power-on oscillation settling time	t_{OSC1}	10	—	ms	26.4
$\overline{\text{RESETP}}$ setup time	t_{RESPS}	20	—	ns	26.4, 26.5
$\overline{\text{RESETM}}$ setup time	t_{RESMS}	0	—	tcyc	
$\overline{\text{RESETP}}$ assert time	t_{RESPW}	20	—	tcyc	
$\overline{\text{RESETM}}$ assert time	t_{RESMW}	20	—	ns	
Standby return oscillation settling time 1	t_{OSC2}	10	—	ms	26.5
Standby return oscillation settling time 2	t_{OSC3}	10	—	ms	26.6
Standby return oscillation settling time 3	t_{OSC4}	11	—	ms	26.7
PLL synchronization settling time	t_{PLL}	100	—	µs	26.8, 26.9, 26.10
IRQ interrupt determination time (RTC used and standby mode)	t_{IRQSTB}	100	—	µs	26.10



Note: The clock input from the EXTAL pin.

Figure 26.1 EXTAL Clock Input Timing

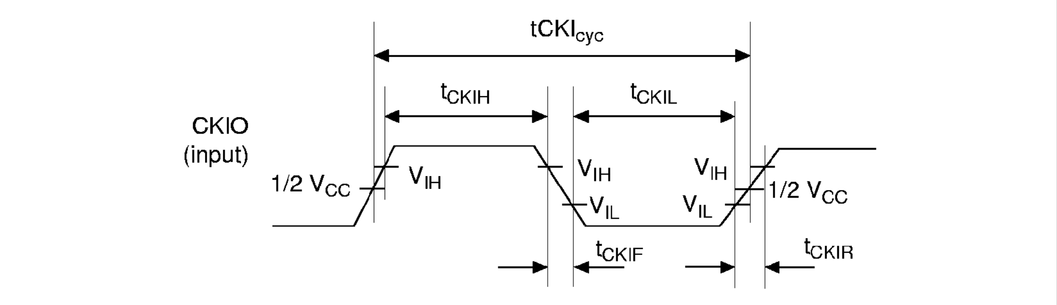


Figure 26.2 CKIO Clock Input Timing

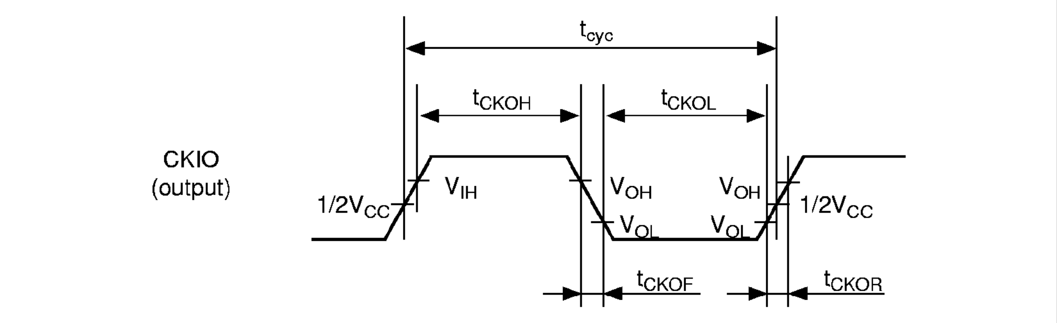
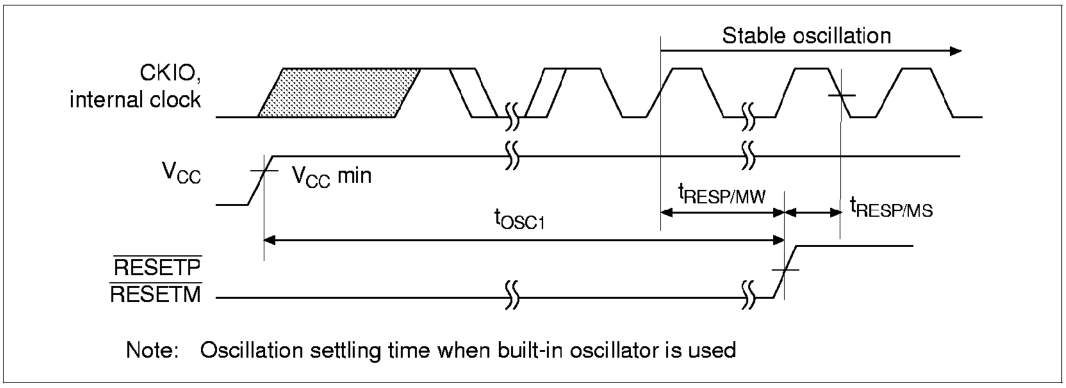
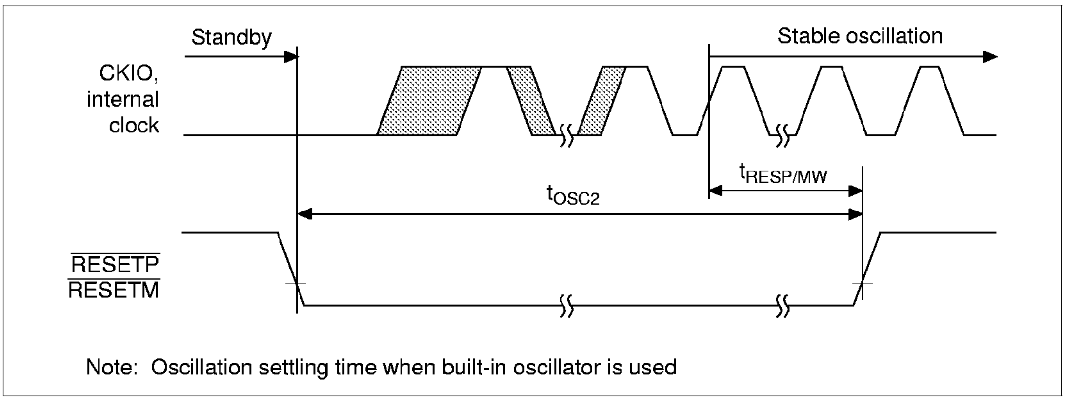


Figure 26.3 CKIO Clock Output Timing



Note: Oscillation settling time when built-in oscillator is used

Figure 26.4 Power-on Oscillation Settling Time



Note: Oscillation settling time when built-in oscillator is used

Figure 26.5 Oscillation Settling Time at Standby Return (Return by Reset)

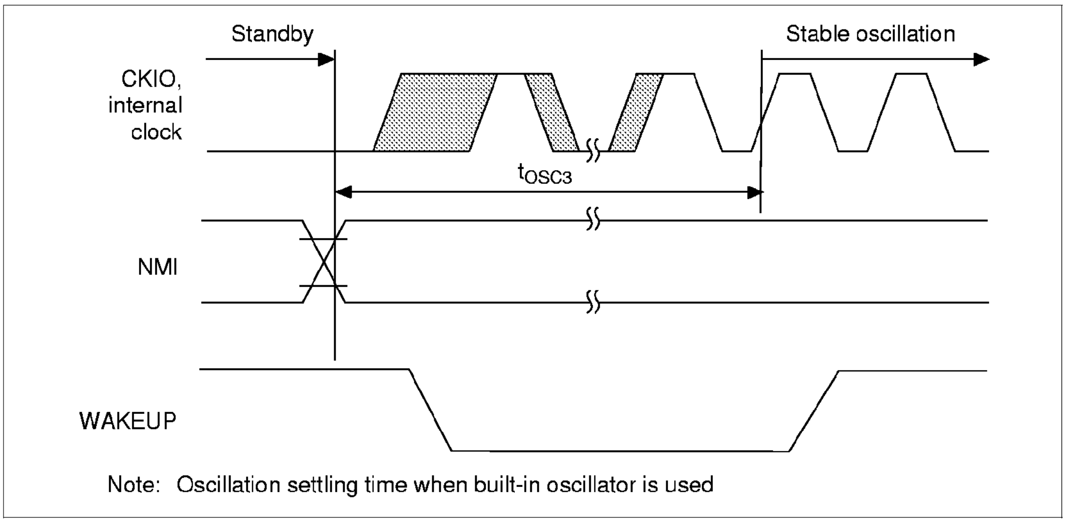

Standby
Stable oscillation
CKIO, internal clock
t_{OSC3}
NMI
WAKEUP
Note: Oscillation settling time when built-in oscillator is used

Figure 26.6 Oscillation Settling Time at Standby Return (Return by NMI)

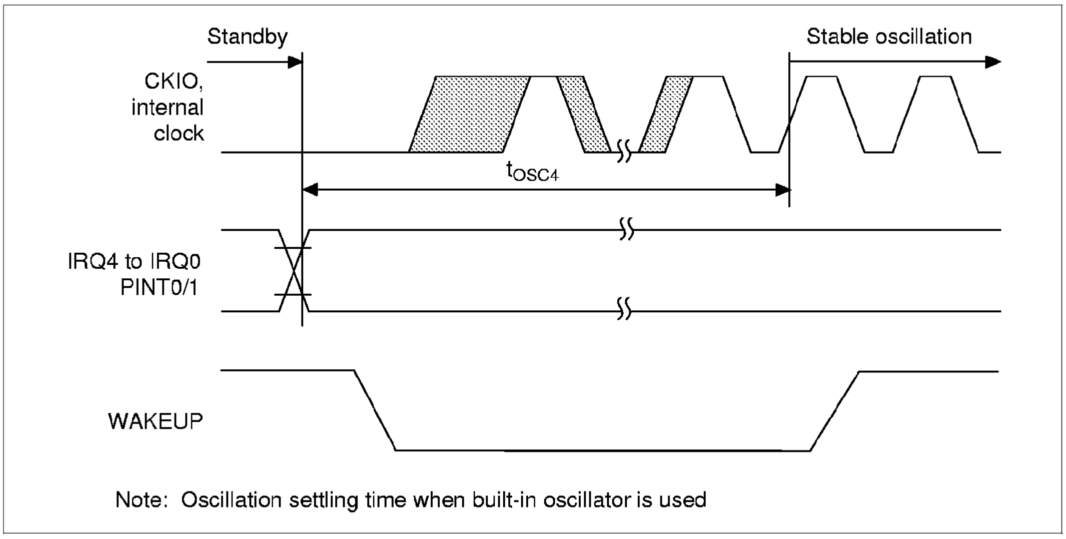


Figure 26.7 Oscillation Settling Time at Standby Return (Return by IRQ4 to IRQ0)

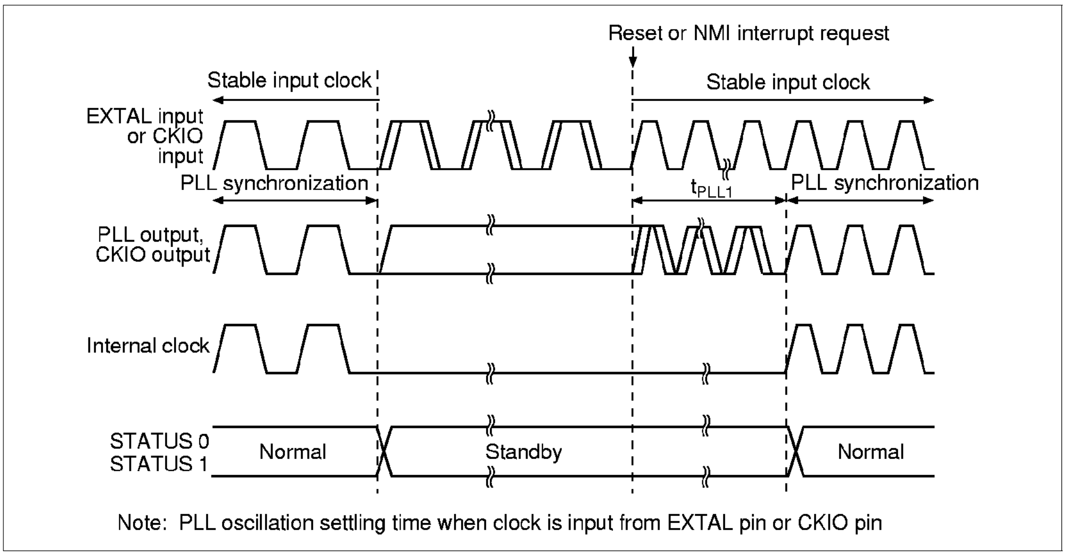


Figure 26.8 PLL Synchronization Settling Time by Reset or NMI

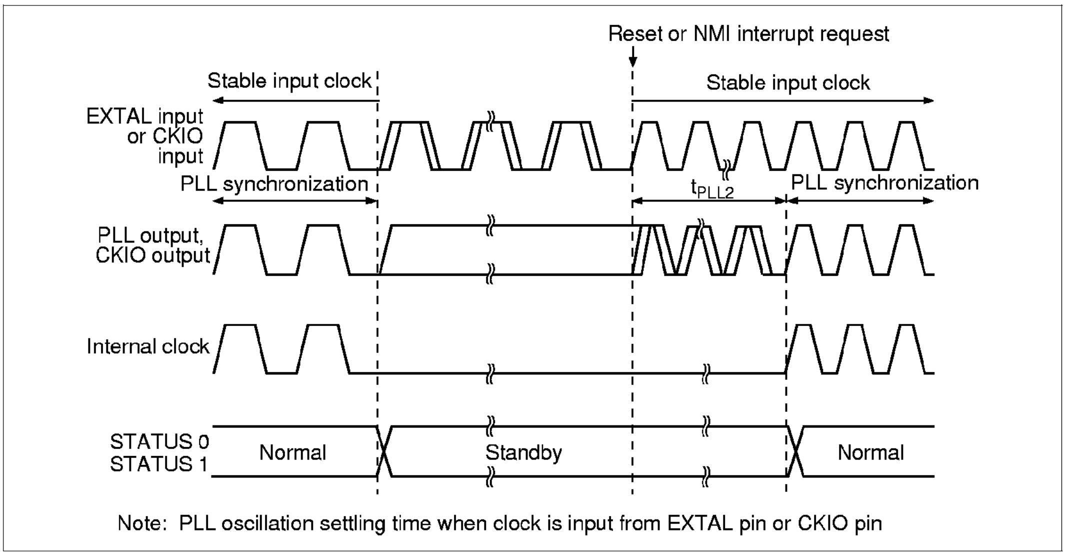
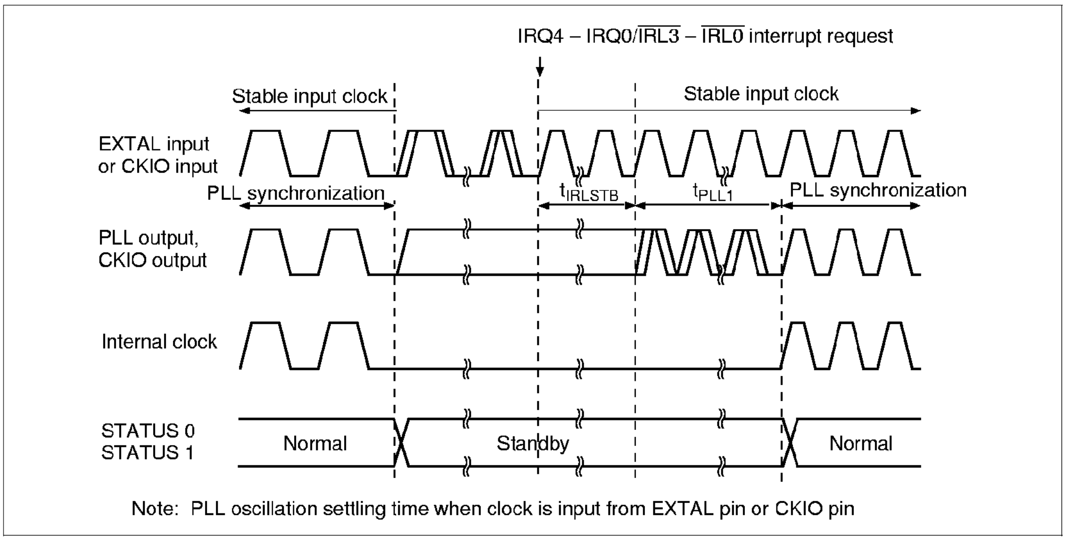
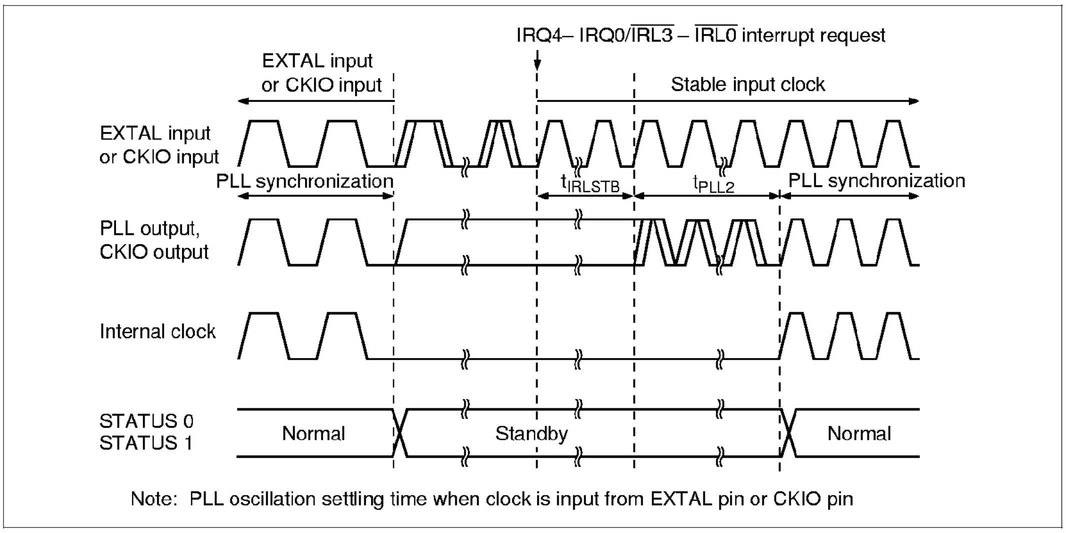


Figure 26.9 PLL Synchronization Settling Time by Reset or NMI



Note: PLL oscillation settling time when clock is input from EXTAL pin or CKIO pin

Figure 26.10 PLL Synchronization Settling Time by IRQ/IRL Interrupt



Note: PLL oscillation settling time when clock is input from EXTAL pin or CKIO pin

Figure 26.11 PLL Synchronization Settling Time by IRQ/IRL Interrupt

26.3.2 Control Signal Timing

Table 26.6Control Signal Timing — **Preliminary** —

(Vcc = 3.3 ± 0.3 V, Vcc (1.8 V) = 3.3 ± 0.3 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

		–66*2			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Figure**
$\overline{\text{RESETP}}$ pulse width	t_{RESPW}	20 *3	—	tcyc	26.12,
$\overline{\text{RESETP}}$ setup time*1	t_{RESPS}	23	—	ns	26.13
$\overline{\text{RESETP}}$ hold time	t_{RESPH}	2	—	ns	
$\overline{\text{RESETM}}$ pulse width	t_{RESMW}	12 *4	—	tcyc	
$\overline{\text{RESETM}}$ setup time	t_{RESMS}	3	—	ns	
$\overline{\text{RESETM}}$ hold time	t_{RESMH}	34	—	ns	
$\overline{\text{BREQ}}$ setup time	t_{BREQS}	12	—	ns	26.15
$\overline{\text{BREQ}}$ hold time	t_{BREQH}	3	—	ns	
NMI setup time *1	t_{NMIS}	10	—	ns	26.13,
NMI hold time	t_{NMIH}	4	—	ns	26.14
IRQ5–IRQ0 setup time *1	t_{IRQS}	10	—	ns	
IRQ5–IRQ0 hold time	t_{IRQH}	4	—	ns	
$\overline{\text{IRQOUT}}$ delay time	t_{IRQOD}	—	12	ns	
$\overline{\text{BACK}}$ delay time	t_{BACKD}	—	12	ns	26.15,
STATUS1, STATUS0 delay time	t_{STD}	—	16	ns	26.16
Bus tri-state delay time 1	t_{BOFF1}	0	15	ns	
Bus tri-state delay time 2	t_{BOFF2}	0	15	ns	
Bus buffer-on time 1	t_{BON1}	0	15	ns	
Bus buffer-on time 2	t_{BON2}	0	15	ns	

Notes: 1. $\overline{\text{RESETP}}$, NMI, and IRQ5 to IRQ0 are asynchronous. Changes are detected at the clock fall when the setup shown is used. When the setup cannot be used, detection can be delayed until the next clock falls.

2. The upper limit of the external bus clock is 66 MHz.

3. In the standby mode, $t_{RESPW} = t_{OSC2}$ (10 ms). In the sleep mode, $t_{RESPW} = t_{PLL1}$ (100 µs). When the clock multiplication ratio is changed, $t_{RESPW} = t_{PLL1}$ (100 µs).

4. In the standby mode, $t_{RESMW} = t_{OSC2}$ (10 ms). In the sleep mode, RESETM must be kept low until STATUS (0-1) changes to reset (HH). When the clock multiplication ratio is changed, RESETM must be kept low until STATUS (0-1) changes to reset (HH).

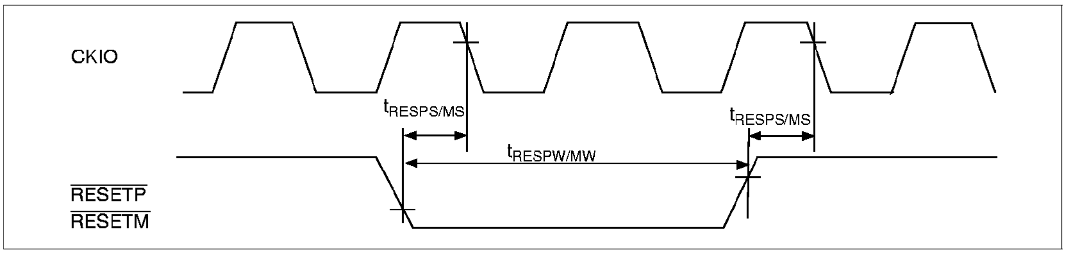


Figure 26.12 Reset Input Timing

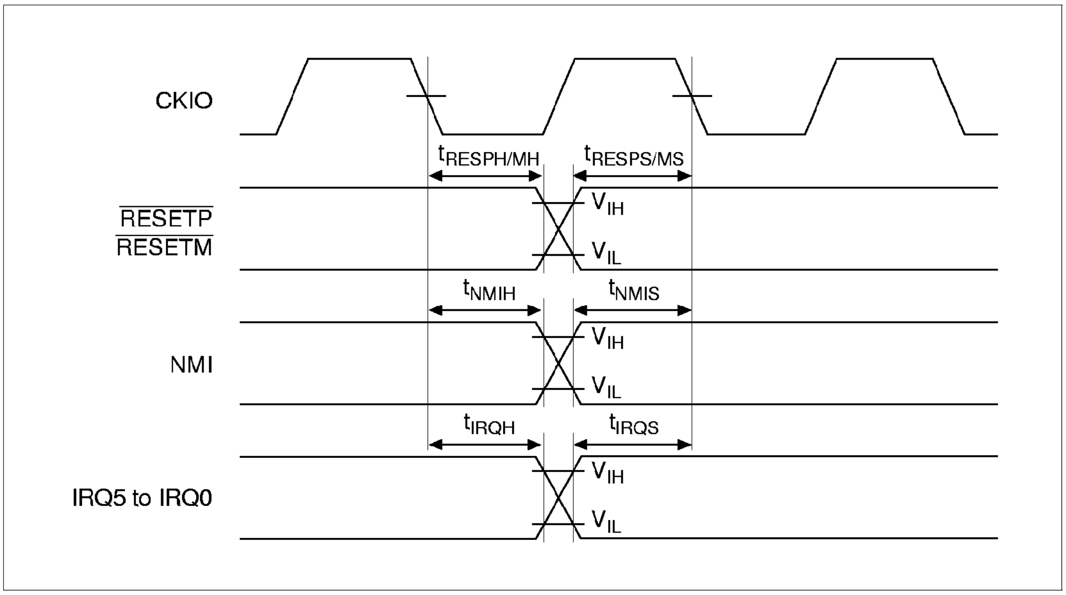


Figure 26.13 Interrupt Signal Input Timing

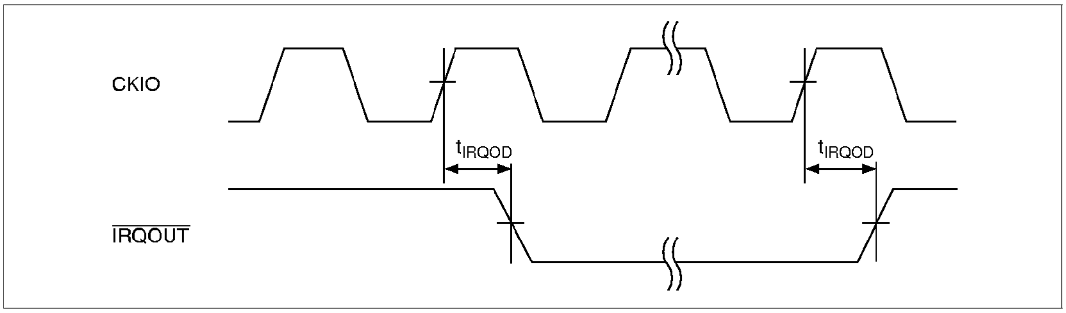
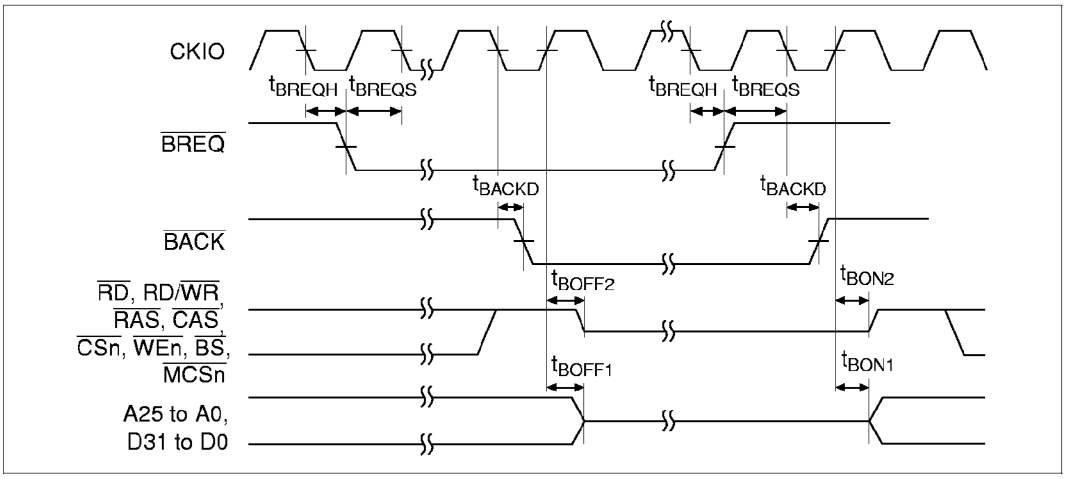
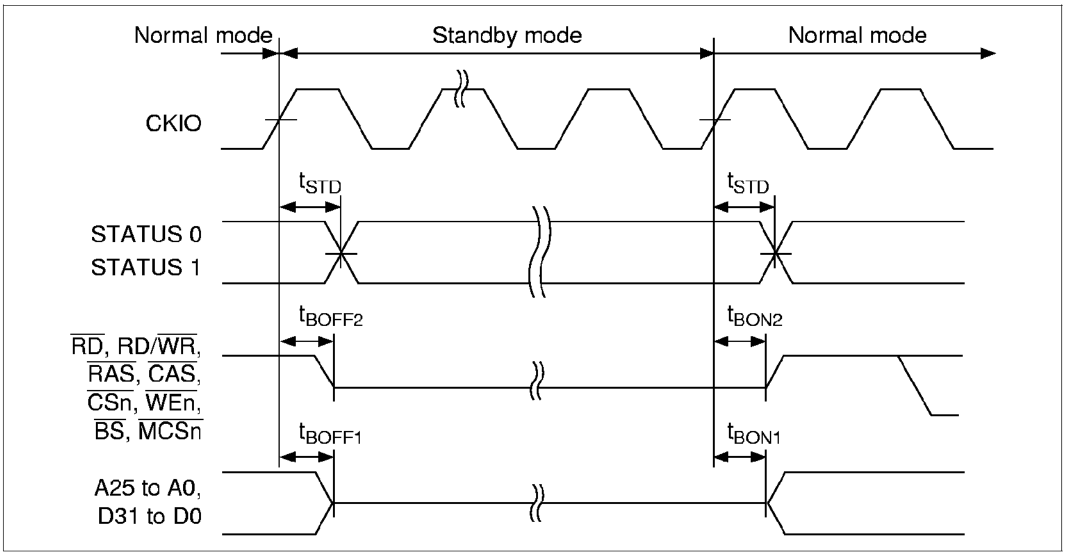


Figure 26.14 $\overline{IRQOUT}$ Timing



CKIO
t_{BREQH}
t_{BREQS}
$\overline{BREQ}$
t_{BACKD}
$\overline{BACK}$
t_{BOFF2}
t_{BON2}
$\overline{RD}$, RD/$\overline{WR}$, $\overline{RAS}$, $\overline{CAS}$, $\overline{CSn}$, $\overline{WEn}$, $\overline{BS}$, $\overline{MCSn}$
t_{BOFF1}
t_{BON1}
A25 to A0, D31 to D0

Figure 26.15 Bus Release Timing



Normal mode
Standby mode
Normal mode
CKIO
t_{STD}
STATUS 0
STATUS 1
t_{BOFF2}
t_{BON2}
$\overline{RD}$, RD/$\overline{WR}$, $\overline{RAS}$, $\overline{CAS}$, $\overline{CSn}$, $\overline{WEn}$, $\overline{BS}$, $\overline{MCSn}$
t_{BOFF1}
t_{BON1}
A25 to A0, D31 to D0

Figure 26.16 Pin Drive Timing at Standby

26.3.3 AC Bus Timing

Table 26.7Bus Timing — Preliminary —
(Clock Modes 0/1/2/3/4/7, VccQ = 3.3 ± 0.3 V, Vcc = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

		–66*			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Figure**
Address delay time	t_{AD}	1	13	ns	26.17–26.30, 26.36–26.50, 26.53–26.60
Address setup time	t_{AS}	0	—	ns	26.23–26.30
Address hold time	t_{AH}	10	—	ns	26.17–26.30
$\overline{BS}$ delay time	t_{BSD}	—	12	ns	26.17–26.30, 26.36–26.50, 26.53–26.60
$\overline{CS}$ delay time 1	t_{CSD1}	1	12	ns	26.17–26.30, 26.36–26.50, 26.53–26.60
$\overline{CS}$ delay time 2	t_{CSD2}	1	12	ns	26.17–26.22
Read/write delay time	t_{RWD}	1	12	ns	26.17–26.30, 26.36–26.50, 26.53–26.60
Read/write hold time	t_{RWH}	0	—	ns	26.17–26.30
Read strobe delay time	t_{RSD}	—		ns	26.17–26.22, 26.54–26.60
Read data setup time 1	t_{RDS1}	12	—	ns	26.17–26.26, 26.54–26.60
Read data setup time 2	t_{RDS2}	7	—	ns	26.27–26.30, 26.36–26.39
Read data setup time 3	t_{RDS3}	12	—	ns	26.31
Read data setup time 4	t_{RDS4}	12	—	ns	26.32
Read data hold time 1	t_{RDH1}	0	—	ns	26.17–26.26, 26.54–26.60
Read data hold time 2	t_{RDH2}	3	—	ns	26.27–26.30, 26.36–26.39, 26.44–26.48
Read data hold time 3	t_{RDH3}	0	—	ns	26.31
Read data hold time 4	t_{RDH4}	6	—	ns	26.32
Write enable delay time	t_{WED}	—	12	ns	26.17–26.19, 26.54–26.55
Write data delay time 1	t_{WDD1}	—	14	ns	26.17–26.19, 26.54–26.55, 26.58–26.60
Write data delay time 2	t_{WDD2}	—	13	ns	26.23–26.30, 26.40–26.43
Write data setup time	t_{WDS}	0	—	ns	26.23–26.30

Table 26.7Bus Timing (cont) — Preliminary —

(Clock Modes 0/1/2/3/4/7, VccQ = 3.3 ± 0.3 V, Vcc = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

		–66*			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Figure**
Write data hold time 1	t_{WDH1}	2	—	ns	26.17–26.19, 26.23–26.30, 26.54–26.60
Write data hold time 2	t_{WDH2}	2	—	ns	26.40–26.43
Write data hold time 3	t_{WDH3}	2	—	ns	26.17–26.19, 26.23–26.30
Write data hold time 4	t_{WDH4}	2	—	ns	26.54–26.60
$\overline{\text{WAIT}}$ setup time	t_{WTS}	12	—	ns	26.18–26.22, 26.55, 26.57, 26.59, 26.60
$\overline{\text{WAIT}}$ setup time 2	t_{WTS2}	5	—	ns	26.19(2)
$\overline{\text{WAIT}}$ hold time	t_{WTH}	0	—	ns	26.18–26.22, 26.47, 26.57, 26.59, 26.60
$\overline{\text{RAS}}$ delay time 1	t_{RASD1}	1	13	ns	26.23–26.33
$\overline{\text{RAS}}$ delay time 2	t_{RASD2}	1	13	ns	26.27–26.30
$\overline{\text{CAS}}$ delay time 1	t_{CASD1}	1	13	ns	26.23–26.30, 26.33–26.25
$\overline{\text{CAS}}$ delay time 2	t_{CASD2}	1	13	ns	26.36–26.53
$\overline{\text{CAS}}$ delay time 3	t_{CASD3}	1	13	ns	26.31–26.32
$\overline{\text{CAS}}$ delay time 4	t_{CASD4}	1	13	ns	26.31–26.32
$\overline{\text{DQM}}$delay time	t_{DQMD}	1	12	ns	26.36–26.50
$\overline{\text{ICIORD}}$ delay time	t_{ICRSD}	—	12	ns	26.58–26.60
$\overline{\text{ICIOWR}}$ delay time	t_{ICWSD}	—	12	ns	26.58–26.60
$\overline{\text{IOIS16}}$ setup time	t_{IO16S}	12	—	ns	26.59, 26.60
$\overline{\text{IOIS16}}$ hold time	t_{IO16H}	4	—	ns	26.59, 26.60
$\overline{\text{DACK}}$ delay time 1	t_{DAKD1}	—	12	ns	26.17–26.32, 26.54–26.60
$\overline{\text{DACK}}$ delay time 2	t_{DAKD2}	—	12	ns	26.31, 26.32

Note: The upper limit of the external bus clock is 66 MHz.

26.3.4 Basic Timing

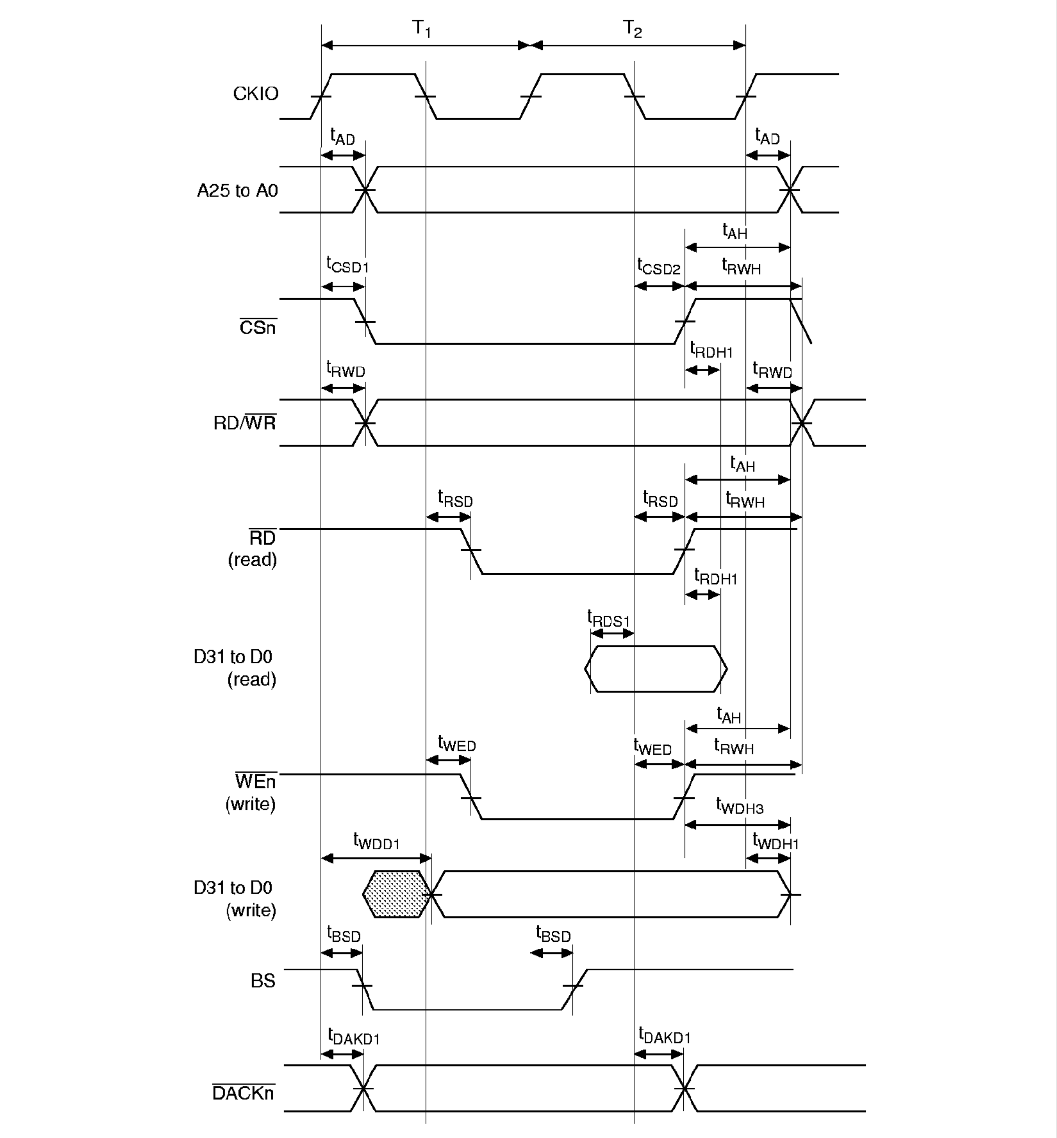


Figure 26.17 Basic Bus Cycle (No Wait)

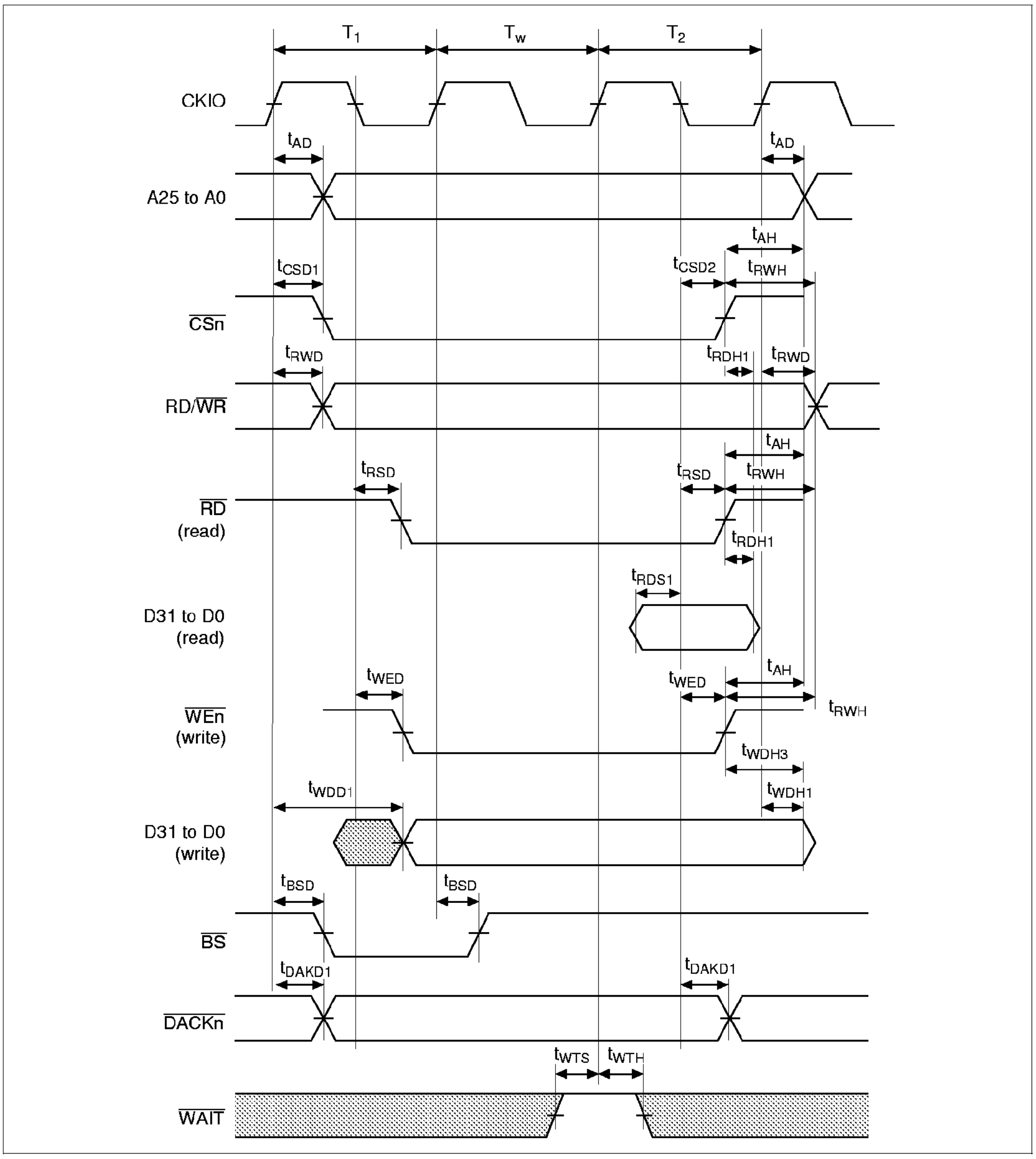


Figure 26.18 Basic Bus Cycle (One Wait)

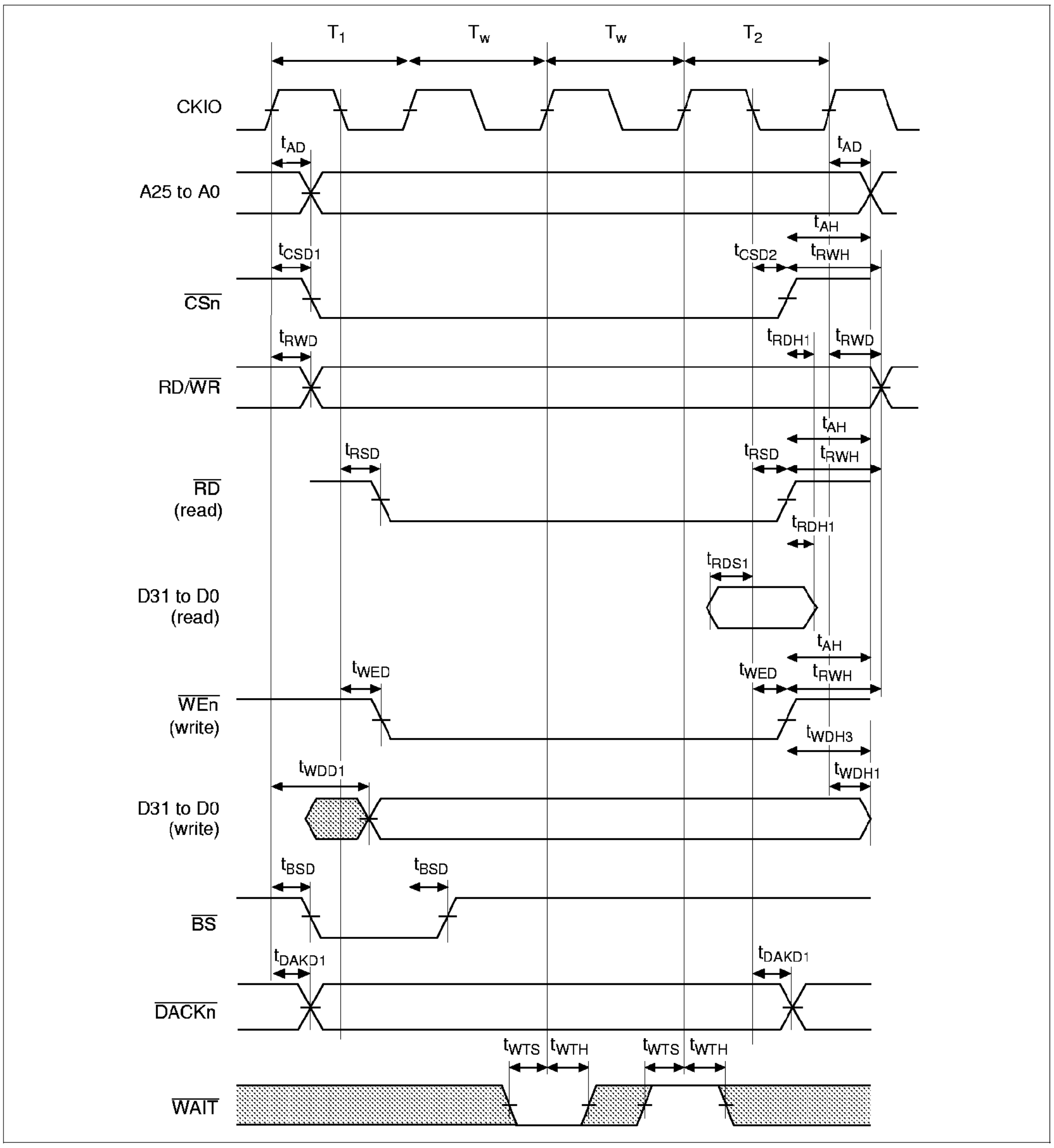


Figure 26.19 (1) Basic Bus Cycle (External Wait, WAITSEL = 0)

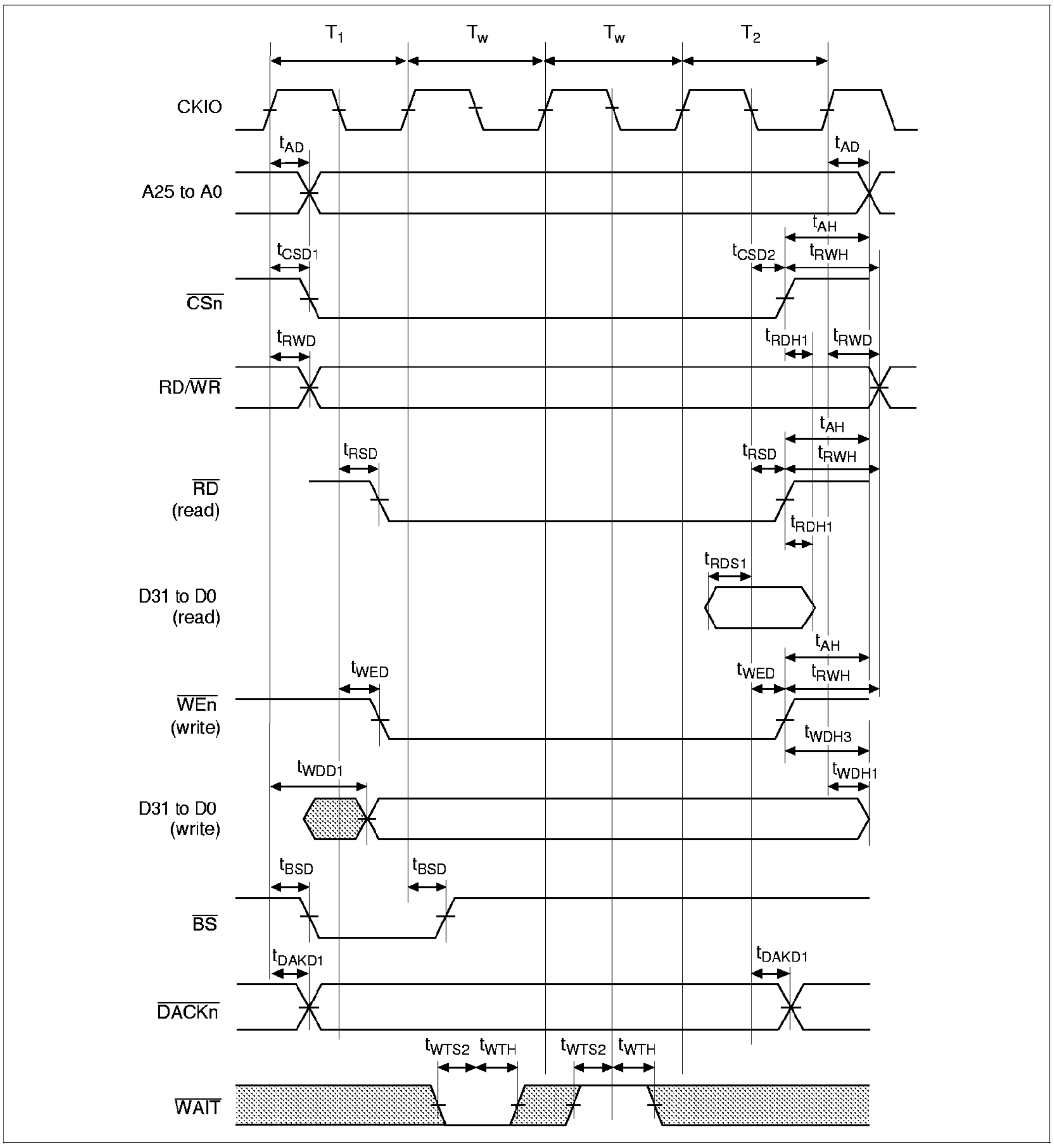
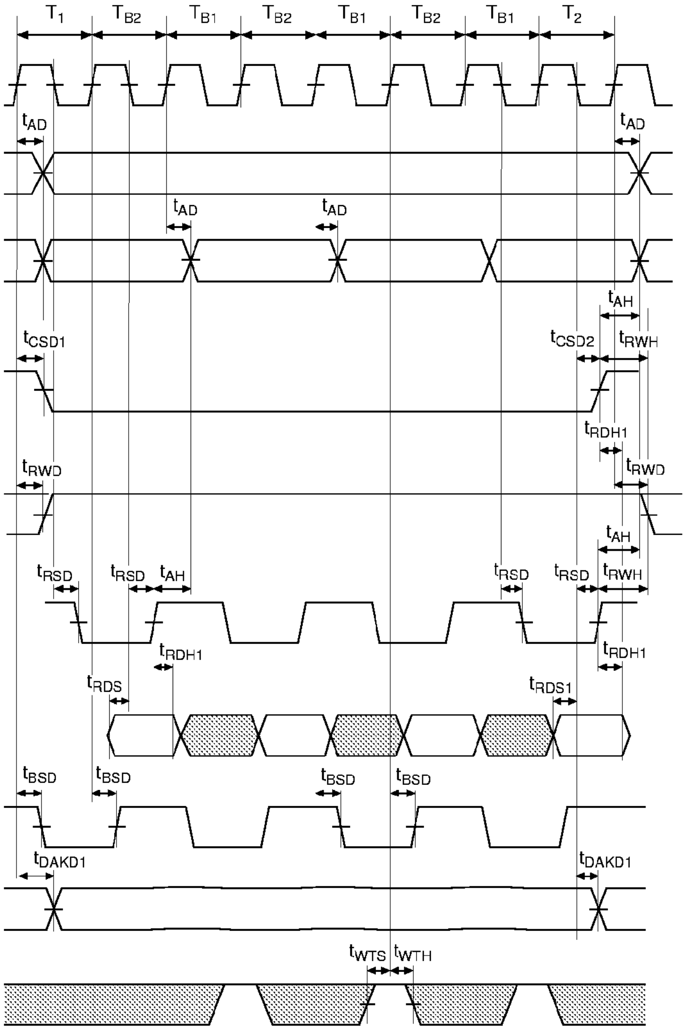


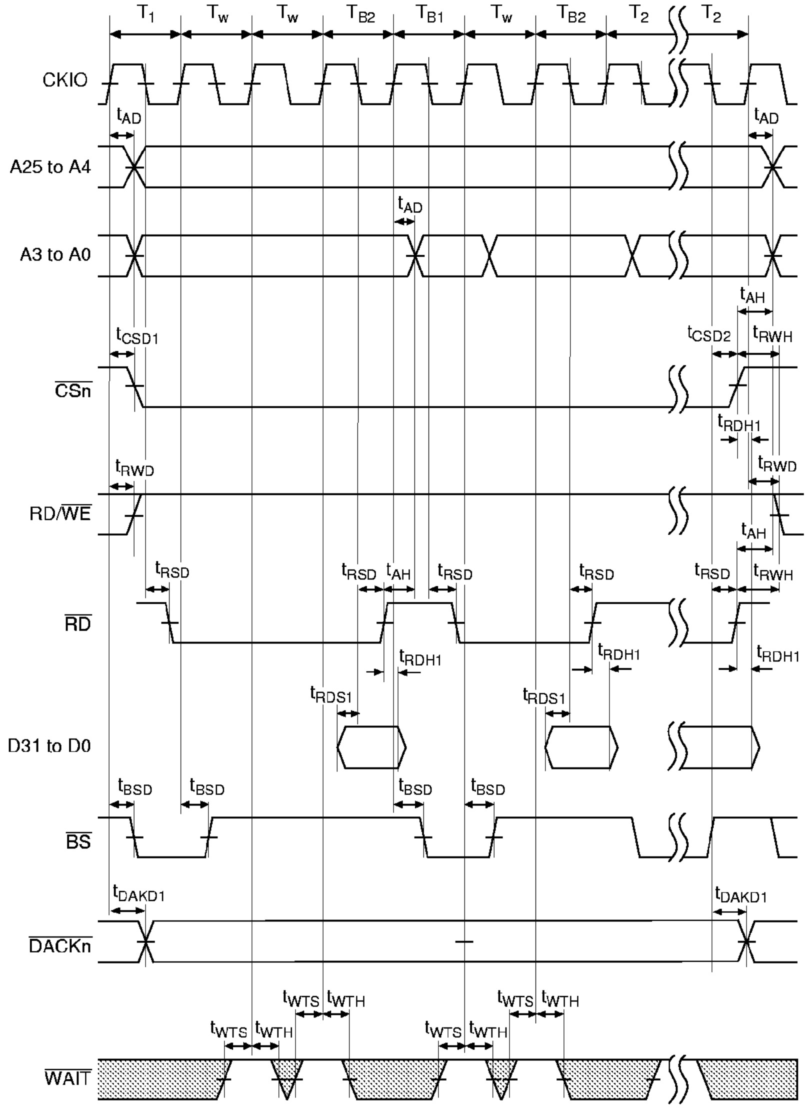
Figure 26.19 (2) Basic Bus Cycle (External Wait, WAITSEL = 1)

26.3.5 Burst ROM Timing



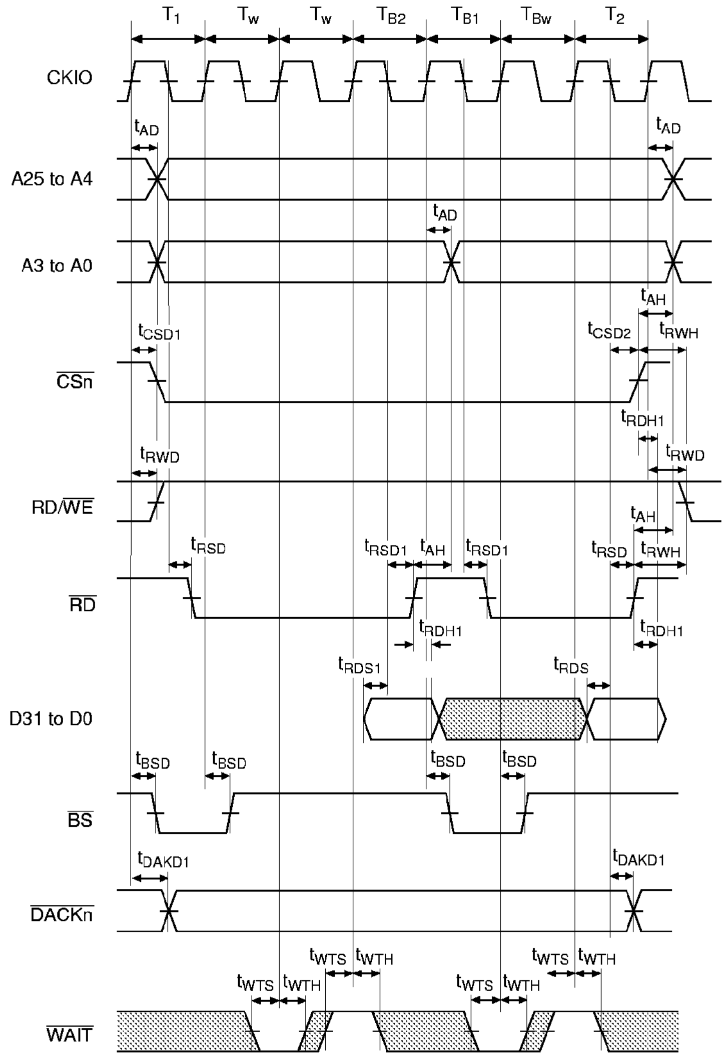
Note: In the write cycle, the basic bus cycle, the basic bus cycle is performed.

Figure 26.20 Burst ROM Bus Cycle (No Wait)



Note: In the write cycle, the basic bus cycle is performed.

Figure 26.21 Burst ROM Bus Cycle (Two Waits)



Note: In the write cycle, the basec bus cycle is performed.

Figure 26.22 Burst ROM Bus Cycle (External Wait, WAITSEL = 0)

26.3.6 DRAM Timing

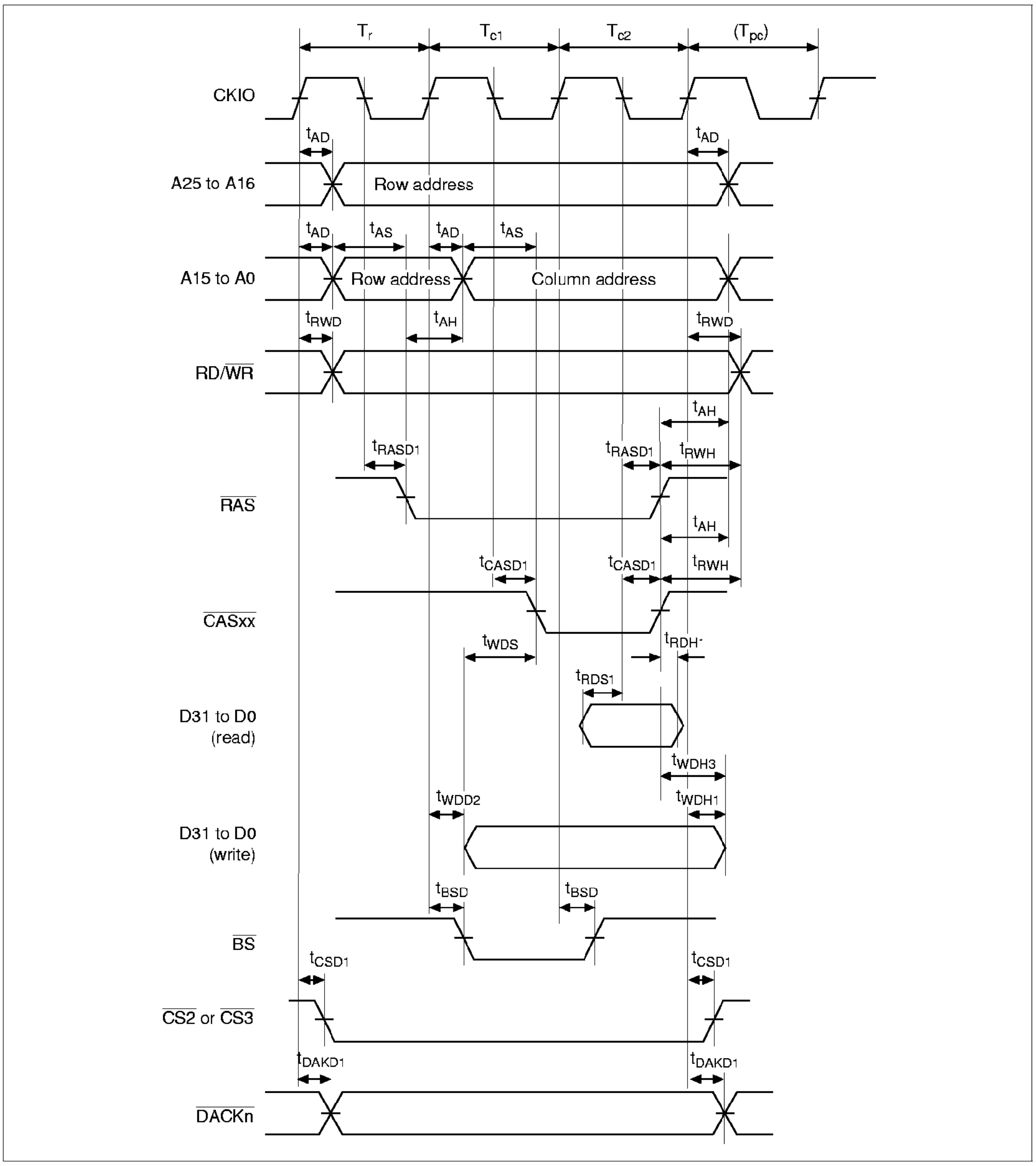


Figure 26.23 DRAM Bus Cycle (TRCD = 0, AnW = 1, TPC = 0)

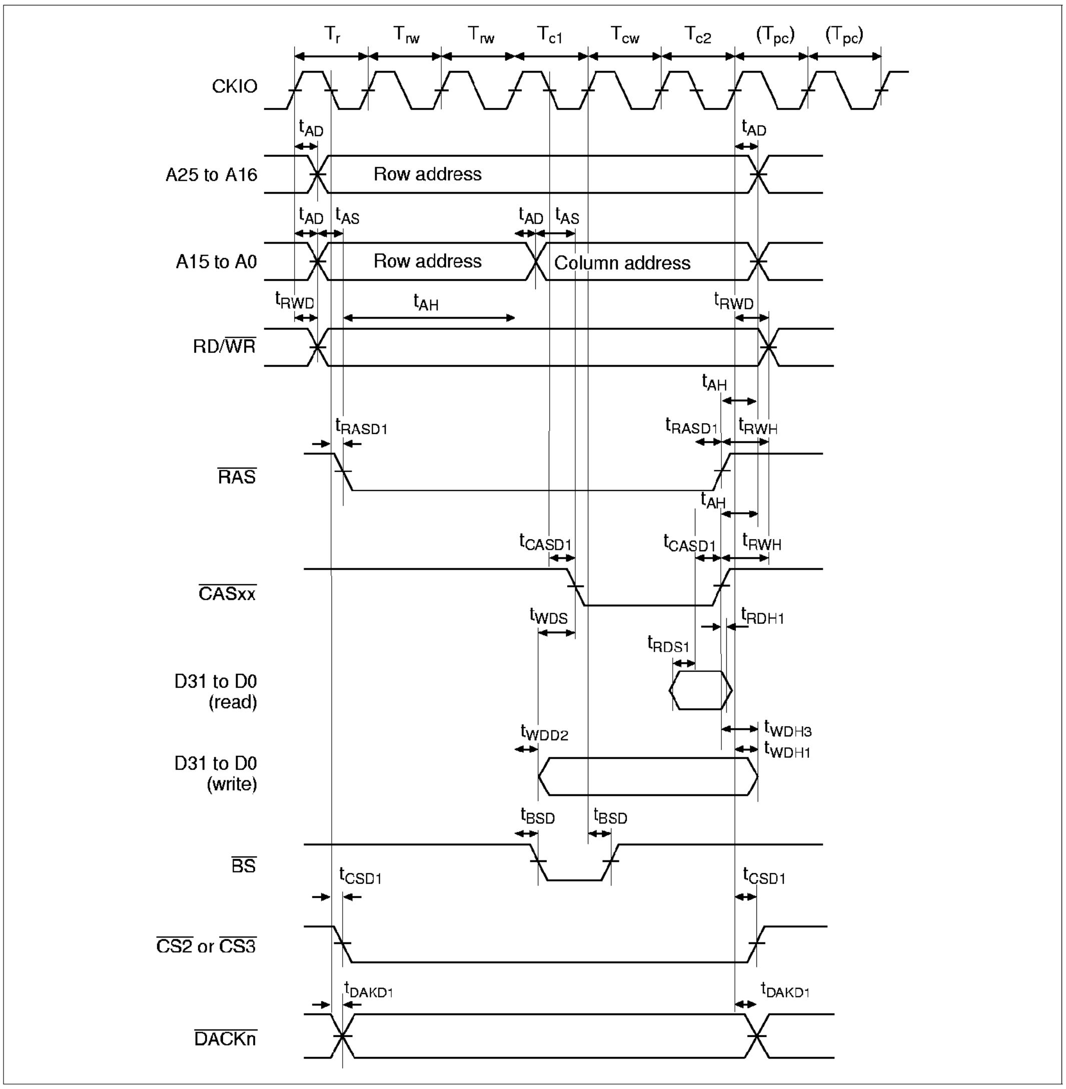


Figure 26.24 DRAM Bus Cycle (TRCD = 2, AnW = 2, TPC = 1)

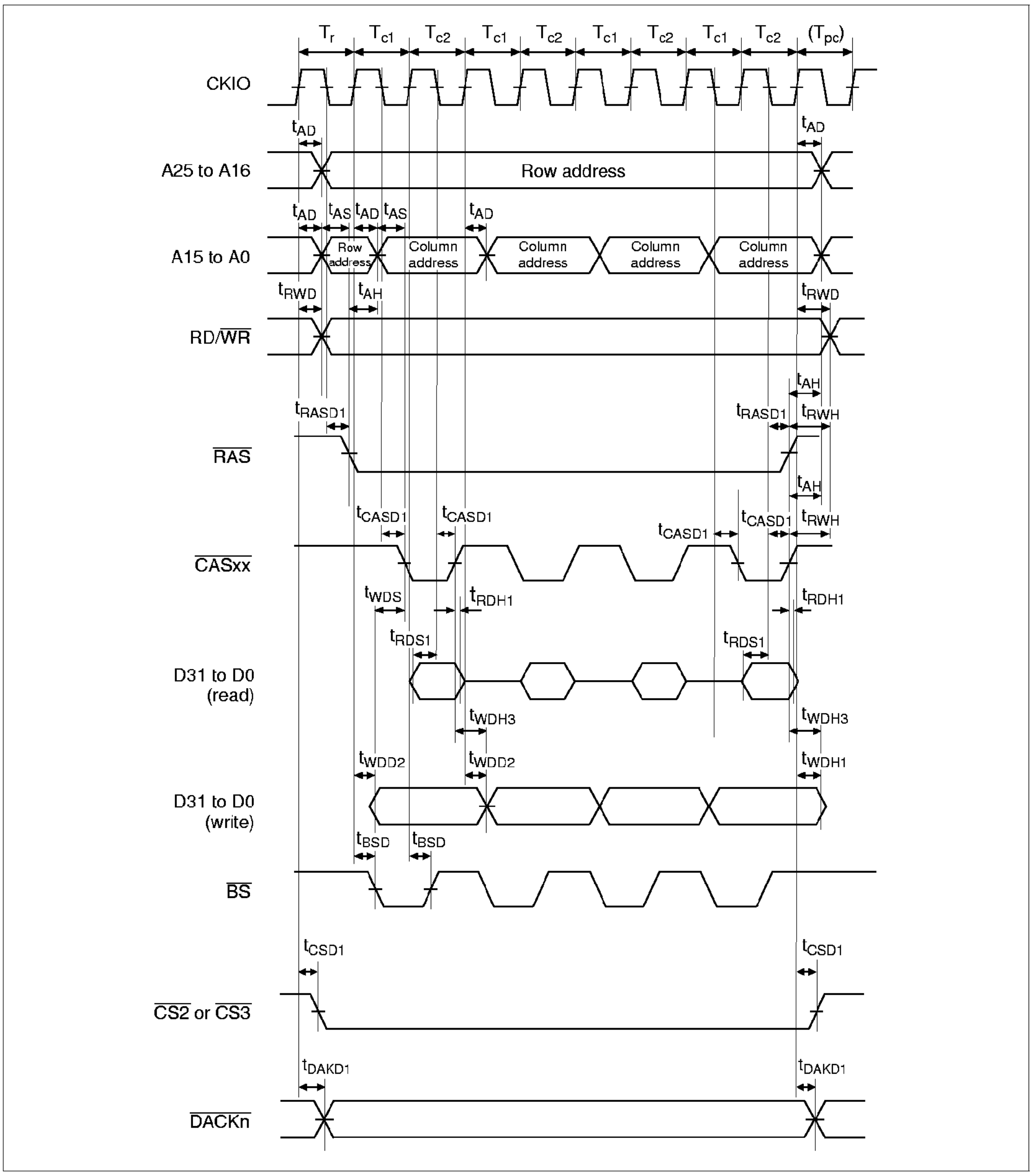


Figure 26.25 DRAM Burst Bus Cycle (TRCD = 0, AnW = 1, TPC = 0)

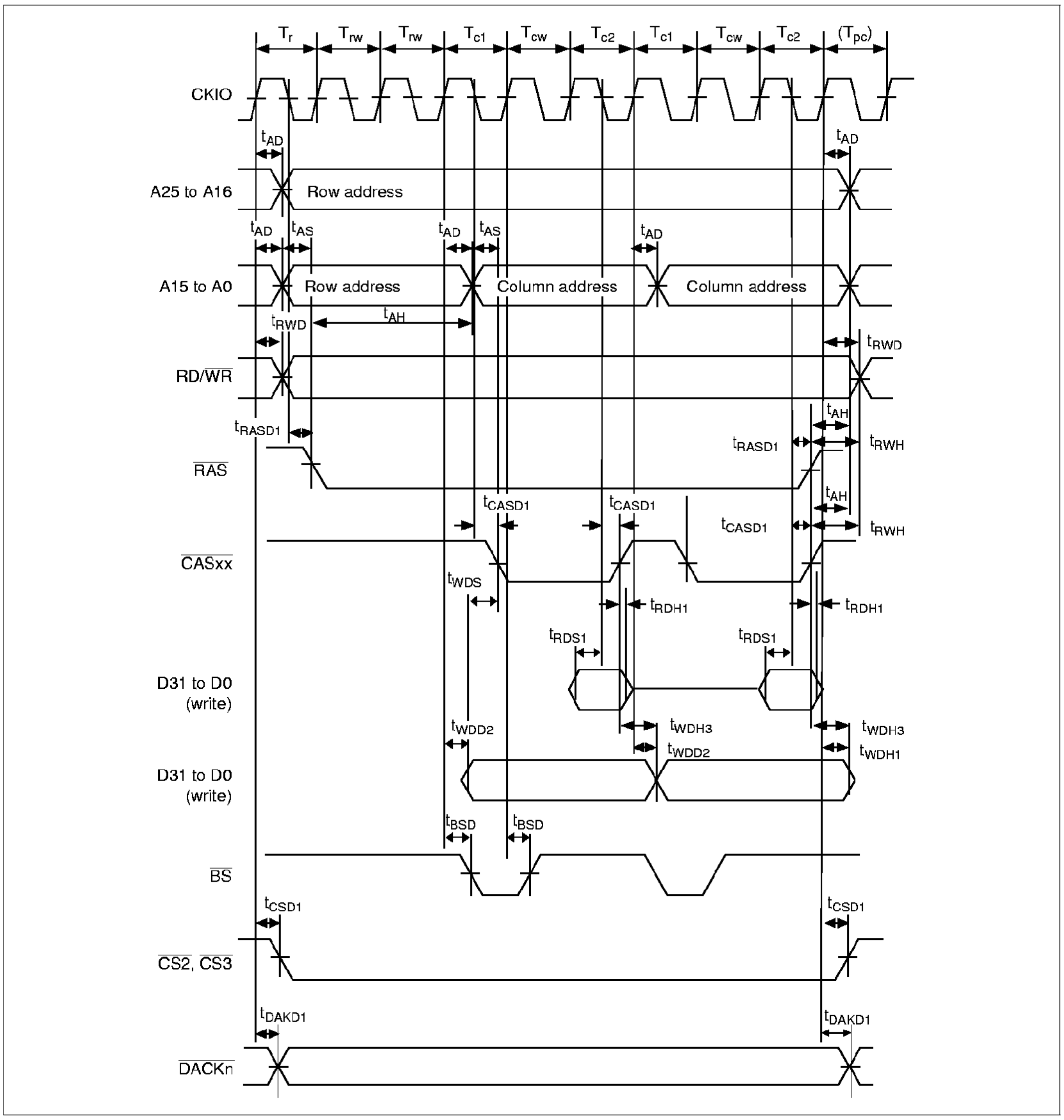


Figure 26.26 DRAM Burst Bus Cycle (TRCD = 2, AnW = 2, TPC = 0)

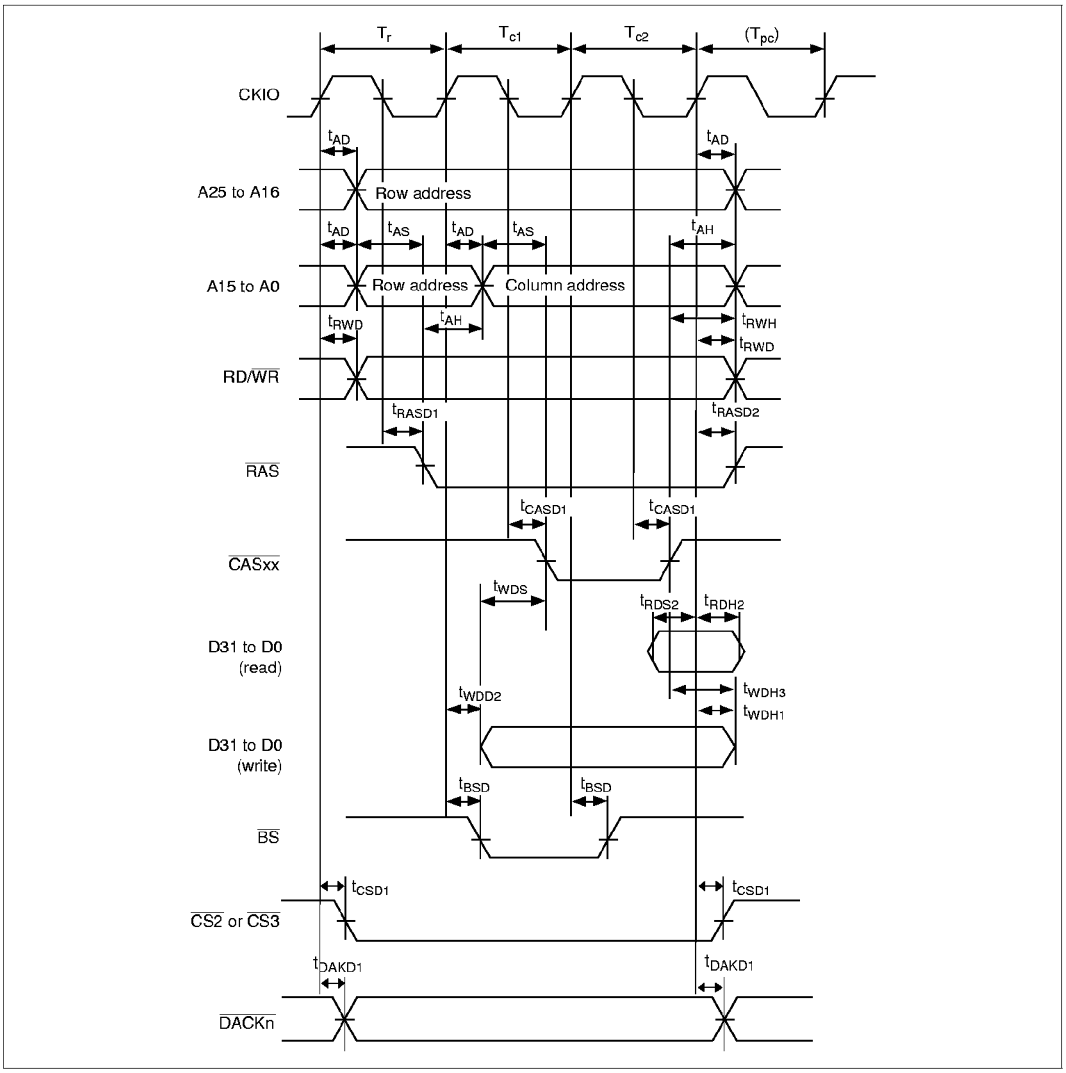


Figure 26.27 DRAM Bus Cycle (EDO mode, TRCD = 0, AnW = 1, TPC = 0)

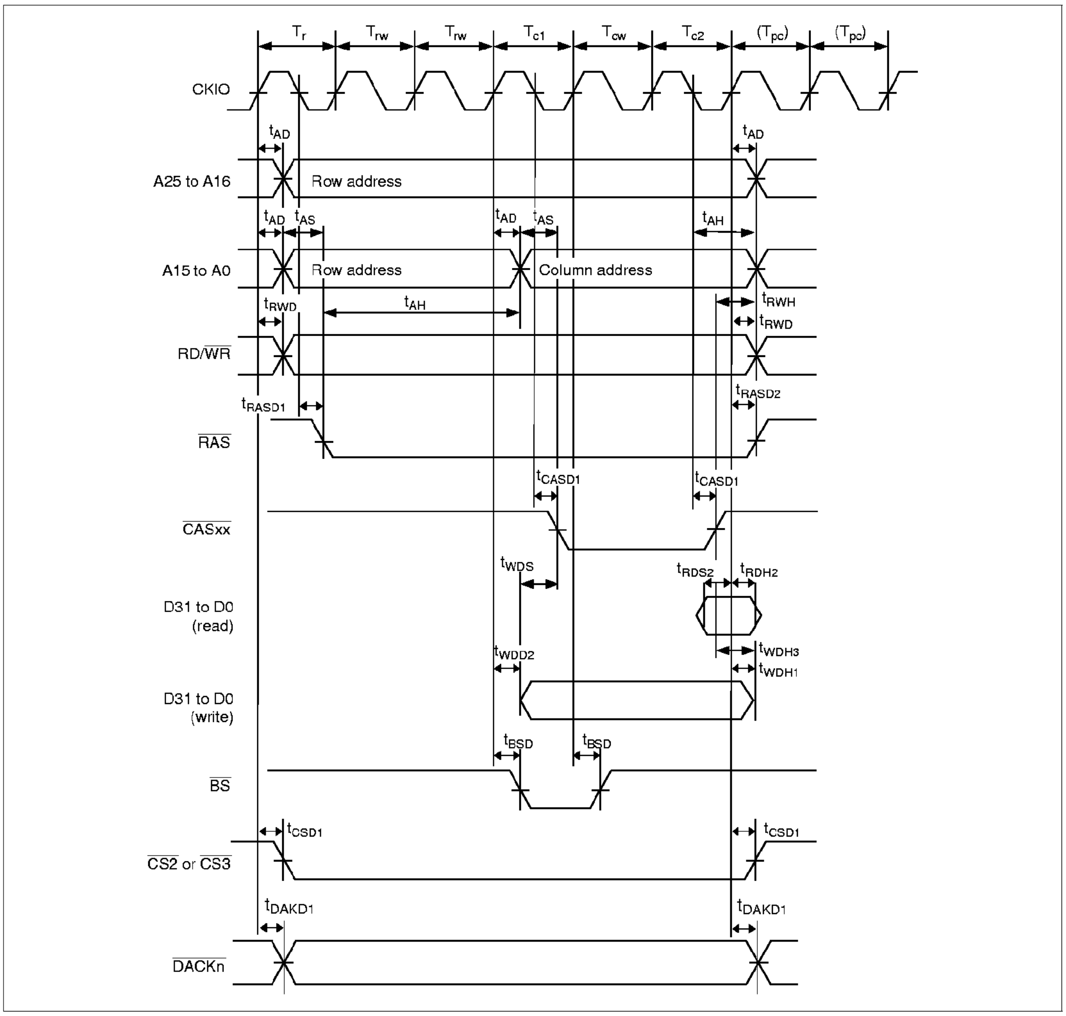


Figure 26.28 DRAM Bus Cycle (EDO mode, TRCD = 2, AnW = 2, TPC = 1)

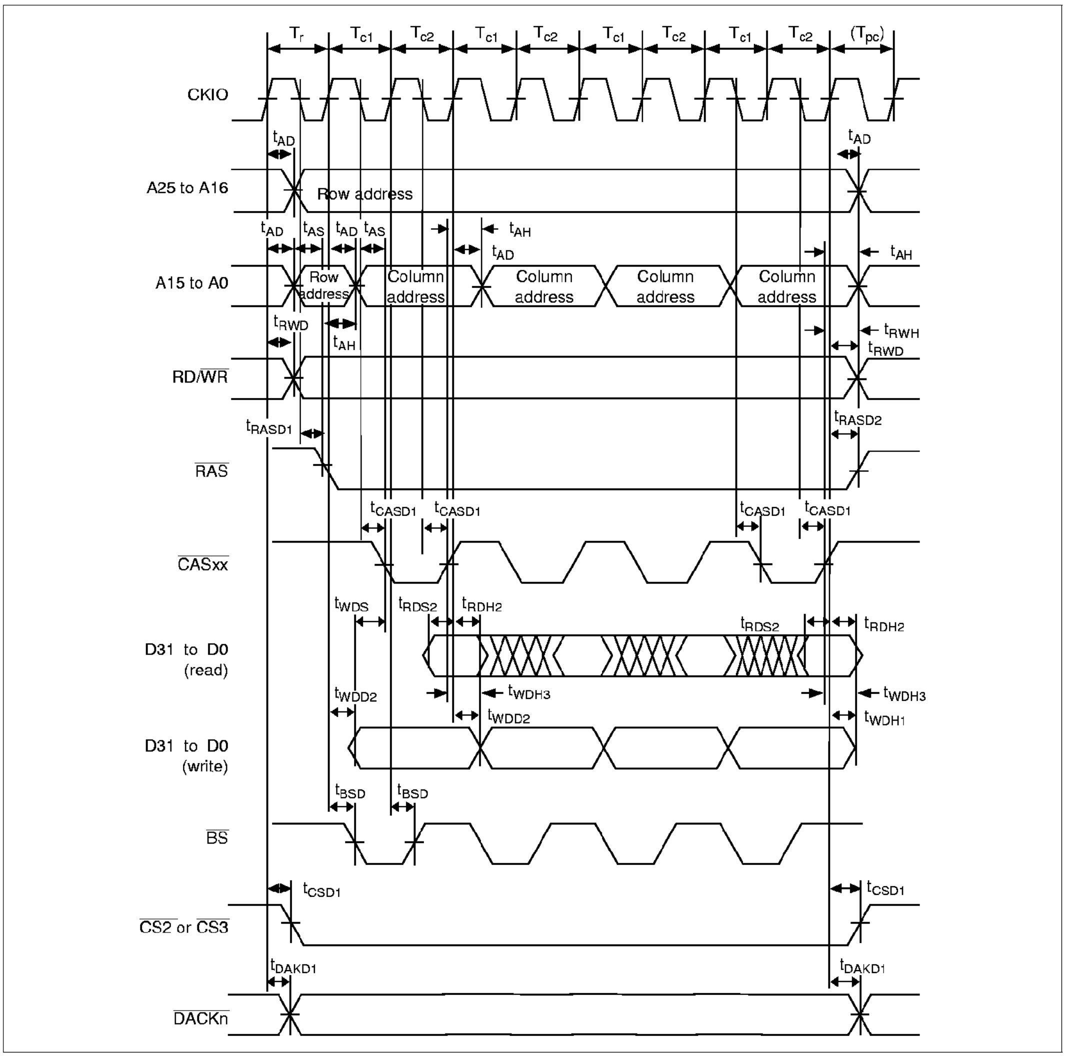


Figure 26.29 DRAM Burst Bus Cycle (EDO mode, TRCD = 0, AnW = 1, TPC = 0)

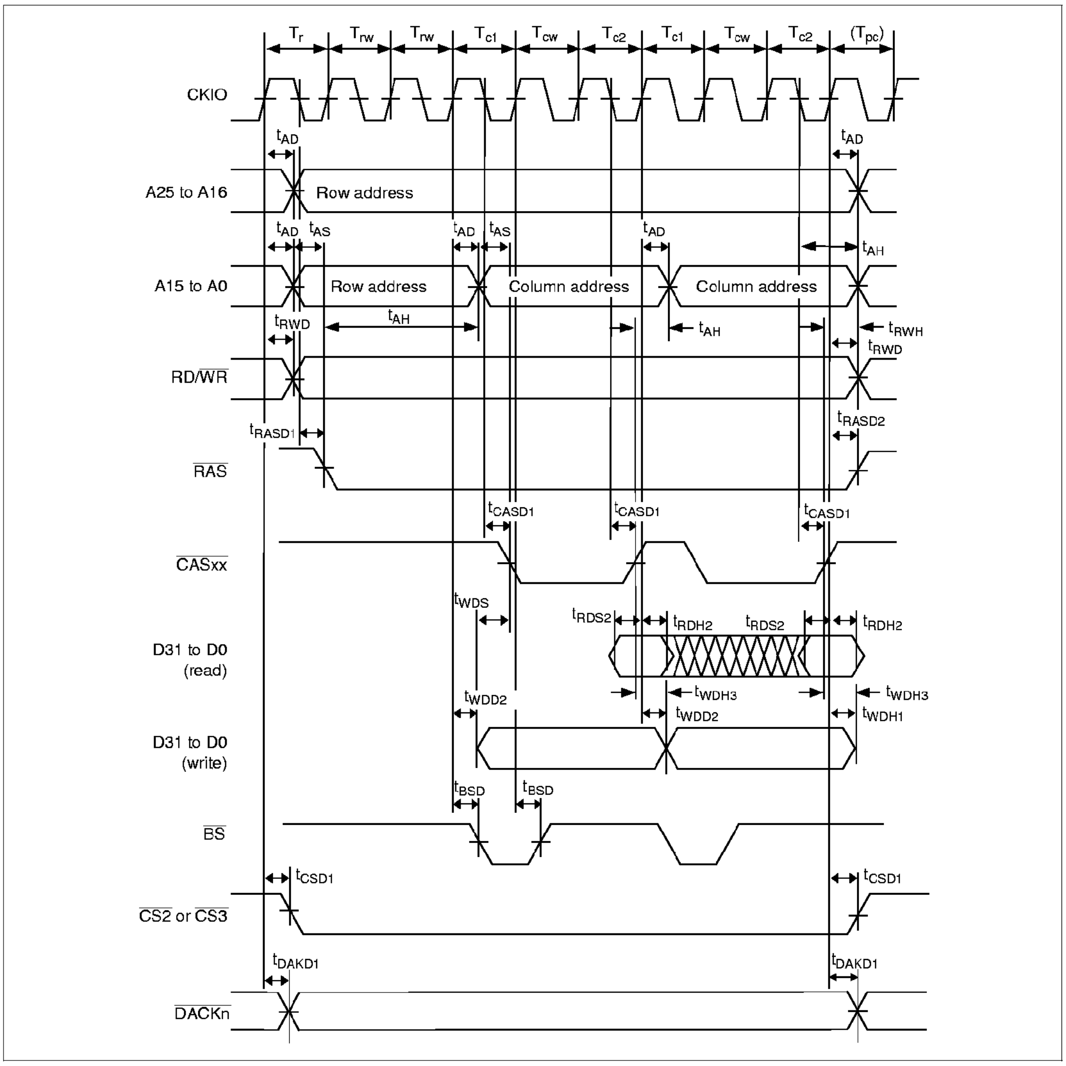


Figure 26.30 DRAM Burst Bus Cycle
(EDO mode, TRCD = 2, AnW = 2, TPC = 0)

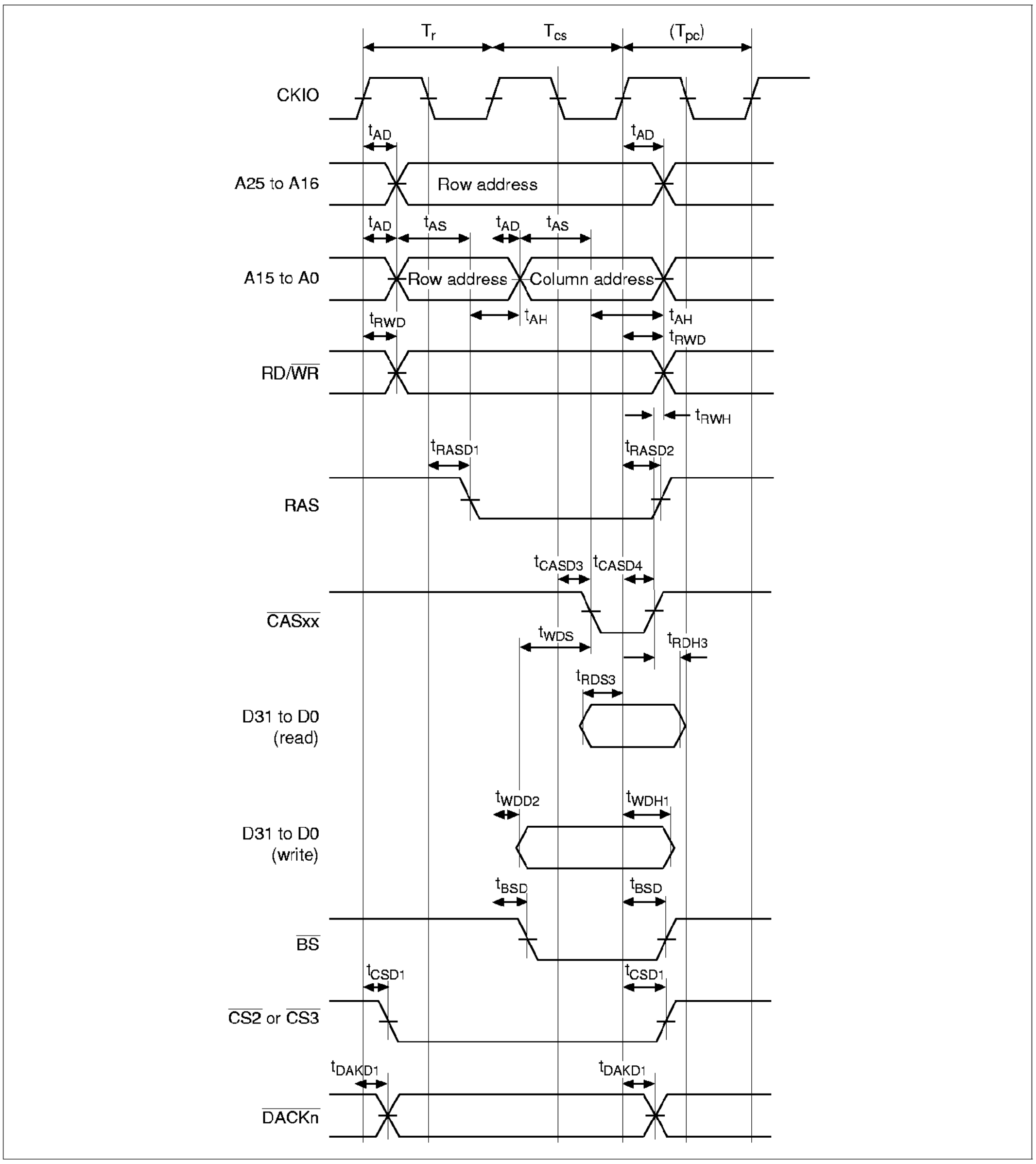


Figure 26.31 DRAM Short-Pitch Access Timing

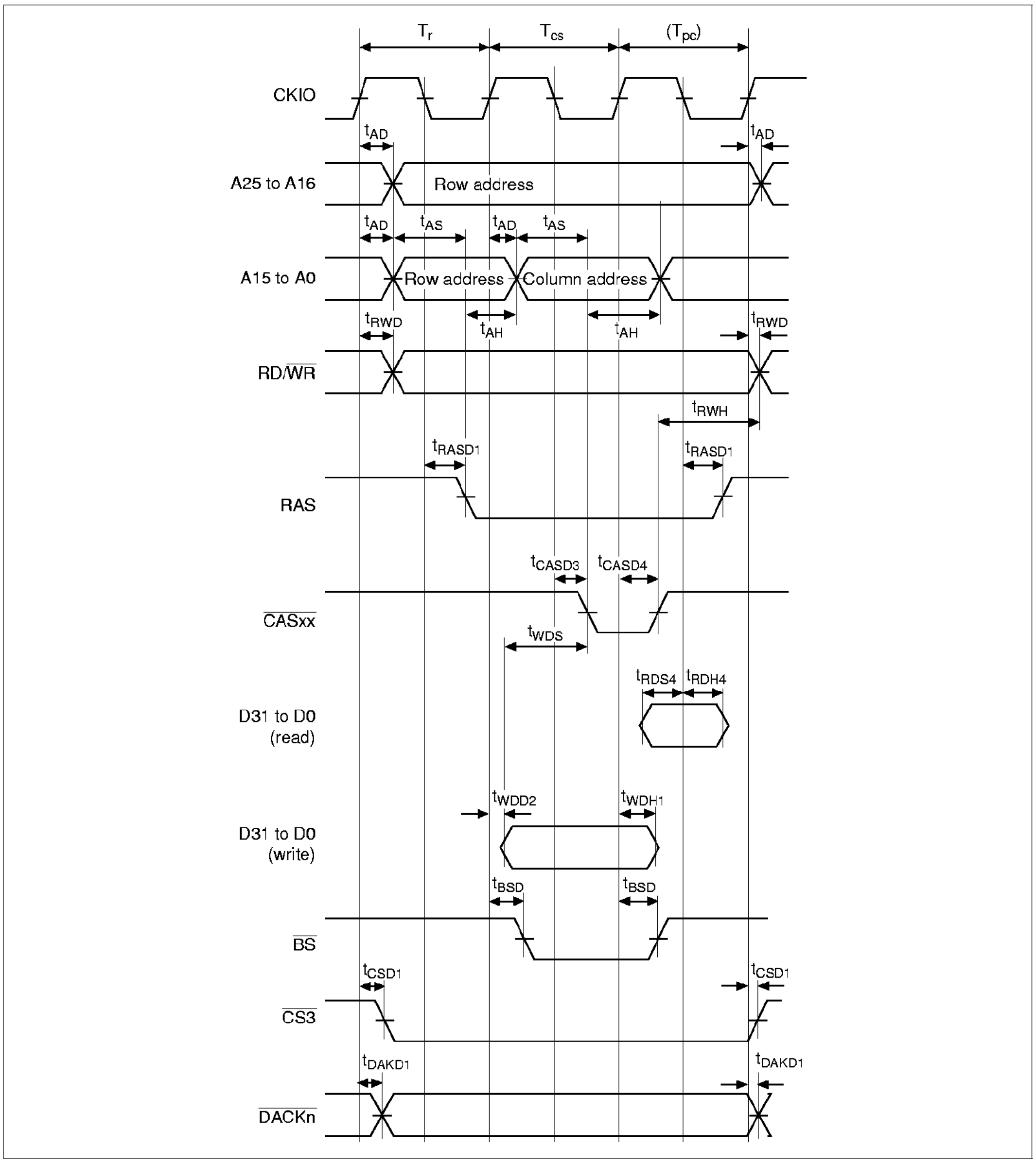


Figure 26.32 DRAM Short-Pitch Access Timing (EDO mode)

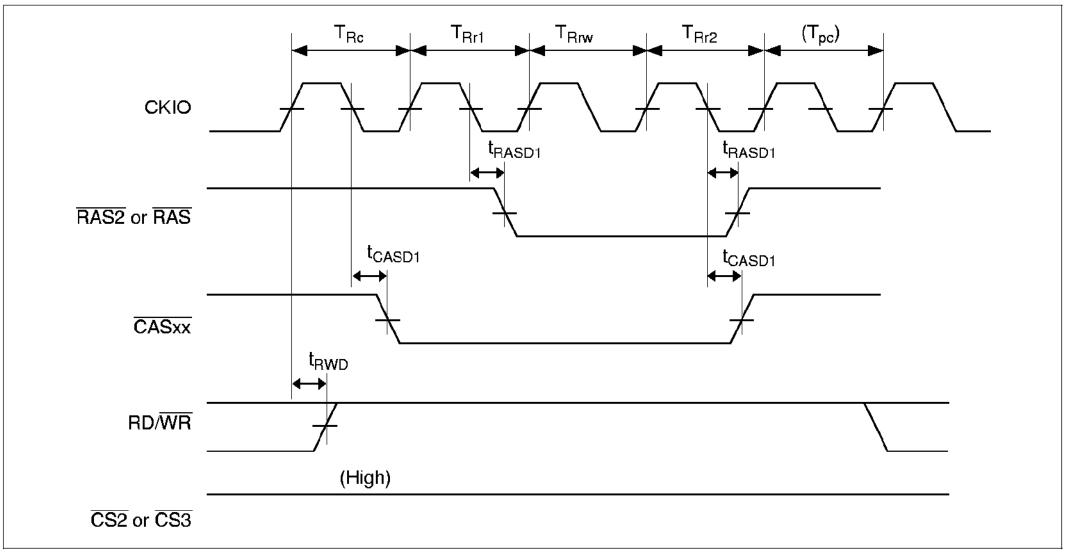


Figure 26.33 DRAM CAS-before-RAS Refresh Cycle (TRAS = 0, TPC = 0)

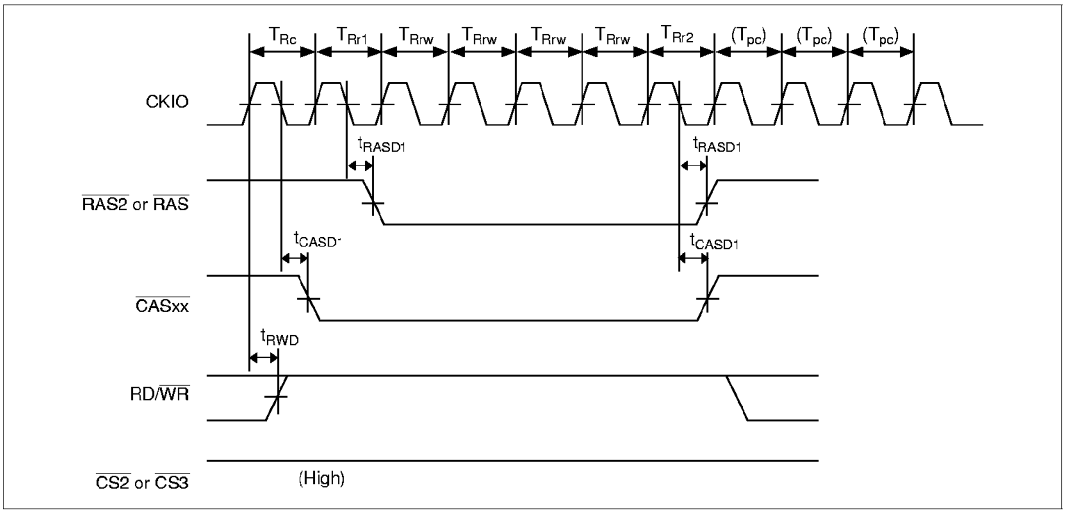


Figure 26.34 DRAM CAS-before-RAS Refresh Cycle (TRAS = 3, TPC = 2)

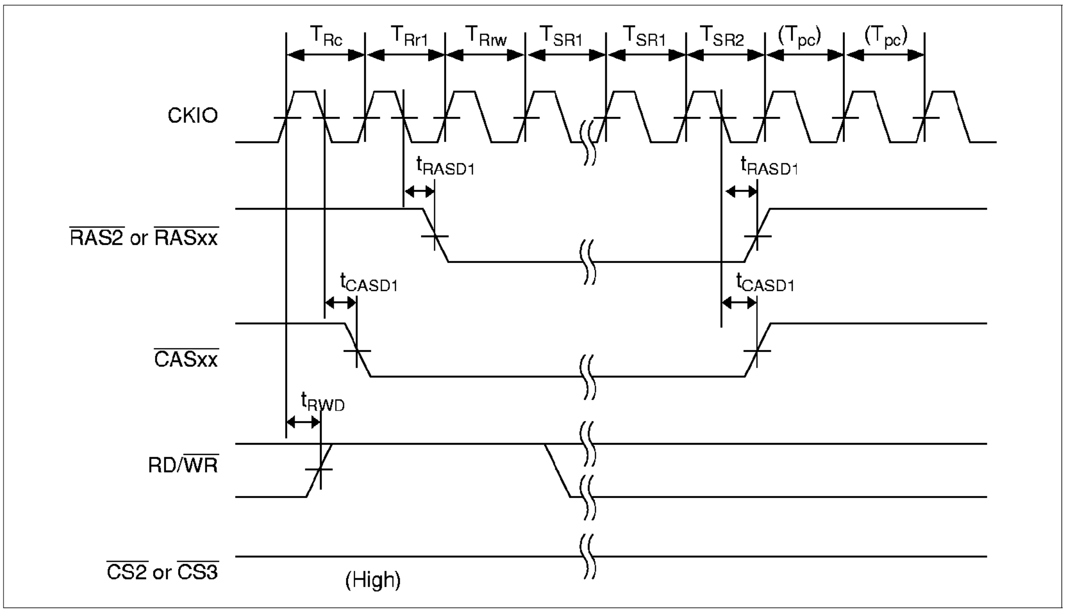


Figure 26.35 DRAM Self-refresh Cycle (TPC = 0)

26.3.7 Synchronous DRAM Timing

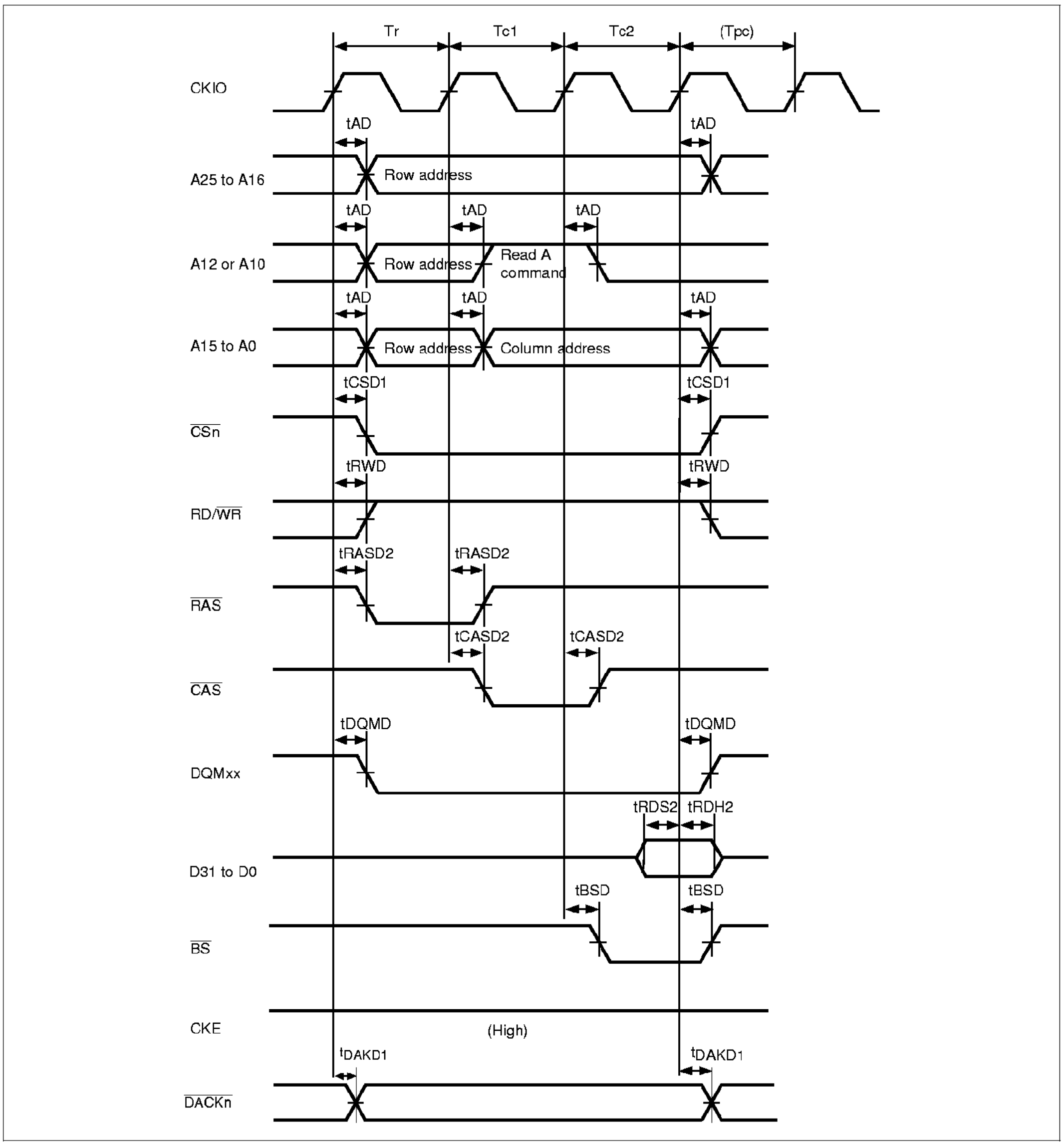


Figure 26.36 Synchronous DRAM Read Bus Cycle (RCD = 0, CAS Latency = 1, TPC = 0)

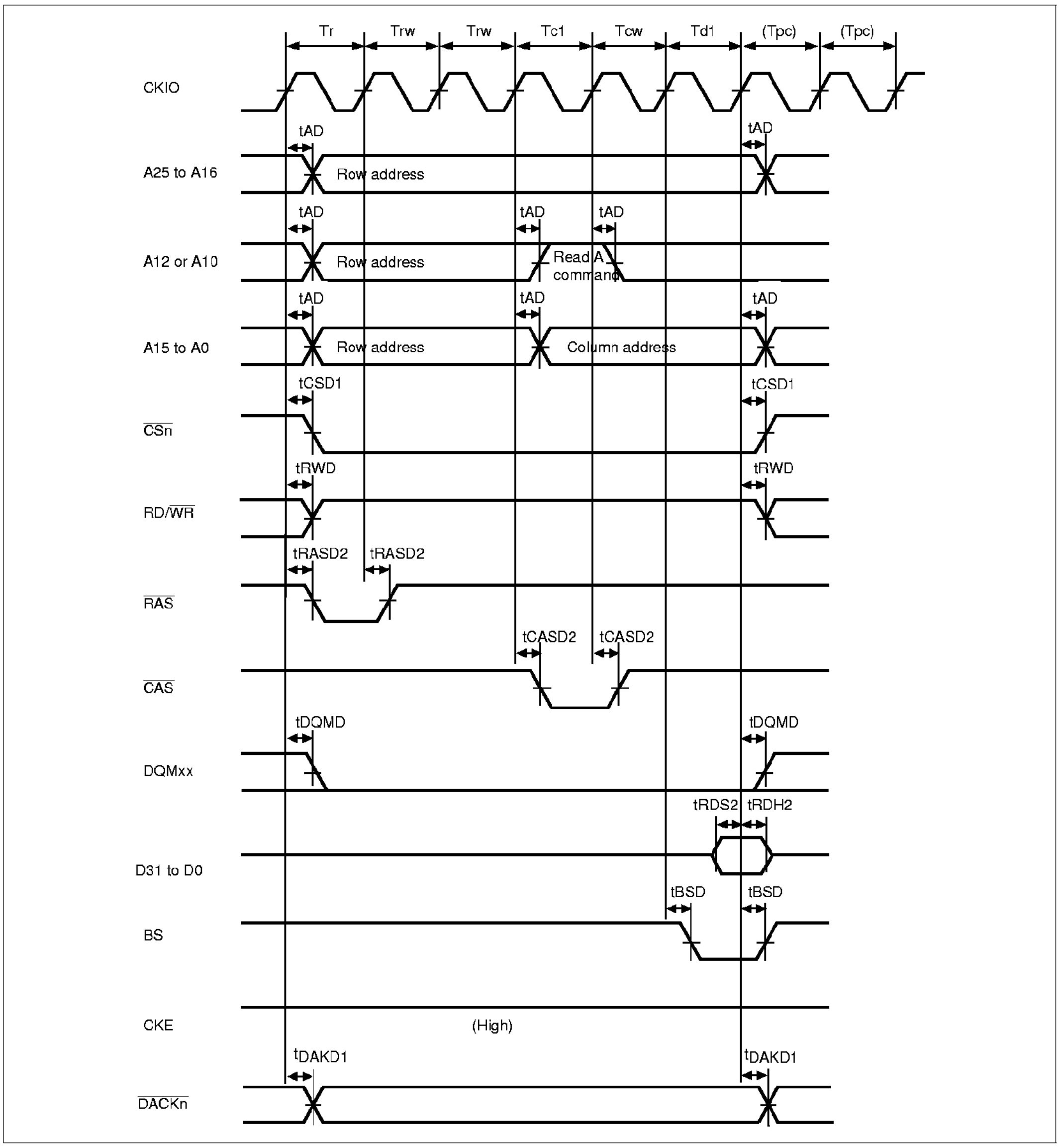


Figure 26.37 Synchronous DRAM Read Bus Cycle (RCD = 2, CAS Latency = 2, TPC = 1)

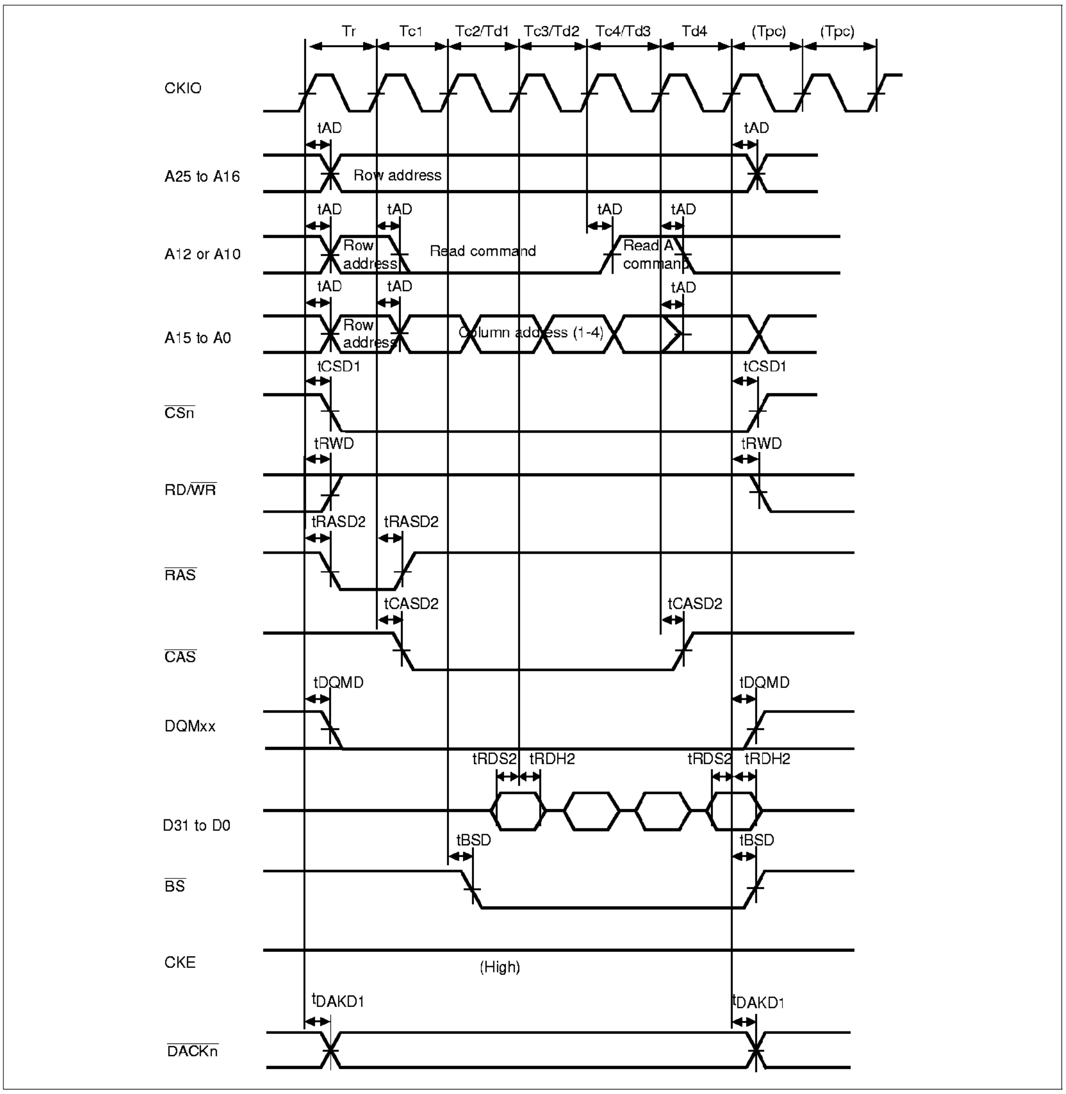


Figure 26.38 Synchronous DRAM Read Bus Cycle (Burst Read (Single Read × 4), RCD = 0, CAS Latency = 1, TPC = 1)

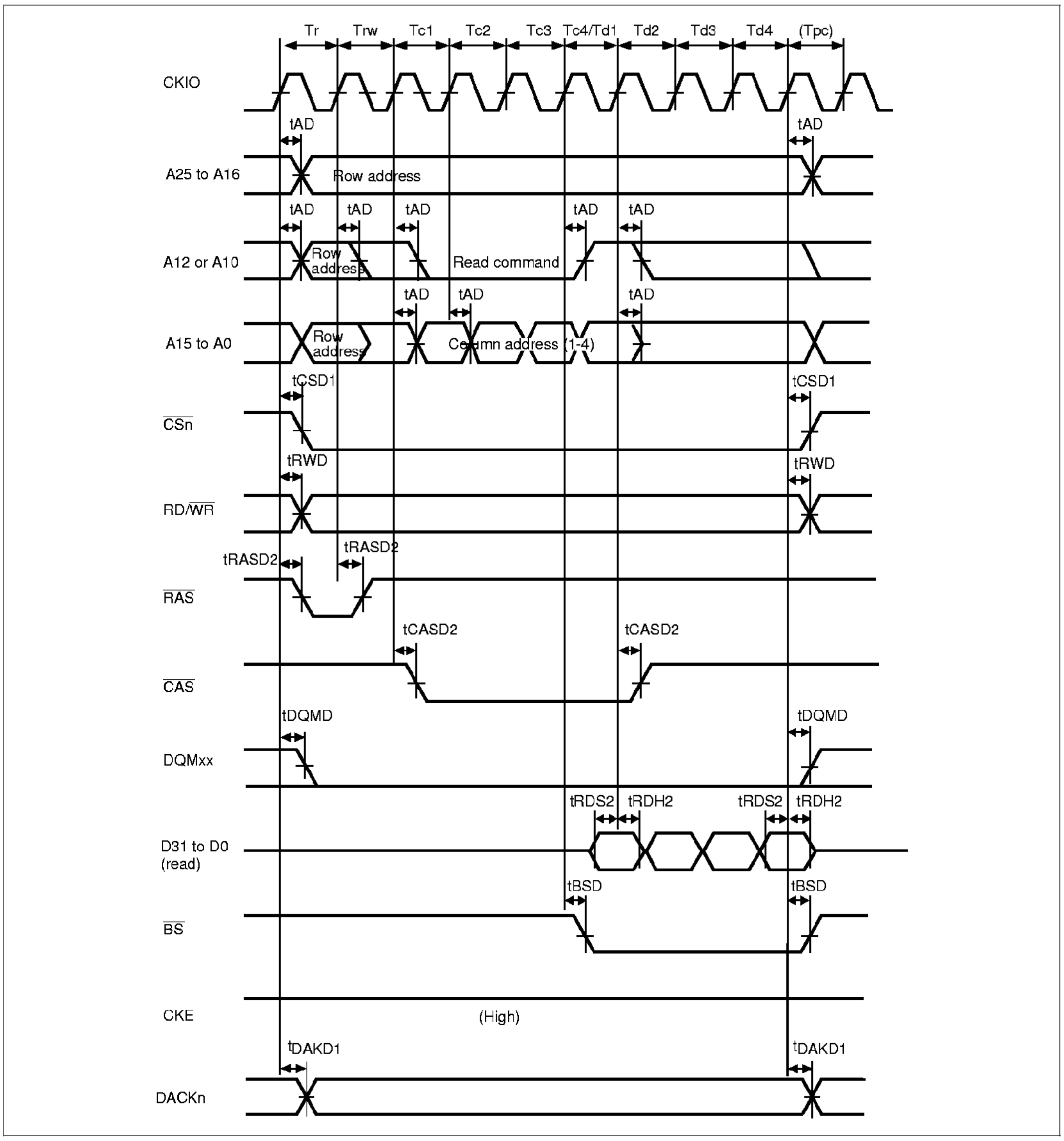


Figure 26.39 Synchronous DRAM Read Bus Cycle (Burst Read (Single Read × 4), RCD = 1, CAS Latency = 3, TPC = 0)

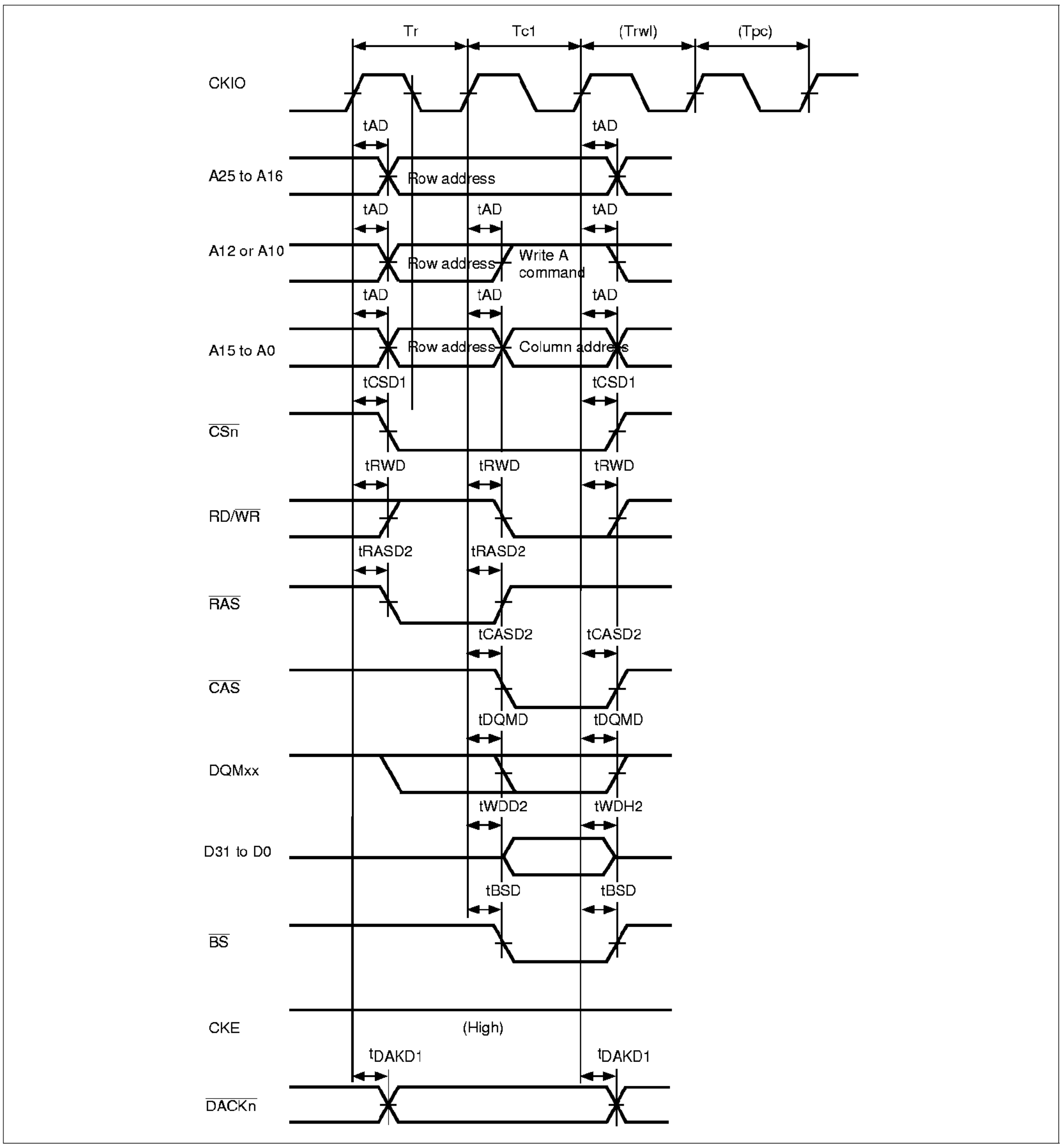


Figure 26.40 Synchronous DRAM Write Bus Cycle (RCD = 0, TPC = 0, TRWL = 0)

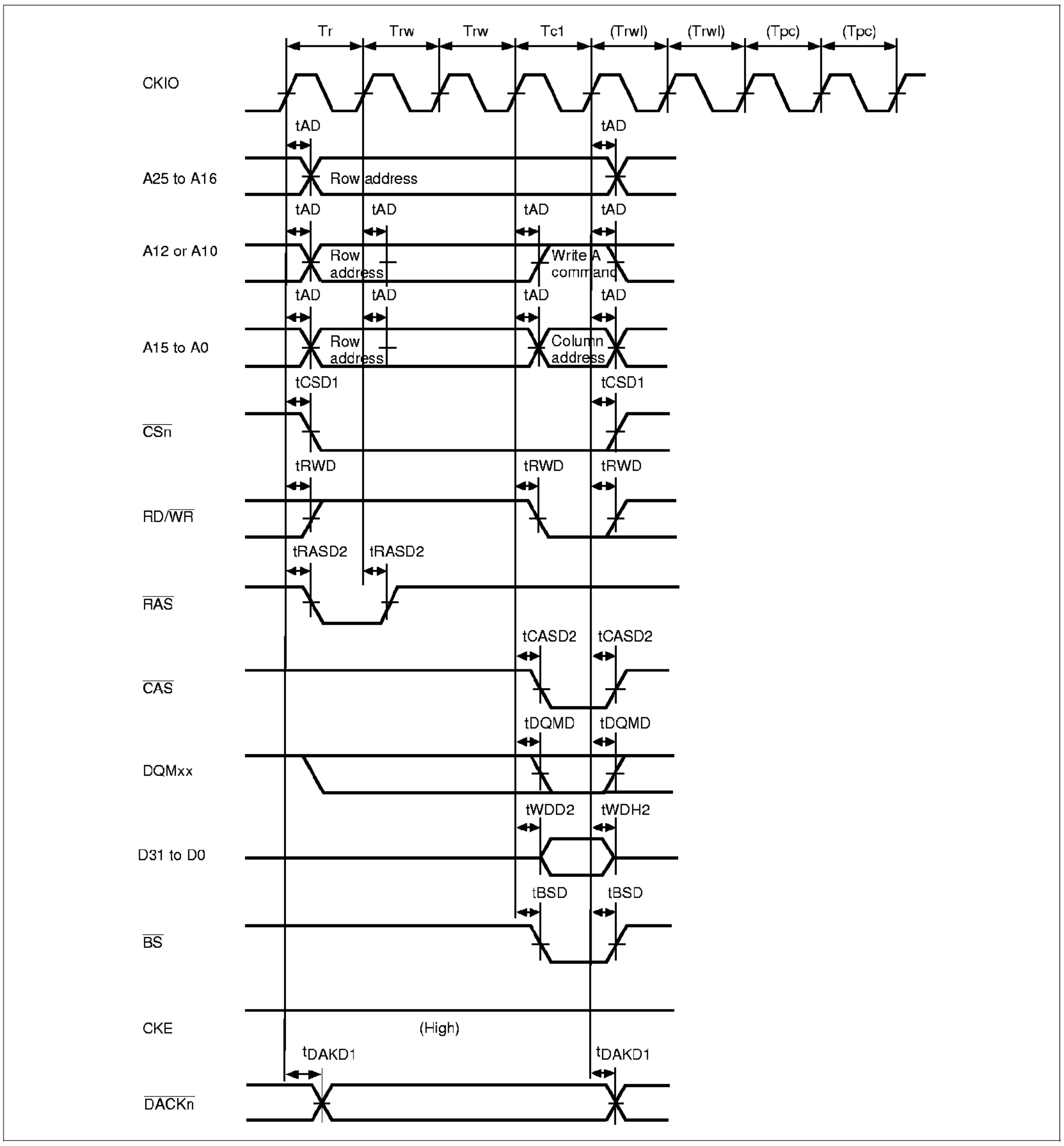


Figure 26.41 Synchronous DRAM Write Bus Cycle (RCD = 2, TPC = 1, TRWL = 1)

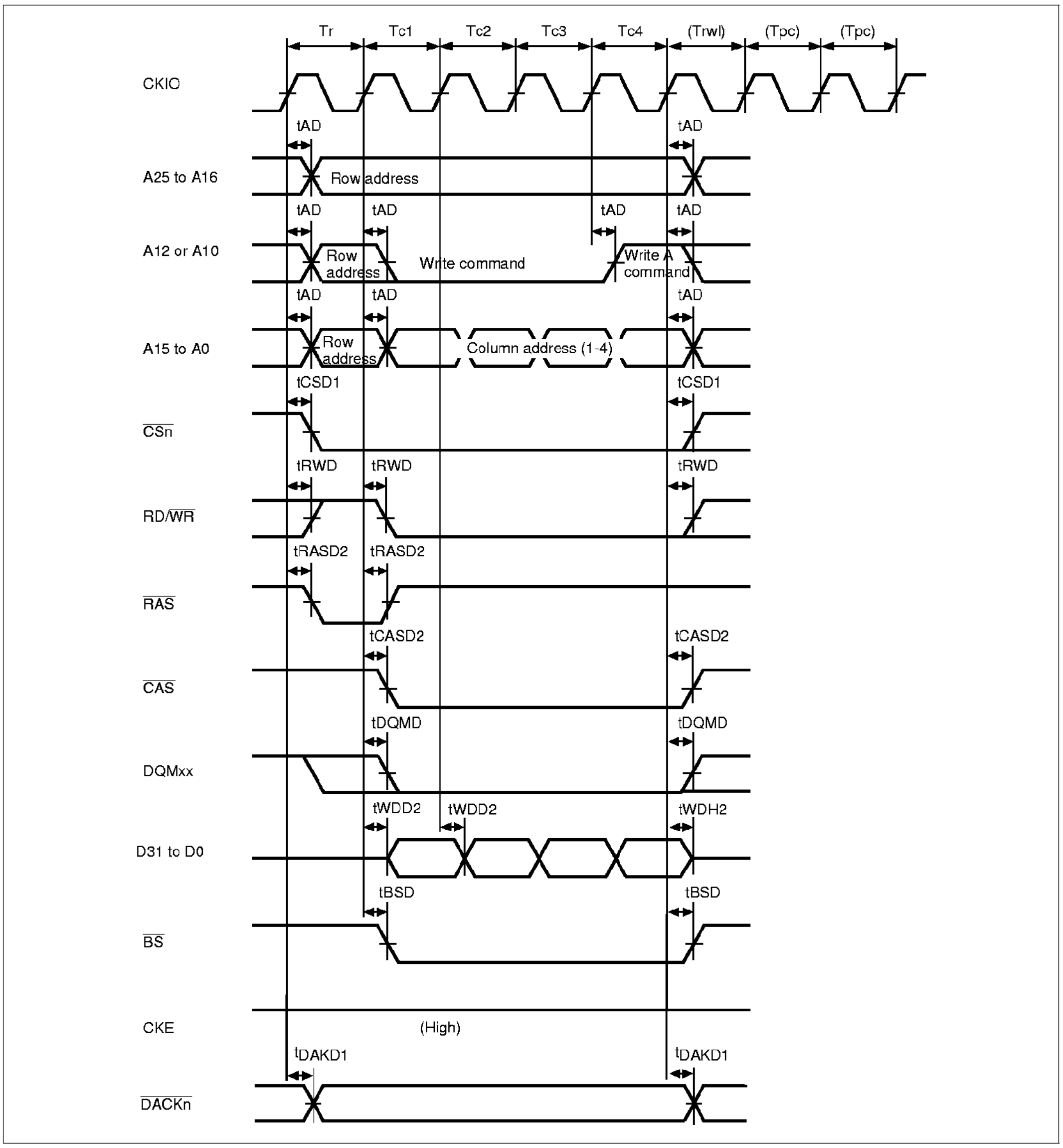


Figure 26.42 Synchronous DRAM Write Bus Cycle (Burst Mode (Single Write × 4), RCD = 0, TPC = 1, TRWL = 0)

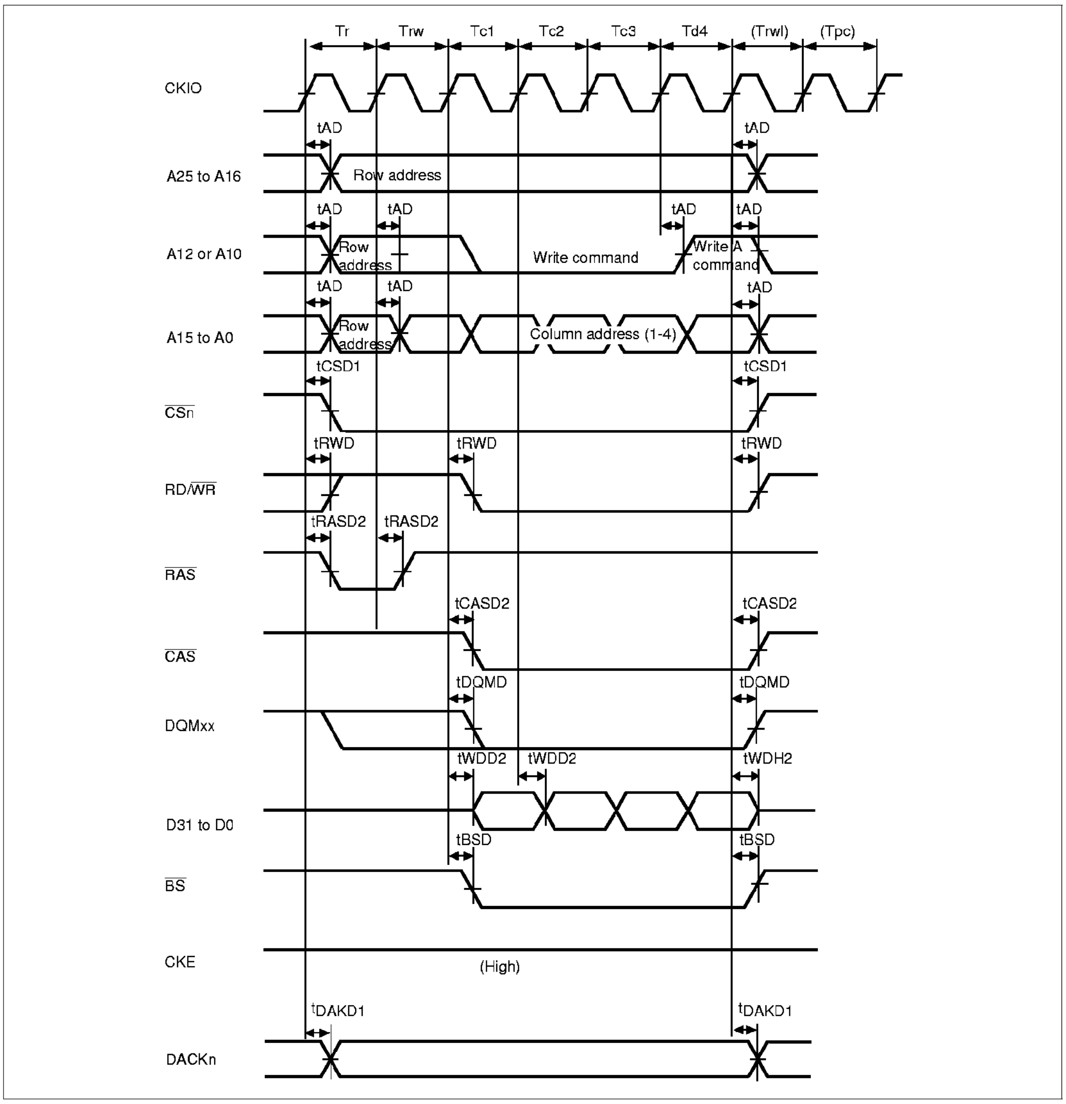


Figure 26.43 Synchronous DRAM Write Bus Cycle (Burst Mode (Single Write × 4), RCD = 1, TPC = 0, TRWL = 0)

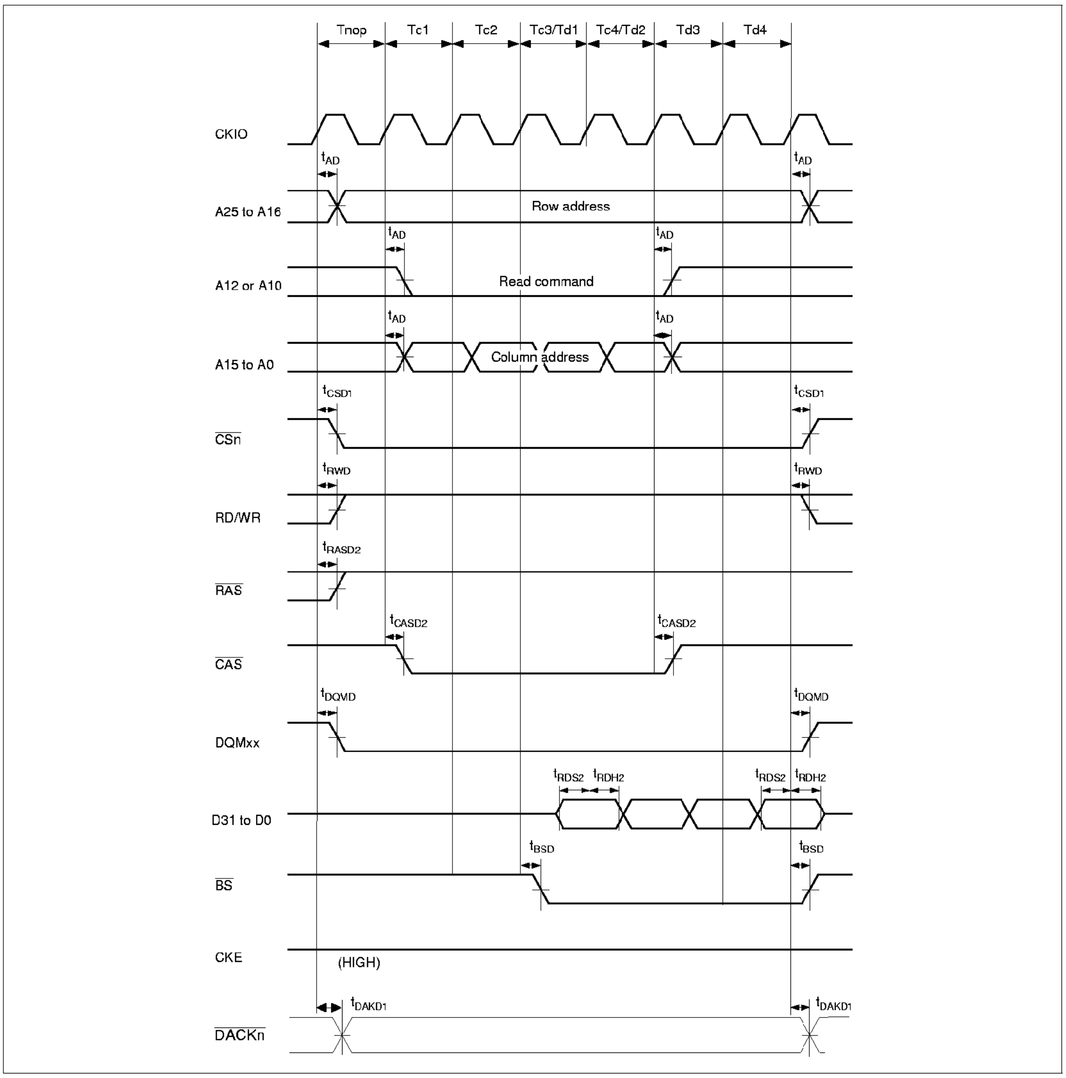


Figure 26.44 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Same Row Address, CAS Latency = 1)

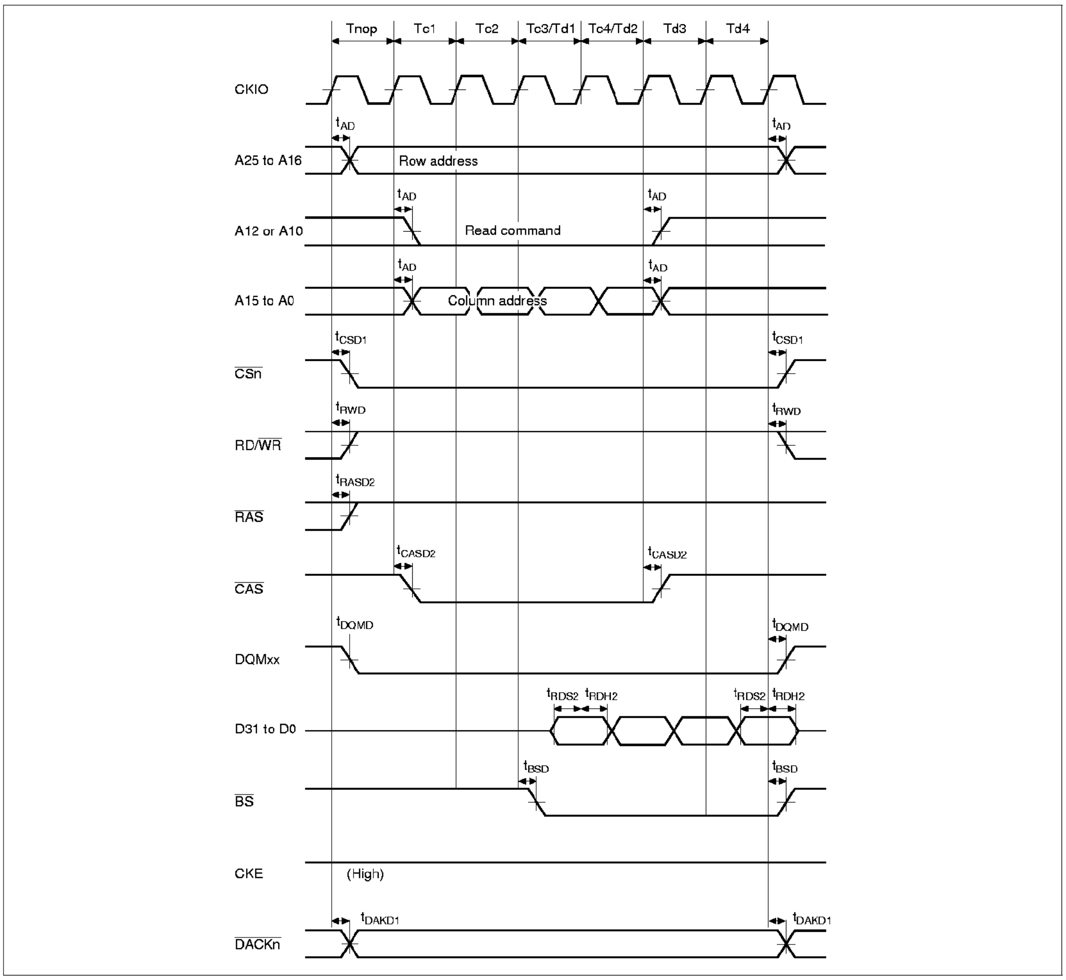


Figure 26.45 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Same Row Address, CAS Latency = 2)

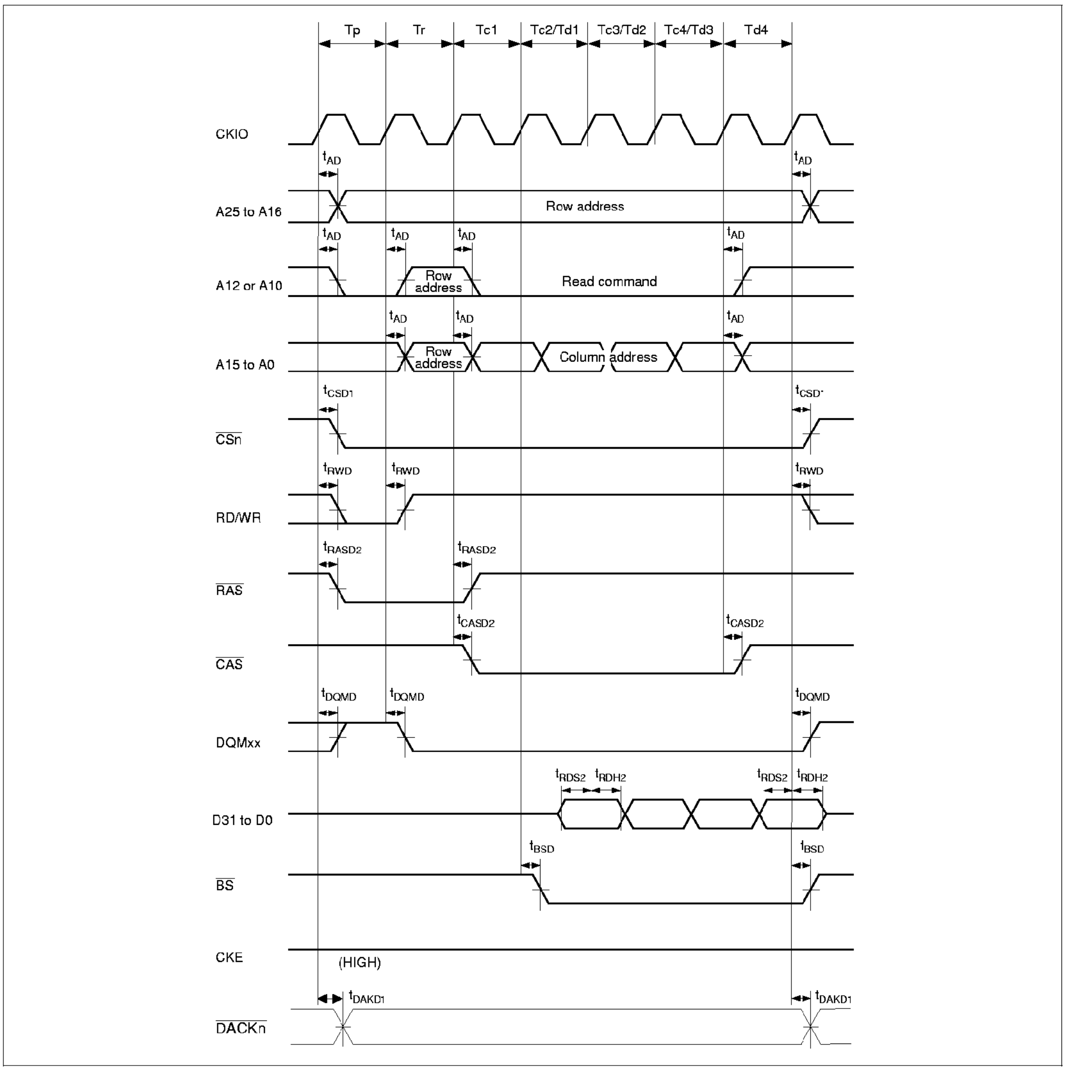


Figure 26.46 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Different Row Address, TPC = 0, RCD = 0, CAS Latency = 1)

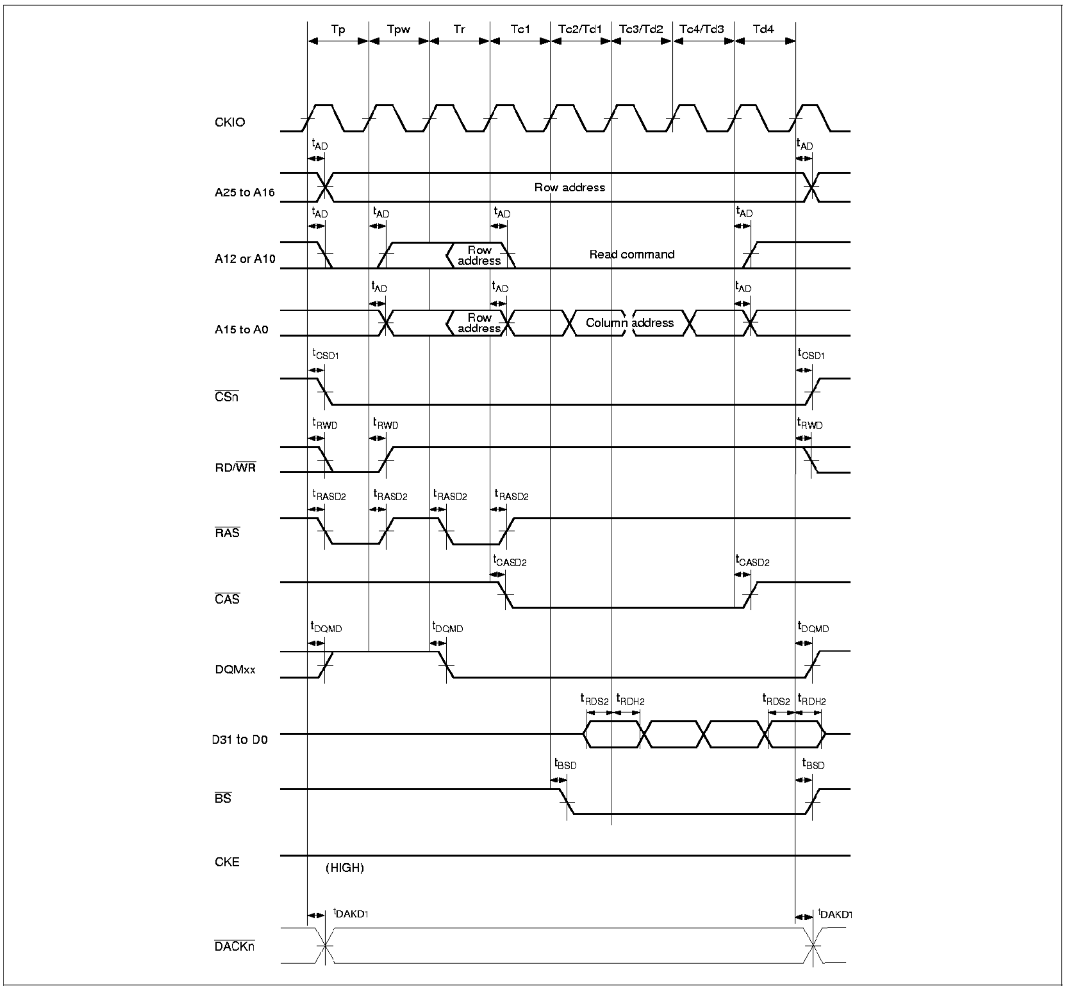


Figure 26.47 Synchronous DRAM Burst Read Bus Cycle
(RAS Down, Different Row Address, TPC = 1, RCD = 0, CAS Latency = 1)

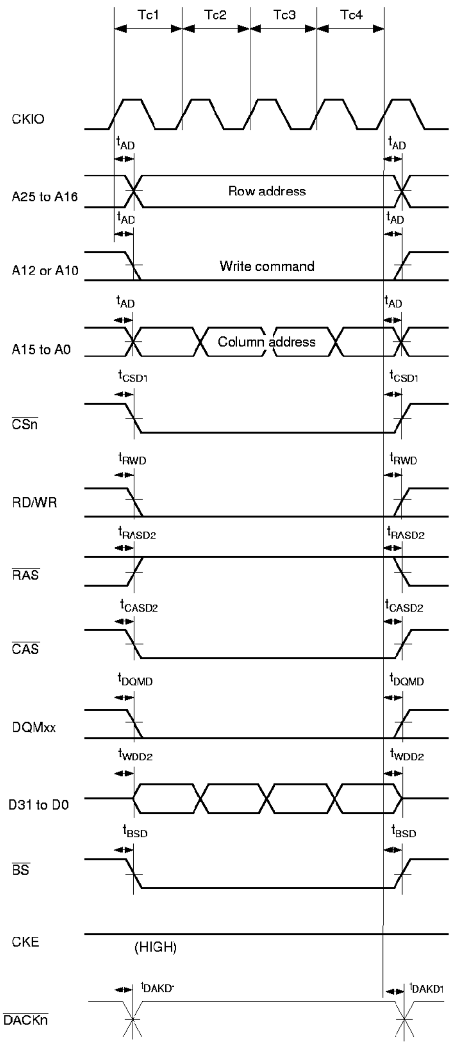


Figure 26.48 Synchronous DRAM Burst Write Bus Cycle (RAS Down, Same Row Address)

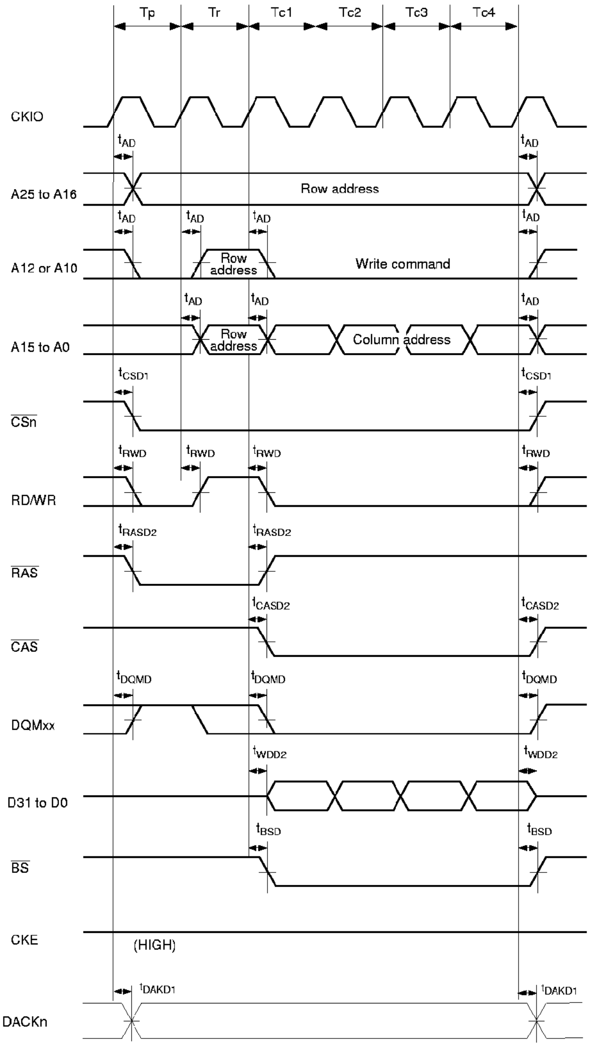


Figure 26.49 Synchronous DRAM Burst Write Bus Cycle (RAS Down, Different Row Address, TPC = 0, RCD = 0)

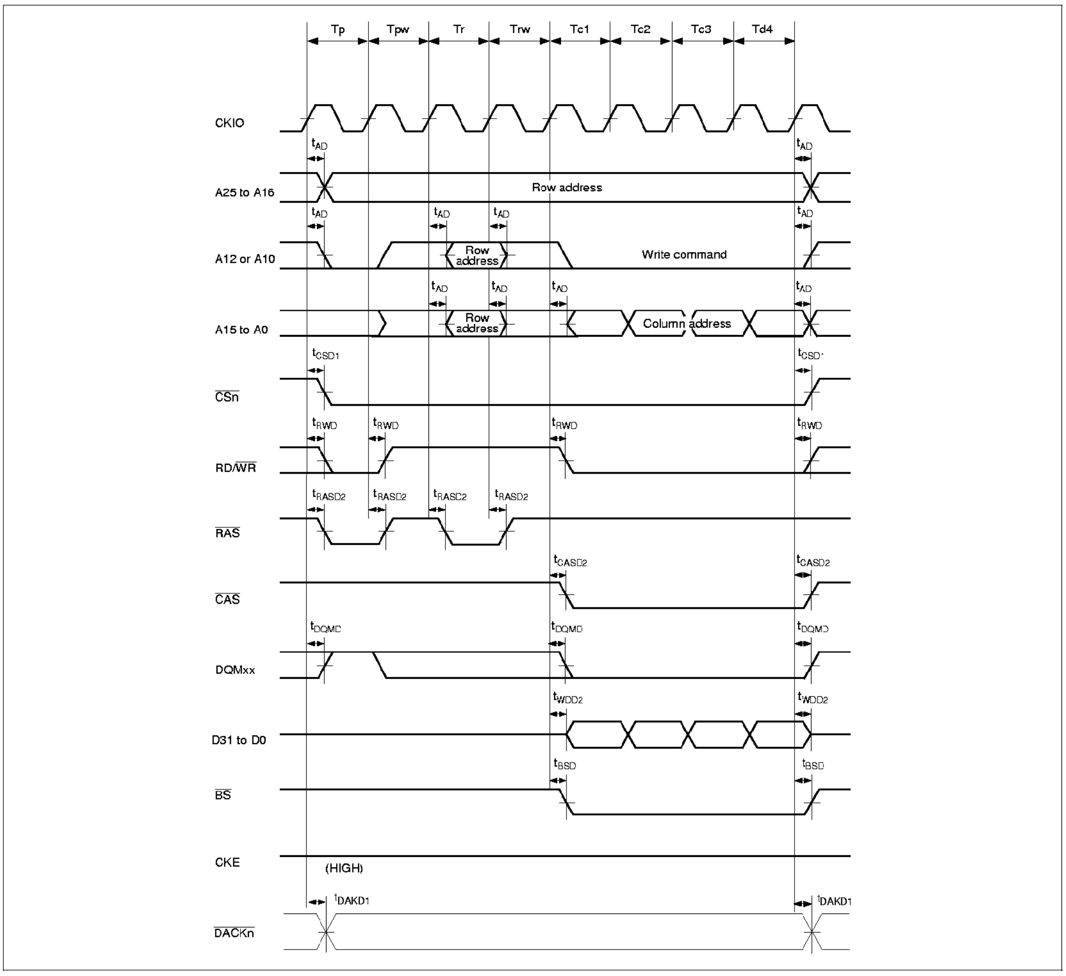


Figure 26.50 Synchronous DRAM Burst Write Bus Cycle (RAS Down, Different Row Address, TPC = 1, RCD = 1)

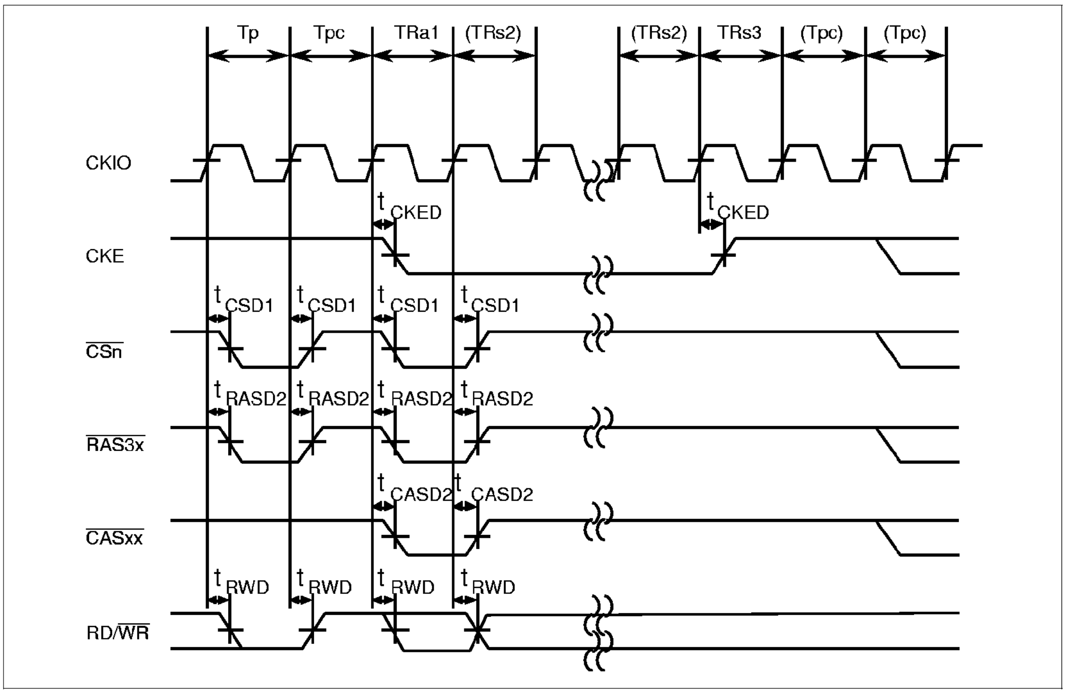


Figure 26.51 Synchronous DRAM Auto-Refresh Timing (TRAS = 1, TPC = 1)

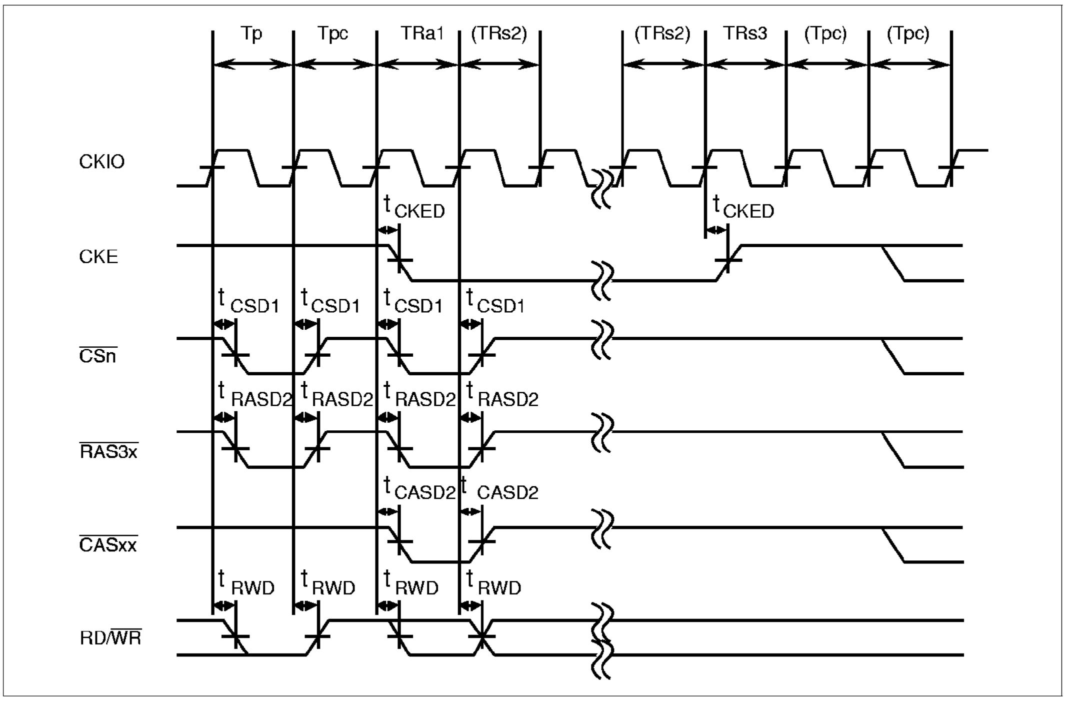


Figure 26.52 Synchronous DRAM Self-Refresh Cycle (TPC = 0)

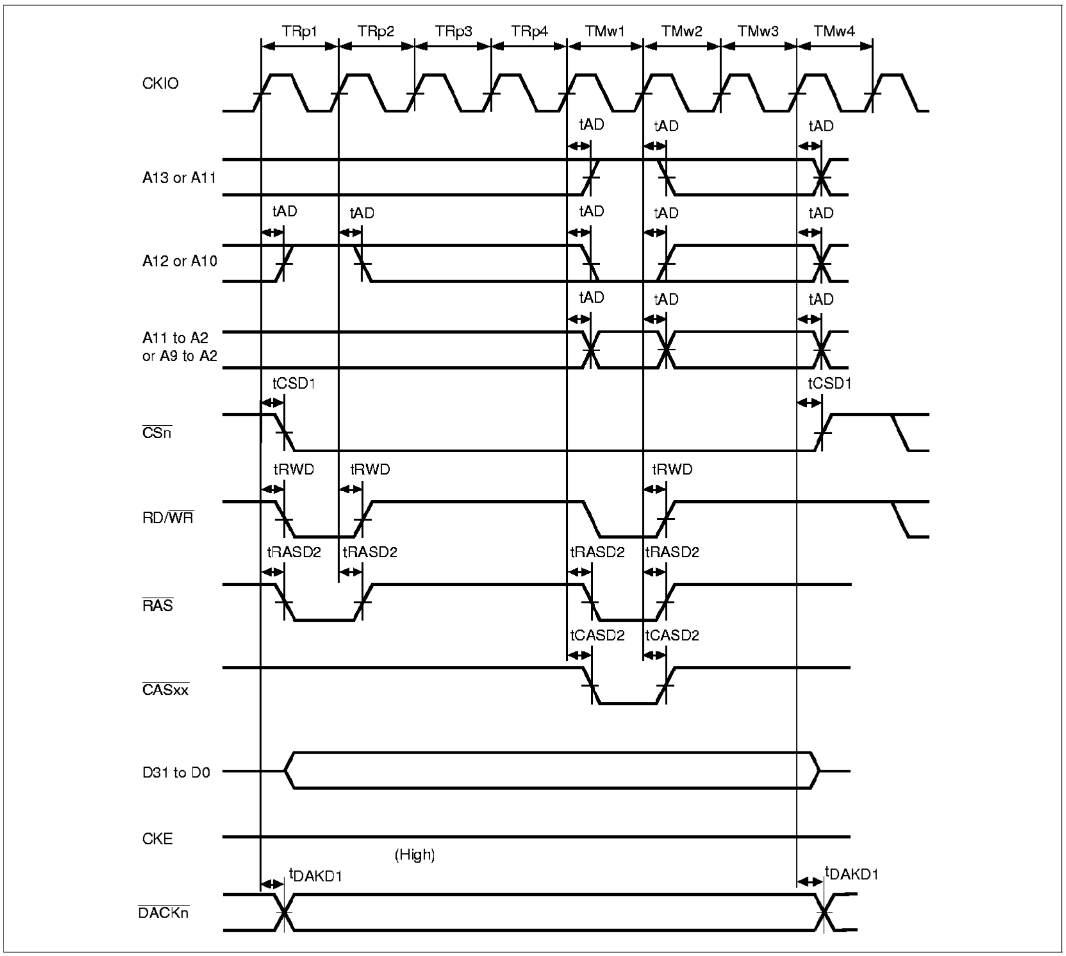


Figure 26.53 Synchronous DRAM Mode Register Write Cycle

26.3.8 PCMCIA Timing

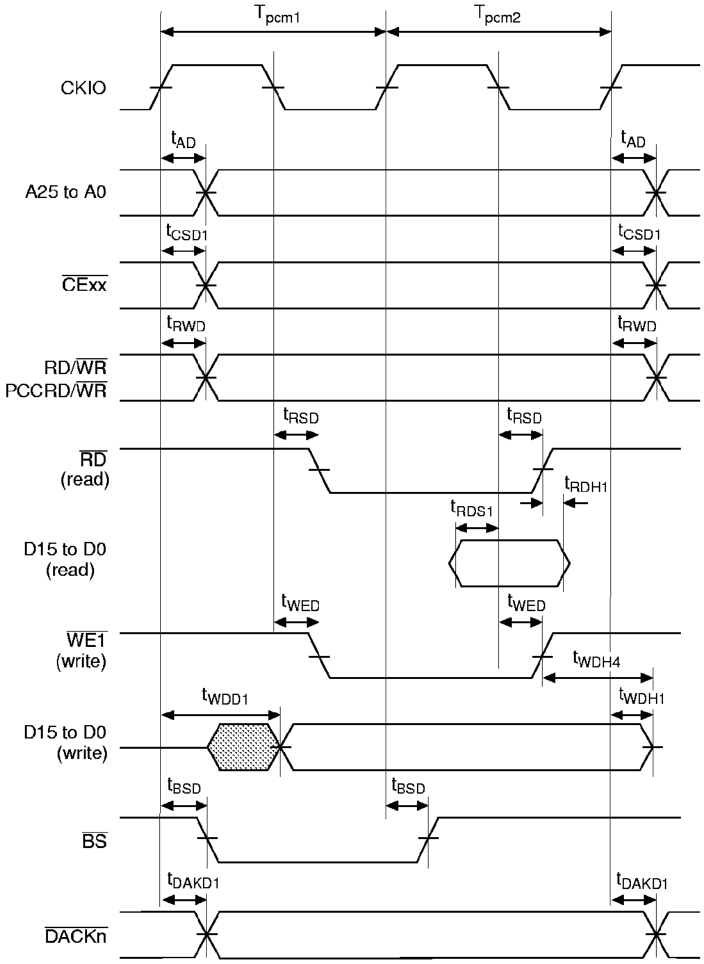


Figure 26.54 PCMCIA Memory Bus Cycle (TED = 0, TEH = 0, No Wait)

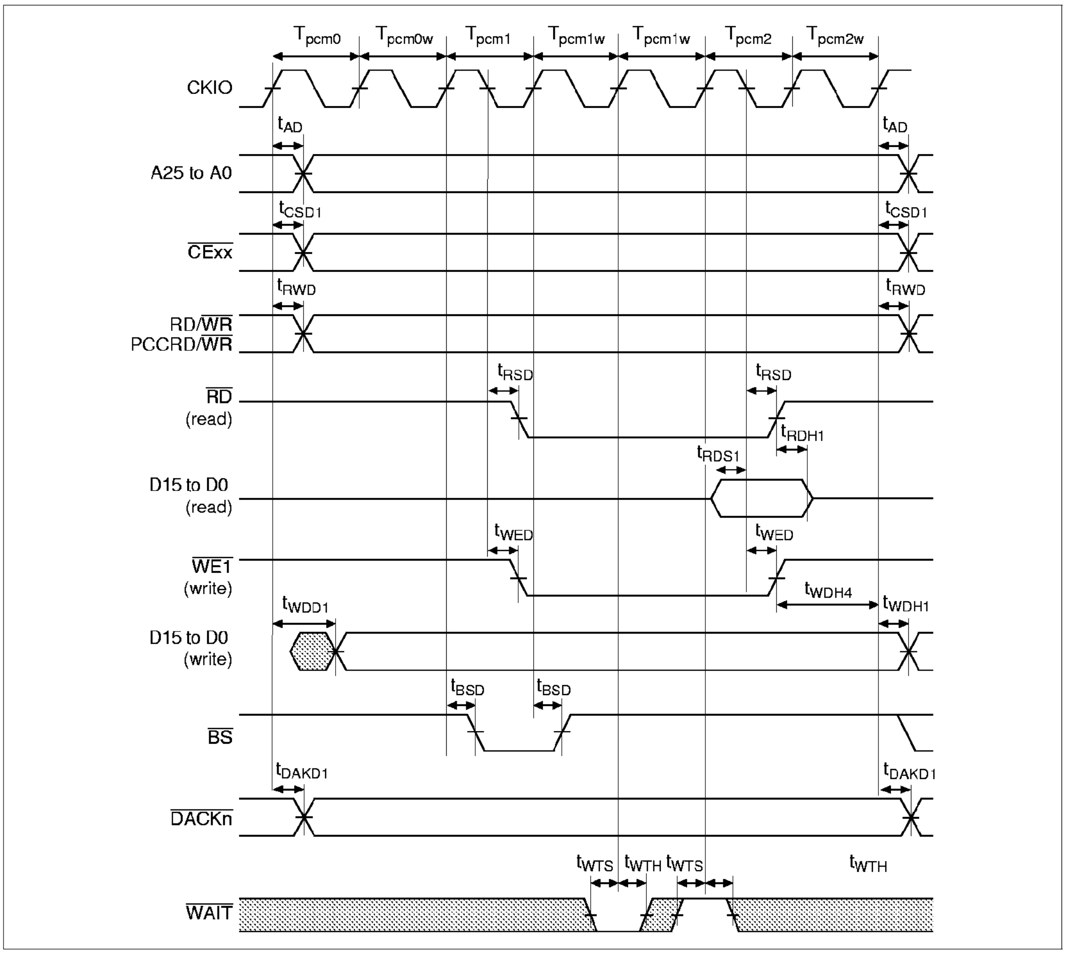
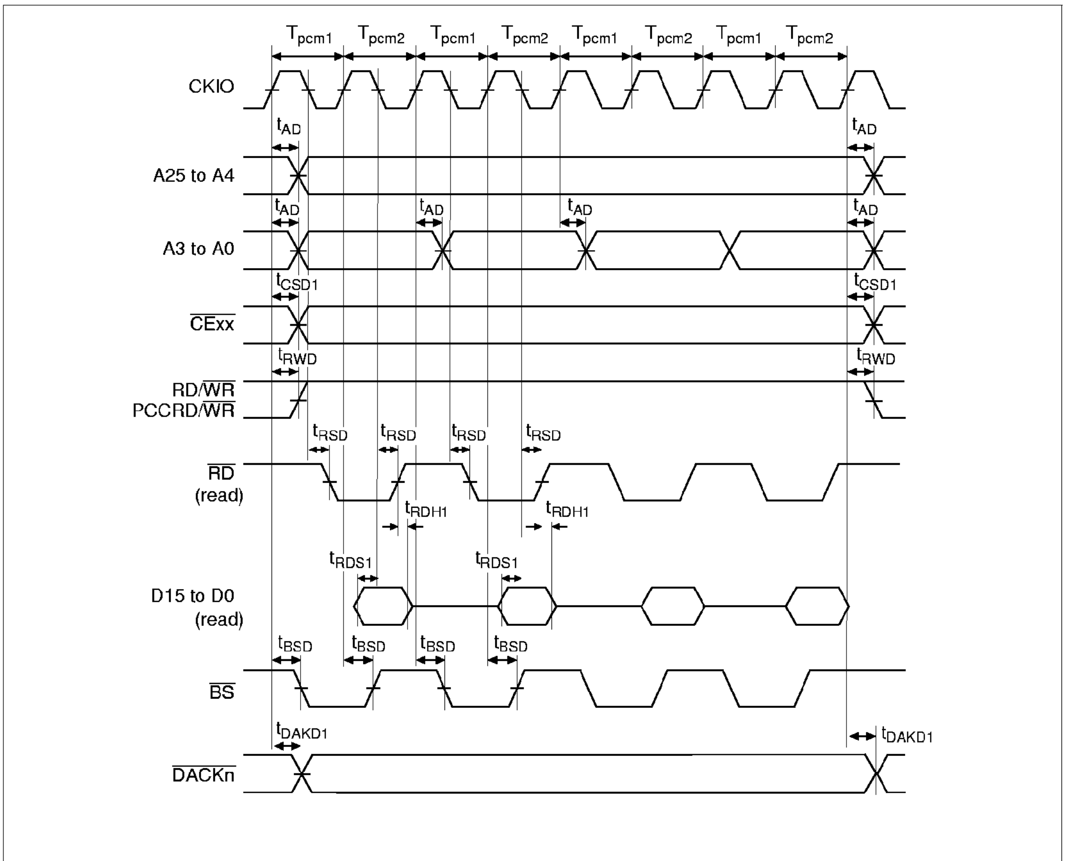
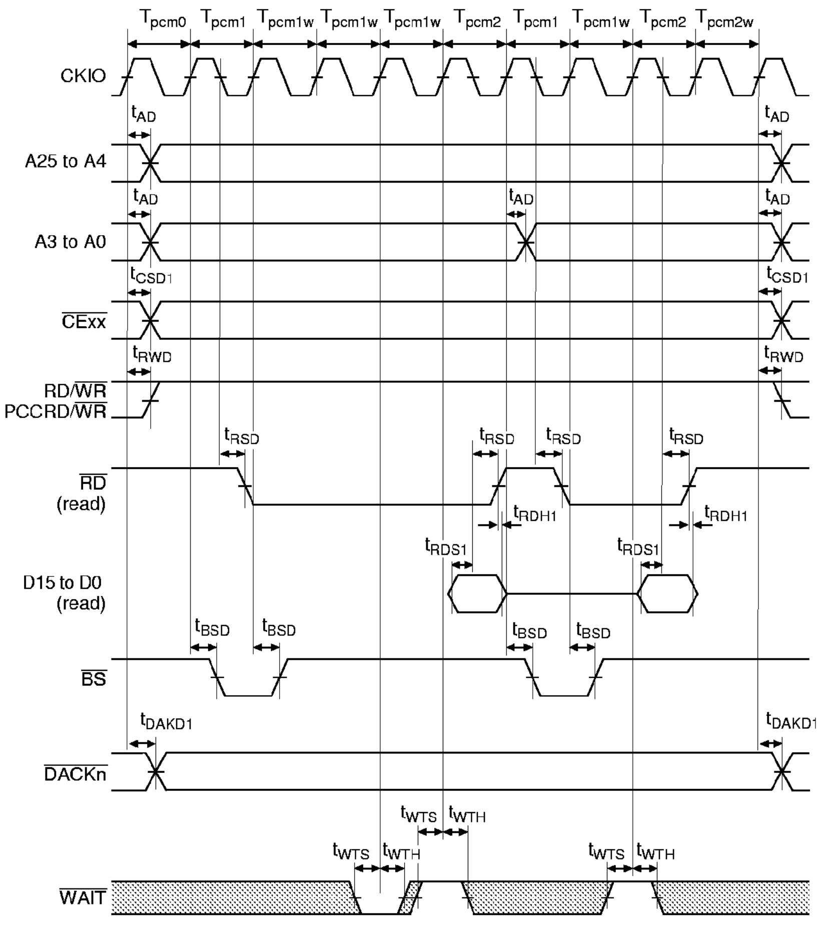


Figure 26.55 PCMCIA Memory Bus Cycle (TED = 2, TEH = 1, One Wait, External Wait, WAITSEL = 0)



Note: Even though burst mode is set, write cycle operation is the same as in normal mode.

Figure 26.56 PCMCIA Memory Bus Cycle (Burst Read, TED = 0, TEH = 0, No Wait)



Note: Even though burst mode is set, the write cycle operation is the same as in normal mode.

Figure 26.57 PCMCIA Memory Bus Cycle (Burst Read, TED = 1, TEH = 1, Two Waits, Burst Pitch = 3, WAITSEL = 0)

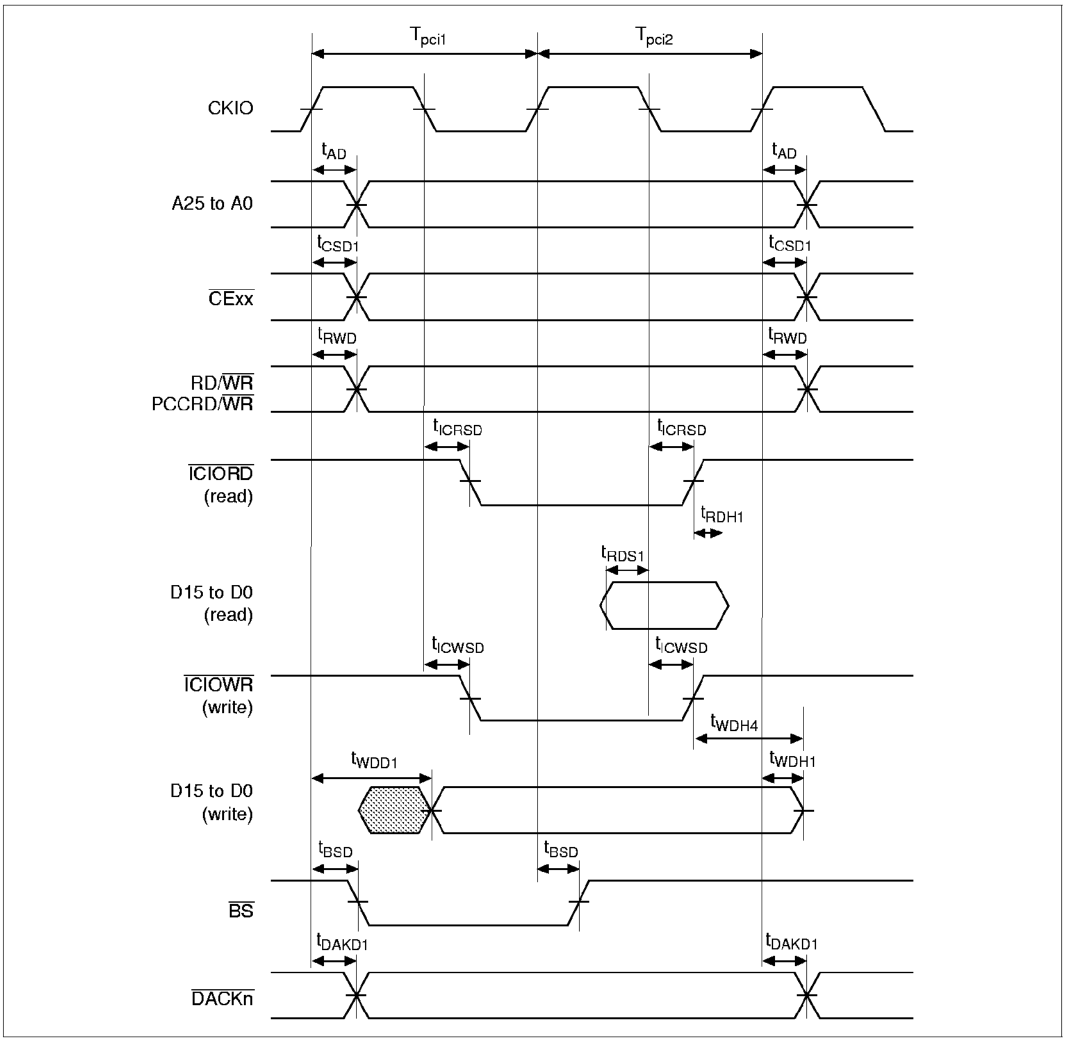


Figure 26.58 PCMCIA I/O Bus Cycle (TED = 0, TEH = 0, No Wait)

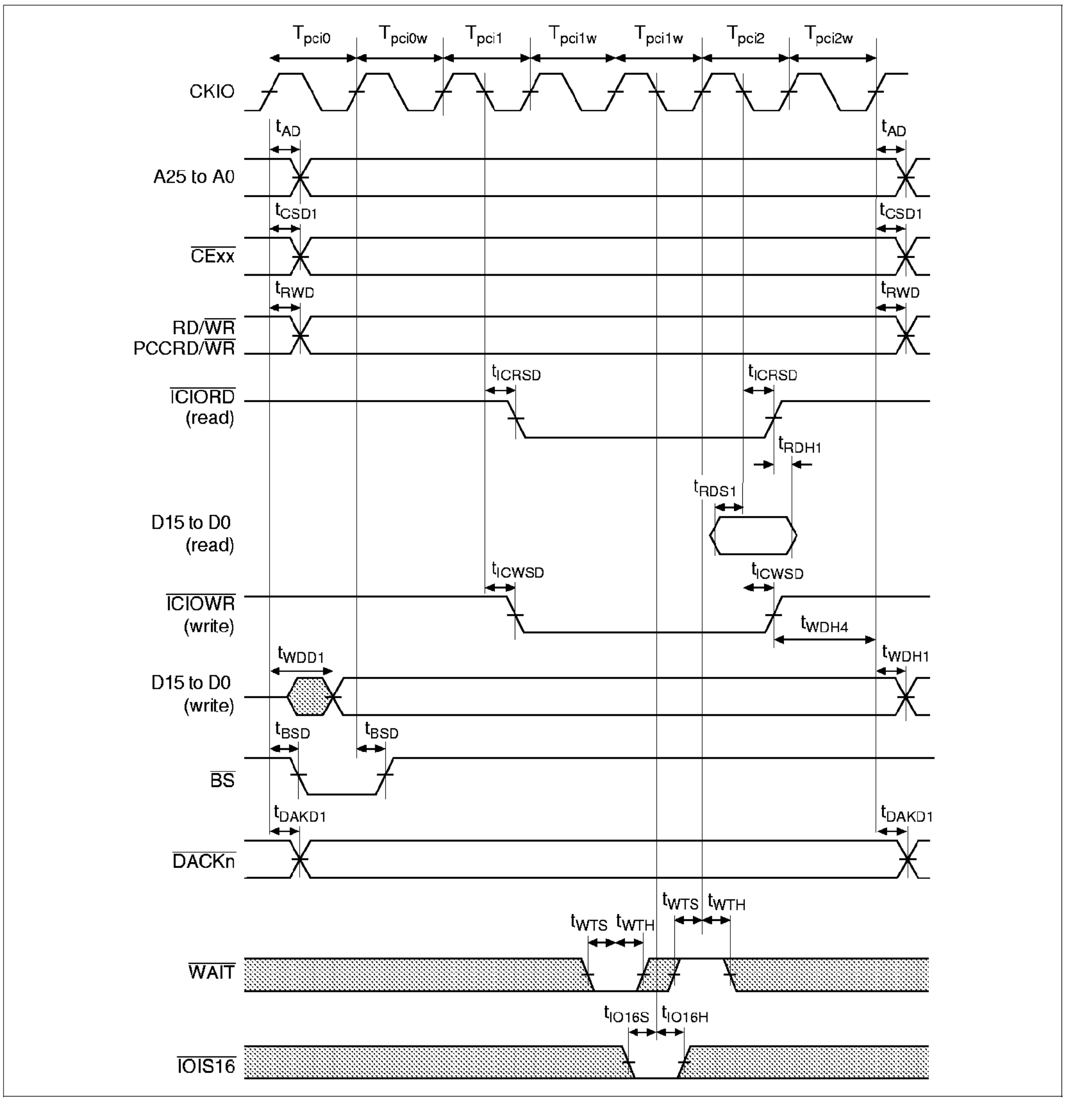


Figure 26.59 PCMCIA I/O Bus Cycle (TED = 2, TEH = 1, One Wait, External Wait, WAITSEL = 0)

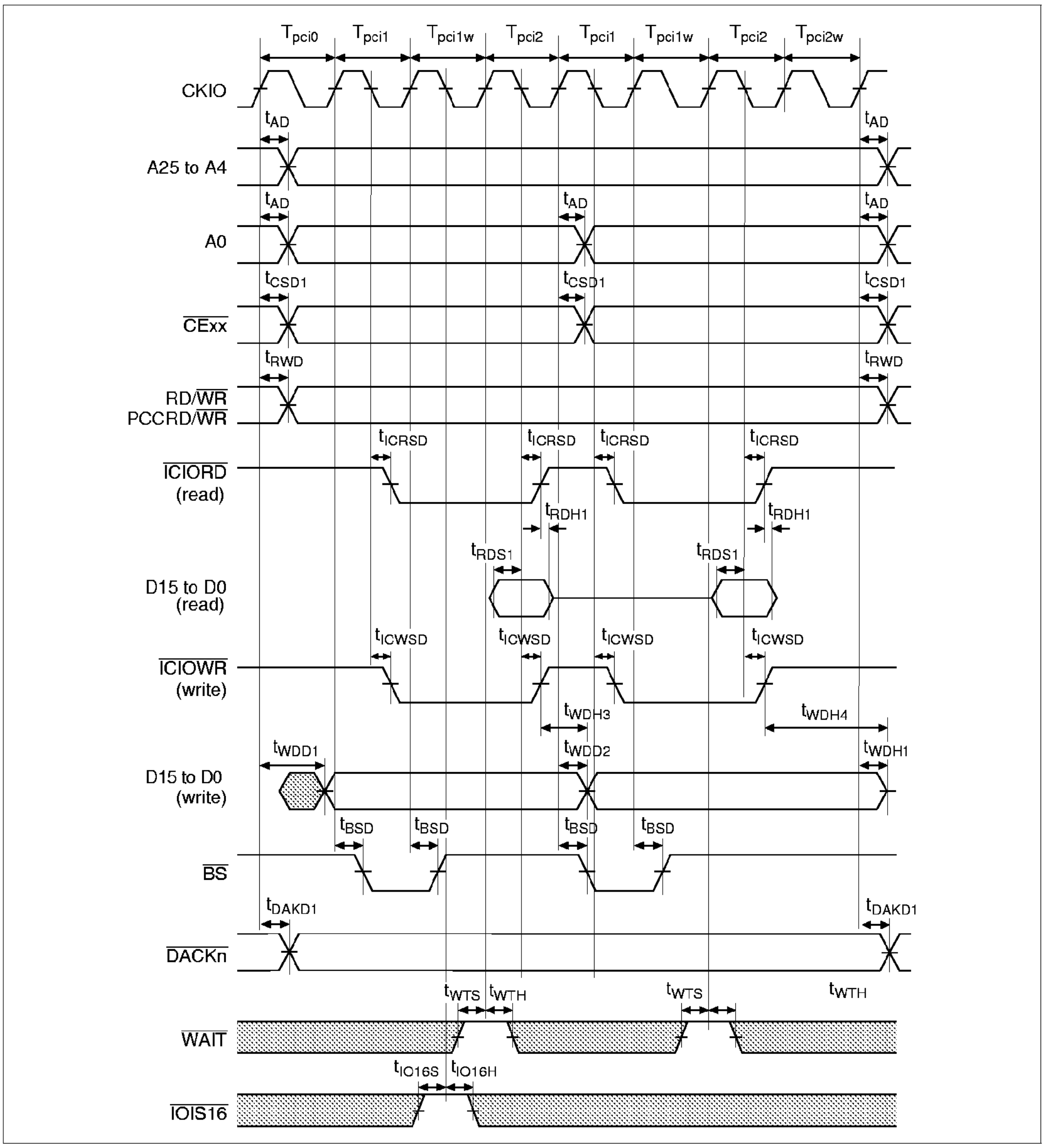


Figure 26.60 PCMCIA I/O Bus Cycle (TED = 1, TEH = 1, One Wait, Bus Sizing, WAITSEL = 0)

26.3.9 Peripheral Module Signal Timing

Table 26.8Peripheral Module Signal Timing — Preliminary —
(VccQ = 3.3 ± 0.3 V, Vcc = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

				−66			
Module	**Item**		**Symbol**	**Min**	**Max**	**Unit**	**Figure**
TMU,	Timer input setup time		t_{TCLKS}	15	—	ns	26.60
RTC	Timer clock input setup time		t_{TCKS}	15	—		26.61
	Timer clock pulse width	Edge specification	t_{TCKWH}	1.5	—	tcyc	
		Both edge specification	t_{TCKWL}	2.5	—		
	Oscillation settling time		t_{ROSC}	—	3	S	26.62
SCI	Input clock cycle	Asynchronization	t_{SCYC}	4	—	tcyc	26.63
		Clock synchronization		6	—		26.63, 26.64
	Input clock rise time		t_{SCKR}	—	1.5		26.63
	Input clock fall time		t_{SCKF}	—	1.5		
	Input clock pulse width		t_{SCKW}	0.4	0.6	tscyc	
	Transmission data delay time		t_{TXD}	—	100	ns	26.64
	Receive data setup time (clock synchronization)		t_{RXS}	100	—		
	Receive data hold time (clock synchronization)		t_{RXH}	100	—		
	$\overline{\text{RTS}}$ delay time		t_{RTSD}	—	100		
	$\overline{\text{CTS}}$ setup time (clock synchronization)		t_{CTSS}	100	—		
	$\overline{\text{CTS}}$ hold time (clock synchronization)		t_{CTSH}	100	—		
Port	Output data delay time		t_{PORTD}	—	17	ns	26.65
	Input data setup time		t_{PORTS1}	15	—		
	Input data hold time		t_{PORTH1}	8	—		
	Input data setup time		t_{PORTS2}	17	—		
	Input data hold time		t_{PORTH2}	10	—		
DMAC	$\overline{\text{DREQ}}$ setup time		t_{DREQ}	12	—	ns	26.66
	$\overline{\text{DREQ}}$ hold time		t_{DREQH}	8	—		
	$\overline{\text{DRAK}}$ delay time		t_{DRAKD}	14	—		26.67

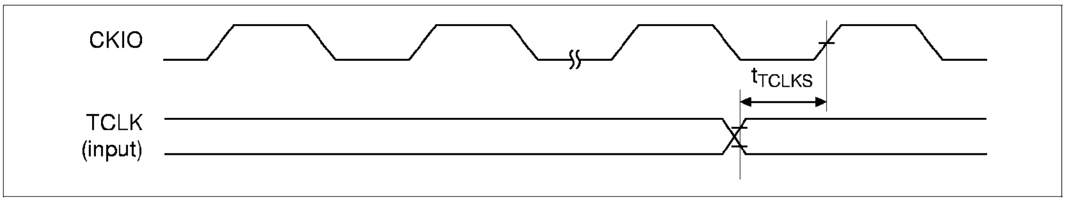


Figure 26.61 TCLK Input Timing

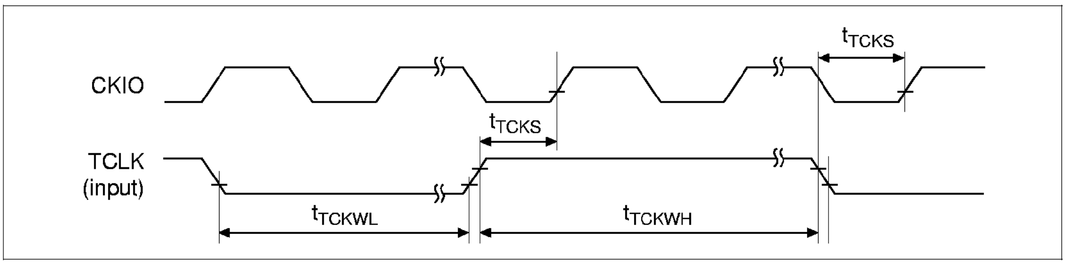


Figure 26.62 TCLK Clock Input Timing

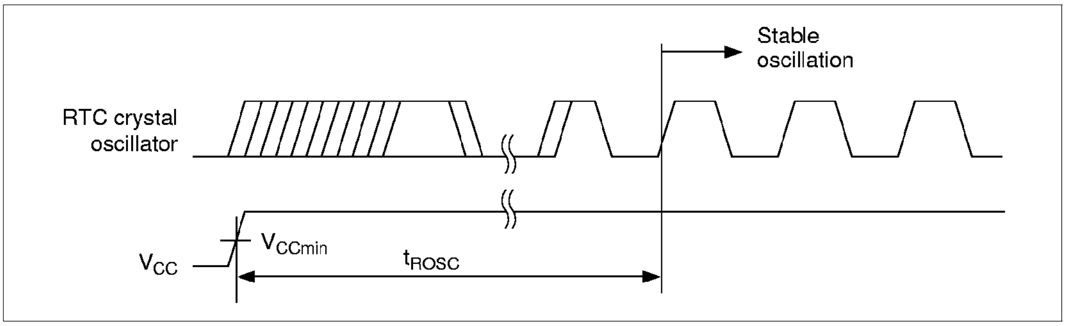


Figure 26.63 Oscillation Settling Time at RTC Crystal Oscillator Power-on

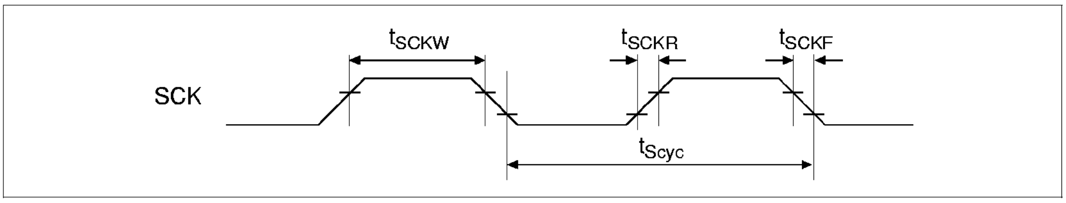


Figure 26.64 SCK Input Clock Timing

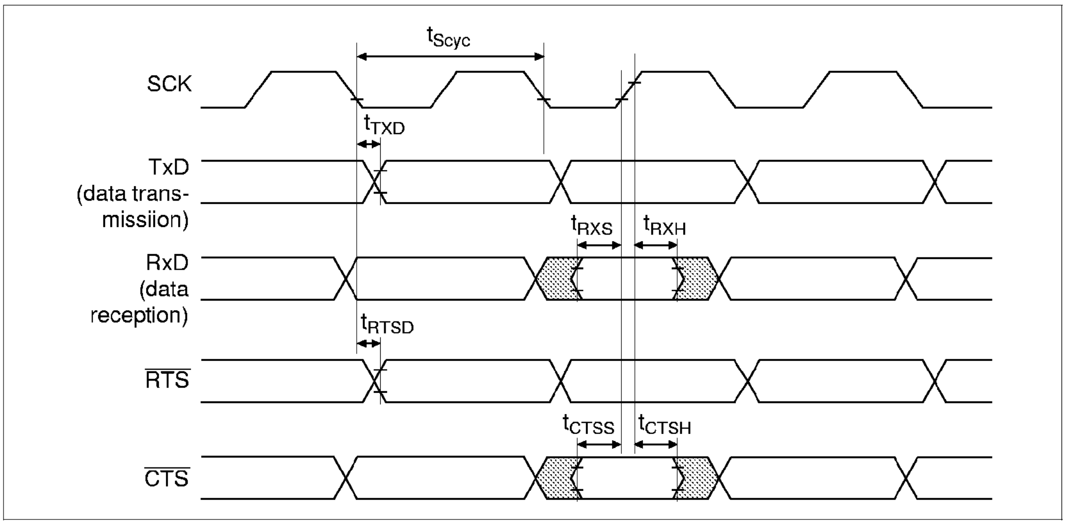


Figure 26.65 SCI I/O Timing in Clock Synchronous Mode

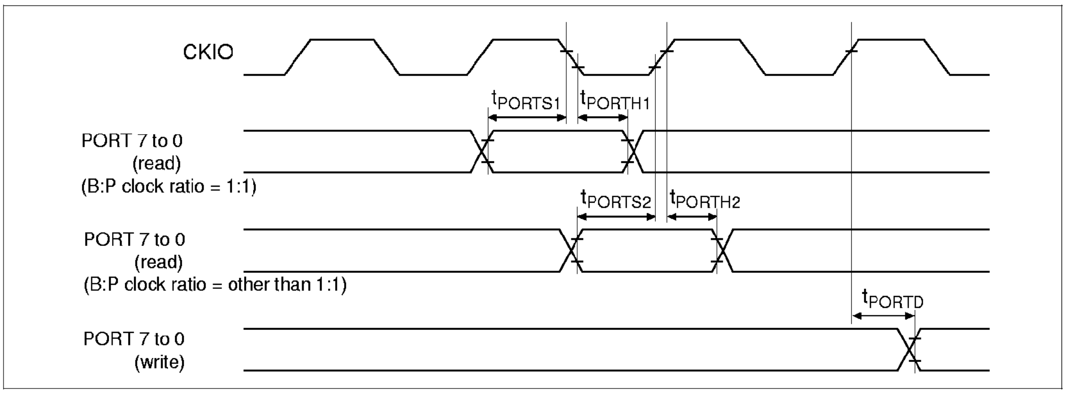


Figure 26.66 I/O Port Timing

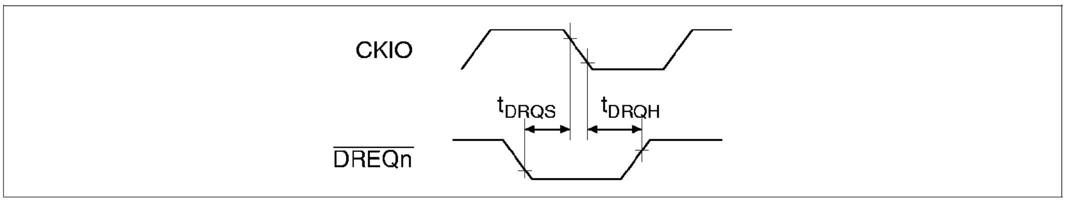


Figure 26.67 $\overline{\text{DREQ}}$ Input Timing

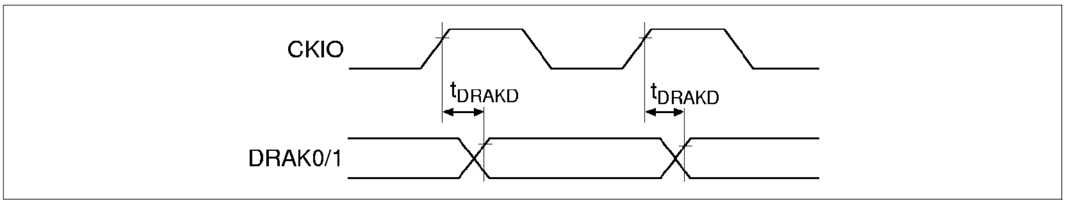


Figure 26.68 DRAK Output Timing

26.3.10 ASE-Related Pin Timing

Table 26.10 ASE-Related Pin Timing

Item	Symbol	Min	Max	Unit	Figure
TCK cycle time	t_{TCKcyc}	33	—	ns	Figure 26.69
TCK high pulse width	t_{TCKH}	10	—	ns	
TCK low pulse width	t_{TCKL}	10	—	ns	
TCK rise/fall time	t_{TCKf}	—	4	ns	
TRST setup time	t_{TRSTS}	12	—	ns	Figure 26.70
TRST hold time	t_{TRSTH}	50	—	t_{cyc}	
TDI setup time	t_{TDIS}	10	—	ns	Figure 26.71
TDI hold time	t_{TDIH}	10	—	ns	
TMS setup time	t_{TMSS}	10	—	ns	
TMS hold time	t_{TMSH}	10	—	ns	
TDO delay time	t_{TDOD}	—	16	ns	
ASEMD setup time	t_{ASEMDH}	12	—	ns	Figure 26.72
ASEMD hold time	t_{ASEMDS}	12	—	ns	
AUDCK cycle time	$t_{AUDCKcyc}$	16	—	ns	Figure 26.73
AUDATA delay time	t_{AUDATA}	—	12	ns	
AUDSYNC delay time	$t_{AUDSYNCD}$	—	12	ns	
AUDATA setup time	$t_{AUDATASD}$	12	—	ns	Figure 26.74
AUDATA hold time	$t_{AUDATAH}$	12	—	ns	

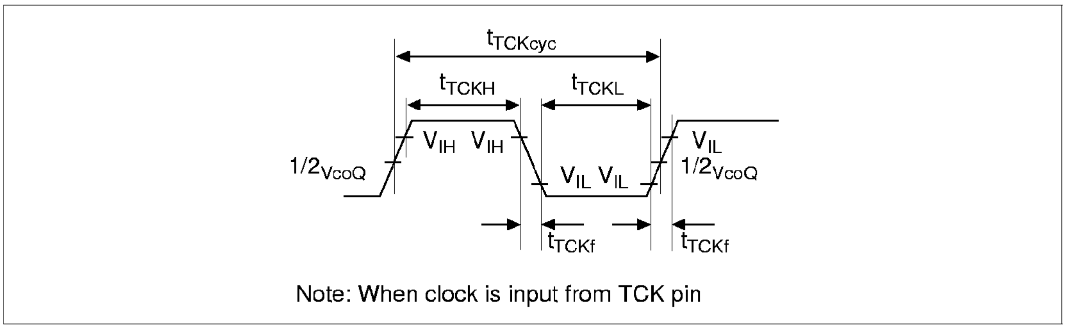


Figure 26.69 TCK Input Timing

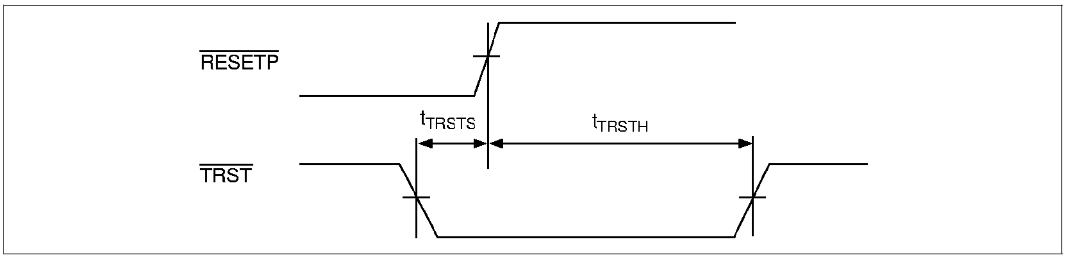


Figure 26.70 TRST Input Timing

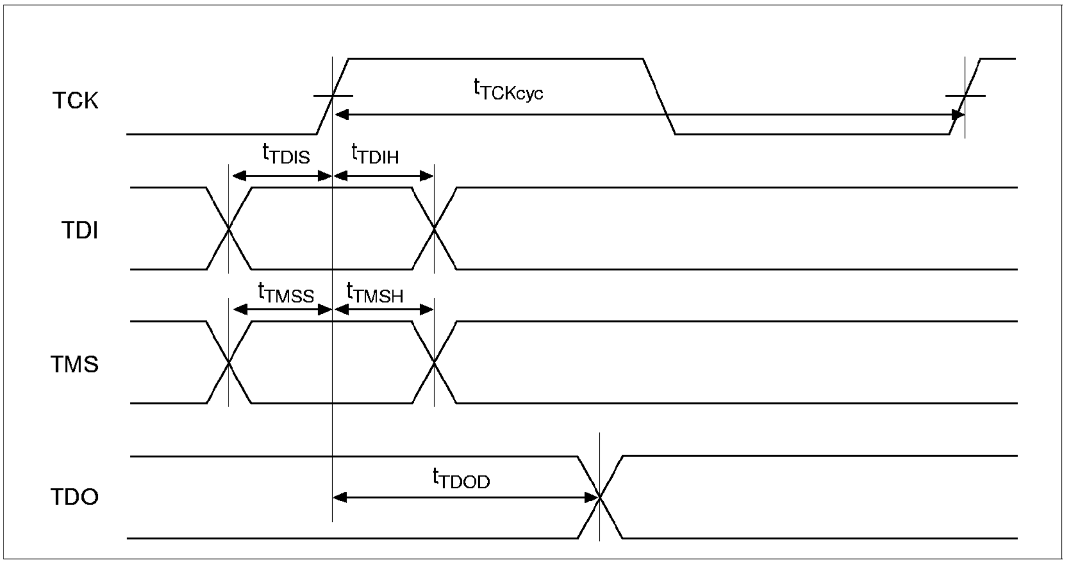


Figure 26.71 H-UDI Data Transfer Timing

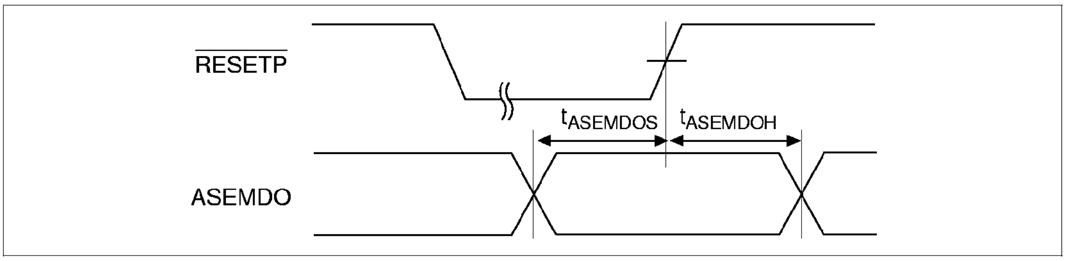


Figure 26.72 ASEMD0 Input Timing

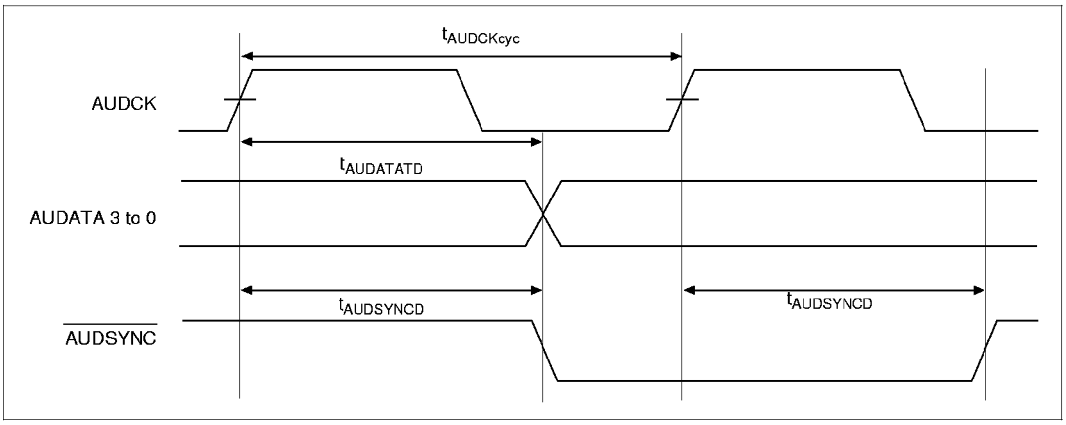


Figure 26.73 AUDATA3—0/AUDSYNC Output Timing

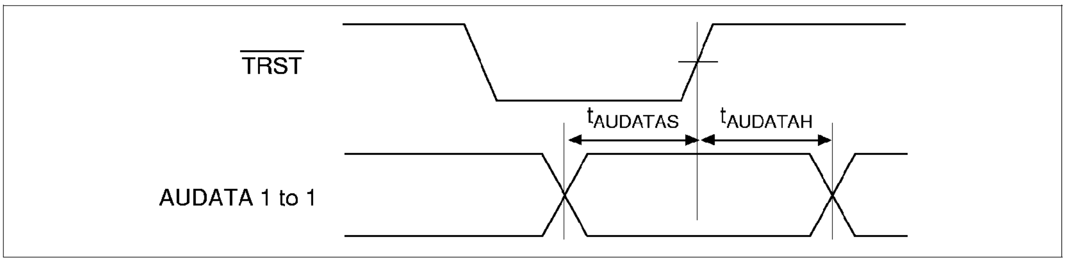
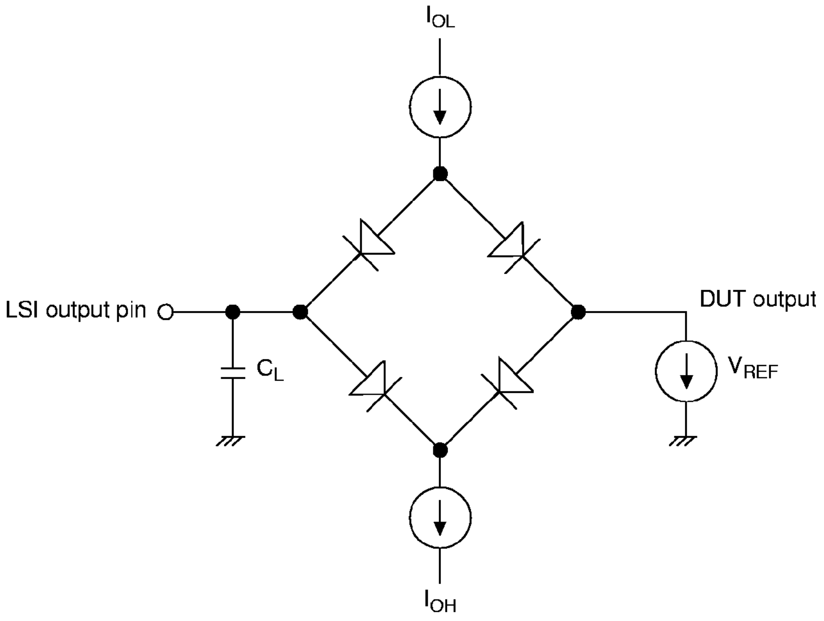


Figure 26.74 AUDATA1—0 Input Timing

26.3.11 AC Characteristics Measurement Conditions

- I/O signal reference level: 1.5 V (VccQ = 3.3 ± 0.3 V, Vcc = 1.8 ± 0.2 V)
- Input pulse level: Vss to 3.0 V (where $\overline{\text{RESETP}}$, $\overline{\text{RESETM}}$, NMI, $\overline{\text{IRQ5}}$–$\overline{\text{IRQ0}}$, CKIO, and MD5–MD0 are within Vss to Vcc)
- 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, RAS , CASxx, CS0 , CS2 –CS6 , CE2A , CE2B , BACK
50pF: All other pins
2. I_{OL} and I_{OH} are the values shown in table 22.3.

Figure 26.75 Output Load Circuit

26.4 A/D Converter Characteristics

Table 26.9 lists the A/D converter characteristics.

Table 26.9A/D Converter Characteristics — Preliminary —
(VccQ = 3.3 ± 0.3 V, Vcc = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

Item	Min	Typ	Max	Unit
Resolution	10	10	10	bits
Conversion time	—	—	8.9	µs
Analog input capacitance	—	—	20	pF
Permissible signal-source (single-source) impedance	—	—	5	kΩ
Nonlinearity error	—	—	±3.0	LSB
Offset error	—	—	±2.0	LSB
Full-scale error	—	—	±2.0	LSB
Quantization error	—	—	±0.5	LSB
Absolute accuracy	—	—	±4.0	LSB

26.5 D/A Converter Characteristics

Table 26.10 lists the D/A converter characteristics.

Table 26.10 D/A Converter Characteristics — Preliminary —
(VccQ = 3.3 ± 0.3 V, Vcc = 1.8 ± 0.2 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

Item	Min	Typ	Max	Unit	Test Conditions
Resolution	8	8	8	bits	
Conversion time	—	—	10.0	µs	20-pF capacitive load
Absolute accuracy	—	±2.5	±4.0	LSB	2-MΩ resistance load

Appendix A Pin Functions

A.1 Pin States

Table A.1 Pin States during Resets, Power-Down States, and Bus-Released State

		Reset		Power-Down		
Category	**Pin**	**Power-On Reset**	**Manual Reset**	**Standby**	**Sleep**	**Bus Released**
Clock	EXTAL	I	I	I	I	I
	XTAL	O*1	O*1	O*1	O*1	O*1
	CKIO	IO*1	IO*1	IO*1	IO*1	IO*1
	EXTAL2	I	I	I	I	I
	XTAL2	O	O	O	O	O
	CAP1, CAP2	—	—	—	—	—
System control	RESETP	I	I	I	I	I
	RESETM	I	I	I	I	I
	BREQ	I	I	I	I	
	BACK	O	O	O	O	L
	MD[5:0]	I	I	I	I	I
	CA	I	I	I	I	I
	STATUS[1:0]/PTJ[7:6]	O	OP*3	OP*3	OP*3	OP*3
Interrupt	IRL[3:0]/IRQ[3:0]/ PTH[3:0]	V*8	I	I	I	I
	IRQ4/ PTH[4]	V*8	I	I	I	I
	NMI	I	I	I	I	I
	IRL[3:0]/PINT[11:8]/ PTF[3:0]	V	I	IZ	I	I
	MCS[7:0]/PINT[7:0]/ PTC[7:0]	V	OP*3	ZH*11 K*3	OP*3	ZP*3
	TCK/PTF4/PINT12	IV	I	IZ	I	I
	TDI/PTFS/PINT13	IV	I	IZ	I	I
	TMS/PTF6/PINT14	IV	I	IZ	I	I
	TRST/PTF7/PINT15	IV	I	IZ	I	I
	IRQOUT	O	O	O	O	O

Table A.1 Pin States during Resets, Power-Down States, and Bus-Released State (cont)

		Reset		Power-Down		
Category	**Pin**	**Power-On Reset**	**Manual Reset**	**Standby**	**Sleep**	**Bus Released**
Address bus	A[25:0]	Z	O	ZL[*10]	O	Z
Data bus	D[15:0]	Z	I	Z	IO	Z
	D[23:16]/PTA[7:0]	Z	IP[*3]	ZK[*3]	IOP[*3]	ZP[*3]
	D[31:24]/PTB[7;0]	Z	IP[*3]	ZK[*3]	IOP[*3]	ZP[*3]
Bus control	CS0/MCS0	H	O	ZH[*11]	O	Z
	CS[2:4]/PTK[0:2]	H	OP[*3]	ZH[*11] K[*3]	OP[*3]	ZP[*3]
	CS5/CE1A/PTK[3]	H	OP[*3]	ZH[*11] K[*3]	OP[*3]	ZP[*3]
	CS6/CE1B	H	O	ZH[*11]	O	Z
	BS/PTK[4]	H	OP[*3]	ZH[*11] K[*3]	OP[*3]	ZP[*3]
	RAS3L/PTJ[0]	H	OP[*3]	ZOK[*4]	OP[*3]	ZOP[*4]
	RAS2L/PTJ[1]	H	OP[*3]	ZOK[*4]	OP[*3]	ZOP[*4]
	RAS3U/PTE[2]	V	OP[*3]	ZOK[*4]	OP[*3]	ZOP[*4]
	RAS2U/PTE[1]	V	OP[*3]	ZOK[*4]	OP[*3]	ZOP[*4]
	CAS2L/PTE[6]	V	OP[*3]	ZOK[*4]	OP[*3]	ZOP[*4]
	CAS2H/PTE[3]	V	OP[*3]	ZOK[*4]	OP[*3]	ZOP[*4]
	CASLL/CAS/PTJ[2]	H	OP[*3]	ZOK[*4]	OP[*3]	ZOP[*4]
	CASLH/PTJ[3]	H	OP[*3]	ZOK[*4]	OP[*3]	ZOP[*4]
	CASHL/ PTJ[4]	H	OP[*3]	ZOK[*4]	OP[*3]	ZOP[*4]
	CASHH/ PTJ[5]	H	OP[*3]	ZOK[*4]	OP[*3]	ZOP[*4]
	WE0/DQMLL	H	O	ZH[*11]	O	Z
	WE1/DQMLU/WE	H	O	ZH[*11]	O	Z
	WE2/DQMUL/ICIORD/ PTK[6]	H	OP[*3]	ZH[*11] K[*3]	OP[*3]	ZP[*3]
	WE3/DQMUU/ICIOWR/ PTK[7]	H	OP[*3]	ZH[*11] K[*3]	OP[*3]	ZP[*3]
	RDWR	H	O	ZH[*11]	O	Z
	RD	H	O	ZH[*11]	O	Z
	CKE/PTK[5]	H	OP[*3]	OK[*3]	OP[*3]	OP[*3]
	WAIT	Z	I	Z	I	Z

Table A.1 Pin States during Resets, Power-Down States, and Bus-Released State (cont)

		Reset		Power-Down		
Category	Pin	Power-On Reset	Manual Reset	Standby	Sleep	Bus Released
DMAC	DREQ0/PTD[4]	V	ZI^{*7}	Z	I	I
	DACK0/PTD[5]	V	OP^{*3}	ZK^{*3}	OP^{*3}	OP^{*3}
	DRAK0/PTD[1]	V	OP^{*3}	ZH^{*11} K^{*3}	OP^{*3}	OP^{*3}
	DREQ1/PTD[6]	V	ZI^{*7}	Z	I	I
	DACK1/PTD[7]	V	OP^{*3}	ZK^{*3}	OP^{*3}	OP^{*3}
	DRAK1/PTD[0]	V	OP^{*3}	ZH^{*11} K^{*3}	OP^{*3}	OP^{*3}
Timer	TCLK/PTH[7]	V	ZP	IOP^{*5}	IOP^{*5}	IOP^{*5}
SCI/ Smart card without FIFO	RxD0/SCPT[0]	Z	ZI^{*7}	Z	IZ^{*6}	IZ^{*6}
	TxD0/SCPT[0]	Z	ZO^{*7}	ZK^{*3}	OZ^{*6}	OZ^{*6}
	SCK0/SCPT[1]	V	ZP^{*3}	ZK^{*3}	IOP^{*5}	IOP^{*5}
SCIF/IrDA with FIFO	RxD1/SCPT[2]	Z	ZI^{*7}	Z	IZ^{*6}	IZ^{*6}
	TxD1/SCPT[2]	Z	ZO^{*7}	ZK^{*3}	OZ^{*6}	OZ^{*6}
	SCK1/SCPT[3]	V	ZP^{*3}	ZK^{*3}	IOP^{*5}	IOP^{*5}
SCIF with FIFO	RxD2/SCPT[4]	Z	ZI^{*7}	Z	IZ^{*6}	IZ^{*6}
	TxD2/SCPT[4]	Z	ZO^{*7}	ZK^{*3}	OZ^{*6}	OZ^{*6}
	SCK2/SCPT[5]	V	ZP^{*3}	ZK^{*3}	IOP^{*5}	IOP^{*5}
	$\overline{\text{RTS2}}$/SCPT[6]	V	OP^{*3}	ZK^{*3}	OP^{*3}	OP^{*3}
	$\overline{\text{CTS2}}$/IRQ5/SCPT[7]	V^{*8}	ZI^{*7}	I	I	I
Port	AUDSYNC/PTE[7]	OV	OP^{*3}	OK^{*3}	OP^{*3}	OP^{*3}
	CE2B/PTE[5]	V	OP^{*3}	ZH^{*11} K^{*3}	OP^{*3}	ZP^{*3}
	CE2A/PTE[4]	V	OP^{*3}	ZH^{*11} K^{*3}	OP^{*3}	ZP^{*3}
	TDO/PTE[0]	OV	OP^{*3}	OK^{*3}	OP^{*3}	OP^{*3}
	IOIS16/PTG[7]	V	I	Z	I	I
	PTG[6:0]	V	I	Z	I	I
	AUDCK/PHT[6]	V	I	Z	I	I
	ADTRG/PTH[5]	V^{*8}	I	IZ	I	I
	WAKEUP/PTD[3]	V	OP^{*3}	OK^{*3}	OP^{*3}	ZP^{*3}
	RESETOUT/PTD[2]	O	OP^{*3}	ZK^{*3}	OP^{*3}	OP^{*3}

Table A.1 Pin States during Resets, Power-Down States, and Bus-Released State (cont)

		Reset		Power-Down		
Category	Pin	Power-On Reset	Manual Reset	Standby	Sleep	Bus Released
Port	AUDATA[3:0]/PTG[3:0]	IV port	I	IZ port	I	I
	AUDATA[4]/PTG[4]	V	I	Z	I	I
	ASEBRKAK/PTG[5]	OV port	OI port	OZ port	OI port	OI port
	ASEMD0/PTG[6]	I (ASEMD)	I	Z	I	I
Analog	AN[5:0]/PTL[5:0]	Z	ZI*[7]	Z	I	I
	AN[6:7]/DA[1:0]/ PTL[6:7]	Z	ZI*[7]	I	IO*[9]	IO*[9]

I: Input
O: Output
H: High-level output
L: Low-level output
Z: High impedance
P: Input or output depending on register setting
K: Input pin is high impedance, output pin holds the state
V: I/O buffer off, pullup MOS on

Notes:
1. Depending on the clock mode (MD2–MD0 setting)
2. Z when the port function is used.
3. K or P when the port function is used.
4. K or P when the port function is used. Z or O when the port function is not used depending on register setting.
5. K or P when the port function is used. I or O when the port function is not used depending on register setting.
6. Depending on register setting
7. I or O when the port function is used.
8. Input Schmidt buffers and pullup MOS of IRQ[5:0] and ADTRG are on; other inputs are off.
9. I when the port function is used. I or O when the port function is not used, depending on register setting.
10. In the standby mode, Z or L depending on register setting.
11. In the standby mode, Z or H depending on register setting.

A.2 Pin Specifications

Table A.2 shows the pin specifications.

Table A.2 Pin Specifications

Pin	Pin No.	I/O	Function
MD5	197	I	Operating mode pin (endian mode)
MD4, MD3	196, 195	I	Operating mode pin (area 0 bus width)
MD2 to MD0	2, 1, 144	I	Operating mode pin (clock mode)
$\overline{\text{RAS3L}}$/PTJ[0]	106	I/O	RAS for area 3 lower 32 MB / I/O port
$\overline{\text{RAS2L}}$/PTJ[1]	107	I/O	RAS for area 2 lower 32 MB / I/O port
$\overline{\text{CE2A}}$/PTE[4]	103	I/O	PCMCIA CE2A / I/O port
$\overline{\text{CE2B}}$/PTE[5]	104	I/O	PCMCIA CE2B / I/O port
RXD0/SCPT[0]	171	I	Serial port 0 data input / input port
RXD1/SCPT[2]	172	I	Serial port 1 data input / input port
RXD2/SCPT[4]	174	I	Serial port 2 data input / input port
TXD0/SCPT[0]	164	O	Serial port 0 data output / output port
TXD1/SCPT[2]	166	O	Serial port 1 data output / output port
TXD2/SCPT[4]	168	O	Serial port 2 data output / output port
SCK0/SCPT[1]	165	I/O	Serial port 0 clock input/output / I/O port
SCK1/SCPT[3]	167	I/O	Serial port 1 clock input/output / I/O port
SCK2/SCPT[5]	169	I/O	Serial port 2 clock input/output / I/O port
$\overline{\text{RTS2}}$/SCPT[6]	170	I/O	Serial port 2 transfer request / I/O port
STATUS1/PTJ[7]	158	I/O	Processor state / I/O port
STATUS0/PTJ[6]	157	I/O	Processor state / I/O port
A25 to A0	86, 84, 82, 80, 78 to 72, 70, 68 to 60, 58, 56 to 53	O	Address bus
D31 to D24/ PTB[7] to PTB[0]	13 to 18, 20, 22	I/O	Data bus / I/O port
D23 to D16/ PTA[7] to PTA[0]	23 to 26, 28, 30 to 32	I/O	Data bus / I/O port
D15 to D0	34, 36 to 44, 46, 48 to 52	I/O	Data bus

Table A.2 Pin Specifications (cont)

Pin	Pin No.	I/O	Function
$\overline{\text{MCS}}$[7:0]/PTC[7:0]/ PINT[7:0]	177 to 180,185 to 188	I/O	Mask ROM chip select 7-0 / I/O port / port interrupt request
$\overline{\text{WAKEUP}}$/PTD[3]	182	I/O	Wakeup / I/O port
$\overline{\text{RESETOUT}}$/ PTD[2]	184	I/O	Reset output / I/O port
DRAK0/PTD[1]	189	I/O	DMA control pin / I/O port
DRAK1/PTD[0]	190	I/O	DMA control pin / I/O port
$\overline{\text{DREQ0}}$/PTD[4]	191	I	DMA transfer request 0 / input port
$\overline{\text{DREQ1}}$/PTD[6]	192	I	DMA transfer request 1 / input port
AN[5:0]/PTL[5:0]	204 to 199	I	Analog input pin / input port
AN[7:6]/DA[1:0]/ PTL[7:6]	207, 206	I/O	Analog I/O pin / input port
$\overline{\text{CS6}}$/$\overline{\text{CE1B}}$	102	O	Chip select 6 / PCMCIA CE1B
$\overline{\text{CS5}}$/$\overline{\text{CE1A}}$/ PTK[3]	101	I/O	Chip select 5 / PCMCIA CE2B / I/O port
$\overline{\text{CS4}}$/PTK[2]	100	I/O	Chip select 4 / I/O port
$\overline{\text{CS3}}$/PTK[1]	99	I/O	Chip select 3 / I/O port
$\overline{\text{CS2}}$/PTK[0]	98	I/O	Chip select 2 / I/O port
$\overline{\text{CS0}}$/$\overline{\text{MCS0}}$	96	O	Chip select 0 / Mask ROM chip select 0
$\overline{\text{BS}}$/PTK[4]	87	I/O	Bus cycle start / I/O port
$\overline{\text{CASHH}}$/PTJ[5]	113	I/O	D31–D24 selection CAS / I/O port
$\overline{\text{CASHL}}$/PTJ[4]	112	I/O	D23–D16 selection CAS / I/O port
$\overline{\text{CASLH}}$/$\overline{\text{CASU}}$/ PTJ[3]	110	I/O	D15–D8 selection CAS / I/O port
$\overline{\text{CASLL}}$/ $\overline{\text{CASL}}$/PTJ[2]	108	I/O	D7–D0 selection CAS / CAS(SDRAM) / I/O port
$\overline{\text{DACK0}}$/PTD[5]	114	I/O	DMA transfer strobe 0 / I/O port
$\overline{\text{DACK1}}$/PTD[7]	115	I/O	DMA transfer strobe 1 / I/O port
$\overline{\text{RD}}$	88	O	Read strobe pin
$\overline{\text{WE0}}$/ DQMLL	89	O	D7–D0 select signal/ DQM(SDRAM)
$\overline{\text{WE1}}$/DQMLU/$\overline{\text{WE}}$	90	O	D15–D8 select signal / DQM(SDRAM)/ PCMCIA $\overline{\text{WE}}$ signal
$\overline{\text{WE2}}$/DQMUL/ $\overline{\text{ICIORD}}$/PTK[6]	91	I/O	D23–D16 select signal / DQM(SDRAM) / PCMCIA $\overline{\text{IORD}}$ signal / I/O port

Table A.2 Pin Specifications (cont)

Pin	Pin No.	I/O	Function
$\overline{\text{WE3}}$/DQMUU/ $\overline{\text{ICIOWR}}$/PTK[7]	92	I/O	D31–D24 select signal /DQM(SDRAM) / PCMCIA $\overline{\text{IOWR}}$ signal / I/O port
RD/$\overline{\text{WR}}$	93	O	Read/write select signal
AUDSYNC/ PTE[7]	94	I/O	AUD synchronous I/O port
$\overline{\text{CAS2L}}$/PTE[6]	116	I/O	CAS for D7–D0 (area 2 DRAM) / I/O port
$\overline{\text{CAS2H}}$/PTE[3]	117	I/O	CAS for D15–D8 (area 2 DRAM) / I/O port
$\overline{\text{RAS3U}}$/PTE[2]	118	I/O	RAS for area 3 upper 32 MB/ I/O port
$\overline{\text{RAS2U}}$/PTE[1]	119	I/O	RAS for area 2 upper 32 MB / I/O port
TDO/PTE[0]	120	I/O	Test data output I/O port
$\overline{\text{RESETM}}$	124	I	Manual reset input
PTH[5]/$\overline{\text{ADTRG}}$	125	I	Input port / ADC trigger request
IOIS16/PTG[7]	126	I	I/O for PC card / input port
$\overline{\text{ASMDO}}$/PTG[6]	127	I	ASE mode / input port
$\overline{\text{ASEBRKAK}}$/ PTG[5]	128	I	ASE break accept / input port
PTG[4]	129	I	Input port
AUDATA[3]/ PTG[3]	130	I	AUD data / input port
AUDATA[2]/ PTG[2]	131	I	AUD data / input port
AUDATA[1]/ PTG[1]	133	I	AUD data / input port
AUDATA[0]/ PTG[0]	135	I	AUD data / input port
TRST/PTF[7]/ PINT[15]	136	I	Test reset / input port / port interrupt request
TRST/PTF[6]/ PINT[14]	137	I	Test mode switch / input port / port interrupt request
TRST/PTF[5]/ PINT[13]	138	I	Test data input / input port / port interrupt request
TRST/PTF[4]/ PINT[12]	139	I	Test clock / input port / port interrupt request
$\overline{\text{IRLS}}$[3:0]/ PTF[3:0]/ PINT[11:8]	140 to 143	I	External interrupt request / input port / port interrupt request

Table A.2 Pin Specifications (cont)

Pin	Pin No.	I/O	Function
AUDCK/PTH[6]	151	I	AUD clock / input port
$\overline{\text{WAIT}}$	123	I	Hardware wait request
$\overline{\text{BREQ}}$	122	I	Bus request
$\overline{\text{BACK}}$	121	O	Bus acknowledge
$\overline{\text{IRQOUT}}$	160	O	Interrupt / refresh request output
$\overline{\text{RESETP}}$	193	I	Power-on reset input
NMI	7	I	Nonmaskable interrupt request
IRQ[3:0]/$\overline{\text{IRL[3:0]}}$/ PTH[3:0]	11 to 8	I	External interrupt request / external interrupt source / input port
IRQ4/ PTH[4]	12	I	External interrupt request / input port
$\overline{\text{CTS2}}$/IRQ5/ SCPT[7]	176	I	Serial port 2 transfer enable / external interrupt request / input port
TCLK/PTH[7]	159	I/O	Clock I/O (for TMU/RTC) / I/O port
EXTAL	156	I	External clock / crystal oscillator pin
XTAL	155	O	Crystal oscillator pin
CAP1	146	—	External capacitance pin (for PLL1)
CAP2	149	—	External capacitance pin (for PLL2)
CKIO	162	I/O	System clock I/O
XTAL2	4	O	Crystal oscillator pin (for on-chip RTC)
EXTAL2	5	I	Crystal oscillator pin (for on-chip RTC)
CKE/PTK[5]	105	I/O	CK enable for SDRAM / I/O port
CA	194	I	Setting hardware standby
$V_{CC}Q$	21, 35, 47, 59, 71, 85, 97, 111, 163, 183	Power supply	Power supply (3.3 V), I/O buffer
V_{CC} – RTC	3	Power supply	RTC oscillator power supply (1.8 V)
V_{CC} – PLL	145, 150	Power supply	PLL power supply (1.8 V)
AV_{CC}	205	Power supply	Analog power supply (3.3 V)
$V_{SS}Q$	19, 33, 45, 57, 69, 83, 95, 109, 161, 181	Power supply	Power supply (0 V), I/O buffer

Table A.2 Pin Specifications (cont)

Pin	Pin No.	I/O	Function
V_{ss}	27, 79, 132, 152, 153, 173	Power supply	Internal power supply (0 V)
VCC	29, 81, 134, 154, 175	Power supply	Internal power supply (1.8 V)
V_{ss} – RTC	6	Power supply	RTC-oscillator power supply (0 V)
V_{ss} – PLL	147, 148	Power supply	PLL power supply (0 V)
AV_{ss}	198, 208	Power supply	Analog power supply (0 V)

Note: Power must be supplied constantly to all power-supply pins.

A.3 Processing of Unused Pins

- When RTC is not used
  - — EXTAL2: Pull up
  - — XTAL2: Leave unconnected
  - — V_{CC} – RTC: Power supply (1.8 V)
  - — V_{SS} – RTC: Power supply (0 V)
- When PLL1 is not used
  - — CAP1: Leave unconnected
  - — V_{CC} – PLL1: Power supply (1.8 V)
  - — V_{SS} – PLL1: Power supply (0 V)
- When PLL2 is not used
  - — CAP2: Leave unconnected
  - — V_{CC} – PLL2: Power supply (1.8 V)
  - — V_{SS} – PLL2: Power supply (0 V)
- When on-chip crystal oscillator is not used
  - — XTAL: Leave unconnected
- When EXTAL terminal is not used
  - — EXTAL: Pull up
- When A/D converter is not used
  - — AN[7:0]: Leave unconnected
  - — AV_{CC}: Power supply (3.3 V)
  - — AV_{SS}: Power supply (0 V)

A.4 Pin States in Access to Each Address Space

Table A.3 Pin States (Normal Memory/Little Endian)

		8-Bit Bus Width	16-Bit Bus Width		
Pin		Byte/Word/Long-word Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/Longword Access
$\overline{\text{CS6}}$ to $\overline{\text{CS2}}$, $\overline{\text{CS0}}$		Enabled	Enabled	Enabled	Enabled
RD	R	Low	Low	Low	Low
	W	High	High	High	High
RD/$\overline{\text{WR}}$	R	High	High	High	High
	W	Low	Low	Low	Low
$\overline{\text{BS}}$		Enabled	Enabled	Enabled	Enabled
$\overline{\text{RAS3U}}$/PTE[2]		High	High	High	High
$\overline{\text{RAS3L}}$/PTJ[0]		High	High	High	High
$\overline{\text{CASLL}}$/$\overline{\text{CAS}}$/PTJ[2]		High	High	High	High
$\overline{\text{CASLH}}$/PTJ[3]		High	High	High	High
$\overline{\text{CASHL}}$/PTJ[4]		High	High	High	High
$\overline{\text{CASHH}}$/PTJ[5]		High	High	High	High
$\overline{\text{CAS2L}}$/PTE[6]		High	High	High	High
$\overline{\text{CAS2H}}$/PTE[3]		High	High	High	High
$\overline{\text{WE0}}$/DQMLL	R	High	High	High	High
	W	Low	Low	High	Low
$\overline{\text{WE1}}$/$\overline{\text{WE}}$/DQMLU	R	High	High	High	High
	W	High	High	Low	Low
$\overline{\text{WE2}}$/$\overline{\text{ICIORD}}$/DQMUL/ PTK[6]	R	High	High	High	High
	W	High	High	High	High
$\overline{\text{WE3}}$/$\overline{\text{ICIOWR}}$/DQMUU/ PTK[7]	R	High	High	High	High
	W	High	High	High	High
$\overline{\text{CE2A}}$/PTE[4]		High	High	High	High
$\overline{\text{CE2B}}$/PTE[5]		High	High	High	High
$\overline{\text{RAS2U}}$/PTE[1]		High	High	High	High
$\overline{\text{RAS2L}}$/PTJ[1]		High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled
$\overline{\text{WAIT}}$		Enabled^{*1}	Enabled^{*1}	Enabled^{*1}	Enabled^{*1}
$\overline{\text{IOIS16}}$		Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address
D7 to D0		Valid data	Valid data	Invalid data	Valid data
D15 to D8		High-Z^{*2}	Invalid data	Valid data	Valid data
D31 to D16		High-Z^{*2}	High-Z^{*2}	High-Z^{*2}	High-Z^{*2}

Table A.3 Pin States (Normal Memory/Little Endian) (cont)

		32-Bit Bus Width						
Pin		**Byte Access (Address 4n)**	**Byte Access (Address 4n + 1)**	**Byte Access (Address 4n + 2)**	**Byte Access (Address 4n + 3)**	**Word Access (Address 4n)**	**Word Access (Address 4n + 2)**	**Longword Access**
CS6 to CS2, CS0		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
RD	R	Low	Low	Low	Low	Low	Low	Low
	W	High	High	High	High	High	High	High
RD/WR	R	High	High	High	High	High	High	High
	W	Low	Low	Low	Low	Low	Low	Low
BS		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
RAS3U/PTE[2]		High	High	High	High	High	High	High
RAS3L/PTJ[0]		High	High	High	High	High	High	High
CASLL/CAS/PTJ[2]		High	High	High	High	High	High	High
CASLH/PTJ[3]		High	High	High	High	High	High	High
CASHL/PTJ[4]		High	High	High	High	High	High	High
CASHH/PTJ[5]		High	High	High	High	High	High	High
CAS2L/PTE[6]		High	High	High	High	High	High	High
CAS2H/PTE[3]		High	High	High	High	High	High	High
WE0/DQMLL	R	High	High	High	High	High	High	High
	W	Low	High	High	High	Low	High	Low
WE1/WE/DQMLU	R	High	High	High	High	High	High	High
	W	High	Low	High	High	Low	High	Low
WE2/ICIORD/DQMUL/ PTK[6]	R	High	High	High	High	High	High	High
	W	High	High	Low	High	High	Low	Low
WE3/ICIOWR/DQMUU/ PTK[7]	R	High	High	High	High	High	High	High
	W	High	High	High	Low	High	Low	Low
CE2A/PTE[4]		High	High	High	High	High	High	High
CE2B/PTE[5]		High	High	High	High	High	High	High
RAS2U/PTE[1]		High	High	High	High	High	High	High
RAS2L/PTJ[1]		High	High	High	High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
WAIT		Enabled*1	Enabled*1	Enabled*1	Enabled*1	Enabled*1	Enabled*1	Enabled*1
IOIS16		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address	Address	Address	Address
D7 to D0		Valid data	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data

Table A.3 Pin States (Normal Memory/Little Endian) (cont)

	32-Bit Bus Width						
Pin	**Byte Access (Address 4n)**	**Byte Access (Address 4n + 1)**	**Byte Access (Address 4n + 2)**	**Byte Access (Address 4n + 3)**	**Word Access (Address 4n)**	**Word Access (Address 4n + 2)**	**Longword Access**
D15 to D8	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D23 to D16	Invalid data	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Valid data
D31 to D24	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data	Valid data

Notes: 1. Disabled when WCR2 register wait setting is 0.

2. Unused data pins should be switched to the port function, or pulled up or down.

Table A.4 Pin States (Normal Memory/Big Endian)

		8-Bit Bus Width	16-Bit Bus Width		
Pin		Byte/Word/Long-word Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/Longword Access
CS6 to CS2, CS0		Enabled	Enabled	Enabled	Enabled
RD	R	Low	Low	Low	Low
	W	High	High	High	High
RD/WR	R	High	High	High	High
	W	Low	Low	Low	Low
BS		Enabled	Enabled	Enabled	Enabled
RAS3U/PTE[2]		High	High	High	High
RAS3L/PTJ[0]		High	High	High	High
CASLL/CAS/PTJ[2]		High	High	High	High
CASLH/PTJ[3]		High	High	High	High
CASHL/PTJ[4]		High	High	High	High
CASHH/PTJ[5]		High	High	High	High
CAS2L/PTE[6]		High	High	High	High
CAS2H/PTE[3]		High	High	High	High
WE0/DQMLL	R	High	High	High	High
	W	Low	High	Low	Low
WE1/WE/DQMLU	R	High	High	High	High
	W	High	Low	High	Low
WE2/ICIORD/DQMUL/ PTK[6]	R	High	High	High	High
	W	High	High	High	High
WE3/ICIOWR/DQMUU/ PTK[7]	R	High	High	High	High
	W	High	High	High	High
CE2A/PTE[4]		High	High	High	High
CE2B/PTE[5]		High	High	High	High
RAS2U/PTE[1]		High	High	High	High
RAS2L/PTJ[1]		High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled
WAIT		Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]
IOIS16		Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address
D7 to D0		Valid data	Invalid data	Valid data	Valid data
D15 to D8		High-Z[*2]	Valid data	Invalid data	Valid data
D31 to D16		High-Z[*2]	High-Z[*2]	High-Z[*2]	High-Z[*2]

Table A.4 Pin States (Normal Memory/Big Endian) (cont)

		32-Bit Bus Width						
Pin		**Byte Access (Address 4n)**	**Byte Access (Address 4n + 1)**	**Byte Access (Address 4n + 2)**	**Byte Access (Address 4n + 3)**	**Word Access (Address 4n)**	**Word Access (Address 4n + 2)**	**Longword Access**
$\overline{\text{CS6}}$ to $\overline{\text{CS2}}$, $\overline{\text{CS0}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RD}}$	R	Low	Low	Low	Low	Low	Low	Low
	W	High	High	High	High	High	High	High
RD/$\overline{\text{WR}}$	R	High	High	High	High	High	High	High
	W	Low	Low	Low	Low	Low	Low	Low
$\overline{\text{BS}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RAS3U}}$/PTE[2]		High	High	High	High	High	High	High
$\overline{\text{RAS3L}}$/PTJ[0]		High	High	High	High	High	High	High
$\overline{\text{CASLL}}$/$\overline{\text{CAS}}$/PTJ[2]		High	High	High	High	High	High	High
$\overline{\text{CASLH}}$/PTJ[3]		High	High	High	High	High	High	High
$\overline{\text{CASHL}}$/PTJ[4]		High	High	High	High	High	High	High
$\overline{\text{CASHH}}$/PTJ[5]		High	High	High	High	High	High	High
$\overline{\text{CAS2L}}$/PTE[6]		High	High	High	High	High	High	High
$\overline{\text{CAS2H}}$/PTE[3]		High	High	High	High	High	High	High
$\overline{\text{WE0}}$/DQMLL	R	High	High	High	High	High	High	High
	W	High	High	High	Low	High	Low	Low
$\overline{\text{WE1}}$/$\overline{\text{WE}}$/DQMLU	R	High	High	High	High	High	High	High
	W	High	High	Low	High	High	Low	Low
$\overline{\text{WE2}}$/$\overline{\text{ICIORD}}$/DQMUL/ PTK[6]	R	High	High	High	High	High	High	High
	W	High	Low	High	High	Low	High	Low
$\overline{\text{WE3}}$/$\overline{\text{ICIOWR}}$/DQMUU/ PTK[7]	R	High	High	High	High	High	High	High
	W	Low	High	High	High	Low	High	Low
$\overline{\text{CE2A}}$/PTE[4]		High	High	High	High	High	High	High
$\overline{\text{CE2B}}$/PTE[5]		High	High	High	High	High	High	High
$\overline{\text{RAS2U}}$/PTE[1]		High	High	High	High	High	High	High
$\overline{\text{RAS2L}}$/PTJ[1]		High	High	High	High	High	High	High

Table A.4 Pin States (Normal Memory/Big Endian) (cont)

	32-Bit Bus Width						
Pin	**Byte Access (Address 4n)**	**Byte Access (Address 4n + 1)**	**Byte Access (Address 4n + 2)**	**Byte Access (Address 4n + 3)**	**Word Access (Address 4n)**	**Word Access (Address 4n + 2)**	**Longword Access**
CKE	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{\text{WAIT}}$	Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]
$\overline{\text{IOIS16}}$	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
A25 to A0	Address	Address	Address	Address	Address	Address	Address
D7 to D0	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data	Valid data
D15 to D8	Invalid data	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Valid data
D23 to D16	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D31 to D24	Valid data	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data

Notes: 1. Disabled when WCR2 register wait setting is 0.

2. Unused data pins should be switched to the port function, or pulled up or down.

Table A.5 Pin States (Burst ROM/Little Endian)

		8-Bit Bus Width	16-Bit Bus Width		
Pin		Byte/Word/Longword Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/Longword Access
$\overline{\text{CS6}}$ to $\overline{\text{CS2}}$, $\overline{\text{CS0}}$		Enabled	Enabled	Enabled	Enabled
$\overline{\text{RD}}$	R	Low	Low	Low	Low
	W	—	—	—	—
RD/$\overline{\text{WR}}$	R	High	High	High	High
	W	—	—	—	—
$\overline{\text{BS}}$		Enabled	Enabled	Enabled	Enabled
$\overline{\text{RAS3U}}$/PTE[2]		High	High	High	High
$\overline{\text{RAS3L}}$/PTJ[0]		High	High	High	High
$\overline{\text{CASLL}}$/$\overline{\text{CAS}}$/PTJ[2]		High	High	High	High
$\overline{\text{CASLH}}$/PTJ[3]		High	High	High	High
$\overline{\text{CASHL}}$/PTJ[4]		High	High	High	High
$\overline{\text{CASHH}}$/PTJ[5]		High	High	High	High
$\overline{\text{CAS2L}}$/PTE[6]		High	High	High	High
$\overline{\text{CAS2H}}$/PTE[3]		High	High	High	High
$\overline{\text{WE0}}$/DQMLL	R	High	High	High	High
	W	—	—	—	—
$\overline{\text{WE1}}$/$\overline{\text{WE}}$/DQMLU	R	High	High	High	High
	W	—	—	—	—
$\overline{\text{WE2}}$/$\overline{\text{ICIORD}}$/DQMUL/PTK[6]	R	High	High	High	High
	W	—	—	—	—
$\overline{\text{WE3}}$/$\overline{\text{ICIOWR}}$/DQMUU/PTK[7]	R	High	High	High	High
	W	—	—	—	—
$\overline{\text{CE2A}}$/PTE[4]		High	High	High	High
$\overline{\text{CE2B}}$/PTE[5]		High	High	High	High
$\overline{\text{RAS2U}}$/PTE[1]		High	High	High	High
$\overline{\text{RAS2L}}$/PTJ[1]		High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled
$\overline{\text{WAIT}}$		Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]
$\overline{\text{IOIS16}}$		Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address
D7 to D0		Valid data	Valid data	Invalid data	Valid data
D15 to D8		High-Z[*2]	Invalid data	Valid data	Valid data
D31 to D16		High-Z[*2]	High-Z[*2]	High-Z[*2]	High-Z[*2]

Table A.5 Pin States (Burst ROM/Little Endian) (cont)

Pin		32-Bit Bus Width						
		Byte Access (Address 4n)	Byte Access (Address 4n + 1)	Byte Access (Address 4n + 2)	Byte Access (Address 4n + 3)	Word Access (Address 4n)	Word Access (Address 4n + 2)	Longword Access
CS6 to CS2, CS0		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
RD	R	Low	Low	Low	Low	Low	Low	Low
	W	—	—	—	—	—	—	—
RD/WR	R	High	High	High	High	High	High	High
	W	—	—	—	—	—	—	—
BS		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
RAS3U/PTE[2]		High	High	High	High	High	High	High
RAS3L/PTJ[0]		High	High	High	High	High	High	High
CASLL/CAS/PTJ[2]		High	High	High	High	High	High	High
CASLH/PTJ[3]		High	High	High	High	High	High	High
CASHL/PTJ[4]		High	High	High	High	High	High	High
CASHH/PTJ[5]		High	High	High	High	High	High	High
CAS2L/PTE[6]		High	High	High	High	High	High	High
CAS2H/PTE[3]		High	High	High	High	High	High	High
WE0/DQMLL	R	High	High	High	High	High	High	High
	W	—	—	—	—	—	—	—
WE1/WE/DQMLU	R	High	High	High	High	High	High	High
	W	—	—	—	—	—	—	—
WE2/ICIORD/DQMUL/ PTK[6]	R	High	High	High	High	High	High	High
	W	—	—	—	—	—	—	—
WE3/ICIOWR/DQMUU/ PTK[7]	R	High	High	High	High	High	High	High
	W	—	—	—	—	—	—	—
CE2A/PTE[4]		High	High	High	High	High	High	High
CE2B/PTE[5]		High	High	High	High	High	High	High
RAS2U/PTE[1]		High	High	High	High	High	High	High
RAS2L/PTJ[1]		High	High	High	High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
WAIT		Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]
IOIS16		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address	Address	Address	Address

Table A.5 Pin States (Burst ROM/Little Endian) (cont)

	32-Bit Bus Width						
Pin	**Byte Access (Address 4n)**	**Byte Access (Address 4n + 1)**	**Byte Access (Address 4n + 2)**	**Byte Access (Address 4n + 3)**	**Word Access (Address 4n)**	**Word Access (Address 4n + 2)**	**Longword Access**
D7 to D0	Valid data	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D15 to D8	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D23 to D16	Invalid data	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Valid data
D31 to D24	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data	Valid data

Notes: 1. Disabled when WCR2 register wait setting is 0.

2. Unused data pins should be switched to the port function, or pulled up or down.

Table A.6 Pin States (Burst ROM/Big Endian)

		8-Bit Bus Width	16-Bit Bus Width		
Pin		Byte/Word/Longword Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/Longword Access
$\overline{\text{CS6}}$ to $\overline{\text{CS2}}$, $\overline{\text{CS0}}$		Enabled	Enabled	Enabled	Enabled
$\overline{\text{RD}}$	R	Low	Low	Low	Low
	W	—	—	—	—
RD/$\overline{\text{WR}}$	R	High	High	High	High
	W	—	—	—	—
$\overline{\text{BS}}$		Enabled	Enabled	Enabled	Enabled
$\overline{\text{RAS3U}}$/PTE[2]		High	High	High	High
$\overline{\text{RAS3L}}$/PTJ[0]		High	High	High	High
$\overline{\text{CASLL}}$/$\overline{\text{CAS}}$/PTJ[2]		High	High	High	High
$\overline{\text{CASLH}}$/PTJ[3]		High	High	High	High
$\overline{\text{CASHL}}$/PTJ[4]		High	High	High	High
$\overline{\text{CASHH}}$/PTJ[5]		High	High	High	High
$\overline{\text{CAS2L}}$/PTE[6]		High	High	High	High
$\overline{\text{CAS2H}}$/PTE[3]		High	High	High	High
$\overline{\text{WE0}}$/DQMLL	R	High	High	High	High
	W	—	—	—	—
$\overline{\text{WE1}}$/$\overline{\text{WE}}$/DQMLU	R	High	High	High	High
	W	—	—	—	—
$\overline{\text{WE2}}$/$\overline{\text{ICIORD}}$/DQMUL/ PTK[6]	R	High	High	High	High
	W	—	—	—	—
$\overline{\text{WE3}}$/$\overline{\text{ICIOWR}}$/DQMUU/ PTK[7]	R	High	High	High	High
	W	—	—	—	—
$\overline{\text{CE2A}}$/PTE[4]		High	High	High	High
$\overline{\text{CE2B}}$/PTE[5]		High	High	High	High
$\overline{\text{RAS2U}}$/PTE[1]		High	High	High	High
$\overline{\text{RAS2L}}$/PTJ[1]		High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled
$\overline{\text{WAIT}}$		Enabled*1	Enabled*1	Enabled*1	Enabled*1
$\overline{\text{IOIS16}}$		Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address
D7 to D0		Valid data	Invalid data	Valid data	Valid data
D15 to D8		High-Z*2	Valid data	Invalid data	Valid data
D31 to D16		High-Z*2	High-Z*2	High-Z*2	High-Z*2

Table A.6 Pin States (Burst ROM/Big Endian) (cont)

		32-Bit Bus Width						
Pin		Byte Access (Address 4n)	Byte Access (Address 4n + 1)	Byte Access (Address 4n + 2)	Byte Access (Address 4n + 3)	Word Access (Address 4n)	Word Access (Address 4n + 2)	Longword Access
$\overline{\text{CS6}}$ to $\overline{\text{CS2}}$, $\overline{\text{CS0}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RD}}$	R	Low	Low	Low	Low	Low	Low	Low
	W	—	—	—	—	—	—	—
RD/$\overline{\text{WR}}$	R	High	High	High	High	High	High	High
	W	—	—	—	—	—	—	—
$\overline{\text{BS}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RAS3U}}$/PTE[2]		High	High	High	High	High	High	High
$\overline{\text{RAS3L}}$/PTJ[0]		High	High	High	High	High	High	High
$\overline{\text{CASLL}}$/$\overline{\text{CAS}}$/PTJ[2]		High	High	High	High	High	High	High
$\overline{\text{CASLH}}$/PTJ[3]		High	High	High	High	High	High	High
$\overline{\text{CASHL}}$/PTJ[4]		High	High	High	High	High	High	High
$\overline{\text{CASHH}}$/PTJ[5]		High	High	High	High	High	High	High
$\overline{\text{CAS2L}}$/PTE[6]		High	High	High	High	High	High	High
$\overline{\text{CAS2H}}$/PTE[3]		High	High	High	High	High	High	High
$\overline{\text{WE0}}$/DQMLL	R	High	High	High	High	High	High	High
	W	—	—	—	—	—	—	—
$\overline{\text{WE1}}$/$\overline{\text{WE}}$/DQMLU	R	High	High	High	High	High	High	High
	W	—	—	—	—	—	—	—
$\overline{\text{WE2}}$/$\overline{\text{ICIORD}}$/DQMUL/ PTK[6]	R	High	High	High	High	High	High	High
	W	—	—	—	—	—	—	—
$\overline{\text{WE3}}$/$\overline{\text{ICIOWR}}$/DQMUU/ PTK[7]	R	High	High	High	High	High	High	High
	W	—	—	—	—	—	—	—
$\overline{\text{CE2A}}$/PTE[4]		High	High	High	High	High	High	High
$\overline{\text{CE2B}}$/PTE[5]		High	High	High	High	High	High	High
$\overline{\text{RAS2U}}$/PTE[1]		High	High	High	High	High	High	High
$\overline{\text{RAS2L}}$/PTJ[1]		High	High	High	High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{\text{WAIT}}$		Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]	Enabled[*1]
$\overline{\text{IOIS16}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address	Address	Address	Address

Table A.6 Pin States (Burst ROM/Big Endian) (cont)

	32-Bit Bus Width						
Pin	Byte Access (Address 4n)	Byte Access (Address 4n + 1)	Byte Access (Address 4n + 2)	Byte Access (Address 4n + 3)	Word Access (Address 4n)	Word Access (Address 4n + 2)	Longword Access
D7 to D0	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data	Valid data
D15 to D8	Invalid data	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Valid data
D23 to D16	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D31 to D24	Valid data	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data

Notes: 1. Disabled when WCR2 register wait setting is 0.

2. Unused data pins should be switched to the port function, or pulled up or down.

Table A.7 Pin States (DRAM/Little Endian)

		16-Bit Bus Width (Area 3)			16-Bit Bus Width (Area 2)		
Pin		Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/ Longword Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/ Longword Access
CS6 to CS2, CS0		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
RD	R	High	High	High	High	High	High
	W	High	High	High	High	High	High
RD/WR	R	High	High	High	High	High	High
	W	Low	Low	Low	Low	Low	Low
BS		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
RAS3U/PTE[2]		High/Low^{*1}	High/Low^{*1}	High/Low^{*1}	High	High	High
RAS3L/PTJ[0]		Low/High^{*1}	Low/High^{*1}	Low/High^{*1}	High	High	High
CASLL/CAS/PTJ[2]		Low	High	Low	High	High	High
CASLH/PTJ[3]		High	Low	Low	High	High	High
CASHL/PTJ[4]		High	High	High	High	High	High
CASHH/PTJ[5]		High	High	High	High	High	High
CAS2L/PTE[6]		High	High	High	Low	High	Low
CAS2H/PTE[3]		High	High	High	High	Low	Low
WE0/DQMLL	R	High	High	High	High	High	High
	W	High	High	High	High	High	High
WE1/WE/DQMLU	R	High	High	High	High	High	High
	W	High	High	High	High	High	High
WE2/ICIORD/DQMUL/ PTK[6]	R	High	High	High	High	High	High
	W	High	High	High	High	High	High
WE3/ICIOWR/DQMUU/ PTK[7]	R	High	High	High	High	High	High
	W	High	High	High	High	High	High
CE2A/PTE[4]		High	High	High	High	High	High
CE2B/PTE[5]		High	High	High	High	High	High
RAS2U/PTE[1]		High	High	High	High/Low^{*1}	High/Low^{*1}	High/Low^{*1}
RAS2L/PTJ[1]		High	High	High	Low/High^{*1}	Low/High^{*1}	Low/High^{*1}
CKE		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
WAIT		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
IOIS16		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address	Address	Address
D7 to D0		Valid data	Invalid data	Valid data	Valid data	Invalid data	Valid data
D15 to D8		Invalid data	Valid data	Valid data	Invalid data	Valid data	Valid data
D31 to D16		High-Z^{*2}	High-Z^{*2}	High-Z^{*2}	High-Z^{*2}	High-Z^{*2}	High-Z^{*2}

Table A.7 Pin States (DRAM/Little Endian) (cont)

		32-Bit Bus Width						
Pin		Byte Access (Address 4n)	Byte Access (Address 4n + 1)	Byte Access (Address 4n + 2)	Byte Access (Address 4n + 3)	Word Access (Address 4n)	Word Access (Address 4n + 2)	Longword Access
$\overline{\text{CS6}}$ to $\overline{\text{CS2}}$, $\overline{\text{CS0}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RD}}$	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
RD/$\overline{\text{WR}}$	R	High	High	High	High	High	High	High
	W	Low	Low	Low	Low	Low	Low	Low
$\overline{\text{BS}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RAS3U}}$/PTE[2]		High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]
$\overline{\text{RAS3L}}$/PTJ[0]		Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]
$\overline{\text{CASLL}}$/$\overline{\text{CAS}}$/PTJ[2]		Low	High	High	High	Low	High	Low
$\overline{\text{CASLH}}$/PTJ[3]		High	Low	High	High	Low	High	Low
$\overline{\text{CASHL}}$/PTJ[4]		High	High	Low	High	High	Low	Low
$\overline{\text{CASHH}}$/PTJ[5]		High	High	High	Low	High	Low	Low
$\overline{\text{CAS2L}}$/PTE[6]		High	High	High	High	High	High	High
$\overline{\text{CAS2H}}$/PTE[3]		High	High	High	High	High	High	High
$\overline{\text{WE0}}$/DQMLL	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
$\overline{\text{WE1}}$/$\overline{\text{WE}}$/DQMLU	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
$\overline{\text{WE2}}$/$\overline{\text{ICIORD}}$/DQMUL/ PTK[6]	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
$\overline{\text{WE3}}$/$\overline{\text{ICIOWR}}$/DQMUU/ PTK[7]	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
$\overline{\text{CE2A}}$/PTE[4]		High	High	High	High	High	High	High
$\overline{\text{CE2B}}$/PTE[5]		High	High	High	High	High	High	High
$\overline{\text{RAS2U}}$/PTE[1]		High	High	High	High	High	High	High
$\overline{\text{RAS2L}}$/PTJ[1]		High	High	High	High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{\text{WAIT}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{\text{IOIS16}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address	Address	Address	Address
D7 to D0		Valid data	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data

Table A.7 Pin States (DRAM/Little Endian) (cont)

	32-Bit Bus Width						
Pin	**Byte Access (Address 4n)**	**Byte Access (Address 4n + 1)**	**Byte Access (Address 4n + 2)**	**Byte Access (Address 4n + 3)**	**Word Access (Address 4n)**	**Word Access (Address 4n + 2)**	**Longword Access**
D15 to D8	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D23 to D16	Invalid data	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Valid data
D31 to D24	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data	Valid data

Note: 1. Lower 32-MB access / Upper 32-MB access.

2. Unused data pins should be switched to the port function, or pulled up or down.

Table A.8 Pin States (DRAM/Big Endian)

Pin		16-Bit Bus Width (Area 3)			16-Bit Bus Width (Area 2)		
		Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/ Longword Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/ Longword Access
CS6 to CS2, CS0		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
RD	R	High	High	High	High	High	High
	W	High	High	High	High	High	High
RD/WR	R	High	High	High	High	High	High
	W	Low	Low	Low	Low	Low	Low
BS		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
RAS3U/PTE[2]		High/Low*1	High/Low*1	High/Low*1	High	High	High
RAS3L/PTJ[0]		Low/High*1	Low/High*1	Low/High*1	High	High	High
CASLL/CAS/PTJ[2]		High	Low	Low	High	High	High
CASLH/PTJ[3]		Low	High	Low	High	High	High
CASHL/PTJ[4]		High	High	High	High	High	High
CASHH/PTJ[5]		High	High	High	High	High	High
CAS2L/PTE[6]		High	High	High	High	Low	Low
CAS2H/PTE[3]		High	High	High	Low	High	Low
WE0/DQMLL	R	High	High	High	High	High	High
	W	High	High	High	High	High	High
WE1/WE/DQMLU	R	High	High	High	High	High	High
	W	High	High	High	High	High	High
WE2/ICIORD/DQMUL/ PTK[6]	R	High	High	High	High	High	High
	W	High	High	High	High	High	High
WE3/ICIOWR/DQMUU/ PTK[7]	R	High	High	High	High	High	High
	W	High	High	High	High	High	High
CE2A/PTE[4]		High	High	High	High	High	High
CE2B/PTE[5]		High	High	High	High	High	High
RAS2U/PTE[1]		High	High	High	High/Low*1	High/Low*1	High/Low*1
RAS2L/PTJ[1]		High	High	High	Low/High*1	Low/High*1	Low/High*1
CKE		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
WAIT		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
IOIS16		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address	Address	Address
D7 to D0		Invalid data	Valid data	Valid data	Invalid data	Valid data	Valid data
D15 to D8		Valid data	Invalid data	Valid data	Valid data	Invalid data	Valid data
D31 to D16		High-Z*2	High-Z*2	High-Z*2	High-Z*2	High-Z*2	High-Z*2

Table A.8 Pin States (DRAM/Big Endian) (cont)

		32-Bit Bus Width						
Pin		Byte Access (Address 4n)	Byte Access (Address 4n + 1)	Byte Access (Address 4n + 2)	Byte Access (Address 4n + 3)	Word Access (Address 4n)	Word Access (Address 4n + 2)	Longword Access
$\overline{CS6}$ to $\overline{CS2}$, $\overline{CS0}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{RD}$	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
RD/$\overline{WR}$	R	High	High	High	High	High	High	High
	W	Low	Low	Low	Low	Low	Low	Low
$\overline{BS}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{RAS3U}$/PTE[2]		High/Low^{*1}	High/Low^{*1}	High/Low^{*1}	High/Low^{*1}	High/Low^{*1}	High/Low^{*1}	High/Low^{*1}
$\overline{RAS3L}$/PTJ[0]		Low/High^{*1}	Low/High^{*1}	Low/High^{*1}	Low/High^{*1}	Low/High^{*1}	Low/High^{*1}	Low/High^{*1}
$\overline{CASLL}$/$\overline{CAS}$/PTJ[2]		High	High	High	Low	High	Low	Low
$\overline{CASLH}$/PTJ[3]		High	High	Low	High	High	Low	Low
$\overline{CASHL}$/PTJ[4]		High	Low	High	High	Low	High	Low
$\overline{CASHH}$/PTJ[5]		Low	High	High	High	Low	High	Low
$\overline{CAS2L}$/PTE[6]		High	High	High	High	High	High	High
$\overline{CAS2H}$/PTE[3]		High	High	High	High	High	High	High
$\overline{WE0}$/DQMLL	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
$\overline{WE1}$/$\overline{WE}$/DQMLU	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
$\overline{WE2}$/$\overline{ICIORD}$/DQMUL/PTK[6]	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
$\overline{WE3}$/$\overline{ICIOWR}$/DQMUU/PTK[7]	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
$\overline{CE2A}$/PTE[4]		High	High	High	High	High	High	High
$\overline{CE2B}$/PTE[5]		High	High	High	High	High	High	High
$\overline{RAS2U}$/PTE[1]		High	High	High	High	High	High	High
$\overline{RAS2L}$/PTJ[1]		High	High	High	High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{WAIT}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{IOIS16}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address	Address	Address	Address
D7 to D0		Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data	Valid data

Table A.8 Pin States (DRAM/Big Endian) (cont)

	32-Bit Bus Width						
Pin	**Byte Access (Address 4n)**	**Byte Access (Address 4n + 1)**	**Byte Access (Address 4n + 2)**	**Byte Access (Address 4n + 3)**	**Word Access (Address 4n)**	**Word Access (Address 4n + 2)**	**Longword Access**
D15 to D8	Invalid data	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Valid data
D23 to D16	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D31 to D24	Valid data	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data

Note: 1. Lower 32-MB access / Upper 32-MB access.

2. Unused data pins should be switched to the port function, or pulled up or down.

Table A.9 Pin States (Synchronous DRAM/Little Endian)

		32-Bit Bus Width						
Pin		**Byte Access (Address 4n)**	**Byte Access (Address 4n + 1)**	**Byte Access (Address 4n + 2)**	**Byte Access (Address 4n + 3)**	**Word Access (Address 4n)**	**Word Access (Address 4n + 2)**	**Longword Access**
$\overline{\text{CS6}}$ to $\overline{\text{CS2}}$, $\overline{\text{CS0}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RD}}$	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
RD/$\overline{\text{WR}}$	R	High	High	High	High	High	High	High
	W	Low	Low	Low	Low	Low	Low	Low
$\overline{\text{BS}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RAS3U}}$/PTE[2]		High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]
$\overline{\text{RAS3L}}$/PTJ[0]		Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]
$\overline{\text{CAS}}$/$\overline{\text{CASLL}}$/PTJ[2]		Low	Low	Low	Low	Low	Low	Low
$\overline{\text{CASLH}}$/PTJ[3]		High	High	High	High	High	High	High
$\overline{\text{CASHL}}$/PTJ[4]		High	High	High	High	High	High	High
$\overline{\text{CASHH}}$/PTJ[5]		High	High	High	High	High	High	High
$\overline{\text{CASHL}}$/PTE[6]		High	High	High	High	High	High	High
$\overline{\text{CASHH}}$/PTE[3]		High	High	High	High	High	High	High
DQMLL/$\overline{\text{WE0}}$	R	Low	High	High	High	Low	High	Low
	W	Low	High	High	High	Low	High	Low
DQMLU/$\overline{\text{WE1}}$	R	High	Low	High	High	Low	High	Low
	W	High	Low	High	High	Low	High	Low
DQMUL/$\overline{\text{WE2}}$/$\overline{\text{ICIORD}}$	R	High	High	Low	High	High	Low	Low
	W	High	High	Low	High	High	Low	Low
DQMUU/$\overline{\text{WE3}}$/$\overline{\text{ICIOWR}}$	R	High	High	High	Low	High	Low	Low
	W	High	High	High	Low	High	Low	Low
$\overline{\text{CE2A}}$/PTE[4]		High	High	High	High	High	High	High
$\overline{\text{CE2B}}$/PTE[5]		High	High	High	High	High	High	High
$\overline{\text{RAS2U}}$/PTE[1]		High	High	High	High	High	High	High
$\overline{\text{RAS2L}}$/PTJ[1]		High	High	High	High	High	High	High
CKE		High[*2]	High[*2]	High[*2]	High[*2]	High[*2]	High[*2]	High[*2]
$\overline{\text{WAIT}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{\text{IOIS16}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
A25 to A0		Address command	Address command	Address command	Address command	Address command	Address command	Address command

Table A.9 Pin States (Synchronous DRAM/Little Endian) (cont)

	32-Bit Bus Width						
Pin	Byte Access (Address 4n)	Byte Access (Address 4n + 1)	Byte Access (Address 4n + 2)	Byte Access (Address 4n + 3)	Word Access (Address 4n)	Word Access (Address 4n + 2)	Longword Access
D7 to D0	Valid data	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D15 to D8	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D23 to D16	Invalid data	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Valid data
D31 to D24	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data	Valid data

Notes: 1. Lower 32-MB access/ Upper 32-MB access

2. Normally high. Low in self-refreshing.

Table A.10 Pin States (Synchronous DRAM/Big Endian)

		32-Bit Bus Width						
Pin		**Byte Access (Address 4n)**	**Byte Access (Address 4n + 1)**	**Byte Access (Address 4n + 2)**	**Byte Access (Address 4n + 3)**	**Word Access (Address 4n)**	**Word Access (Address 4n + 2)**	**Longword Access**
$\overline{\text{CS6}}$ to $\overline{\text{CS2}}$, $\overline{\text{CS0}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RD}}$	R	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High
RD/$\overline{\text{WR}}$	R	High	High	High	High	High	High	High
	W	Low	Low	Low	Low	Low	Low	Low
$\overline{\text{BS}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RAS3U}}$/PTE[2]		High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]	High/Low[*1]
$\overline{\text{RAS3L}}$/PTJ[0]		Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]	Low/High[*1]
$\overline{\text{CAS}}$/$\overline{\text{CASLL}}$/PTJ[2]		Low	Low	Low	Low	Low	Low	Low
$\overline{\text{CASLH}}$/PTJ[3]		High	High	High	High	High	High	High
$\overline{\text{CASHL}}$/PTJ[4]		High	High	High	High	High	High	High
$\overline{\text{CASHH}}$/PTJ[5]		High	High	High	High	High	High	High
$\overline{\text{CASHL}}$/PTE[6]		High	High	High	High	High	High	High
$\overline{\text{CASHH}}$/PTE[3]		High	High	High	High	High	High	High
DQMLL/$\overline{\text{WE0}}$	R	High	High	High	Low	High	Low	Low
	W	High	High	High	Low	High	Low	Low
DQMLU/$\overline{\text{WE1}}$	R	High	High	Low	High	High	Low	Low
	W	High	High	Low	High	High	Low	Low
DQMUL/$\overline{\text{WE2}}$/$\overline{\text{ICIORD}}$	R	High	Low	High	High	Low	High	Low
	W	High	Low	High	High	Low	High	Low
DQMUU/$\overline{\text{WE3}}$/$\overline{\text{ICIOWR}}$	R	Low	High	High	High	Low	High	Low
	W	Low	High	High	High	Low	High	Low
$\overline{\text{CE2A}}$/PTE[4]		High	High	High	High	High	High	High
$\overline{\text{CE2B}}$/PTE[5]		High	High	High	High	High	High	High
$\overline{\text{RAS2U}}$/PTE[1]		High	High	High	High	High	High	High
$\overline{\text{RAS2L}}$/PTJ[1]		High	High	High	High	High	High	High
CKE		High[*2]	High[*2]	High[*2]	High[*2]	High[*2]	High[*2]	High[*2]
$\overline{\text{WAIT}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{\text{IOIS16}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
A25 to A0		Address command	Address command	Address command	Address command	Address command	Address command	Address command

Table A.10 Pin States (Synchronous DRAM/Big Endian) (cont)

	32-Bit Bus Width						
Pin	Byte Access (Address 4n)	Byte Access (Address 4n + 1)	Byte Access (Address 4n + 2)	Byte Access (Address 4n + 3)	Word Access (Address 4n)	Word Access (Address 4n + 2)	Longword Access
D7 to D0	Valid data	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D15 to D8	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data
D23 to D16	Invalid data	Invalid data	Valid data	Invalid data	Invalid data	Valid data	Valid data
D31 to D24	Invalid data	Invalid data	Invalid data	Valid data	Invalid data	Valid data	Valid data

Notes: 1. Lower 32-MB access/ Upper 32-MB access

2. Normally high. Low in self-refreshing.

Table A.11 Pin States (PCMCIA/Little Endian)

		PCMCIA Memory Interface (Area 5)			
		8-Bit Bus Width	16-Bit Bus Width		
Pin		Byte/Word/Longword Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/Longword Access
CS6 to CS2, CS0		Enabled	Enabled	High	Enabled
RD	R	Low	Low	Low	Low
	W	High	High	High	High
RD/WR	R	High	High	High	High
	W	Low	Low	Low	Low
BS		Enabled	Enabled	Enabled	Enabled
RAS3U/PTE[2]		High	High	High	High
RAS3L/PTJ[0]		High	High	High	High
CASLL/CAS/PTJ[2]		High	High	High	High
CASLH/PTJ[3]		High	High	High	High
CASHL/PTJ[4]		High	High	High	High
CASHH/PTJ[5]		High	High	High	High
CAS2L/PTE[6]		High	High	High	High
CAS2H/PTE[3]		High	High	High	High
WE0/DQMLL	R	High	High	High	High
	W	High	High	High	High
WE1/WE/DQMLU	R	High	High	High	High
	W	Low	Low	Low	Low
WE2/ICIORD/DQMUL/PTK[6]	R	High	High	High	High
	W	High	High	High	High
WE3/ICIOWR/DQMUU/PTK[7]	R	High	High	High	High
	W	High	High	High	High
CE2A/PTE[4]		High	High	Low	Low
CE2B/PTE[5]		High	High	High	High
RAS2U/PTE[1]		High	High	High	High
RAS2L/PTJ[1]		High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled
WAIT		Disabled	Disabled	Disabled	Disabled
IOIS16		Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address
D7 to D0		Valid data	Valid data	Invalid data	Valid data
D15 to D8		High-Z[*2]	Invalid data	Valid data	Valid data
D31 to D16		High-Z[*2]	High-Z[*2]	High-Z[*2]	High-Z[*2]

Table A.11 Pin States (PCMCIA/Little Endian) (cont)

		PCMCIA Memory Interface (Area 6)				PCMCIA/IO Interface (Area 6)			
		8-Bit Bus Width	16-Bit Bus Width			8-Bit Bus Width	16-Bit Bus Width		
Pin		Byte/ Word/ Long-word Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/ Long-word Access	Byte/ Word/ Long-word Access	Byte Access (Address 2n)	Byte Access (Address 2n+1)	Word/ Long-word Access
$\overline{\text{CS6}}$ to $\overline{\text{CS2}}$, $\overline{\text{CS0}}$		Enabled	Enabled	High	Enabled	Enabled	Enabled	High	Enabled
$\overline{\text{RD}}$	R	Low	Low	Low	Low	High	High	High	High
	W	High	High	High	High	High	High	High	High
RD/$\overline{\text{WR}}$	R	High	High	High	High	High	High	High	High
	W	Low	Low	Low	Low	Low	Low	Low	Low
$\overline{\text{BS}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\text{RAS3U}}$/PTE[2]		High	High	High	High	High	High	High	High
$\overline{\text{RAS3L}}$/PTJ[0]		High	High	High	High	High	High	High	High
$\overline{\text{CASLL}}$/$\overline{\text{CAS}}$/PTJ[2]		High	High	High	High	High	High	High	High
$\overline{\text{CASLH}}$/PTJ[3]		High	High	High	High	High	High	High	High
$\overline{\text{CASHL}}$/PTJ[4]		High	High	High	High	High	High	High	High
$\overline{\text{CASHH}}$/PTJ[5]		High	High	High	High	High	High	High	High
$\overline{\text{CAS2L}}$/PTE[6]		High	High	High	High	High	High	High	High
$\overline{\text{CAS2H}}$/PTE[3]		High	High	High	High	High	High	High	High
$\overline{\text{WE0}}$/DQMLL	R	High	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High	High
$\overline{\text{WE1}}$/$\overline{\text{WE}}$/DQMLU	R	High	High	High	High	High	High	High	High
	W	Low	Low	Low	Low	High	High	High	High
$\overline{\text{WE2}}$/$\overline{\text{ICIORD}}$/DQMUL/ PTK[6]	R	High	High	High	High	Low	Low	Low	Low
	W	High	High	High	High	High	High	High	High
$\overline{\text{WE3}}$/$\overline{\text{ICIOWR}}$/DQMUU/ PTK[7]	R	High	High	High	High	High	High	High	High
	W	High	High	High	High	Low	Low	Low	Low
$\overline{\text{CE2A}}$/PTE[4]		High	High	High	High	High	High	High	High
$\overline{\text{CE2B}}$/PTE[5]		High	High	Low	Low	High	High	Low	Low
$\overline{\text{RAS2U}}$/PTE[1]		High	High	High	High	High	High	High	High
$\overline{\text{RAS2}}$/PTJ[1]		High	High	High	High	High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{\text{WAIT}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{\text{IOIS16}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Enabled	Enabled
A25 to A0		Address	Address	Address	Address	Address	Address	Address	Address

Table A.11 Pin States (PCMCIA/Little Endian) (cont)

	PCMCIA Memory Interface (Area 6)				PCMCIA/IO Interface (Area 6)			
	8-Bit Bus Width	16-Bit Bus Width			8-Bit Bus Width	16-Bit Bus Width		
Pin	Byte/ Word/ Long-word Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/ Long-word Access	Byte/ Word/ Long-word Access	Byte Access (Address 2n)	Byte Access (Address 2n+1)	Word/ Long-word Access
D7 to D0	Valid data	Valid data	Invalid data	Valid data	Valid data	Valid data	Invalid data	Valid data
D15 to D8	High-Z*2	Invalid data	Valid data	Valid data	High-Z*2	Invalid data	Valid data	Valid data
D31 to D16	High-Z*2	High-Z*2	High-Z*2	High-Z*2	High-Z*2	High-Z*2	High-Z*2	High-Z*2

Notes: 1. Disabled when WCR2 register wait setting is 0.

2. Unused data pins should be switched to the port function, or pulled up or down.

Table A.12 Pin States (PCMCIA/Big Endian)

		PCMCIA Memory Interface (Area 5)			
		8-Bit Bus Width	16-Bit Bus Width		
Pin		Byte/Word/Longword Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/Longword Access
$\overline{CS6}$ to $\overline{CS2}$, $\overline{CS0}$		Enabled	Enabled	High	Enabled
$\overline{RD}$	R	Low	Low	Low	Low
	W	High	High	High	High
RD/$\overline{WR}$	R	High	High	High	High
	W	Low	Low	Low	Low
$\overline{BS}$		Enabled	Enabled	Enabled	Enabled
$\overline{RAS3U}$/PTE[2]		High	High	High	High
$\overline{RAS3L}$/PTJ[0]		High	High	High	High
$\overline{CASLL}$/$\overline{CAS}$/PTJ[2]		High	High	High	High
$\overline{CASLH}$/PTJ[3]		High	High	High	High
$\overline{CASHL}$/PTJ[4]		High	High	High	High
$\overline{CASHH}$/PTJ[5]		High	High	High	High
$\overline{CAS2L}$/PTE[6]		High	High	High	High
$\overline{CAS2H}$/PTE[3]		High	High	High	High
$\overline{WE0}$/DQMLL	R	High	High	High	High
	W	High	High	High	High
$\overline{WE1}$/$\overline{WE}$/DQMLU	R	High	High	High	High
	W	Low	Low	Low	Low
$\overline{WE2}$/$\overline{ICIORD}$/DQMUL/PTK[6]	R	High	High	High	High
	W	High	High	High	High
$\overline{WE3}$/$\overline{ICIOWR}$/DQMUU/PTK[7]	R	High	High	High	High
	W	High	High	Low	Low
$\overline{CE2A}$/PTE[4]		High	High	Low	Low
$\overline{CE2B}$/PTE[5]		High	High	High	High
$\overline{RAS2U}$/PTE[1]		High	High	High	High
$\overline{RAS2}$/PTJ[1]		High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled
$\overline{WAIT}$		Disabled	Disabled	Disabled	Disabled
$\overline{IOIS16}$		Disabled	Disabled	Disabled	Disabled
A25 to A0		Address	Address	Address	Address
D7 to D0		Valid data	Valid data	Invalid data	Valid data
D15 to D8		High-Z[*2]	Invalid data	Valid data	Valid data
D31 to D16		High-Z[*2]	High-Z[*2]	High-Z[*2]	High-Z[*2]

Table A.12 Pin States (PCMCIA/Big Endian) (cont)

		PCMCIA Memory Interface (Area 6)				PCMCIA/IO Interface (Area 6)			
		8-Bit Bus Width	16-Bit Bus Width			8-Bit Bus Width	16-Bit Bus Width		
Pin		Byte/ Word/ Longword Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/ Longword Access	Byte/ Word/ Longword Access	Byte Access (Address 2n)	Byte Access (Address 2n+1)	Word/ Longword Access
$\overline{\mathrm{CS6}}$ to $\overline{\mathrm{CS2}}$, $\overline{\mathrm{CS0}}$		Enabled	Enabled	High	Enabled	Enabled	Enabled	High	Enabled
$\overline{\mathrm{RD}}$	R	Low	Low	Low	Low	High	High	High	High
	W	High	High	High	High	High	High	High	High
RD/$\overline{\mathrm{WR}}$	R	High	High	High	High	High	High	High	High
	W	Low	Low	Low	Low	Low	Low	Low	Low
$\overline{\mathrm{BS}}$		Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled	Enabled
$\overline{\mathrm{RAS3U}}$/PTE[2]		High	High	High	High	High	High	High	High
$\overline{\mathrm{RAS3L}}$/PTJ[0]		High	High	High	High	High	High	High	High
$\overline{\mathrm{CASLL}}$/$\overline{\mathrm{CAS}}$/PTJ[2]		High	High	High	High	High	High	High	High
$\overline{\mathrm{CASLH}}$/PTJ[3]		High	High	High	High	High	High	High	High
$\overline{\mathrm{CASHL}}$/PTJ[4]		High	High	High	High	High	High	High	High
$\overline{\mathrm{CASHH}}$/PTJ[5]		High	High	High	High	High	High	High	High
$\overline{\mathrm{CAS2L}}$/PTE[6]		High	High	High	High	High	High	High	High
$\overline{\mathrm{CAS2H}}$/PTE[3]		High	High	High	High	High	High	High	High
$\overline{\mathrm{WE0}}$/DQMLL	R	High	High	High	High	High	High	High	High
	W	High	High	High	High	High	High	High	High
$\overline{\mathrm{WE1}}$/$\overline{\mathrm{WE}}$/DQMLU	R	High	High	High	High	High	High	High	High
	W	Low	Low	Low	Low	High	High	High	High
$\overline{\mathrm{WE2}}$/$\overline{\mathrm{ICIORD}}$/DQMUL/ PTK[6]	R	High	High	High	High	Low	Low	Low	Low
	W	High	High	High	High	High	High	High	High
$\overline{\mathrm{WE3}}$/$\overline{\mathrm{ICIOWR}}$/DQMUU/ PTK[7]	R	High	High	High	High	High	High	High	High
	W	High	High	High	High	Low	Low	Low	Low
$\overline{\mathrm{CE2A}}$/PTE[4]		High	High	High	High	High	High	High	High
$\overline{\mathrm{CE2B}}$/PTE[5]		High	High	Low	Low	High	High	Low	Low
$\overline{\mathrm{RAS2U}}$/PTE[1]		High	High	High	High	High	High	High	High
$\overline{\mathrm{RAS2}}$/PTJ[1]		High	High	High	High	High	High	High	High
CKE		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{\mathrm{WAIT}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Disabled
$\overline{\mathrm{IOIS16}}$		Disabled	Disabled	Disabled	Disabled	Disabled	Disabled	Enabled	Enabled
A25 to A0		Address	Address	Address	Address	Address	Address	Address	Address

Table A.12 Pin States (PCMCIA/Big Endian) (cont)

	PCMCIA Memory Interface (Area 6)				PCMCIA/IO Interface (Area 6)			
	8-Bit Bus Width	16-Bit Bus Width			8-Bit Bus Width	16-Bit Bus Width		
Pin	Byte/ Word/ Longword Access	Byte Access (Address 2n)	Byte Access (Address 2n + 1)	Word/ Longword Access	Byte/ Word/ Longword Access	Byte Access (Address 2n)	Byte Access (Address 2n+1)	Word/ Longword Access
D7 to D0	Valid data	Valid data	Invalid data	Valid data	Valid data	Valid data	Invalid data	Valid data
D15 to D8	High-Z*2	Valid data	Invalid data	Valid data	High-Z*2	Invalid data	Invalid data	Valid data
D31 to D16	High-Z*2	High-Z*2	High-Z*2	High-Z*2	High-Z*2	High-Z*2	High-Z*2	High-Z*2

Notes: 1. Disabled when WCR2 register wait setting is 0.

2. Unused data pins should be switched to the port function, or pulled up or down.

Appendix B Memory-Mapped Control Registers

B.1 Register Address Map

Table B.1 Memory-Mapped Control Registers

Control Register	Module *1	Bus *2	Address *4	Size (Bits)	Access Size (Bits) *3
PTEH	CCN	L	FFFFFFF0	32	32
PTEL	CCN	L	FFFFFFF4	32	32
TTB	CCN	L	FFFFFFF8	32	32
TEA	CCN	L	FFFFFFFC	32	32
MMUCR	CCN	L	FFFFFFE0	32	32
BASRA	CCN	L	FFFFFFE4	32	32
BASRB	CCN	L	FFFFFFE8	32	32
CCR	CCN	L	FFFFFFEC	32	32
CCR2	CCN	I	40000B0	32	32
TRA	CCN	L	FFFFFFD0	32	32
EXPEVT	CCN	L	FFFFFFD4	32	32
INTEVT	CCN	L	FFFFFFD8	32	32
BARA	UBC	L	FFFFFFB0	32	32
BAMRA	UBC	L	FFFFFFB4	8	8
BBRA	UBC	L	FFFFFFB8	16	16
BARB	UBC	L	FFFFFFA0	32	32
BAMRB	UBC	L	FFFFFFA4	8	8
BBRB	UBC	L	FFFFFFA8	16	16
BDRB	UBC	L	FFFFFF90	32	32
BDMRB	UBC	L	FFFFFF94	32	32
BRCR	UBC	L	FFFFFF98	16	16
FRQCR	CPG	I	FFFFFF80	16	16
STBCR	CPG	I	FFFFFF82	8	8
STBCR2	CPG	I	FFFFFF88	8	8
WTCNT	CPG	I	FFFFFF84	8	16
WTCSR	CPG	I	FFFFFF86	8	16
BCR1	BSC	I	FFFFFF60	16	16
BCR2	BSC	I	FFFFFF62	16	16

Table B.1 Memory-Mapped Control Registers (cont)

Control Register	Module *1	Bus *2	Address *4	Size (Bits)	Access Size (Bits) *3
WCR1	BSC	I	FFFFFF64	16	16
WCR2	BSC	I	FFFFFF66	16	16
MCR	BSC	I	FFFFFF68	16	16
DCR	BSC	I	FFFFFF6A	16	16
PCR	BSC	I	FFFFFF6C	16	16
RTCSR	BSC	I	FFFFFF6E	16	16
RTCNT	BSC	I	FFFFFF70	16	16
RTCOR	BSC	I	FFFFFF72	16	16
RFCR	BSC	I	FFFFFF74	16	16
BCR3	BSC	I	FFFFFF7E	16	16
SDMR	BSC	I	FFFFD000 – FFFFEFFE	—	8
MCSCR0	BSC	I	FFFFFF50	16	16
MCSCR1	BSC	I	FFFFFF52	16	16
MCSCR2	BSC	I	FFFFFF54	16	16
MCSCR3	BSC	I	FFFFFF56	16	16
MCSCR4	BSC	I	FFFFFF58	16	16
MCSCR5	BSC	I	FFFFFF5A	16	16
MCSCR6	BSC	I	FFFFFF5C	16	16
MCSCR7	BSC	I	FFFFFF5E	16	16
R64CNT	RTC	P	FFFFFEC0	8	8
RSECCNT	RTC	P	FFFFFEC2	8	8
RMINCNT	RTC	P	FFFFFEC4	8	8
RHRCNT	RTC	P	FFFFFEC6	8	8
RWKCNT	RTC	P	FFFFFEC8	8	8
RDAYCNT	RTC	P	FFFFFECA	8	8
RMONCNT	RTC	P	FFFFFECC	8	8
RYRCNT	RTC	P	FFFFFECE	8	8
RSECAR	RTC	P	FFFFFED0	8	8
RMINAR	RTC	P	FFFFFED2	8	8
RHRAR	RTC	P	FFFFFED4	8	8
RWKAR	RTC	P	FFFFFED6	8	8

Table B.1 Memory-Mapped Control Registers (cont)

Control Register	Module *[1]	Bus *[2]	Address *[4]	Size (Bits)	Access Size (Bits) *[3]
RDAYAR	RTC	P	FFFFFED8	8	8
RMONAR	RTC	P	FFFFFEDA	8	8
RCR1	RTC	P	FFFFFEDC	8	8
RCR2	RTC	P	FFFFFEDE	8	8
ICR0	INTC	I	FFFFFEE0	16	16
IPRA	INTC	I	FFFFFEE2	16	16
IPRB	INTC	I	FFFFFEE4	16	16
TOCR	TMU	P	FFFFFE90	8	8
TSTR	TMU	P	FFFFFE92	8	8
TCOR0	TMU	P	FFFFFE94	32	32
TCNT0	TMU	P	FFFFFE98	32	32
TCR0	TMU	P	FFFFFE9C	16	16
TCOR1	TMU	P	FFFFFEA0	32	32
TCNT1	TMU	P	FFFFFEA4	32	32
TCR1	TMU	P	FFFFFEA8	16	16
TCOR2	TMU	P	FFFFFEAC	32	32
TCNT2	TMU	P	FFFFFEB0	32	32
TCR2	TMU	P	FFFFFEB4	16	16
TCPR2	TMU	P	FFFFFEB8	32	32
SCSMR	SCI	P	FFFFFE80	8	8
SCBRR	SCI	P	FFFFFE82	8	8
SCSCR	SCI	P	FFFFFE84	8	8
SCTDR	SCI	P	FFFFFE86	8	8
SCSSR	SCI	P	FFFFFE88	8	8
SCRDR	SCI	P	FFFFFE8A	8	8

Notes: 1. Module: CCN: Cache controller, UBC: User break controller, CPG: Clock pulse generator, BSC: Bus state controller, RTC: Realtime clock, INTC: Interrupt controller, TMU: Timer unit, SCI: Serial communication interface

2. Internal bus: L: Connects CPU, CCN, cache, TLB, and DSP, I: Connects BSC and cache, DMAC, INTC, CPG, and UDI, P: Connects BSC and peripheral modules RTC, TMU, SCI, SCIF, IRDA, A/D, D/A, DMAC, PORT, and CMT

3. Access size for the control registers. If sizes other than those listed are used, the register will not be correctly accessed.

4. When the control register in area 1 is not used for address translation by the MMU, set the top three bits of the logical address to 101 to allocate in the P2 space.

Table B.1 Memory-Mapped Control Registers (cont)

Control Register	Module *1	Bus *2	Address *4	Size (Bits)	Access Size (Bits) *3
INTEVT2	INTC	I	4000000	32	32
IRR0	INTC	I	4000004	16	8
IRR1	INTC	I	4000006	16	8
IRR2	INTC	I	4000008	16	8
ICR1	INTC	I	4000010	16	16
ICR2	INTC	I	4000012	16	16
INTER	INTC	I	4000014	16	16
IPRC	INTC	I	4000016	16	16
IPRD	INTC	I	4000018	16	16
IPRE	INTC	I	400001A	16	16
SAR0	DMAC	P	4000020	32	16,32
DAR0	DMAC	P	4000024	32	16,32
DMATCR0	DMAC	P	4000028	32	16,32
CHCR0	DMAC	P	400002C	32	8,16,32
SAR1	DMAC	P	4000030	32	16,32
DAR1	DMAC	P	4000034	32	16,32
DMATCR1	DMAC	P	4000038	32	16,32
CHCR1	DMAC	P	400003C	32	8,16,32
SAR2	DMAC	P	4000040	32	16,32
DAR2	DMAC	P	4000044	32	16,32
DMATCR2	DMAC	P	4000048	32	16,32
CHCR2	DMAC	P	400004C	32	8,16,32
SAR3	DMAC	P	4000050	32	16,32
DAR3	DMAC	P	4000054	32	16,32
DMATCR3	DMAC	P	4000058	32	16,32
CHCR3	DMAC	P	400005C	32	8,16,32
DMAOR	DMAC	P	4000060	16	8,16
CMSTR	CMT	P	4000070	16	8,16,32
CMCSR	CMT	P	4000072	16	8,16,32

Table B.1 Memory-Mapped Control Registers (cont)

Control Register	Module *1	Bus *2	Address *4	Size (Bits)	Access Size (Bits) *3
CMCNT	CMT	P	4000074	16	8,16,32
CMCOR	CMT	P	4000076	16	8,16,32
ADDRAH	A/D	P	4000080	8	8,16,32*5,*6
ADDRAL	A/D	P	4000082	8	8,16*5
ADDRBH	A/D	P	4000084	8	8,16,32*5,*6
ADDRBL	A/D	P	4000086	8	8,16*5
ADDRCH	A/D	P	4000088	8	8,16,32*5,*6
ADDRCL	A/D	P	400008A	8	8,16*5
ADDRDH	A/D	P	400008C	8	8,16,32*5,*6
ADDRDL	A/D	P	400008E	8	8,16*5
ADCSR	A/D	P	4000090	8	8,16,32*5,*6
ADCR	A/D	P	4000092	8	8,16
DADR0	D/A	P	40000A0	8	8,16,32*5,*6
DADR1	D/A	P	40000A2	8	8,16*5
DACR	D/A	P	40000A4	8	8,16,32
PACR	PORT	P	4000100	16	16
PBCR	PORT	P	4000102	16	16
PCCR	PORT	P	4000104	16	16
PDCR	PORT	P	4000106	16	16
PECR	PORT	P	4000108	16	16
PFCR	PORT	P	400010A	16	16
PGCR	PORT	P	400010C	16	16
PHCR	PORT	P	400010E	16	16
PJCR	PORT	P	4000110	16	16
PKCR	PORT	P	4000112	16	16
PLCR	PORT	P	4000114	16	16
SCPCR	PORT	P	4000116	16	16
PADR	PORT	P	4000120	8	8
PBDR	PORT	P	4000122	8	8
PCDR	PORT	P	4000124	8	8

Table B.1 Memory-Mapped Control Registers (cont)

Control Register	Module *1	Bus *2	Address *4	Size (Bits)	Access Size (Bits) *3
PDDR	PORT	P	4000126	8	8
PEDR	PORT	P	4000128	8	8
PFDR	PORT	P	400012A	8	8
PGDR	PORT	P	400012C	8	8
PHDR	PORT	P	400012E	8	8
PJDR	PORT	P	4000130	8	8
PKDR	PORT	P	4000132	8	8
PLDR	PORT	P	4000134	8	8
SCPDR	PORT	P	4000136	8	8
SCSMR1	IrDA	P	4000140	8	8
SCBRR1	IrDA	P	4000142	8	8
SCSCR1	IrDA	P	4000144	8	8
SCFTDR1	IrDA	P	4000146	8	8
SCSSR1	IrDA	P	4000148	16	16
SCFRDR1	IrDA	P	400014A	8	8
SCFCR1	IrDA	P	400014C	8	8
SCFDR1	IrDA	P	400014E	16	16
SCSMR2	SCIF	P	4000150	8	8
SCBRR2	SCIF	P	4000152	8	8
SCSCR2	SCIF	P	4000154	8	8
SCFTDR2	SCIF	P	4000156	8	8
SCSSR2	SCIF	P	4000158	16	16
SCFRDR2	SCIF	P	400015A	8	8
SCFCR2	SCIF	P	400015C	8	8
SCFDR2	SCIF	P	400015E	16	16
SDIR	UDI	I	4000200	16	16
SDSR	UDI	I	4000204	16	16
SDDR/SDDRH	UDI	I	4000208	16/32	16/32
SDDRL	UDI	I	400020A	16	16
SDAR	UDI	I	400020C	16	16
SDARE	UDI	I	4000210	16	16

Notes: 1. Modules:

CCN: Cache controller
CPG: Clock pulse generator
RTC: Realtime clock
TMU: Timer unit
UBC: User break controller
BSC: Bus state controller
INTC: Interrupt controller
SCI: Serial communication interface

2. Internal buses:

L: CPU, CCN, cache, TLB, and DSP connected
I: BSC, cache, DMAC, INTC, CPG, and UDI connected
P: BSC and peripheral modules (RTC, TMU, SCI, SCIF, IrDA, A/D, D/A, DMAC, ports, CMT) connected

3. The access size shown is for control register access (read/write). An incorrect result will be obtained if a different size from that shown is used for access.
4. To exclude area 1 control registers from address translation by the MMU, set the first 3 bits of the logical address to 101, to locate the registers in the P2 space.
5. With 16-bit access, it is not possible to read data in two registers simultaneously.
6. With 32-bit access, it is not possible to read data in the register at [accessed address + 2] simultaneously.

B.2 Register Bits

Table B.2 Register Bits

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
SDMR	—	—	—	—					BSC
SCSMR	C/A	CHR	PE	O/E	STOP	MP	CKS1	CKS0	SCI
SCBRR									SCI
SCSCR	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	SCI
SCTDR									SCI
SCSSR	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT	SCI
SCRDR									SCI
SCSCMR	—	—	—	—	SDIR	SINV	—	SMIF	SCI
TOCR	—	—	—	—	—	—	—	TCOE	TMU
TSTR	—	—	—	—	—	STR2	STR1	STR0	TMU
TCOR0									TMU
TCNT0									TMU
TCR0	—	—	—	—	—	—	—	UNF	TMU
	—	—	UNIE	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	
TCOR1									TMU
TCNT1									TMU

Table B.2 Register Bits (cont)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
TCR1	—	—	—	—	—	—	—	UNF	TMU
	—	—	UNIE	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	
TCOR2									TMU
TCNT2									TMU
TCR2	—	—	—	—	—	—	ICPF	UNF	TMU
	ICPE1	ICPE0	UNIE	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	
TCPR2									TMU
R64CNT	—	1 Hz	2 Hz	4 Hz	8 Hz	16 Hz	32 Hz	64 Hz	RTC
RSECCNT	—	10 sec			1 sec				RTC
RMINCNT	—	10 min			1 min				RTC
RHRCNT	—	—	10 hours		1 hour				RTC
RWKCNT	—	—	—	—	—	day of week			RTC
RDAYCNT	—	—	10 days		1 day				RTC
RMONCNT	—	—	—	10 months	1 month				RTC
RYRCNT	10 years				1 year				RTC
RSECAR	ENB	10 sec			1 sec				RTC
RMINAR	ENB	10 min			1 min				RTC
RHRAR	ENB	—	10 hours		1 hour				RTC
RWKAR	ENB	—	—	—	—	day of week			RTC
RDAYAR	ENB	—	10 days		1 day				RTC
RMONAR	ENB	—	—	10 months	1 month				RTC

Table B.2 Register Bits (cont)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
RCR1	CF	—	—	CIE	AIE	—	—	AF	RTC
RCR2	PEF	PES2	PES1	PES0	RTCEN	ADJ	RESET	START	RTC
ICR0	NML	—	—	—	—	—	—	NMIE	INTC
	—	—	—	—	—	—	—	—	
IPRA	TMU0				TMU1				INTC
	TMU2				RTC				
IPRB	WDT				REF				INTC
	SCI				—	—	—	—	
BCR1	PULA	PULD	HIZMEM	HIZCNT	ENDIAN	A0BST1	A0BST0	A5BST1	BSC
	A5BST0	A6BST1	A6BST0	DRAMTP2	DRAMTP1	DRAMTP0	A5PCM	A6PCM	
BCR2	—	—	A6SZ1	A6SZ0	A5SZ1	A5SZ0	A4SZ1	A4SZ0	BSC
	A3SZ1	A3SZ0	A2SZ1	A2SZ0	—	—	—	—	
BCR3	EXTEND	—	TPC31	TPC30	RCD31	RCD30	TRAS31	TRAS30	BSC
	—	—	TPC21	TPC20	RCD21	RCD20	TRAS21	TRAS20	
WCR1	WAITSEL	—	A6IW1	A6IW0	A5IW1	A5IW0	A4IW1	A4IW0	BSC
	A3IW1	A3IW0	A2IW1	A2IW0	—	—	A0IW1	A0IW0	
WCR2	A6W2	A6W1	A6W0	A5W2	A5W1	A5W0	A4W2	A4W1	BSC
	A4W0	A3W1	A3W0	A2W1	A2W0	A0W2	A0W1	A0W0	
MCR	TPC1	TPC0	RCD1	RCD0	TRWL1	TRWL0	TRAS1	TRAS0	BSC
	RASD	BE	AMX2	AMX1	AMX0	RFSH	RMODE	EDOMODE	
DCR	TPC1	TPC0	RCD1	RCD0	—	—	TRAS1	TRAS0	BSC
	—	BE	—	AMX1	AMX0	RFSH	RMODE	—	
PCR	A6W3	A5W3	—	—	A5TED2	A6TED2	A5TEH2	A6TEH2	BSC
	A5TED1	A5TED0	A6TED1	A6TED0	A5TEH1	A5TEH0	A6TEH1	A6TEH0	
RTCSR	—	—	—	—	—	—	—	—	BSC
	CMF	CMIE	CKS2	CKS1	CKS0	OVF	OVIE	LMTS	
RTCNT	—	—	—	—	—	—	—	—	BSC
RTCOR	—	—	—	—	—	—	—	—	BSC

Table B.2 Register Bits (cont)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
RFCR	—	—	—	—	—	—	—	—	BSC
FRQCR	STC2	IFC2	PFC2	—	—	—	SLPFRQ	CKOEN	CPG
	PLLEN	PSTBY	STC1	STC0	IFC1	IFC0	PFC1	PFC0	
STBCR	STBY	—	—	—	—	MSTP2	MSTP1	MSTP0	CPG
STBCR2	—	MDCHG	MSTP8	MSTP7	MSTP6	MSTP5	MSTP4	MSTP3	CPG
WTCNT									CPG
WTCSR	TME	WT/IT	RSTS	WOVF	IOVF	CKS2	CKS1	CKS0	CPG
BDRB									UBC
BDMRB									UBC
BRCR	CMFA	CMFB	—	—	—	PCBA	—	—	UBC
	DBEB	PCBB	—	—	SEQ	—	—	—	
BARB									UBC
BAMRB	—	—	—	—	—	BASM	BAM	BAM	UBC
BBRB	—	—	—	—	—	—	—	—	UBC
	—	—	ID	ID	RW	RW	SZ	SZ	
BARA									UBC

Table B.2 Register Bits (cont)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
BAMRA	—	—	—	—	—	BASM	BAM	BAM	UBC
BBRA	—	—	—	—	—	—	—	—	UBC
	—	—	ID	ID	RW	RW	SZ	SZ	
TRA	—	—	—	—	—	—	—	—	CCN
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—			
							—	—	
EXPEVT	—	—	—	—	—	—	—	—	CCN
	—	—	—	—	—	—	—	—	
	—	—	—	—					
INTEVT	—	—	—	—	—	—	—	—	CCN
	—	—	—	—	—	—	—	—	
	—	—	—	—					
MMUCR	—	—	—	—	—	—	—	—	CCN
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	SV	
	—	—	RC	RC	—	TF	IX	AT	
BASRA									UBC
BASRB									UBC
CCR	—	—	—	—	—	—	—	—	CCN
	—	—	—	—	—	—	—	—	
	—	—	—	—	—	—	—	—	
	—	—	RA	0	CF	CB	WT	CE	
PTEH									CCN
							—	—	

Table B.2 Register Bits (cont)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
PTEL									CCN
							—	V	
	—	PR	PR	SZ	C	D	SH	—	
TTB									CCN
TEA									CCN
INTEVT2									INTC
IRR0									INTC
	PINT0R	PINT1R	IRQ5R	IRQ4R	IRQ3R	IRQ2R	IRQ1R	IRQ0R	
IRR1									INTC
	TXI1R	BRI1R	RXI1R	ERI1R	DEI3R	DEI2R	DEI1R	DEI0R	
IRR2									INTC
	—	—	—	ADIR	TXI2R	BRI2R	RXI2R	ERI2R	
ICR1	MAI	IRQLVL	BLMSK	—	IRQ51S	IRQ50S	IRQ41S	IRQ40S	INTC
	IRQ31S	IRQ30S	IRQ21S	IRQ20S	IRQ11S	IRQ10S	IRQ01S	IRQ00S	
ICR2	INT15S	INT14S	INT13S	INT12S	INT11S	INT10S	INT9S	INT8S	INTC
	INT7S	INT6S	INT5S	INT4S	INT3S	INT2S	INT1S	INT0S	
INTER	INT15E	INT14E	INT13E	INT12E	INT11E	INT10E	INT9E	INT8E	INTC
	INT7E	INT6E	INT5E	INT4E	INT3E	INT2E	INT1E	INT0E	
IPRC	IRQ3's level				IRQ2's level				INTC
	IRQ1's level				IRQ0's level				

Table B.2 Register Bits (cont)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
IPRD	PINT0 to 7's level				PINT8 to 15's level				INTC
	IRQ5's level				IRQ4's level				
IPRE	DMAC's level				IrDA's level				INTC
	SCIF's level				A/D's level				
									INTC
SAR0									DMAC
DAR0									DMAC
DMATCR0	—	—	—	—	—	—	—	—	DMAC
CHCR0	—	—	—	—	—	—	—	—	DMAC
	—	—	—	DI	RO	—	AM	AL	
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
	—	DS	TM	TS1	TS0	IE	TE	DE	
SAR1									DMAC
DAR1									DMAC

Table B.2 Register Bits (cont)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
DMATCR1	—	—	—	—	—	—	—	—	DMAC
CHCR1	—	—	—	—	—	—	—	—	DMAC
	—	—	—	DI	RO	—	AM	AL	
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
	—	DS	TM	TS1	TS0	IE	TE	DE	
SAR2									DMAC
DAR2									DMAC
DMATCR2	—	—	—	—	—	—	—	—	DMAC
CHCR2	—	—	—	—	—	—	—	—	DMAC
	—	—	—	DI	RO	RL	AM	AL	
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
	—	DS	TM	TS1	TS0	IE	TE	DE	
SAR3									DMAC

Table B.2 Register Bits (cont)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
DAR3									DMAC
DMATCR3	—	—	—	—	—	—	—	—	DMAC
CHCR3	—	—	—	—	—	—	—	—	DMAC
	—	—	—	DI	RO	RL	AM	AL	
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
	—	DS	TM	TS1	TS0	IE	TE	DE	
DMAOR	—	—	—	—	—	—	PR1	PR0	DMAC
	—	—	—	—	—	AE	NMIF	DME	
CMSTR	—	—	—	—	—	—	—	—	CMT
	—	—	—	—	—	—	—	STR	
CMCSR	—	—	—	—	—	—	—	—	CMT
	CMF	—	—	—	—	—	CKS1	CKS0	
CMCNT									CMT
CMCOR									CMT
ADDRAH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D
ADDRAL	AD1	AD0	—	—	—	—	—	—	A/D
ADDRBH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D
ADDRBL	AD1	AD0	—	—	—	—	—	—	A/D
ADDRCH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D
ADDRCL	AD1	AD0	—	—	—	—	—	—	A/D
ADDRDH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D
ADDRDL	AD1	AD0	—	—	—	—	—	—	A/D
ADCSR	ADF	ADE	ADST	SCAN	CKS	CH2	CH1	CH0	A/D

Table B.2 Registers Bits (cont)

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
ADCR	TRGE	—	—	—	—	—	—	—	A/D
DADR0									D/A
DADR1									D/A
DACR	DAOE1	DAOE0	DAE	—	—	—	—	—	D/A
DASTCR		—	—	—	—	—	—	DASTE	D/A
PACR	PA7M D1	PA7M D0	PA6M D1	PA6M D0	PA5M D1	PA5M D0	PA4M D1	PA4M D0	PORT
	PA3M D1	PA3M D0	PA2M D1	PA2M D0	PA1M D1	PA1M D0	PA0M D1	PA0M D0	
PBCR	PB7M D1	PB7M D0	PB6M D1	PB6M D0	PB5M D1	PB5M D0	PB4M D1	PB4M D0	PORT
	PB3M D1	PB3M D0	PB2M D1	PB2M D0	PB1M D1	PB1M D0	PB0M D1	PB0M D0	
PCDR	PC7M D1	PC7M D0	PC6M D1	PC6M D0	PC5M D1	PC5M D0	PC4M D1	PC4M D0	PORT
	PC3M D1	PC3M D0	PC2M D1	PC2M D0	PC1M D1	PC1M D0	PC0M D1	PC0M D0	
PDCR	PD7M D1	PD7M D0	PD6M D1	PD6M D0	PD5M D1	PD5M D0	PD4M D1	PD4M D0	PORT
	—	—	—	—	—	—	—	—	
PECR	PE7M D1	PE7M D0	PE6M D1	PE6M D0	PE5M D1	PE5M D0	PE4M D1	PE4M D0	PORT
	PE3M D1	PE3M D0	PE2M D1	PE2M D0	PE1M D1	PE1M D0	PE0M D1	PE0M D0	
PFCR	PF7M D1	PF7M D0	PF6M D1	PF6M D0	PF5M D1	PF5M D0	PF4M D1	PF4M D0	PORT
	PF3M D1	PF3M D0	PF2M D1	PF2M D0	PF1M D1	PF1M D0	PF0M D1	PF0M D0	
PGCR	PG7M D1	PG7M D0	PG6M D1	PG6M D0	PG5M D1	PG5M D0	PG4M D1	PG4M D0	PORT
	PG3M D1	PG3M D0	PG2M D1	PG2M D0	PG1M D1	PG1M D0	PG0M D1	PG0M D0	
PHCR	PH7M D1	PH7M D0	PH6M D1	PH6M D0	PH5M D1	PH5M D0	PH4M D1	PH4M D0	PORT
	PH3M D1	PH3M D0	PH2M D1	PH2M D0	PH1M D1	PH1M D0	PH0M D1	PH0M D0	

Table B.2 Register Bits (cont)

Register	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	Module
PJCR	PJ7MD1	PJ7MD0	PJ6MD1	PJ6MD0	PJ5MD1	PJ5MD0	PJ4MD1	PJ4MD0	PORT
	PJ3MD1	PJ3MD0	PJ2MD1	PJ2MD0	PJ1MD1	PJ1MD0	PJ0MD1	PJ0MD0	
PKCR	PK7MD1	PK7MD0	PK6MD1	PK6MD0	PK5MD1	PK5MD0	PK4MD1	PK4MD0	PORT
	PK3MD1	PK3MD0	PK2MD1	PK2MD0	PK1MD1	PK1MD0	PK0MD1	PK0MD0	
PLCR	PL7MD1	PL7MD0	PL6MD1	PL6MD0	PL5MD1	PL5MD0	PL4MD1	PL4MD0	PORT
	PL3MD1	PL3MD0	PL2MD1	PL2MD0	PL1MD1	PL1MD0	PL0MD1	PL0MD0	
SCPCR	SCP7MD1	SCP7MD0	SCP6MD1	SCP6MD0	SCP5MD1	SCP5MD0	SCP4MD1	SCP4MD0	PORT
	SCP3MD1	SCP3MD0	SCP2MD1	SCP2MD0	SCP1MD1	SCP1MD0	SCP0MD1	SCP0MD0	
PADR	PA7DT	PA6DT	PA5DT	PA4DT	PA3DT	PA2DT	PA1DT	PA0DT	PORT
PBDR	PB7DT	PB6DT	PB5DT	PB4DT	PB3DT	PB2DT	PB1DT	PB0DT	PORT
PCDR	PC7DT	PC6DT	PC5DT	PC4DT	PC3DT	PC2DT	PC1DT	PC0DT	PORT
PDDR	PD7DT	PD6DT	PD5DT	PD4DT	PD3DT	PD2DT	PD1DT	PD0DT	PORT
PEDR	PE7DT	PE6DT	PE5DT	PE4DT	PE3DT	PE2DT	PE1DT	PE0DT	PORT
PFDR	PF7DT	PF6DT	PF5DT	PF4DT	PF3DT	PF2DT	PF1DT	PF0DT	PORT
PGDR	PG7DT	PG6DT	PG5DT	PG4DT	PG3DT	PG2DT	PG1DT	PG0DT	PORT
PHDR	PH7DT	PH6DT	PH5DT	PH4DT	PH3DT	PH2DT	PH1DT	PH0DT	PORT
PJDR	PJ7DT	PJ6DT	PJ5DT	PJ4DT	PJ3DT	PJ2DT	PJ1DT	PJ0DT	PORT
PKDR	PK7DT	PK6DT	PK5DT	PK4DT	PK3DT	PK2DT	PK1DT	PK0DT	PORT
PLDR	PL7DT	PL6DT	PL5DT	PL4DT	PL3DT	PL2DT	PL1DT	PL0DT	PORT
SCPDR	SCP7DT	SCP6DT	SCP5DT	SCP4DT	SCP3DT	SCP2DT	SCP1DT	SCP0DT	PORT

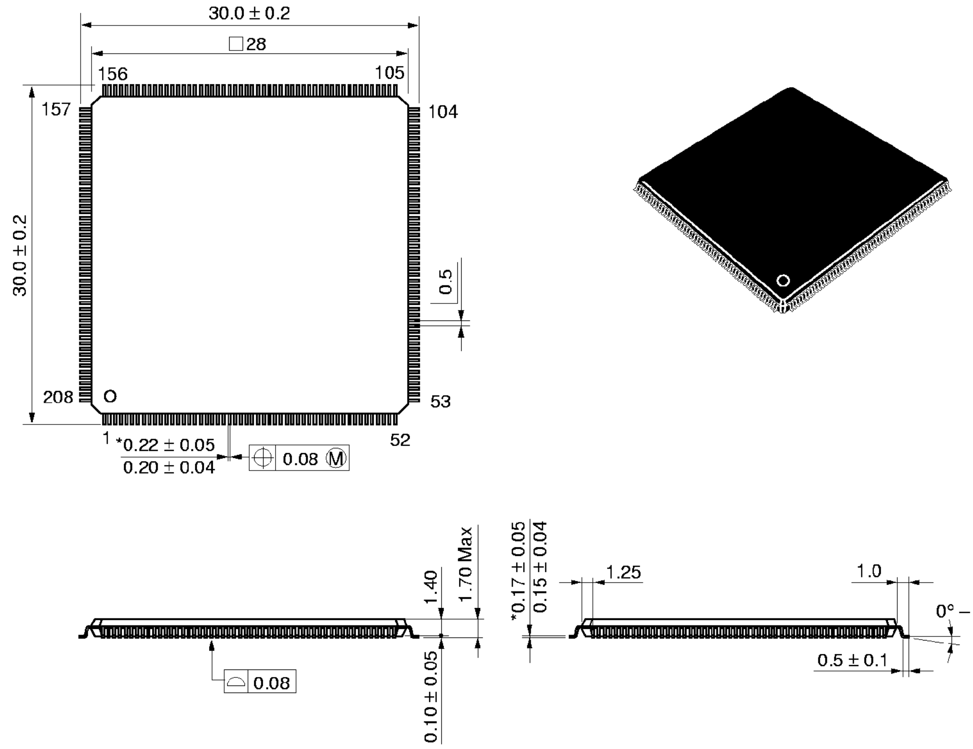
Table B.2 Register Bits (cont)

Register	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0	Module
SDIR	TI3	TI2	TI1	TI0	—	—	—	—	UDI
	—	—	—	—	—	—	—	—	
SDSR	PR3	PR2	PR1	PR0	VR3	VR2	VR1	VR0	UDI
	—	—	—	—	—	ASEMW	BRKAF	SDTRF	
SDDR (SDDRL)									UDI
SDAR	—	—	—	—	—	—	AR9	AR8	UDI
	AR7	AR6	AR5	AR4	AR3	AR2	—	—	
SDARE	—	—	—	—	—	—	ARE9	ARE8	UDI
	ARE7	ARE6	ARE5	ARE4	ARE3	ARE2	—	—	

Appendix C Package Dimensions

Figures C.1 and C.2 show the SH7729 package dimensions.

Unit: mm



Hitachi Code	FP-208C
JEDEC	—
EIAJ	Conforms
Weight (reference value)	2.7 g

*Dimension including the plating thickness
Base material dimension

Figure C.1 Package Dimensions (FP-208C)

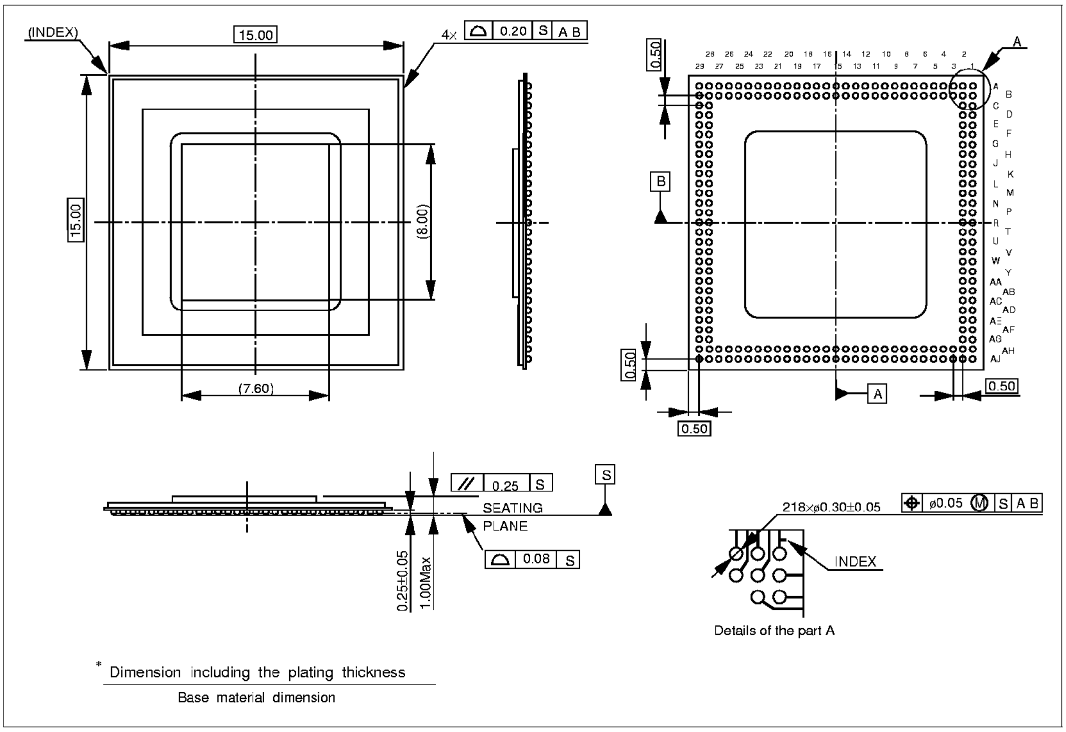


Figure C.2 Package Dimensions (CSP-216)

SH7729 Hardware Manual

Publication Date: 1st Edition, March 1999
Published by: Electronic Devices Sales & Marketing Business Group
Semiconductor & Integrated Circuits Group
Hitachi, Ltd.
Edited by: Technical Documentation Group
UL Media Co., Ltd.