SH-4

Hardware Manual
Preliminary

HITACHI

Version 1.1
7/31/98
Hitachi, Ltd.

Contents

Contents..... i

Section 1 Overview 13
1.1 SH7750 Features 13
1.2 Block Diagram 20

Section 2 Programming Model 21
2.1 Data Formats 21
2.2 Register Configuration 22
2.2.1 Privileged Mode and Banks 22
2.2.2 General Registers 25
2.2.3 Floating-Point Registers 27
2.2.4 Control Registers 29
2.2.5 System Registers 30
2.3 Memory-Mapped Registers 32
2.4 Data Format in Registers 33
2.5 Data Formats in Memory 33
2.6 Processor States 34
2.7 Processor Modes 36
3.1 Overview 37
3.1.1 Features 37
3.1.2 Role of the MMU 37
3.1.3 Register Configuration 40
3.1.4 Caution 40
3.2 Register Descriptions 41
3.3 Memory Space 44
3.3.1 Physical Memory Space 44
3.3.2 External Memory Space 47
3.3.3 Virtual Memory Space 48
3.3.4 On-Chip RAM Space 49
3.3.5 Address Translation 49
3.3.6 Single Virtual Memory Mode and Multiple Virtual Memory Mode....................50
3.3.7 Address Space Identifier (ASID) 50
3.4 TLB Functions 51
3.4.1 Unified TLB (UTLB) Configuration 51
3.4.2 Instruction TLB (ITLB) Configuration 55
3.4.3 Address Translation Method 56
3.5 MMU Functions 58
3.5.1 MMU Hardware Management 58

3.5.2 MMU Software Management 58
3.5.3 MMU Instruction (LDTLB) 58
3.5.4 Hardware ITLB Miss Handling 59
3.5.5 Avoiding Synonym Problems 60
3.6 MMU Exceptions 61
3.6.1 Instruction TLB Multiple Hit Exception 61
3.6.2 Instruction TLB Miss Exception 62
3.6.3 Instruction TLB Protection Violation Exception 63
3.6.4 Data TLB Multiple Hit Exception 64
3.6.5 Data TLB Miss Exception 64
3.6.6 Data TLB Protection Violation Exception 65
3.6.7 Initial Page Write Exception 66
3.7 Memory-Mapped TLB Configuration 67
3.7.1 ITLB Address Array 68
3.7.2 ITLB Data Array 1 69
3.7.3 ITLB Data Array 2 70
3.7.4 UTLB Address Array 71
3.7.5 UTLB Data Array 1 72
3.7.6 UTLB Data Array 2 73
4.1 Overview 75
4.1.1 Features 75
4.1.2 Register Configuration 76
4.2 Register Descriptions 76
4.3 Operand Cache (OC) 79
4.3.1 Configuration 79
4.3.2 Read Operation 80
4.3.3 Write Operation 81
4.3.4 Write-Back Buffer 83
4.3.5 Write-Through Buffer 83
4.3.6 RAM Mode 83
4.3.7 OC Index Mode 84
4.3.8 Coherency between Cache and External Memory 85
4.3.9 Prefetch Operation 85
4.4 Instruction Cache (IC) 86
4.4.1 Configuration 86
4.4.2 Read Operation 87
4.4.3 IC Index Mode 88
4.5 Memory-Mapped Cache Configuration 88
4.5.1 IC Address Array 88
4.5.2 IC Data Array 90
4.5.3 OC Address Array 91
4.5.4 OC Data Array 92
4.6 Store Queues 93

4.6.1 SQ Configuration 93
4.6.2 SQ Writes 94
4.6.3 Transfer to External Memory 94
4.6.4 SQ Protection 95

Section 5 Exceptions 97
5.1 Overview 97
5.1.1 Features 97
5.1.2 Register Configuration 97
5.2 Register Descriptions 98
5.3 Exception Handling Functions 99
5.3.1 Exception Handling Flow 99
5.3.2 Exception Handling Vector Addresses 99
5.4 Exception Types and Priorities 100
5.5 Exception Flow 103
5.5.1 Exception Flow 103
5.5.2 Exception Source Acceptance 104
5.5.3 Exception Requests and BL Bit 106
5.5.4 Return from Exception Handling 106
5.6 Description of Exceptions 107
5.6.1 Resets 107
5.6.2 General Exceptions 112
5.6.3 Interrupts 126
5.6.4 Priority Order with Multiple Exceptions 129
5.7 Usage Notes 130
5.8 Restrictions 131

Section 6 Floating-Point Unit 133
6.1 Overview 133
6.2 Data Formats 133
6.2.1 Floating-Point Format 133
6.2.2 Non-Numbers (NaN) 135
6.2.3 Denormalized Numbers 136
6.3 Registers 137
6.3.1 Floating-Point Registers 137
6.3.2 Floating-Point Status/Control Register (FPSCR) 139
6.3.3 Floating-Point Communication Register (FPUL) 140
6.4 Rounding 140
6.5 Floating-Point Exceptions 141
6.6 Graphics Support Functions 143
6.6.1 Geometric Operation Instructions 143
6.6.2 Pair Single-Precision Data Transfer 144
7.1 Execution Environment 145

7.2 Addressing Modes.. 147
7.3 Instruction Set.. 151

Section 8 Pipelining...165
8.1 Pipelines 165
8.2 Parallel-Executability.. 172
8.3 Execution Cycles and Pipeline Stalling.. 176
9.1 Overview 193
9.1.1 Types of Power-Down Modes.. 193
9.1.2 Register Configuration... 195
9.1.3 Pin Configuration 195
9.2 Register Descriptions... 195
9.2.1 Standby Control Register (STBCR) 195
9.2.2 Peripheral Module Pin High Impedance Control.. 198
9.2.3 Peripheral Module Pin Pull-Up Control 198
9.2.4 Standby Control Register 2 (STBCR2).. 199
9.3 Sleep Mode.. 200
9.3.1 Transition to Sleep Mode... 200
9.3.2 Exit from Sleep Mode.. 200
9.4 Deep Sleep Mode... 200
9.4.1 Transition to Deep Sleep Mode... 200
9.4.2 Exit from Deep Sleep Mode... 200
9.5 Standby Mode.. 201
9.5.1 Transition to Standby Mode... 201
9.5.2 Exit from Standby Mode.. 202
9.5.3 Clock Pause Function 202
9.6 Module Standby Function.. 203
9.6.1 Transition to Module Standby Function.. 203
9.6.2 Exit from Module Standby Function.. 203
9.7 STATUS Pin Change Timing... 204
9.7.1 In Reset 204
9.7.2 In Exit from Standby Mode.. 205
9.7.3 In Exit from Sleep Mode.. 207
9.7.4 In Exit from Deep Sleep Mode 210

Section 10 Clock Oscillation Circuits..213
10.1 Overview 213
10.1.1 Features 213
10.2 Overview of CPG... 215
10.2.1 Block Diagram of CPG.. 215
10.2.2 CPG Pin Configuration.. 217
10.2.3 CPG Register Configuration 217
10.3 Clock Operating Modes 218

10.4 CPG Register Description 220
10.4.1 Frequency Control Register (FRQCR).......... 220
10.5 Changing the Frequency.......... 222
10.5.1 Changing PLL Circuit 1 Starting/Stopping (When PLL Circuit 2 is Off).......... 222
10.5.2 Changing PLL Circuit 1 Starting/Stopping (When PLL Circuit 2 is On) 223
10.5.3 Changing Bus Clock Division Ratio (When PLL Circuit 2 is On) 223
10.5.4 Changing Bus Clock Division Ratio (When PLL Circuit 2 is Off).......... 223
10.5.5 Changing CPU or Peripheral Module Clock Division Ratio 224
10.6 Output Clock Control.......... 224
10.7 Overview of Watchdog Timer.......... 224
10.7.1 Block Diagram.......... 224
10.7.2 Register Configuration.......... 225
10.8 WDT Register Descriptions.......... 225
10.8.1 Watchdog Timer Counter (WTCNT) 225
10.8.2 Watchdog Timer Control/Status Register (WTCSR) 226
10.8.3 Notes on Register Access.......... 228
10.9 Using the WDT.......... 229
10.9.1 Standby Clearing Procedure.......... 229
10.9.2 Frequency Changing Procedure.......... 229
10.9.3 Using Watchdog Timer Mode.......... 230
10.9.4 Using Interval Timer Mode.......... 230
10.10 Notes on Board Design.......... 231
11.1 Overview 233
11.1.1 Features 233
11.1.2 Block Diagram.......... 234
11.1.3 Pin Configuration.......... 235
11.1.4 Register Configuration.......... 235
11.2 Register Descriptions 237
11.2.1 64 Hz Counter (R64CNT) 237
11.2.2 Second Counter (RSECCNT).......... 237
11.2.3 Minute Counter (RMINCNT).......... 238
11.2.4 Hour Counter (RHRCNT) 238
11.2.5 Day-of-Week Counter (RWKCNT).......... 239
11.2.6 Day Counter (RDAYCNT).......... 240
11.2.7 Month Counter (RMONCNT) 241
11.2.8 Year Counter (RYRCNT) 242
11.2.9 Second Alarm Register (RSECAR).......... 243
11.2.10 Minute Alarm Register (RMINAR).......... 244
11.2.11 Hour Alarm Register (RHRAR) 245
11.2.12 Day-of-Week Alarm Register (RWKAR).......... 246
11.2.13 Day Alarm Register (RDAYAR).......... 247
11.2.14 Month Alarm Register (RMONAR) 248
11.2.15 RTC Control Register 1 (RCR1) 249

11.2.16 RTC Control Register 2 (RCR2) 251
11.3 Operation 253
11.3.1 Time Setting Procedures 253
11.3.2 Time Reading Procedures 254
11.3.3 Alarm Function 255
11.4 Interrupts 256
11.5 Usage Notes 256
11.5.1 Register Initialization 256
11.5.2 Crystal Oscillator Circuit 256
12.1 Overview 259
12.1.1 Features 259
12.1.2 Block Diagram 260
12.1.3 Pin Configuration 260
12.1.4 Register Configuration 261
12.2 Register Descriptions 262
12.2.1 Timer Output Control Register (TOCR) 262
12.2.2 Timer Start Register (TSTR) 263
12.2.3 Timer Constant Registers (TCOR) 264
12.2.4 Timer Counters (TCNT) 264
12.2.5 Timer Control Registers (TCR) 265
12.2.6 Input Capture Register (TCPR2) 268
12.3 Operation 269
12.3.1 Counter Operation 269
12.3.2 Input Capture Function 272
12.4 Interrupts 274
12.5 Usage Notes 275
12.5.1 Register Writes 275
12.5.2 TCNT Register Reads 275
12.5.3 Resetting the RTC Frequency Divider 275
12.5.4 External Clock Frequency 275

Section 13 Bus State Controller (BSC) 277
13.1 Overview 277
13.1.1 Features 277
13.1.2 Block Diagram 279
13.1.3 Pin Configuration 280
13.1.4 Register Configuration 283
13.1.5 Overview of Areas 284
13.1.6 PCMCIA Support 287
13.2 Register Descriptions 291
13.2.1 Bus Control Register 1 (BCR1) 291
13.2.2 Bus Control Register 2 (BCR2) 299
13.2.3 Wait Control Register 1 (WCR1) 300

13.2.4 Wait Control Register 2 (WCR2) 303
13.2.5 Wait Control Register 3 (WCR3) 311
13.2.6 Memory Control Register (MCR)........ 313
13.2.7 PCMCIA Control Register (PCR) 320
13.2.8 Synchronous DRAM Mode Register (SDMR) 323
13.2.9 Refresh Timer Control/Status Register (RTSCR) 325
13.2.10 Refresh Timer Counter (RTCNT)........ 327
13.2.11 Refresh Time Constant Register (RTCOR)........ 328
13.2.12 Refresh Count Register (RFCR) 329
13.2.13 Notes on Accessing Refresh Control Registers 330
13.3 Operation 331
13.3.1 Endian/Access Size and Data Alignment........ 331
13.3.2 Areas 342
13.3.3 Basic Interface 347
13.3.4 DRAM Interface 355
13.3.5 Synchronous DRAM Interface 372
13.3.6 Burst ROM Interface........ 396
13.3.7 PCMCIA Interface 399
13.3.8 MPX Interface 408
13.3.9 Byte Control SRAM........ 415
13.3.10 Waits between Access Cycles 420
13.3.11 Bus Arbitration 421
13.3.12 Master Mode........ 424
13.3.13 Slave Mode........ 425
13.3.14 Partial-Sharing Master Mode........ 426
13.3.15 Cooperation between Master and Slave........ 427

Section 14 Direct Memory Access Controller (DMAC)........429
14.1 Overview 429
14.1.1 Features 429
14.1.2 Block Diagram........ 431
14.1.3 Pin Configuration........ 432
14.1.4 Register Configuration........ 433
14.2 Register Descriptions 435
14.2.1 DMA Source Address Registers 0–3 (SAR0–SAR3) 435
14.2.2 DMA Destination Address Registers 0–3 (DAR0–DAR3)........ 436
14.2.3 DMA Transfer Count Registers 0–3 (DMATCR0–DMATCR3) 437
14.2.4 DMA Channel Control Registers 0–3 (CHCR0–CHCR3)........ 438
14.2.5 DMA Operation Register (DMAOR)........ 446
14.3 Operation 448
14.3.1 DMA Transfer Procedure........ 448
14.3.2 DMA Transfer Requests........ 450
14.3.3 Channel Priorities 453

14.3.4 Types of DMA Transfer 456
14.3.5 Number of Bus Cycle States and DREQ Pin Sampling Timing 464
14.3.6 Ending DMA Transfer 479
14.4 Examples of Use 482
14.4.1 Examples of Transfer between External Memory and an External Device with DACK .482
14.5 On-Demand Data Transfer Mode 483
14.5.1 Operation 483
14.5.2 Notes on Use of DDT Module 485
14.6 Usage Notes 487

Section 15 Serial Communication Interface (SCI) 489
15.1 Overview 489
15.1.1 Features 489
15.1.2 Block Diagram 491
15.1.3 Pin Configuration 492
15.1.4 Register Configuration 492
15.2 Register Descriptions 493
15.2.1 Receive Shift Register (SCRSR1) 493
15.2.2 Receive Data Register (SCRDR1) 493
15.2.3 Transmit Shift Register (SCTSR1) 494
15.2.4 Transmit Data Register (SCTDR1) 494
15.2.5 Serial Mode Register (SCSMR1) 495
15.2.6 Serial Control Register (SCSCR1) 498
15.2.7 Serial Status Register (SCSSR1) 501
15.2.8 Serial Port Register (SCSPTR1) 505
15.2.9 Bit Rate Register (SCBRR1) 509
15.3 Operation 516
15.3.1 Overview 516
15.3.2 Operation in Asynchronous Mode 519
15.3.3 Multiprocessor Communication Function 529
15.3.4 Operation in Synchronous Mode 537
15.4 SCI Interrupt Sources and DMAC 547
15.5 Usage Notes 548

Section 16 Serial Communication Interface with FIFO (SCIF) 553
16.1 Overview 553
16.1.1 Features 553
16.1.2 Block Diagram 555
16.1.3 Pin Configuration 556
16.1.4 Register Configuration 557

16.2 Register Descriptions 558
16.2.1 Receive Shift Register (SCRSR2) 558
16.2.2 Receive FIFO Data Register (SCFRDR2).......... 558
16.2.3 Transmit Shift Register (SCTSR2).......... 559
16.2.4 Transmit FIFO Data Register (SCFTDR2) 559
16.2.5 Serial Mode Register (SCSMR2) 560
16.2.6 Serial Control Register (SCSCR2) 562
16.2.7 Serial Status Register (SCFSR2) 565
16.2.8 Bit Rate Register (SCBRR2).......... 572
16.2.9 FIFO Control Register (SCFCR2) 573
16.2.10 FIFO Data Count Register (SCFDR2) 575
16.2.11 Serial Port Register (SCSPTR2).......... 576
16.2.12 Line Status Register (SCLSR2).......... 581
16.3 Operation.......... 582
16.3.1 Overview 582
16.3.2 Serial Operation.......... 584
16.4 SCIF Interrupt Sources and the DMAC 594
16.5 Usage Notes.......... 595

Section 17 Smart Card Interface..........599
17.1 Overview 599
17.1.1 Features 599
17.1.2 Block Diagram.......... 600
17.1.3 Pin Configuration.......... 601
17.1.4 Register Configuration.......... 601
17.2 Register Descriptions 602
17.2.1 Smart Card Mode Register (SCSCMR1) 602
17.2.2 Serial Mode Register (SCSMR1) 603
17.2.3 Serial Control Register (SCSCR1) 604
17.2.4 Serial Status Register (SCSSR1) 605
17.3 Operation.......... 607
17.3.1 Overview 607
17.3.2 Pin Connections.......... 608
17.3.3 Data Format.......... 609
17.3.4 Register Settings 610
17.3.5 Clock 613
17.3.6 Data Transfer Operations 616
17.4 Usage Notes.......... 623

Section 18 I/O Ports 629
18.1 Overview 629
18.1.1 Features 629
18.1.2 Block Diagrams 630

18.1.3 Pin Configuration 636
18.1.4 Register Configuration 638
18.2 Register Descriptions 639
18.2.1 Port Control Register A (PCTRA) 639
18.2.2 Port Data Register A (PDTRA) 640
18.2.3 Port Control Register B (PCTRB) 641
18.2.4 Port Data Register B (PDTRB) 642
18.2.5 GPIO Interrupt Control Register (GPIOIC) 643
18.2.6 Serial Port Register (SCSPTR1) 644
18.2.7 Serial Port Register (SCSPTR2) 645
19.1 Overview 649
19.1.1 Features 649
19.1.2 Block Diagram 650
19.1.3 Pin Configuration 651
19.1.4 Register Configuration 651
19.2 Interrupt Sources 652
19.2.1 NMI Interrupt 652
19.2.2 IRL Interrupts 653
19.2.3 On-Chip Peripheral Module Interrupts 655
19.2.4 Interrupt Exception Handling and Priority 656
19.3 Register Descriptions 659
19.3.1 Interrupt Priority Registers A to C (IPRA–IPRC) 659
19.3.2 Interrupt Control Register (ICR) 660
19.4 INTC Operation 662
19.4.1 Interrupt Operation Sequence 662
19.4.2 Multiple Interrupts 664
19.4.3 Interrupt Masking with MAI Bit 664
19.5 Interrupt Response Time 665
20.1 Overview 667
20.1.1 Features 667
20.1.2 Block Diagram 668
20.2 Register Descriptions 670
20.2.1 Access to UBC Control Registers 670
20.2.2 Break Address Register A (BARA) 671
20.2.3 Break ASID Register A (BASRA) 672
20.2.4 Break Address Mask Register A (BAMRA) 672
20.2.5 Break Bus Cycle Register A (BBRA) 673
20.2.6 Break Address Register B (BARB) 675
20.2.7 Break ASID Register B (BASRB) 675
20.2.8 Break Address Mask Register B (BAMRB) 675
20.2.9 Break Data Register B (BDRB) 675
20.2.10 Break Data Mask Register B (BDMRB) 676
20.2.11 Break Bus Cycle Register B (BBRB) 677

20.2.12 Break Control Register (BRCR) 677
20.3 Operation 680
20.3.1 Explanation of Terms Relating to Accesses 680
20.3.2 Explanation of Terms Relating to Instruction Intervals 681
20.3.3 User Break Operation Sequence 681
20.3.4 Instruction Access Cycle Break 682
20.3.5 Operand Access Cycle Break 683
20.3.6 Condition Match Flag Setting 684
20.3.7 Program Counter (PC) Value Saved 684
20.3.8 Contiguous A and B Settings for Sequential Conditions 685
20.3.9 Usage Notes 686
20.4 User Break Debug Support Function 687
20.5 Examples of Use 689

Section 21 Hitachi User Debug Interface (Hitachi-UDI) 691
21.1 Overview 691
21.1.1 Features 691
21.1.2 Block Diagram 692
21.1.3 Pin Configuration 693
21.1.4 Register Configuration 694
21.2 Register Descriptions 695
21.2.1 Instruction Register (SDIR) 695
21.2.2 Data Register (SDDR) 696
21.2.3 Bypass Register (SDBPR) 696
21.3 Operation 697
21.3.1 TAP Control 697
21.3.2 Hitachi-UDI Reset 698
21.3.3 Hitachi-UDI Interrupt 698
21.3.4 Bypass 698
21.4 Usage Notes 699

Section 22 Pin Description 700
22.1 Pin Arrangement 700
22.2 Pin Functions 702
22.2.1 Pin Functions (256-Pin BGA) 702
22.2.2 Pin Functions (208-Pin QFP) 712

Section 23 Electrical Characteristics 721
23.1 Absolute Maximum Ratings 721
23.2 DC Characteristics 722
23.3 AC Characteristics 724
23.3.1 Clock and Control Signal Timing 725
23.3.2 Control Signal Timing 732

23.3.3. Bus Timing 734
23.3.4 Peripheral Module Signal Timing 783
23.3.5 AC Characteristic Test Conditions 788
23.3.6 Delay Time Variation Due to Load Capacitance 789

Appendix A Address List 791

Appendix B Package Dimensions 795

Appendix C Mode Pin Settings 797

Appendix D CKIO2ENB Pin Configuration 799

Appendix E Pin Functions 801
E.1 Pin States 801
E.2 Handling of Unused Pins 804

Appendix F Synchronous DRAM Address Multiplexing Tables 805

Appendix G SH7750 On-Demand Data Transfer Mode 823
G.1 Pins in DDT Mode 823
G.2 Transfer Request Acceptance on Each Channel 826

Section 1 Overview

1.1 SH7750 Features

The SH7750 is a 32-bit RISC (reduced instruction set computer) microprocessor, featuring object code upward-compatibility with SH-1, SH-2, SH-3, and SH-3E microcomputers. It includes an 8-kbyte instruction cache, a 16-kbyte operand cache with a choice of copy-back or write-through mode, and an MMU (memory management unit) with a 64-entry fully-associative unified TLB (translation lookaside buffer).

The SH7750 has an on-chip bus state controller (BSC) that allows direct connection to DRAM and synchronous DRAM without external circuitry. Its 16-bit fixed-length instruction set enables program code size to be reduced by almost 50% compared with 32-bit instructions.

The features of the SH7750 are summarized in table 1.1.

Table 1.1 SH7750 Features

Item	Features
LSI	• Operating frequency: 200 MHz • Performance: — 360 MIPS (200 MHz) — 1.4 GFLOPS (200 MHz) • Superscalar architecture: Parallel execution of two instructions • Voltage: 1.8 V (internal), 3.3 V (I/O) • Packages: 256-pin BGA, 208-pin QFP • External buses — Separate 26-bit address and 64-bit data buses — External bus frequency of 1/2, 1/3, 1/4, 1/6, or 1/8 times internal bus frequency
CPU	• Original Hitachi SH architecture • 32-bit internal data bus • General register file: — Sixteen 32-bit general registers (and eight 32-bit shadow registers) — Seven 32-bit control registers — Four 32-bit system registers • RISC-type instruction set (upward-compatible with SH Series) — Fixed 16-bit instruction length for improved code efficiency — Load-store architecture — Delayed branch instructions — Conditional execution — C-based instruction set • Superscalar architecture (providing simultaneous execution of two instructions) including FPU • Instruction execution time: Maximum 2 instructions/cycle • Virtual address space: 4 Gbytes (448-Mbyte external memory space) • Space identifier ASIDs: 8 bits, 256 virtual address spaces • On-chip multiplier • Five-stage pipeline

Table 1.1 SH7750 Features (cont)

Item	Features
FPU	• On-chip floating-point coprocessor • Supports single-precision (32 bits) and double-precision (64 bits) • Supports IEEE754-compliant data types and exceptions • Two rounding modes: Round to Nearest and Round to Zero • Handling of denormalized numbers: Truncation to zero or interrupt generation for compliance with IEEE754 • Floating-point registers: 32 bits × 16 words × 2 banks (single-precision × 16 words or double-precision × 8 words) × 2 banks • 32-bit CPU-FPU floating-point communication register (FPUL) • Supports FMAC (multiply-and-accumulate) instruction • Supports FDIV (divide) and FSQRT (square root) instructions • Supports FLDI0/FLDI1 (load constant 0/1) instructions • Instruction execution times — Latency (FMAC/FADD/FSUB/FMUL): 3 cycles (single-precision), 8 cycles (double-precision) — Pitch (FMAC/FADD/FSUB/FMUL): 1 cycle (single-precision), 6 cycles (double-precision) Note: FMAC is supported for single-precision only. • 3-D graphics instructions (single-precision only): — 4-dimensional vector conversion and matrix operations (FTRV): 4 cycles (pitch), 7 cycles (latency) — 4-dimensional vector (FIPR) inner product: 1 cycle (pitch), 4 cycles (latency) • Five-stage pipeline

Table 1.1 SH7750 Features (cont)

Item	Features
Clock pulse generator (CPG)	• Choice of main clock: 1/2, 1, 3, or 6 times EXTAL • Clock modes: — CPU frequency: 1, 1/2, 1/3, 1/4, 1/6, or 1/8 times main clock: maximum 200 MHz — Bus frequency: 1/2, 1/3, 1/4, 1/6, or 1/8 times main clock: maximum 100 MHz — Peripheral frequency: 1/2, 1/3, 1/4, 1/6, or 1/8 times main clock: maximum 50 MHz • Power-down modes — Sleep mode — Standby mode — Module standby function • Single-channel watchdog timer
Memory management unit (MMU)	• 4-Gbyte address space, 256 address space identifiers (8-bit ASIDs) • Single virtual mode and multiple virtual memory mode • Supports multiple page sizes: 1 kbyte, 4 kbytes, 64 kbytes, 1 Mbyte • 4-entry fully-associative TLB for instructions • 64-entry fully-associative TLB for instructions and operands • Supports software-controlled replacement and random-counter replacement algorithm • TLB contents can be accessed directly by address mapping

Table 1.1 SH7750 Features (cont)

Item	Features
Cache memory	• Instruction cache (IC) — 8 kbytes, direct mapping — 256 entries, 32-byte block length — Normal mode (8-kbyte cache) — Index mode • Operand cache (OC) — 16 kbytes, direct mapping — 512 entries, 32-byte block length — Normal mode (16-kbyte cache) — Index mode — RAM mode (8-kbyte cache + 8-kbyte RAM) — Choice of write method (copy-back or write-through) • Single-stage copy-back buffer, single-stage write-through buffer • Cache memory contents can be accessed directly by address mapping (usable as on-chip memory) • Store queue (32 bytes × 2 entries)
Interrupt controller (INTC)	• Five independent external interrupts (NMI, IRL3 to IRL0) • 15-level signed external interrupts: IRL3 to IRL0 • On-chip peripheral module interrupts: Priority level can be set for each module
User break controller (UBC)	• Supports debugging by means of user break interrupts • Two break channels • Address, data value, access type, and data size can all be set as break conditions • Supports sequential break function

Table 1.1 SH7750 Features (cont)

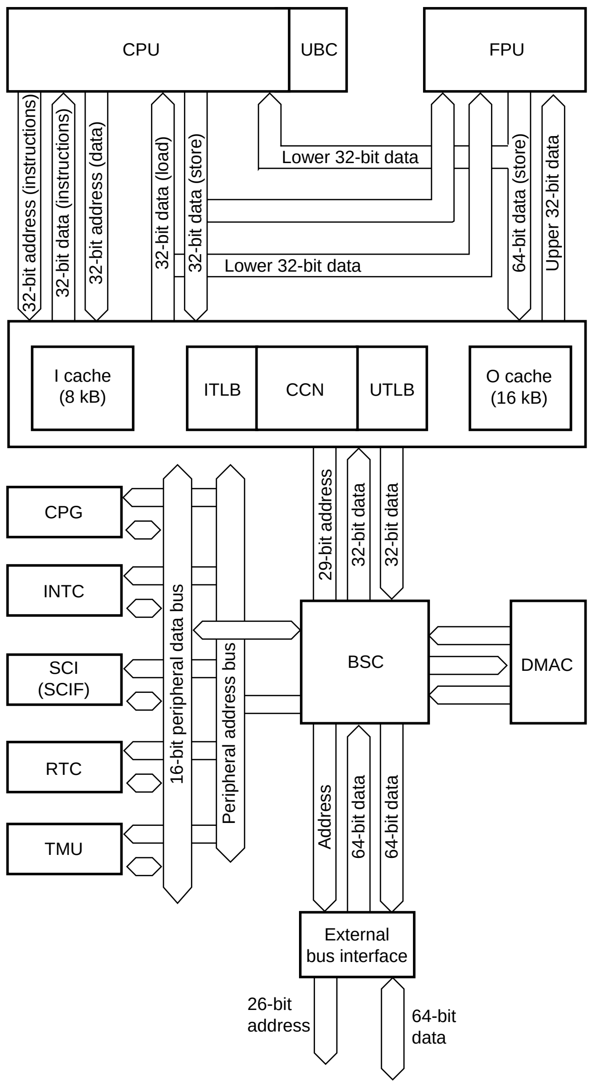
Item	Features
Bus state controller (BSC)	• Supports external memory access — 64/32/16/8-bit external data bus • External memory space divided into seven areas, each of up to 64 Mbytes, with the following parameters settable for each area: — Bus size (8, 16, 32, or 64 bits) — Number of wait cycles (hardware wait function also supported) — Direct connection of DRAM, synchronous DRAM, and burst ROM possible by setting space type — Supports fast page mode and DRAM EDO — Supports PCMCIA interface — Chip select signals ($\overline{\text{CS0}}$ to $\overline{\text{CS6}}$) output for relevant areas • DRAM/synchronous DRAM refresh functions — Programmable refresh interval — Supports CAS-before-RAS refresh mode and self-refresh mode • DRAM/synchronous DRAM burst access function • Big endian or little endian mode can be set
Direct memory access controller (DMAC)	• 4-channel physical address DMA controller • Transfer data size: 8, 16, 32, or 64 bits, or 32 bytes • Address modes: — 1-bus-cycle single address mode — 2-bus-cycle dual address mode • Transfer requests: External, on-chip module, or auto-requests • Bus modes: Cycle-steal or burst mode • Supports on-demand data transfer
Timer unit (TMU)	• 3-channel auto-reload 32-bit timer • Input capture function • Choice of seven counter input clocks
Realtime clock (RTC)	• On-chip clock and calendar functions • Built-in 32 kHz crystal oscillator with maximum 1/256 second resolution (cycle interrupts)

Table 1.1 SH7750 Features (cont)

Item	Features
Serial communication interface (SCI, SCIF)	• Two full-duplex communication channels (SCI, SCIF) • Channel 1 (SCI): — Choice of asynchronous mode or synchronous mode — Supports smart card interface • Channel 2 (SCIF): — Supports asynchronous mode — Separate 16-byte FIFOs provided for transmitter and receiver
Packages	• 256-pin BGA, 208-pin QFP

1.2 Block Diagram

Figure 1.1 shows an internal block diagram of the SH7750.



CCN:	Cache and TLB controller	UTLB:	Unified TLB (translation lookaside buffer)
BSC:	Bus state controller	RTC:	Realtime clock
CPG:	Clock pulse generator	SCI:	Serial communication interface
DMAC:	Direct memory access controller	SCIF:	Serial communication interface with FIFO
FPU:	Floating-point unit	TMU:	Timer unit
INTC:	Interrupt controller	UBC:	User break controller
ITLB:	Instruction TLB (translation lookaside buffer)		

Figure 1.1 Block Diagram of SH7750 Functions

Section 2 Programming Model

2.1 Data Formats

The data formats handled by the SH7750 are shown in figure 2.1.

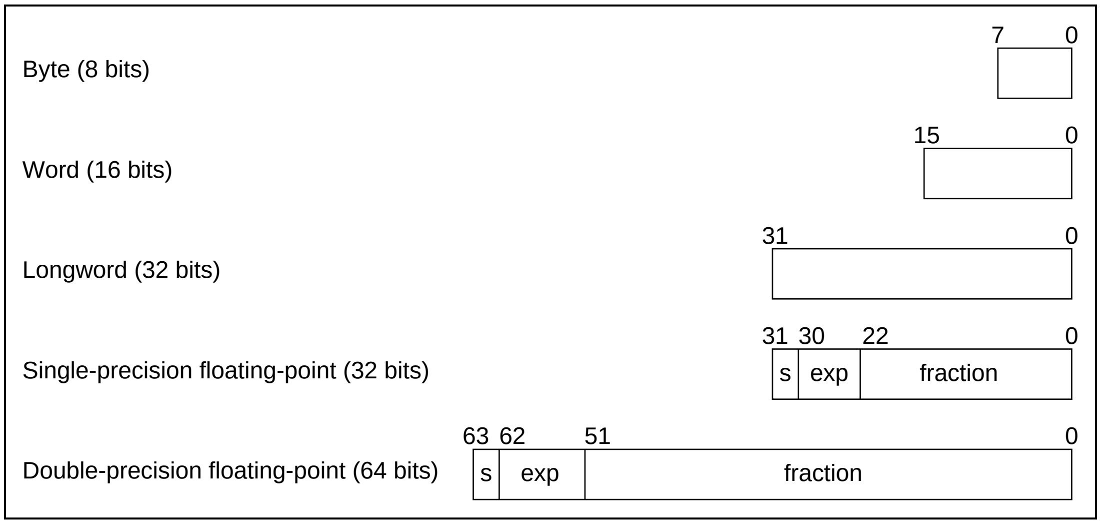


Figure 2.1 Data Formats

2.2 Register Configuration

2.2.1 Privileged Mode and Banks

Processor Modes: The SH7750 has two processor modes, user mode and privileged mode. The SH7750 normally operates in user mode, and switches to privileged mode when an exception occurs or an interrupt is accepted. There are four kinds of registers—general registers, system registers, control registers, and floating-point registers—and the registers that can be accessed differ in the two processor modes.

General Registers: There are 16 general registers, designated R0 to R15. General registers R0 to R7 are banked registers which are switched by a processor mode change.

In privileged mode, the register bank bit (RB) in the status register (SR) defines which banked register set is accessed as general 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 (that is, when bank 1 is selected), the 16 registers comprising bank 1 general registers R0_BANK1 to R7_BANK1 and non-banked general registers R8 to R15 can be accessed as general registers R0 to R15. In this case, the eight registers comprising bank 0 general registers R0_BANK0 to R7_BANK0 are accessed by the LDC/STC instructions. When the RB bit is 0 (that is, when bank 0 is selected), the 16 registers comprising bank 0 general registers R0_BANK0 to R7_BANK0 and non-banked general registers R8 to R15 can be accessed as general registers R0 to R15. In this case, the eight registers comprising bank 1 general registers R0_BANK1 to R7_BANK1 are accessed by the LDC/STC instructions.

In user mode, the 16 registers comprising bank 0 general registers R0_BANK0 to R7_BANK0 and non-banked general registers R8 to R15 can be accessed as general registers R0 to R15. The eight registers comprising bank 1 general registers R0_BANK1 to R7_BANK1 cannot be accessed.

Control Registers: Control registers comprise the global base register (GBR) and status register (SR), which can be accessed in both processor modes, and the saved status register (SSR), saved program counter (SPC), vector base register (VBR), saved general register 15 (SGR), and debug base register (DBR), which can only be accessed in privileged mode. Some bits of the status register (such as the RB bit) can only be accessed in privileged mode.

System Registers: System registers comprise the multiply-and-accumulate registers (MACH/MACL), the procedure register (PR), the program counter (PC), the floating-point status/control register (FPSCR), and the floating-point communication register (FPUL). Access to these registers does not depend on the processor mode.

Floating-Point Registers: There are thirty-two floating-point registers, FR0–FR15 and XF0–XF15. FR0–FR15 and XF0–XF15 can be assigned to either of two banks (FPR0_BANK0–FPR15_BANK0 or FPR0_BANK1–FPR15_BANK1).

FR0–FR15 can be used as the eight registers DR0/2/4/6/8/10/12/14 (double-precision floating-point registers, or pair registers) or the four registers FV0/4/8/12 (register vectors), while XF0–XF15 can be used as the eight registers XD0/2/4/6/8/10/12/14 (register pairs) or register matrix XMTRX.

Register values after a reset are shown in table 2.1.

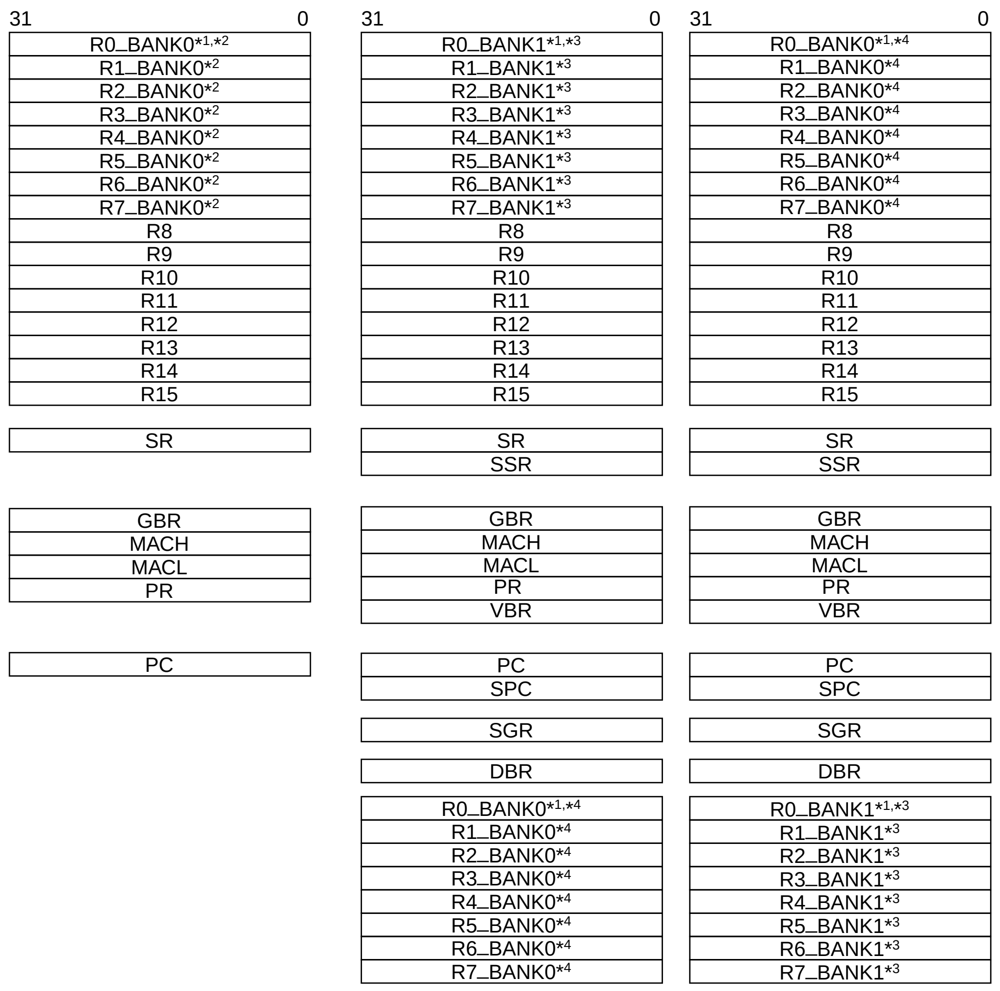
Table 2.1 Initial Register Values

Type	Registers	Initial Value*
General registers	R0_BANK0–R7_BANK0, R0_BANK1–R7_BANK1, R8–R15	Undefined
Control registers	SR	MD bit = 1, RB bit = 1, BL bit = 1, FD bit = 0, I3–I0 = 1111 (H'F), reserved bits = 0, others undefined
	GBR, SSR, SPC, SGR, DBR	Undefined
	VBR	H'00000000
System registers	MACH, MACL, PR, FPUL	Undefined
	PC	H'A0000000
	FPSCR	H'00040001
Floating-point registers	FR0–FR15, XF0–XF15	Undefined

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

The register configuration in each processor is shown in figure 2.2.

Switching between user mode and privileged mode is controlled by the processor mode bit (MD) in the status register.



(a) Register configuration in user mode

(b) Register configuration in privileged mode (RB = 1)

(c) Register configuration in privileged mode (RB = 0)

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

Figure 2.2 CPU Register Configuration in Each Processor Mode

2.2.2 General Registers

Figure 2.3 shows the relationship between the processor modes and general registers. The SH7750 has twenty-four 32-bit general registers (R0_BANK0–R7_BANK0, R0_BANK1–R7_BANK1, and R8–R15). However, only 16 of these can be accessed as general registers R0–R15 in one processor mode. The SH7750 has two processor modes, user mode and privileged mode, in which R0–R7 are assigned as shown below.

- R0_BANK0–R7_BANK0

  In user mode (SR.MD = 0), R0–R7 are always assigned to R0_BANK0–R7_BANK0.

  In privileged mode (SR.MD = 1), R0–R7 are assigned to R0_BANK0–R7_BANK0 only when SR.RB = 0.
- R0_BANK1–R7_BANK1

  In user mode, R0_BANK1–R7_BANK1 cannot be accessed.

  In privileged mode, R0–R7 are assigned to R0_BANK1–R7_BANK1 only when SR.RB = 1.

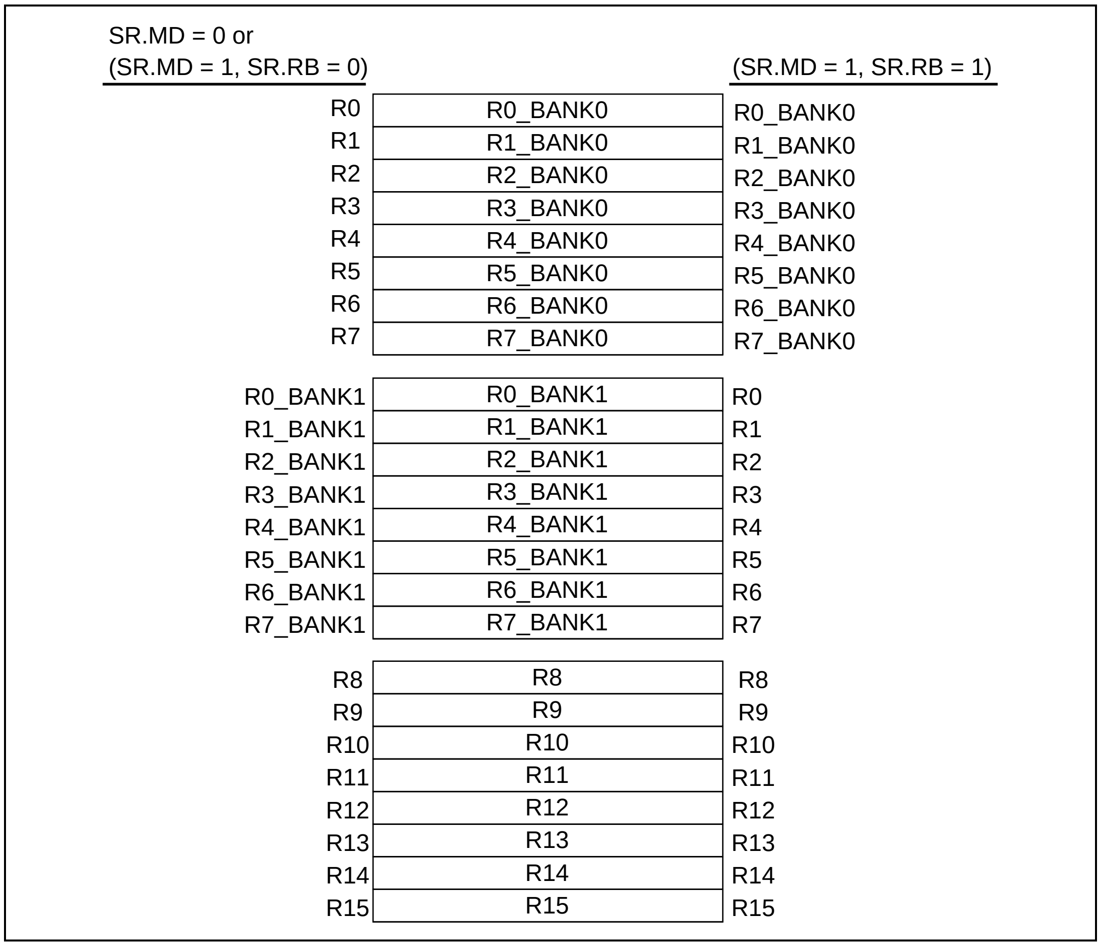


Figure 2.3 General Registers

Programming Note: As the user's R0–R7 are assigned to R0_BANK0–R7_BANK0, and after an exception or interrupt R0–R7 are assigned to R0_BANK1–R7_BANK1, it is not necessary for the interrupt handler to save and restore the user's R0–R7 (R0_BANK0–R7_BANK0).

After a reset, the values of R0_BANK0–R7_BANK0, R0_BANK1–R7_BANK1, and R8–R15 are undefined.

2.2.3 Floating-Point Registers

Figure 2.4 shows the floating-point registers. There are thirty-two 32-bit floating-point registers, divided into two banks (FPR0_BANK0–FPR15_BANK0 and FPR0_BANK1–FPR15_BANK1). These 32 registers are referenced as FR0–FR15, DR0/2/4/6/8/10/12/14, FV0/4/8/12, XF0–XF15, XD0/2/4/6/8/10/12/14, or XMTRX. The correspondence between FPRn_BANKi and the reference name is determined by the FR bit in FPSCR (see figure 2.4).

- Floating-point registers, FPRn_BANKi (32 registers)

  FPR0_BANK0, FPR1_BANK0, FPR2_BANK0, FPR3_BANK0, FPR4_BANK0, FPR5_BANK0, FPR6_BANK0, FPR7_BANK0, FPR8_BANK0, FPR9_BANK0, FPR10_BANK0, FPR11_BANK0, FPR12_BANK0, FPR13_BANK0, FPR14_BANK0, FPR15_BANK0

  FPR0_BANK1, FPR1_BANK1, FPR2_BANK1, FPR3_BANK1, FPR4_BANK1, FPR5_BANK1, FPR6_BANK1, FPR7_BANK1, FPR8_BANK1, FPR9_BANK1, FPR10_BANK1, FPR11_BANK1, FPR12_BANK1, FPR13_BANK1, FPR14_BANK1, FPR15_BANK1
- Single-precision floating-point registers, FRi (16 registers)

  When FPSCR.FR = 0, FR0–FR15 are assigned to FPR0_BANK0–FPR15_BANK0.

  When FPSCR.FR = 1, FR0–FR15 are assigned to FPR0_BANK1–FPR15_BANK1.
- Double-precision floating-point registers or single-precision floating-point register pairs, DRi (8 registers): A DR register comprises two FR registers.

  DR0 = {FR0, FR1}, DR2 = {FR2, FR3}, DR4 = {FR4, FR5}, DR6 = {FR6, FR7}, DR8 = {FR8, FR9}, DR10 = {FR10, FR11}, DR12 = {FR12, FR13}, DR14 = {FR14, FR15}
- Single-precision floating-point vector registers, FVi (4 registers): An FV register comprises four FR registers

  FV0 = {FR0, FR1, FR2, FR3}, FV4 = {FR4, FR5, FR6, FR7}, FV8 = {FR8, FR9, FR10, FR11}, FV12 = {FR12, FR13, FR14, FR15}
- Single-precision floating-point extended registers, XFi (16 registers)

  When FPSCR.FR = 0, XF0–XF15 are assigned to FPR0_BANK1–FPR15_BANK1.

  When FPSCR.FR = 1, XF0–XF15 are assigned to FPR0_BANK0–FPR15_BANK0.
- Single-precision floating-point extended register pairs, XDi (8 registers): An XD register comprises two XF registers

  XD0 = {XF0, XF1}, XD2 = {XF2, XF3}, XD4 = {XF4, XF5}, XD6 = {XF6, XF7}, XD8 = {XF8, XF9}, XD10 = {XF10, XF11}, XD12 = {XF12, XF13}, XD14 = {XF14, XF15}
- Single-precision floating-point extended register matrix, XMTRX: XMTRX comprises all 16 XF registers

$$\mathrm{XMTRX} = \begin{bmatrix} \mathrm{XF0} & \mathrm{XF4} & \mathrm{XF8} & \mathrm{XF12} \\ \mathrm{XF1} & \mathrm{XF5} & \mathrm{XF9} & \mathrm{XF13} \\ \mathrm{XF2} & \mathrm{XF6} & \mathrm{XF10} & \mathrm{XF14} \\ \mathrm{XF3} & \mathrm{XF7} & \mathrm{XF11} & \mathrm{XF15} \end{bmatrix}$$

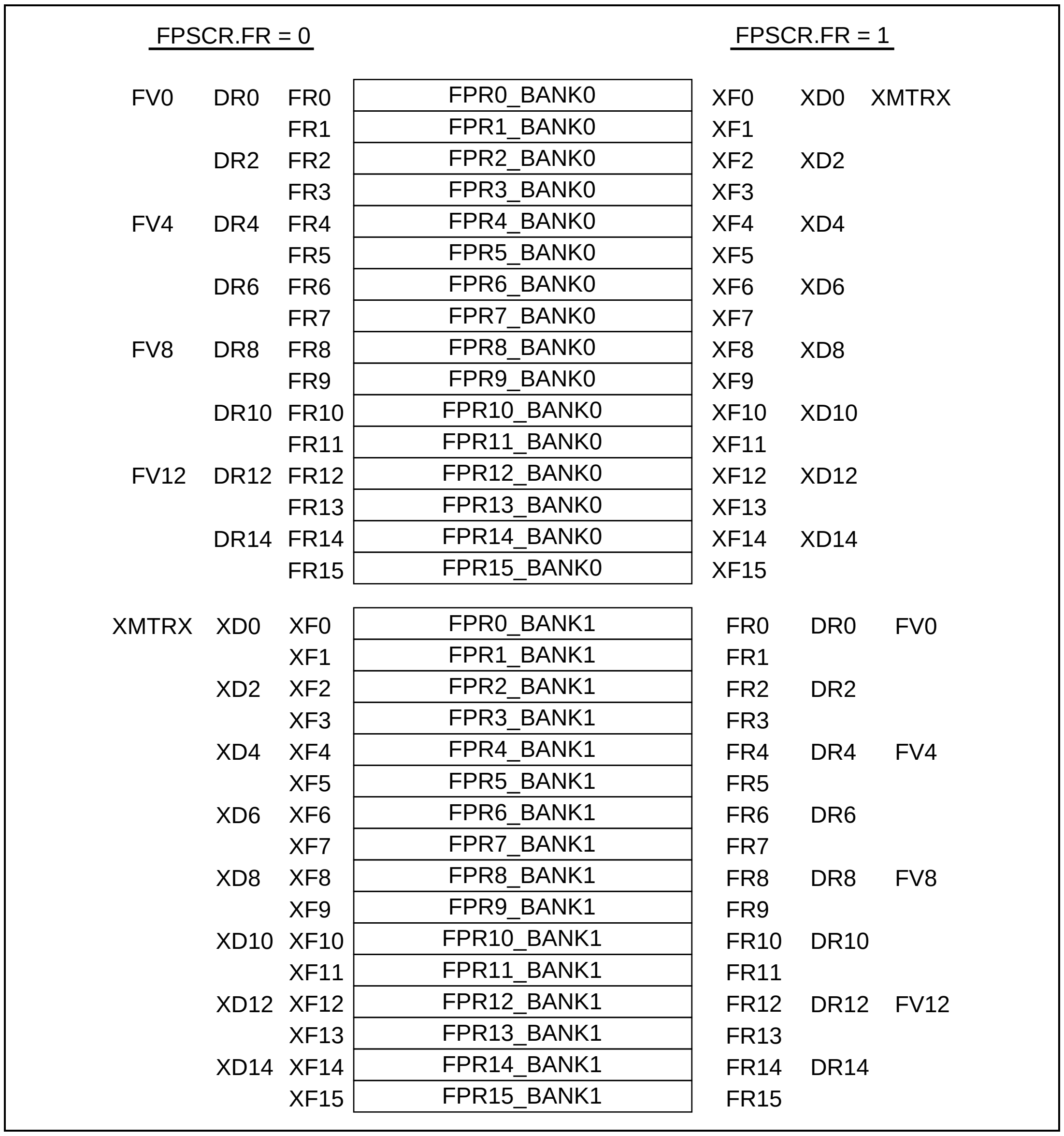

FPSCR.FR = 0
FPSCR.FR = 1
FV0 DR0 FR0 FPR0_BANK0 XF0 XD0 XMTRX
FR1 FPR1_BANK0 XF1
DR2 FR2 FPR2_BANK0 XF2 XD2
FR3 FPR3_BANK0 XF3
FV4 DR4 FR4 FPR4_BANK0 XF4 XD4
FR5 FPR5_BANK0 XF5
DR6 FR6 FPR6_BANK0 XF6 XD6
FR7 FPR7_BANK0 XF7
FV8 DR8 FR8 FPR8_BANK0 XF8 XD8
FR9 FPR9_BANK0 XF9
DR10 FR10 FPR10_BANK0 XF10 XD10
FR11 FPR11_BANK0 XF11
FV12 DR12 FR12 FPR12_BANK0 XF12 XD12
FR13 FPR13_BANK0 XF13
DR14 FR14 FPR14_BANK0 XF14 XD14
FR15 FPR15_BANK0 XF15
XMTRX XD0 XF0 FPR0_BANK1 FR0 DR0 FV0
XF1 FPR1_BANK1 FR1
XD2 XF2 FPR2_BANK1 FR2 DR2
XF3 FPR3_BANK1 FR3
XD4 XF4 FPR4_BANK1 FR4 DR4 FV4
XF5 FPR5_BANK1 FR5
XD6 XF6 FPR6_BANK1 FR6 DR6
XF7 FPR7_BANK1 FR7
XD8 XF8 FPR8_BANK1 FR8 DR8 FV8
XF9 FPR9_BANK1 FR9
XD10 XF10 FPR10_BANK1 FR10 DR10
XF11 FPR11_BANK1 FR11
XD12 XF12 FPR12_BANK1 FR12 DR12 FV12
XF13 FPR13_BANK1 FR13
XD14 XF14 FPR14_BANK1 FR14 DR14
XF15 FPR15_BANK1 FR15

Figure 2.4 Floating-Point Registers

Programming Note: After a reset, the values of FPR0_BANK0–FPR15_BANK0 and FPR0_BANK1–FPR15_BANK1 are undefined.

2.2.4 Control Registers

Status register, SR (32 bits, privilege protection, initial value = 0111 0000 0000 0000 0000 00XX 1111 00XX)

31	30	29	28	27 16	15	14 10	9	8	7 4	3 2	1	0
—	MD	RB	BL	—	FD	—	M	Q	IMASK	—	S	T

Note: —: Reserved. These bits are always read as 0, and should only be written with 0.
X: Undefined

- MD: Processor mode

  MD = 0: User mode (some instructions cannot be executed, and some resources cannot be accessed)

  MD = 1: Privileged mode
- RB: General register bank specifier in privileged mode (set to 1 by a reset, exception, or interrupt)

  RB = 0: R0_BANK0–R7_BANK0 are accessed as general registers R0–R7. (R0_BANK1–R7_BANK1 can be accessed using LDC/STC R0_BANK–R7_BANK instructions.)

  RB = 1: R0_BANK1–R7_BANK1 are accessed as general registers R0–R7. (R0_BANK0–R7_BANK0 can be accessed using LDC/STC R0_BANK–R7_BANK instructions.)
- BL: Exception/interrupt block bit (set to 1 by a reset, exception, or interrupt)

  BL = 1: Interrupt requests are masked. If a general exception other than a user break occurs while BL = 1, the processor switches to the reset state.
- FD: FPU disable bit (cleared to 0 by a reset)

  FD = 1: An FPU instruction causes a general FPU disable exception, and if the FPU instruction is in a delay slot, a slot FPU disable exception is generated. (FPU instructions: H'F*** instructions, LDC(.L)/STS(.L) instructions for FPUL/FPSCR)
- M, Q: Used by the DIV0S, DIV0U, and DIV1 instructions.
- IMASK: Interrupt mask level

  External interrupts of a lower level than IMASK are masked.
- S: Specifies a saturation operation for a MAC instruction.
- T: True/false condition or carry/borrow bit

Saved status register, SSR (32 bits, privilege protection, initial value undefined): The current contents of SR are saved to SSR in the event of an exception or interrupt.

Saved program counter, SPC (32 bits, privilege protection, initial value undefined): The address of an instruction at which an interrupt or exception occurs is saved to SPC.

Global base register, GBR (32 bits, initial value undefined): GBR is referenced as the base address in a GBR-referencing MOV instruction.

Vector base register, VBR (32 bits, privilege protection, initial value = H'0000 0000): VBR is referenced as the branch destination base address in the event of an exception or interrupt. For details, see section 5, Exceptions.

Saved general register 15, SGR (32 bits, privilege protection, initial value undefined): The contents of R15 are saved to SGR in the event of an exception or interrupt.

Debug base register, DBR (32 bits, privilege protection, initial value undefined): When the user break debug function is enabled (BRCR.UBDE = 1), DBR is referenced as the user break handler branch destination address instead of VBR.

2.2.5 System Registers

Multiply-and-accumulate register high, MACH (32 bits, initial value undefined)
Multiply-and-accumulate register low, MACL (32 bits, initial value undefined)
MACH/MACL is used for the added value in a MAC instruction, and to store a MAC instruction or MUL operation result.

Procedure register, PR (32 bits, initial value undefined): The return address is stored in PR in a subroutine call using a BSR, BSRF, or JSR instruction, and PR is referenced by the subroutine return instruction (RTS).

Program counter, PC (32 bits, initial value = H'A000 0000): PC indicates the instruction fetch address.

Floating-point status/control register, FPSCR (32 bits, initial value = H'0004 0001)

31 22	21	20	19	18	17 12	11 7	6 2	1 0
—	FR	SZ	PR	DN	Cause	Enable	Flag	RM

Note: —: Reserved. These bits are always read as 0, and should only be written with 0.

- FR: Floating-point register bank

  FR = 0: FPR0_BANK0–FPR15_BANK0 are assigned to FR0–FR15; FPR0_BANK1–FPR15_BANK1 are assigned to XF0–XF15.

  FR = 1: FPR0_BANK0–FPR15_BANK0 are assigned to XF0–XF15; FPR0_BANK1–FPR15_BANK1 are assigned to FR0–FR15.
- SZ: Transfer size mode

  SZ = 0: The data size of the FMOV instruction is 32 bits.

  SZ = 1: The data size of the FMOV instruction is a 32-bit register pair (64 bits).
- PR: Precision mode

  PR = 0: Floating-point instructions are executed as single-precision operations.

  PR = 1: Floating-point instructions are executed as double-precision operations (the result of instructions for which double-precision is not supported is undefined).

  Do not set SZ and PR to 1 simultaneously; this setting is reserved.

  [SZ, PR = 11]: Reserved (FPU operation instruction is undefined.)
- DN: Denormalization mode

  DN = 0: A denormalized number is treated as such.

  DN = 1: A denormalized number is treated as zero.

		FPU Error (E)	**Invalid Operation (V)**	**Division by Zero (Z)**	**Overflow (O)**	**Underflow (U)**	**Inexact (I)**
Cause	FPU exception cause field	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12
Enable	FPU exception enable field	None	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7
Flag	FPU exception flag field	None	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2

When an FPU operation instruction is executed, the cause field is cleared to zero first. When the next FPU exception is requested, the corresponding bits in the cause field and flag field are set to 1. The flag field holds the status of the exception generated after the field was last cleared.

- RM: Rounding mode

  RM = 00: Round to Nearest

  RM = 01: Round to Zero

  RM = 10: Reserved

  RM = 11: Reserved
- Bits 22 to 31: Reserved

Floating-point communication register, FPUL (32 bits, initial value undefined): Data transfer between FPU registers and CPU registers is carried out via the FPUL register.

Programming Note: When SZ = 1 and big endian mode is selected, FMOV can be used for double-precision floating-point load or store operations. In little endian mode, two 32-bit data size moves must be executed, with SZ = 0, to load or store a double-precision floating-point number.

2.3 Memory-Mapped Registers

Appendix A shows the control registers mapped to memory. The control registers are double-mapped to the following two memory areas. All registers have two addresses.

H'1F00 0000–H'1FFF FFFF
H'FF00 0000–H'FFFF FFFF

These two areas are used as follows.

- H'1F00 0000–H'1FFF FFFF

  This area must be accessed in address translation mode using the TLB. Since external memory is defined as a 29-bit address space in the SH7750 architecture, the TLB’s physical page numbers do not cover a 32-bit address space. In address translation, the page numbers of this area can be set in the corresponding field of the TLB by accessing a memory-mapped register. The page numbers of this area should be used as the actual page numbers set in the TLB. When address translation is not performed, the operation of accesses to this area is undefined.
- H'FF00 0000–H'FFFF FFFF

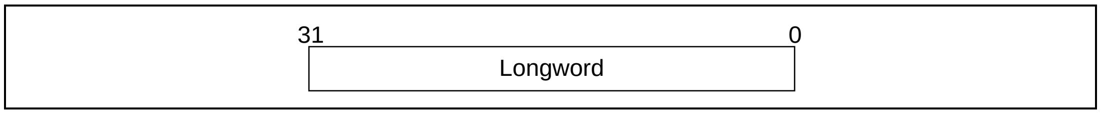
  This area must be accessed without address translation.

  Do not access undefined locations in either area The operation of an access to an undefined location is undefined. Also, memory-mapped registers must be accessed using a fixed data size. The operation of an access using an invalid data size is undefined.

Programming Note: Access to area H'FF00 0000–H'FFFF FFFF in user mode will cause an address error. Memory-mapped registers can be referenced in user mode by means of access that involves address translation.

2.4 Data Format in Registers

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



2.5 Data Formats in Memory

Memory data formats are classified into bytes, words, and longwords. Memory can be accessed in 8-bit byte, 16-bit word, or 32-bit longword form. A memory operand less than 32 bits in length is sign-extended before being loaded into a register.

A word operand must be accessed starting from a word boundary (even address of a 2-byte unit: address 2n), and a longword operand starting from a longword boundary (even address of a 4-byte unit: address 4n). An address error will result if this rule is not observed. A byte operand can be accessed from any address.

Big endian or little endian byte order can be selected for the data format. The endian should be set with the MD5 external pin in a power-on reset. Big endian is selected when the MD5 pin is low, and little endian when high. The endian cannot be changed dynamically. Bit positions are numbered left to right from most-significant to least-significant. Thus, in a 32-bit longword, the leftmost bit, bit 31, is the most significant bit and the rightmost bit, bit 0, is the least significant bit.

The data format in memory is shown in figure 2.5. In little endian mode, data written as byte-size (8 bits) should be read as byte size, and data written as word-size (16 bits) should be read as word size.

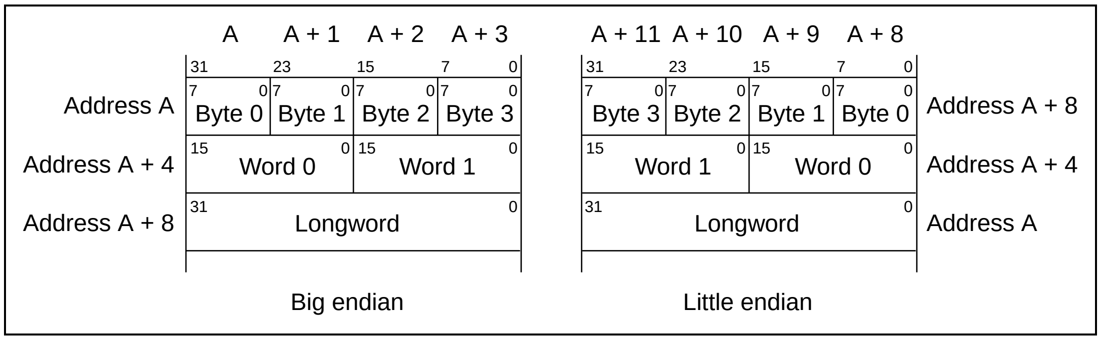


Figure 2.5 Data Formats In Memory

Note: The SH7750 does not support endian conversion for the 64-bit data format. Therefore, if double-precision floating-point format (64-bit) access is performed in little endian mode, the upper and lower 32 bits will be reversed.

2.6 Processor States

The SH7750 has five processor states: the reset state, exception-handling state, bus-released state, program execution state, and power-down state.

Reset State: In this state the CPU is reset. The reset state is entered when the $\overline{\text{RESET}}$ pin goes low. The CPU enters the power-on reset state if the $\overline{\text{MRESET}}$ pin is high, and the manual reset state if the $\overline{\text{MRESET}}$ pin is low. For more information on resets, see section 5, Exceptions.

In the power-on reset state, the internal state of the CPU and the on-chip peripheral module registers are initialized. In the manual reset state, the internal state of the CPU and registers of on-chip peripheral modules other than the bus state controller (BSC) are initialized. Since the bus state controller (BSC) is not initialized in the manual reset state, refreshing operations continue. Refer to the register configurations in the relevant sections for further details.

Exception-Handling State: This is a transient state during which the CPU's processor state flow is altered by a reset, general exception, or interrupt exception handling source.

In the case of a reset, the CPU branches to address H'A000 0000 and starts executing the user-coded exception handling program.

In the case of a general exception or interrupt, the program counter (PC) contents are saved in the saved program counter (SPC), the status register (SR) contents are saved in the saved status register (SSR), and the R15 contents are saved in saved general register 15 (SGR). The CPU branches to the start address of the user-coded exception service routine found from the sum of the contents of the vector base address and the vector offset. See section 5, Exceptions, for more information on resets, general exceptions, and interrupts.

Program Execution State: In this state the CPU executes program instructions in sequence.

Power-Down State: In the power-down state, CPU operation halts and power consumption is reduced. The power-down state is entered by executing a SLEEP instruction. There are two modes in the power-down state: sleep mode and standby mode. For details, see section 9, Power-Down Modes.

Bus-Released State: In this state the CPU has released the bus to a device that requested it.

Transitions between the states are shown in figure 2.6.

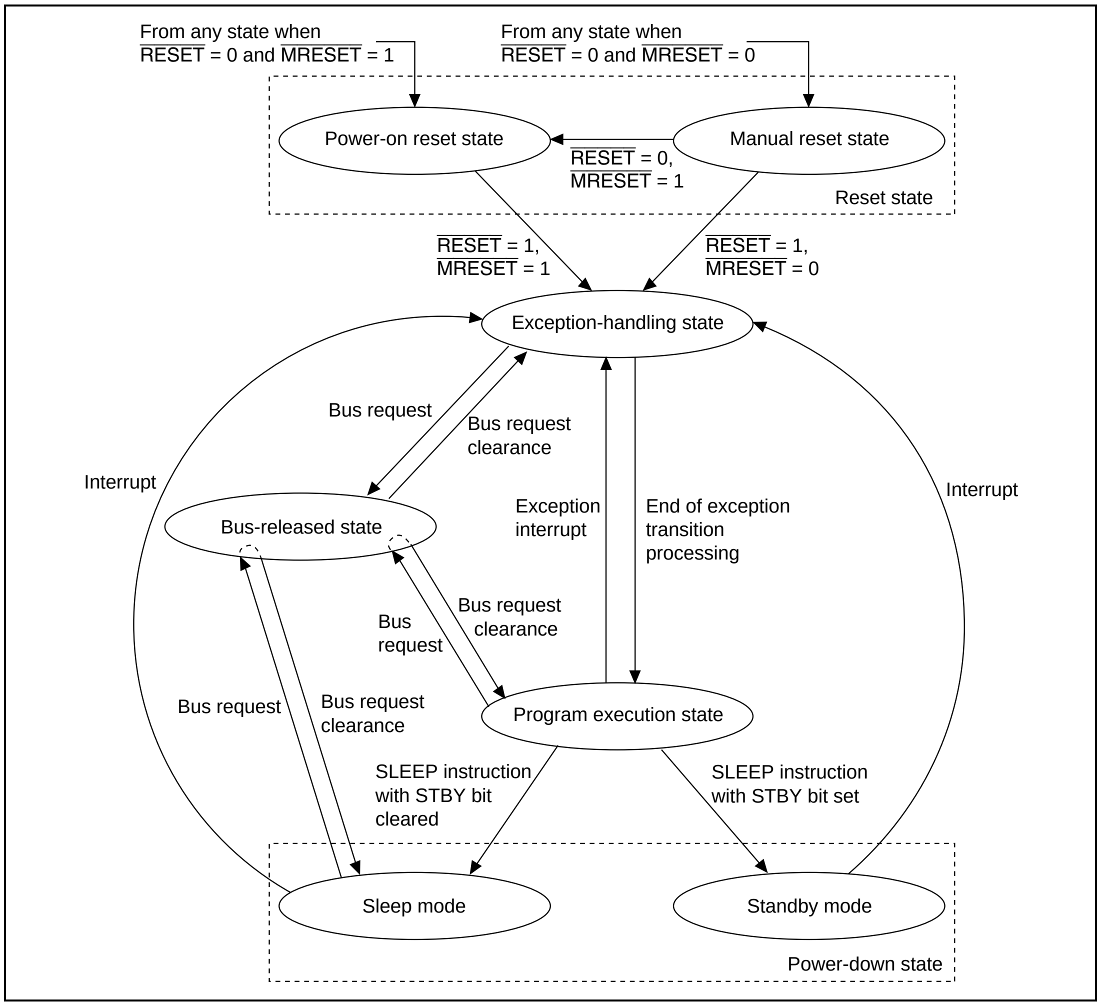

From any state when $\overline{\text{RESET}}$ = 0 and $\overline{\text{MRESET}}$ = 1
From any state when $\overline{\text{RESET}}$ = 0 and $\overline{\text{MRESET}}$ = 0
Power-on reset state
Manual reset state
$\overline{\text{RESET}}$ = 0, $\overline{\text{MRESET}}$ = 1
Reset state
$\overline{\text{RESET}}$ = 1, $\overline{\text{MRESET}}$ = 1
$\overline{\text{RESET}}$ = 1, $\overline{\text{MRESET}}$ = 0
Exception-handling state
Bus request
Bus request clearance
Interrupt
Interrupt
Bus-released state
Exception interrupt
End of exception transition processing
Bus request clearance
Bus request
Bus request
Bus request clearance
Program execution state
SLEEP instruction with STBY bit cleared
SLEEP instruction with STBY bit set
Sleep mode
Standby mode
Power-down state

Figure 2.6 Processor State Transitions

2.7 Processor Modes

There are two processor modes: user mode and privileged mode. The processor mode is determined by the processor mode bit (MD) in the status register (SR). User mode is selected when the MD bit is cleared to 0, and privileged mode when the MD bit is set to 1. When the reset state or exception state is entered, the MD bit is set to 1. When exception handling ends, the MD bit is cleared to 0 and user mode is entered. There are certain registers and bits which can only be accessed in privileged mode.

Section 3 Memory Management Unit (MMU)

3.1 Overview

3.1.1 Features

The SH7750 can handle 29-bit external memory space from an 8-bit address space identifier and 32-bit logical (virtual) address space. Address translation from virtual address to physical address is performed using the memory management unit (MMU) built into the SH7750. The MMU performs high-speed address translation by caching user-created address translation table information in an address translation buffer (translation lookaside buffer: TLB). The SH7750 has four instruction TLB (ITLB) entries and 64 unified TLB (UTLB) entries. UTLB copies are stored in the ITLB by hardware. A paging system is used for address translation, with support for four page sizes (1, 4, and 64 kbytes, and 1 Mbyte). It is possible to set the virtual address space access right and implement storage protection independently for privileged mode and user mode.

3.1.2 Role of the MMU

The MMU was conceived as a means of making efficient use of physical memory. As shown in figure 3.1, when a process is smaller in size than the physical memory, the entire process can be mapped onto physical memory, but if the process increases in size to the point where it does not fit into physical memory, it becomes necessary to divide the process into smaller parts, and map the parts requiring execution onto physical memory on an ad hoc basis ((1)). Having this mapping onto physical memory executed consciously by the process itself imposes a heavy burden on the process. The virtual memory system was devised as a means of handling all physical memory mapping to reduce this burden ((2)). With a virtual memory system, the size of the available virtual memory is much larger than the actual physical memory, and processes are mapped onto this virtual memory. Thus processes only have to consider their operation in virtual memory, and mapping from virtual memory to physical memory is handled by the MMU. The MMU is normally managed by the OS, and physical memory switching is carried out so as to enable the virtual memory required by a task to be mapped smoothly onto physical memory. Physical memory switching is performed via secondary storage, etc.

The virtual memory system that came into being in this way works to best effect in a time sharing system (TSS) that allows a number of processes to run simultaneously ((3)). Running a number of processes in a TSS did not increase efficiency since each process had to take account of physical memory mapping. Efficiency is improved and the load on each process reduced by the use of a virtual memory system ((4)). In this system, virtual memory is allocated to each process. The task of the MMU is to map a number of virtual memory areas onto physical memory in an efficient manner. It is also provided with memory protection functions to prevent a 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 happen that the translation information has not been recorded in the MMU, or the virtual memory of a different process is accessed by mistake. In such cases, 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 be implemented by software alone, having address translation performed by software each time a process accessed physical memory would be very inefficient. For this reason, a buffer for address translation (the translation lookaside buffer: TLB) is provided in hardware, and frequently used address translation information is placed here. The TLB can be described as a cache for address translation information. However, unlike a cache, if address translation fails—that is, if an exception occurs—switching of the address translation information is normally performed by software. Thus memory management can be performed in a flexible manner by software.

There are two methods by which the MMU can perform mapping from virtual memory to physical memory: the paging method, using fixed-length address translation, and the segment method, using variable-length address translation. With the paging method, the unit of translation is a fixed-size address space called a page (usually from 1 to 64 kbytes in size).

In the following descriptions, the address space in virtual memory in the SH7750 is referred to as virtual address space, and the address space in physical memory as physical address space.

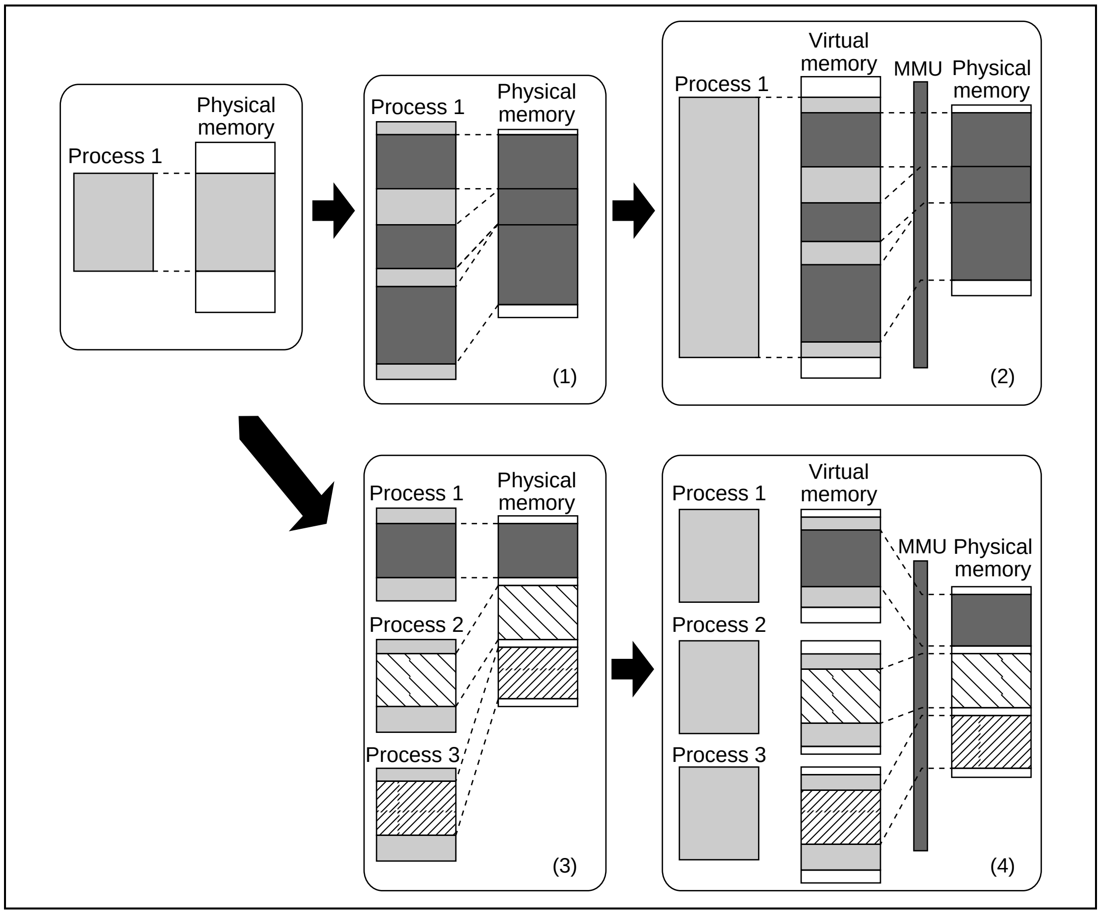


Figure 3.1 Role of the MMU

3.1.3 Register Configuration

The MMU registers are shown in table 3.1.

Table 3.1 MMU Registers

Name	Abbrevia-tion	R/W	Initial Value*[1]	P4 Address*[2]	Area 7 Address*[2]	Access Size
Page table entry high register	PTEH	R/W	Undefined	H'FF00 0000	H'1F00 0000	32
Page table entry low register	PTEL	R/W	Undefined	H'FF00 0004	H'1F00 0004	32
Page table entry assistance register	PTEA	R/W	Undefined	H'FF00 0034	H'1F00 0034	32
Translation table base register	TTB	R/W	Undefined	H'FF00 0008	H'1F00 0008	32
TLB exception address register	TEA	R/W	Undefined	H'FF00 000C	H'1F00 000C	32
MMU control register	MMUCR	R/W	H'0000 0000	H'FF00 0010	H'1F00 0010	32

Notes: 1. The initial value is the value after a power-on reset or manual reset.

2. This is the address when using the virtual/physical address space P4 area. When making an access from physical address space area 7 using the TLB, the upper 3 bits of the address are ignored.

3.1.4 Caution

Operation is not guaranteed if an area designated as a reserved area in this manual is accessed.

3.2 Register Descriptions

There are six MMU-related registers.

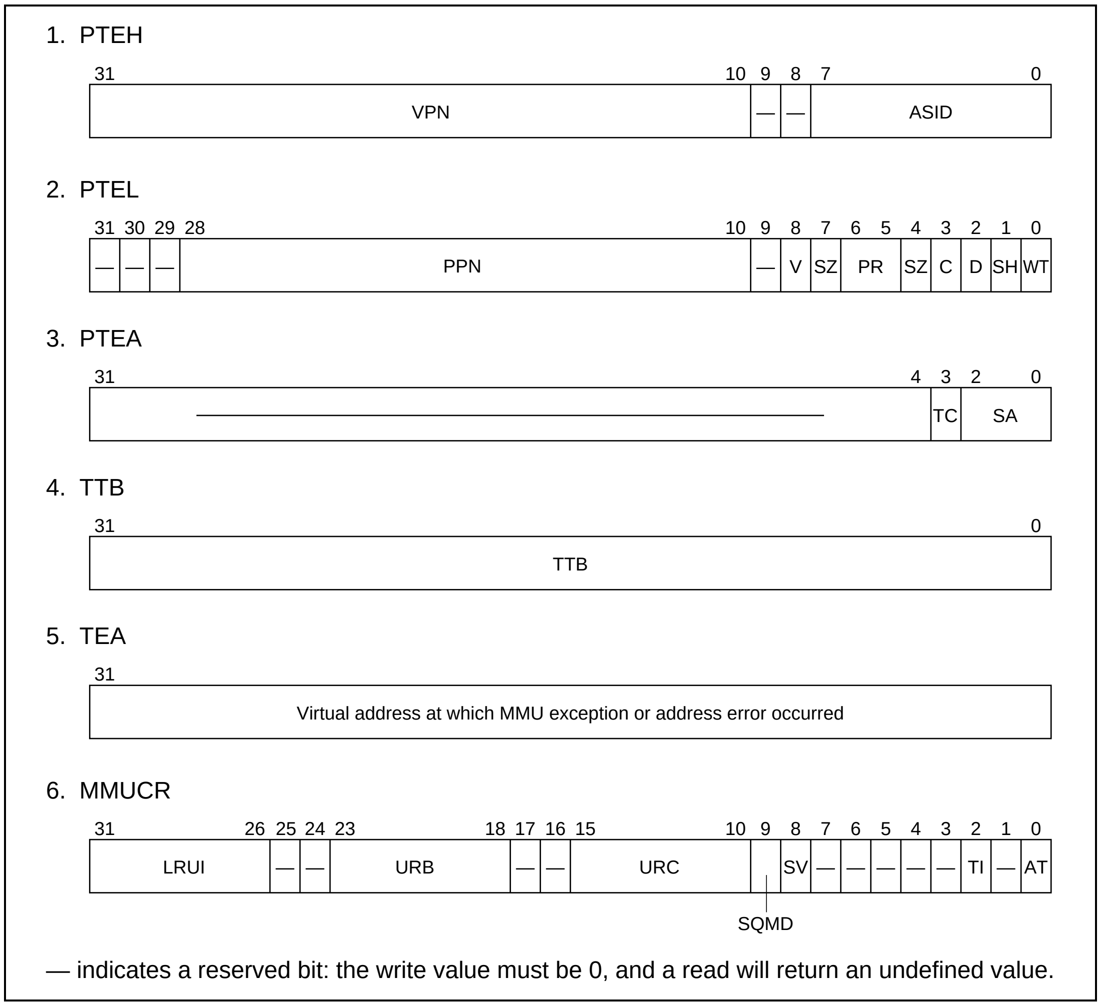


Figure 3.2 MMU-Related Registers

1. **Page table entry high register (PTEH):** Longword access to PTEH can be performed from H'FF00 0000 in the P4 area and H'1F00 0000 in area 7. PTEH consists of the virtual page number (VPN) and address space identifier (ASID). When an MMU exception or address error exception occurs, the VPN of the virtual address at which the exception occurred is set in the VPN field by hardware. VPN varies according to the page size, but the VPN set by hardware when an exception occurs consists of the upper 22 bits of the virtual address which caused the exception. VPN setting can also be carried out by software. The number of the currently executing process is set in the ASID field by software. ASID is not updated by hardware. VPN and ASID are recorded in the UTLB by means of the LDLTB instruction.

2. **Page table entry low register (PTEL):** Longword access to PTEL can be performed from H'FF00 0004 in the P4 area and H'1F00 0004 in area 7. PTEL is used to hold the physical page number and page management information to be recorded in the UTLB by means of the LDTLB instruction. The contents of this register are not changed unless a software directive is issued.

3. **Page table entry assistance register (PTEA):** Longword access to PTEA can be performed from H'FF00 0034 in the P4 area and H'1F00 0034 in area 7. PTEL is used to store assistance bits for PCMCIA access to the UTLB by means of the LDTLB instruction. The contents of this register are not changed unless a software directive is issued.

4. **Translation table base register (TTB):** Longword access to TTB can be performed from H'FF00 0008 in the P4 area and H'1F00 0008 in area 7. TTB is used, for example, to hold the base address of the currently used page table. The contents of TTB are not changed unless a software directive is issued. This register can be freely used by software.

5. **TLB exception address register (TEA):** Longword access to TEA can be performed from H'FF00 000C in the P4 area and H'1F00 000C in area 7. After an MMU exception or address error exception occurs, the virtual address at which the exception occurred is set in TEA by hardware. The contents of this register can be changed by software.

6. **MMU control register (MMUCR):** MMUCR contains the following bits:

LRUI:	Least recently used ITLB
URB:	UTLB replace boundary
URC:	UTLB replace counter
SQMD:	Store queue mode bit
SV:	Single virtual mode bit
TI:	TLB invalidate
AT:	Address translation bit

Longword access to MMUCR can be performed from H'FF00 0010 in the P4 area and H'1F00 0010 in area 7. The individual bits perform MMU settings as shown below. Therefore, MMUCR rewriting should be performed by a program in the P1 or P2 area. After MMUCR is updated, an instruction that performs data access to the P0, P3, U0, or store queue area should be located at least four instructions after the MMUCR update instruction. Also, a branch instruction to the P0, P3, or U0 area should be located at least eight instructions after the MMUCR update instruction. MMUCR contents can be changed by software. The LRUI bits and URC bits may also be updated by hardware.

- LRUI: The LRU (least recently used) method is used to decide the ITLB entry to be replaced in the event of an ITLB miss. The entry to be purged from the ITLB can be confirmed using the LRUI bits. LRUI is updated by means of the algorithm shown below. A dash in this table means that updating is not performed.

	LRUI					
	[5]	**[4]**	**[3]**	**[2]**	**[1]**	**[0]**
When ITLB entry 0 is used	0	0	0	—	—	—
When ITLB entry 1 is used	1	—	—	0	0	—
When ITLB entry 2 is used	—	1	—	1	—	0
When ITLB entry 3 is used	—	—	1	—	1	1
Other than the above	—	—	—	—	—	—

When the LRUI bit settings are as shown below, the corresponding ITLB entry is updated by an ITLB miss. An asterisk in this table means “don’t care”.

	LRUI					
	[5]	**[4]**	**[3]**	**[2]**	**[1]**	**[0]**
ITLB entry 0 is updated	1	1	1	*	*	*
ITLB entry 1 is updated	0	*	*	1	1	*
ITLB entry 2 is updated	*	0	*	0	*	1
ITLB entry 3 is updated	*	*	0	*	0	0
Other than the above	Setting prohibited					

Ensure that values for which “Setting prohibited” is indicated in the above table are not set at the discretion of software. After a power-on or manual reset the LRUI bits are initialized to 0, and therefore a prohibited setting is never made by a hardware update.

- URB: Bits that indicate the UTLB entry boundary at which replacement is to be performed. Valid only when URB > 0.

- URC: Random counter for indicating the UTLB entry for which replacement is to be performed with an LDTLB instruction. URC is incremented each time the UTLB is accessed. When URB > 0, URC is reset to 0 when the condition URC = URB occurs. Also note that, if a value is written to URC by software which results in the condition URC > URB, incrementing is first performed in excess of URB until URC = H'3F. URC is not incremented by an LDTLB instruction.

- SQMD: Store queue mode bit. Specifies the right of access to the store queues.
  0: User/privileged access possible
  1: Privileged access possible (address error exception in case of user access)
- SV: Bit that switches between single virtual memory mode and multiple virtual memory mode.
  0: Multiple virtual memory mode
  1: Single virtual memory mode
  When this bit is changed, ensure that 1 is also written to the TI bit.
- TI: Writing 1 to this bit invalidates (clears to 0) all valid UTLB/ITLB bits. This bit always returns 0 when read.
- AT: Specifies MMU enabling or disabling.
  0: MMU disabled
  1: MMU enabled
  MMU exceptions are not generated when the AT bit is 0. In the case of software that does not use the MMU, therefore, the AT bit should be cleared to 0.

3.3 Memory Space

3.3.1 Physical Memory Space

The SH7750 supports a 32-bit physical memory space, and can access a 4-Gbyte address space. When the MMUCR.AT bit is cleared to 0 and the MMU is disabled, the address space is this physical memory space. The physical memory space is divided into a number of areas, as shown in figure 3.3. The physical memory space is permanently mapped onto 29-bit external memory space; this correspondence can be implemented by ignoring the upper 3 bits of the physical memory space addresses. In privileged mode, the 4-Gbyte space from the P0 area to the P4 area can be accessed. In user mode, a 2-Gbyte space in the U0 area can be accessed. Accessing the P1 to P4 areas (except the store queue area) in user mode will cause an address error.

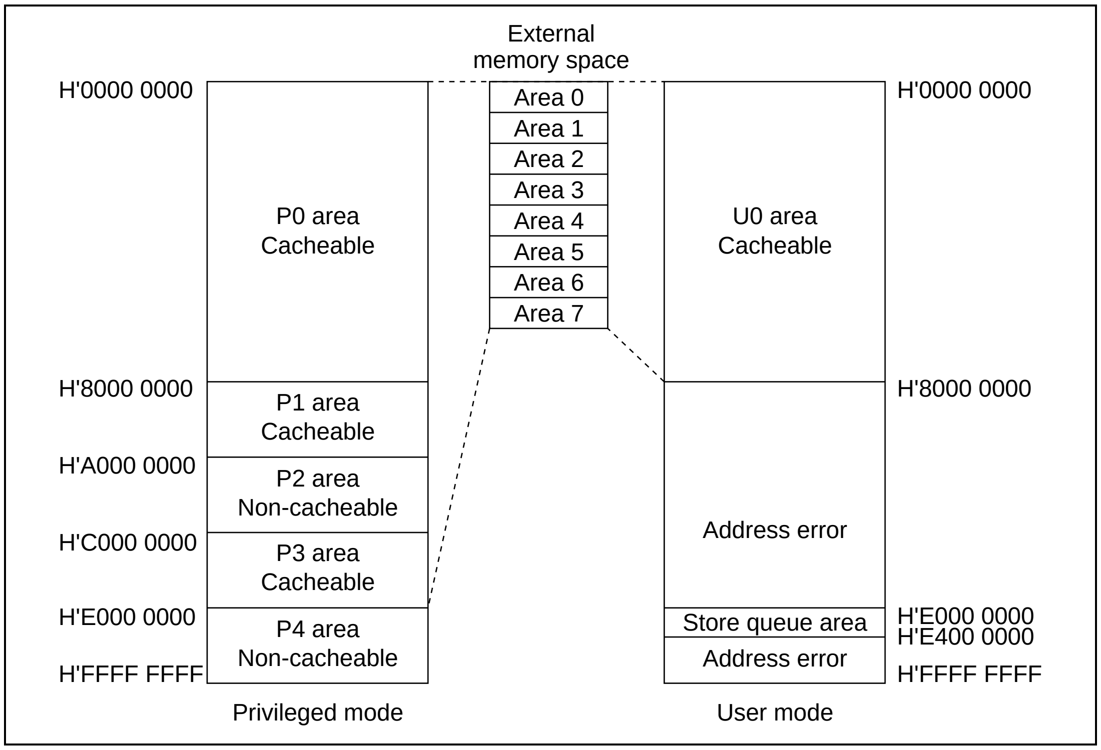


Figure 3.3 Physical Memory Space (MMUCR.AT = 0)

P0, P1, P3, U0 Areas: The P0, P1, P3, and U0 areas can be accessed using the cache. Whether or not the cache is used is determined by the cache control register (CCR). When the cache is used, with the exception of the P1 area, switching between the copy-back method and the write-through method for write accesses is specified by the CCR.WT bit. For the P1 area, switching is specified by the CCR.CB bit. Zeroizing the upper 3 bits of an address in these areas gives the corresponding external memory space address. However, since area 7 in the external memory space is a reserved area, a reserved area also appears in these areas.

P2 Area: The P2 area cannot be accessed using the cache. In the P2 area, zeroizing the upper 3 bits of an address gives the corresponding external memory space address. However, since area 7 in the external memory space is a reserved area, a reserved area also appears in this area.

P4 Area: The P4 area is mapped onto SH7750 on-chip I/O channels. This area cannot be accessed using the cache. The P4 area is shown in detail in figure 3.4.

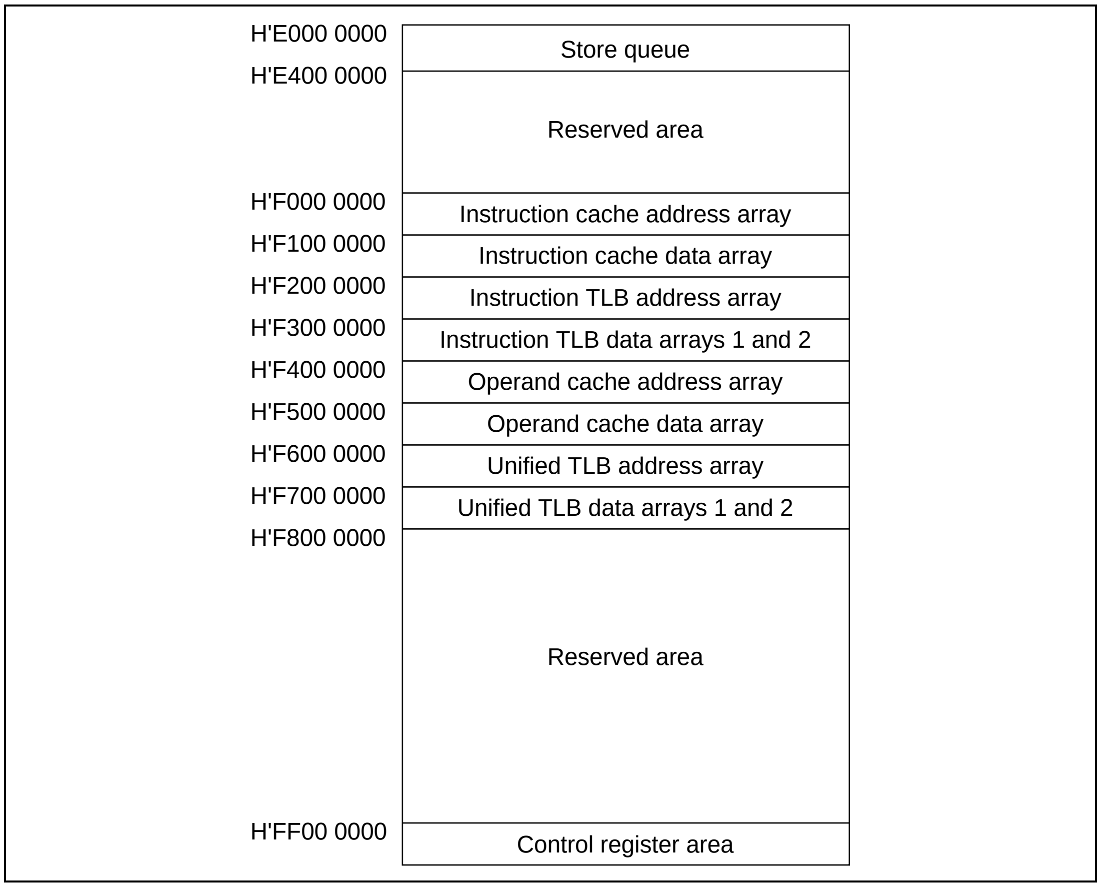


Figure 3.4 P4 Area

The area from H'E000 0000 to H'E3FF FFFF comprises addresses for accessing the store queues (SQs). When the MMU is disabled (MMUCR.AT = 0), the SQ access right is specified by the MMUCR.SQMD bit. For details, see section 4.6, Store Queues.

The area from H'F000 0000 to H'F0FF FFFF is used for direct access to the instruction cache address array. For details, see section 4.5.1, IC Address Array.

The area from H'F100 0000 to H'F1FF FFFF is used for direct access to the instruction cache data array. For details, see section 4.5.2, IC Data Array.

The area from H'F200 0000 to H'F2FF FFFF is used for direct access to the instruction TLB address array. For details, see section 3.7.1, ITLB Address Array.

The area from H'F300 0000 to H'F3FF FFFF is used for direct access to instruction TLB data arrays 1 and 2. For details, see sections 3.7.2, ITLB Data Array 1, and 3.7.3, ITLB Data Array 2.

The area from H'F400 0000 to H'F4FF FFFF is used for direct access to the operand cache address array. For details, see section 4.5.3, OC Address Array.

The area from H'F500 0000 to H'F5FF FFFF is used for direct access to the operand cache data array. For details, see section 4.5.4, OC Data Array.

The area from H'F600 0000 to H'F6FF FFFF is used for direct access to the unified TLB address array. For details, see section 3.7.4, UTLB Address Array.

The area from H'F700 0000 to H'F7FF FFFF is used for direct access to unified TLB data arrays 1 and 2. For details, see sections 3.7.5, UTLB Data Array 1, and 3.7.6, UTLB Data Array 2.

The area from H'FF00 0000 to H'FFFF FFFF is the on-chip peripheral module control register area.

3.3.2 External Memory Space

The SH7750 supports a 29-bit external memory space. The external memory space is divided into eight areas as shown in figure 3.5. Areas 0 to 6 relate to memory, such as SRAM, synchronous DRAM, DRAM, and PCMCIA. Area 7 is a reserved area. For details, see section 13, Bus State Controller (BSC).

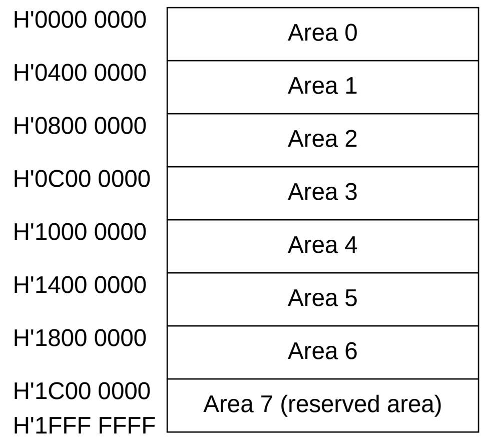


Figure 3.5 External Memory Space

3.3.3 Virtual Memory Space

Setting the MMUCR.AT bit to 1 enables the P0, P3, and U0 areas of the physical memory space in the SH7750 to be mapped onto any external memory space in 1-, 4-, or 64-kbyte, or 1-Mbyte, page units. By using an 8-bit address space identifier, the P0, U0, P3, and store queue areas can be increased to a maximum of 256. This is called the virtual memory space. Mapping from virtual memory space to 29-bit external memory space is carried out using the TLB. Only when area 7 in external memory space is accessed using virtual memory space, addresses H'1F00 0000 to H'1FFF FFFF of area 7 are not designated as a reserved area, but are equivalent to the P4 area control register area in the physical memory space. Virtual memory space is illustrated in figure 3.6.

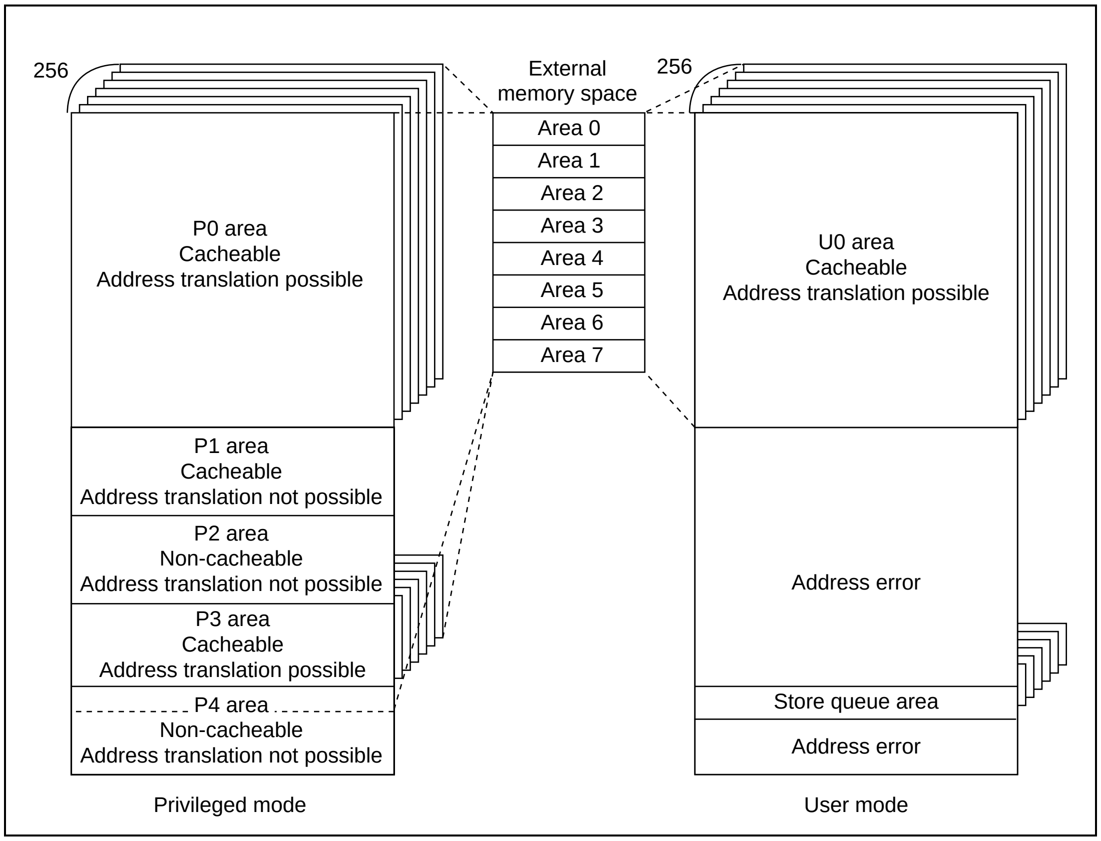


Figure 3.6 Virtual Memory Space (MMUCR.AT = 1)

P0, P3, U0 Areas: The P0 area (excluding addresses H'7C00 0000 to H'7FFF FFFF), P3 area, and U0 area allow access using the cache and address translation using the TLB. These areas can be mapped onto any external memory space in 1-, 4-, or 64-kbyte, or 1-Mbyte, page units. When CCR is in the cache-enabled state and the TLB enable bit (C bit) is 1, accesses can be performed using the cache. In write accesses to the cache, switching between the copy-back method and the write-through method is indicated by the TLB write-through bit (WT bit), and is specified in page units.

Only when the P0, P3, and U0 areas are mapped onto external memory space by means of the TLB, addresses H'1F00 0000 to H'1FFF FFFF of area 7 in external memory space are allocated to the control register area. This enables on-chip peripheral module control registers to be accessed from the U0 area in user mode. In this case, the C bit for the corresponding page must be cleared to 0.

In the cache enabled state, when areas P0, P3, and U0 are mapped onto the PCMCIA space by means of TLB, it is necessary either to specify 1 for the WT bit or to specify 0 for the C bit; it is not possible to use copy-back mode cache for the PCMCIA space.

P1, P2, P4 Areas: Address translation using the TLB cannot be performed for the P1, P2, or P4 area (except for the store queue area). Accesses to these areas are the same as for physical memory space. The store queue area can be mapped onto any external memory space by the MMU. However, operation in the case of an exception differs from that for normal P0, U0, and P3 spaces. For details, see section 4.6, Store Queues.

3.3.4 On-Chip RAM Space

In the SH7750, half (8 kbytes) of the instruction cache (16 kbytes) can be used as on-chip RAM. This can be done by changing the CCR settings.

When the operand cache is used as on-chip RAM (CCR.ORA = 1), P0 area addresses H'7C00 0000 to H'7FFF FFFF are an on-chip RAM area. Data accesses (byte/word/longword/quadword) can be used in this area. This area can only be used in RAM mode.

3.3.5 Address Translation

When the MMU is used, the virtual address space is divided into units called pages, and translation to physical addresses is carried out in these page units. The address translation table in external memory contains the physical addresses corresponding to virtual addresses and additional information such as memory protection codes. Fast address translation is achieved by caching the contents of the address translation table located in external memory into the TLB. In the SH7750, basically, the ITLB is used for instruction accesses and the UTLB for data accesses. In the event of an access to an area other than the P4 area, the accessed virtual address is translated to a physical address. If the virtual address belongs to the P1 or P2 area, the physical address is uniquely determined without accessing the TLB. If the virtual address belongs to the

P0, U0, or P3 area, the TLB is searched using the virtual address, and if the virtual address is recorded in the TLB, a TLB hit is made and the corresponding physical address is read from the TLB. If the accessed virtual address is not recorded in the TLB, a TLB miss exception is generated and processing switches to the TLB miss exception routine. In the TLB miss exception routine, the address translation table in external memory is searched, and the corresponding physical address and page management information are recorded in the TLB. After the return from the exception handling routine, the instruction which caused the TLB miss exception is re-executed.

3.3.6 Single Virtual Memory Mode and Multiple Virtual Memory Mode

There are two virtual memory systems, single virtual memory and multiple virtual memory, either of which can be selected with the MMUCR.SV bit. In the single virtual memory system, a number of processes run simultaneously, using virtual address space on an exclusive basis, and the physical address corresponding to a particular virtual address is uniquely determined. In the multiple virtual memory system, a number of processes run while sharing the virtual address space, and a particular virtual address may be translated into different physical addresses depending on the process. The only difference between the single virtual memory and multiple virtual memory systems in terms of operation is in the TLB address comparison method (see section 3.4.3, Address Translation Method).

3.3.7 Address Space Identifier (ASID)

In multiple virtual memory mode, the 8-bit address space identifier (ASID) is used to distinguish between processes running simultaneously while sharing the virtual address space. Software can set the ASID of the currently executing process in PTEH in the MMU. The TLB does not have to be purged when processes are switched by means of ASID.

In single virtual memory mode, ASID is used to provide memory protection for processes running simultaneously while using the virtual memory space on an exclusive basis.

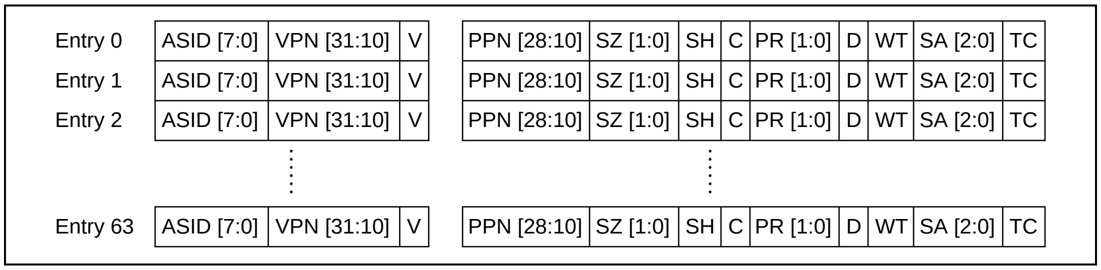
3.4 TLB Functions

3.4.1 Unified TLB (UTLB) Configuration

The unified TLB (UTLB) is so called because of its use for the following two purposes:

1. To translate a virtual address to a physical address in a data access
2. As a table of address translation information to be recorded in the instruction TLB in the event of an ITLB miss

Information in the address translation table located in external memory is cached into the UTLB. The address translation table contains virtual page numbers and address space identifiers, and corresponding physical page numbers and page management information. Figure 3.7 shows the overall configuration of the UTLB. The UTLB consists of 64 fully-associative type entries. Figure 3.8 shows the relationship between the address format and page size.



Entry 0	ASID [7:0]	VPN [31:10]	V	PPN [28:10]	SZ [1:0]	SH	C	PR [1:0]	D	WT	SA [2:0]	TC
Entry 1	ASID [7:0]	VPN [31:10]	V	PPN [28:10]	SZ [1:0]	SH	C	PR [1:0]	D	WT	SA [2:0]	TC
Entry 2	ASID [7:0]	VPN [31:10]	V	PPN [28:10]	SZ [1:0]	SH	C	PR [1:0]	D	WT	SA [2:0]	TC
	⋮			⋮								
Entry 63	ASID [7:0]	VPN [31:10]	V	PPN [28:10]	SZ [1:0]	SH	C	PR [1:0]	D	WT	SA [2:0]	TC

Figure 3.7 UTLB Configuration

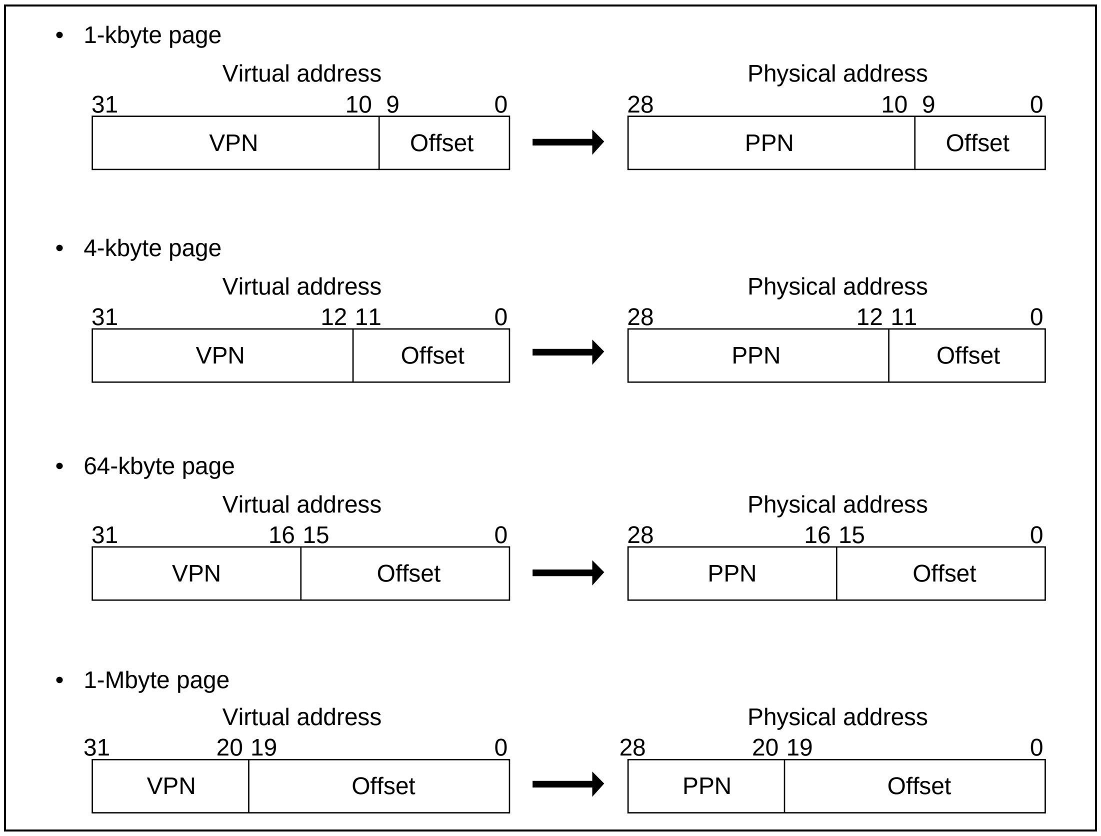


Figure 3.8 Relationship between Page Size and Address Format

- VPN: Virtual page number
  For 1-kbyte page: upper 22 bits of virtual address
  For 4-kbyte page: upper 20 bits of virtual address
  For 64-kbyte page: upper 16 bits of virtual address
  For 1-Mbyte page: upper 12 bits of virtual address

- 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, this identifier is compared with the ASID in PTEH when address comparison is performed.

- SH: Share status bit
  When 0, pages are not shared by processes.
  When 1, pages are shared by processes.

- SZ: Page size bits
  Specify the page size.
  00: 1-kbyte page
  01: 4-kbyte page
  10: 64-kbyte page
  11: 1-Mbyte page

- V: Validity bit
  Indicates whether the 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
  Upper 22 bits of the physical address.
  With a 1-kbyte page, PPN bits [28:10] are valid.
  With a 4-kbyte page, PPN bits [28:12] are valid.
  With a 64-kbyte page, PPN bits [28:16] are valid.
  With a 1-Mbyte page, PPN bits [28:20] are valid.
  The synonym problem must be taken into account when setting the PPN (see section 3.5.5, Avoiding Synonym Problems).

- PR: Protection key data
  2-bit data expressing the page access right as a code.
  00: Can be read only, in privileged mode
  01: Can be read and written in privileged mode
  10: Can be read only, in privileged or user mode
  11: Can be read and written in privileged mode or user mode

- C: Cacheability bit
  Indicates whether a page is cacheable.
  0: Not cacheable
  1: Cacheable
  When control register space is mapped, this bit must be cleared to 0.
  When performing PCMCIA space mapping in the cache enabled state, either clear this bit to 0 or set the WT bit to 1.

- D: Dirty bit

  Indicates whether a write has been performed to a page.

  0: Write has not been performed

  1: Write has been performed

- WT: Write-through bit

  Specifies the cache write mode.

  0: Copy-back mode

  1: Write-through mode

  When performing PCMCIA space mapping in the cache enabled state, either set this bit to 1 or clear the C bit to 0.

- SA: Space attribute bits

  Valid only when the page is mapped onto PCMCIA connected to area 5 or 6.

  000: Undefined

  001: Variable-size I/O space (base size according to $\overline{\text{IOIS16}}$ signal)

  010: 8-bit I/O space

  011: 16-bit I/O space

  100: 8-bit common memory space

  101: 16-bit common memory space

  110: 8-bit attribute memory space

  111: 16-bit attribute memory space

- TC: Timing control bit

  Used to select wait control register bits in the bus control unit for areas 5 and 6.

  0: WCR2 (A5W2–A5W0) and PCR (A5PCW1–A5PCW0, A5TED2–A5TED0, A5TEH2–A5TEH0) are used

  1: WCR2 (A6W2–A6W0) and PCR (A6PCW1–A6PCW0, A6TED2–A6TED0, A6TEH2–A6TEH0) are used

3.4.2 Instruction TLB (ITLB) Configuration

The ITLB is used to translate a virtual address to a physical address in an instruction access. Information in the address translation table located in the UTLB is cached into the ITLB. Figure 3.9 shows the overall configuration of the ITLB. The ITLB consists of 4 fully-associative type entries. The address translation information is almost the same as that in the UTLB, but with the following differences:

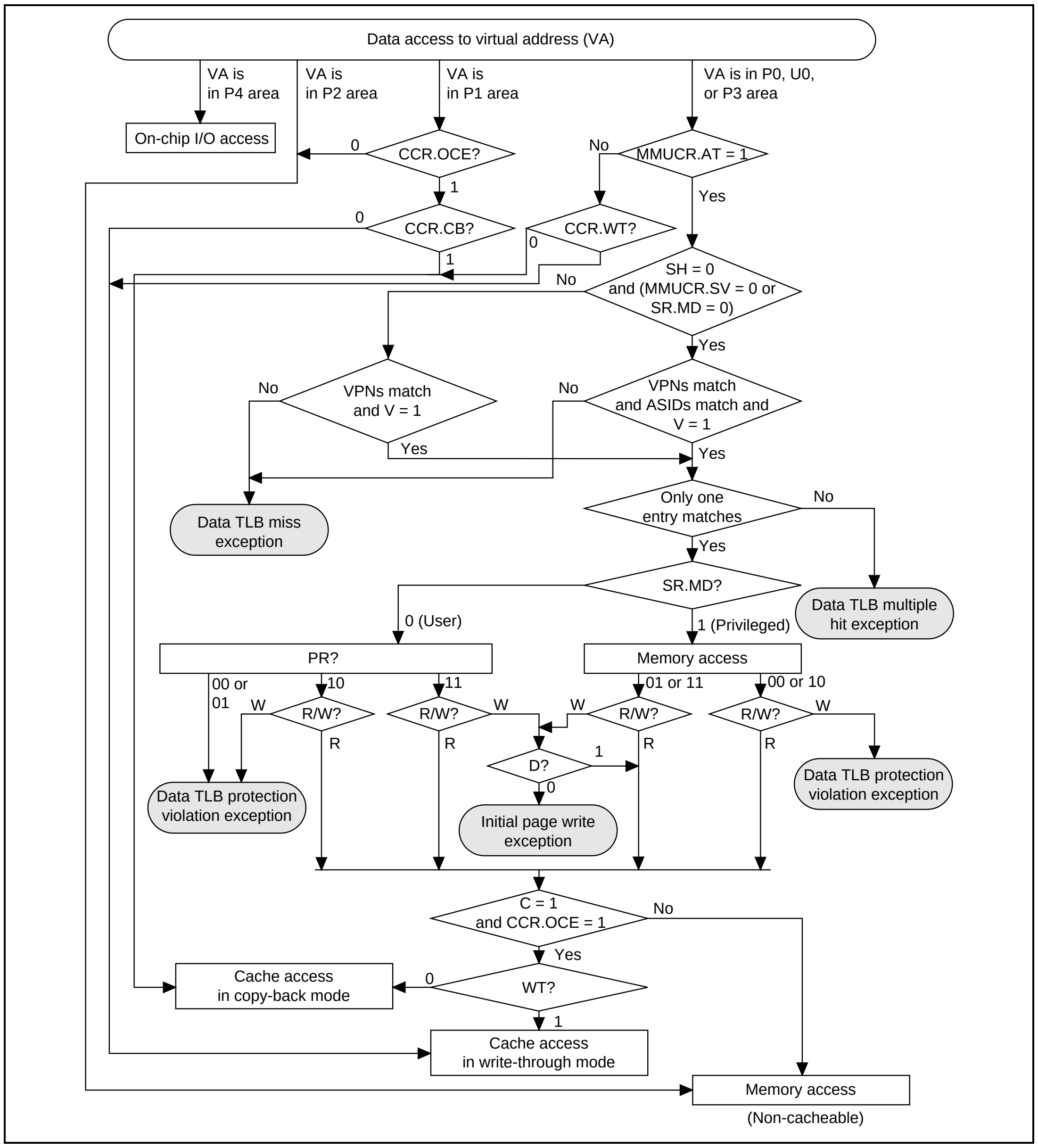
1. D and WT bits are not supported.
2. There is only one PR bit, corresponding to the upper of the PR bits in the UTLB.

Entry 0	ASID [7:0]	VPN [31:10]	V	PPN [28:10]	SZ [1:0]	SH	C	PR	SA [2:0]	TC
Entry 1	ASID [7:0]	VPN [31:10]	V	PPN [28:10]	SZ [1:0]	SH	C	PR	SA [2:0]	TC
Entry 2	ASID [7:0]	VPN [31:10]	V	PPN [28:10]	SZ [1:0]	SH	C	PR	SA [2:0]	TC
Entry 3	ASID [7:0]	VPN [31:10]	V	PPN [28:10]	SZ [1:0]	SH	C	PR	SA [2:0]	TC

Figure 3.9 ITLB Configuration

3.4.3 Address Translation Method

Figures 3.10 and 3.11 show flowcharts of memory accesses using the UTLB and ITLB.



Data access to virtual address (VA)
VA is in P4 area
VA is in P2 area
VA is in P1 area
VA is in P0, U0, or P3 area
On-chip I/O access
CCR.OCE?
0
1
No
MMUCR.AT = 1
Yes
CCR.CB?
CCR.WT?
0
1
0
SH = 0 and (MMUCR.SV = 0 or SR.MD = 0)
No
Yes
VPNs match and V = 1
No
Yes
VPNs match and ASIDs match and V = 1
No
Yes
Data TLB miss exception
Only one entry matches
No
Yes
SR.MD?
0 (User)
1 (Privileged)
Data TLB multiple hit exception
PR?
Memory access
00 or 01
10
11
01 or 11
00 or 10
R/W?
W
R
D?
1
0
Data TLB protection violation exception
Initial page write exception
Data TLB protection violation exception
C = 1 and CCR.OCE = 1
No
Yes
WT?
0
1
Cache access in copy-back mode
Cache access in write-through mode
Memory access
(Non-cacheable)

Figure 3.10 Flowchart of Memory Access Using UTLB

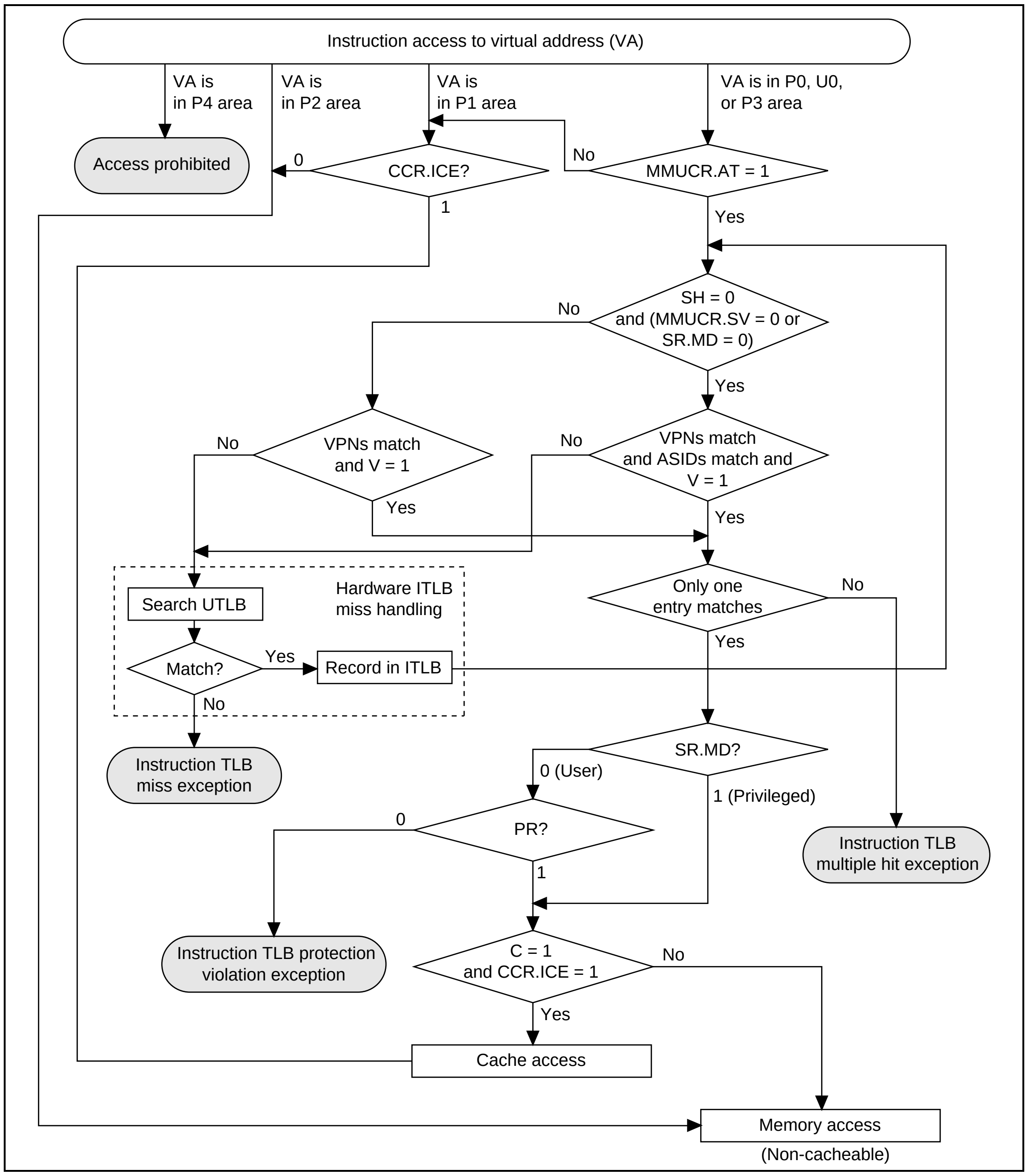


Figure 3.11 Flowchart of Memory Access Using ITLB

3.5 MMU Functions

3.5.1 MMU Hardware Management

The SH7750 supports the following MMU functions.

1. The MMU decodes the virtual address to be accessed by software, and performs address translation by controlling the UTLB/ITLB in accordance with the MMUCR settings.
2. The MMU determines the cache access status on the basis of the page management information read during address translation (C, WT, SA, and TC bits).
3. If address translation cannot be performed normally in a data access or instruction access, the MMU notifies software by means of an MMU exception.
4. If address translation information is not recorded in the ITLB in an instruction access, the MMU searches the UTLB, and if the necessary address translation information is recorded in the UTLB, the MMU copies this information into the ITLB in accordance with MMUCR.LRUI.

3.5.2 MMU Software Management

Software processing for the MMU consists of the following:

1. Setting of MMU-related registers. Some registers are also partially updated by hardware automatically.
2. Recording, deletion, and reading of TLB entries. There are two methods of recording UTLB entries: by using the LDTLB instruction, or by writing directly to the memory-mapped UTLB. ITLB entries can only be recorded by writing directly to the memory-mapped ITLB. For deleting or reading UTLB/ITLB entries, it is possible to access the memory-mapped UTLB/ITLB.
3. MMU exception handling. When an MMU exception occurs, processing is performed based on information set by hardware.

3.5.3 MMU Instruction (LDTLB)

A TLB load instruction (LDTLB) is provided for recording UTLB entries. When an LDTLB instruction is issued, the SH7750 copies the contents of PTEH, PTEL, and PTEA to the UTLB entry indicated by MMUCR.URC. ITLB entries are not updated by the LDTLB instruction, and therefore address translation information purged from the UTLB entry may still remain in the ITLB entry. As the LDTLB instruction changes address translation information, ensure that it is issued by a program in the P1 or P2 area. The operation of the LDTLB instruction is shown in figure 3.12.

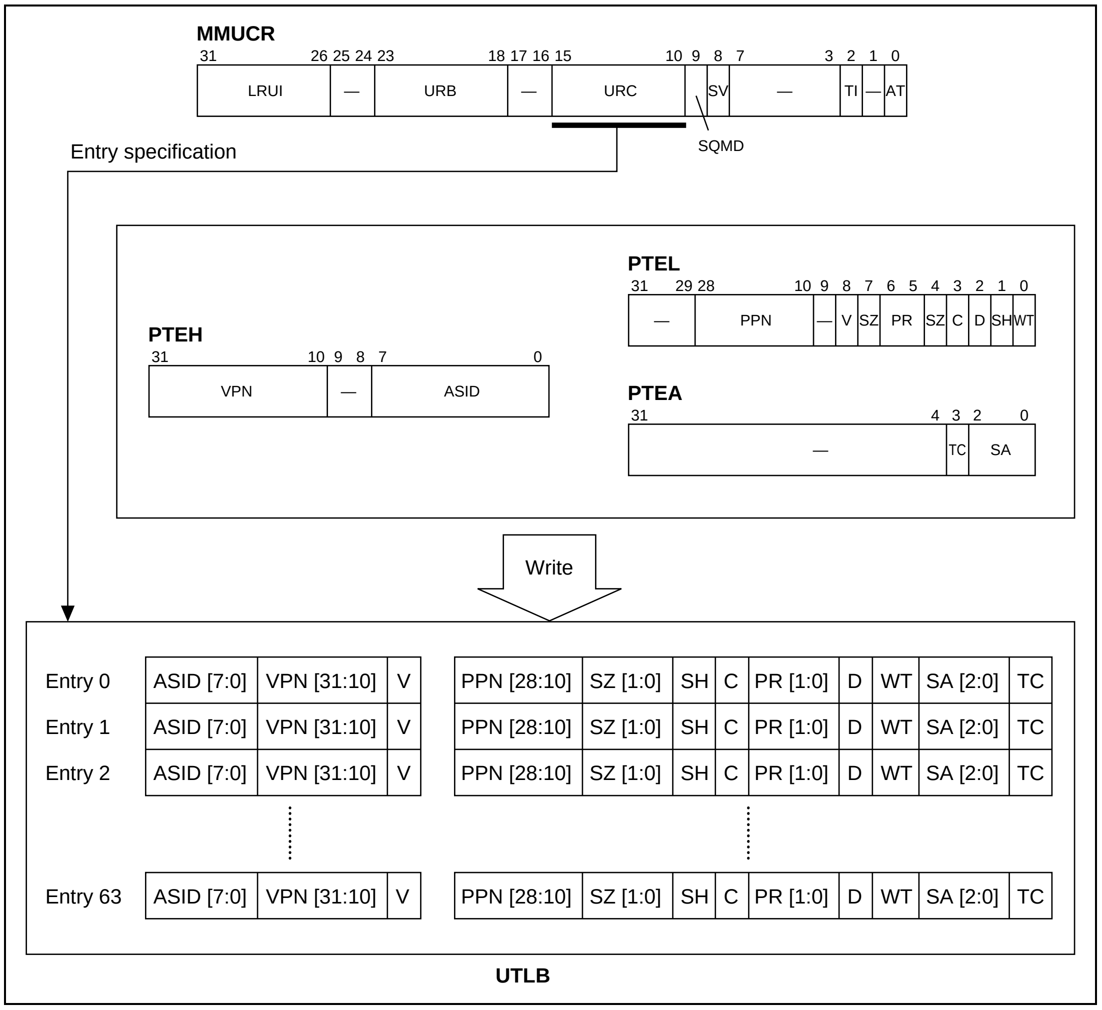


Figure 3.12 Operation of LDTLB Instruction

3.5.4 Hardware ITLB Miss Handling

In an instruction access, the SH7750 searches the ITLB. If it cannot find the necessary address translation information (i.e. in the event of an ITLB miss), the UTLB is searched by hardware, and if the necessary address translation information is present, it is recorded in the ITLB. This procedure is known as hardware ITLB miss handling. If the necessary address translation information is not found in the UTLB search, an instruction TLB miss exception is generated and processing passes to software.

3.5.5 Avoiding Synonym Problems

When 1- or 4-kbyte pages are recorded in TLB entries, a synonym problem may arise. The problem is that, when a number of virtual addresses are mapped onto a single physical address, the same physical address data is recorded in a number of cache entries, and it becomes impossible to guarantee data integrity. This problem does not occur with the instruction TLB or instruction cache . In the SH7750, entry specification is performed using bits [13:5] of the virtual address in order to achieve fast operand cache operation. However, bits [13:10] of the virtual address in the case of a 1-kbyte page, and bits [13:12] of the virtual address in the case of a 4-kbyte page, are subject to address translation. As a result, bits [13:10] of the physical address after translation may differ from bits [13:10] of the virtual address.

Consequently, the following restrictions apply to the recording of address translation information in UTLB entries.

1. When address translation information whereby a number of 1-kbyte page UTLB entries are translated into the same physical address is recorded in the UTLB, ensure that the VPN [13:10] values are the same.
2. When address translation information whereby a number of 4-kbyte page UTLB entries are translated into the same physical address is recorded in the UTLB, ensure that the VPN [13:12] values are the same.
3. Do not use 1-kbyte page UTLB entry physical addresses with UTLB entries of a different page size.
4. Do not use 4-kbyte page UTLB entry physical addresses with UTLB entries of a different page size.

The above restrictions apply only when performing accesses using the cache. When cache index mode is used, VPN [25] is used for the entry address instead of VPN [13], and therefore the above restrictions apply to VPN [25].

Note: When multiple items of address translation information use the same physical memory to provide for future SH Series expansion, ensure that the VPN [20:10] values are the same. Also, do not use the same physical address for address translation information of different page sizes.

3.6 MMU Exceptions

There are seven MMU exceptions: the instruction TLB multiple hit exception, instruction TLB miss exception, instruction TLB protection violation exception, data TLB multiple hit exception, data TLB miss exception, data TLB protection violation exception, and initial page write exception. Refer to figures 3.10 and 3.11 for the conditions under which each of these exceptions occurs.

3.6.1 Instruction TLB Multiple Hit Exception

An instruction TLB multiple hit exception occurs when more than one ITLB entry matches the virtual address to which an instruction access has been made. If multiple hits occur when the UTLB is searched by hardware in hardware ITLB miss handling, a data TLB multiple hit exception will result.

When an instruction TLB multiple hit exception occurs a reset is executed, and cache coherency is not guaranteed.

Hardware Processing: In the event of an instruction TLB multiple hit exception, hardware carries out the following processing:

1. Sets the virtual address at which the exception occurred in TEA.
2. Sets exception code H'140 in EXPEVT.
3. Branches to the reset handling routine (H'A000 0000).

Software Processing (Reset Routine): The ITLB entries which caused the multiple hit exception are checked in the reset handling routine. This exception is intended for use in program debugging, and should not normally be generated.

3.6.2 Instruction TLB Miss Exception

An instruction TLB miss exception occurs when address translation information for the virtual address to which an instruction access is made is not found in the UTLB entries by the hardware ITLB miss handling procedure. The instruction TLB miss exception processing carried out by hardware and software is shown below. This is the same as the processing for a data TLB miss exception.

Hardware Processing: In the event of an instruction TLB miss exception, hardware carries out the following processing:

1. Sets the VPN of the virtual address at which the exception occurred in PTEH.
2. Sets the virtual address at which the exception occurred in TEA.
3. Sets exception code H'040 in EXPEVT.
4. Sets the PC value indicating the address of the instruction at which the exception occurred in SPC. If the exception occurred at a delay slot, sets the PC value indicating the address of the delayed branch instruction in SPC.
5. Sets the SR contents at the time of the exception in SSR.
6. Sets the MD bit in SR to 1, and switches to privileged mode.
7. Sets the BL bit in SR to 1, and masks subsequent exception requests.
8. Sets the RB bit in SR to 1.
9. Branches to the address obtained by adding offset H'0000 0400 to the contents of VBR, and starts the instruction TLB miss exception handling routine.

Software Processing (Instruction TLB Miss Exception Handling Routine): Software is responsible for searching the external memory page table and assigning the necessary page table entry. Software should carry out the following processing in order to find and assign the necessary page table entry.

1. Write to PTEL the values of the PPN, PR, SZ, C, D, SH, V, and WT bits in the page table entry recorded in the external memory address translation table. If necessary, the values of the SA and TC bits should be written to PTEA.
2. When the entry to be replaced in entry replacement is specified by software, write that value to URC in the MMUCR register. If URC is greater than URB at this time, the value should be changed to an appropriate value after issuing an LDTLB instruction.
3. Execute the LDTLB instruction and write the contents of PTEH, PTEL, and PTEA to the TLB.
4. Finally, execute the exception handling return instruction (RTE), terminate the exception handling routine, and return control to the normal flow. The RTE instruction should be issued at least one instruction after the LDTLB instruction.

3.6.3 Instruction TLB Protection Violation Exception

An instruction TLB protection violation exception occurs when, even though an ITLB entry contains address translation information matching the virtual address to which an instruction access is made, the actual access type is not permitted by the access right specified by the PR bit. The instruction TLB protection violation exception processing carried out by hardware and software is shown below.

Hardware Processing: In the event of an instruction TLB protection violation exception, hardware carries out the following processing:

1. Sets the VPN of the virtual address at which the exception occurred in PTEH.
2. Sets the virtual address at which the exception occurred in TEA.
3. Sets exception code H'0A0 in EXPEVT.
4. Sets the PC value indicating the address of the instruction at which the exception occurred in SPC. If the exception occurred at a delay slot, sets the PC value indicating the address of the delayed branch instruction in SPC.
5. Sets the SR contents at the time of the exception in SSR.
6. Sets the MD bit in SR to 1, and switches to privileged mode.
7. Sets the BL bit in SR to 1, and masks subsequent exception requests.
8. Sets the RB bit in SR to 1.
9. Branches to the address obtained by adding offset H'0000 0100 to the contents of VBR, and starts the instruction TLB protection violation exception handling routine.

Software Processing (Instruction TLB Protection Violation Exception Handling Routine): Resolve the instruction TLB protection violation, execute the exception handling return instruction (RTE), terminate the exception handling routine, and return control to the normal flow. The RTE instruction should be issued at least one instruction after the LDTLB instruction.

3.6.4 Data TLB Multiple Hit Exception

A data TLB multiple hit exception occurs when more than one UTLB entry matches the virtual address to which a data access has been made. A data TLB multiple hit exception is also generated if multiple hits occur when the UTLB is searched in hardware ITLB miss handling.

When a data TLB multiple hit exception occurs a reset is executed, and cache coherency is not guaranteed. The contents of PPN in the UTLB prior to the exception may also be corrupted.

Hardware Processing: In the event of a data TLB multiple hit exception, hardware carries out the following processing:

1. Sets the virtual address at which the exception occurred in TEA.
2. Sets exception code H'140 in EXPEVT.
3. Branches to the reset handling routine (H'A000 0000).

Software Processing (Reset Routine): The UTLB entries which caused the multiple hit exception are checked in the reset handling routine. This exception is intended for use in program debugging, and should not normally be generated.

3.6.5 Data TLB Miss Exception

A data TLB miss exception occurs when address translation information for the virtual address to which a data access is made is not found in the UTLB entries. The data TLB miss exception processing carried out by hardware and software is shown below.

Hardware Processing: In the event of a data TLB miss exception, hardware carries out the following processing:

1. Sets the VPN of the virtual address at which the exception occurred in PTEH.
2. Sets the virtual address at which the exception occurred in TEA.
3. Sets exception code H'040 in the case of a read, or H'060 in the case of a write, in EXPEVT (OCBP, OCBWB: read; OCBI, MOVCA.L: write).
4. Sets the PC value indicating the address of the instruction at which the exception occurred in SPC. If the exception occurred at a delay slot, sets the PC value indicating the address of the delayed branch instruction in SPC.
5. Sets the SR contents at the time of the exception in SSR.
6. Sets the MD bit in SR to 1, and switches to privileged mode.
7. Sets the BL bit in SR to 1, and masks subsequent exception requests.
8. Sets the RB bit in SR to 1.
9. Branches to the address obtained by adding offset H'0000 0400 to the contents of VBR, and starts the data TLB miss exception handling routine.

Software Processing (Data TLB Miss Exception Handling Routine): Software is responsible for searching the external memory page table and assigning the necessary page table entry. Software should carry out the following processing in order to find and assign the necessary page table entry.

1. Write to PTEL the values of the PPN, PR, SZ, C, D, SH, V, and WT bits in the page table entry recorded in the external memory address translation table. If necessary, the values of the SA and TC bits should be written to PTEA.
2. When the entry to be replaced in entry replacement is specified by software, write that value to URC in the MMUCR register. If URC is greater than URB at this time, the value should be changed to an appropriate value after issuing an LDTLB instruction.
3. Execute the LDTLB instruction and write the contents of PTEH, PTEL, and PTEA to the UTLB.
4. Finally, execute the exception handling return instruction (RTE), terminate the exception handling routine, and return control to the normal flow. The RTE instruction should be issued at least one instruction after the LDTLB instruction.

3.6.6 Data TLB Protection Violation Exception

A data TLB protection violation exception occurs when, even though a UTLB entry contains address translation information matching the virtual address to which a data access is made, the actual access type is not permitted by the access right specified by the PR bit. The data TLB protection violation exception processing carried out by hardware and software is shown below.

Hardware Processing: In the event of a data TLB protection violation exception, hardware carries out the following processing:

1. Sets the VPN of the virtual address at which the exception occurred in PTEH.
2. Sets the virtual address at which the exception occurred in TEA.
3. Sets exception code H'0A0 in the case of a read, or H'0C0 in the case of a write, in EXPEVT (OCBP, OCBWB: read; OCBI, MOVCA.L: write).
4. Sets the PC value indicating the address of the instruction at which the exception occurred in SPC. If the exception occurred at a delay slot, sets the PC value indicating the address of the delayed branch instruction in SPC.
5. Sets the SR contents at the time of the exception in SSR.
6. Sets the MD bit in SR to 1, and switches to privileged mode.
7. Sets the BL bit in SR to 1, and masks subsequent exception requests.
8. Sets the RB bit in SR to 1.
9. Branches to the address obtained by adding offset H'0000 0100 to the contents of VBR, and starts the data TLB protection violation exception handling routine.

Software Processing (Data TLB Protection Violation Exception Handling Routine): Resolve the data TLB protection violation, execute the exception handling return instruction (RTE), terminate the exception handling routine, and return control to the normal flow. The RTE instruction should be issued at least one instruction after the LDTLB instruction.

3.6.7 Initial Page Write Exception

An initial page write exception occurs when the D bit is 0 even though a UTLB entry contains address translation information matching the virtual address to which a data access (write) is made, and the access is permitted. The initial page write exception processing carried out by hardware and software is shown below.

Hardware Processing: In the event of an initial page write exception, hardware carries out the following processing:

1. Sets the VPN of the virtual address at which the exception occurred in PTEH.
2. Sets the virtual address at which the exception occurred in TEA.
3. Sets exception code H'080 in EXPEVT.
4. Sets the PC value indicating the address of the instruction at which the exception occurred in SPC. If the exception occurred at a delay slot, sets the PC value indicating the address of the delayed branch instruction in SPC.
5. Sets the SR contents at the time of the exception in SSR.
6. Sets the MD bit in SR to 1, and switches to privileged mode.
7. Sets the BL bit in SR to 1, and masks subsequent exception requests.
8. Sets the RB bit in SR to 1.
9. Branches to the address obtained by adding offset H'0000 0100 to the contents of VBR, and starts the initial page write exception handling routine.

Software Processing (Initial Page Write Exception Handling Routine): The following processing should be carried out as the responsibility of software:

1. Retrieve the necessary page table entry from external memory.
2. Write 1 to the D bit in the external memory page table entry.
3. Write to PTEL the values of the PPN, PR, SZ, C, D, WT, SH, and V bits in the page table entry recorded in external memory. If necessary, the values of the SA and TC bits should be written to PTEA.
4. When the entry to be replaced in entry replacement is specified by software, write that value to URC in the MMUCR register. If URC is greater than URB at this time, the value should be changed to an appropriate value after issuing an LDTLB instruction.
5. Execute the LDTLB instruction and write the contents of PTEH, PTEL, and PTEA to the UTLB.
6. Finally, execute the exception handling return instruction (RTE), terminate the exception handling routine, and return control to the normal flow. The RTE instruction should be issued at least one instruction after the LDTLB instruction.

3.7 Memory-Mapped TLB Configuration

To enable the ITLB and UTLB to be managed by software, their contents can be read and written by a P2 area program with a MOV instruction in privileged mode. Operation is not guaranteed if access is made from a program in another area. A branch to an area other than the P2 area should be made at least 8 instructions after this MOV instruction. The ITLB and UTLB are allocated to the P4 area in physical memory space. VPN, V, and ASID in the ITLB can be accessed as an address array, PPN, V, SZ, PR, C, and SH as data array 1, and SA and TC as data array 2. VPN, D, V, and ASID in the UTLB can be accessed as an address array, PPN, V, SZ, PR, C, D, WT, and SH as data array 1, and SA and TC as data array 2. V and D can be accessed from both the address array side and the data array side. Only longword access is possible. Instruction fetches cannot be performed in these areas. For reserved bits, a write value of 0 should be specified; their read value is undefined.

3.7.1 ITLB Address Array

The ITLB address array is allocated to addresses H'F200 0000 to H'F2FF FFFF in the P4 area. An address array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and VPN, V, and ASID to be written to the address array are specified in the data field.

In the address field, bits [31:24] have the value H'F2 indicating the ITLB address array, and the entry is selected by bits [9:8]. As longword access is used, 0 should be specified for address field bits [1:0].

In the data field, VPN is indicated by bits [31:10], V by bit [8], and ASID by bits [7:0].

The following two kinds of operation can be used on the ITLB address array:

1. ITLB address array read

   VPN, V, and ASID are read into the data field from the ITLB entry corresponding to the entry set in the address field.
2. ITLB address array write

   VPN, V, and ASID specified in the data field are written to the ITLB entry corresponding to the entry set in the address field.

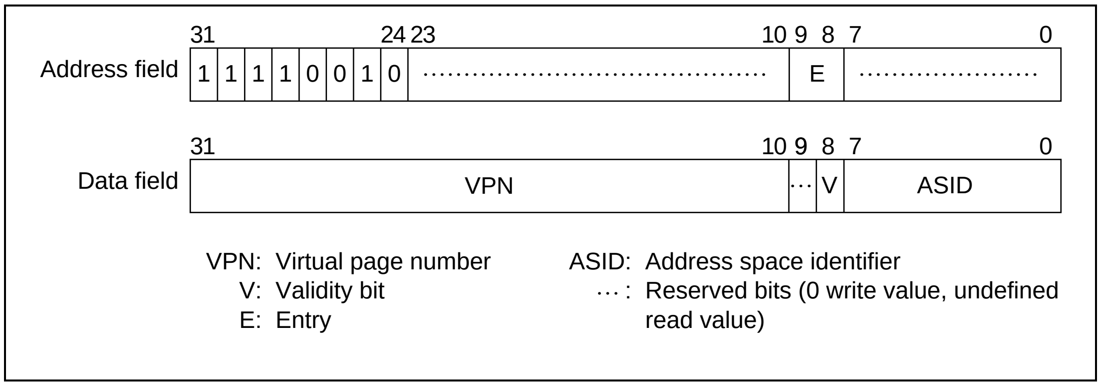


Figure 3.13 Memory-Mapped ITLB Address Array

3.7.2 ITLB Data Array 1

ITLB data array 1 is allocated to addresses H'F300 0000 to H'F37F FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and PPN, V, SZ, PR, C, and SH to be written to the data array are specified in the data field.

In the address field, bits [31:23] have the value H'F30 indicating ITLB data array 1, and the entry is selected by bits [9:8].

In the data field, PPN is indicated by bits [28:10], V by bit [8], SZ by bits [7] and [4], PR by bit [6], C by bit [3], and SH by bit [1].

The following two kinds of operation can be used on ITLB data array 1:

1. ITLB data array 1 read

   PPN, V, SZ, PR, C, and SH are read into the data field from the ITLB entry corresponding to the entry set in the address field.
2. ITLB data array 1 write

   PPN, V, SZ, PR, C, and SH specified in the data field are written to the ITLB entry corresponding to the entry set in the address field.

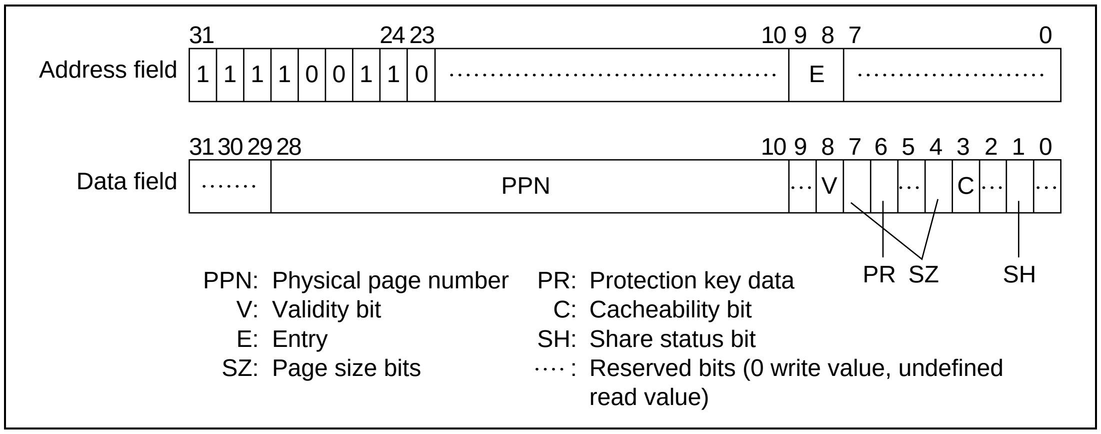


Figure 3.14 Memory-Mapped ITLB Data Array 1

3.7.3 ITLB Data Array 2

ITLB data array 2 is allocated to addresses H'F380 0000 to H'F3FF FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and SA and TC to be written to data array 2 are specified in the data field.

In the address field, bits [31:23] have the value H'F38 indicating ITLB data array 2, and the entry is selected by bits [9:8].

In the data field, SA is indicated by bits [2:0], and TC by bit [3].

The following two kinds of operation can be used on ITLB data array 2:

1. ITLB data array 2 read
   SA and TC are read into the data field from the ITLB entry corresponding to the entry set in the address field.
2. ITLB data array 2 write
   SA and TC specified in the data field are written to the ITLB entry corresponding to the entry set in the address field.

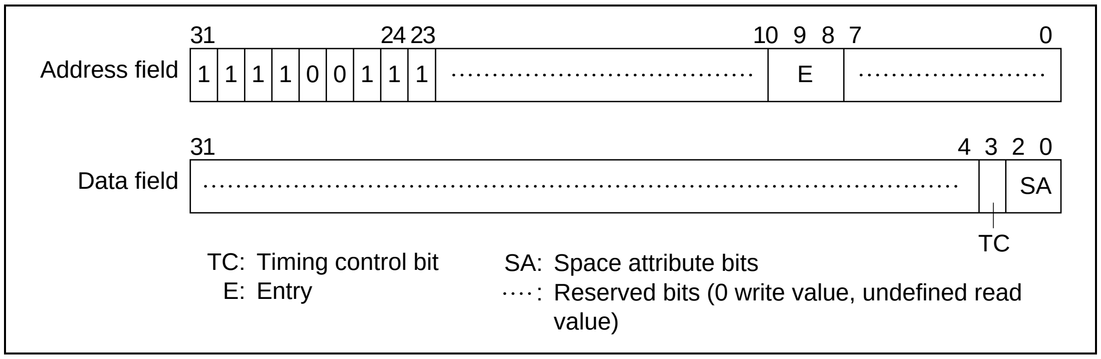


Figure 3.15 Memory-Mapped ITLB Data Array 2

HITACHI

3.7.4 UTLB Address Array

The UTLB address array is allocated to addresses H'F600 0000 to H'F6FF FFFF in the P4 area. An address array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and VPN, D, V, and ASID to be written to the address array are specified in the data field.

In the address field, bits [31:24] have the value H'F6 indicating the UTLB address array, and the entry is selected by bits [13:8]. The address array bit [7] association bit (A bit) specifies whether or not address comparison is performed when writing to the UTLB address array.

In the data field, VPN is indicated by bits [31:10], D by bit [9], V by bit [8], and ASID by bits [7:0].

The following three kinds of operation can be used on the UTLB address array:

1. UTLB address array read

   VPN, D, V, and ASID are read into the data field from the UTLB entry corresponding to the entry set in the address field. In a read, associative operation is not performed regardless of whether the association bit specified in the address field is 1 or 0.
2. UTLB address array write (non-associative)

   VPN, D, V, and ASID specified in the data field are written to the UTLB entry corresponding to the entry set in the address field. The A bit in the address field should be cleared to 0.
3. UTLB address array write (associative)

   When a write is performed with the A bit in the address field set to 1, comparison of all the UTLB entries is carried out using the VPN specified in the data field and PTEH.ASID. The usual address comparison rules are followed, but if a UTLB miss occurs, the result is no operation, and an exception is not generated. If the comparison identifies a UTLB entry corresponding to the VPN specified in the data field, D and V specified in the data field are written to that entry. If there is more than one matching entry, a data TLB multiple hit exception results. This associative operation is simultaneously carried out on the ITLB, and if a matching entry is found in the ITLB, V is written to that entry. Even if the UTLB comparison results in no operation, a write to the ITLB side only is performed as long as there is an ITLB match. If there is a match in both the UTLB and ITLB, the UTLB information is also written to the ITLB.

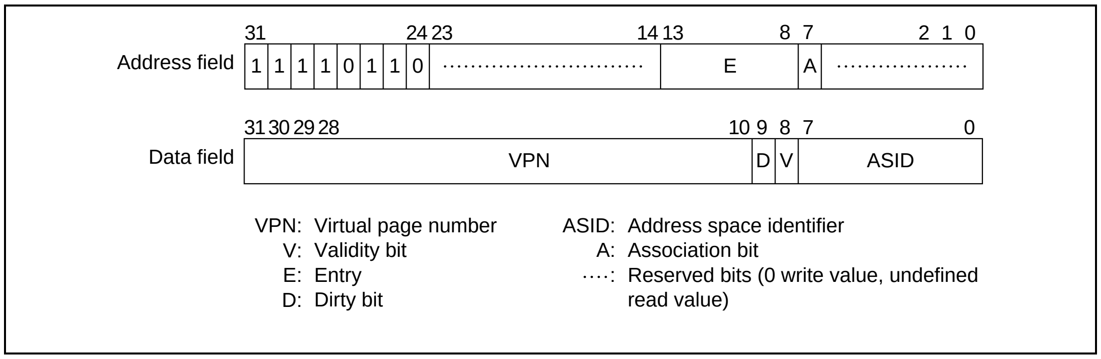


Figure 3.16 Memory-Mapped UTLB Address Array

3.7.5 UTLB Data Array 1

UTLB data array 1 is allocated to addresses H'F700 0000 to H'F77F FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and PPN, V, SZ, PR, C, D, SH, and WT to be written to the data array are specified in the data field.

In the address field, bits [31:23] have the value H'F70 indicating UTLB data array 1, and the entry is selected by bits [13:8].

In the data field, PPN is indicated by bits [28:10], V by bit [8], SZ by bits [7] and [4], PR by bits [6:5], C by bit [3], D by bit [2], SH by bit [1], and WT by bit [0].

The following two kinds of operation can be used on UTLB data array 1:

1. UTLB data array 1 read

   PPN, V, SZ, PR, C, D, SH, and WT are read into the data field from the UTLB entry corresponding to the entry set in the address field.
2. UTLB data array 1 write

   PPN, V, SZ, PR, C, D, SH, and WT specified in the data field are written to the UTLB entry corresponding to the entry set in the address field.

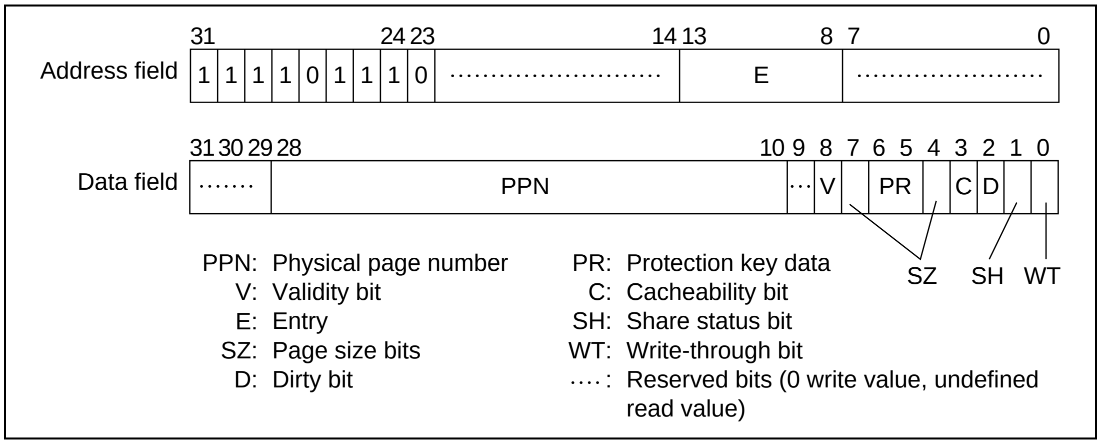


Figure 3.17 Memory-Mapped UTLB Data Array 1

3.7.6 UTLB Data Array 2

UTLB data array 2 is allocated to addresses H'F780 0000 to H'F7FF FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification (when writing). Information for selecting the entry to be accessed is specified in the address field, and SA and TC to be written to data array 2 are specified in the data field.

In the address field, bits [31:23] have the value H'F78 indicating UTLB data array 2, and the entry is selected by bits [13:8].

In the data field, TC is indicated by bit [3], and SA by bits [2:0].

The following two kinds of operation can be used on UTLB data array 2:

1. UTLB data array 2 read

   SA and TC are read into the data field from the UTLB entry corresponding to the entry set in the address field.
2. UTLB data array 2 write

   SA and TC specified in the data field are written to the UTLB entry corresponding to the entry set in the address field.

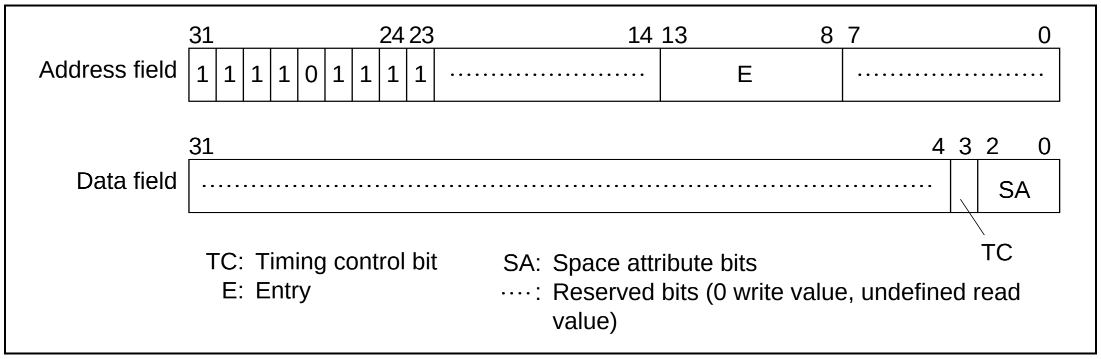


Figure 3.18   Memory-Mapped UTLB Data Array 2

Section 4 Caches

4.1 Overview

4.1.1 Features

The SH7750 has an on-chip 8-kbyte instruction cache (IC) for instructions and 16-kbyte operand cache (OC) for data. Half of the memory of the operand cache (8 kbytes) can also be used as on-chip RAM. The features of these caches are summarized in table 4.1.

Table 4.1 Cache Features

Item	Instruction Cache	Operand Cache
Capacity	8-kbyte cache	16-kbyte cache or 8-kbyte cache + 8-kbyte RAM
Type	Direct mapping	Direct mapping
Line size	32 bytes	32 bytes
Entries	256	512
Write method		Copy-back/write-through selectable

Item	Store Queues
Capacity	2 × 32 bytes
Addresses	H'E000 0000 to H'E3FF FFFF
Write	Store instruction (1-cycle write)
Write-back	Prefetch instruction
Access right	MMU off: according to MMUCR.SQMD MMU on: according to individual page PR

4.1.2 Register Configuration

Table 4.2 shows the cache control registers.

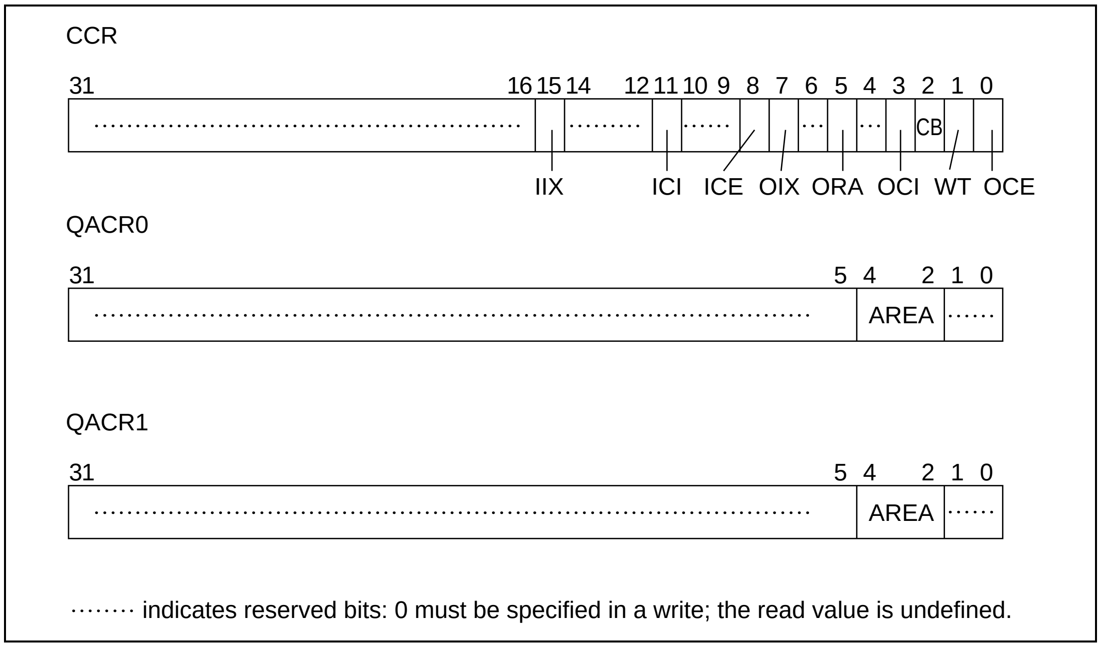
Table 4.2 Cache Control Registers

Name	Abbreviation	R/W	Initial Value*[1]	P4 Address*[2]	Area 7 Address*[2]	Access Size
Cache control register	CCR	R/W	H'0000 0000	H'FF00 001C	H'1F00 001C	32
Queue address control register 0	QACR0	R/W	Undefined	H'FF00 0038	H'1F00 0038	32
Queue address control register 1	QACR1	R/W	Undefined	H'FF00 003C	H'1F00 003C	32

Notes: 1. The initial value is the value after a power-on or manual reset.
2. This is the address when using the virtual/physical address space P4 area. When making an access from physical address space area 7 using the TLB, the upper 3 bits of the address are ignored.

4.2 Register Descriptions

There are three cache and store queue related control registers, as shown in figure 4.1.



CCR
31
16
15
14
12
11
10
9
8
7
6
5
4
3
2
1
0
CB
IIX
ICI
ICE
OIX
ORA
OCI
WT
OCE
QACR0
31
5
4
2
1
0
AREA
QACR1
31
5
4
2
1
0
AREA
········ indicates reserved bits: 0 must be specified in a write; the read value is undefined.

Figure 4.1 Cache and Store Queue Control Registers

(1) Cache Control Register (CCR): CCR contains the following bits:

IIX:	IC index enable
ICI:	IC invalidation
ICE:	IC enable
OIX:	OC index enable
ORA:	OC RAM enable
OCI:	OC invalidation
CB:	Copy-back enable
WT:	Write-through enable
OCE:	OC enable

Longword access to CCR can be performed from H'FF00 001C in the P4 area and H'1F00 001C in area 7. The CCR bits are used for the cache settings described below. Consequently, CCR modifications must only be made by a program in the non-cached P2 area. After CCR is updated, an instruction that performs data access to the P0, P1, P3, or U0 area should be located at least four instructions after the CCR update instruction. Also, a branch instruction to the P0, P1, P3, or U0 area should be located at least eight instructions after the CCR update instruction.

- IIX: IC index enable bit

  0: Address bits [12:5] used for IC entry selection

  1: Address bits [25] and [11:5] used for IC entry selection
- ICI: IC invalidation bit

  When 1 is written to this bit, the V bits of all IC entries are cleared to 0. This bit always returns 0 when read.
- ICE: IC enable bit

  Indicates whether or not the IC is to be used. When address translation is performed, the IC cannot be used unless the C bit in the page management information is also 1.

  0: IC not used

  1: IC used
- OIX: OC index enable bit

  0: Address bits [13:5] used for OC entry selection

  1: Address bits [25] and [12:5] used for OC entry selection
- ORA: OC RAM enable bit

  When the OC is enabled (OCE = 1), the ORA bit specifies whether the 8 kbytes from entry 128 to entry 255 and from entry 384 to entry 511 of the OC are to be used as RAM. When the OC is not enabled (OCE = 0), the ORA bit should be cleared to 0.

  0: 16 kbytes used as cache

  1: 8 kbytes used as cache, and 8 kbytes as RAM

- OCI: OC invalidation bit

  When 1 is written to this bit, the V and U bits of all OC entries are cleared to 0. This bit always returns 0 when read.
- CB: Copy-back bit

  Indicates the P1 area cache write mode.

  0: Write-through mode

  1: Copy-back mode
- WT: Write-through bit

  Indicates the P0, U0, and P3 area cache write mode. When address translation is performed, the value of the WT bit in the page management information has priority.

  0: Copy-back mode

  1: Write-through mode
- OCE: OC enable bit

  Indicates whether or not the OC is to be used. When address translation is performed, the OC cannot be used unless the C bit in the page management information is also 1.

  0: OC not used

  1: OC used

(2) Queue Address Control Register 0 (QACR0): Longword access to QACR0 can be performed from H'FF00 0038 in the P4 area and H'1F00 0038 in area 7. QACR0 specifies the area onto which store queue 0 (SQ0) is mapped when the MMU is off.

(3) Queue Address Control Register 1 (QACR1): Longword access to QACR1 can be performed from H'FF00 003C in the P4 area and H'1F00 003C in area 7. QACR1 specifies the area onto which store queue 1 (SQ1) is mapped when the MMU is off.

4.3 Operand Cache (OC)

4.3.1 Configuration

Figure 4.2 shows the configuration of the operand cache.

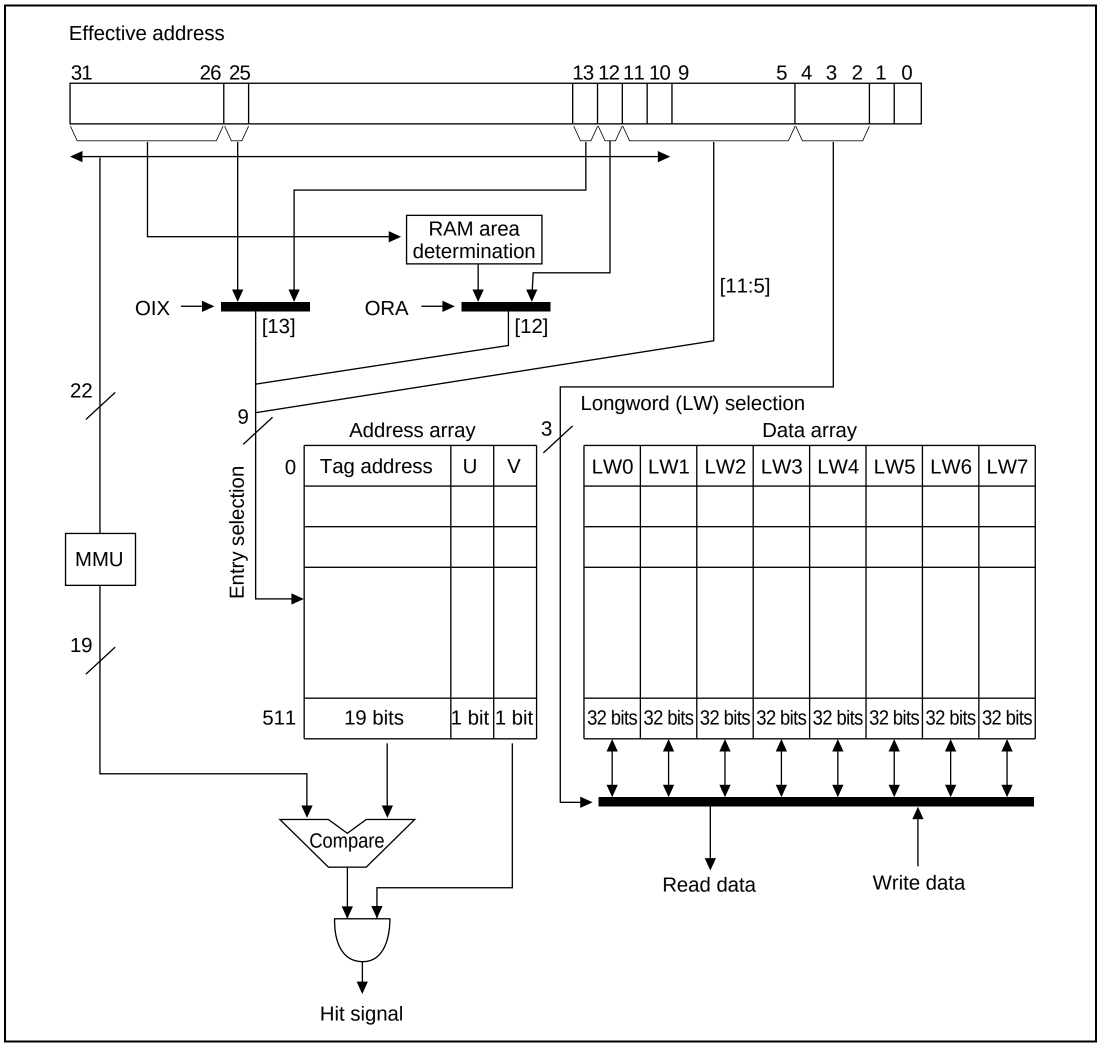


Figure 4.2 Configuration of Operand Cache

The operand cache consists of 512 cache lines, each composed of a 19-bit tag, V bit, U bit, and 32-byte data.

- Tag

  Stores the upper 19 bits of the 29-bit external memory address of the data line to be cached. The tag is not initialized by a power-on or manual reset.
- V bit (validity bit)

  Indicates that valid data is stored in the cache line. When this bit is 1, the cache line data is valid. The V bit is initialized to 0 by a power-on reset, but retains its value in a manual reset.
- U bit (dirty bit)

  The U bit is set to 1 if data is written to the cache line while the cache is being used in copy-back mode. That is, the U bit indicates a mismatch between the data in the cache line and the data in external memory. The U bit is never set to 1 while the cache is being used in write-through mode, unless it is modified by accessing the memory-mapped cache (see section 4.5, Memory-Mapped Cache Configuration). The U bit is initialized to 0 by a power-on reset, but retains its value in a manual reset.
- Data field

  The data field holds 32 bytes (256 bits) of data per cache line. The data array is not initialized by a power-on or manual reset.

4.3.2 Read Operation

When the OC is enabled (CCR.OCE = 1) and data is read by means of an effective address from a cacheable area, the cache operates as follows:

1. The tag, V bit, and U bit are read from the cache line indexed by effective address bits [13:5].
2. The tag is compared with bits [28:10] of the address resulting from effective address translation by the MMU:

• If the tag matches and the V bit is 1	→ (3a)
• If the tag matches and the V bit is 0	→ (3b)
• If the tag does not match and the V bit is 0	→ (3b)
• If the tag does not match, the V bit is 1, and the U bit is 0	→ (3b)
• If the tag does not match, the V bit is 1, and the U bit is 1	→ (3c)

3a. Cache hit

The data indexed by effective address bits [4:0] is read from the data field of the cache line indexed by effective address bits [13:5] in accordance with the access size (quadword/longword/word/byte).

3b. Cache miss (no write-back)

Data is read into the cache line from the external memory space corresponding to the effective address. Data reading is performed, using the wraparound method, in order from the longword data corresponding to the effective address, and when the corresponding data arrives in the cache, the read data is returned to the CPU. While the remaining one cache line of data is being read, the CPU can execute the next processing. When reading of one line of data is completed, the tag corresponding to the effective address is recorded in the cache, and 1 is written to the V bit.

3c. Cache miss (with write-back)

The tag and data field of the cache line indexed by effective address bits [13:5] are saved in the write-back buffer. Then data is read into the cache line from the external memory space corresponding to the effective address. Data reading is performed, using the wraparound method, in order from the longword data corresponding to the effective address, and when the corresponding data arrives in the cache, the read data is returned to the CPU. While the remaining one cache line of data is being read, the CPU can execute the next processing. When reading of one line of data is completed, the tag corresponding to the effective address is recorded in the cache, 1 is written to the V bit, and 0 to the U bit. The data in the write-back buffer is then written back to external memory.

4.3.3 Write Operation

When the OC is enabled (CCR.OCE = 1) and data is written by means of an effective address to a cacheable area, the cache operates as follows:

1. The tag, V bit, and U bit are read from the cache line indexed by effective address bits [13:5].
2. The tag is compared with bits [28:10] of the address resulting from effective address translation by the MMU:

	Copy-back	Write-through
• If the tag matches and the V bit is 1	→ (3a)	→ (3b)
• If the tag matches and the V bit is 0	→ (3c)	→ (3d)
• If the tag does not match and the V bit is 0	→ (3c)	→ (3d)
• If the tag does not match, the V bit is 1, and the U bit is 0	→ (3c)	→ (3d)
• If the tag does not match, the V bit is 1, and the U bit is 1	→ (3e)	→ (3d)

3a. Cache hit (copy-back)

A data write in accordance with the access size (quadword/longword/word/byte) is performed for the data indexed by bits [4:0] of the effective address of the data field of the cache line indexed by effective address bits [13:5]. Then 1 is set in the U bit.

3b. Cache hit (write-through)

A data write in accordance with the access size (quadword/longword/word/byte) is performed for the data indexed by bits [4:0] of the effective address of the data field of the cache line indexed by effective address bits [13:5]. A write is also performed to the corresponding external memory using the specified access size.

3c. Cache miss (no copy-back/write-back)

A data write in accordance with the access size (quadword/longword/word/byte) is performed for the data indexed by bits [4:0] of the effective address of the data field of the cache line indexed by effective address bits [13:5]. Then, data is read into the cache line from the external memory space corresponding to the effective address. Data reading is performed, using the wraparound method, in order from the longword data corresponding to the effective address, and one cache line of data is read excluding the written data. During this time, the CPU can execute the next processing. When reading of one line of data is completed, the tag corresponding to the effective address is recorded in the cache, and 1 is written to the V bit and U bit.

3d. Cache miss (write-through)

A write of the specified access size is performed to the external memory corresponding to the effective address. In this case, a write to cache is not performed.

3e. Cache miss (with copy-back/write-back)

The tag and data field of the cache line indexed by effective address bits [13:5] are first saved in the write-back buffer, and then a data write in accordance with the access size (quadword/longword/word/byte) is performed for the data indexed by bits [4:0] of the effective address of the data field of the cache line indexed by effective address bits [13:5]. Then, data is read into the cache line from the external memory space corresponding to the effective address. Data reading is performed, using the wraparound method, in order from the longword data corresponding to the effective address, and one cache line of data is read excluding the written data. During this time, the CPU can execute the next processing. When reading of one line of data is completed, the tag corresponding to the effective address is recorded in the cache, and 1 is written to the V bit and U bit. The data in the write-back buffer is then written back to external memory.

4.3.4 Write-Back Buffer

In order to give priority to data reads to the cache and improve performance, the SH7750 has a write-back buffer which holds the relevant cache entry when it becomes necessary to purge a dirty cache entry into external memory as the result of a cache miss. The write-back buffer contains one cache line of data and the physical address of the purge destination.

Physical address bits [28:5]	LW0	LW1	LW2	LW3	LW4	LW5	LW6	LW7

Figure 4.3 Configuration of Write-Back Buffer

4.3.5 Write-Through Buffer

The SH7750 has a 64-bit buffer for holding write data when writing data in write-through mode or writing to a non-cacheable area. This allows the CPU to proceed to the next operation as soon as the write to the write-through buffer is completed, without waiting for completion of the write to external memory.

Physical address bits [28:0]	LW0	LW1

Figure 4.4 Configuration of Write-Through Buffer

4.3.6 RAM Mode

Setting CCR.ORA to 1 enables 8 kbytes of the operand cache to be used as RAM. The operand cache entries used as RAM are entries 128 to 255 and 384 to 511 . Other entries can still be used as cache. RAM can be accessed using addresses H'7C00 0000 to H'7FFF FFFF. Byte-, word-, longword-, and quadword-size data reads and writes can be performed in the operand cache RAM area. Instruction fetches cannot be performed in this area.

An example of RAM use is shown below. Here, the 4 kbytes comprising OC entries 128 to 256 are designated as RAM area 1, and the 4 kbytes comprising OC entries 384 to 511 as RAM area 2.

- When OC index mode is off (CCR.OIX = 0)

  H'7C00 0000 to H'7C00 0FFF (4 kB): Corresponds to RAM area 1

  H'7C00 1000 to H'7C00 1FFF (4 kB): Corresponds to RAM area 1

  H'7C00 2000 to H'7C00 2FFF (4 kB): Corresponds to RAM area 2

  H'7C00 3000 to H'7C00 3FFF (4 kB): Corresponds to RAM area 2

  H'7C00 4000 to H'7C00 4FFF (4 kB): Corresponds to RAM area 1

  : : :

  RAM areas 1 and 2 then repeat every 8 kbytes up to H'7FFF FFFF.

  Thus, to secure a continuous 8-kbyte RAM area, the area from H'7C00 1000 to H'7C00 2FFF can be used, for example.

- When OC index mode is on (CCR.OIX = 1)

  H'7C00 0000 to H'7C00 0FFF (4 kB): Corresponds to RAM area 1

  H'7C00 1000 to H'7C00 1FFF (4 kB): Corresponds to RAM area 1

  H'7C00 2000 to H'7C00 2FFF (4 kB): Corresponds to RAM area 1

  : : :

  H'7DFF F000 to H'7DFF FFFF (4 kB): Corresponds to RAM area 1

  H'7E00 0000 to H'7E00 0FFF (4 kB): Corresponds to RAM area 2

  H'7E00 1000 to H'7E00 1FFF (4 kB): Corresponds to RAM area 2

  : : :

  H'7FFF F000 to H'7FFF FFFF (4 kB): Corresponds to RAM area 2

  As the distinction between RAM areas 1 and 2 is indicated by address bit [25], the area from H'7DFF F000 to H'7E00 0FFF should be used to secure a continuous 8-kbyte RAM area.

4.3.7 OC Index Mode

Setting CCR.OIX to 1 enables OC indexing to be performed using bit [25] of the effective address. This is called OC index mode. In normal mode, with CCR.OIX cleared to 0, OC indexing is performed using bits [13:5] of the effective address; therefore, when 16 kbytes or more of consecutive data is handled, the OC is fully used by this data. This results in frequent cache misses. Using index mode allows the OC to be handled as two 8-kbyte areas by means of effective address bit [25], providing efficient use of the cache.

4.3.8 Coherency between Cache and External Memory

Coherency between cache and external memory should be assured by software. In the SH7750, the following four new instructions are supported for cache operations. Details of these instructions are given in the Programming Manual.

Invalidate instruction:	OCBI @Rn	Cache invalidation (no write-back)
Purge instruction:	OCBP @Rn	Cache invalidation (with write-back)
Write-back instruction:	OCBWB @Rn	Cache write-back
Allocate instruction:	MOVCA.L R0,@Rn	Cache allocation

4.3.9 Prefetch Operation

The SH7750 supports a prefetch instruction to reduce the cache fill penalty incurred as the result of a cache miss. If it is known that a cache miss will result from a read or write operation, it is possible to fill the cache with data beforehand by means of the prefetch instruction to prevent a cache miss due to the read or write operation, and so improve software performance. If a prefetch instruction is executed for data already held in the cache, or if the prefetch address results in a UTLB miss or a protection violation, the result is no operation, and an exception is not generated. Details of the prefetch instruction are given in the Programming Manual.

Prefetch instruction:	PREF @Rn

4.4 Instruction Cache (IC)

4.4.1 Configuration

Figure 4.5 shows the configuration of the instruction cache.

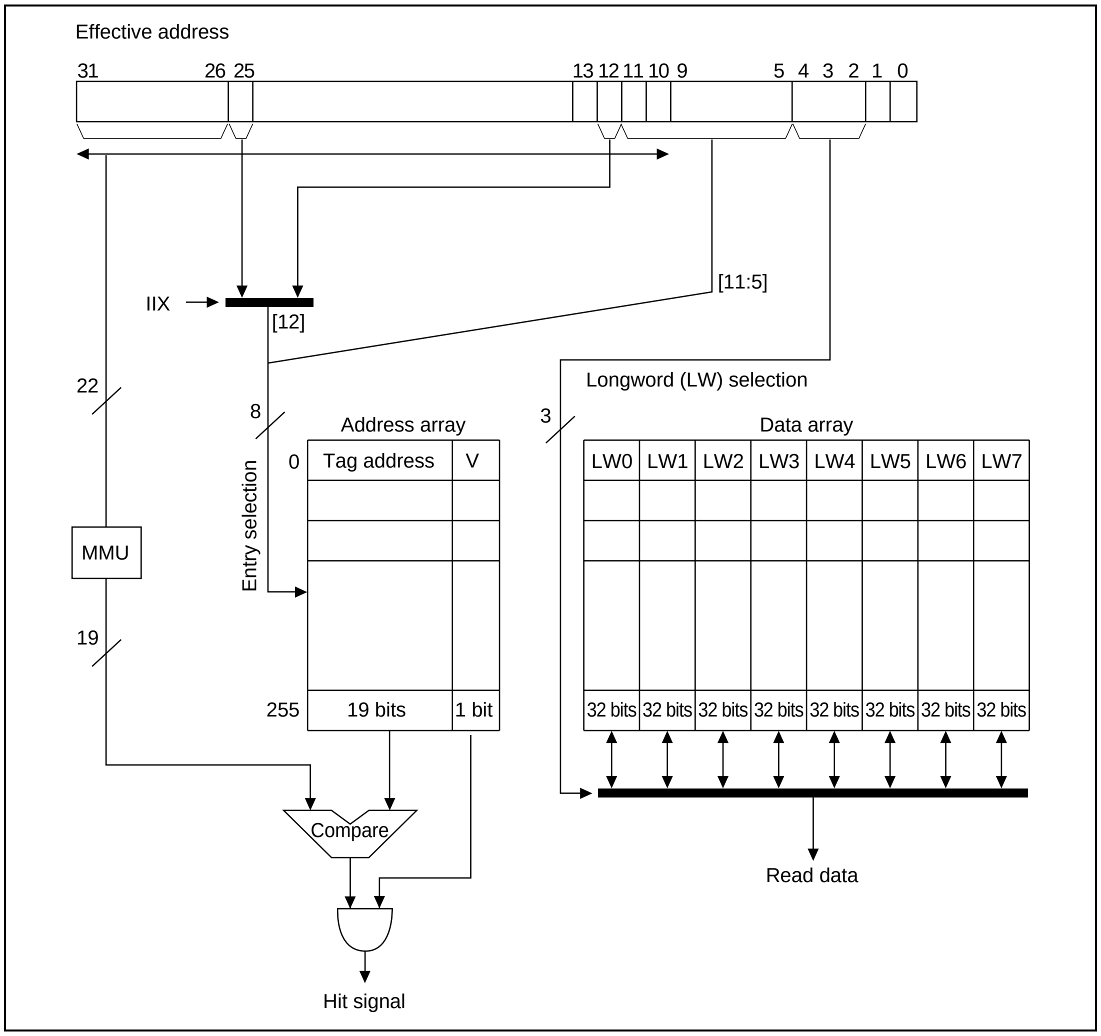


Figure 4.5 Configuration of Instruction Cache

The instruction cache consists of 256 cache lines, each composed of a 19-bit tag, V bit, and 32-byte data (16 instructions).

- Tag

  Stores the upper 19 bits of the 29-bit external memory address of the data line to be cached. The tag is not initialized by a power-on or manual reset.
- V bit (validity bit)

  Indicates that valid data is stored in the cache line. When this bit is 1, the cache line data is valid. The V bit is initialized to 0 by a power-on reset, but retains its value in a manual reset.
- Data array

  The data field holds 32 bytes (256 bits) of data per cache line. The data array is not initialized by a power-on or manual reset.

4.4.2 Read Operation

When the IC is enabled (CCR.ICE = 1) and instruction fetches are performed by means of an effective address from a cacheable area, the instruction cache operates as follows:

1. The tag and V bit are read from the cache line indexed by effective address bits [12:5].
2. The tag is compared with bits [28:10] of the address resulting from effective address translation by the MMU:
   - If the tag matches and the V bit is 1 $\rightarrow$ (3a)
   - If the tag matches and the V bit is 0 $\rightarrow$ (3b)
   - If the tag does not match and the V bit is 0 $\rightarrow$ (3b)
   - If the tag does not match and the V bit is 1 $\rightarrow$ (3b)

3a. Cache hit

The data indexed by effective address bits [4:2] is read as an instruction from the data field of the cache line indexed by effective address bits [12:5].

3b. Cache miss

Data is read into the cache line from the external memory space corresponding to the effective address. Data reading is performed, using the wraparound method, in order from the longword data corresponding to the effective address, and when the corresponding data arrives in the cache, the read data is returned to the CPU as an instruction. When reading of one line of data is completed, the tag corresponding to the effective address is recorded in the cache, and 1 is written to the V bit.

4.4.3 IC Index Mode

Setting CCR.IIX to 1 enables IC indexing to be performed using bit [25] of the effective address. This is called IC index mode. In normal mode, with CCR.IIX cleared to 0, IC indexing is performed using bits [12:5] of the effective address; therefore, when 8 kbytes or more of consecutive program instructions are handled, the IC is fully used by this program. This results in frequent cache misses. Using index mode allows the IC to be handled as two 4-kbyte areas by means of effective address bit [25], providing efficient use of the cache.

4.5 Memory-Mapped Cache Configuration

To enable the IC and OC to be managed by software, their contents can be read and written by a P2 area program with a MOV instruction in privileged mode. Operation is not guaranteed if access is made from a program in another area. In this case, a branch to the P0, U0, P1, or P3 area should be made at least 8 instructions after this MOV instruction. The IC and OC are allocated to the P4 area in physical memory space. Only data accesses can be used on both the IC address array and data array and the OC address array and data array, and accesses are always longword-size. Instruction fetches cannot be performed in these areas. For reserved bits, a write value of 0 should be specified; their read value is undefined.

4.5.1 IC Address Array

The IC address array is allocated to addresses H'F000 0000 to H'F0FF FFFF in the P4 area. An address array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The entry to be accessed is specified in the address field, and the write tag and V bit are specified in the data field.

In the address field, bits [31:24] have the value H'F0 indicating the IC address array, and the entry is specified by bits [12:5]. CCR.IIX has no effect on this entry specification. The address array bit [3] association bit (A bit) specifies whether or not association is performed when writing to the IC address array. As only longword access is used, 0 should be specified for address field bits [1:0].

In the data field, the tag is indicated by bits [31:10], and the V bit by bit [0]. As the IC address array tag is 19 bits in length, data field bits [31:29] are not used in the case of a write in which association is not performed. Data field bits [31:29] are used for the virtual address specification only in the case of a write in which association is performed.

The following three kinds of operation can be used on the IC address array:

1. IC address array read

   The tag and V bit are read into the data field from the IC entry corresponding to the entry set in the address field. In a read, associative operation is not performed regardless of whether the association bit specified in the address field is 1 or 0.
2. IC address array write (non-associative)

   The tag and V bit specified in the data field are written to the IC entry corresponding to the entry set in the address field. The A bit in the address field should be cleared to 0.
3. IC address array write (associative)

   When a write is performed with the A bit in the address field set to 1, the tag stored in the entry specified in the address field is compared with the tag specified in the data field. If the MMU is enabled at this time, comparison is performed after the virtual address specified by data field bits [31:10] has been translated to a physical address using the ITLB. If the addresses match and the V bit is 1, the V bit specified in the data field is written into the IC entry. This operation is used to invalidate a specific IC entry. If an ITLB miss occurs during address translation, or the comparison shows a mismatch, no operation results and the write is not performed. If an instruction TLB multiple hit exception occurs during address translation, processing switches to the instruction TLB multiple hit exception handling routine.

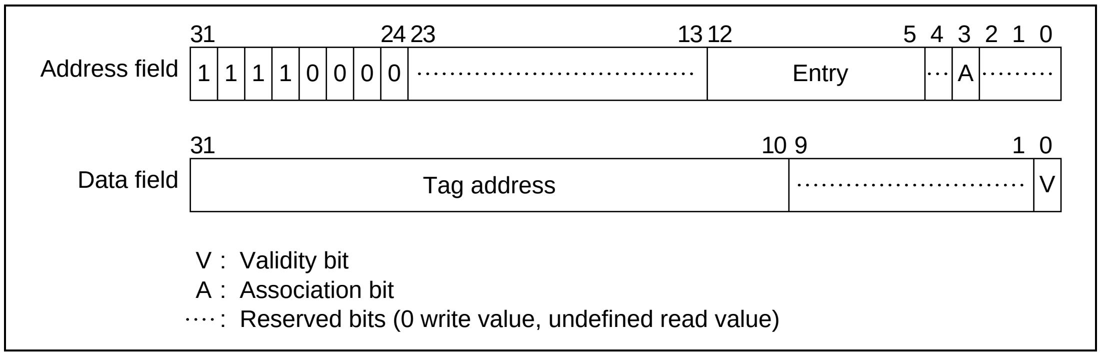


Figure 4.6 Memory-Mapped IC Address Array

4.5.2 IC Data Array

The IC data array is allocated to addresses H'F100 0000 to H'F1FF FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The entry to be accessed is specified in the address field, and the longword data to be written is specified in the data field.

In the address field, bits [31:24] have the value H'F1 indicating the IC data array, and the entry is specified by bits [12:5]. CCR.IIX has no effect on this entry specification. Address field bits [4:2] are used for the longword data specification in the entry. As only longword access is used, 0 should be specified for address field bits [1:0].

The data field is used for the longword data specification.

The following two kinds of operation can be used on the IC data array:

1. IC data array read
   Longword data is read into the data field from the data specified by the longword specification bits in the address field in the IC entry corresponding to the entry set in the address field.
2. IC data array write
   The longword data specified in the data field is written for the data specified by the longword specification bits in the address field in the IC entry corresponding to the entry set in the address field.

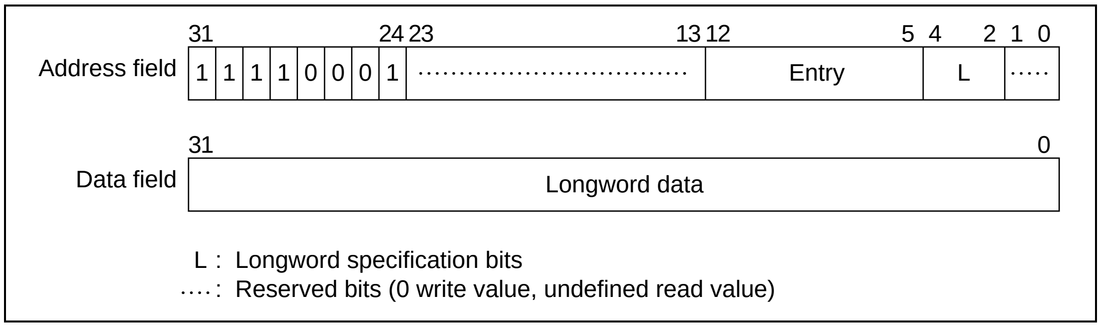

31
24 23
13 12
5 4
2 1 0
Address field
1 1 1 1 0 0 0 1
Entry
L
31
0
Data field
Longword data
L : Longword specification bits
···· : Reserved bits (0 write value, undefined read value)

Figure 4.7 Memory-Mapped IC Data Array

4.5.3 OC Address Array

The OC address array is allocated to addresses H'F400 0000 to H'F4FF FFFF in the P4 area. An address array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The entry to be accessed is specified in the address field, and the write tag, U bit, and V bit are specified in the data field.

In the address field, bits [31:24] have the value H'F4 indicating the OC address array, and the entry is specified by bits [13:5]. CCR.OIX and CCR.ORA have no effect on this entry specification. The address array bit [3] association bit (A bit) specifies whether or not association is performed when writing to the OC address array. As only longword access is used, 0 should be specified for address field bits [1:0].

In the data field, the tag is indicated by bits [31:10], the U bit by bit [1], and the V bit by bit [0]. As the OC address array tag is 19 bits in length, data field bits [31:29] are not used in the case of a write in which association is not performed. Data field bits [31:29] are used for the virtual address specification only in the case of a write in which association is performed.

The following three kinds of operation can be used on the OC address array:

1. OC address array read

   The tag, U bit, and V bit are read into the data field from the OC entry corresponding to the entry set in the address field. In a read, associative operation is not performed regardless of whether the association bit specified in the address field is 1 or 0.
2. OC address array write (non-associative)

   The tag, U bit, and V bit specified in the data field are written to the OC entry corresponding to the entry set in the address field. The A bit in the address field should be cleared to 0.

   When a write is performed to a cache line for which the U bit and V bit are both 1, after write-back of that cache line, the tag, U bit, and V bit specified in the data field are written.
3. OC address array write (associative)

   When a write is performed with the A bit in the address field set to 1, the tag stored in the entry specified in the address field is compared with the tag specified in the data field. If the MMU is enabled at this time, comparison is performed after the virtual address specified by data field bits [31:10] has been translated to a physical address using the UTLB. If the addresses match and the V bit is 1, the U bit and V bit specified in the data field are written into the OC entry. This operation is used to invalidate a specific OC entry. If the OC entry U bit is 1, and 0 is written to the V bit or to the U bit, write-back is performed. If an UTLB miss occurs during address translation, or the comparison shows a mismatch, no operation results and the write is not performed. If a data TLB multiple hit exception occurs during address translation, processing switches to the data TLB multiple hit exception handling routine.

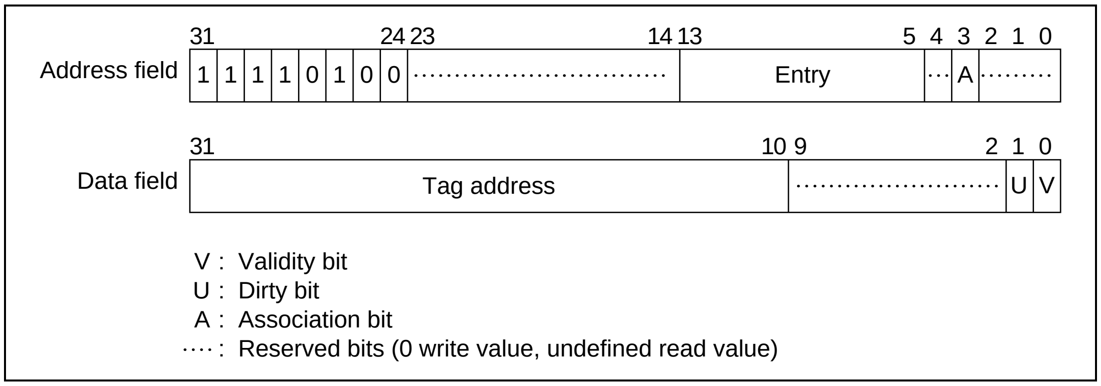


Figure 4.8 Memory-Mapped OC Address Array

4.5.4 OC Data Array

The OC data array is allocated to addresses H'F500 0000 to H'F5FF FFFF in the P4 area. A data array access requires a 32-bit address field specification (when reading or writing) and a 32-bit data field specification. The entry to be accessed is specified in the address field, and the longword data to be written is specified in the data field.

In the address field, bits [31:24] have the value H'F5 indicating the OC data array, and the entry is specified by bits [13:5]. CCR.OIX and CCR.ORA have no effect on this entry specification. Address field bits [4:2] are used for the longword data specification in the entry. As only longword access is used, 0 should be specified for address field bits [1:0].

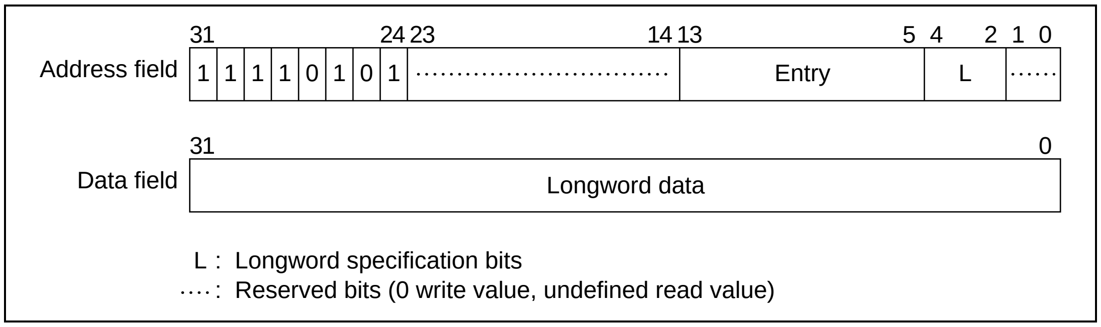
The data field is used for the longword data specification.

The following two kinds of operation can be used on the OC data array:

1. OC data array read

   Longword data is read into the data field from the data specified by the longword specification bits in the address field in the OC entry corresponding to the entry set in the address field.
2. OC data array write

   The longword data specified in the data field is written for the data specified by the longword specification bits in the address field in the OC entry corresponding the entry set in the address field. This write does not set the U bit to 1 on the address array side.



Address field
31
24
23
14
13
5
4
2
1
0
1 1 1 1 0 1 0 1
Entry
L
Data field
31
0
Longword data
L : Longword specification bits
···· : Reserved bits (0 write value, undefined read value)

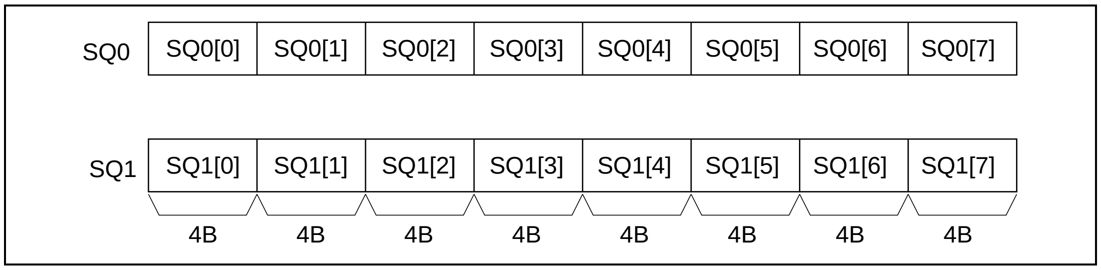
Figure 4.9 Memory-Mapped OC Data Array

4.6 Store Queues

Two 32-byte store queues (SQs) are supported to perform high-speed writes to external memory.

4.6.1 SQ Configuration

There are two 32-byte store queues, SQ0 and SQ1, as shown in figure 4.10. These two store queues can be set independently.



SQ0
SQ0[0] SQ0[1] SQ0[2] SQ0[3] SQ0[4] SQ0[5] SQ0[6] SQ0[7]
SQ1
SQ1[0] SQ1[1] SQ1[2] SQ1[3] SQ1[4] SQ1[5] SQ1[6] SQ1[7]
4B 4B 4B 4B 4B 4B 4B 4B

Figure 4.10 Store Queue Configuration

4.6.2 SQ Writes

A write to the SQs can be performed using a store instruction (MOV) on P4 area H'E000 0000 to H'E3FF FFFC. A longword or quadword access size can be used. The meaning of the address bits is as follows:

[31:26]:	111000	Store queue specification
[25:6]:	Don’t care	Used for external memory transfer/access right
[5]:	0/1	0: SQ0 specification 1: SQ1 specification
[4:2]:	LW specification	Specifies longword position in SQ0/SQ1
[1:0]	00	Fixed at 0

4.6.3 Transfer to External Memory

Transfer from the SQs to external memory can be performed with a prefetch instruction (PREF). Issuing a PREF instruction for P4 area H'E000 0000 to H'E3FF FFFC starts a burst transfer from the SQs to external memory. The burst transfer length is fixed at 32 bytes, and the start address is always at a 32-byte boundary. While the contents of one SQ are being transferred to external memory, the other SQ can be written to without a penalty cycle, but writing to the SQ involved in the transfer to external memory is deferred until the transfer is completed.

The SQ transfer destination external memory address bit [28:0] specification is as shown below, according to whether the MMU is on or off.

- When MMU is on

  The SQ area (H'E000 0000 to H'E3FF FFFF) is set in VPN of the UTLB, and the transfer destination external memory address in PPN. The ASID, V, SZ, SH, PR, and D bits have the same meaning as for normal address translation, but the C and WT bits have no meaning with regard to this page. Since burst transfer is prohibited for PCMCIA areas, the SA and TC bits also have no meaning.

  When a prefetch instruction is issued for the SQ area, address translation is performed and external memory address bits [28:10] are generated in accordance with the SZ bit specification. For external memory address bits [9:5], the address prior to address translation is generated in the same way as when the MMU is off. External memory address bits [4:0] are fixed at 0. Transfer from the SQs to external memory is performed to this address.

- When MMU is off

  The SQ area (H'E000 0000 to H'E3FF FFFF) is specified as the address at which a prefetch is performed. The meaning of address bits [31:0] is as follows:

[31:26]:	111000	Store queue specification
[25:6]:	Address	External memory address bits [25:6]
[5]:	0/1	0: SQ0 specification 1: SQ1 specification and external memory address bit [5]
[4:2]:	Don’t care	No meaning in a prefetch
[1:0]	00	Fixed at 0

  External memory address bits [28:26], which cannot be generated from the above address, are generated from the QACR0/1 registers.

  QACR0 [4:2]: External memory address bits [28:26] corresponding to SQ0

  QACR1 [4:2]: External memory address bits [28:26] corresponding to SQ1

  External memory address bits [4:0] are always fixed at 0 since burst transfer starts at a 32-byte boundary.

4.6.4 SQ Protection

It is possible to set protection against SQ writes and transfers to external memory. If an SQ write violates the protection setting, an exception will be generated but the SQ contents will be corrupted. If a transfer from the SQs to external memory (prefetch instruction) violates the protection setting, the transfer to external memory will be inhibited and an exception will be generated.

- When MMU is on

  Operation is in accordance with the address translation information recorded in the UTLB, and MMUCR.SQMD. Write type exception judgment is performed for writes to the SQs, and read type for transfer from the SQs to external memory (PREF instruction), and a TLB miss exception, protection violation exception, or initial page write exception is generated. However, if SQ access is enabled, in privileged mode only, by MMUCR.SQMD, an address error will be flagged in user mode even if address translation is successful.

- When MMU is off

  Operation is in accordance with MMUCR.SQMD.

  0: Privileged/user access possible

  1: Privileged access possible

  If the SQ area is accessed in user mode when MMUCR.SQMD is set to 1, an address error will be flagged.

Section 5 Exceptions

5.1 Overview

5.1.1 Features

Exception handling is processing handled by a special routine, separate from normal program processing, that is executed by the CPU in case of abnormal events. For example, if the executing instruction ends abnormally, appropriate action must be taken in order to return to the original program sequence, or report the abnormality before terminating the processing. The process of generating an exception handling request in response to abnormal termination, and passing control to a user-written exception handling routine, in order to support such functions, is given the generic name of exception handling.

SH7750 exception handling is of three kinds: for resets, general exceptions, and interrupts.

5.1.2 Register Configuration

The registers used in exception handling are shown in table 5.1.

Table 5.1 Exception-Related Registers

Name	Abbrevia-tion	R/W	Initial Value*1	P4 Address*2	Area 7 Address*2	Access Size
TRAPA exception register	TRA	R/W	Undefined	H'FF00 0020	H'1F00 0020	32
Exception event register	EXPEVT	R/W	H'0000 0000/ H'0000 0020*1	H'FF00 0024	H'1F00 0024	32
Interrupt event register	INTEVT	R/W	Undefined	H'FF00 0028	H'1F00 0028	32

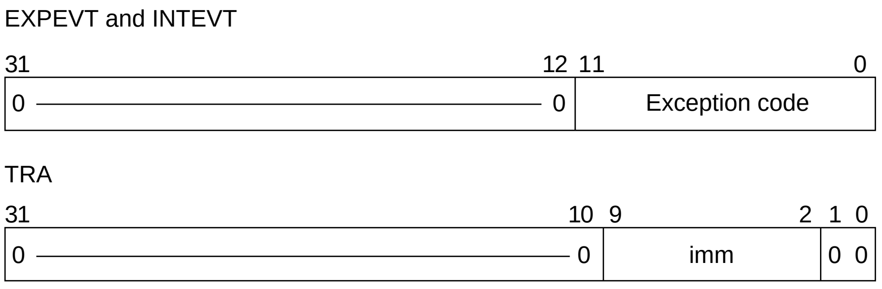
Notes: 1. H'0000 0000 is set in a power-on reset, and H'0000 0020 in a manual reset.
2. This is the address when using the virtual/physical address space P4 area. When making an access from physical address space area 7 using the TLB, the upper 3 bits of the address are ignored.

5.2 Register Descriptions

There are three registers related to exception handling. These are allocated to memory, and can be accessed by specifying the P4 address or area 7 address.

1. The exception event register (EXPEVT) resides at P4 address H'FF00 0024, 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 P4 address H'FF00 0028, and contains a 12-bit exception code. The exception code set in INTEVT is that for an interrupt request. The exception code is set automatically by hardware when an exception occurs. INTEVT can also be modified by software.
3. The TRAPA exception register (TRA) resides at P4 address H'FF00 0020, 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 EXPEVT, INTEVT, and TRA are shown in figure 5.1.



0: Reserved bits. These bits are always read as 0, and should only be written with 0.
imm: 8-bit immediate data of the TRAPA instruction

Figure 5.1 Register Bit Configurations

5.3 Exception Handling Functions

5.3.1 Exception Handling Flow

In exception handling, the contents of the program counter (PC) and status register (SR) are saved in the saved program counter (SPC) and saved status register (SSR), and the CPU starts execution of the appropriate exception handling routine according to the vector address. An exception handling routine is a program written by the user to handle a specific exception. The exception handling routine is terminated and control returned to the original program by executing a return-from-exception instruction (RTE). This instruction restores the PC and SR contents and returns control to the normal processing routine at the point at which the exception occurred.

The basic processing flow is as follows. See section 2, Data Formats and Registers, for the meaning of the individual SR bits.

1. The PC and SR contents are saved in SPC and SSR.
2. The block bit (BL) in SR is set to 1.
3. The mode bit (MD) in SR is set to 1.
4. The register bank bit (RB) in SR is set to 1.
5. In a reset, the FPU disable bit (FD) in SR is cleared to 0.
6. The exception code is written to bits 11–0 of the exception event register (EXPEVT) or interrupt event register (INTEVT).
7. The CPU branches to the determined exception handling vector address, and the exception handling routine begins.

5.3.2 Exception Handling Vector Addresses

The reset vector address is fixed at H'A000 0000. Exception and interrupt vector addresses are determined by adding the offset for the specific event to the vector base address, which is set by software in the vector base register (VBR). In the case of the TLB miss exception, for example, the offset is H'0000 0400, so if H'9C08 0000 is set in VBR, the exception handling vector address will be H'9C08 0400. If a further exception occurs at the exception handling vector address, a duplicate exception will result, and recovery will be difficult; therefore, fixed physical addresses (P1, P2) should be specified for vector addresses.

5.4 Exception Types and Priorities

Table 5.2 shows the types of exceptions, with their relative priorities, vector addresses, and exception/interrupt codes.

Table 5.2 Exceptions

Exception Category	Execution Mode	Exception	Priority Level	Priority Order	Vector Address	Offset	Exception Code
Reset	Abort type	Power-on reset	1	1	H'A000 0000	—	H'000
		Manual reset	1	2	H'A000 0000	—	H'020
		Hitachi-UDI reset	1	1	H'A000 0000	—	H'000
		Instruction TLB multiple-hit exception	1	3	H'A000 0000	—	H'140
		Data TLB multiple-hit exception	1	4	H'A000 0000	—	H'140
General exception	Re-execution type	User break before instruction execution*[1]	2	0	(VBR/DBR)	H'100/—	H'1E0
		Instruction address error	2	1	(VBR)	H'100	H'0E0
		Instruction TLB miss exception	2	2	(VBR)	H'400	H'040
		Instruction TLB protection violation exception	2	3	(VBR)	H'100	H'0A0
		General illegal instruction exception	2	4	(VBR)	H'100	H'180
		Slot illegal instruction exception	2	4	(VBR)	H'100	H'1A0
		General FPU disable exception	2	4	(VBR)	H'100	H'800
		Slot FPU disable exception	2	4	(VBR)	H'100	H'820
		Data address error (read)	2	5	(VBR)	H'100	H'0E0
		Data address error (write)	2	5	(VBR)	H'100	H'100
		Data TLB miss exception (read)	2	6	(VBR)	H'400	H'040
		Data TLB miss exception (write)	2	6	(VBR)	H'400	H'060
		Data TLB protection violation exception (read)	2	7	(VBR)	H'100	H'0A0
		Data TLB protection violation exception (write)	2	7	(VBR)	H'100	H'0C0
		FPU exception	2	8	(VBR)	H'100	H'120
		Initial page write exception	2	9	(VBR)	H'100	H'080
	Completion type	Unconditional trap (TRAPA)	2	4	(VBR)	H'100	H'160
		User break after instruction execution*[1]	2	10	(VBR/DBR)	H'100/—	H'1E0

Table 5.2 Exceptions (cont)

Exception Category	Execution Mode	Exception			Priority Level	Priority Order	Vector Address	Offset	Exception Code
Interrupt	Completion type	Nonmaskable interrupt			3	—	(VBR)	H'600	H'1C0
		External interrupts	IRL3–IRL0	0	4	*2	(VBR)	H'600	H'200
				1					H'220
				2					H'240
				3					H'260
				4					H'280
				5					H'2A0
				6					H'2C0
				7					H'2E0
				8					H'300
				9					H'320
				A					H'340
				B					H'360
				C					H'380
				D					H'3A0
				E					H'3C0
		Peripheral module interrupt (module/ source)	TMU0	TUNI0	4	*2	(VBR)	H'600	H'400
			TMU1	TUNI1					H'420
			TMU2	TUNI2					H'440
				TICPI2					H'460
			RTC	ATI					H'480
				PRI					H'4A0
				CUI					H'4C0
			SCI	ERI					H'4E0
				RXI					H'500
				TXI					H'520
				TEI					H'540
			WDT	ITI					H'560
			REF	RCMI					H'580
				ROVI					H'5A0
			Hitachi-UDI	Hitachi-UDI					H'600
			GPIO	GPIO1					H'620

Table 5.2 Exceptions (cont)

Exception Category	Execution Mode	Exception			Priority Level	Priority Order	Vector Address	Offset	Exception Code
Interrupt	Completion type	Peripheral module interrupt (module/ source)	DMAC	DMTE0	4	*2	(VBR)	H'600	H'640
				DMTE1					H'660
				DMTE2					H'680
				DMTE3					H'6A0
				DMAE					H'6C0
			SCIF	ERI					H'700
				RXI					H'720
				BRI					H'740
				TXI					H'760

Priority: Priority is first assigned by priority level, then by priority order within each level (the lowest number represents the highest priority).

Exception transition destination: Control passes to H'A000 0000 in a reset, and to [VBR + offset] in other cases.

Exception code: Stored in EXPEVT for a reset or general exception, and in INTEVT for an interrupt.

IRL: Interrupt request level (pins IRL3–IRL0).

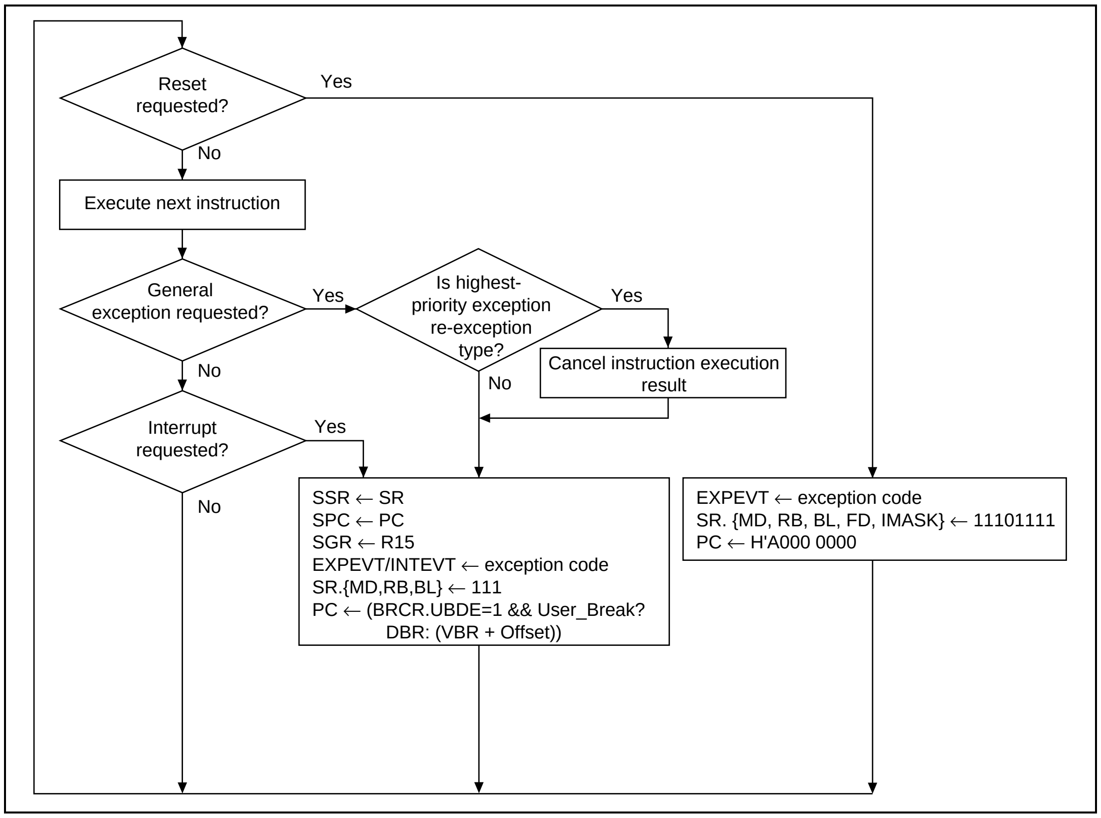
Module/source: See the sections on the relevant peripheral modules.

Notes: 1. When BRCR.UBDE = 1, PC = DBR. In other cases, PC = VBR + H'100.
2. The priority order of external interrupts and peripheral module interrupts can be set by software.

5.5 Exception Flow

5.5.1 Exception Flow

Figure 5.2 shows an outline flowchart of the basic operations in instruction execution and exception handling. For the sake of clarity, the following description assumes that instructions are executed sequentially, one by one. Figure 5.2 shows the relative priority order of the different kinds of exceptions (reset/general exception/interrupt). Register settings in the event of an exception are shown only for SSR, SPC, EXPEVT/INTEVT, SR, and PC, but other registers may be set automatically by hardware, depending on the exception. For details, see section 5.6, Description of Exceptions. Also, see section 5.6.4, Priority Order with Multiple Exceptions, for exception handling during execution of a delayed branch instruction and a delay slot instruction, and in the case of instructions in which two data accesses are performed.



Reset requested?
Yes
No
Execute next instruction
General exception requested?
Yes
Is highest-priority exception re-exception type?
Yes
Cancel instruction execution result
No
No
Interrupt requested?
Yes
No
SSR ← SR
SPC ← PC
SGR ← R15
EXPEVT/INTEVT ← exception code
SR.{MD,RB,BL} ← 111
PC ← (BRCR.UBDE=1 && User_Break? DBR: (VBR + Offset))
EXPEVT ← exception code
SR. {MD, RB, BL, FD, IMASK} ← 11101111
PC ← H'A000 0000

Figure 5.2 Instruction Execution and Exception Handling

5.5.2 Exception Source Acceptance

A priority ranking is provided for all exceptions for use in determining which of two or more simultaneously generated exceptions should be accepted. Five of the general exceptions—the general illegal instruction exception, slot illegal instruction exception, general FPU disable exception, slot FPU disable exception, and unconditional trap exception—are detected in the process of instruction decoding, and do not occur simultaneously in the instruction pipeline. These exceptions therefore all have the same priority. General exceptions are detected in the order of instruction execution. However, exception handling is performed in the order of instruction flow (program order). Thus, an exception for an earlier instruction is accepted before that for a later instruction. An example of the order of acceptance for general exceptions is shown in figure 5.3.

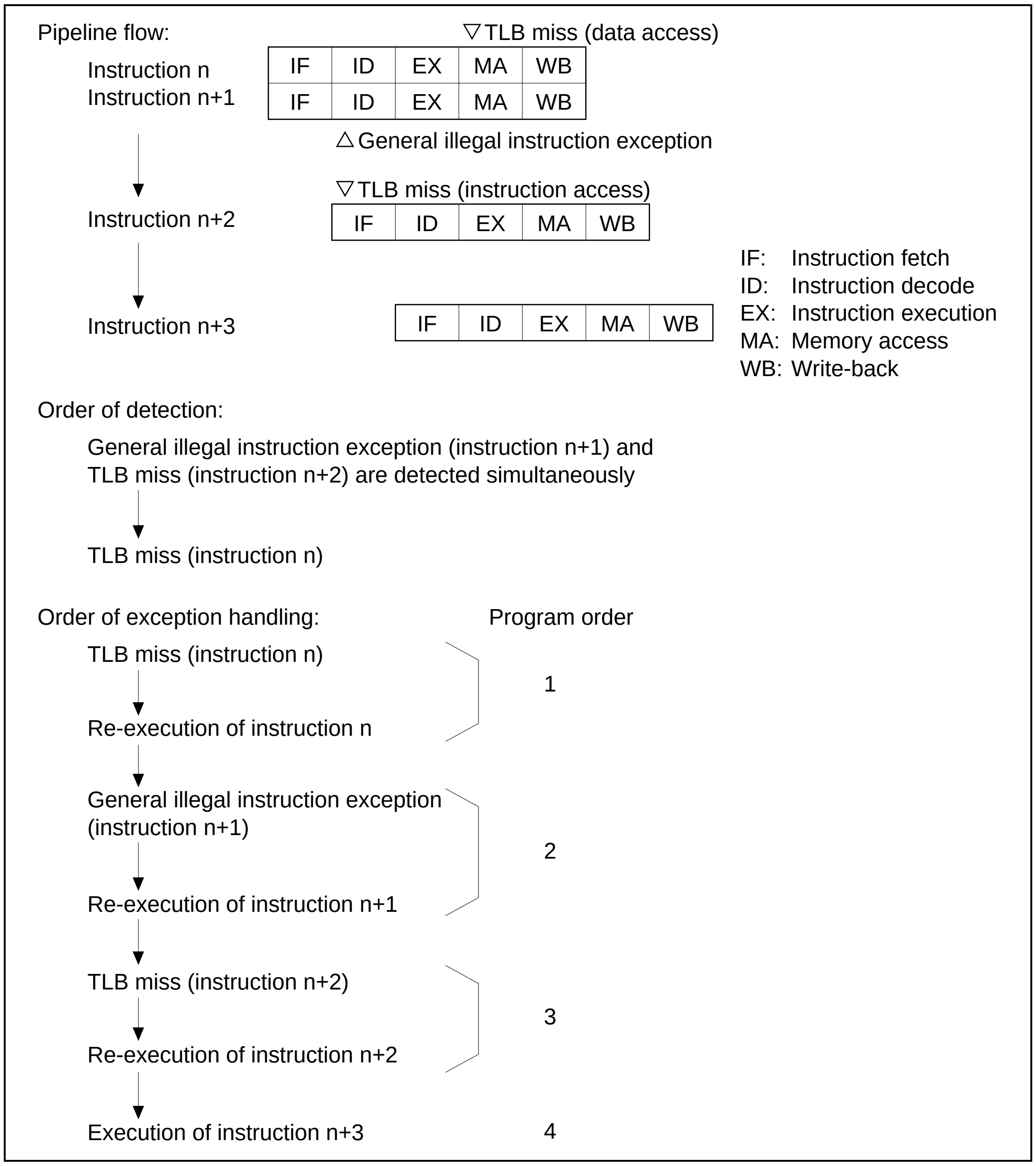

Pipeline flow:
▽TLB miss (data access)
Instruction n
Instruction n+1
IF ID EX MA WB
IF ID EX MA WB
△General illegal instruction exception
▽TLB miss (instruction access)
Instruction n+2
IF ID EX MA WB
Instruction n+3
IF ID EX MA WB
IF: Instruction fetch
ID: Instruction decode
EX: Instruction execution
MA: Memory access
WB: Write-back
Order of detection:
General illegal instruction exception (instruction n+1) and TLB miss (instruction n+2) are detected simultaneously
TLB miss (instruction n)
Order of exception handling:
Program order
TLB miss (instruction n)
Re-execution of instruction n
1
General illegal instruction exception (instruction n+1)
Re-execution of instruction n+1
2
TLB miss (instruction n+2)
Re-execution of instruction n+2
3
Execution of instruction n+3
4

Figure 5.3 Example of General Exception Acceptance Order

5.5.3 Exception Requests and BL Bit

When the BL bit in SR is 0, exceptions and interrupts are accepted.

When the BL bit in SR is 1 and an exception other than a user break is generated, the CPU's internal registers are set to their post-reset state, the registers of the other modules retain their contents prior to the exception, and the CPU branches to the same address as in a reset (H'A000 0000). For the operation in the event of a user break, see section 20, User Break Controller. If an ordinary interrupt occurs, the interrupt request is held pending and is accepted after the BL bit has been cleared to 0 by software. If a nonmaskable interrupt (NMI) occurs, it can be held pending or accepted according to the setting made by software.

Thus, normally, SPC and SSR are saved and then the BL bit in SR is cleared to 0, to enable multiple exception state acceptance.

5.5.4 Return from Exception Handling

The RTE instruction is used to return from exception handling. When the RTE instruction is executed, the SPC contents are restored to PC and the SSR contents to SR, and the CPU returns from the exception handling routine by branching to the SPC address. If SPC and SSR were saved to external memory, set the BL bit in SR to 1 before restoring the SPC and SSR contents and issuing the RTE instruction.

5.6 Description of Exceptions

The various exception handling operations are described here, covering exception sources, transition addresses, and processor operation when a transition is made.

5.6.1 Resets

(1) Power-On Reset

- Sources:
  - — SCK2 pin high level and $\overline{\text{RESET}}$ pin low level
  - — When the watchdog timer overflows while the WT/$\overline{\text{IT}}$ bit is set to 1 and the RSTS bit is cleared to 0 in WTCSR. For details, see section 10, Clock Oscillation Circuits.
- Transition address: H'A000 0000
- Transition operations:

  Exception code H'000 is set in EXPEVT, initialization of VBR and SR is performed, and a branch is made to PC = H'A000 0000.

  In the initialization processing, the VBR register is set to H'0000 0000, and in SR, the MD, RB, and BL bits are set to 1, the FD bit is cleared to 0, and the interrupt mask bits (I3–I0) are set to B'1111.

  CPU and on-chip peripheral module initialization is performed. For details, see the register descriptions in the relevant sections. For some CPU functions, the $\overline{\text{TRST}}$ pin and $\overline{\text{RESET}}$ pin must be driven low. It is therefore essential to execute a power-on reset and drive the $\overline{\text{TRST}}$ pin low when powering on.

```
Power_on_reset()
{
   EXPEVT = H'00000000;
   VBR = H'00000000;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   SR.(I0-I3) = B'1111;
   SR.FD=0;
   Initialize_CPU();
   Initialize_Module(PowerOn);
   PC = H'A0000000;
}
```

(2) Manual Reset

- Sources:
  - SCK2 pin low level and $\overline{\text{RESET}}$ pin low level
  - When a general exception other than a user break occurs while the BL bit is set to 1 in SR
  - When the watchdog timer overflows while the RSTS bit is set to 1 in WTCSR. For details, see section 10, Clock Oscillation Circuits.
- Transition address: H'A000 0000
- Transition operations:

  Exception code H'020 is set in EXPEVT, initialization of VBR and SR is performed, and a branch is made to PC = H'A000 0000.

  In the initialization processing, the VBR register is set to H'0000 0000, and in SR, the MD, RB, and BL bits are set to 1, the FD bit is cleared to 0, and the interrupt mask bits (I3–I0) are set to B'1111.

  CPU and on-chip peripheral module initialization is performed. For details, see the register descriptions in the relevant sections.

```
Manual_reset()
{
   EXPEVT = H'00000020;
   VBR = H'00000000;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   SR.(I0-I3) = B'1111;
   SR.FD = 0;
   Initialize_CPU();
   Initialize_Module(Manual);
   PC = H'A0000000;
}
```

Table 5-3 Types of Reset

	Reset State Transition Conditions		Internal States	
Type	**$\overline{\text{SCK2}}$**	**$\overline{\text{RESET}}$**	**CPU**	**On-Chip Peripheral Modules**
Power-on reset	High	Low	Initialized	See Register Configuration in each section
Manual reset	Low	Low	Initialized	

(3) Hitachi-UDI Reset

- Source: SDIR.TI3–TI0 = B'0110 (negation) or B'0111 (assertion)
- Transition address: H'A000 0000
- Transition operations:

  Exception code H'000 is set in EXPEVT, initialization of VBR and SR is performed, and a branch is made to PC = H'A000 0000.

  In the initialization processing, the VBR register is set to H'0000 0000, and in SR, the MD, RB, and BL bits are set to 1, the FD bit is cleared to 0, and the interrupt mask bits (I3–I0) are set to B'1111.

  CPU and on-chip peripheral module initialization is performed. For details, see the register descriptions in the relevant sections.

```
Hitachi-UDI_reset()
{
   EXPEVT = H'00000000;
   VBR = H'00000000;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   SR.(I0-I3) = B'1111;
   SR.FD = 0;
   Initialize_CPU();
   Initialize_Module(PowerOn);
   PC = H'A0000000;
}
```

(4) Instruction TLB Multiple-Hit Exception

- Source: Multiple ITLB address matches
- Transition address: H'A000 0000
- Transition operations:

  The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

  Exception code H'140 is set in EXPEVT, initialization of VBR and SR is performed, and a branch is made to PC = H'A000 0000.

  In the initialization processing, the VBR register is set to H'0000 0000, and in SR, the MD, RB, and BL bits are set to 1, the FD bit is cleared to 0, and the interrupt mask bits (I3–I0) are set to B'1111.

  CPU and on-chip peripheral module initialization is performed in the same way as in a manual reset. For details, see the register descriptions in the relevant sections.

```
TLB_multi_hit()
{
   TEA = EXCEPTION_ADDRESS;
   PTEH.VPN = PAGE_NUMBER;
   EXPEVT = H'00000140;
   VBR = H'00000000;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   SR.(I0-I3) = B'1111;
   SR.FD = 0;
   Initialize_CPU();
   Initialize_Module(Manual);
   PC = H'A0000000;
}
```

(5) Operand TLB Multiple-Hit Exception

- Source: Multiple UTLB address matches
- Transition address: H'A000 0000
- Transition operations:

  The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

  Exception code H'140 is set in EXPEVT, initialization of VBR and SR is performed, and a branch is made to PC = H'A000 0000.

  In the initialization processing, the VBR register is set to H'0000 0000, and in SR, the MD, RB, and BL bits are set to 1, the FD bit is cleared to 0, and the interrupt mask bits (I3–I0) are set to B'1111.

  CPU and on-chip peripheral module initialization is performed in the same way as in a manual reset. For details, see the register descriptions in the relevant sections.

```
TLB_multi_hit()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    EXPEVT = H'00000140;
    VBR = H'00000000;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    SR.(I0-I3) = B'1111;
    SR.FD = 0;
    Initialize_CPU();
    Initialize_Module(Manual);
    PC = H'A0000000;
}
```

5.6.2 General Exceptions

(1) Data TLB Miss Exception

- Source: Address mismatch in UTLB address comparison
- Transition address: VBR + H'0000 0400
- Transition operations:

  The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

  The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR.

  Exception code H'040 (for a read access) or H'060 (for a write access) is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0400.

  To speed up TLB miss processing, the offset is separate from that of other exceptions.

```
Data_TLB_miss_exception()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    SPC = PC;
    SSR = SR;
    EXPEVT = read_access ? H'00000040 : H'00000060;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000400;
}
```

(2) Instruction TLB Miss Exception

- Source: Address mismatch in ITLB address comparison
- Transition address: VBR + H'0000 0400
- Transition operations:

  The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

  The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR.

  Exception code H'040 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0400.

  To speed up TLB miss processing, the offset is separate from that of other exceptions.

```
ITLB_miss_exception()
{
   TEA = EXCEPTION_ADDRESS;
   PTEH.VPN = PAGE_NUMBER;
   SPC = PC;
   SSR = SR;
   EXPEVT = H'00000040;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000400;
}
```

(3) Initial Page Write Exception

- Source: TLB is hit in a store access, but dirty bit D = 0
- Transition address: VBR + H'0000 0100
- Transition operations:

  The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

  The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR.

  Exception code H'080 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
Initial_write_exception()
{
   TEA = EXCEPTION_ADDRESS;
   PTEH.VPN = PAGE_NUMBER;
   SPC = PC;
   SSR = SR;
   EXPEVT = H'00000080;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000100;
}
```

(4) Data TLB Protection Violation Exception

- Source: The access does not accord with the UTLB protection information (PR bits) shown below.

PR	Privileged Mode	User Mode
00	Only read access possible	Access not possible
01	Read/write access possible	Access not possible
10	Only read access possible	Only read access possible
11	Read/write access possible	Read/write access possible

- Transition address: VBR + H'0000 0100
- Transition operations:

  The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

  The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR.

  Exception code H'0A0 (for a read access) or H'0C0 (for a write access) is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
Data_TLB_protection_violation_exception()
{
   TEA = EXCEPTION_ADDRESS;
   PTEH.VPN = PAGE_NUMBER;
   SPC = PC;
   SSR = SR;
   EXPEVT = read_access ? H'000000A0 : H'000000C0;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000100;
}
```

(5) Instruction TLB Protection Violation Exception

- Source: The access does not accord with the ITLB protection information (PR bits) shown below.

PR	Privileged Mode	User Mode
0	Access possible	Access not possible
1	Access possible	Access possible

- Transition address: VBR + H'0000 0100
- Transition operations:

  The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

  The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR.

  Exception code H'0A0 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
ITLB_protection_violation_exception()
{
    TEA = EXCEPTION_ADDRESS;
    PTEH.VPN = PAGE_NUMBER;
    SPC = PC;
    SSR = SR;
    EXPEVT = H'000000A0;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000100;
}
```

(6) Data Address Error

- Sources:
  - Word data access from other than a word boundary (2n +1)
  - Longword data access from other than a longword data boundary (4n +1, 4n + 2, or 4n +3)
  - Quadword data access from other than a quadword data boundary (8n +1, 8n + 2, 8n +3, 8n + 4, 8n + 5, 8n + 6, or 8n + 7)
  - Access to area H'8000 0000–H'FFFF FFFF in user mode
- Transition address: VBR + H'0000 0100
- Transition operations:

  The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

  The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR.

  Exception code H'0E0 (for a read access) or H'100 (for a write access) is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100. For details, see section 3, Memory Management Unit (MMU).

```
Data_address_error()
{
    TEA = EXCEPTION_ADDRESS;
    PTEN.VPN = PAGE_NUMBER;
    SPC = PC;
    SSR = SR;
    EXPEVT = read_access? H'000000E0: H'00000100;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000100;
}
```

(7) Instruction Address Error

- Sources:
  - Instruction fetch from other than a word boundary (2n +1)
  - Instruction fetch from area H'8000 0000–H'FFFF FFFF in user mode
- Transition address: VBR + H'0000 0100
- Transition operations:

  The virtual address (32 bits) at which this exception occurred is set in TEA, and the corresponding virtual page number (22 bits) is set in PTEH [31:10]. ASID in PTEH indicates the ASID when this exception occurred.

  The PC and SR contents for the instruction at which this exception occurred are saved in the SPC and SSR.

  Exception code H'0E0 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100. For details, see section 3, Memory Management Unit (MMU).

```
Instruction_address_error()
{
   TEA = EXCEPTION_ADDRESS;
   PTEN.VPN = PAGE_NUMBER;
   SPC = PC;
   SSR = SR;
   EXPEVT = H'000000E0;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000100;
}
```

(8) Unconditional Trap

- Source: Execution of TRAPA instruction
- Transition address: VBR + H'0000 0100
- Transition operations:

  As this is a processing-completion-type exception, the PC contents for the instruction following the TRAPA instruction are saved in SPC. The value of SR when the TRAPA instruction is executed are saved in SSR. The 8-bit immediate value in the TRAPA instruction is multiplied by 4, and the result is set in TRA [9]. Exception code H'160 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
TRAPA_exception()
{
   SPC = PC + 2;
   SSR = SR;
   TRA = imm << 2;
   EXPEVT = H'00000160;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000100;
}
```

(9) General Illegal Instruction Exception

- Sources:
  - Decoding of an undefined instruction not in a delay slot

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

    Undefined instruction: H'FFFD
  - Decoding in user mode of a privileged instruction not in a delay slot

    Privileged instructions: LDC, STC, RTE, LDTLB, SLEEP, but excluding LDC/STC instructions that access GBR
- Transition address: VBR + H'0000 0100
- Transition operations:

  The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR.

  Exception code H'180 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100. Operation is not guaranteed if an undefined code other than H'FFFD is decoded.

```
General_illegal_instruction_exception()
{
   SPC = PC;
   SSR = SR;
   EXPEVT = H'00000180;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000100;
}
```

(10) Slot Illegal Instruction Exception

- Sources:
  - Decoding of an undefined instruction in a delay slot

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

    Undefined instruction: H'FFFD
  - Decoding of an instruction that modifies PC in a delay slot

    Instructions that modify PC: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT, BF, BT/S, BF/S, TRAPA, LDC Rm, SR, LDC.L @Rm+, SR
  - Decoding in user mode of a privileged instruction in a delay slot

    Privileged instructions: LDC, STC, RTE, LDTLB, SLEEP, but excluding LDC/STC instructions that access GBR
  - Decoding of a PC-relative MOV instruction or MOVA instruction in a delay slot
- Transition address: VBR + H'0000 0100
- Transition operations:

  The PC contents for the preceding delayed branch instruction are saved in SPC. The SR contents when this exception occurred are saved in SSR.

  Exception code H'1A0 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100. Operation is not guaranteed if an undefined code other than H'FFFD is decoded.

```
Slot_illegal_instruction_exception()
{
   SPC = PC - 2;
   SSR = SR;
   EXPEVT = H'000001A0;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000100;
}
```

(11) General FPU Disable Exception

- Source: Decoding of an FPU instruction* not in a delay slot with SR.FD =1
- Transition address: VBR + H'0000 0100
- Transition operations:

  The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR.

  Exception code H'800 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

Note: * FPU instructions are instructions in which the first 4 bits of the instruction code are F (but excluding undefined instruction H'FFFD), and the LDS, STS, LDS.L, and STS.L instructions corresponding to FPUL and FPSCR.

```
General_fpu_disable_exception()
{
   SPC = PC;
   SSR = SR;
   EXPEVT = H'00000800;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000100;
}
```

(12) Slot FPU Disable Exception

- Source: Decoding of an FPU instruction in a delay slot with SR.FD =1
- Transition address: VBR + H'0000 0100
- Transition operations:

  The PC contents for the preceding delayed branch instruction are saved in SPC. The SR contents when this exception occurred are saved in SSR.

  Exception code H'820 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
Slot_fpu_disable_exception()
{
    SPC = PC - 2;
    SSR = SR;
    EXPEVT = H'00000820;
    SR.MD = 1;
    SR.RB = 1;
    SR.BL = 1;
    PC = VBR + H'00000100;
}
```

(13) User Breakpoint Trap

- Source: Fulfilling of a break condition set in the user break controller
- Transition address: VBR + H'0000 0100, or DBR
- Transition operations:

  In the case of a post-execution break, the PC contents for the instruction following the instruction at which the breakpoint is set are set in SPC. In the case of a pre-execution break, the PC contents for the instruction at which the breakpoint is set are set in SPC.

  The SR contents when the break occurred are saved in SSR. Exception code H'1E0 is set in EXPEVT.

  The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100. It is also possible to branch to PC = DBR.

  For details of PC, etc., when a data break is set, see section 20, User Break Controller.

```
User_break_exception()
{
   SPC = (pre_execution break? PC : PC + 2);
   SSR = SR;
   EXPEVT = H'000001E0;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = (BRCR.UBDE==1 ? DBR : VBR + H’00000100);
}
```

(14) FPU Exception

- Source: Exception due to execution of a floating-point operation
- Transition address: VBR + H'0000 0100
- Transition operations:
  The PC and SR contents for the instruction at which this exception occurred are saved in SPC and SSR. Exception code H'120 is set in EXPEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0100.

```
FPU_exception()
{
   SPC = PC;
   SSR = SR;
   EXPEVT = H'00000120;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000100;
}
```

5.6.3 Interrupts

(1) NMI

- Source: NMI pin edge detection
- Transition address: VBR + H'0000 0600
- Transition operations:

  The PC and SR contents for the instruction at which this exception is accepted are saved in SPC and SSR.

  Exception code H'1C0 is set in INTEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to PC = VBR + H'0600. When the BL bit in SR is 0, this interrupt is not masked by the interrupt mask bits in SR, and is accepted at the highest priority level. When the BL bit in SR is 1, a software setting can specify whether this interrupt is to be masked or accepted. For details, see section 19, Interrupt Controller.

```
NMI()
{
   SPC = PC;
   SSR = SR;
   INTEVT = H'000001C0;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000600;
}
```

(2) IRL Interrupts

- Source: The interrupt mask bit setting in SR is smaller than the IRL (3–0) level, and the BL bit in SR is 0 (accepted at instruction boundary).
- Transition address: VBR + H'0000 0600
- Transition operations:

  The PC contents immediately after the instruction at which the interrupt is accepted are set in SPC. The SR contents at the time of acceptance are set in SSR.

  The code corresponding to the IRL (3–0) level is set in INTEVT. See table 19.5, Interrupt Exception Handling Sources and Priority Order, for the corresponding codes. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to VBR + H'0600. The acceptance level is not set in the interrupt mask bits in SR. When the BL bit in SR is 1, the interrupt is masked. For details, see section 19, Interrupt Controller.

```
IRL()
{
   SPC = PC;
   SSR = SR;
   INTEVT = H'00000200 ~ H'000003C0;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000600;
}
```

(3) Peripheral Module Interrupts

- Source: The interrupt mask bit setting in SR is smaller than the peripheral module (Hitachi-UDI, GPIO, DMAC, TMU, RTC, SCI, SCIF, WDT, or REF) interrupt level, and the BL bit in SR is 0 (accepted at instruction boundary).
- Transition address: VBR + H'0000 0600
- Transition operations:

  The PC contents immediately after the instruction at which the interrupt is accepted are set in SPC. The SR contents at the time of acceptance are set in SSR.

  The code corresponding to the interrupt source is set in INTEVT. The BL, MD, and RB bits are set to 1 in SR, and a branch is made to VBR + H'0600. The module interrupt levels should be set as values between B'0000 and B'1111 in the interrupt priority registers (IPRA–IPRC) in the interrupt controller. For details, see section 19, Interrupt Controller.

```
Module_interruption()
{
   SPC = PC;
   SSR = SR;
   INTEVT = H'00000400 ~ H'00000760;
   SR.MD = 1;
   SR.RB = 1;
   SR.BL = 1;
   PC = VBR + H'00000600;
}
```

5.6.4 Priority Order with Multiple Exceptions

With some instructions, such as instructions that make two accesses to memory, and the indivisible pair comprising a delayed branch instruction and delay slot instruction, multiple exceptions occur. Care is required in these cases, as the exception priority order differs from the normal order.

1. Instructions that make two accesses to memory

   With MAC instructions, memory-to-memory arithmetic/logic instructions, and TAS instructions, two data transfers are performed by a single instruction, and an exception will be detected for each of these data transfers. In these cases, therefore, the following order is used to determine priority.

   a. Data address error in first data transfer
   b. TLB miss in first data transfer
   c. TLB protection violation in first data transfer
   d. Initial page write exception in first data transfer
   e. Data address error in second data transfer
   f. TLB miss in second data transfer
   g. TLB protection violation in second data transfer
   h. Initial page write exception in second data transfer

2. Indivisible delayed branch instruction and delay slot instruction

   As a delayed branch instruction and its associated delay slot instruction are indivisible, they are treated as a single instruction. Consequently, the priority order for exceptions that occur in these instructions differs from the usual priority order. The priority order shown below is for the case where the delay slot instruction has only one data transfer.

   a. The delayed branch instruction is checked for priority levels 1 and 2.
   b. The delay slot instruction is checked for priority levels 1 and 2.
   c. A check is performed for priority level 3 in the delayed branch instruction and priority level 3 in the delay slot instruction. (There is no priority ranking between these two.)
   d. A check is performed for priority level 4 in the delayed branch instruction and priority level 4 in the delay slot instruction. (There is no priority ranking between these two.)

   If the delay slot instruction has a second data transfer, two checks are performed in step b, as in 1 above.

   If the accepted exception (the highest-priority exception) is a delay slot instruction re-execution type exception, the branch instruction PR register write operation (PC $\rightarrow$ PR operation performed in BSR, BSRF, JSR) is inhibited.

5.7 Usage Notes

1. Return from exception handling
   a. Check the BL bit in SR with software. If SPC and SSR have been saved to external memory, set the BL bit in SR to 1 before restoring them.
   b. Issue an RTE instruction. When RTE is executed, the SPC contents are set in PC, the SSR contents are set in SR, and branch is made to the SPC address to return from the exception handling routine.

2. If an exception or interrupt occurs when SR.BL = 1
   a. Exception

      When an exception other than a user break occurs, the CPU's internal registers are set to their post-reset state, the registers of the other modules retain their contents prior to the exception, and the CPU branches to the same address as in a reset (H'A000 0000). The value in EXPEVT at this time is H'0000 0020; the value of the SPC and SSR registers is undefined.
   b. Interrupt

      If an ordinary interrupt occurs, the interrupt request is held pending and is accepted after the BL bit in SR has been cleared to 0 by software. If a nonmaskable interrupt (NMI) occurs, it can be held pending or accepted according to the setting made by software. In the sleep or standby state, however, an interrupt is accepted even if the BL bit in SR is set to 1.

3. SPC when an exception occurs
   a. Re-execution type exception

      The PC value for the instruction in which the exception occurred is set in SPC, and the instruction is re-executed after returning from exception handling. If an exception occurs in a delay slot instruction, however, the PC value for the delay slot instruction is saved in SPC regardless of whether or not the preceding delay slot instruction condition is satisfied.
   b. Completion type exception or interrupt

      The PC value for the instruction following that in which the exception occurred is set in SPC. If an exception occurs in a branch instruction with delay slot, however, the PC value for the branch destination is saved in SPC.

4. An exception must not be generated in an RTE instruction delay slot, as the operation will be undefined in this case.

5.8 Restrictions

1. Restrictions on first instruction of exception handling routine

- Do not locate a BT, BF, BT/S, BF/S, BRA, or BSR instruction at address VBR + H'100, VBR + H'400, or VBR + H'600.
- When the UBDE bit in the BRCR register is set to 1 and the user break debug support function* is used, do not locate a BT, BF, BT/S, BF/S, BRA, or BSR instruction at the address indicated by the DBR register.

Note: * See section 20.4.

Section 6 Floating-Point Unit

6.1 Overview

The floating-point unit (FPU) has the following features:

- Conforms to IEEE754 standard
- 32 single-precision floating-point registers (can also be referenced as 16 double-precision registers)
- Two rounding modes: Round to Nearest and Round to Zero
- Two denormalization modes: Flush to Zero and Treat Denormalized Number
- Six exception sources: FPU Error, Invalid Operation, Divide By Zero, Overflow, Underflow, and Inexact
- Comprehensive instructions: Single-precision, double-precision, graphics support, system control

When the FD bit in SR is set to 1, the FPU cannot be used, and an attempt to execute an FPU instruction will cause an FPU disable exception.

6.2 Data Formats

6.2.1 Floating-Point Format

A floating-point number consists of the following three fields:

- Sign (s)
- Exponent (e)
- Fraction (f)

The SH7750 can handle single-precision and double-precision floating-point numbers, using the formats shown in figures 6.1 and 6.2.

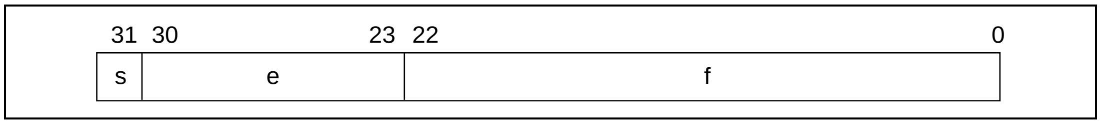


Figure 6.1 Format of Single-Precision Floating-Point Number

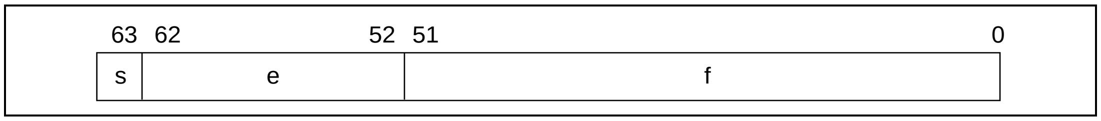


Figure 6.2 Format of Double-Precision Floating-Point Number

The exponent is expressed in biased form, as follows:

e = E + bias

The range of unbiased exponent E is $E_{min} – 1$ to $E_{max} + 1$. The two values $E_{min} – 1$ and $E_{max} + 1$ are distinguished as follows. $E_{min} – 1$ indicates zero (both positive and negative sign) and a denormalized number, and $E_{max} + 1$ indicates positive or negative infinity or a non-number (NaN). Table 6.1 shows bias, E_{min}, and E_{max} values.

Table 6.1 Floating-Point Number Formats and Parameters

Parameter	**Single-Precision**	**Double-Precision**
Total bit width	32 bits	64 bits
Sign bit	1 bit	1 bit
Exponent field	8 bits	11 bits
Fraction field	23 bits	52 bits
Precision	24 bits	53 bits
Bias	+127	+1023
E_{max}	+127	+1023
E_{min}	–126	–1022

Floating-point number value v is determined as follows:

If $E = E_{max} + 1$ and $f \neq 0$, v is a non-number (NaN) irrespective of sign s
If $E = E_{max} + 1$ and $f = 0$, $v = (–1)^s$ (infinity) [positive or negative infinity]
If $E_{min} \leq E \leq E_{max}$, $v = (–1)^s 2^E$ (1.f) [normalized number]
If $E = E_{min} – 1$ and $f \neq 0$, $v = (–1)^s 2^{Emin}$ (0.f) [denormalized number]
If $E = E_{min} – 1$ and $f = 0$, $v = (–1)^s 0$ [positive or negative zero]

Table 6.2 shows the ranges of the various numbers in hexadecimal notation.

Table 6.2 Floating-Point Ranges

Type	Single-Precision	Double-Precision
Signaling non-number	H'7FFFFFFF to H'7FC00000	H'7FFFFFFF H'FFFFFFFF to H'7FF80000 H'00000000
Quiet non-number	H'7FBFFFFF to H'7F800001	H'7FF7FFFF H'FFFFFFFF to H'7FF00000 H'00000001
Positive infinity	H'7F800000	H'7FF00000 H'00000
Positive normalized number	H'7F7FFFFF to H'00800000	H'7FEFFFFF H'FFFFFFFF to H'00100000 H'00000000
Positive denormalized number	H'007FFFFF to H'00000001	H'000FFFFF H'FFFFFFFF to H'00000000 H'00000001
Positive zero	H'00000000	H'00000000 H'00000000
Negative zero	H'80000000	H'80000000 H'00000000
Negative denormalized number	H'80000001 to H'807FFFFF	H'80000000 H'00000001 to H'800FFFFF H'FFFFFFFF
Negative normalized number	H'80800000 to H'FF7FFFFF	H'80100000 H'00000000 to H'FFEFFFFF H'FFFFFFFF
Negative infinity	H'FF800000	H'FFF00000 H'00000000
Quiet non-number	H'FF800001 to H'FFBFFFFF	H'FFF00000 H'00000001 to H'FFF7FFFF H'FFFFFFFF
Signaling non-number	H'FFC00000 to H'FFFFFFFF	H'FFF80000 H'00000000 to H'FFFFFFFF H'FFFFFFFF

6.2.2 Non-Numbers (NaN)

Figure 6.3 shows the bit pattern of a non-number (NaN). A value is NaN in the following case:

- Sign bit: Don’t care
- Exponent field: All bits are 1
- Fraction field: At least one bit is 1

The NaN is a signaling NaN (sNaN) if the MSB of the fraction field is 1, and a quiet NaN (qNaN) if the MSB is 0.

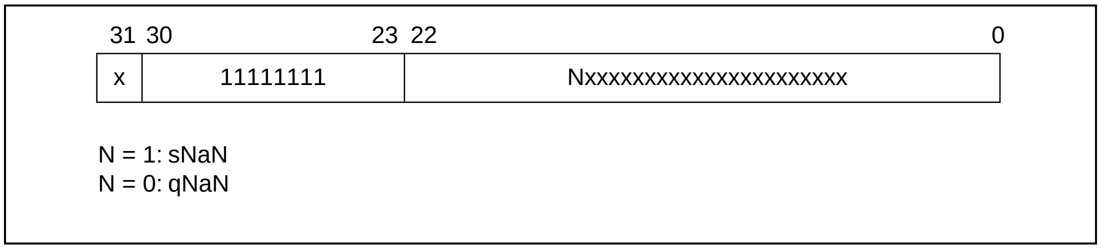


Figure 6.3 Single-Precision NaN Bit Pattern

An sNAN is input in an operation, except copy, FABS, and FNEG, that generates a floating-point value.

- When the EN.V bit in the FPSCR register is 0, the operation result (output) is a qNaN.
- When the EN.V bit in the FPSCR register is 1, an invalid operation exception will be generated. In this case, the contents of the operation destination register are unchanged.

If a qNaN is input in an operation that generates a floating-point value, and an sNaN has not been input in that operation, the output will always be a qNaN irrespective of the setting of the EN.V bit in the FPSCR register. An exception will not be generated in this case.

The qNAN values generated by the SH7750 as operation results are as follows:

- Single-precision qNaN: H'7FBFFFFF
- Double-precision qNaN: H'7FF7FFFF FFFFFFFF

See the individual instruction descriptions for details of floating-point operations when a non-number (NaN) is input.

6.2.3 Denormalized Numbers

For a denormalized number floating-point value, the exponent field is expressed as 0, and the fraction field as a non-zero value.

When the DN bit in the FPU’s status register FPSCR is 1, a denormalized number (source operand or operation result) is always flushed to 0 in a floating-point operation that generates a value (an operation other than copy, FNEG, or FABS).

When the DN bit in FPSCR is 0, a denormalized number (source operand or operation result) is processed as it is. See the individual instruction descriptions for details of floating-point operations when a denormalized number is input.

6.3 Registers

6.3.1 Floating-Point Registers

Figure 6.4 shows the floating-point register configuration. There are thirty-two 32-bit floating-point registers, referenced by specifying FR0–FR15, DR0/2/4/6/8/10/12/14, FV0/4/8/12, XF0–XF15, XD0/2/4/6/8/10/12/14, or XMTRX.

1. Floating-point registers, FPRi_BANKj (32 registers)
   FPR0_BANK0–FPR15_BANK0
   FPR0_BANK1–FPR15_BANK1

2. Single-precision floating-point registers, FRi (16 registers)
   When FPSCR.FR = 0, FR0–FR15 indicate FPR0_BANK0–FPR15_BANK0;
   when FPSCR.FR = 1, FR0–FR15 indicate FPR0_BANK1–FPR15_BANK1.

3. Double-precision floating-point registers, DRi (8 registers): A DR register comprises two FR registers
   DR0 = {FR0, FR1}, DR2 = {FR2, FR3}, DR4 = {FR4, FR5}, DR6 = {FR6, FR7},
   DR8 = {FR8, FR9}, DR10 = {FR10, FR11}, DR12 = {FR12, FR13}, DR14 = {FR14, FR15}

4. Single-precision floating-point vector registers, FVi (4 registers): An FV register comprises four FR registers
   FV0 = {FR0, FR1, FR2, FR3}, FV4 = {FR4, FR5, FR6, FR7},
   FV8 = {FR8, FR9, FR10, FR11}, FV12 = {FR12, FR13, FR14, FR15}

5. Single-precision floating-point extended registers, XFi (16 registers)
   When FPSCR.FR = 0, XF0–XF15 indicate FPR0_BANK1–FPR15_BANK1;
   when FPSCR.FR = 1, XF0–XF15 indicate FPR0_BANK0–FPR15_BANK0.

6. Double-precision floating-point extended registers, XDi (8 registers): An XD register comprises two XF registers
   XD0 = {XF0, XF1}, XD2 = {XF2, XF3}, XD4 = {XF4, XF5}, XD6 = {XF6, XF7},
   XD8 = {XF8, XF9}, XD10 = {XF10, XF11}, XD12 = {XF12, XF13}, XD14 = {XF14, XF15}

7. Single-precision floating-point extended register matrix, XMTRX: XMTRX comprises all 16 XF registers

$$\text{XMTRX} = \begin{bmatrix} \text{XF0} & \text{XF4} & \text{XF8} & \text{XF12} \\ \text{XF1} & \text{XF5} & \text{XF9} & \text{XF13} \\ \text{XF2} & \text{XF6} & \text{XF10} & \text{XF14} \\ \text{XF3} & \text{XF7} & \text{XF11} & \text{XF15} \end{bmatrix}$$

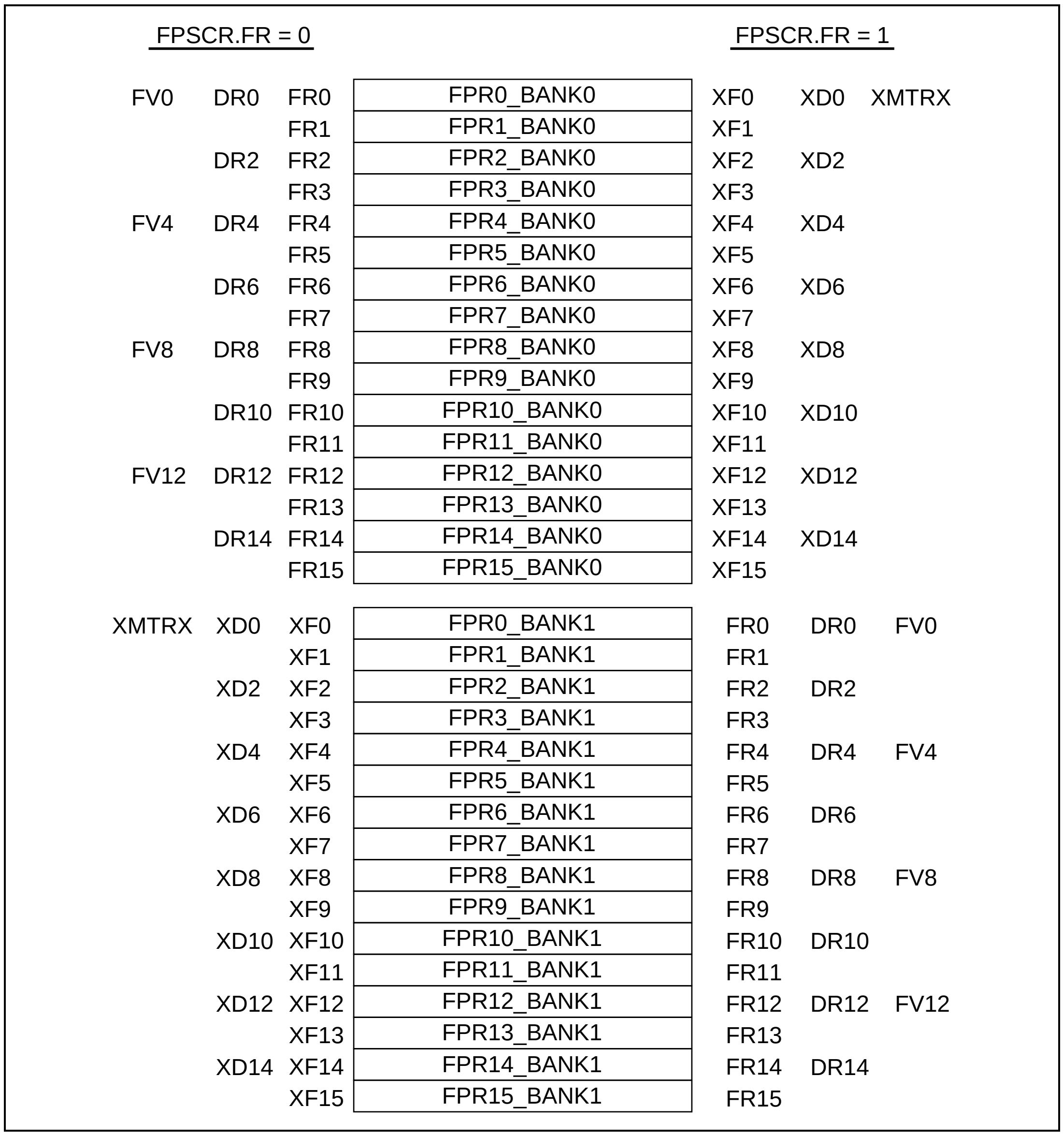

FPSCR.FR = 0
FPSCR.FR = 1
FV0 DR0 FR0 FPR0_BANK0 XF0 XD0 XMTRX
FR1 FPR1_BANK0 XF1
DR2 FR2 FPR2_BANK0 XF2 XD2
FR3 FPR3_BANK0 XF3
FV4 DR4 FR4 FPR4_BANK0 XF4 XD4
FR5 FPR5_BANK0 XF5
DR6 FR6 FPR6_BANK0 XF6 XD6
FR7 FPR7_BANK0 XF7
FV8 DR8 FR8 FPR8_BANK0 XF8 XD8
FR9 FPR9_BANK0 XF9
DR10 FR10 FPR10_BANK0 XF10 XD10
FR11 FPR11_BANK0 XF11
FV12 DR12 FR12 FPR12_BANK0 XF12 XD12
FR13 FPR13_BANK0 XF13
DR14 FR14 FPR14_BANK0 XF14 XD14
FR15 FPR15_BANK0 XF15
XMTRX XD0 XF0 FPR0_BANK1 FR0 DR0 FV0
XF1 FPR1_BANK1 FR1
XD2 XF2 FPR2_BANK1 FR2 DR2
XF3 FPR3_BANK1 FR3
XD4 XF4 FPR4_BANK1 FR4 DR4 FV4
XF5 FPR5_BANK1 FR5
XD6 XF6 FPR6_BANK1 FR6 DR6
XF7 FPR7_BANK1 FR7
XD8 XF8 FPR8_BANK1 FR8 DR8 FV8
XF9 FPR9_BANK1 FR9
XD10 XF10 FPR10_BANK1 FR10 DR10
XF11 FPR11_BANK1 FR11
XD12 XF12 FPR12_BANK1 FR12 DR12 FV12
XF13 FPR13_BANK1 FR13
XD14 XF14 FPR14_BANK1 FR14 DR14
XF15 FPR15_BANK1 FR15

Figure 6.4 Floating-Point Registers

6.3.2 Floating-Point Status/Control Register (FPSCR)

Floating-point status/control register, FPSCR (32 bits, initial value = H'0004 0001)

- FR: Floating-point register bank

  FR = 0: FPR0_BANK0–FPR15_BANK0 are assigned to FR0–FR15; FPR0_BANK1–FPR15_BANK1 are assigned to XF0–XF15.

  FR = 1: FPR0_BANK0–FPR15_BANK0 are assigned to XF0–XF15; FPR0_BANK1–FPR15_BANK1 are assigned to FR0–FR15.

- SZ: Transfer size mode

  SZ = 0: The data size of the FMOV instruction is 32 bits.

  SZ = 1: The data size of the FMOV instruction is a 32-bit register pair (64 bits).

- PR: Precision mode

  PR = 0: Floating-point instructions are executed as single-precision operations.

  PR = 1: Floating-point instructions are executed as double-precision operations (graphics support instructions are undefined).

  Do not set SZ and PR to 1 simultaneously; this setting is reserved.

  [SZ, PR = 11]: Reserved (FPU operation instruction is undefined.)

- DN: Denormalization mode

  DN = 0: A denormalized number is treated as such.

  DN = 1: A denormalized number is treated as zero.

		FPU Error (E)	**Invalid Operation (V)**	**Division by Zero (Z)**	**Overflow (O)**	**Underflow (U)**	**Inexact (I)**
Cause	FPU exception cause field	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12
Enable	FPU exception enable field	None	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7
Flag	FPU exception flag field	None	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2

When an FPU exception is requested, the corresponding bits in the cause and flag fields are set to 1. Each time an FPU operation instruction is executed, the cause field is cleared to 0 first. The flag field retains the value of 1 until cleared to 0 by software.

- RM: Rounding mode
  RM = 00: Round to Nearest
  RM = 01: Round to Zero
  RM = 10: Reserved
  RM = 11: Reserved

- Bits 22 to 31: Reserved
  These bits are always read as 0, and should only be written with 0.

Notes: The following functions have been added to the FPU of the SH7750 (not provided in the FPU of the SH7718):

1. The FR, SZ, and PR bits have been added.
2. Exception O (overflow), U (underflow), and I (inexact) bits have been added to the cause, enable, and flag fields.
3. An exception E (FPU error) bit has been added to the cause field.

6.3.3 Floating-Point Communication Register (FPUL)

Information is transferred between the FPU and CPU via the FPUL register. The 32-bit FPUL register is a system register, and is accessed from the CPU side by means of LDS and STS instructions. For example, to convert the integer stored in general register R1 to a single-precision floating-point number, the processing flow is as follows:

R1 → (LDS instruction) → FPUL → (single-precision FLOAT instruction) → FR1

6.4 Rounding

In a floating-point instruction, rounding is performed when generating the final operation result from the intermediate result. Therefore, the result of combination instructions such as FMAC, FTRV, and FIPR will differ from the result when using a basic instruction such as FADD, FSUB, or FMUL. Rounding is performed once in FMAC, but twice in FADD, FSUB, and FMUL.

There are two rounding methods, the method to be used being determined by the RM field in FPSCR.

- RM = 00: Round to Nearest
- RM = 01: Round to Zero

Round to Nearest: The value is rounded to the nearest expressible value. If there are two nearest expressible values, the one with an LSB of 0 is selected.

If the unrounded value is $2^{Emax} (2 – 2^{–P})$ or more, the result will be infinity with the same sign as the unrounded value. The values of Emax and P, respectively, are 127 and 24 for single-precision, and 1023 and 53 for double-precision.

Round to Zero: The digits below the round bit of the unrounded value are discarded.

If the unrounded value is larger than the maximum expressible absolute value, the value will be the maximum expressible absolute value.

6.5 Floating-Point Exceptions

FPU-related exceptions are as follows:

- General illegal instruction/slot illegal instruction exception
  The exception occurs if an FPU instruction is executed when SR.FD = 1.

- FPU exceptions
  The exception sources are as follows:
  - FPU error (E): When FPSCR.DN = 0 and a denormalized number is input
  - Invalid operation (V): In case of an invalid operation, such as NaN input
  - Division by zero (Z): Division with a zero divisor
  - Overflow (O): When the operation result overflows
  - Underflow (U): When the operation result underflows
  - Inexact exception (I): When overflow, underflow, or rounding occurs

  The FPSCR cause field contains bits corresponding to all of above sources E, V, Z, O, U, and I, and the FPSCR flag and enable fields contain bits corresponding to sources V, Z, O, U, and I, but not E. Thus, FPU errors cannot be disabled.

  When an exception source occurs, the corresponding bit in the cause field is set to 1, and 1 is added to the corresponding bit in the flag field. When an exception source does not occur, the corresponding bit in the cause field is cleared to 0, but the corresponding bit in the flag field remains unchanged.

- Enable/disable exception handling
  The SH7750 supports enable exception handling and disable exception handling.
  Enable exception handling is initiated in the following cases:
  - FPU error (E): FPSCR.DN = 0 and a denormalized number is input
  - Invalid operation (V): FPSCR.EN.V = 1 and (instruction = FTRV or invalid operation)
  - Division by zero (Z): FPSCR.EN.Z = 1 and division with a zero divisor

— Overflow (O): FPSCR.EN.O = 1 and instruction with possibility of operation result overflow

— Underflow (U): FPSCR.EN.U = 1 and instruction with possibility of operation result underflow

— Inexact exception (I): FPSCR.EN.I = 1 and instruction with possibility of inexact operation result

These possibilities are shown in the individual instruction descriptions. All exception events that originate in the FPU are assigned as the same exception event. The meaning of an exception is determined by software by reading system register FPSCR and interpreting the information it contains. If no bits are set in the cause field of FPSCR when one or more of bits O, U, I, and V (in case of FTRV only) are set in the enable field, this indicates that an actual exception source is not generated. Also, the destination register is not changed by any enable exception handling operation.

Except for the above, the FPU disables exception handling. In all processing, the bit corresponding to source V, Z, O, U, or I is set to 1, and disable exception handling is provided for each exception.

— Invalid operation (V): qNAN is generated as the result.

— Division by zero (Z): Infinity with the same sign as the unrounded value is generated.

— Overflow (O):

When rounding mode = RZ, the maximum normalized number, with the same sign as the unrounded value, is generated.

When rounding mode = RN, infinity with the same sign as the unrounded value is generated.

— Underflow (U):

When FPSCR.DN = 0, a denormalized number with the same sign as the unrounded value, or zero with the same sign as the unrounded value, is generated.

When FPSCR.DN = 1, zero with the same sign as the unrounded value, is generated.

— Inexact exception (I): An inexact result is generated.

6.6 Graphics Support Functions

The SH7750 supports two kinds of graphics functions: new instructions for geometric operations, and pair single-precision transfer instructions that enable high-speed data transfer.

6.6.1 Geometric Operation Instructions

Geometric operation instructions perform approximate-value computations. To enable high-speed computation with a minimum of hardware, the SH7750 ignores comparatively small values in the partial computation results of four multiplications. Consequently, the error shown below is produced in the result of the computation:

$$\text{Maximum error} = \text{MAX}(\text{individual multiplication result} \times 2^{-\text{MIN (number of multiplier significant digits}-1\text{, number of multiplicand significant digits}-1)}) + \text{MAX}(\text{result value} \times 2^{-23}, 2^{-149})$$

The number of significant digits is 24 for a normalized number and 23 for a denormalized number (number of leading zeros in the fractional part).

In future version of SH series, the above error is guaranteed, but the same result as SH7750 is not guaranteed.

FIPR FVm, FVn (m, n: 0, 4, 8, 12): This instruction is basically used for the following purposes:

- Inner product ($m \neq n$):
  This operation is generally used for surface/rear surface determination for polygon surfaces.
- Sum of square of elements ($m = n$):
  This operation is generally used to find the length of a vector.

Since approximate-value computations are performed to enable high-speed computation, the inexact exception (I) bit in the cause field and flag field is always set to 1 when an FIPR instruction is executed. Therefore, if the corresponding bit is set in the enable field, enable exception handling will be executed.

FTRV XMTRX, FVn (n: 0, 4, 8, 12): This instruction is basically used for the following purposes:

- Matrix (4 × 4) · vector (4):
  This operation is generally used for viewpoint changes, angle changes, or movements called vector transformations (4-dimensional). Since affine transformation processing for angle + parallel movement basically requires a 4 × 4 matrix, the SH7750 supports 4-dimensional operations.
- Matrix (4 × 4) × matrix (4 × 4):
  This operation requires the execution of four FTRV instructions.

Since approximate-value computations are performed to enable high-speed computation, the inexact exception (I) bit in the cause field and flag field is always set to 1 when an FTRV instruction is executed. Therefore, if the corresponding bit is set in the enable field, enable exception handling will be executed. For the same reason, it is not possible to check all data types in the registers beforehand when executing an FTRV instruction. If the V bit is set in the enable field, enable exception handling will be executed.

FRCHG: This instruction modifies banked registers. For example, when the FTRV instruction is executed, matrix elements must be set in an array in the background bank. However, to create the actual elements of a translation matrix, it is easier to use registers in the foreground bank. When the LDC instruction is used on FPSCR, this instruction expends 4 to 5 cycles in order to maintain the FPU state. With the FRCHG instruction, an FPSCR.FR bit modification can be performed in one cycle.

6.6.2 Pair Single-Precision Data Transfer

In addition to the powerful new geometric operation instructions, the SH7750 also supports high-speed data transfer instructions.

When FPSCR.SZ = 1, the SH7750 can perform data transfer by means of pair single-precision data transfer instructions.

- FMOV DRm/XDm, DRn/XDRn (m, n: 0, 2, 4, 6, 8, 10, 12, 14)
- FMOV DRm/XDm, @Rn (m: 0, 2, 4, 6, 8, 10, 12, 14; n: 0 to 15)

These instructions enable two single-precision (2 × 32-bit) data items to be transferred; that is, the transfer performance of these instructions is doubled.

- FSCHG
  This instruction changes the value of the SZ bit in FPSCR, enabling fast switching between use and non-use of pair single-precision data transfer.

Section 7 Instruction Set

7.1 Execution Environment

PC: At the start of instruction execution, PC indicates the address of the instruction itself.

Data sizes and data types: The SH7750's instruction set is implemented with 16-bit fixed-length instructions. The SH7750 can use byte (8-bit), word (16-bit), longword (32-bit), and quadword (64-bit) data sizes for memory access. Single-precision floating-point data (32 bits) can be moved to and from memory using longword or quadword size. Double-precision floating-point data (64 bits) can be moved to and from memory using longword size. When a double-precision floating-point operation is specified (FPSCR.PR = 1), the result of an operation using quadword access will be undefined. When the SH7750 moves byte-size or word-size data from memory to a register, the data is sign-extended.

Load-Store Architecture: The SH7750 features a load-store architecture in which operations are basically executed using registers. Except for bit-manipulation operations such as logical AND that are executed directly in memory, operands in an operation that requires memory access are loaded into registers and the operation is executed between the registers.

Delayed Branches: Except for the two branch instructions BF and BT, the SH7750's branch instructions and RTE are delayed branches. In a delayed branch, the instruction following the branch is executed before the branch destination instruction. This execution slot following a delayed branch is called a delay slot. For example, the BRA execution sequence is as follows:

Static Sequence		**Dynamic Sequence**		
BRA	TARGET	BRA	TARGET	
ADD next_2	R1, R0	ADD target_instr	R1, R0	ADD in delay slot is executed before branching to TARGET

Delay Slot: An illegal instruction exception may occur when a specific instruction is executed in a delay slot. See section 5, Exceptions. The instruction following BF/S or BT/S for which the branch is not taken is also a delay slot instruction.

T Bit: The T bit in the status register (SR) is used to show the result of a compare operation, and is referenced by a conditional branch instruction. An example of the use of a conditional branch instruction is shown below.

```
ADD #1, R0       ; T bit is not changed by ADD operation
CMP/EQ R1, R0 ; If R0 = R1, T bit is set to 1
BT TARGET       ; Branches to TARGET if T bit = 1 (R0 = R1)
```

In an RTE delay slot, status register (SR) bits are referenced as follows. In instruction access, the MD bit is used before modification, and in data access, the MD bit is accessed after modification. The other bits—S, T, M, Q, FD, BL, and RB—after modification are used for delay slot instruction execution. The STC and STC.L SR instructions access all SR bits after modification.

Constant Values: An 8-bit constant value can be specified by the instruction code and an immediate value. 16-bit and 32-bit constant values can be defined as literal constant values in memory, and can be referenced by a PC-relative load instruction.

```
MOV.W    @(disp, PC), Rn
MOV.L     @(disp, PC), Rn
```

There are no PC-relative load instructions for floating-point operations. However, it is possible to set 0.0 or 1.0 by using the FLDI0 or FLDI1 instruction on a single-precision floating-point register.

7.2 Addressing Modes

Addressing modes and effective address calculation methods are shown in table 7.1. When a location in virtual memory space is accessed (MMUCR.AT = 1), the effective address is translated into a physical memory address. If multiple virtual memory space systems are selected (MMUCR.SV = 0), the least significant bit of PTEH is also referenced as the access ASID. See section 3, Memory Management Unit (MMU).

Table 7.1 Addressing Modes and Effective Addresses

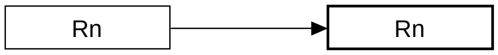
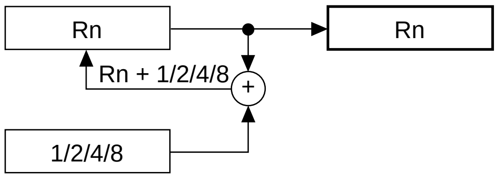
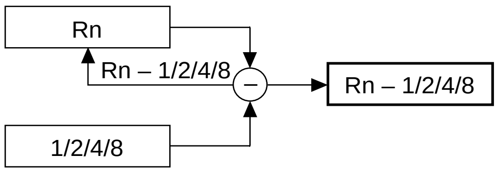
Addressing Mode	Instruction Format	Effective Address Calculation Method	Calculation Formula
Register direct	Rn	Effective address is register Rn. (Operand is register Rn contents.)	—
Register indirect	@Rn	Effective address is register Rn contents. 	Rn → EA (EA: effective address)
Register indirect with post-increment	@Rn+	Effective address is register Rn contents. A constant is added to Rn after instruction execution: 1 for a byte operand, 2 for a word operand, 4 for a longword operand, 8 for a quadword operand. 	Rn → EA After instruction execution Byte: Rn + 1 → Rn Word: Rn + 2 → Rn Longword: Rn + 4 → Rn Quadword: Rn + 8 → Rn
Register indirect with pre-decrement	@–Rn	Effective address is register Rn contents, decremented by a constant beforehand: 1 for a byte operand, 2 for a word operand, 4 for a longword operand, 8 for a quadword operand. 	Byte: Rn – 1 → Rn Word: Rn – 2 → Rn Longword: Rn – 4 → Rn Quadword: Rn – 8 → Rn Rn → EA (Instruction executed with Rn after calculation)

Table 7.1 Addressing Modes and Effective Addresses (cont)

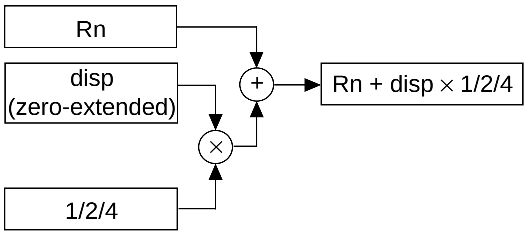
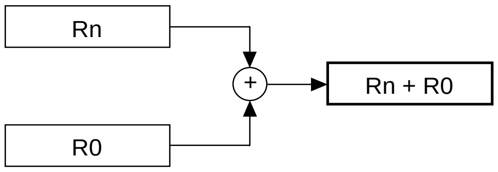
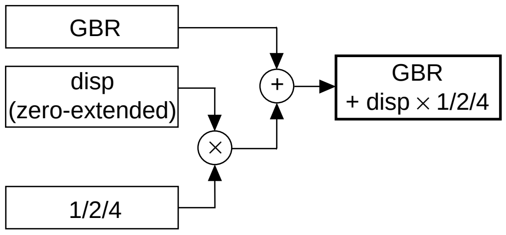
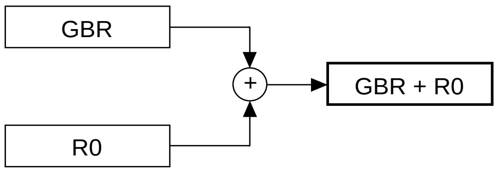
Addressing Mode	Instruction Format	Effective Address Calculation Method	Calculation Formula
Register indirect with displacement	@(disp:4, Rn)	Effective address is register Rn contents with 4-bit displacement disp added. After disp is zero-extended, it is multiplied by 1 (byte), 2 (word), or 4 (longword),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

Table 7.1 Addressing Modes and Effective Addresses (cont)

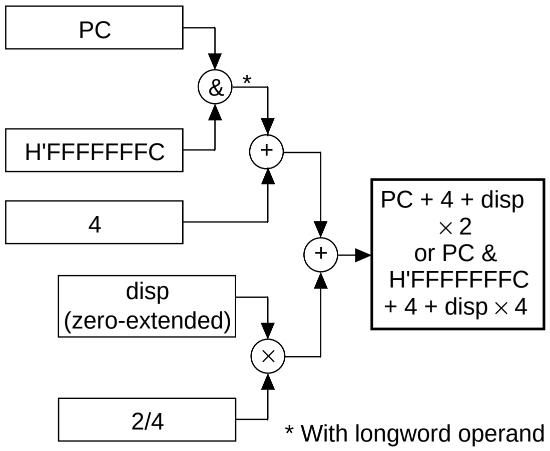
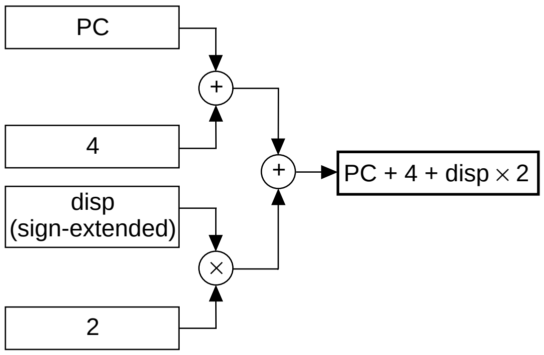
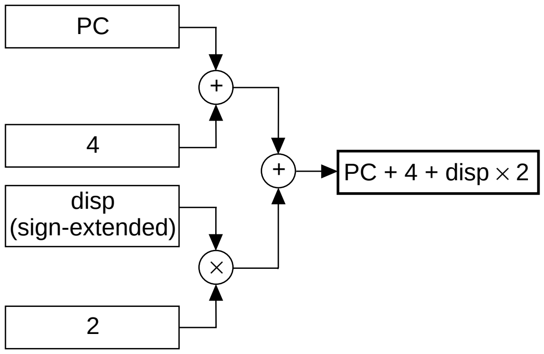
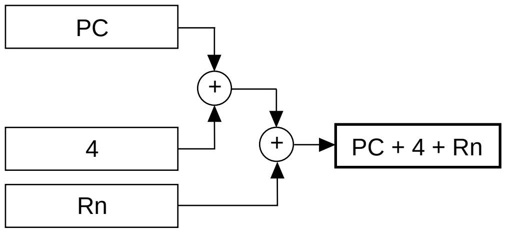
Addressing Mode	Instruction Format	Effective Address Calculation Method	Calculation Formula
PC-relative with displacement	@(disp:8, PC)	Effective address is PC+4 with 8-bit displacement disp added. After disp is zero-extended, it is multiplied by 2 (word), or 4 (longword), according to the operand size.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

Table 7.1 Addressing Modes and Effective Addresses (cont)

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

Note: For the addressing modes below that use a displacement (disp), the assembler descriptions in this manual show the value before scaling (×1, ×2, or ×4) is performed according to the operand size. This is done to clarify the operation of the chip. Refer to the relevant assembler notation rules for the actual assembler descriptions.

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

7.3 Instruction Set

Table 7.2 shows the notation used in the following SH instruction list.

Table 7.2 Notation Used in Instruction List

Item	Format	Description	
Instruction mnemonic	OP.Sz SRC, DEST	OP:	Operation code
		Sz:	Size
		SRC:	Source
		DEST:	Source and/or destination operand
Summary of operation		→, ←	Transfer direction
		(xx)	Memory operand
		M/Q/T	SR flag bits
		&	Logical AND of individual bits
		|	Logical OR of individual bits
		^	Logical exclusive-OR of individual bits
		~	Logical NOT of individual bits
		<<n, >>n	n-bit shift
Instruction code	MSB ↔ LSB	mmmm:	Register number (Rm, FRm)
		nnnn:	Register number (Rn, FRn)
		0000:	R0, FR0
		0001:	R1, FR1
		:	
		1111:	R15, FR15
		mmm:	Register number (DRm, XDm, Rm_BANK)
		nnn:	Register number (DRm, XDm, Rn_BANK)
		000:	DR0, XD0, R0_BANK
		001:	DR2, XD2, R1_BANK
		:	
		111:	DR14, XD14, R7_BANK
		mm:	Register number (FVm)
		nn:	Register number (FVn)
		00:	FV0
		01:	FV4
		10:	FV8
		11:	FV12
		iiii:	Immediate data
		dddd:	Displacement
Privileged mode		"Privileged" means the instruction can only be executed in privileged mode.	
T bit	Value of T bit after instruction execution	—: No change	

Note: Scaling (×1, ×2, ×4, or ×8) is executed according to the size of the instruction operand(s).

Table 7.3 Fixed-Point Transfer Instructions

Instruction		Operation	Instruction Code	Privileged	T Bit
MOV	#imm,Rn	imm → sign extension → Rn	`1110nnnniiiiiiii`	—	—
MOV.W	@(disp,PC),Rn	(disp × 2 + PC + 4) → sign extension → Rn	`1001nnnndddddddd`	—	—
MOV.L	@(disp,PC),Rn	(disp × 4 + PC & H'FFFFFFFC + 4) → Rn	`1101nnnndddddddd`	—	—
MOV	Rm,Rn	Rm → Rn	`0110nnnnmmmm0011`	—	—
MOV.B	Rm,@Rn	Rm → (Rn)	`0010nnnnmmmm0000`	—	—
MOV.W	Rm,@Rn	Rm → (Rn)	`0010nnnnmmmm0001`	—	—
MOV.L	Rm,@Rn	Rm → (Rn)	`0010nnnnmmmm0010`	—	—
MOV.B	@Rm,Rn	(Rm) → sign extension → Rn	`0110nnnnmmmm0000`	—	—
MOV.W	@Rm,Rn	(Rm) → sign extension → Rn	`0110nnnnmmmm0001`	—	—
MOV.L	@Rm,Rn	(Rm) → Rn	`0110nnnnmmmm0010`	—	—
MOV.B	Rm,@-Rn	Rn-1 → Rn, Rm → (Rn)	`0010nnnnmmmm0100`	—	—
MOV.W	Rm,@-Rn	Rn-2 → Rn, Rm → (Rn)	`0010nnnnmmmm0101`	—	—
MOV.L	Rm,@-Rn	Rn-4 → Rn, Rm → (Rn)	`0010nnnnmmmm0110`	—	—
MOV.B	@Rm+,Rn	(Rm)→ sign extension → Rn, Rm + 1 → Rm	`0110nnnnmmmm0100`	—	—
MOV.W	@Rm+,Rn	(Rm) → sign extension → Rn, Rm + 2 → Rm	`0110nnnnmmmm0101`	—	—
MOV.L	@Rm+,Rn	(Rm) → Rn, Rm + 4 → Rm	`0110nnnnmmmm0110`	—	—
MOV.B	R0,@(disp,Rn)	R0 → (disp + Rn)	`10000000nnnndddd`	—	—
MOV.W	R0,@(disp,Rn)	R0 → (disp × 2 + Rn)	`10000001nnnndddd`	—	—
MOV.L	Rm,@(disp,Rn)	Rm → (disp × 4 + Rn)	`0001nnnnmmmmdddd`	—	—
MOV.B	@(disp,Rm),R0	(disp + Rm) → sign extension → R0	`10000100mmmmdddd`	—	—
MOV.W	@(disp,Rm),R0	(disp × 2 + Rm) → sign extension → R0	`10000101mmmmdddd`	—	—
MOV.L	@(disp,Rm),Rn	(disp × 4 + Rm) → Rn	`0101nnnnmmmmdddd`	—	—
MOV.B	Rm,@(R0,Rn)	Rm → (R0 + Rn)	`0000nnnnmmmm0100`	—	—
MOV.W	Rm,@(R0,Rn)	Rm → (R0 + Rn)	`0000nnnnmmmm0101`	—	—
MOV.L	Rm,@(R0,Rn)	Rm → (R0 + Rn)	`0000nnnnmmmm0110`	—	—
MOV.B	@(R0,Rm),Rn	(R0 + Rm) → sign extension → Rn	`0000nnnnmmmm1100`	—	—
MOV.W	@(R0,Rm),Rn	(R0 + Rm) → sign extension → Rn	`0000nnnnmmmm1101`	—	—
MOV.L	@(R0,Rm),Rn	(R0 + Rm) → Rn	`0000nnnnmmmm1110`	—	—

Table 7.3 Fixed-Point Transfer Instructions (cont)

Instruction		Operation	Instruction Code	Privileged	T Bit
MOV.B	R0,@(disp,GBR)	R0 → (disp + GBR)	`11000000dddddddd`	—	—
MOV.W	R0,@(disp,GBR)	R0 → (disp × 2 + GBR)	`11000001dddddddd`	—	—
MOV.L	R0,@(disp,GBR)	R0 → (disp × 4 + GBR)	`11000010dddddddd`	—	—
MOV.B	@(disp,GBR),R0	(disp + GBR) → sign extension → R0	`11000100dddddddd`	—	—
MOV.W	@(disp,GBR),R0	(disp × 2 + GBR) → sign extension → R0	`11000101dddddddd`	—	—
MOV.L	@(disp,GBR),R0	(disp × 4 + GBR) → R0	`11000110dddddddd`	—	—
MOVA	@(disp,PC),R0	disp × 4 + PC & H'FFFFFFFC + 4 → R0	`11000111dddddddd`	—	—
MOVT	Rn	T → Rn	`0000nnnn00101001`	—	—
SWAP.B	Rm,Rn	Rm → swap lower 2 bytes → Rn	`0110nnnnmmmm1000`	—	—
SWAP.W	Rm,Rn	Rm → swap upper/lower words → Rn	`0110nnnnmmmm1001`	—	—
XTRCT	Rm,Rn	Rm:Rn middle 32 bits → Rn	`0010nnnnmmmm1101`	—	—

Table 7.4 Arithmetic Operation Instructions

Instruction		Operation	Instruction Code	Privileged	T Bit
ADD	Rm,Rn	Rn + Rm → Rn	`0011nnnnmmmm1100`	—	—
ADD	#imm,Rn	Rn + imm → Rn	`0111nnnniiiiiiii`	—	—
ADDC	Rm,Rn	Rn + Rm + T → Rn, carry → T	`0011nnnnmmmm1110`	—	Carry
ADDV	Rm,Rn	Rn + Rm → Rn, overflow → T	`0011nnnnmmmm1111`	—	Overflow
CMP/EQ	#imm,R0	When R0 = imm, 1 → T Otherwise, 0 → T	`10001000iiiiiiii`	—	Comparison result
CMP/EQ	Rm,Rn	When Rn = Rm, 1 → T Otherwise, 0 → T	`0011nnnnmmmm0000`	—	Comparison result
CMP/HS	Rm,Rn	When Rn ≥ Rm (unsigned), 1 → T Otherwise, 0 → T	`0011nnnnmmmm0010`	—	Comparison result
CMP/GE	Rm,Rn	When Rn ≥ Rm (signed), 1 → T Otherwise, 0 → T	`0011nnnnmmmm0011`	—	Comparison result
CMP/HI	Rm,Rn	When Rn > Rm (unsigned), 1 → T Otherwise, 0 → T	`0011nnnnmmmm0110`	—	Comparison result
CMP/GT	Rm,Rn	When Rn > Rm (signed), 1 → T Otherwise, 0 → T	`0011nnnnmmmm0111`	—	Comparison result
CMP/PZ	Rn	When Rn ≥ 0, 1 → T Otherwise, 0 → T	`0100nnnn00010001`	—	Comparison result
CMP/PL	Rn	When Rn > 0, 1 → T Otherwise, 0 → T	`0100nnnn00010101`	—	Comparison result
CMP/STR	Rm,Rn	When any bytes are equal, 1 → T Otherwise, 0 → T	`0010nnnnmmmm1100`	—	Comparison result
DIV1	Rm,Rn	1-step division (Rn ÷ Rm)	`0011nnnnmmmm0100`	—	Calculation result
DIV0S	Rm,Rn	MSB of Rn → Q, MSB of Rm → M, M^Q → T	`0010nnnnmmmm0111`	—	Calculation result
DIV0U		0 → M/Q/T	`0000000000011001`	—	0
DMULS.L	Rm,Rn	Signed, Rn × Rm → MAC, 32 × 32 → 64 bits	`0011nnnnmmmm1101`	—	—
DMULU.L	Rm,Rn	Unsigned, Rn × Rm → MAC, 32 × 32 → 64 bits	`0011nnnnmmmm0101`	—	—
DT	Rn	Rn – 1 → Rn; when Rn = 0, 1 → T When Rn ≠ 0, 0 → T	`0100nnnn00010000`	—	Comparison result
EXTS.B	Rm,Rn	Rm sign-extended from byte → Rn	`0110nnnnmmmm1110`	—	—

Table 7.4 Arithmetic Operation Instructions (cont)

Instruction		Operation	Instruction Code	Privileged	T Bit
EXTS.W	Rm,Rn	Rm sign-extended from word $\rightarrow$ Rn	`0110nnnnmmmm1111`	—	—
EXTU.B	Rm,Rn	Rm zero-extended from byte $\rightarrow$ Rn	`0110nnnnmmmm1100`	—	—
EXTU.W	Rm,Rn	Rm zero-extended from word $\rightarrow$ Rn	`0110nnnnmmmm1101`	—	—
MAC.L	@Rm+,@Rn+	Signed, (Rn) $\times$ (Rm) + MAC $\rightarrow$ MAC Rn + 4 $\rightarrow$ Rn, Rm + 4 $\rightarrow$ Rm 32 $\times$ 32 + 64 $\rightarrow$ 64 bits	`0000nnnnmmmm1111`	—	—
MAC.W	@Rm+,@Rn+	Signed, (Rn) $\times$ (Rm) + MAC $\rightarrow$ MAC Rn + 2 $\rightarrow$ Rn, Rm + 2 $\rightarrow$ Rm 16 $\times$ 16 + 64 $\rightarrow$ 64 bits	`0100nnnnmmmm1111`	—	—
MUL.L	Rm,Rn	Rn $\times$ Rm $\rightarrow$ MACL 32 $\times$ 32 $\rightarrow$ 32 bits	`0000nnnnmmmm0111`	—	—
MULS.W	Rm,Rn	Signed, Rn $\times$ Rm $\rightarrow$ MACL 16 $\times$ 16 $\rightarrow$ 32 bits	`0010nnnnmmmm1111`	—	—
MULU.W	Rm,Rn	Unsigned, Rn $\times$ Rm $\rightarrow$ MACL 16 $\times$ 16 $\rightarrow$ 32 bits	`0010nnnnmmmm1110`	—	—
NEG	Rm,Rn	0 – Rm $\rightarrow$ Rn	`0110nnnnmmmm1011`	—	—
NEGC	Rm,Rn	0 – Rm – T $\rightarrow$ Rn, borrow $\rightarrow$ T	`0110nnnnmmmm1010`	—	Borrow
SUB	Rm,Rn	Rn – Rm $\rightarrow$ Rn	`0011nnnnmmmm1000`	—	—
SUBC	Rm,Rn	Rn – Rm – T $\rightarrow$ Rn, borrow $\rightarrow$ T	`0011nnnnmmmm1010`	—	Borrow
SUBV	Rm,Rn	Rn – Rm $\rightarrow$ Rn, underflow $\rightarrow$ T	`0011nnnnmmmm1011`	—	Underflow

Table 7.5 Logic Operation Instructions

Instruction		Operation	Instruction Code	Privileged	T Bit
AND	Rm,Rn	Rn & Rm → Rn	`0010nnnnmmmm1001`	—	—
AND	#imm,R0	R0 & imm → R0	`11001001iiiiiiii`	—	—
AND.B	#imm,@(R0,GBR)	(R0 + GBR) & imm → (R0 + GBR)	`11001101iiiiiiii`	—	—
NOT	Rm,Rn	~Rm → Rn	`0110nnnnmmmm0111`	—	—
OR	Rm,Rn	Rn | Rm → Rn	`0010nnnnmmmm1011`	—	—
OR	#imm,R0	R0 | imm → R0	`11001011iiiiiiii`	—	—
OR.B	#imm,@(R0,GBR)	(R0 + GBR) | imm → (R0 + GBR)	`11001111iiiiiiii`	—	
TAS.B	@Rn	When (Rn) = 0, 1 → T Otherwise, 0 → T In both cases, 1 → MSB of (Rn)	`0100nnnn00011011`	—	Test result
TST	Rm,Rn	Rn & Rm; when result = 0, 1 → T Otherwise, 0 → T	`0010nnnnmmmm1000`	—	Test result
TST	#imm,R0	R0 & imm; when result = 0, 1 → T Otherwise, 0 → T	`11001000iiiiiiii`	—	Test result
TST.B	#imm,@(R0,GBR)	(R0 + GBR) & imm; when result = 0, 1 → T Otherwise, 0 → T	`11001100iiiiiiii`	—	Test result
XOR	Rm,Rn	Rn ∧ Rm → Rn	`0010nnnnmmmm1010`	—	—
XOR	#imm,R0	R0 ∧ imm → R0	`11001010iiiiiiii`	—	—
XOR.B	#imm,@(R0,GBR)	(R0 + GBR) ∧ imm → (R0 + GBR)	`11001110iiiiiiii`	—	—

Table 7.6 Shift Instructions

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

Table 7.7 Branch Instructions

Instruction		Operation	Instruction Code	Privileged	T Bit
BF	label	When T = 0, disp $\times$ 2 + PC + 4 $\rightarrow$ PC When T = 1, nop	`10001011dddddddd`	—	—
BF/S	label	Delayed branch; when T = 0, disp $\times$ 2 + PC + 4 $\rightarrow$ PC When T = 1, nop	`10001111dddddddd`	—	—
BT	label	When T = 1, disp $\times$ 2 + PC + 4 $\rightarrow$ PC When T = 0, nop	`10001001dddddddd`	—	—
BT/S	label	Delayed branch; when T = 1, disp $\times$ 2 + PC + 4 $\rightarrow$ PC When T = 0, nop	`10001101dddddddd`	—	—
BRA	label	Delayed branch, disp $\times$ 2 + PC + 4 $\rightarrow$ PC	`1010dddddddddddd`	—	—
BRAF	Rn	Rn + PC + 4 $\rightarrow$ PC	`0000nnnn00100011`	—	—
BSR	label	Delayed branch, PC + 4 $\rightarrow$ PR, disp $\times$ 2 + PC + 4 $\rightarrow$ PC	`1011dddddddddddd`	—	—
BSRF	Rn	Delayed branch, PC + 4 $\rightarrow$ PR, Rn + PC + 4 $\rightarrow$ PC	`0000nnnn00000011`	—	—
JMP	@Rn	Delayed branch, Rn $\rightarrow$ PC	`0100nnnn00101011`	—	—
JSR	@Rn	Delayed branch, PC + 4 $\rightarrow$ PR, Rn $\rightarrow$ PC	`0100nnnn00001011`	—	—
RTS		Delayed branch, PR $\rightarrow$ PC	`0000000000001011`	—	—

Table 7.8 System Control Instructions

Instruction		Operation	Instruction Code	Privileged	T Bit
CLRMAC		0 → MACH, MACL	0000000000101000	—	—
CLRS		0 → S	0000000001001000	—	—
CLRT		0 → T	0000000000001000	—	0
LDC	Rm,SR	Rm → SR	0100mmmm00001110	Privileged	LSB
LDC	Rm,GBR	Rm → GBR	0100mmmm00011110	—	—
LDC	Rm,VBR	Rm → VBR	0100mmmm00101110	Privileged	—
LDC	Rm,SSR	Rm → SSR	0100mmmm00111110	Privileged	—
LDC	Rm,SPC	Rm → SPC	0100mmmm01001110	Privileged	—
LDC	Rm,DBR	Rm → DBR	0100mmmm11111010	Privileged	—
LDC	Rm,Rn_BANK	Rm → Rn_BANK (n = 0 to 7)	0100mmmm1nnn1110	Privileged	—
LDC.L	@Rm+,SR	(Rm) → SR, Rm + 4 → Rm	0100mmmm00000111	Privileged	LSB
LDC.L	@Rm+,GBR	(Rm) → GBR, Rm + 4 → Rm	0100mmmm00010111	—	—
LDC.L	@Rm+,VBR	(Rm) → VBR, Rm + 4 → Rm	0100mmmm00100111	Privileged	—
LDC.L	@Rm+,SSR	(Rm) → SSR, Rm + 4 → Rm	0100mmmm00110111	Privileged	—
LDC.L	@Rm+,SPC	(Rm) → SPC, Rm + 4 → Rm	0100mmmm01000111	Privileged	—
LDC.L	@Rm+,DBR	(Rm) → DBR, Rm + 4 → Rm	0100mmmm11110110	Privileged	—
LDC.L	@Rm+,Rn_BANK	(Rm) → Rn_BANK, Rm + 4 → Rm	0100mmmm1nnn0111	Privileged	—
LDS	Rm,MACH	Rm → MACH	0100mmmm00001010	—	—
LDS	Rm,MACL	Rm → MACL	0100mmmm00011010	—	—
LDS	Rm,PR	Rm → PR	0100mmmm00101010	—	—
LDS.L	@Rm+,MACH	(Rm) → MACH, Rm + 4 → Rm	0100mmmm00000110	—	—
LDS.L	@Rm+,MACL	(Rm) → MACL, Rm + 4 → Rm	0100mmmm00010110	—	—
LDS.L	@Rm+,PR	(Rm) → PR, Rm + 4 → Rm	0100mmmm00100110	—	—
LDTLB		PTEH/PTEL → TLB	0000000000111000	Privileged	—
MOVCA.L	R0,@Rn	R0 → (Rn) (without fetching cache block)	0000nnnn11000011	—	—
NOP		No operation	0000000000001001	—	—
OCBI	@Rn	Invalidates operand cache block	0000nnnn10010011	—	—
OCBP	@Rn	Writes back and invalidates operand cache block	0000nnnn10100011	—	—
OCBWB	@Rn	Writes back operand cache block	0000nnnn10110011	—	—
PREF	@Rn	(Rn) → operand cache	0000nnnn10000011	—	—
RTE		Delayed branch, SSR/SPC → SR/PC	0000000000101011	Privileged	—

Table 7.8 System Control Instructions (cont)

Instruction		Operation	Instruction Code	Privileged	T Bit
SETS		1 → S	0000000001011000	—	—
SETT		1 → T	0000000000011000	—	1
SLEEP		Sleep or standby	0000000000011011	Privileged	—
STC	SR,Rn	SR → Rn	0000nnnn00000010	Privileged	—
STC	GBR,Rn	GBR → Rn	0000nnnn00010010	—	—
STC	VBR,Rn	VBR → Rn	0000nnnn00100010	Privileged	—
STC	SSR,Rn	SSR → Rn	0000nnnn00110010	Privileged	—
STC	SPC,Rn	SPC → Rn	0000nnnn01000010	Privileged	—
STC	SGR,Rn	SGR → Rn	0000nnnn00111010	Privileged	—
STC	DBR,Rn	DBR → Rn	0000nnnn11111010	Privileged	—
STC	Rm_BANK,Rn	Rm_BANK → Rn (m = 0 to 7)	0000nnnn1mmm0010	Privileged	—
STC.L	SR,@-Rn	Rn – 4 → Rn, SR → (Rn)	0100nnnn00000011	Privileged	—
STC.L	GBR,@-Rn	Rn – 4 → Rn, GBR → (Rn)	0100nnnn00010011	—	—
STC.L	VBR,@-Rn	Rn – 4 → Rn, VBR → (Rn)	0100nnnn00100011	Privileged	—
STC.L	SSR,@-Rn	Rn – 4 → Rn, SSR → (Rn)	0100nnnn00110011	Privileged	—
STC.L	SPC,@-Rn	Rn – 4 → Rn, SPC → (Rn)	0100nnnn01000011	Privileged	—
STC.L	SGR,@-Rn	Rn – 4 → Rn, SGR → (Rn)	0100nnnn00110010	Privileged	—
STC.L	DBR,@-Rn	Rn – 4 → Rn, DBR → (Rn)	0100nnnn11110010	Privileged	—
STC.L	Rm_BANK,@-Rn	Rn – 4 → Rn, Rm_BANK → (Rn) (m = 0 to 7)	0100nnnn1mmm0011	Privileged	—
STS	MACH,Rn	MACH → Rn	0000nnnn00001010	—	—
STS	MACL,Rn	MACL → Rn	0000nnnn00011010	—	—
STS	PR,Rn	PR → Rn	0000nnnn00101010	—	—
STS.L	MACH,@-Rn	Rn – 4 → Rn, MACH → (Rn)	0100nnnn00000010	—	—
STS.L	MACL,@-Rn	Rn – 4 → Rn, MACL → (Rn)	0100nnnn00010010	—	—
STS.L	PR,@-Rn	Rn – 4 → Rn, PR → (Rn)	0100nnnn00100010	—	—
TRAPA	#imm	PC + 2 → SPC, SR → SSR, #imm << 2 → TRA, H'160 → EXPEVT, VBR + H'0100 → PC	11000011iiiiiiii	—	—

Table 7.9 Floating-Point Single-Precision Instructions

Instruction		Operation	Instruction Code	Privileged	T Bit
FLDI0	FRn	H'00000000 → FRn	`1111nnnn10001101`	—	—
FLDI1	FRn	H'3F800000 → FRn	`1111nnnn10011101`	—	—
FMOV	FRm,FRn	FRm → FRn	`1111nnnnmmmm1100`	—	—
FMOV.S	@Rm,FRn	(Rm) → FRn	`1111nnnnmmmm1000`	—	—
FMOV.S	@(R0,Rm),FRn	(R0 + Rm) → FRn	`1111nnnnmmmm0110`	—	—
FMOV.S	@Rm+,FRn	(Rm) → FRn, Rm + 4 → Rm	`1111nnnnmmmm1001`	—	—
FMOV.S	FRm,@Rn	FRm → (Rn)	`1111nnnnmmmm1010`	—	—
FMOV.S	FRm,@-Rn	Rn-4 → Rn, FRm → (Rn)	`1111nnnnmmmm1011`	—	—
FMOV.S	FRm,@(R0,Rn)	FRm → (R0 + Rn)	`1111nnnnmmmm0111`	—	—
FMOV	DRm,DRn	DRm → DRn	`1111nnn0mmm01100`	—	—
FMOV	@Rm,DRn	(Rm) → DRn	`1111nnn0mmmm1000`	—	—
FMOV	@(R0,Rm),DRn	(R0 + Rm) → DRn	`1111nnn0mmmm0110`	—	—
FMOV	@Rm+,DRn	(Rm) → DRn, Rm + 8 → Rm	`1111nnn0mmmm1001`	—	—
FMOV	DRm,@Rn	DRm → (Rn)	`1111nnnnmmm01010`	—	—
FMOV	DRm,@-Rn	Rn-8 → Rn, DRm → (Rn)	`1111nnnnmmm01011`	—	—
FMOV	DRm,@(R0,Rn)	DRm → (R0 + Rn)	`1111nnnnmmm00111`	—	—
FLDS	FRm,FPUL	FRm → FPUL	`1111mmmm00011101`	—	—
FSTS	FPUL,FRn	FPUL → FRn	`1111nnnn00001101`	—	—
FABS	FRn	FRn & H'7FFF FFFF → FRn	`1111nnnn01011101`	—	—
FADD	FRm,FRn	FRn + FRm → FRn	`1111nnnnmmmm0000`	—	—
FCMP/EQ	FRm,FRn	When FRn = FRm, 1 → T Otherwise, 0 → T	`1111nnnnmmmm0100`	—	Comparison result
FCMP/GT	FRm,FRn	When FRn > FRm, 1 → T Otherwise, 0 → T	`1111nnnnmmmm0101`	—	Comparison result
FDIV	FRm,FRn	FRn/FRm → FRn	`1111nnnnmmmm0011`	—	—
FLOAT	FPUL,FRn	(float) FPUL → FRn	`1111nnnn00101101`	—	—
FMAC	FR0,FRm,FRn	FR0*FRm + FRn → FRn	`1111nnnnmmmm1110`	—	—
FMUL	FRm,FRn	FRn*FRm → FRn	`1111nnnnmmmm0010`	—	—
FNEG	FRn	FRn ∧ H'80000000 → FRn	`1111nnnn01001101`	—	—
FSQRT	FRn	√FRn → FRn	`1111nnnn01101101`	—	—
FSUB	FRm,FRn	FRn – FRm → FRn	`1111nnnnmmmm0001`	—	—
FTRC	FRm,FPUL	(long) FRm → FPUL	`1111mmmm00111101`	—	—

Table 7.10 Floating-Point Double-Precision Instructions

Instruction		Operation	Instruction Code	Privileged	T Bit
FABS	DRn	DRn & H'7FFF FFFF FFFF FFFF → DRn	`1111nnn001011101`	—	—
FADD	DRm,DRn	DRn + DRm → DRn	`1111nnn0mmm00000`	—	—
FCMP/EQ	DRm,DRn	When DRn = DRm, 1 → T Otherwise, 0 → T	`1111nnn0mmm00100`	—	Comparison result
FCMP/GT	DRm,DRn	When DRn > DRm, 1 → T Otherwise, 0 → T	`1111nnn0mmm00101`	—	Comparison result
FDIV	DRm,DRn	DRn /DRm → DRn	`1111nnn0mmm00011`	—	—
FCNVDS	DRm,FPUL	double_to_ float[DRm] → FPUL	`1111mmm010111101`	—	—
FCNVSD	FPUL,DRn	float_to_ double [FPUL] → DRn	`1111nnn010101101`	—	—
FLOAT	FPUL,DRn	(float)FPUL → DRn	`1111nnn000101101`	—	—
FMUL	DRm,DRn	DRn *DRm → DRn	`1111nnn0mmm00010`	—	—
FNEG	DRn	DRn ^ H'8000 0000 0000 0000 → DRn	`1111nnn001001101`	—	—
FSQRT	DRn	√DRn → DRn	`1111nnn001101101`	—	—
FSUB	DRm,DRn	DRn – DRm → DRn	`1111nnn0mmm00001`	—	—
FTRC	DRm,FPUL	(long) DRm → FPUL	`1111mmm000111101`	—	—

Table 7.11 Floating-Point Control Instructions

Instruction		Operation	Instruction Code	Privileged	T Bit
LDS	Rm,FPSCR	Rm → FPSCR	`0100mmmm01101010`	—	—
LDS	Rm,FPUL	Rm → FPUL	`0100mmmm01011010`	—	—
LDS.L	@Rm+,FPSCR	(Rm) → FPSCR, Rm+4 → Rm	`0100mmmm01100110`	—	—
LDS.L	@Rm+,FPUL	(Rm) → FPUL, Rm+4 → Rm	`0100mmmm01010110`	—	—
STS	FPSCR,Rn	FPSCR → Rn	`0000nnnn01101010`	—	—
STS	FPUL,Rn	FPUL → Rn	`0000nnnn01011010`	—	—
STS.L	FPSCR,@-Rn	Rn – 4 → Rn, FPSCR → (Rn)	`0100nnnn01100010`	—	—
STS.L	FPUL,@-Rn	Rn – 4 → Rn, FPUL → (Rn)	`0100nnnn01010010`	—	—

Table 7.12 Floating-Point Graphics Acceleration Instructions

Instruction		Operation	Instruction Code	Privileged	T Bit
FMOV	DRm,XDn	DRm → XDn	1111nnn1mmm01100	—	—
FMOV	XDm,DRn	XDm → DRn	1111nnn0mmm11100	—	—
FMOV	XDm,XDn	XDm → XDn	1111nnn1mmm11100	—	—
FMOV	@Rm,XDn	(Rm) → XDn	1111nnn1mmmm1000	—	—
FMOV	@Rm+,XDn	(Rm) → XDn, Rm + 8 → Rm	1111nnn1mmmm1001	—	—
FMOV	@(R0,Rm),XDn	(R0 + Rm) → XDn	1111nnn1mmmm0110	—	—
FMOV	XDm,@Rn	XDm → (Rn)	1111nnnnmmm11010	—	—
FMOV	XDm,@-Rn	Rn – 8 → Rn, XDm → (Rn)	1111nnnnmmm11011	—	—
FMOV	XDm,@(R0,Rn)	XDm → (R0+Rn)	1111nnnnmmm10111	—	—
FIPR	FVm,FVn	inner_product [FVm, FVn] → FR[n+3]	1111nnmm11101101	—	—
FTRV	XMTRX,FVn	transform_vector [XMTRX, FVn] → FVn	1111nn0111111101	—	—
FRCHG		~FPSCR.FR → SPFCR.FR	1111101111111101	—	—
FSCHG		~FPSCR.SZ → SPFCR.SZ	1111001111111101	—	—

Section 8 Pipelining

The SH7750 is a 2-ILP (instruction-level-parallelism) superscalar pipelining microprocessor. Instruction execution is pipelined, and two instructions can be executed in parallel. The execution cycles depend on the implementation of a processor. Definitions in this section may not be applicable to SH-4 Series models other than the SH7750.

8.1 Pipelines

Figure 8.1 shows the basic pipelines. Normally, a pipeline consists of five or six stages: instruction fetch (I), decode and register read (D), execution (EX/SX/F0/F1/F2/F3), data access (NA/MA), and write-back (S/FS). An instruction is executed as a combination of basic pipelines. Figure 8.2 shows the instruction execution patterns.

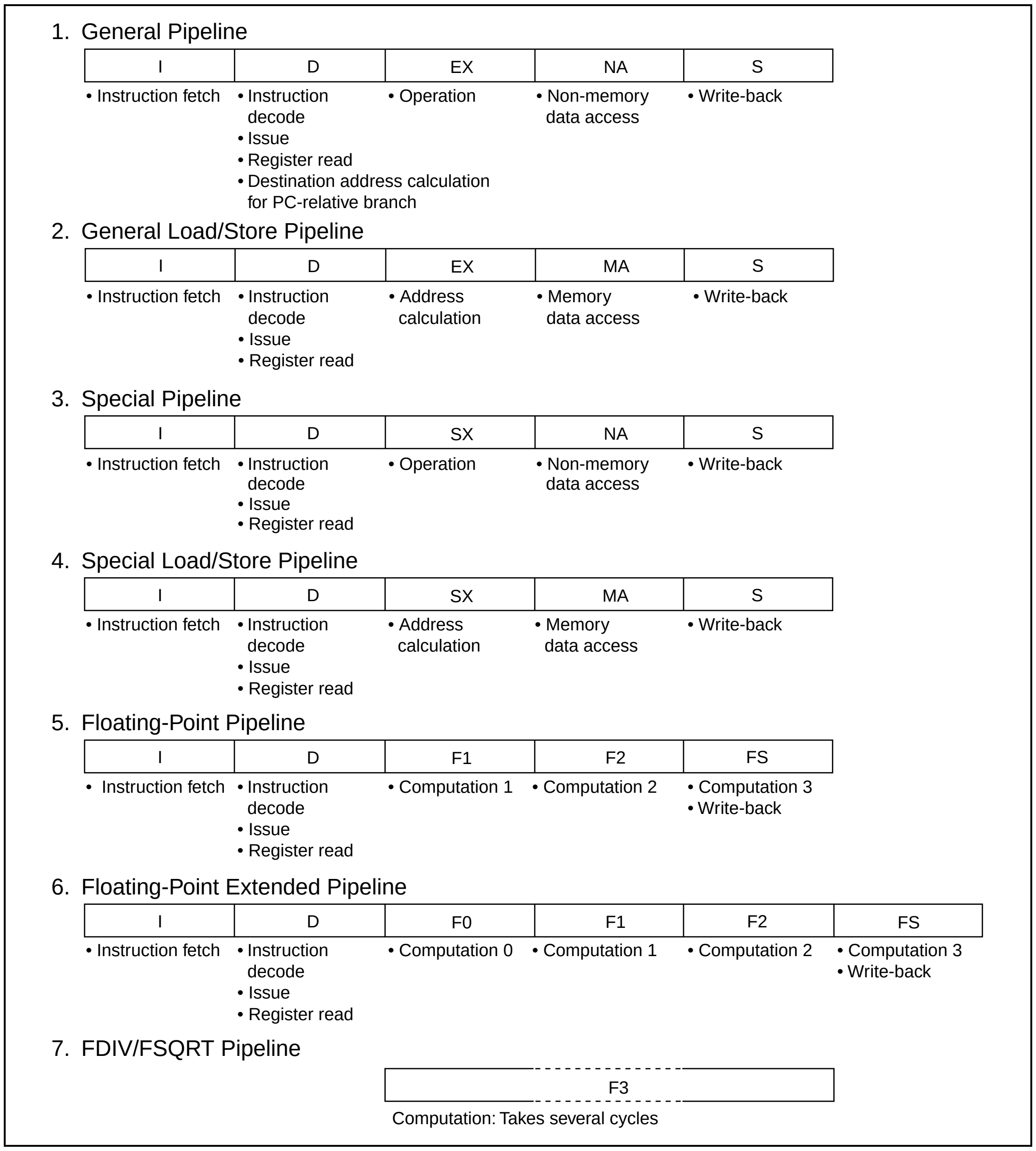
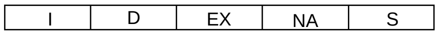
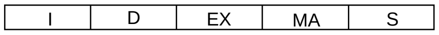


Figure 8.1 Basic Pipelines

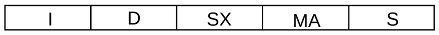
1. 1-step operation: 1 issue cycle
EXT[SU].[BW], MOV, MOV#, MOVA, MOVT, SWAP.[BW], XTRCT, ADD*, CMP*, DIV*, DT, NEG*, SUB*, AND, AND#, NOT, OR, OR#, TST, TST#, XOR, XOR#, ROT*, SHA*, SHL*, BF*, BT*, BRA, NOP, CLRS, CLRT, SETS, SETT, LDS to FPUL, STS from FPUL/FPSCR, FLDI0, FLDI1, FMOV, FLDS, FSTS, single-/double-precision FABS/FNEG



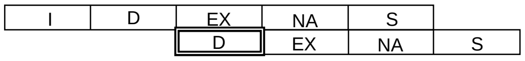
2. Load/store: 1 issue cycle
MOV.[BWL]. FMOV*@, LDS.L to FPUL, LDTLB, PREF, STS.L from FPUL/FPSCR



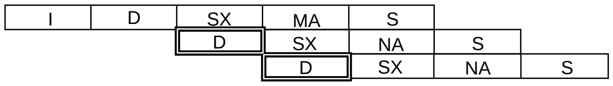
3. GBR-based load/store: 1 issue cycle
MOV.[BWL]@(d,GBR)



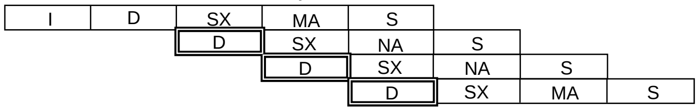
4. JMP, RTS, BRAF: 2 issue cycles



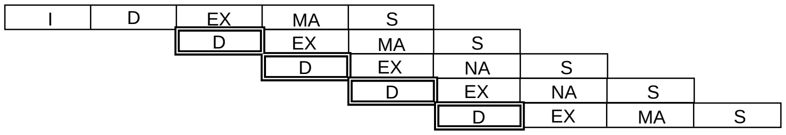
5. TST.B: 3 issue cycles



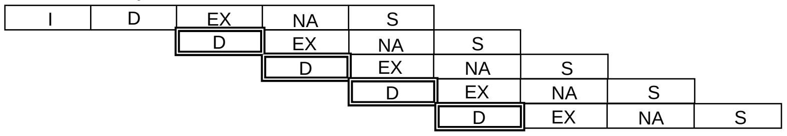
6. AND.B, OR.B, XOR.B: 4 issue cycles



7. TAS.B: 5 issue cycles



8. RTE: 5 issue cycles



9. SLEEP: 4 issue cycles

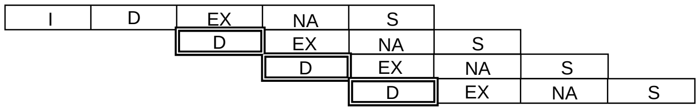


Figure 8.2 Instruction Execution Patterns

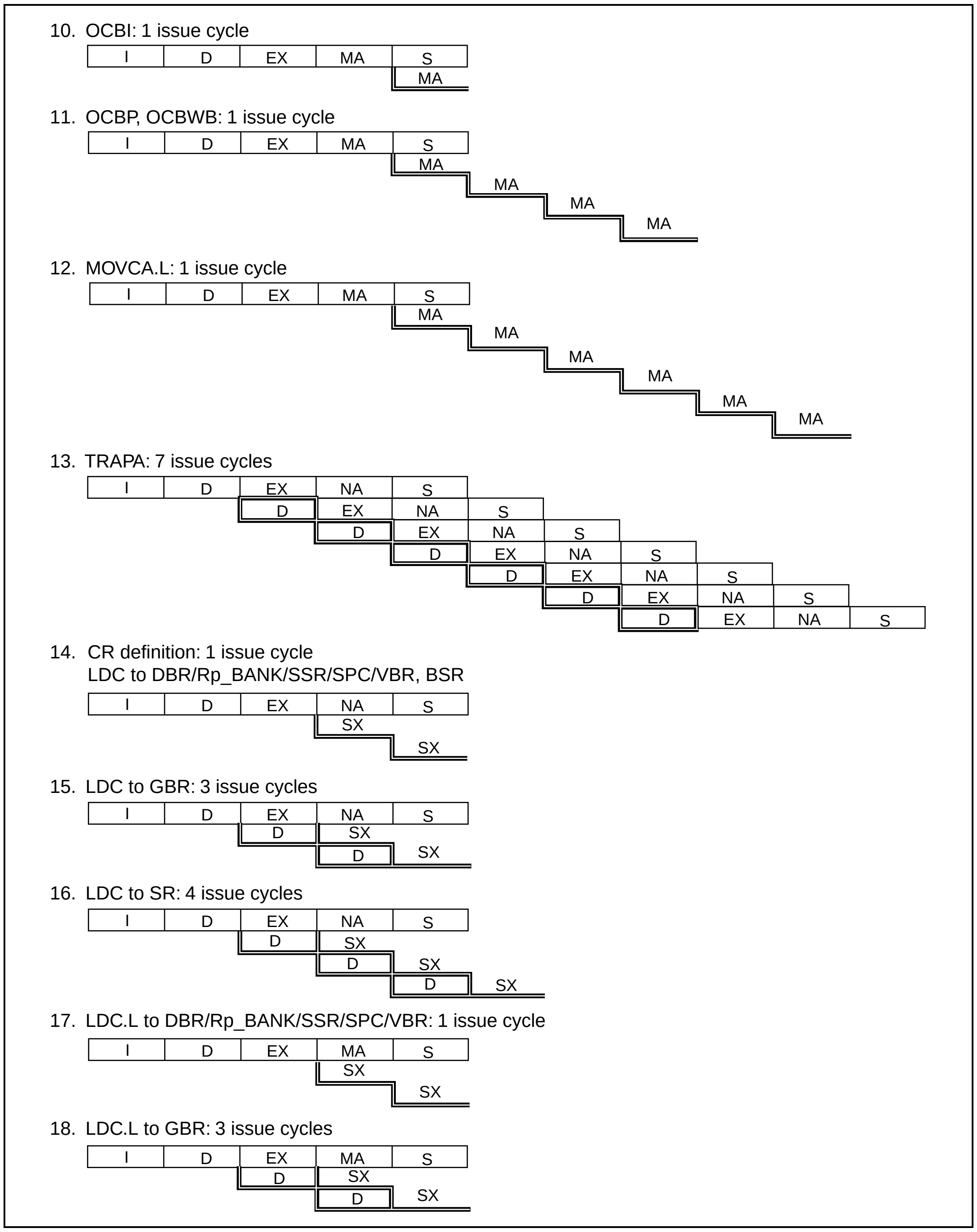
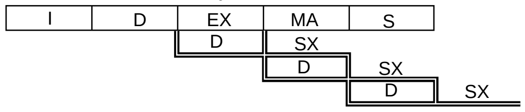
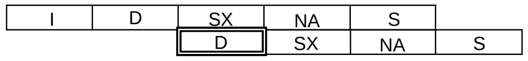

10. OCBI: 1 issue cycle
I D EX MA S
MA
11. OCBP, OCBWB: 1 issue cycle
I D EX MA S
MA
MA
MA
MA
12. MOVCA.L: 1 issue cycle
I D EX MA S
MA
MA
MA
MA
MA
MA
13. TRAPA: 7 issue cycles
I D EX NA S
D EX NA S
D EX NA S
D EX NA S
D EX NA S
D EX NA S
D EX NA S
14. CR definition: 1 issue cycle
LDC to DBR/Rp_BANK/SSR/SPC/VBR, BSR
I D EX NA S
SX
SX
15. LDC to GBR: 3 issue cycles
I D EX NA S
D SX
D SX
16. LDC to SR: 4 issue cycles
I D EX NA S
D SX
D SX
D SX
17. LDC.L to DBR/Rp_BANK/SSR/SPC/VBR: 1 issue cycle
I D EX MA S
SX
SX
18. LDC.L to GBR: 3 issue cycles
I D EX MA S
D SX
D SX

Figure 8.2 Instruction Execution Patterns (cont)

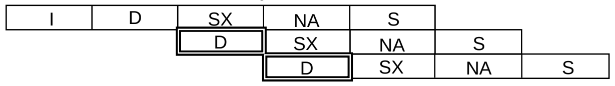
19. LDC.L to SR: 4 issue cycles



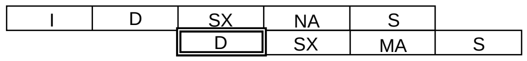
20. STC from DBR/GBR/Rp_BANK/SR/SSR/SPC/VBR: 2 issue cycles



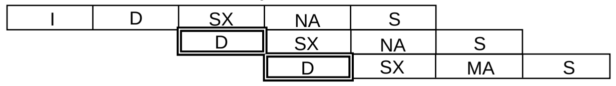
21. STC.L from SGR: 3 issue cycles



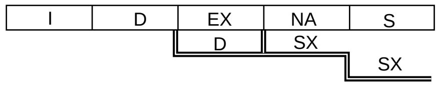
22. STC.L from DBR/GBR/Rp_BANK/SR/SSR/SPC/VBR: 2 issue cycles



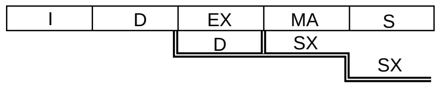
23. STC.L from SGR: 3 issue cycles



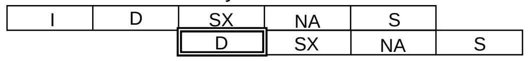
24. LDS to PR, JSR, BSRF: 2 issue cycles



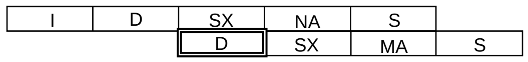
25. LDS.L to PR: 2 issue cycles



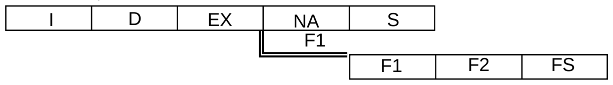
26. STS from PR: 2 issue cycles



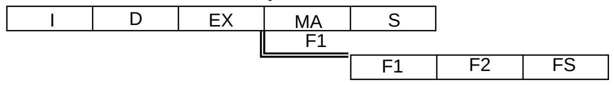
27. STS.L from PR: 2 issue cycles



28. MACH/L definition: 1 issue cycle
CLRMAC, LDS to MACH/L



29. LDS.L to MACH/L: 1 issue cycle



30. STS from MACH/L: 1 issue cycle

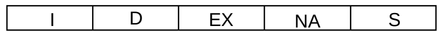
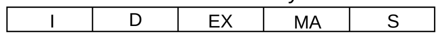
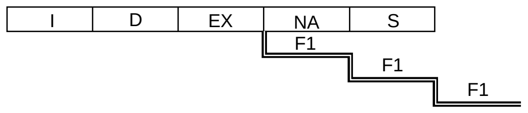


Figure 8.2 Instruction Execution Patterns (cont)

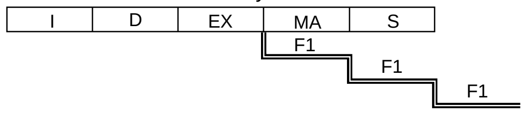
31. STS.L from MACH/L: 1 issue cycle



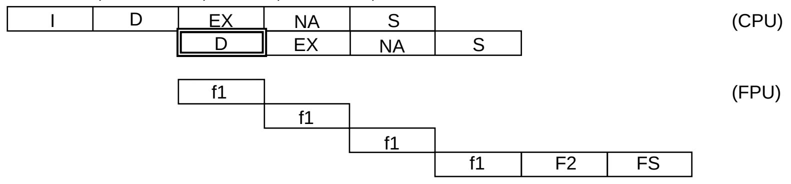
32. LDS to FPSCR: 1 issue cycle



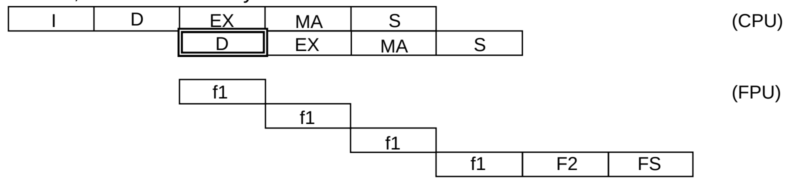
33. LDS.L to FPSCR: 1 issue cycle



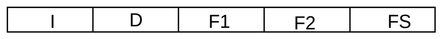
34. Fixed-point multiplication: 2 issue cycles
DMULS.L, DMULU.L, MUL.L, MULS.W, MULU.W



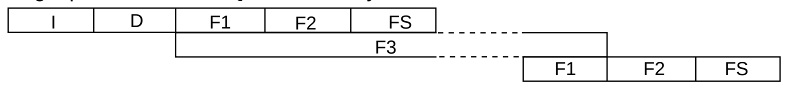
35. MAC.W, MAC.L: 2 issue cycles



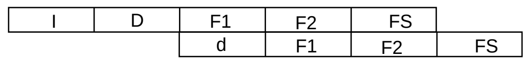
36. Single-precision floating-point computation: 1 issue cycle
FCMP/EQ,FCMP/GT, FADD,FLOAT,FMAC,FMUL,FSUB,FTRC,FRCHG,FSCHG



37. Single-precision FDIV/SQRT: 1 issue cycle



38. Double-precision floating-point computation 1: 1 issue cycle
FCNVDS, FCNVSD, FLOAT, FTRC



39. Double-precision floating-point computation 2: 1 issue cycle
FADD, FMUL, FSUB

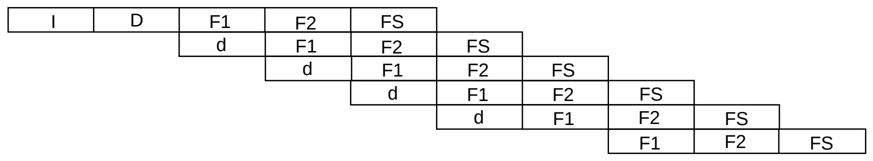
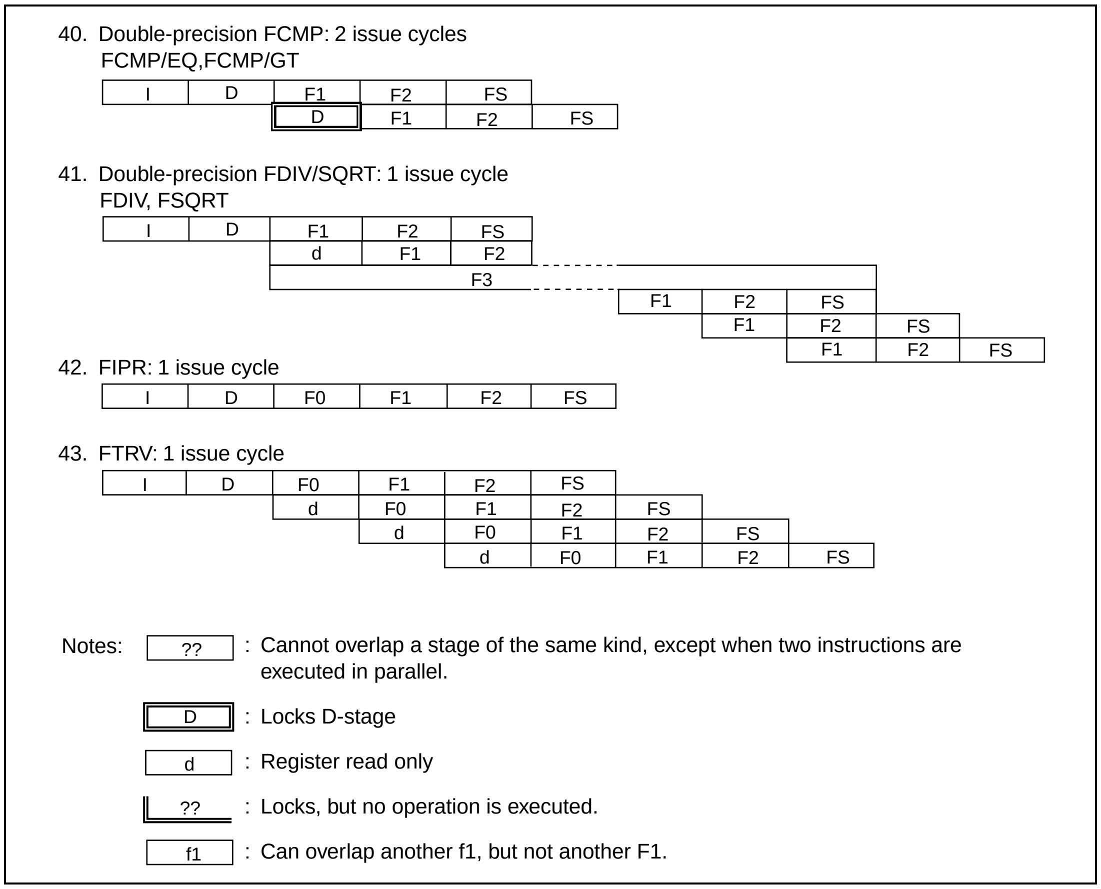


Figure 8.2 Instruction Execution Patterns (cont)



Notes: ?? : Cannot overlap a stage of the same kind, except when two instructions are executed in parallel.

D : Locks D-stage

d : Register read only

?? : Locks, but no operation is executed.

f1 : Can overlap another f1, but not another F1.

Figure 8.2 Instruction Execution Patterns (cont)

8.2 Parallel-Executability

Instructions are categorized into six groups according to the internal function blocks used, as shown in table 8.1. Table 8.2 shows the parallel-executability of pairs of instructions in terms of groups. For example, ADD in the EX group and BRA in the BR group can be executed in parallel.

Table 8.1 Instruction Groups

1. MT Group

CLRT		CMP/HI	Rm,Rn	MOV	Rm,Rn
CMP/EQ	#imm,R0	CMP/HS	Rm,Rn	NOP	
CMP/EQ	Rm,Rn	CMP/PL	Rn	SETT	
CMP/GE	Rm,Rn	CMP/PZ	Rn	TST	#imm,R0
CMP/GT	Rm,Rn	CMP/STR	Rm,Rn	TST	Rm,Rn

2. EX Group

ADD	#imm,Rn	MOVT	Rn	SHLL2	Rn
ADD	Rm,Rn	NEG	Rm,Rn	SHLL8	Rn
ADDC	Rm,Rn	NEGC	Rm,Rn	SHLR	Rn
ADDV	Rm,Rn	NOT	Rm,Rn	SHLR16	Rn
AND	#imm,R0	OR	#imm,R0	SHLR2	Rn
AND	Rm,Rn	OR	Rm,Rn	SHLR8	Rn
DIV0S	Rm,Rn	ROTCL	Rn	SUB	Rm,Rn
DIV0U		ROTCR	Rn	SUBC	Rm,Rn
DIV1	Rm,Rn	ROTL	Rn	SUBV	Rm,Rn
DT	Rn	ROTR	Rn	SWAP.B	Rm,Rn
EXTS.B	Rm,Rn	SHAD	Rm,Rn	SWAP.W	Rm,Rn
EXTS.W	Rm,Rn	SHAL	Rn	XOR	#imm,R0
EXTU.B	Rm,Rn	SHAR	Rn	XOR	Rm,Rn
EXTU.W	Rm,Rn	SHLD	Rm,Rn	XTRCT	Rm,Rn
MOV	#imm,Rn	SHLL	Rn		
MOVA	@(disp,PC),R0	SHLL16	Rn		

3. BR Group

BF	disp	BRA	disp	BT	disp
BF/S	disp	BSR	disp	BT/S	disp

Table 8.1 Instruction Groups (cont)

4. LS Group

FABS	DRn	FMOV.S	@Rm+,FRn	MOV.L	R0,@(disp,GBR)
FABS	FRn	FMOV.S	FRm,@(R0,Rn)	MOV.L	Rm,@(disp,Rn)
FLDI0	FRn	FMOV.S	FRm,@-Rn	MOV.L	Rm,@(R0,Rn)
FLDI1	FRn	FMOV.S	FRm,@Rn	MOV.L	Rm,@-Rn
FLDS	FRm,FPUL	FNEG	DRn	MOV.L	Rm,@Rn
FMOV	@(R0,Rm),DRn	FNEG	FRn	MOV.W	@(disp,GBR),R0
FMOV	@(R0,Rm),XDn	FSTS	FPUL,FRn	MOV.W	@(disp,PC),Rn
FMOV	@Rm,DRn	LDS	Rm,FPUL	MOV.W	@(disp,Rm),R0
FMOV	@Rm,XDn	MOV.B	@(disp,GBR),R0	MOV.W	@(R0,Rm),Rn
FMOV	@Rm+,DRn	MOV.B	@(disp,Rm),R0	MOV.W	@Rm,Rn
FMOV	@Rm+,XDn	MOV.B	@(R0,Rm),Rn	MOV.W	@Rm+,Rn
FMOV	DRm,@(R0,Rn)	MOV.B	@Rm,Rn	MOV.W	R0,@(disp,GBR)
FMOV	DRm,@-Rn	MOV.B	@Rm+,Rn	MOV.W	R0,@(disp,Rn)
FMOV	DRm,@Rn	MOV.B	R0,@(disp,GBR)	MOV.W	Rm,@(R0,Rn)
FMOV	DRm,DRn	MOV.B	R0,@(disp,Rn)	MOV.W	Rm,@-Rn
FMOV	DRm,XDn	MOV.B	Rm,@(R0,Rn)	MOV.W	Rm,@Rn
FMOV	FRm,FRn	MOV.B	Rm,@-Rn	MOVCA.L	R0,@Rn
FMOV	XDm,@(R0,Rn)	MOV.B	Rm,@Rn	OCBI	@Rn
FMOV	XDm,@-Rn	MOV.L	@(disp,GBR),R0	OCBP	@Rn
FMOV	XDm,@Rn	MOV.L	@(disp,PC),Rn	OCBWB	@Rn
FMOV	XDm,DRn	MOV.L	@(disp,Rm),Rn	PREF	@Rn
FMOV	XDm,XDn	MOV.L	@(R0,Rm),Rn	STS	FPUL,Rn
FMOV.S	@(R0,Rm),FRn	MOV.L	@Rm,Rn		
FMOV.S	@Rm,FRn	MOV.L	@Rm+,Rn		

Table 8.1 Instruction Groups (cont)

5. FE Group

FADD	DRm,DRn	FIPR	FVm,FVn	FSQRT	DRn
FADD	FRm,FRn	FLOAT	FPUL,DRn	FSQRT	FRn
FCMP/EQ	FRm,FRn	FLOAT	FPUL,FRn	FSUB	DRm,DRn
FCMP/GT	FRm,FRn	FMAC	FR0,FRm,FRn	FSUB	FRm,FRn
FCNVDS	DRm,FPUL	FMUL	DRm,DRn	FTRC	DRm,FPUL
FCNVSD	FPUL,DRn	FMUL	FRm,FRn	FTRC	FRm,FPUL
FDIV	DRm,DRn	FRCHG		FTRV	XMTRX,FVn
FDIV	FRm,FRn	FSCHG			

Table 8.1 Instruction Groups (cont)

6. CO Group

AND.B	#imm,@(R0,GBR)	LDS	Rm,FPSCR	STC	SR,Rn
BRAF	Rm	LDS	Rm,MACH	STC	SSR,Rn
BSRF	Rm	LDS	Rm,MACL	STC	VBR,Rn
CLRMAC		LDS	Rm,PR	STC.L	DBR,@-Rn
CLRS		LDS.L	@Rm+,FPSCR	STC.L	GBR,@-Rn
DMULS.L	Rm,Rn	LDS.L	@Rm+,FPUL	STC.L	Rp_BANK,@-Rn
DMULU.L	Rm,Rn	LDS.L	@Rm+,MACH	STC.L	SGR,@-Rn
FCMP/EQ	DRm,DRn	LDS.L	@Rm+,MACL	STC.L	SPC,@-Rn
FCMP/GT	DRm,DRn	LDS.L	@Rm+,PR	STC.L	SR,@-Rn
JMP	@Rn	LDTLB		STC.L	SSR,@-Rn
JSR	@Rn	MAC.L	@Rm+,@Rn+	STC.L	VBR,@-Rn
LDC	Rm,DBR	MAC.W	@Rm+,@Rn+	STS	FPSCR,Rn
LDC	Rm,GBR	MUL.L	Rm,Rn	STS	MACH,Rn
LDC	Rm,Rp_BANK	MULS.W	Rm,Rn	STS	MACL,Rn
LDC	Rm,SPC	MULU.W	Rm,Rn	STS	PR,Rn
LDC	Rm,SR	OR.B	#imm,@(R0,GBR)	STS.L	FPSCR,@-Rn
LDC	Rm,SSR	RTE		STS.L	FPUL,@-Rn
LDC	Rm,VBR	RTS		STS.L	MACH,@-Rn
LDC.L	@Rm+,DBR	SETS		STS.L	MACL,@-Rn
LDC.L	@Rm+,GBR	SLEEP		STS.L	PR,@-Rn
LDC.L	@Rm+,Rp_BANK	STC	DBR,Rn	TAS.B	@Rn
LDC.L	@Rm+,SPC	STC	GBR,Rn	TRAPA	#imm
LDC.L	@Rm+,SR	STC	Rp_BANK,Rn	TST.B	#imm,@(R0,GBR)
LDC.L	@Rm+,SSR	STC	SGR,Rn	XOR.B	#imm,@(R0,GBR)
LDC.L	@Rm+,VBR	STC	SPC,Rn		

Table 8.2 Parallel-Executability

		2nd Instruction					
		MT	EX	BR	LS	FE	CO
1st Instruction	MT	O	O	O	O	O	X
	EX	O	X	O	O	O	X
	BR	O	O	X	O	O	X
	LS	O	O	O	X	O	X
	FE	O	O	O	O	X	X
	CO	X	X	X	X	X	X

O: Can be executed in parallel

X: Cannot be executed in parallel

8.3 Execution Cycles and Pipeline Stalling

There are three basic clocks in this processor: the I-clock, B-clock, and P-clock. Each hardware unit operates on one of these clocks, as follows:

- I-clock: CPU, FPU, MMU, caches
- B-clock: External bus controller
- P-clock: Peripheral units

The frequency ratios of the three clocks are determined with the frequency control register (FRQCR). In this section, machine cycles are based on the I-clock unless otherwise specified. For details of FRQCR, see section 10, Clock Oscillation Circuits.

Instruction execution cycles are summarized in table 8.3. Penalty cycles due to a pipeline stall or freeze are not considered in this table.

- Issue rate: Interval between the issue of an instruction and that of the next instruction
- Latency: Interval between the issue of an instruction and the generation of its result (completion)
- Instruction execution pattern (see figure 8.2)
- Locked pipeline stages
- Interval between the issue of an instruction and the start of locking
- Lock time: Period of locking in machine cycle units

The instruction execution sequence is expressed as a combination of the execution patterns shown in figure 8.2. One instruction is separated from the next by the number of machine cycles for its issue rate. Normally, execution, data access, and write-back stages cannot be overlapped onto the same stages of another instruction; the only exception is when two instructions are executed in parallel under parallel-executability conditions. Refer to (a) through (d) in figure 8.3 for some simple examples.

Latency is the interval between issue and completion of an instruction, and is also the interval between the execution of two instructions with an interdependent relationship. When there is interdependency between two instructions fetched simultaneously, the latter of the two is stalled for the following number of cycles:

- (Latency) cycles when there is flow dependency (read-after-write)
- (Latency - 1) or (latency - 2) cycles when there is output dependency (write-after-write)
  - — Single/double-precision FDN, FSQRT is the preceding instruction (latency – 1) cycles
  - — The other FE group is the preceding instruction (latency – 2) cycles
  - — (Latency - 2) cycles
  - — Single-precision FDIV, FSQRT: (latency - 1) cycles
- 5 or 2 cycles when there is anti-flow dependency (write-after-read), as in the following cases:
  - — FTRV is the preceding instruction (5 cycle)
  - — A double-precision FADD, FSUB, or FMUL is the preceding instruction (2 cycles)

In the case of flow dependency, latency may be exceptionally increased or decreased, depending on the combination of sequential instructions (figure 8.3 (e)).

- When a floating-point (FP) computation is followed by an FP register store, the latency of the FP computation may be decreased by 1 cycle.
- If there is a load of the shift amount immediately before an SHAD/SHLD instruction, the latency of the load is increased by 1 cycle.
- If an instruction with a latency of less than 2 cycles, including write-back to an FP register, is followed by a double-precision FP instruction, FIPR, or FTRV, the latency of the first instruction is increased to 2 cycles.

The number of cycles in a pipeline stall due to flow dependency will vary depending on the combination of interdependent instructions or the fetch timing (see figure 8.3. (e)).

Output dependency occurs when the destination operands are the same in a preceding FE group instruction and a following LS group instruction.

For the stall cycles of an instruction with output dependency, the longest latency to the last write-back among all the destination operands must be applied instead of "latency" (see figure 8.3 (f)). A stall due to output dependency with respect to FPSCR, which reflects the result of an FP operation, never occurs. For example, when FADD follows FDIV with no dependency between FP registers, FADD is not stalled even if both instructions update the cause field of FPSCR.

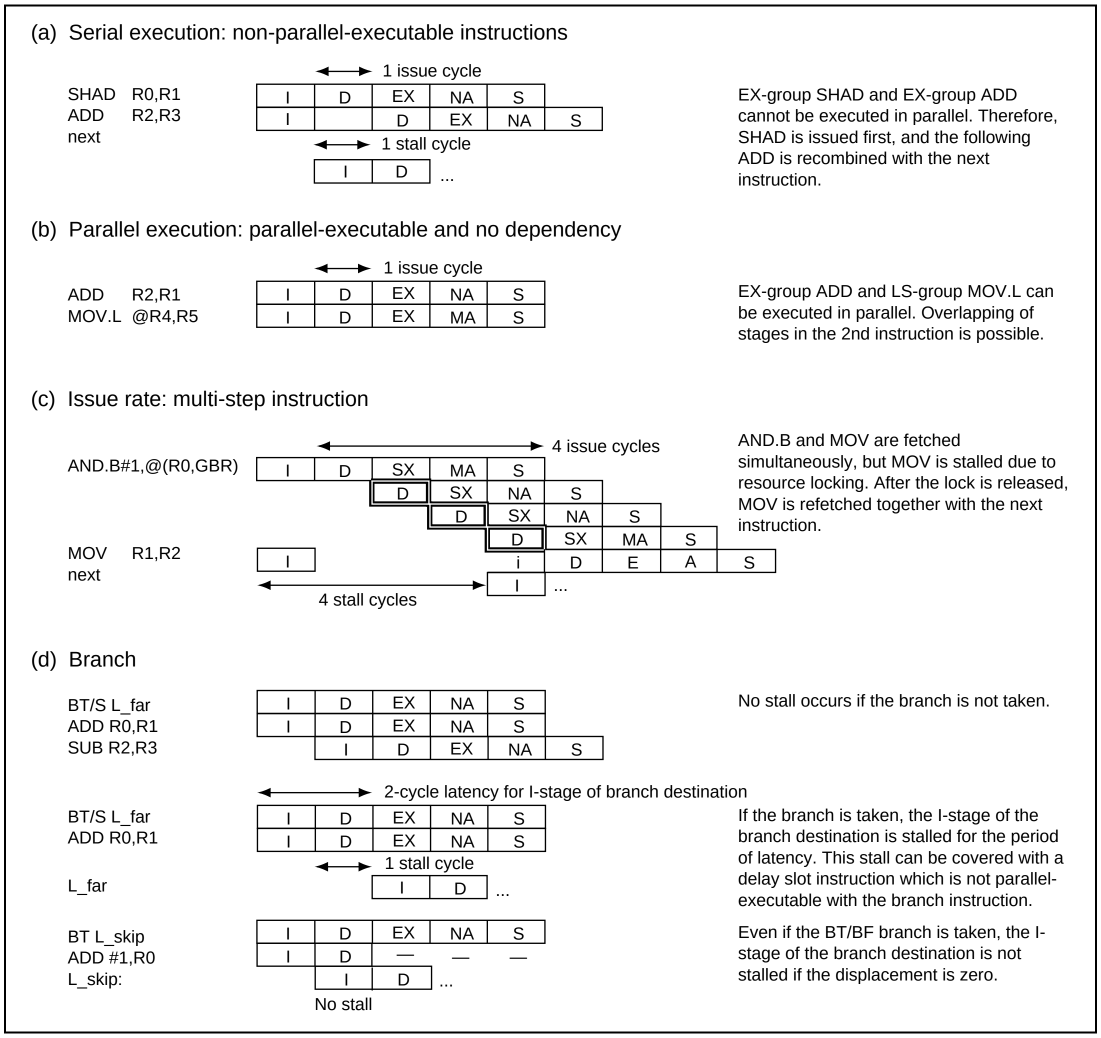
Anti-flow dependency can occur only between a preceding double-precision FADD, FMUL, FSUB, or FTRV and a following FMOV, FLDI0, FLDI1, FABS, FNEG, or FSTS. See figure 8.3 (g).

If an executing instruction locks any resource—i.e. a function block that performs a basic operation—a following instruction that happens to attempt to use the locked resource must be stalled (figure 8.3 (h)). This kind of stall can be compensated by inserting one or more instructions independent of the locked resource to separate the interfering instructions. For example, when a load instruction and an ADD instruction that references the loaded value are consecutive, the 2-cycle stall of the ADD is eliminated by inserting three instructions without dependency. Software performance can be improved by such instruction scheduling.

Other penalties arise in the event of exceptions or external data accesses, as follows.

- Instruction TLB miss: a penalty of 7 CPU clocks
- Instruction access to external memory (instruction cache miss, etc.)
- Data access to external memory (operand cache miss, etc.): a penalty of 2 CPU clocks + 3 bus clocks
- Data access to a memory-mapped control register. The penalty differs from register to register, and depends on the kind of operation (read or write), the clock mode, and the bus use conditions when the access is made.

During the penalty cycles of an instruction TLB miss or external instruction access, no instruction is issued, but execution of instructions that have already been issued continues. The penalty for a data access is a pipeline freeze: that is, the execution of uncompleted instructions is interrupted until the arrival of the requested data. The number of penalty cycles for instruction and data accesses is largely dependent on the user's memory subsystems.



EX-group SHAD and EX-group ADD cannot be executed in parallel. Therefore, SHAD is issued first, and the following ADD is recombined with the next instruction.

EX-group ADD and LS-group MOV.L can be executed in parallel. Overlapping of stages in the 2nd instruction is possible.

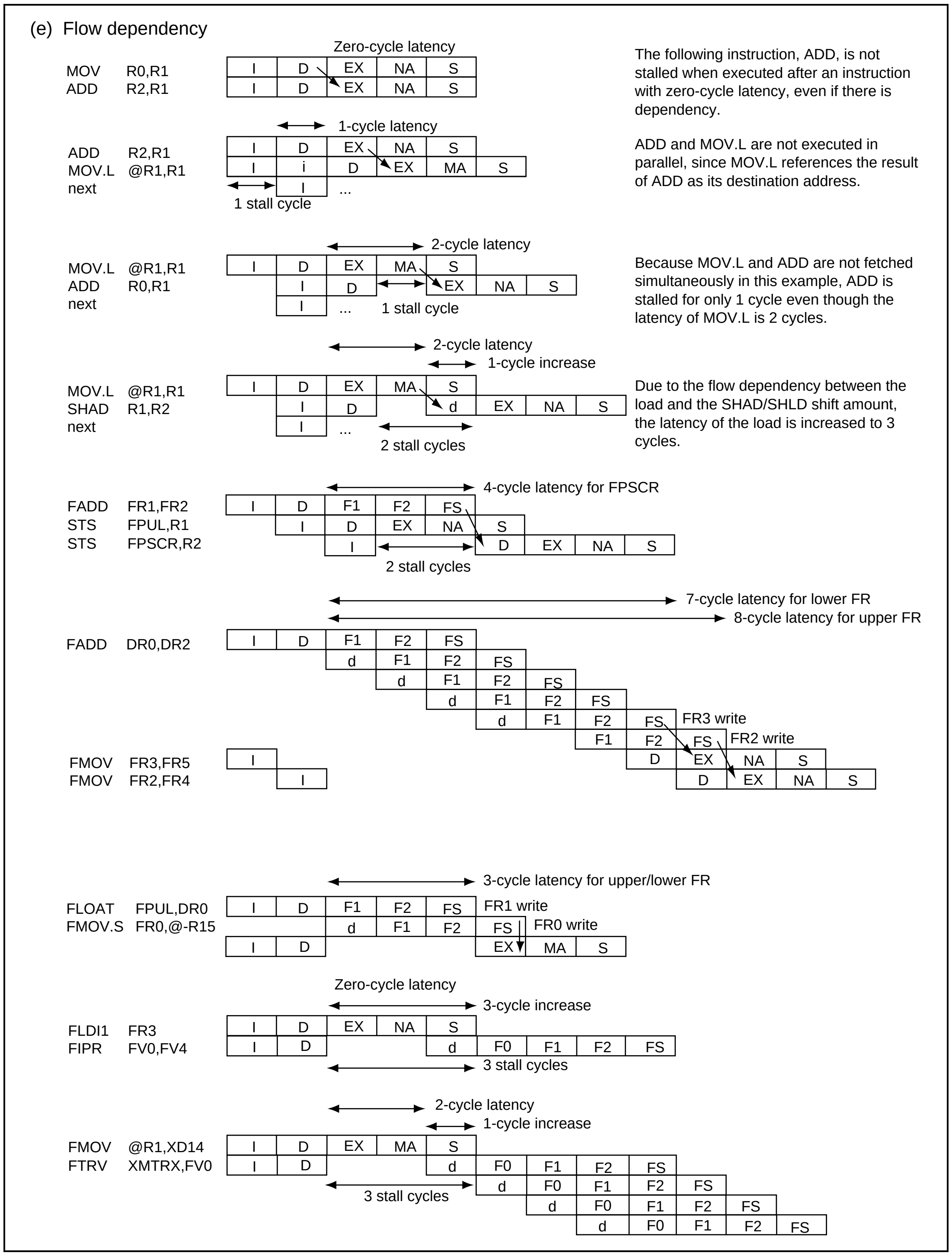
AND.B and MOV are fetched simultaneously, but MOV is stalled due to resource locking. After the lock is released, MOV is refetched together with the next instruction.

No stall occurs if the branch is not taken.

If the branch is taken, the I-stage of the branch destination is stalled for the period of latency. This stall can be covered with a delay slot instruction which is not parallel-executable with the branch instruction.

Even if the BT/BF branch is taken, the I-stage of the branch destination is not stalled if the displacement is zero.

Figure 8.3 Examples of Pipelined Execution



(e) Flow dependency
MOV R0,R1
ADD R2,R1
Zero-cycle latency
I D EX NA S
The following instruction, ADD, is not stalled when executed after an instruction with zero-cycle latency, even if there is dependency.
ADD R2,R1
MOV.L @R1,R1
next
1-cycle latency
1 stall cycle
ADD and MOV.L are not executed in parallel, since MOV.L references the result of ADD as its destination address.
MOV.L @R1,R1
ADD R0,R1
next
2-cycle latency
1 stall cycle
Because MOV.L and ADD are not fetched simultaneously in this example, ADD is stalled for only 1 cycle even though the latency of MOV.L is 2 cycles.
MOV.L @R1,R1
SHAD R1,R2
next
2-cycle latency
1-cycle increase
2 stall cycles
Due to the flow dependency between the load and the SHAD/SHLD shift amount, the latency of the load is increased to 3 cycles.
FADD FR1,FR2
STS FPUL,R1
STS FPSCR,R2
4-cycle latency for FPSCR
2 stall cycles
FADD DR0,DR2
FMOV FR3,FR5
FMOV FR2,FR4
7-cycle latency for lower FR
8-cycle latency for upper FR
FR3 write
FR2 write
FLOAT FPUL,DR0
FMOV.S FR0,@-R15
3-cycle latency for upper/lower FR
FR1 write
FR0 write
FLDI1 FR3
FIPR FV0,FV4
Zero-cycle latency
3-cycle increase
3 stall cycles
FMOV @R1,XD14
FTRV XMTRX,FV0
2-cycle latency
1-cycle increase
3 stall cycles

Figure 8.3 Examples of Pipelined Execution (cont)

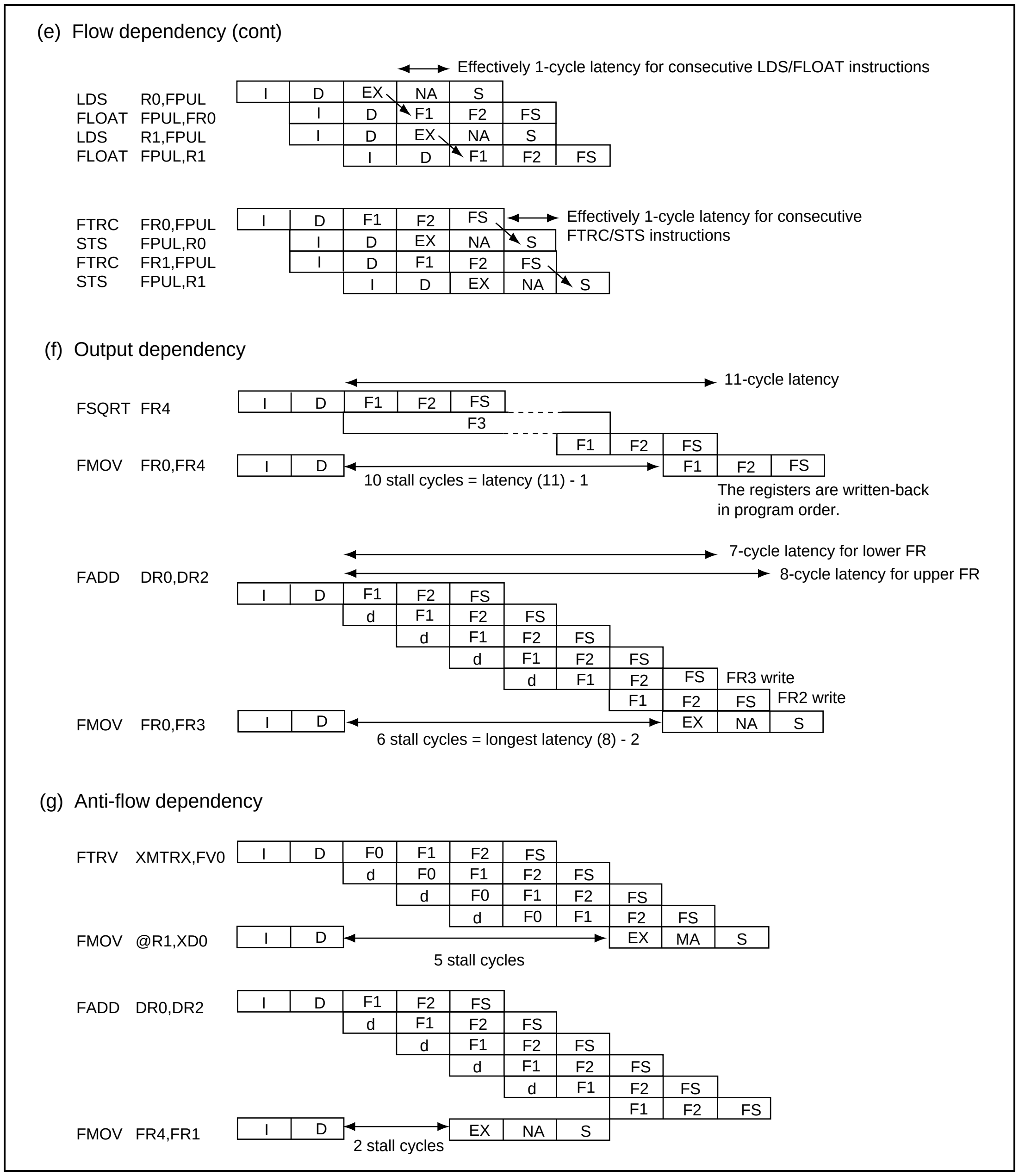

(e) Flow dependency (cont)
Effectively 1-cycle latency for consecutive LDS/FLOAT instructions
LDS R0,FPUL
FLOAT FPUL,FR0
LDS R1,FPUL
FLOAT FPUL,R1
FTRC FR0,FPUL
STS FPUL,R0
FTRC FR1,FPUL
STS FPUL,R1
Effectively 1-cycle latency for consecutive FTRC/STS instructions
(f) Output dependency
11-cycle latency
FSQRT FR4
FMOV FR0,FR4
10 stall cycles = latency (11) - 1
The registers are written-back in program order.
7-cycle latency for lower FR
8-cycle latency for upper FR
FADD DR0,DR2
FR3 write
FR2 write
FMOV FR0,FR3
6 stall cycles = longest latency (8) - 2
(g) Anti-flow dependency
FTRV XMTRX,FV0
FMOV @R1,XD0
5 stall cycles
FADD DR0,DR2
FMOV FR4,FR1
2 stall cycles

Figure 8.3 Examples of Pipelined Execution (cont)

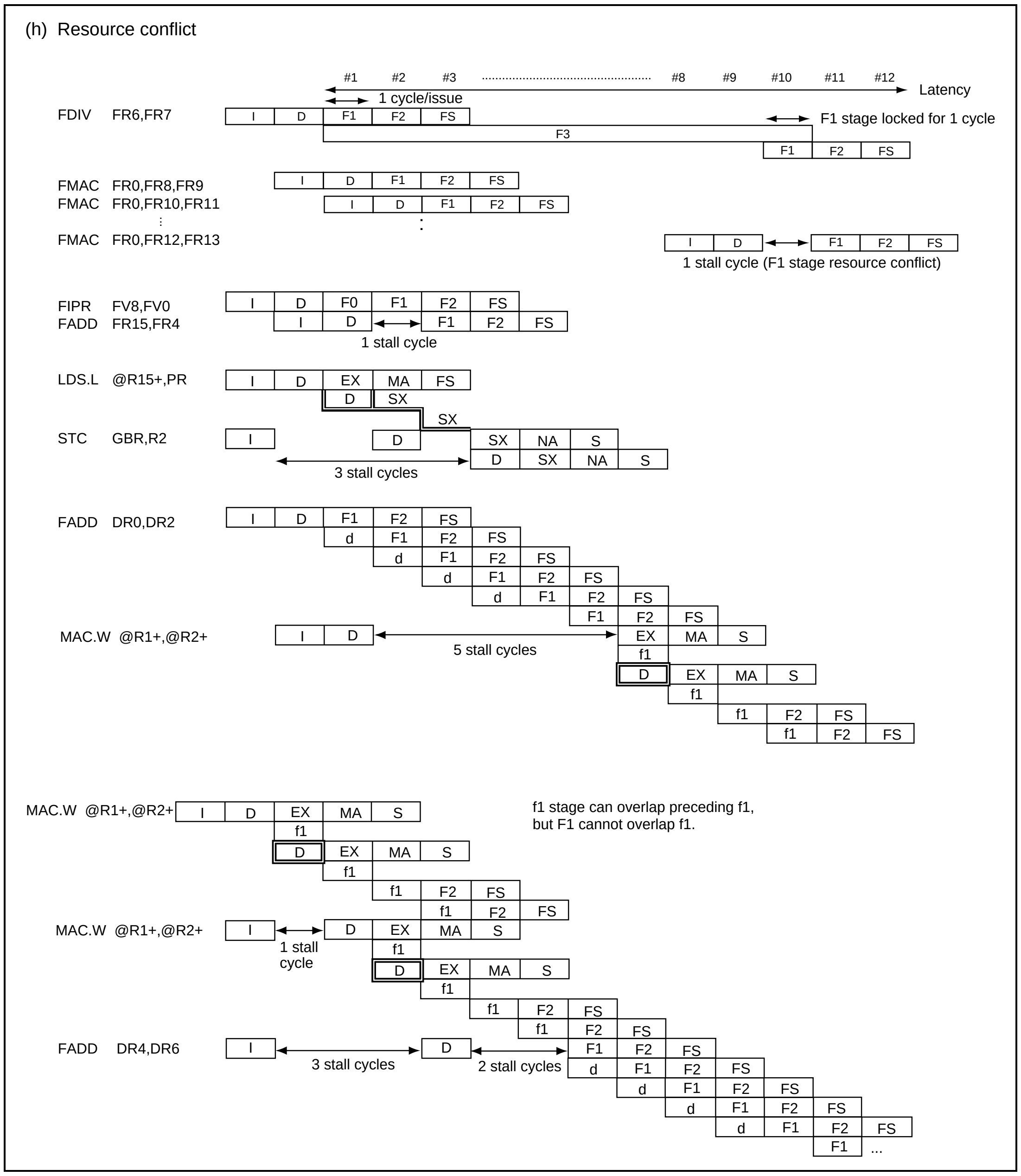


Figure 8.3 Examples of Pipelined Execution (cont)

Table 8.3 Execution Cycles

Functional Category	No.	Instruction		Instruction Group	Issue Rate	Latency	Execution Pattern	Lock Stage	Lock Start	Lock Cycles
Data transfer instructions	1	EXTS.B	Rm,Rn	EX	1	1	#1	—	—	—
	2	EXTS.W	Rm,Rn	EX	1	1	#1	—	—	—
	3	EXTU.B	Rm,Rn	EX	1	1	#1	—	—	—
	4	EXTU.W	Rm,Rn	EX	1	1	#1	—	—	—
	5	MOV	Rm,Rn	MT	1	0	#1	—	—	—
	6	MOV	#imm,Rn	EX	1	1	#1	—	—	—
	7	MOVA	@(disp,PC),R0	EX	1	1	#1	—	—	—
	8	MOV.W	@(disp,PC),Rn	LS	1	2	#2	—	—	—
	9	MOV.L	@(disp,PC),Rn	LS	1	2	#2	—	—	—
	10	MOV.B	@Rm,Rn	LS	1	2	#2	—	—	—
	11	MOV.W	@Rm,Rn	LS	1	2	#2	—	—	—
	12	MOV.L	@Rm,Rn	LS	1	2	#2	—	—	—
	13	MOV.B	@Rm+,Rn	LS	1	1/2	#2	—	—	—
	14	MOV.W	@Rm+,Rn	LS	1	1/2	#2	—	—	—
	15	MOV.L	@Rm+,Rn	LS	1	1/2	#2	—	—	—
	16	MOV.B	@(disp,Rm),R0	LS	1	2	#2	—	—	—
	17	MOV.W	@(disp,Rm),R0	LS	1	2	#2	—	—	—
	18	MOV.L	@(disp,Rm),Rn	LS	1	2	#2	—	—	—
	19	MOV.B	@(R0,Rm),Rn	LS	1	2	#2	—	—	—
	20	MOV.W	@(R0,Rm),Rn	LS	1	2	#2	—	—	—
	21	MOV.L	@(R0,Rm),Rn	LS	1	2	#2	—	—	—
	22	MOV.B	@(disp,GBR),R0	LS	1	2	#3	—	—	—
	23	MOV.W	@(disp,GBR),R0	LS	1	2	#3	—	—	—
	24	MOV.L	@(disp,GBR),R0	LS	1	2	#3	—	—	—
	25	MOV.B	Rm,@Rn	LS	1	1	#2	—	—	—
	26	MOV.W	Rm,@Rn	LS	1	1	#2	—	—	—
	27	MOV.L	Rm,@Rn	LS	1	1	#2	—	—	—
	28	MOV.B	Rm,@-Rn	LS	1	1/1	#2	—	—	—
	29	MOV.W	Rm,@-Rn	LS	1	1/1	#2	—	—	—
	30	MOV.L	Rm,@-Rn	LS	1	1/1	#2	—	—	—
	31	MOV.B	R0,@(disp,Rn)	LS	1	1	#2	—	—	—

Table 8.3 Execution Cycles (cont)

Functional Category	No.	Instruction		Instruction Group	Issue Rate	Latency	Execution Pattern	Lock Stage	Lock Start	Lock Cycles
Data transfer instructions	32	MOV.W	R0,@(disp,Rn)	LS	1	1	#2	—	—	—
	33	MOV.L	Rm,@(disp,Rn)	LS	1	1	#2	—	—	—
	34	MOV.B	Rm,@(R0,Rn)	LS	1	1	#2	—	—	—
	35	MOV.W	Rm,@(R0,Rn)	LS	1	1	#2	—	—	—
	36	MOV.L	Rm,@(R0,Rn)	LS	1	1	#2	—	—	—
	37	MOV.B	R0,@(disp,GBR)	LS	1	1	#3	—	—	—
	38	MOV.W	R0,@(disp,GBR)	LS	1	1	#3	—	—	—
	39	MOV.L	R0,@(disp,GBR)	LS	1	1	#3	—	—	—
	40	MOVCA.L	R0,@Rn	LS	1	3–7	#12	MA	4	3–7
	41	MOVT	Rn	EX	1	1	#1	—	—	—
	42	OCBI	@Rn	LS	1	1–2	#10	MA	4	1–2
	43	OCBP	@Rn	LS	1	1–5	#11	MA	4	1–5
	44	OCBWB	@Rn	LS	1	1–5	#11	MA	4	1–5
	45	PREF	@Rn	LS	1	1	#2	—	—	—
	46	SWAP.B	Rm,Rn	EX	1	1	#1	—	—	—
	47	SWAP.W	Rm,Rn	EX	1	1	#1	—	—	—
	48	XTRCT	Rm,Rn	EX	1	1	#1	—	—	—
Fixed-point arithmetic instructions	49	ADD	Rm,Rn	EX	1	1	#1	—	—	—
	50	ADD	#imm,Rn	EX	1	1	#1	—	—	—
	51	ADDC	Rm,Rn	EX	1	1	#1	—	—	—
	52	ADDV	Rm,Rn	EX	1	1	#1	—	—	—
	53	CMP/EQ	#imm,R0	MT	1	1	#1	—	—	—
	54	CMP/EQ	Rm,Rn	MT	1	1	#1	—	—	—
	55	CMP/GE	Rm,Rn	MT	1	1	#1	—	—	—
	56	CMP/GT	Rm,Rn	MT	1	1	#1	—	—	—
	57	CMP/HI	Rm,Rn	MT	1	1	#1	—	—	—
	58	CMP/HS	Rm,Rn	MT	1	1	#1	—	—	—
	59	CMP/PL	Rn	MT	1	1	#1	—	—	—
	60	CMP/PZ	Rn	MT	1	1	#1	—	—	—
	61	CMP/STR	Rm,Rn	MT	1	1	#1	—	—	—
	62	DIV0S	Rm,Rn	EX	1	1	#1	—	—	—

Table 8.3 Execution Cycles (cont)

Functional Category	No.	Instruction		Instruction Group	Issue Rate	Latency	Execution Pattern	Lock Stage	Lock Start	Lock Cycles
Fixed-point arithmetic instructions	63	DIV0U		EX	1	1	#1	—	—	—
	64	DIV1	Rm,Rn	EX	1	1	#1	—	—	—
	65	DMULS.L	Rm,Rn	CO	2	4/4	#34	F1	4	2
	66	DMULU.L	Rm,Rn	CO	2	4/4	#34	F1	4	2
	67	DT	Rn	EX	1	1	#1	—	—	—
	68	MAC.L	@Rm+,@Rn+	CO	2	2/2/4/4	#35	F1	4	2
	69	MAC.W	@Rm+,@Rn+	CO	2	2/2/4/4	#35	F1	4	2
	70	MUL.L	Rm,Rn	CO	2	4/4	#34	F1	4	2
	71	MULS.W	Rm,Rn	CO	2	4/4	#34	F1	4	2
	72	MULU.W	Rm,Rn	CO	2	4/4	#34	F1	4	2
	73	NEG	Rm,Rn	EX	1	1	#1	—	—	—
	74	NEGC	Rm,Rn	EX	1	1	#1	—	—	—
	75	SUB	Rm,Rn	EX	1	1	#1	—	—	—
	76	SUBC	Rm,Rn	EX	1	1	#1	—	—	—
	77	SUBV	Rm,Rn	EX	1	1	#1	—	—	—
Logical instructions	78	AND	Rm,Rn	EX	1	1	#1	—	—	—
	79	AND	#imm,R0	EX	1	1	#1	—	—	—
	80	AND.B	#imm,@(R0,GBR)	CO	4	4	#6	—	—	—
	81	NOT	Rm,Rn	EX	1	1	#1	—	—	—
	82	OR	Rm,Rn	EX	1	1	#1	—	—	—
	83	OR	#imm,R0	EX	1	1	#1	—	—	—
	84	OR.B	#imm,@(R0,GBR)	CO	4	4	#6	—	—	—
	85	TAS.B	@Rn	CO	5	5	#7	—	—	—
	86	TST	Rm,Rn	MT	1	1	#1	—	—	—
	87	TST	#imm,R0	MT	1	1	#1	—	—	—
	88	TST.B	#imm,@(R0,GBR)	CO	3	3	#5	—	—	—
	89	XOR	Rm,Rn	EX	1	1	#1	—	—	—
	90	XOR	#imm,R0	EX	1	1	#1	—	—	—
	91	XOR.B	#imm,@(R0,GBR)	CO	4	4	#6	—	—	—

Table 8.3 Execution Cycles (cont)

Functional Category	No.	Instruction		Instruction Group	Issue Rate	Latency	Execution Pattern	Lock Stage	Lock Start	Lock Cycles
Shift instructions	92	ROTL	Rn	EX	1	1	#1	—	—	—
	93	ROTR	Rn	EX	1	1	#1	—	—	—
	94	ROTCL	Rn	EX	1	1	#1	—	—	—
	95	ROTCR	Rn	EX	1	1	#1	—	—	—
	96	SHAD	Rm,Rn	EX	1	1	#1	—	—	—
	97	SHAL	Rn	EX	1	1	#1	—	—	—
	98	SHAR	Rn	EX	1	1	#1	—	—	—
	99	SHLD	Rm,Rn	EX	1	1	#1	—	—	—
	100	SHLL	Rn	EX	1	1	#1	—	—	—
	101	SHLL2	Rn	EX	1	1	#1	—	—	—
	102	SHLL8	Rn	EX	1	1	#1	—	—	—
	103	SHLL16	Rn	EX	1	1	#1	—	—	—
	104	SHLR	Rn	EX	1	1	#1	—	—	—
	105	SHLR2	Rn	EX	1	1	#1	—	—	—
	106	SHLR8	Rn	EX	1	1	#1	—	—	—
	107	SHLR16	Rn	EX	1	1	#1	—	—	—
Branch instructions	108	BF	disp	BR	1	2 (or 1)	#1	—	—	—
	109	BF/S	disp	BR	1	2 (or 1)	#1	—	—	—
	110	BT	disp	BR	1	2 (or 1)	#1	—	—	—
	111	BT/S	disp	BR	1	2 (or 1)	#1	—	—	—
	112	BRA	disp	BR	1	2	#1	—	—	—
	113	BRAF	Rn	CO	2	3	#4	—	—	—
	114	BSR	disp	BR	1	2	#14	SX	3	2
	115	BSRF	Rn	CO	2	3	#24	SX	3	2
	116	JMP	@Rn	CO	2	3	#4	—	—	—
	117	JSR	@Rn	CO	2	3	#24	SX	3	2
	118	RTS		CO	2	3	#4	—	—	—

Table 8.3 Execution Cycles (cont)

Functional Category	No.	Instruction		Instruction Group	Issue Rate	Latency	Execution Pattern	Lock Stage	Lock Start	Lock Cycles
System control instructions	119	NOP		MT	1	0	#1	—	—	—
	120	CLRMAC		CO	1	3	#28	F1	3	2
	121	CLRS		CO	1	1	#1	—	—	—
	122	CLRT		MT	1	1	#1	—	—	—
	123	SETS		CO	1	1	#1	—	—	—
	124	SETT		MT	1	1	#1	—	—	—
	125	TRAPA	#imm	CO	7	7	#13	—	—	—
	126	RTE		CO	5	5	#8	—	—	—
	127	SLEEP		CO	4	4	#9	—	—	—
	128	LDTLB		CO	1	1	#2	—	—	—
	129	LDC	Rm,DBR	CO	1	3	#14	SX	3	2
	130	LDC	Rm,GBR	CO	3	3	#15	SX	3	2
	131	LDC	Rm,Rp_BANK	CO	1	3	#14	SX	3	2
	132	LDC	Rm,SR	CO	4	4	#16	SX	3	2
	133	LDC	Rm,SSR	CO	1	3	#14	SX	3	2
	134	LDC	Rm,SPC	CO	1	3	#14	SX	3	2
	135	LDC	Rm,VBR	CO	1	3	#14	SX	3	2
	136	LDC.L	@Rm+,DBR	CO	1	1/3	#17	SX	3	2
	137	LDC.L	@Rm+,GBR	CO	3	3/3	#18	SX	3	2
	138	LDC.L	@Rm+,Rp_BANK	CO	1	1/3	#17	SX	3	2
	139	LDC.L	@Rm+,SR	CO	4	4/4	#19	SX	3	2
	140	LDC.L	@Rm+,SSR	CO	1	1/3	#17	SX	3	2
	141	LDC.L	@Rm+,SPC	CO	1	1/3	#17	SX	3	2
	142	LDC.L	@Rm+,VBR	CO	1	1/3	#17	SX	3	2
	143	LDS	Rm,MACH	CO	1	3	#28	F1	3	2
	144	LDS	Rm,MACL	CO	1	3	#28	F1	3	2
	145	LDS	Rm,PR	CO	2	3	#24	SX	3	2
	146	LDS.L	@Rm+,MACH	CO	1	1/3	#29	F1	3	2
	147	LDS.L	@Rm+,MACL	CO	1	1/3	#29	F1	3	2
	148	LDS.L	@Rm+,PR	CO	2	2/3	#25	SX	3	2
	149	STC	DBR,Rn	CO	2	2	#20	—	—	—
	150	STC	SGR,Rn	CO	3	3	#21	—	—	—

Table 8.3 Execution Cycles (cont)

Functional Category	No.	Instruction		Instruction Group	Issue Rate	Latency	Execution Pattern	Lock Stage	Lock Start	Lock Cycles
System control instructions	151	STC	GBR,Rn	CO	2	2	#20	—	—	—
	152	STC	Rp_BANK,Rn	CO	2	2	#20	—	—	—
	153	STC	SR,Rn	CO	2	2	#20	—	—	—
	154	STC	SSR,Rn	CO	2	2	#20	—	—	—
	155	STC	SPC,Rn	CO	2	2	#20	—	—	—
	156	STC	VBR,Rn	CO	2	2	#20	—	—	—
	157	STC.L	DBR,@-Rn	CO	2	2/2	#22	—	—	—
	158	STC.L	SGR,@-Rn	CO	3	3/3	#23	—	—	—
	159	STC.L	GBR,@-Rn	CO	2	2/2	#22	—	—	—
	160	STC.L	Rp_BANK,@-Rn	CO	2	2/2	#22	—	—	—
	161	STC.L	SR,@-Rn	CO	2	2/2	#22	—	—	—
	162	STC.L	SSR,@-Rn	CO	2	2/2	#22	—	—	—
	163	STC.L	SPC,@-Rn	CO	2	2/2	#22	—	—	—
	164	STC.L	VBR,@-Rn	CO	2	2/2	#22	—	—	—
	165	STS	MACH,Rn	CO	1	3	#30	—	—	—
	166	STS	MACL,Rn	CO	1	3	#30	—	—	—
	167	STS	PR,Rn	CO	2	2	#26	—	—	—
	168	STS.L	MACH,@-Rn	CO	1	1/1	#31	—	—	—
	169	STS.L	MACL,@-Rn	CO	1	1/1	#31	—	—	—
	170	STS.L	PR,@-Rn	CO	2	2/2	#27	—	—	—
Single-precision floating-point instructions	171	FLDI0	FRn	LS	1	0	#1	—	—	—
	172	FLDI1	FRn	LS	1	0	#1	—	—	—
	173	FMOV	FRm,FRn	LS	1	0	#1	—	—	—
	174	FMOV.S	@Rm,FRn	LS	1	2	#2	—	—	—
	175	FMOV.S	@Rm+,FRn	LS	1	1/2	#2	—	—	—
	176	FMOV.S	@(R0,Rm),FRn	LS	1	2	#2	—	—	—
	177	FMOV.S	FRm,@Rn	LS	1	1	#2	—	—	—
	178	FMOV.S	FRm,@-Rn	LS	1	1/1	#2	—	—	—
	179	FMOV.S	FRm,@(R0,Rn)	LS	1	1	#2	—	—	—
	180	FLDS	FRm,FPUL	LS	1	0	#1	—	—	—
	181	FSTS	FPUL,FRn	LS	1	0	#1	—	—	—

Table 8.3 Execution Cycles (cont)

Functional Category	No.	Instruction		Instruction Group	Issue Rate	Latency	Execution Pattern	Lock Stage	Lock Start	Lock Cycles
Single-precision floating-point instructions	182	FABS	FRn	LS	1	0	#1	—	—	—
	183	FADD	FRm,FRn	FE	1	3/4	#36	—	—	—
	184	FCMP/EQ	FRm,FRn	FE	1	2/4	#36	—	—	—
	185	FCMP/GT	FRm,FRn	FE	1	2/4	#36	—	—	—
	186	FDIV	FRm,FRn	FE	1	12/13	#37	F3	2	10
								F1	11	1
	187	FLOAT	FPUL,FRn	FE	1	3/4	#36	—	—	—
	188	FMAC	FR0,FRm,FRn	FE	1	3/4	#36	—	—	—
	189	FMUL	FRm,FRn	FE	1	3/4	#36	—	—	—
	190	FNEG	FRn	LS	1	0	#1	—	—	—
	191	FSQRT	FRn	FE	1	11/12	#37	F3	2	9
								F1	10	1
	192	FSUB	FRm,FRn	FE	1	3/4	#36	—	—	—
	193	FTRC	FRm,FPUL	FE	1	3/4	#36	—	—	—
	194	FMOV	DRm,DRn	LS	1	0	#1	—	—	—
	195	FMOV	@Rm,DRn	LS	1	2	#2	—	—	—
	196	FMOV	@Rm+,DRn	LS	1	1/2	#2	—	—	—
	197	FMOV	@(R0,Rm),DRn	LS	1	2	#2	—	—	—
	198	FMOV	DRm,@Rn	LS	1	1	#2	—	—	—
	199	FMOV	DRm,@-Rn	LS	1	1/1	#2	—	—	—
	200	FMOV	DRm,@(R0,Rn)	LS	1	1	#2	—	—	—
Double-precision floating-point instructions	201	FABS	DRn	LS	1	0	#1	—	—	—
	202	FADD	DRm,DRn	FE	1	(7, 8)/9	#39	F1	2	6
	203	FCMP/EQ	DRm,DRn	CO	2	3/5	#40	F1	2	2
	204	FCMP/GT	DRm,DRn	CO	2	3/5	#40	F1	2	2
	205	FCNVDS	DRm,FPUL	FE	1	4/5	#38	F1	2	2
	206	FCNVSD	FPUL,DRn	FE	1	(3, 4)/5	#38	F1	2	2
	207	FDIV	DRm,DRn	FE	1	(24, 25)/ 26	#41	F3	2	23
								F1	22	3
								F1	2	2
	208	FLOAT	FPUL,DRn	FE	1	(3, 4)/5	#38	F1	2	2
	209	FMUL	DRm,DRn	FE	1	(7, 8)/9	#39	F1	2	6

Table 8.3 Execution Cycles (cont)

Functional Category	No.	Instruction		Instruction Group	Issue Rate	Latency	Execution Pattern	Lock		
								Stage	Start	Cycles
Double-precision floating-point instructions	210	FNEG	DRn	LS	1	0	#1	—	—	—
	211	FSQRT	DRn	FE	1	(23, 24)/ 25	#41	F3	2	22
								F1	21	3
								F1	2	2
	212	FSUB	DRm,DRn	FE	1	(7, 8)/9	#39	F1	2	6
	213	FTRC	DRm,FPUL	FE	1	4/5	#38	F1	2	2
FPU system control instructions	214	LDS	Rm,FPUL	LS	1	1	#1	—	—	—
	215	LDS	Rm,FPSCR	CO	1	4	#32	F1	3	3
	216	LDS.L	@Rm+,FPUL	CO	1	1/2	#2	—	—	—
	217	LDS.L	@Rm+,FPSCR	CO	1	1/4	#33	F1	3	3
	218	STS	FPUL,Rn	LS	1	3	#1	—	—	—
	219	STS	FPSCR,Rn	CO	1	3	#1	—	—	—
	220	STS.L	FPUL,@-Rn	CO	1	1/1	#2	—	—	—
	221	STS.L	FPSCR,@-Rn	CO	1	1/1	#2	—	—	—
Graphics acceleration instructions	222	FMOV	DRm,XDn	LS	1	0	#1	—	—	—
	223	FMOV	XDm,DRn	LS	1	0	#1	—	—	—
	224	FMOV	XDm,XDn	LS	1	0	#1	—	—	—
	225	FMOV	@Rm,XDn	LS	1	2	#2	—	—	—
	226	FMOV	@Rm+,XDn	LS	1	1/2	#2	—	—	—
	227	FMOV	@(R0,Rm),XDn	LS	1	2	#2	—	—	—
	228	FMOV	XDm,@Rn	LS	1	1	#2	—	—	—
	229	FMOV	XDm,@-Rm	LS	1	1/1	#2	—	—	—
	230	FMOV	XDm,@(R0,Rn)	LS	1	1	#2	—	—	—
	231	FIPR	FVm,FVn	FE	1	4/5	#42	F1	3	1
	232	FRCHG		FE	1	1/4	#36	—	—	—
	233	FSCHG		FE	1	1/4	#36	—	—	—
	234	FTRV	XMTRX,FVn	FE	1	(5, 5, 6, 7)/8	#43	F0	2	4
								F1	3	4

Notes: 1. See table 8.1 for the instruction groups.

2. Latency "L1/L2...": Latency corresponding to a write to each register, including MACH/MACL/FPSCR.

Example: MOV.B @Rm+, Rn "1/2": The latency for Rm is 1 cycle, and the latency for Rn is 2 cycles.

3. Branch latency: Interval until the branch destination instruction is fetched
4. Conditional branch latency “2 (or 1)”: The latency is 2 for a nonzero displacement, and 1 for a zero displacement.
5. Double-precision floating-point instruction latency “(L1, L2)/L3”: L1 is the latency for FR [n+1], L2 that for FR [n], and L3 that for FPSCR.
6. FTRV latency “(L1, L2, L3, L4)/L5”: L1 is the latency for FR [n], L2 that for FR [n+1], L3 that for FR [n+2], L4 that for FR [n+3], and L5 that for FPSCR.
7. Latency “L1/L2/L3/L4” of MAC.L and MAC.W instructions: L1 is the latency for Rm, L2 that for Rn, L3 that for MACH, and L4 that for MACL.
8. Latency “L1/L2” of MUL.L, MULS.W, MULU.W, DMULS.L, and DMULU.L instructions: L1 is the latency for MACH, and L2 that for MACL.
9. Execution pattern: The instruction execution pattern number (see figure 8.2)
10. Lock/stage: Stage locked by the instruction
11. Lock/start: Locking start cycle; 1 is the first D-stage of the instruction.
12. Lock/cycles: Number of cycles locked

Exceptions:

1. When a floating-point computation instruction is followed by an FMOV store, an STS FPUL, Rn instruction, or an STS.L FPUL, @-Rn instruction, the latency of the floating-point computation is decreased by 1 cycle.
2. When the preceding instruction loads the shift amount of the following SHAD/SHLD, the latency of the load is increased by 1 cycle.
3. When an LS group instruction with a latency of less than 3 cycles is followed by a double-precision floating-point instruction, FIPR, or FTRV, the latency of the first instruction is increased to 3 cycles.

   Example: In the case of FMOV FR4,FR0 and FIPR FV0,FV4, FIPR is stalled for 2 cycles.
4. When MAC*/MUL*/DMUL* is followed by an STS.L MAC*, @-Rn instruction, the latency of MAC*/MUL*/DMUL* is 5 cycles.
5. In the case of consecutive executions of MAC*/MUL*/DMUL*, the latency is decreased to 2 cycles.
6. When an LDS to MAC* is followed by an STS.L MAC*, @-Rn instruction, the latency of the LDS to MAC* is 4 cycles.
7. When an LDS to MAC* is followed by MAC*/MUL*/DMUL*, the latency of the LDS to MAC* is 1 cycle.
8. When an FSCHG or FRCHG instruction is followed by an LS group instruction that reads or writes to a floating-point register, the aforementioned LS group instruction[s] cannot be executed in parallel.
9. When a single-precision FTRC instruction is followed by an STS FPUL, Rn instruction, the latency of the single-precision FTRC instruction is 1 cycle.

Section 9   Power-Down Modes

9.1 Overview

In the power-down modes, some of the on-chip peripheral modules and the CPU functions are halted, enabling power consumption to be reduced.

9.1.1 Types of Power-Down Modes

The following power-down modes and functions are provided:

- Sleep mode
- Deep sleep mode
- Standby mode
- Module standby function (TMU, RTC, SCI/SCIF, and DMAC on-chip peripheral modules)

Table 9.1 shows the conditions for entering these modes from the program execution state, the status of the CPU and peripheral modules in each mode, and the method of exiting each mode.

Table 9.1 Status of CPU and Peripheral Modules in Power-Down Modes

		Status						
Power-Down Mode	**Entering Conditions**	**CPG**	**CPU**	**On-Chip Memory**	**On-chip Peripheral Modules**	**Pins**	**External Memory**	**Exiting Method**
Sleep	SLEEP instruction executed while STBY bit is 0 in STBCR	Operating	Halted (registers held)	Held	Operating	Held	Refreshing	• Interrupt • Reset
Deep sleep	SLEEP instruction executed while STBY bit is 0 in STBCR, and DSLP bit is 1 in STBCR2	Operating	Halted (registers held)	Held	Operating (DMA halted)	Held	Self-refreshing	• Interrupt • Reset
Standby	SLEEP instruction executed while STBY bit is 1 in STBCR	Halted	Halted (registers held)	Held	Halted*	Held	Self-refreshing	• Interrupt • Reset
Module standby	Setting MSTP bit to 1 in STBCR	Operating	Operating	Held	Specified modules halted*	Held	Refreshing	• Clearing MSTP bit to 0 • Reset

Note: The RTC operates when the START bit in RCR2 is 1 (see section 11, Realtime Clock (RTC)).

9.1.2 Register Configuration

Table 9.2 shows the registers used for power-down mode control.

Table 9.2 Power-Down Mode Registers

Name	Abbreviation	R/W	Initial Value	P4 Address	Area 7 Address	Access Size
Standby control register	STBCR	R/W	H'00	H'FFC00004	H'1FC00004	8
Standby control register 2	STBCR2	R/W	H'00	H'FFC00010	H'1FC00010	8

9.1.3 Pin Configuration

Table 9.3 shows the pins used for power-down mode control.

Table 9.3 Power-Down Mode Pins

Pin Name	Abbreviation	I/O	Function
Processor status 1	STATUS1	Output	Indicate the processor's operating status.
Processor status 0	STATUS0		HH: Reset HL: Sleep mode LH: Standby mode LL: Normal operation

Note: H: High level
L: Low level

9.2 Register Descriptions

9.2.1 Standby Control Register (STBCR)

The standby control register (STBCR) is an 8-bit readable/writable register that specifies the power-down mode status. It is initialized to H'00 by a power-on reset via the $\overline{\text{RESET}}$ pin or due to watchdog timer overflow.

Bit:	7	6	5	4	3	2	1	0
	STBY	PHZ	PPU	MSTP4	MSTP3	MSTP2	MSTP1	MSTP0
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—Standby (STBY): Specifies a transition to standby mode.

Bit 7: STBY	Description	
0	Transition to sleep mode on execution of SLEEP instruction	(Initial value)
1	Transition to standby mode on execution of SLEEP instruction	

Bit 6—Peripheral Module Pin High Impedance Control (PHZ): Controls the state of peripheral module related pins in standby mode. When the PHZ bit is set to 1, peripheral module related pins go to the high-impedance state in standby mode.

For the relevant pins, see section 9.2.2, Peripheral Module Pin High Impedance Control.

Bit 6: PHZ	Description	
0	Peripheral module related pins are in normal state	(Initial value)
1	Peripheral module related pins go to high-impedance state	

Bit 5—Peripheral Module Pin Pull-Up Control (PPU): Controls the state of peripheral module related pins. When the PPU bit is cleared to 0, the pull-up resistor is turned on for peripheral module related pins in the input or high-impedance state.

For the relevant pins, see section 9.2.3, Peripheral Module Pin Pull-Up Control.

Bit 5: PPU	Description	
0	Peripheral module related pin pull-up resistors are on	(Initial value)
1	Peripheral module related pin pull-up resistors are off	

Bit 4—Module Stop 4 (MSTP4): Specifies stopping of the clock supply to the DMAC among the on-chip peripheral modules. The clock supply to the DMAC is stopped when the MSTP4 bit is set to 1. When DMA transfer is used, stop the transfer before setting the MSTP4 bit to 1. When DMA transfer is performed after clearing the MSTP4 bit to 0, DMAC settings must be made again.

Bit 4: MSTP4	Description	
0	DMAC operates	(Initial value)
1	DMAC clock supply is stopped	

Bit 3—Module Stop 3 (MSTP3): Specifies stopping of the clock supply to serial communication interface channel 2 (SCIF) among the on-chip peripheral modules. The clock supply to the SCIF is stopped when the MSTP3 bit is set to 1.

Bit 3: MSTP3	Description	
0	SCIF operates	(Initial value)
1	SCIF clock supply is stopped	

Bit 2—Module Stop 2 (MSTP2): Specifies stopping of the clock supply to the timer unit (TMU) among the on-chip peripheral modules. The clock supply to the TMU is stopped when the MSTP2 bit is set to 1.

Bit 2: MSTP2	Description	
0	TMU operates	(Initial value)
1	TMU clock supply is stopped	

Bit 1—Module Stop 1 (MSTP1): Specifies stopping of the clock supply to the realtime clock (RTC) among the on-chip peripheral modules. The clock supply to the RTC is stopped when the MSTP1 bit is set to 1. When the clock supply is stopped, RTC registers cannot be accessed but the counters continue to operate.

Bit 1: MSTP1	Description	
0	RTC operates	(Initial value)
1	RTC clock supply is stopped	

Bit 0—Module Stop 0 (MSTP0): Specifies stopping of the clock supply to serial communication interface channel 1 (SCI) among the on-chip peripheral modules. The clock supply to the SCI is stopped when the MSTP0 bit is set to 1.

Bit 0: MSTP0	Description	
0	SCI operates	(Initial value)
1	SCI clock supply is stopped	

9.2.2 Peripheral Module Pin High Impedance Control

When bit 6 in the standby control register (STBCR) is set to 1, peripheral module related pins go to the high-impedance state in standby mode.

- Relevant Pins

SCI related pins	MD0/SCK	MD1/TXD2
	MD7/TXD	MD8/RTS2
	CTS2	
DMA related pins	DACK0	DRAK0
	DACK1	DRAK1

- Other Information

  High impedance control is not performed when the above pins are used as port output pins.

9.2.3 Peripheral Module Pin Pull-Up Control

When bit 5 in the standby control register (STBCR) is cleared to 0, peripheral module related pins are pulled up when in the input or high-impedance state.

- Relevant Pins

SCI related pins	MD0/SCK	MD1/TXD2	MD2/RXD2
	MD7/TXD	MD8/RTS2	SCK2/$\overline{\text{MRESET}}$
	RXD	CTS2	
DMA related pins	$\overline{\text{DREQ0}}$	DACK0	DRAK0
	$\overline{\text{DREQ1}}$	DACK1	DRAK1
TMU related pin	TCLK		

9.2.4 Standby Control Register 2 (STBCR2)

Standby control register 2 (STBCR2) is an 8-bit readable/writable register that specifies the sleep mode and deep sleep mode transition conditions. It is initialized to H'00 by a power-on reset via the $\overline{\text{RESET}}$ pin or due to watchdog timer overflow.

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

Bit 7—Deep Sleep (DSLP): Specifies a transition to deep sleep mode

Bit 7: DSLP	Description
0	Transition to sleep mode or standby mode on execution of SLEEP instruction, according to setting of STBY bit in STBCR register(Initial value)
1	Transition to deep sleep mode on execution of SLEEP instruction*

Note: * When the STBY bit in the STBCR register is 0

Bits 6 to 0—Reserved: Only 0 should only be written to these bits; operation cannot be guaranteed if 1 is written. These bits are always read as 0.

9.3 Sleep Mode

9.3.1 Transition to Sleep Mode

If a SLEEP instruction is executed when the STBY bit in STBCR is cleared to 0, the chip switches from the program execution state to sleep mode. After execution of the SLEEP instruction, the CPU halts but its register contents are retained. The on-chip peripheral modules continue to operate, and the clock continues to be output from the CKIO pin.

In sleep mode, a high-level signal is output at the STATUS1 pin, and a low-level signal at the STATUS0 pin.

9.3.2 Exit from Sleep Mode

Sleep mode is exited by means of an interrupt (NMI, IRL, or on-chip peripheral module) or a reset. In sleep mode, interrupts are accepted even if the BL bit in the SR register is 1. If necessary, SPC and SSR should be saved to the stack before executing the SLEEP instruction.

Exit by Interrupt: When an NMI, IRL, or on-chip peripheral module interrupt is generated, sleep mode is exited and interrupt exception handling is executed. The code corresponding to the interrupt source is set in the INTEVT register.

Exit by Reset: Sleep mode is exited by means of a power-on or manual reset via the $\overline{\text{RESET}}$ pin, or a power-on or manual reset executed when the watchdog timer overflows.

9.4 Deep Sleep Mode

9.4.1 Transition to Deep Sleep Mode

If a SLEEP instruction is executed when the STBY bit in STBCR is cleared to 0 and the DSLP bit in STBCR2 is set to 1, the chip switches from the program execution state to deep sleep mode. After execution of the SLEEP instruction, the CPU halts but its register contents are retained. Except for the DMAC, on-chip peripheral modules continue to operate, and the clock continues to be output from the CKIO pin.

In deep sleep mode, a high-level signal is output at the STATUS1 pin, and a low-level signal at the STATUS0 pin.

9.4.2 Exit from Deep Sleep Mode

As with sleep mode, deep sleep mode is exited by means of an interrupt (NMI, IRL, or on-chip peripheral module) or a reset.

9.5 Standby Mode

9.5.1 Transition to Standby Mode

If a SLEEP instruction is executed when the STBY bit in STBCR is set to 1, the chip switches from the program execution state to standby mode. In standby mode, the on-chip peripheral modules halt as well as the CPU. Clock output from the CKIO pin is also stopped.

The CPU and cache register contents are retained. Some on-chip peripheral module registers are initialized. The state of the peripheral module registers in standby mode is shown in table 9.4.

Table 9.4 State of Registers in Standby Mode

Module	Initialized Registers	Registers That Retain Their Contents
Interrupt controller	—	All registers
User break controller	—	All registers
Bus state controller	—	All registers
On-chip oscillation circuits	—	All registers
Timer unit	TSTR register*	All registers except TSTR
Realtime clock	—	All registers
Direct memory access controller	—	All registers
Serial communication interface	See Appendix A, Address List	See Appendix A, Address List

Note: * Not initialized when the realtime clock (RTC) is in use (see section 12, Timer Unit (TMU)).

Note: DMA transfer should be terminated before making a transition to standby mode. Transfer results are not guaranteed if standby mode is entered during transfer.

The procedure for a transition to standby mode is shown below.

1. Clear the TME bit in the WDT timer control register (WTCSR) to 0, and stop the WDT.
   Set the initial value for the up-count in the WDT timer counter (WTCNT), and set the clock to be used for the up-count in bits CKS2–CKS0 in the WTCSR register.
2. Set the STBY bit in the STBCR register to 1, then execute a SLEEP instruction.
3. When standby mode is entered and the chip's internal clock stops, a low-level signal is output at the STATUS1 pin, and a high-level signal at the STATUS0 pin.

9.5.2 Exit from Standby Mode

Standby mode is exited by means of an interrupt (NMI, IRL, or on-chip peripheral module) or a reset via the $\overline{\text{RESET}}$ pin.

Exit by Interrupt: A hot start can be performed by means of the on-chip WDT. When an NMI, IRL[*1], or on-chip peripheral module (except interval timer)[*2] interrupt is detected, the WDT starts counting. After the count overflows, clocks are supplied to the entire chip, standby mode is exited, and the STATUS1 and STATUS0 pins both go low. Interrupt exception handling is then executed, and the code corresponding to the interrupt source is set in the INTEVT register. In standby mode, interrupts are accepted even if the BL bit in the SR register is 1, and so, if necessary, SPC and SSR should be saved to the stack before executing the SLEEP instruction.

The phase of the CKIO pin clock output may be unstable immediately after an interrupt is detected, until standby mode is exited.

Notes: 1. Only when the RTC clock (32.768 kHz) is operating (see section 19.2.2, IRL Interrupts), standby mode can be exited by means of IRL3–IRL0 (when the IRL3–IRL0 level is higher than the SR register I3–I0 mask level).
2. Standby mode can be exited by means of an RTC interrupt.

Exit by Reset: Standby mode is exited by means of a reset (power-on or manual) via the $\overline{\text{RESET}}$ pin. The $\overline{\text{RESET}}$ pin should be held low until clock oscillation stabilizes. The internal clock continues to be output at the CKIO pin.

9.5.3 Clock Pause Function

In standby mode, it is possible to stop or change the frequency of the clock input from the EXTAL pin. This function is used as follows.

1. Enter standby mode following the transition procedure described above.
2. When standby mode is entered and the chip’s internal clock stops, a low-level signal is output at the STATUS1 pin, and a high-level signal at the STATUS0 pin.
3. The input clock is stopped, or its frequency changed, after the STATUS1 pin goes low and the STATUS0 pin high.
4. When the frequency is changed, input an NMI or IRL interrupt after the change. When the clock is stopped, input an NMI or IRL interrupt after applying the clock.
5. After the time set in the WDT, clock supply begins inside the chip, the STATUS1 and STATUS0 pins both go low, and operation is resumed from interrupt exception handling.

9.6 Module Standby Function

9.6.1 Transition to Module Standby Function

Setting the MSTP4–MSTP0 bits in the standby control register to 1 enables the clock supply to the corresponding on-chip peripheral modules to be halted. Use of this function allows power consumption in sleep mode to be further reduced.

In the module standby state, the on-chip peripheral module external pins retain their states prior to halting of the modules, and most registers retain their states prior to halting of the modules.

Bit		Description
MSTP4	0	DMAC operates
	1	Clock supplied to DMAC is stopped
MSTP3	0	SCIF operates
	1	Clock supplied to SCIF is stopped
MSTP2	0	TMU operates
	1	Clock supplied to TMU is stopped, and register is initialized*[1]
MSTP1	0	RTC operates
	1	Clock supplied to RTC is stopped*[2]
MSTP0	0	SCI operates
	1	Clock supplied to SCI is stopped

Notes: 1. The register initialized is the same as in standby mode, but initialization is not performed if the RTC clock is not in use (see section 12, Timer Unit (TMU)).

2. The counter operates when the START bit in RCR2 is 1 (see section 11, Realtime Clock (RTC)).

9.6.2 Exit from Module Standby Function

The module standby function is exited by clearing the MSTP4–MSTP0 bits to 0, or by a power-on reset via the $\overline{\text{RESET}}$ pin or a power-on reset caused by watchdog timer overflow.

9.7 STATUS Pin Change Timing

The STATUS1 and STATUS0 pin change timing is shown below.

The meaning of the STATUS pin settings 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

9.7.1 In Reset

Power-On Reset

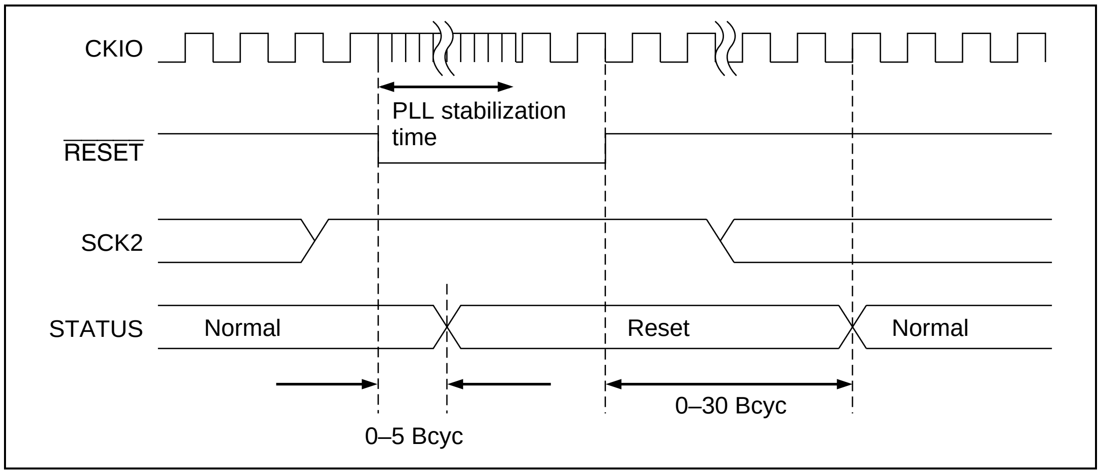
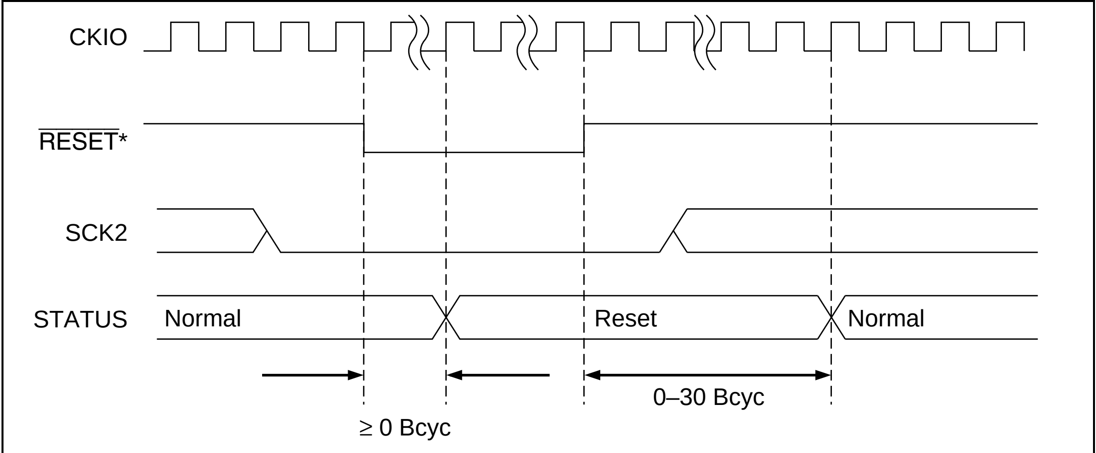


Figure 9.1 STATUS Output in Power-On Reset

Manual Reset



Note: * In a manual reset, STATUS = HH (reset) is set and an internal reset started after waiting until the end of the currently executing bus cycle.

Figure 9.2 STATUS Output in Manual Reset

9.7.2 In Exit from Standby Mode

Standby → Interrupt

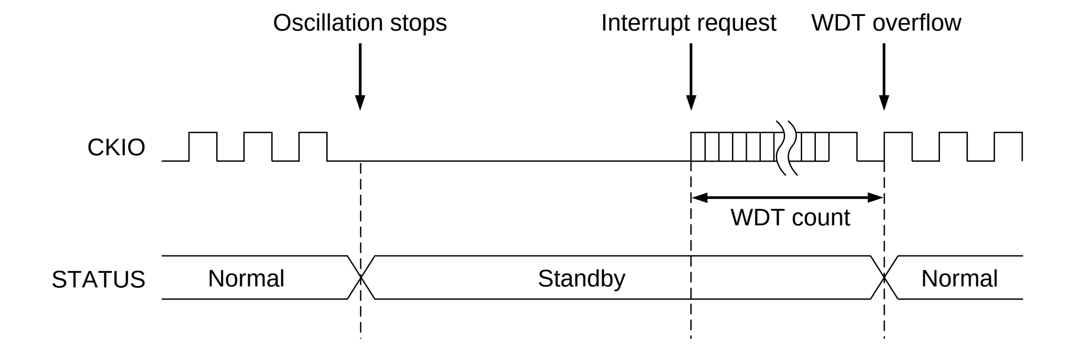
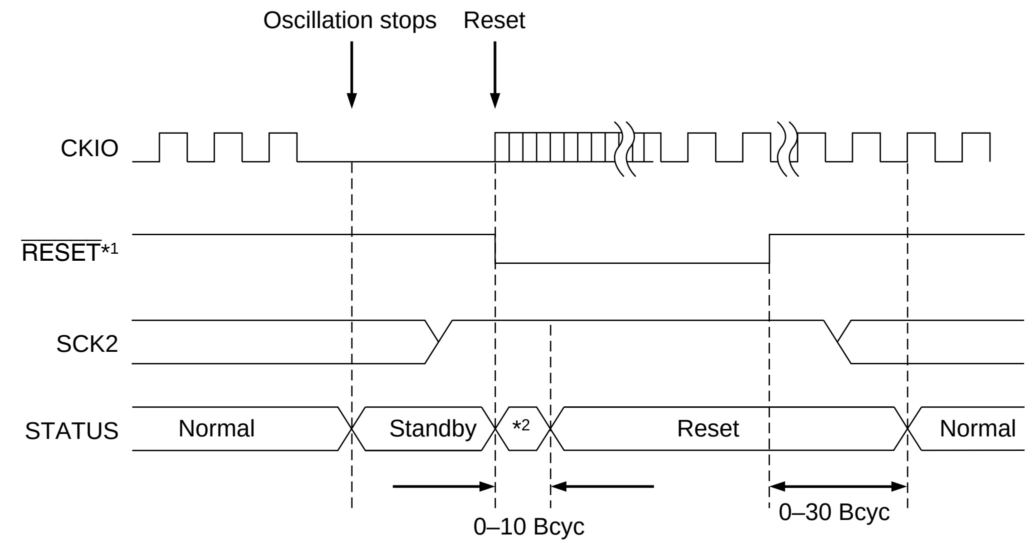


Figure 9.3 STATUS Output in Standby → Interrupt Sequence

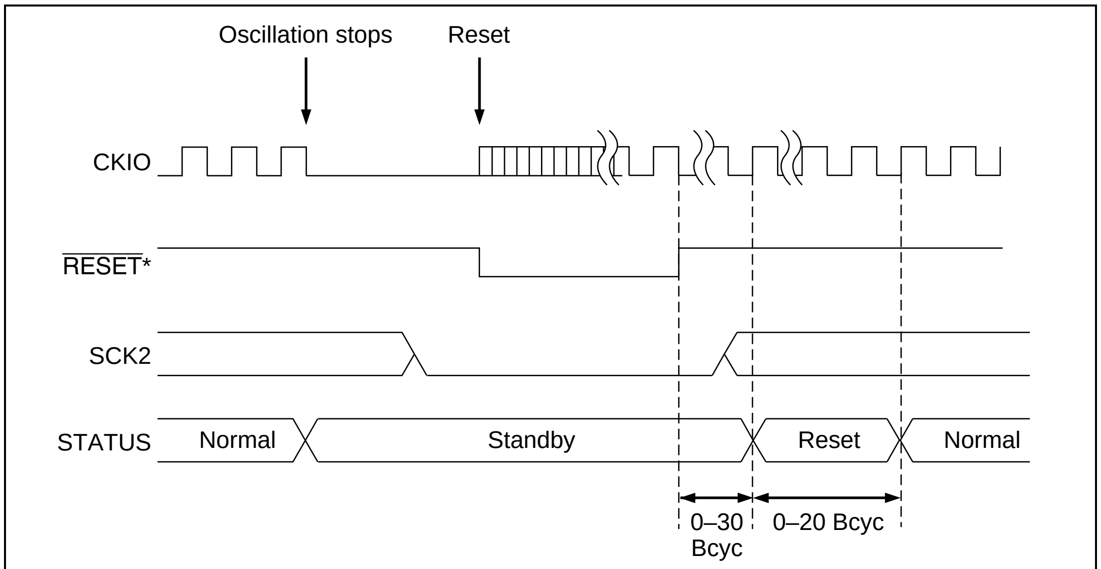
Standby → Power-On Reset



Notes: 1. When standby mode is exited by means of a power-on reset, a WDT count is not performed. Hold $\overline{\text{RESET}}$ low for the PLL oscillation stabilization time.
2. Undefined

Figure 9.4 STATUS Output in Standby → Power-On Reset Sequence

Standby → Manual Reset



Note: * When standby mode is exited by means of a manual reset, a WDT count is not performed. Hold RESET low for the PLL oscillation stabilization time.

Figure 9.5 STATUS Output in Standby → Manual Reset Sequence

9.7.3 In Exit from Sleep Mode

Sleep → Interrupt

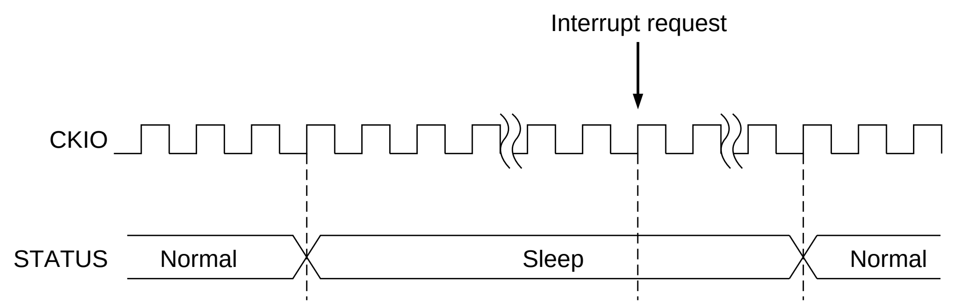
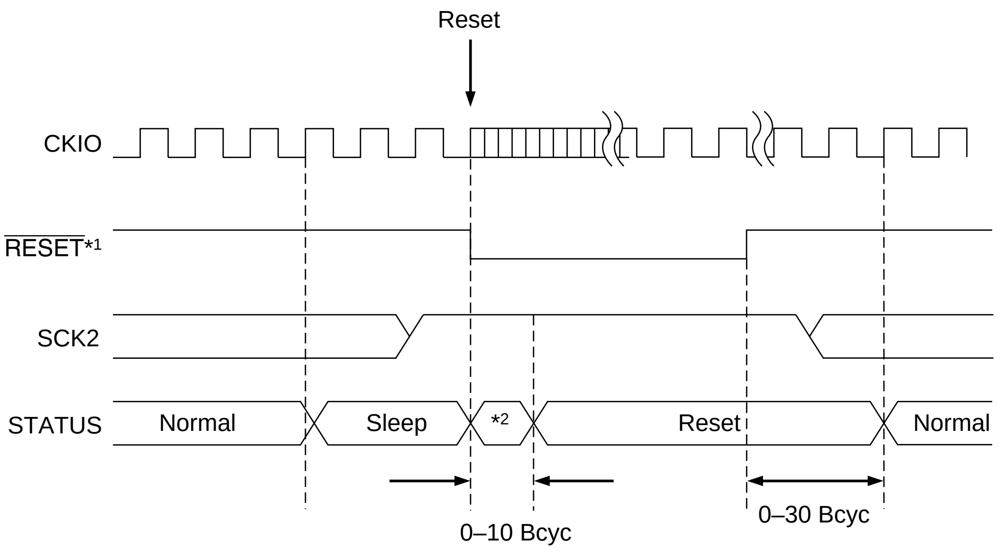


Figure 9.6 STATUS Output in Sleep → Interrupt Sequence

Sleep → Power-On Reset



Notes: 1. When sleep mode is exited by means of a power-on reset, hold RESET low for the oscillation stabilization time.
2. Undefined

Figure 9.7 STATUS Output in Sleep → Power-On Reset Sequence

Sleep → Manual Reset

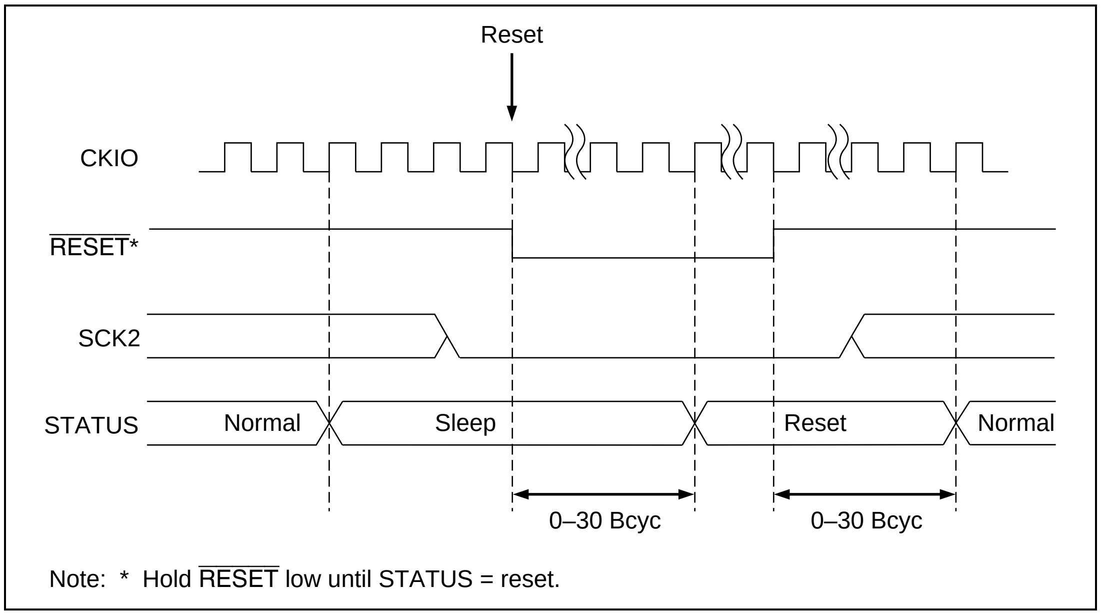

Reset
CKIO
$\overline{\text{RESET}}$*
SCK2
STATUS
Normal
Sleep
Reset
Normal
0–30 Bcyc
0–30 Bcyc
Note: * Hold $\overline{\text{RESET}}$ low until STATUS = reset.

Figure 9.8 STATUS Output in Sleep → Manual Reset Sequence

9.7.4 In Exit from Deep Sleep Mode

Deep Sleep → Interrupt

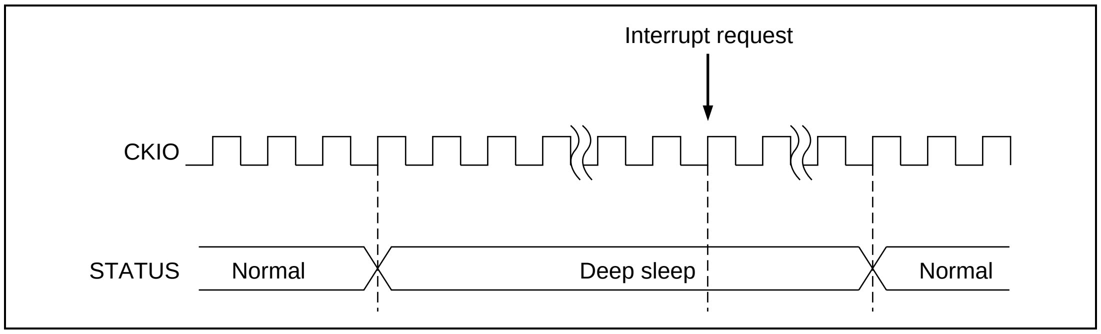
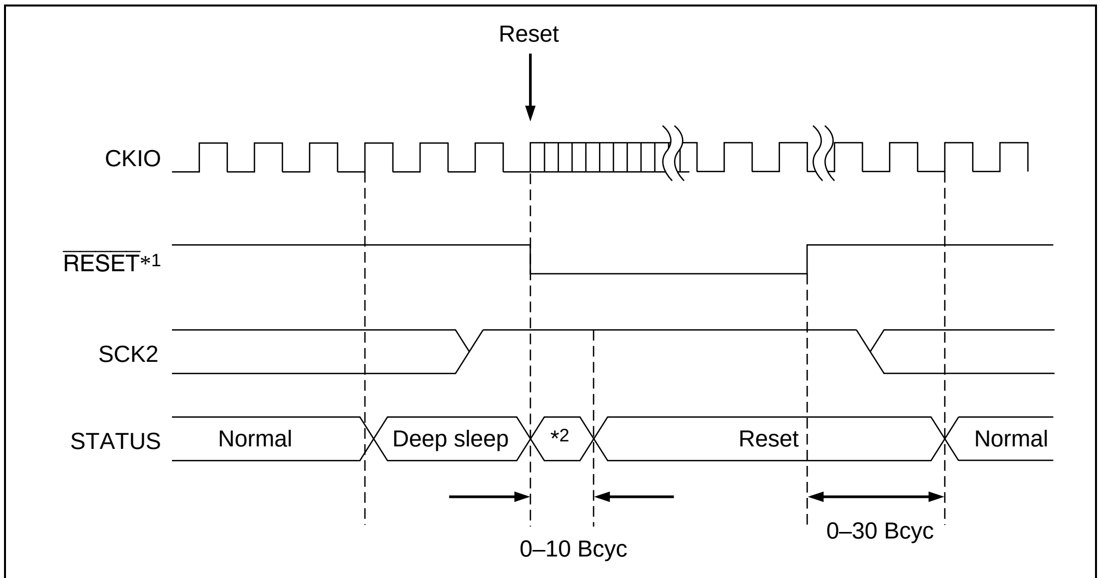


Figure 9.9 STATUS Output in Deep Sleep → Interrupt Sequence

Deep Sleep → Power-On Reset



Notes: 1. When deep sleep mode is exited by means of a power-on reset, hold $\overline{\text{RESET}}$ low for the oscillation stabilization time.
2. Undefined

Figure 9.10 STATUS Output in Deep Sleep → Power-On Reset Sequence

Deep Sleep → Manual Reset

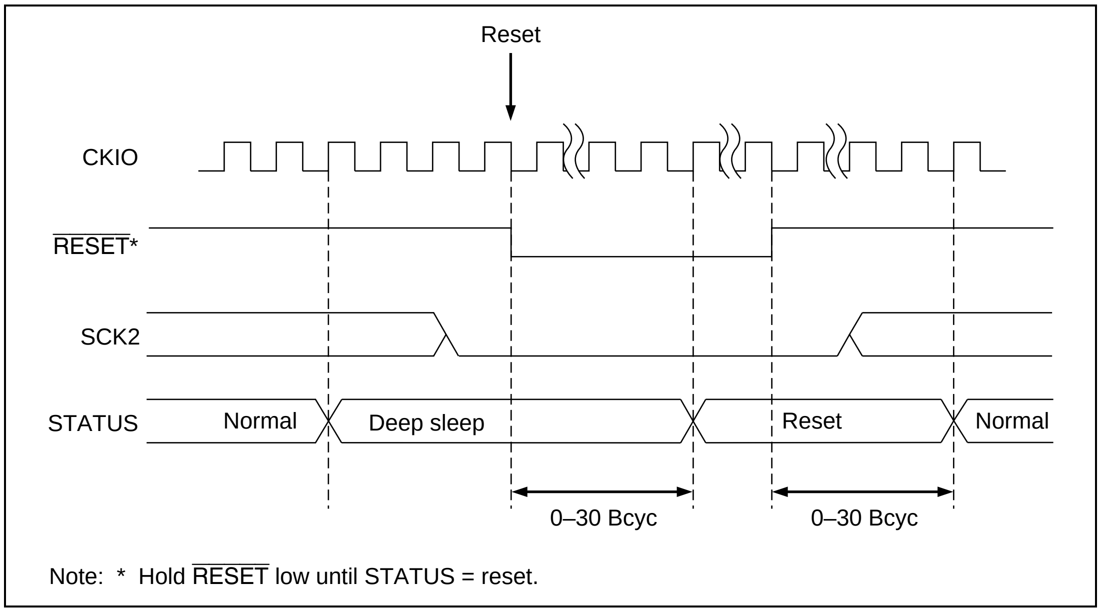


Figure 9.11 STATUS Output in Deep Sleep → Manual Reset Sequence

Section 10   Clock Oscillation Circuits

10.1 Overview

The on-chip oscillation circuits comprise a clock pulse generator (CPG) and a watchdog timer (WDT).

The CPG generates the clocks supplied inside the processor and performs power-down mode control.

The WDT is a single-channel timer used to count the clock stabilization time when exiting standby mode or a temporary standby state when the frequency is changed. It can be used as a normal watchdog timer or an interval timer.

10.1.1 Features

The CPG has the following features:

- Three clocks

  The CPG can generate independently the CPU clock (Iϕ) used by the CPU, FPU, caches, and TLB, the peripheral module clock (Pϕ) used by the peripheral modules, and the bus clock (CKIO) used by the external bus interface.
- Six clock modes

  Any of six clock operating modes can be selected, with different combinations of CPU clock, bus clock, and peripheral module clock division ratios after a power-on reset.
- Frequency change function

  PLL (phase-locked loop) circuits and a frequency divider in the CPG enable the CPU clock, bus clock, and peripheral module clock frequencies to be changed independently. Frequency changes are performed by software in accordance with the settings in the frequency control register (FRQCR).
- PLL on/off control

  Power consumption can be reduced by stopping the PLL circuits during low-frequency operation.
- Power-down mode control

  It is possible to stop the clock in sleep mode and standby mode, and to stop specific modules with the module standby function.

The WDT has the following features

- Can be used to secure clock stabilization time

  Used when exiting standby mode or a temporary standby state when the clock frequency is changed.
- Can be switched between watchdog timer mode and interval timer mode
- Internal reset generation in watchdog timer mode

  An internal reset is executed on counter overflow.

  Power-on reset or manual reset can be selected.
- Interrupt generation in interval timer mode

  An interval timer interrupt is generated on counter overflow.
- Selection of eight counter input clocks

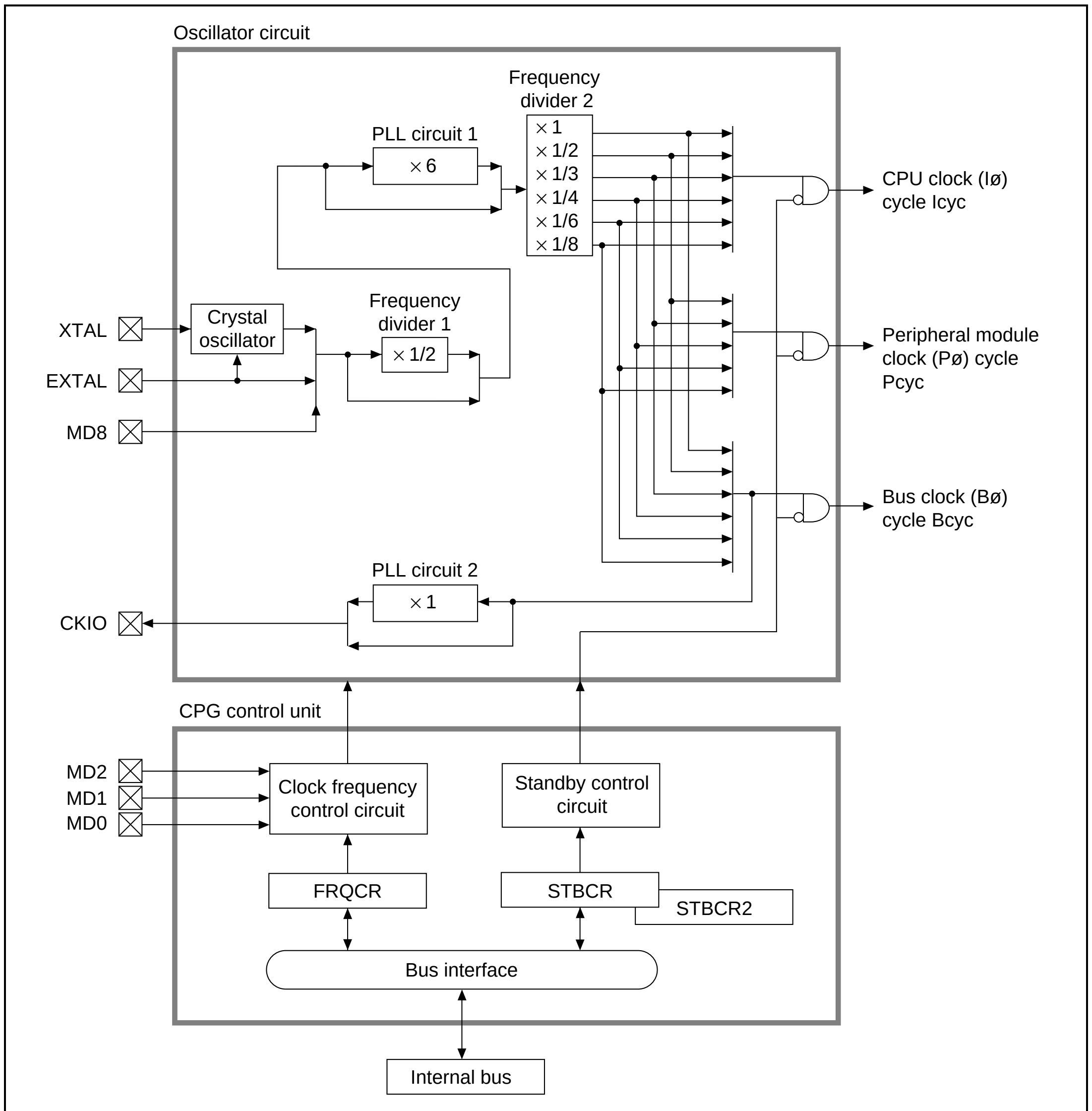
  Any of eight clocks can be selected, scaled from the ×1 clock of frequency divider 2 shown in figure 10.1.

The CPG is described in sections 10.2 to 10.6, and the WDT in sections 10.7 to 10.9.

10.2 Overview of CPG

10.2.1 Block Diagram of CPG

Figure 10.1 shows a block diagram of the CPG.



FRQCR: Frequency control register
STBCR: Standby control register
STBCR2: Standby control register 2

Figure 10.1 Block Diagram of CPG

The function of each of the CPG blocks is described below.

PLL Circuit 1: PLL circuit 1 has a function for multiplying the clock frequency from the EXTAL pin or crystal oscillator by 6. Starting and stopping is controlled by a frequency control register setting. Control is performed so that the internal clock rising edge phase matches the input clock rising edge phase.

PLL Circuit 2: PLL circuit 2 coordinates the phases of the bus clock and the CKIO pin output clock. Starting and stopping is controlled by a frequency control register setting.

Crystal Oscillator: This is the oscillator circuit used when a crystal resonator is connected to the XTAL and EXTAL pins. Use of the crystal oscillator can be selected with the MD8 pin.

Frequency Divider 1: Frequency divider 1 has a function for adjusting the clock waveform duty to 50% by halving the input clock frequency when clock input from the EXTAL pin is supplied internally without using PLL circuit 1.

Frequency Divider 2: Frequency divider 2 generates the CPU clock (Iϕ), bus clock (Bϕ), and peripheral module clock (Pϕ). The division ratio is set in the frequency control register.

Clock Frequency Control Circuit: The clock frequency control circuit controls the clock frequency by means of the MD pins and frequency control register.

Standby Control Circuit: The standby control circuit controls the state of the on-chip oscillation circuits and other modules when the clock is switched and in sleep and standby modes.

Frequency Control Register (FRQCR): The frequency control register contains control bits for clock output from the CKIO pin, PLL circuit 1 and 2 on/off control, and the CPU clock, bus clock, and peripheral module clock frequency division ratios.

Standby Control Register (STBCR): The standby control register contains power save mode control bits. For further information on the standby control register, see section 9, Power-Down Modes.

Standby Control Register 2 (STBCR2): Standby control register 2 contains a power save mode control bit. For further information on standby control register 2, see section 9, Power-Down Modes.

10.2.2 CPG Pin Configuration

Table 10.1 shows the CPG pins and their functions.

Table 10.1 CPG Pins

Pin Name	Abbreviation	I/O	Function
Mode control pins	MD0	Input	Set clock operating mode
	MD1		
	MD2		
Crystal I/O pins (clock input pins)	XTAL	Output	Connects crystal resonator
	EXTAL	Input	Connects crystal resonator, or used as external clock input pin
	MD8	Input	Selects use/non-use of crystal resonator When MD8 = 0, external clock is input from EXTAL When MD8 = 1, crystal resonator is connected directly to EXTAL and XTAL
Clock output pin	CKIO	Output	Used as external clock output pin Level can also be fixed
CKIO enable pin	CKE	Output	0 when CKIO output clock is unstable

10.2.3 CPG Register Configuration

Table 10.2 shows the CPG register configuration.

Table 10.2 CPG Register

Name	Abbreviation	R/W	Initial Value	P4 Address	Area 7 Address	Access Size
Frequency control register	FRQCR	R/W	Undefined*	H'FFC00000	H'1FC00000	16

Note: * Depends on the clock operating mode set by pins MD2–MD0.

10.3 Clock Operating Modes

Table 10.3 shows the clock operating modes corresponding to various combinations of mode control pin (MD2–MD0) settings.

Table 10.4 shows FRQCR settings and internal clock frequencies.

Table 10.3 Clock Operating Modes

Clock Operating Mode	External Pin Combination			1/2 Frequency Divider	PLL1	PLL2	Frequency (vs. Input Clock)			Input Clock Frequency Range (MHz)
	MD2	MD1	MD0				CPU Clock	Bus Clock	Peripheral Module Clock	
0	0	0	0	Off	On	On	6	3/2	3/2	17–33
1			1	Off	On	On	6	1	1	25–33
2		1	0	On	On	On	3	1	1/2	25–66
3			1	Off	On	On	6	2	1	13–33
4	1	0	0	On	On	On	3	3/2	3/4	17–66
5			1	Off	On	On	6	3	3/2	9–33

Notes: 1. The maximum frequencies of the CPU clock, bus clock, and peripheral module clock, respectively, are 200 MHz, 100 MHz, and 50 MHz.

2. The frequency range of PLL2 is 25 MHz to 100 MHz.

Table 10.4 FRQCR Settings and Internal Clock Frequencies

FRQCR (Lower 9 Bits)	Frequency Division Ratio			Clock Ratio (I:B:P)*			
	CPU Clock	Bus Clock	Peripheral Module Clock	1/2 Frequency Divider Off PLL1 Off	1/2 Frequency Divider Off PLL1 On	1/2 Frequency Divider On PLL1 Off	1/2 Frequency Divider On PLL1 On
H'008	1	1/2	1/2	1:1/2:1/2	6:3:3	1/2:1/4:1/4	3:3/2:3/2
H'00A			1/4	1:1/2:1/4	6:3:3/2	1/2:1/4:1/8	3:3/2:3/4
H'00C			1/8	1:1/2:1/8	6:3:3/4	1/2:1/4:1/16	3:3/2:3/8
H'011		1/3	1/3	1:1/3:1/3	6:2:2	1/2:1/6:1/6	3:1:1
H'013			1/6	1:1/3:1/6	6:2:1	1/2:1/6:1/12	3:1:1/2
H'01A		1/4	1/4	1:1/4:1/4	6:3/2:3/2	1/2:1/8:1/8	3:3/4:3/4
H'01C			1/8	1:1/4:1/8	6:3/2:3/4	1/2:1/8:1/16	3:3/4:3/8
H'023		1/6	1/6	1:1/6:1/6	6:1:1	1/2:1/12:1/12	3:1/2:1/2
H'02C		1/8	1/8	1:1/8:1/8	6:3/4:3/4	1/2:1/16:1/16	3:3/8:3/8
H'05A	1/2	1/4	1/4	1/2:1/4:1/4	3:3/2:3/2	1/4:1/8:1/8	3/2:3/4:3/4
H'05C			1/8	1/2:1/4:1/8	3:3/2:3/4	1/4:1/8:1/16	3/2:3/4:3/8
H'063		1/6	1/6	1/2:1/6:1/6	3:1:1	1/4:1/12:1/12	3/2:1/2:1/2
H'06C		1/8	1/8	1/2:1/8:1/8	3:3/4:3/4	1/4:1/16:1/16	3/2:3/8:3/8
H'0A3	1/3	1/6	1/6	1/3:1/6:1/6	2:1:1	1/6:1/12:1/12	1:1/2:1/2
H'0EC	1/4	1/8	1/8	1/4:1/8:1/8	3/2:3/4:3/4	1/8:1/16:1/16	3/4:3/8:3/8

Note: * Taking input clock value as 1.
Do not set values other than those shown in the table.

10.4 CPG Register Description

10.4.1 Frequency Control Register (FRQCR)

The frequency control register (FRQCR) is a 16-bit readable/writable register that specifies use/non-use of clock output from the CKIO pin, PLL circuit 1 and 2 on/off control, and the CPU clock, bus clock, and peripheral module clock frequency division ratios. Only word access can be used on FRQCR.

FRQCR is initialized only by a power-on reset via the $\overline{\text{RESET}}$ pin. The initial value of each bit is determined by the clock operating mode.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	CKOEN	PLL1EN	PLL2EN	IFC2
Initial value:	0	0	0	0	1	1	1	—
R/W:	R	R	R	R	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	IFC1	IFC0	BFC2	BFC1	BFC0	PFC2	PFC1	PFC0
Initial value:	—	—	—	—	—	—	—	—
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 to 12—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 11—Clock Output Enable (CKOEN): Specifies whether a clock is output from the CKIO pin or the CKIO pin is placed in the high-impedance state. When the CKIO pin goes to the high-impedance state, operation continues at the operating frequency before this state was entered. When the CKIO pin becomes high-impedance, it is pulled up.

Bit 11: CKOEN	Description	
0	CKIO pin goes to high-impedance state	
1	Clock is output from CKIO pin	(Initial value)

Bit 10—PLL Circuit 1 Enable (PLL1EN): Specifies whether PLL circuit 1 is on or off.

Bit 10: PLL1EN	Description	
0	PLL circuit 1 is not used	
1	PLL circuit 1 is used	(Initial value)

Bit 9—PLL Circuit 2 Enable (PLL2EN): Specifies whether PLL circuit 2 is on or off.

Bit 9: PLL2EN	Description
0	PLL circuit 2 is not used
1	PLL circuit 2 is used (Initial value)

Bits 8 to 6—CPU Clock Frequency Division Ratio (IFC): These bits specify the CPU clock frequency division ratio with respect to the input clock, 1/2 frequency divider, or PLL circuit 1 output frequency.

Bit 8: IFC2	Bit 7: IFC1	Bit 6: IFC0	Description
0	0	0	× 1
		1	× 1/2
	1	0	× 1/3
		1	× 1/4
1	0	0	× 1/6
		1	× 1/8
Other than the above			Setting prohibited (Do not set)

Bits 5 to 3—Bus Clock Frequency Division Ratio (BFC): These bits specify the bus clock frequency division ratio with respect to the input clock, 1/2 frequency divider, or PLL circuit 1 output frequency.

Bit 5: BFC2	Bit 4: BFC1	Bit 3: BFC0	Description
0	0	0	× 1
		1	× 1/2
	1	0	× 1/3
		1	× 1/4
1	0	0	× 1/6
		1	× 1/8
Other than the above			Setting prohibited (Do not set)

Bits 2 to 0—Peripheral Module Clock Frequency Division Ratio (PFC): These bits specify the peripheral module clock frequency division ratio with respect to the input clock, 1/2 frequency divider, or PLL circuit 1 output frequency.

Bit 2: PFC2	Bit 1: PFC1	Bit 0: PFC0	Description
0	0	0	× 1/2
		1	× 1/3
	1	0	× 1/4
		1	× 1/6
1	0	0	× 1/8
Other than the above			Setting prohibited (Do not set)

10.5 Changing the Frequency

There are two methods of changing the internal clock frequency: by changing stopping and starting of PLL circuit 1, and by changing the frequency division ratio of each clock. In both cases, control is performed by software by means of the frequency control register. These methods are described below.

10.5.1 Changing PLL Circuit 1 Starting/Stopping (When PLL Circuit 2 is Off)

When PLL circuit 1 is changed from the stopped to started state, a PLL stabilization time is required. The oscillation stabilization time count is performed by the on-chip WDT.

1. Set a value in WDT to provide the specified oscillation stabilization time, and stop the WDT. The following settings are necessary:

   WTCSR register TME bit = 0: WDT stopped

   WTCSR register CKS2–CKS0 bits: WDT count clock division ratio

   WTCNT counter: Initial counter value
2. Set the PLL1EN bit to 1.
3. Internal processor operation stops temporarily, and the WDT starts counting up. The internal clock stops and an unstable clock is output to the CKIO pin.
4. After the WDT count overflows, clock supply begins within the chip and the processor resumes operation. The WDT stops after overflowing.

10.5.2 Changing PLL Circuit 1 Starting/Stopping (When PLL Circuit 2 is On)

When PLL circuit 2 is on, a PLL circuit 1 and PLL circuit 2 oscillation stabilization time is required.

1. Make WDT settings as in 10.5.1.
2. Set the PLL1EN bit to 1.
3. Internal processor operation stops temporarily, PLL circuit 1 oscillates, and the WDT starts counting up. The internal clock stops and an unstable clock is output to the CKIO pin.
4. After the WDT count overflows, PLL circuit 2 starts oscillating. The WDT resumes its up-count from the value set in step 1 above. During this time, also, the internal clock is stopped and an unstable clock is output to the CKIO pin.
5. After the WDT count overflows, clock supply begins within the chip and the processor resumes operation. The WDT stops after overflowing.

10.5.3 Changing Bus Clock Division Ratio (When PLL Circuit 2 is On)

If PLL circuit 2 is on when the bus clock frequency division ratio is changed, a PLL circuit 2 oscillation stabilization time is required.

1. Make WDT settings as in 10.5.1.
2. Set the BFC2–BFC0 bits to the desired value.
3. Internal processor operation stops temporarily, and the WDT starts counting up. The internal clock stops and an unstable clock is output to the CKIO pin.
4. After the WDT count overflows, clock supply begins within the chip and the processor resumes operation. The WDT stops after overflowing.

10.5.4 Changing Bus Clock Division Ratio (When PLL Circuit 2 is Off)

If PLL circuit 2 is off when the bus clock frequency division ratio is changed, a WDT count is not performed.

1. Set the BFC2–BFC0 bits to the desired value.
2. The set clock is switched to immediately.

10.5.5 Changing CPU or Peripheral Module Clock Division Ratio

When the CPU or peripheral module clock frequency division ratio is changed, a WDT count is not performed.

1. Set the IFC2–IFC0 or PFC2–PFC0 bits to the desired value.
2. The set clock is switched to immediately.

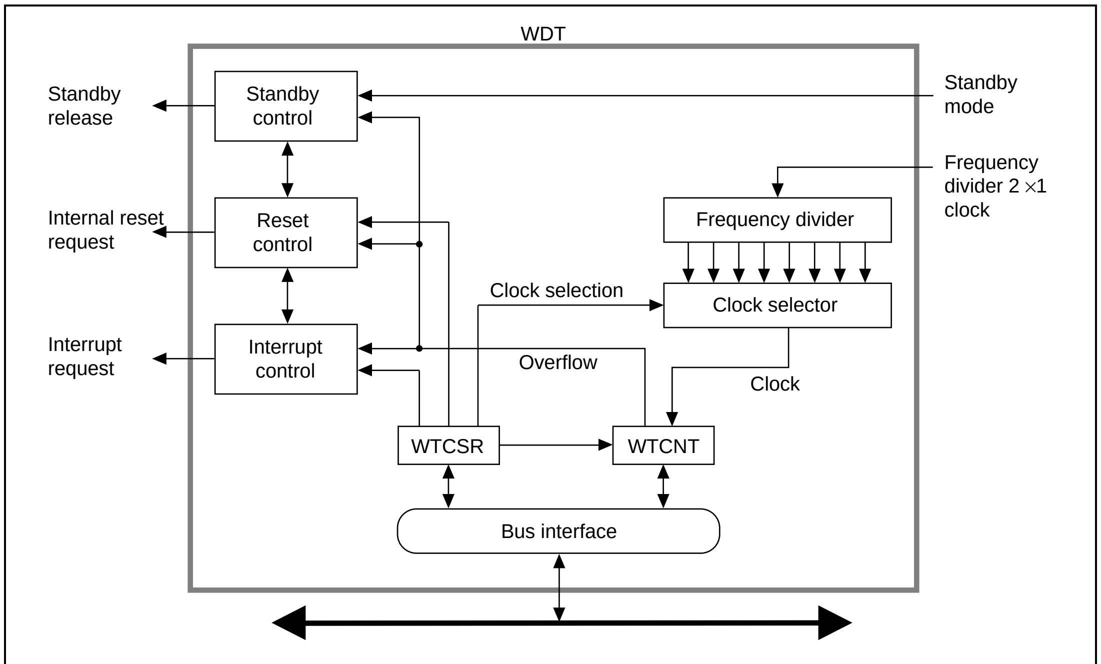
10.6 Output Clock Control

The CKIO pin can be switched between clock output and a fixed level setting by means of the CKOEN bit in the FRQCR register.

10.7 Overview of Watchdog Timer

10.7.1 Block Diagram

Figure 10.2 shows a block diagram of the WDT.



WTCSR: Watchdog timer control/status register
WTCNT: Watchdog timer counter

Figure 10.2 Block Diagram of WDT

10.7.2 Register Configuration

The WDT has the two registers summarized in table 10.5. These registers control clock selection and timer mode switching.

Table 10.5 WDT Registers

Name	Abbreviation	R/W	Initial Value	P4 Address	Area 7 Address	Access Size
Watchdog timer counter	WTCNT	R/W*	H'00	H'FFC00008	H'1FC00008	R: 8, W: 16*
Watchdog timer control/status register	WTCSR	R/W*	H'00	H'FFC0000C	H'1FC0000C	R: 8, W: 16*

Note: Use word-size access when writing. Perform the write with the upper byte set to H'5A or H'A5, respectively. Byte- and longword-size writes cannot be used.
Use byte access when reading.

10.8 WDT Register Descriptions

10.8.1 Watchdog Timer Counter (WTCNT)

The watchdog timer counter (WTCNT) is an 8-bit readable/writable counter that counts up on the selected clock. When WTCNT overflows, a reset is generated in watchdog timer mode, or an interrupt in interval timer mode. WTCNT is initialized to H'00 only by a power-on reset via the $\overline{\text{RESET}}$ pin.

To write to the WTCNT counter, use a word-size access with the upper byte set to H'5A. To read WTCNT, use a byte-size access.

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

10.8.2 Watchdog Timer Control/Status Register (WTCSR)

The watchdog timer control/status register (WTCSR) is an 8-bit readable/writable register containing bits for selecting the count clock and timer mode, and overflow flags.

WTCSR is initialized to H'00 only by a power-on reset via the $\overline{\text{RESET}}$ pin. It retains its value in an internal reset due to WDT overflow. When used to count the clock stabilization time when exiting standby mode, WTCSR retains its value after the counter overflows.

To write to the WTCSR register, use a word-size access with the upper byte set to H'A5. To read WTCSR, use a byte-size access.

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): Specifies starting and stopping of timer operation. Clear this bit to 0 when using the WDT in standby mode or to change a clock frequency.

Bit 7: TME	Description	
0	Up-count stopped, WTCNT value retained	(Initial value)
1	Up-count started	

Bit 6—Timer Mode Select (WT/$\overline{\text{IT}}$): Specifies whether the WDT is used as a watchdog timer or interval timer.

Bit 6: WT/$\overline{\text{IT}}$	Description	
0	Interval timer mode	(Initial value)
1	Watchdog timer mode	

Note: The up-count may not be performed correctly if WT/$\overline{\text{IT}}$ is modified while the WDT is running.

Bit 5—Reset Select (RSTS): Specifies the kind of reset to be performed when WTCNT overflows in watchdog timer mode. This setting is ignored in interval timer mode.

Bit 5: RSTS	Description	
0	Power-on reset	(Initial value)
1	Manual reset	

Bit 4—Watchdog Timer Overflow Flag (WOVF): Indicates that WTCNT has overflowed in watchdog timer mode. This flag 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 Flag (IOVF): Indicates that WTCNT has overflowed in interval timer mode. This flag 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 to 0 (CKS2–CKS0): These bits select the clock used for the WTCNT count from eight clocks obtained by dividing the frequency divider 2 input clock. The overflow periods shown in the following table are for use of a 33 MHz input clock, with frequency divider 1 off, and PLL circuit 1 on.

			Description	
Bit 2: CKS2	Bit 1: CKS1	Bit 0: CKS0	Clock Division Ratio	Overflow Period
0	0	0	1/32 (Initial value)	41 µs
		1	1/64	82 µs
	1	0	1/128	164 µs
		1	1/256	328 µs
1	0	0	1/512	656 µs
		1	1/1024	1.31 ms
	1	0	1/2048	2.62 ms
		1	1/4096	5.25 ms

Note: The up-count may not be performed correctly if bits CKS2–CKS0 are modified while the WDT is running. Always stop the WDT before modifying these bits.

10.8.3 Notes on Register Access

The watchdog timer counter (WTCNT) and watchdog timer control/status register (WTCSR) differ from other registers in being more difficult to write to. The procedure for writing to these registers is given below.

Writing to WTCNT and WTCSR: These registers must be written to with a word transfer instruction. They cannot be written to with a byte or longword transfer instruction. When writing to WTCNT, perform the transfer with the upper byte set to H'5A and the lower byte containing the write data. When writing to WTCSR, perform the transfer with the upper byte set to H'A5 and the lower byte containing the write data. This transfer procedure writes the lower byte data to WTCNT or WTCSR. The write formats are shown in figure 10.3.

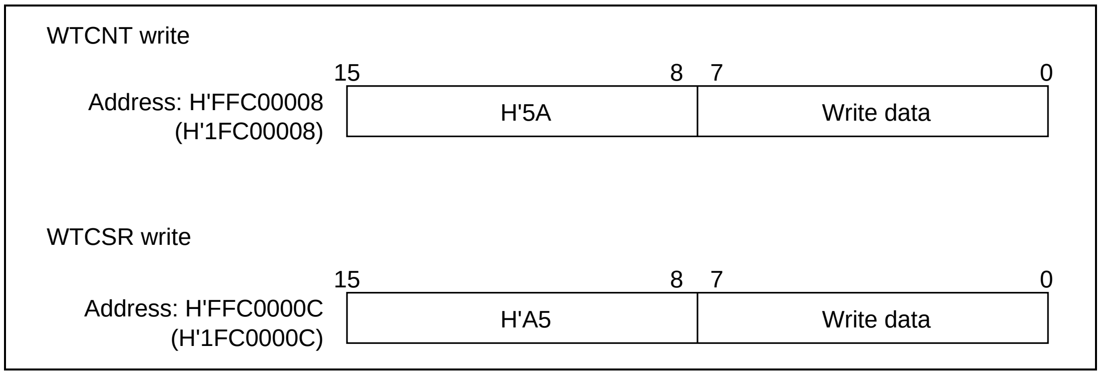


Figure 10.3 Writing to WTCNT and WTCSR

10.9 Using the WDT

10.9.1 Standby Clearing Procedure

The WDT is used when clearing standby mode by means of an NMI or other interrupt. The procedure is shown below. (As the WDT does not operate when standby mode is cleared with a reset, the $\overline{\text{RESET}}$ pin should be held low until the clock stabilizes.)

1. Be sure to clear the TME bit in the WTCSR register to 0 before making a transition to standby mode. If the TME bit is set to 1, an inadvertent reset or interval timer interrupt may be caused when the count overflows.
2. Select the count clock to be used with bits CKS2–CKS0 in the WTCSR register, and set the initial value in the WTCNT counter. Make these settings so that the time until the count overflows is at least as long as the clock oscillation stabilization time.
3. Make a transition to standby mode, and stop the clock, by executing a SLEEP instruction.
4. The WDT starts counting on detection of an NMI signal transition edge or an interrupt.
5. When the WDT count overflows, the CPG starts clock supply and the processor resumes operation. The WOVF flag in the WTCSR register is not set at this time.
6. The counter stops at a value of H'00–H'01. The value at which the counter stops depends on the clock ratio.

10.9.2 Frequency Changing Procedure

The WDT is used in a frequency change using the PLL. It is not used when the frequency is changed simply by making a frequency divider switch.

1. Be sure to clear the TME bit in the WTCSR register to 0 before making a frequency change. If the TME bit is set to 1, an inadvertent reset or interval timer interrupt may be caused when the count overflows.
2. Select the count clock to be used with bits CKS2–CKS0 in the WTCSR register, and set the initial value in the WTCNT counter. Make these settings so that the time until the count overflows is at least as long as the clock oscillation stabilization time.
3. When the frequency control register (FRQCR) is modified, the clock stops, and the standby state is entered temporarily. The WDT starts counting.
4. When the WDT count overflows, the CPG starts clock supply and the processor resumes operation. The WOVF flag in the WTCSR register is not set at this time.
5. The counter stops at a value of H'00–H'01. The value at which the counter stops depends on the clock ratio.
6. When re-setting WTCNT immediately after modifying the frequency control register (FRQCR), first read the counter and confirm that its value is as described in step 5 above.

10.9.3 Using Watchdog Timer Mode

1. Set the WT/$\overline{\text{IT}}$ bit in the WTCSR register to 1, select the type of reset with the RSTS bit, and the count clock with bits CKS2–CKS0, and set the initial value in the WTCNT counter.
2. When the TME bit in the WTCSR register is set to 1, the count starts in watchdog timer mode.
3. During operation in watchdog timer mode, write H'00 to the counter periodically so that it does not overflow.
4. When the counter overflows, the WDT sets the WOVF flag in the WTCSR register to 1, and generates a reset of the type specified by the RSTS bit. The counter then continues counting.

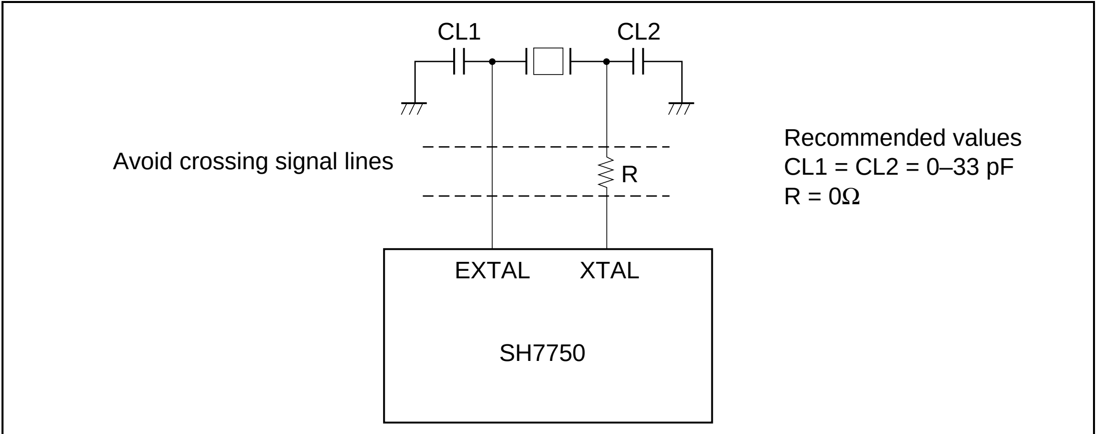
10.9.4 Using Interval Timer Mode

When the WDT is operating in interval timer mode, an interval timer interrupt is generated each time the counter overflows. This enables interrupts to be generated at fixed intervals.

1. Clear the WT/$\overline{\text{IT}}$ bit in the WTCSR register to 0, select the count clock with bits CKS2–CKS0, and set the initial value in the WTCNT counter.
2. When the TME bit in the WTCSR register is set to 1, the count starts in interval timer mode.
3. When the counter overflows, the WDT sets the IOVF flag in the WTCSR register to 1, and sends an interval timer interrupt request to INTC. The counter continues counting.

10.10 Notes on Board Design

When Using a Crystal Resonator: Place the crystal resonator and capacitors close to the EXTAL and XTAL pins. To prevent induction from interfering with correct oscillation, ensure that no other signal lines cross the signal lines for these pins.



Note: The values for CL1, CL2, and the damping resistance should be determined after consultation with the crystal resonator manufacturer.

Figure 10.4 Points for Attention when Using Crystal Resonator

When Inputting External Clock from EXTAL Pin: Make no connection to the XTAL pin.

When Using a PLL Oscillator Circuit: Separate VDD-CPG and VSS-CPG from the other VDD and VSS lines at the board power supply source, and insert resistors RCB and RB, and decoupling capacitors CPB and CB, close to the pins.

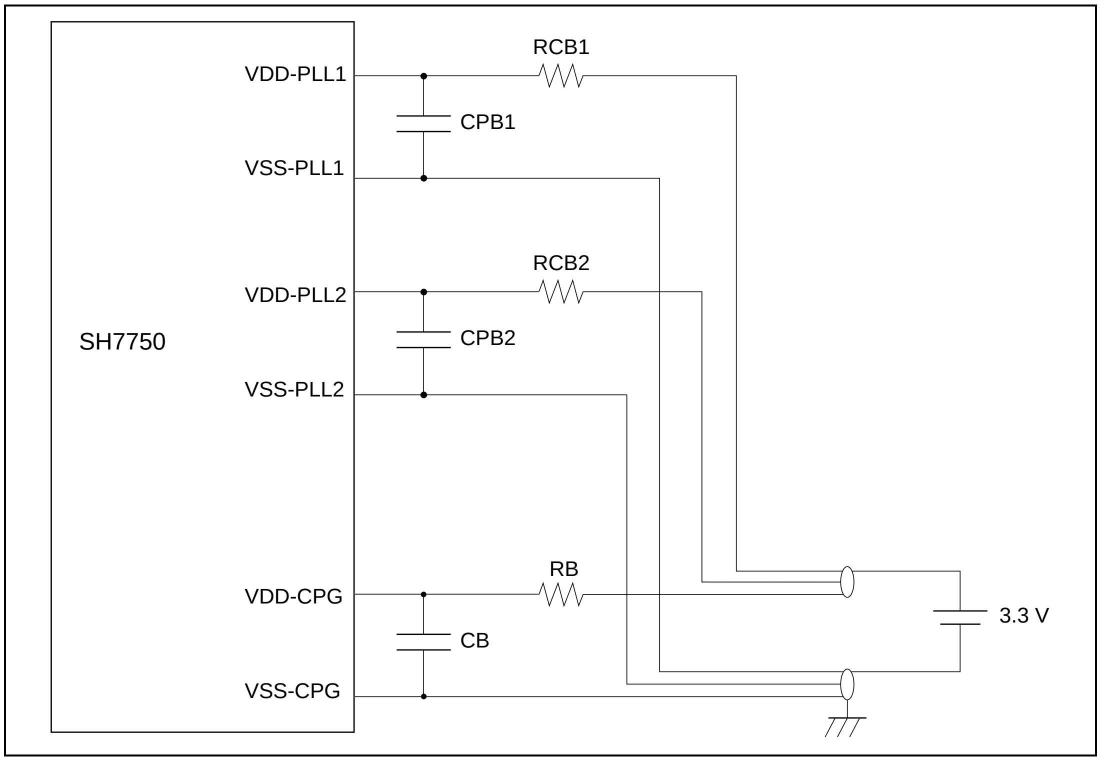


Figure 10.5 Points for Attention when Using PLL Oscillator Circuit

Section 11 Realtime Clock (RTC)

11.1 Overview

The SH7750 includes an on-chip realtime clock (RTC) and a 32.768 kHz crystal oscillator for use by the RTC.

11.1.1 Features

The RTC has the following features.

- Clock and calendar functions (BCD display)
  Counts seconds, minutes, hours, day-of-week, days, months, and years.
- 1 to 64 Hz timer (binary display)
  The 64 Hz counter register indicates a state of 64 Hz to 1 Hz within the RTC frequency divider
- Start/stop function
- 30-second adjustment function
- Alarm interrupts
  Comparison with second, minute, hour, day-of-week, day, or month can be selected as the alarm interrupt condition
- Periodic interrupts
  An interrupt period of 1/256 second, 1/64 second, 1/16 second, 1/4 second, 1/2 second, 1 second, or 2 seconds can be selected
- Carry interrupt
  Carry interrupt function indicating a second counter carry, or a 64 Hz counter carry when the 64 Hz counter is read
- Automatic leap year adjustment

11.1.2 Block Diagram

Figure 11.1 shows a block diagram of the RTC.

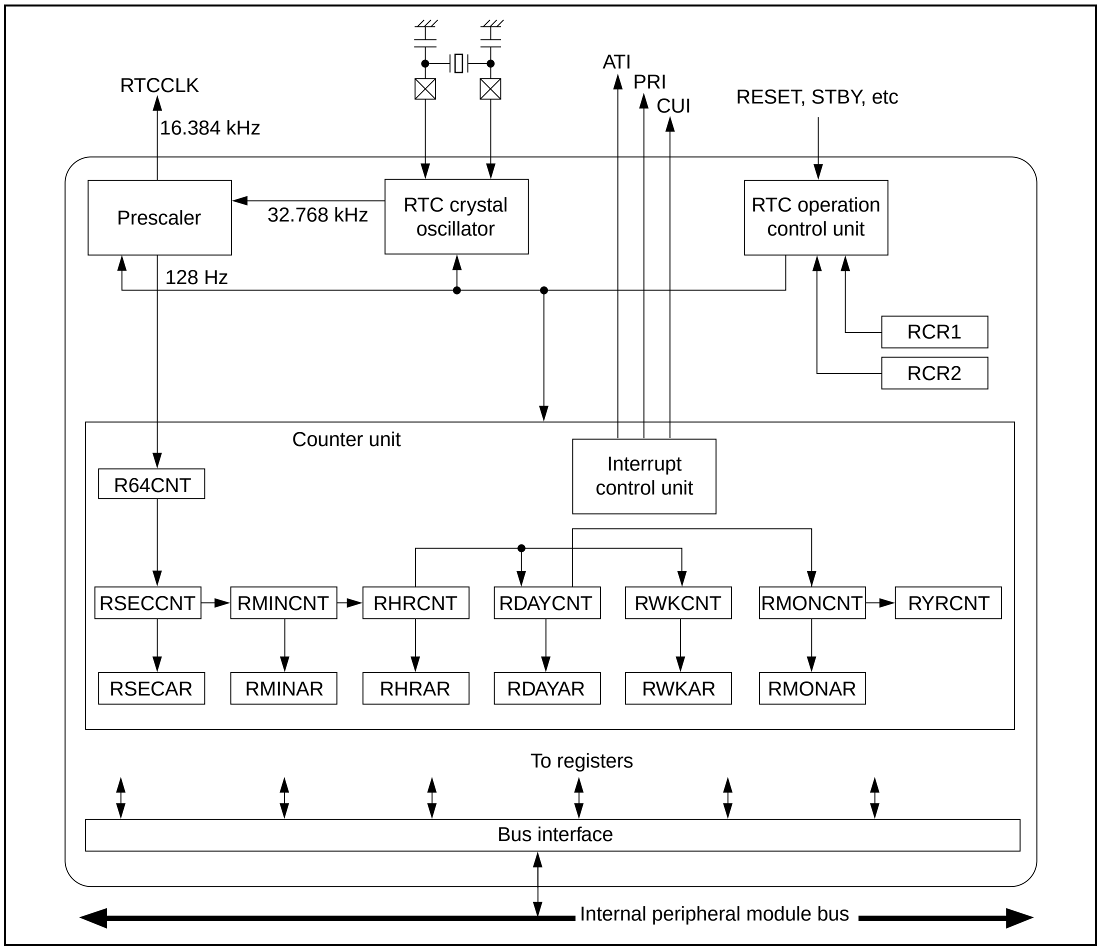


Figure 11.1 Block Diagram of RTC

11.1.3 Pin Configuration

Table 11.1 shows the RTC pins.

Table 11.1 RTC Pins

Pin Name	Abbreviation	I/O	Function
RTC oscillator crystal pin	EXTAL2	Input	Connects crystal to RTC oscillator
RTC oscillator crystal pin	XTAL2	Output	Connects crystal to RTC oscillator
Clock input/clock output	TCLK	I/O	External clock input pin/input capture control input pin/RTC output pin (shared with TMU)
Dedicated RTC power supply	V_{cc} (RTC)	—	RTC oscillator power supply pin*
Dedicated RTC GND pin	V_{ss} (RTC)	—	RTC oscillator GND pin*

Note: Power must be supplied to the RTC power supply pins even when the RTC is not used. When the RTC is used, power should be supplied to all power supply pins including these pins. In standby mode, also, power should be supplied to all power supply pins including these pins.

11.1.4 Register Configuration

Table 11.2 summarizes the RTC registers.

Table 11.2 RTC Registers

			Initialization						
Name	**Abbrevia-tion**	**R/W**	**Power-On Reset**	**Manual Reset**	**Standby Mode**	**Initial Value**	**P4 Address**	**Area 7 Address**	**Access Size**
64 Hz counter	R64CNT	R	Counts	Counts	Counts	Undefined	H'FFC80000	H'1FC80000	8
Second counter	RSECCNT	R/W	Counts	Counts	Counts	Undefined	H'FFC80004	H'1FC80004	8
Minute counter	RMINCNT	R/W	Counts	Counts	Counts	Undefined	H'FFC80008	H'1FC80008	8
Hour counter	RHRCNT	R/W	Counts	Counts	Counts	Undefined	H'FFC8000C	H'1FC8000C	8
Day-of-week counter	RWKCNT	R/W	Counts	Counts	Counts	Undefined	H'FFC80010	H'1FC80010	8
Day counter	RDAYCNT	R/W	Counts	Counts	Counts	Undefined	H'FFC80014	H'1FC80014	8

Table 11.2 RTC Registers

			Initialization						
Name	**Abbrevia-tion**	**R/W**	**Power-On Reset**	**Manual Reset**	**Standby Mode**	**Initial Value**	**P4 Address**	**Area 7 Address**	**Access Size**
Month counter	RMONCNT	R/W	Counts	Counts	Counts	Undefined	H'FFC80018	H'1FC80018	8
Year counter	RYRCNT	R/W	Counts	Counts	Counts	Undefined	H'FFC8001C	H'1FC8001C	16
Second alarm register	RSECAR	R/W	Initialized*1	Held	Held	Undefined*1	H'FFC80020	H'1FC80020	8
Minute alarm register	RMINAR	R/W	Initialized*1	Held	Held	Undefined*1	H'FFC80024	H'1FC80024	8
Hour alarm register	RHRAR	R/W	Initialized*1	Held	Held	Undefined*1	H'FFC80028	H'1FC80028	8
Day-of-week alarm register	RWKAR	R/W	Initialized*1	Held	Held	Undefined*1	H'FFC8002C	H'1FC8002C	8
Day alarm register	RDAYAR	R/W	Initialized*1	Held	Held	Undefined*1	H'FFC80030	H'1FC80030	8
Month alarm register	RMONAR	R/W	Initialized*1	Held	Held	Undefined*1	H'FFC80034	H'1FC80034	8
RTC control register 1	RCR1	R/W	Initialized	Initialized	Held	H'00*3	H'FFC80038	H'1FC80038	8
RTC control register 2	RCR2	R/W	Initialized	Initialized*2	Held	H'09*4	H'FFC8003C	H'1FC8003C	8

Notes: 1. The ENB bit in each register is initialized.
2. Bits other than the RTCEN bit and START bit are initialized.
3. The value of the CF bit and AF bit is undefined.
4. The value of the PEF bit is undefined.

11.2 Register Descriptions

11.2.1 64 Hz Counter (R64CNT)

R64CNT is an 8-bit read-only register that indicates a state of 64 Hz to 1 Hz within the RTC frequency divider.

If this register is read when a carry is generated from the 128 kHz frequency division stage, bit 7 (CF) in RTC control register 1 (RCR1) is set to 1, indicating the simultaneous occurrence of the carry and the 64 Hz counter read. In this case, the read value is not valid, and so R64CNT must be read again after first writing 0 to the CF bit in RCR1 to clear it.

When the RESET bit or ADJ bit in RTC control register 2 (RCR2) is set to 1, the RTC frequency divider is initialized and R64CNT is initialized to H'00.

R64CNT is not initialized by a power-on or manual reset, or in standby mode.

Bit 7 is always read as 0 and cannot be modified.

Bit:	7	6	5	4	3	2	1	0
	—	1 Hz	2 Hz	4 Hz	8 Hz	16 Hz	32 Hz	64 Hz
Initial value:	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R	R	R	R	R	R	R	R

11.2.2 Second Counter (RSECCNT)

RSECCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded second value in the RTC. It counts on the carry generated once per second by the 64 Hz counter.

The setting range is decimal 00 to 59. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RSECCNT is not initialized by a power-on or manual reset, or in standby mode.

Bit 7 is always read as 0. A write to this bit is invalid, but the write value should always be 0.

Bit:	7	6	5	4	3	2	1	0
	—	10-second units			1-second units			
Initial value:	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

11.2.3 Minute Counter (RMINCNT)

RMINCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded minute value in the RTC. It counts on the carry generated once per minute by the second counter.

The setting range is decimal 00 to 59. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RMINCNT is not initialized by a power-on or manual reset, or in standby mode.

Bit 7 is always read as 0. A write to this bit is invalid, but the write value should always be 0.

Bit:	7	6	5	4	3	2	1	0
	—	10-minute units			1-minute units			
Initial value:	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

11.2.4 Hour Counter (RHRCNT)

RHRCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded hour value in the RTC. It counts on the carry generated once per hour by the minute counter.

The setting range is decimal 00 to 23. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RHRCNT is not initialized by a power-on or manual reset, or in standby mode.

Bits 7 and 6 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit:	7	6	5	4	3	2	1	0
	—	—	10-hour units		1-hour units			
Initial value:	0	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

11.2.5 Day-of-Week Counter (RWKCNT)

RWKCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded day-of-week value in the RTC. It counts on the carry generated once per day by the hour counter.

The setting range is decimal 0 to 6. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RWKCNT is not initialized by a power-on or manual reset, or in standby mode.

Bits 7 to 3 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	Day of week		
Initial value:	0	0	0	0	0	Undefined	Undefined	Undefined
R/W:	R	R	R	R	R	R/W	R/W	R/W

Day-of-week code	0	1	2	3	4	5	6
Day of week	Sun	Mon	Tue	Wed	Thu	Fri	Sat

11.2.6 Day Counter (RDAYCNT)

RDAYCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded day value in the RTC. It counts on the carry generated once per day by the hour counter.

The setting range is decimal 01 to 31. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RDAYCNT is not initialized by a power-on or manual reset, or in standby mode.

The setting range for RDAYCNT depends on the month and whether the year is a leap year, so care is required when making the setting.

Bits 7 and 6 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit:	7	6	5	4	3	2	1	0
	—	—	10-day units		1-day units			
Initial value:	0	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

11.2.7 Month Counter (RMONCNT)

RMONCNT is an 8-bit readable/writable register used as a counter for setting and counting the BCD-coded month value in the RTC. It counts on the carry generated once per month by the day counter.

The setting range is decimal 01 to 12. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RMONCNT is not initialized by a power-on or manual reset, or in standby mode.

Bits 7 to 5 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	10-month unit	1-month units			
Initial value:	0	0	0	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W

11.2.8 Year Counter (RYRCNT)

RYRCNT is a 16-bit readable/writable register used as a counter for setting and counting the BCD-coded year value in the RTC. It counts on the carry generated once per year by the month counter.

The setting range is decimal 0000 to 9999. The RTC will not operate normally if any other value is set. Write processing should be performed after stopping the count with the START bit in RCR2, or by using the carry flag.

RYRCNT is not initialized by a power-on or manual reset, or in standby mode.

Bit:	15	14	13	12	11	10	9	8
	1000-year units				100-year units			
Initial value:	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
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
	10-year units				1-year units			
Initial value:	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

11.2.9 Second Alarm Register (RSECAR)

RSECAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded second value counter, RSECCNT. When the ENB bit is set to 1, the RSECAR value is compared with the RSECCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 00 to 59 + ENB bit. The RTC will not operate normally if any other value is set.

The ENB bit in RSECAR is initialized to 0 by a power-on reset. The other fields in RSECAR are not initialized by a power-on or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
	ENB	10-second units			1-second units			
Initial value:	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

11.2.10 Minute Alarm Register (RMINAR)

RMINAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded minute value counter, RMINCNT. When the ENB bit is set to 1, the RMINAR value is compared with the RMINCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 00 to 59 + ENB bit. The RTC will not operate normally if any other value is set.

The ENB bit in RMINAR is initialized by a power-on reset. The other fields in RMINAR are not initialized by a power-on or manual reset, or in standby mode.

Bit:	7	6	5	4	3	2	1	0
	ENB	10-minute units			1-minute units			
Initial value:	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

11.2.11 Hour Alarm Register (RHRAR)

RHRAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded hour value counter, RHRCNT. When the ENB bit is set to 1, the RHRAR value is compared with the RHRCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 00 to 23 + ENB bit. The RTC will not operate normally if any other value is set.

The ENB bit in RHRAR is initialized by a power-on reset. The other fields in RHRAR are not initialized by a power-on or manual reset, or in standby mode.

Bit 6 is always read as 0. A write to this bit is invalid, but the write value should always be 0.

Bit:	7	6	5	4	3	2	1	0
	ENB	—	10-hour units		1-hour units			
Initial value:	0	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

11.2.12 Day-of-Week Alarm Register (RWKAR)

RWKAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded day-of-week value counter, RWKCNT. When the ENB bit is set to 1, the RWKAR value is compared with the RWKCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 0 to 6 + ENB bit. The RTC will not operate normally if any other value is set.

The ENB bit in RWKAR is initialized by a power-on reset. The other fields in RWKAR are not initialized by a power-on or manual reset, or in standby mode.

Bits 6 to 3 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit:	7	6	5	4	3	2	1	0
	ENB	—	—	—	—	Day of week		
Initial value:	0	0	0	0	0	Undefined	Undefined	Undefined
R/W:	R/W	R	R	R	R	R/W	R/W	R/W

Day-of-week code	0	1	2	3	4	5	6
Day of week	Sun	Mon	Tue	Wed	Thu	Fri	Sat

11.2.13 Day Alarm Register (RDAYAR)

RDAYAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded day value counter, RDAYCNT. When the ENB bit is set to 1, the RDAYAR value is compared with the RDAYCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 01 to 31 + ENB bit. The RTC will not operate normally if any other value is set. The setting range for RDAYAR depends on the month and whether the year is a leap year, so care is required when making the setting.

The ENB bit in RDAYAR is initialized by a power-on reset. The other fields in RDAYAR are not initialized by a power-on or manual reset, or in standby mode.

Bit 6 is always read as 0. A write to this bit is invalid, but the write value should always be 0.

Bit:	7	6	5	4	3	2	1	0
	ENB	—	10-day units		1-day units			
Initial value:	0	0	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

11.2.14 Month Alarm Register (RMONAR)

RMONAR is an 8-bit readable/writable register used as an alarm register for the RTC's BCD-coded month value counter, RMONCNT. When the ENB bit is set to 1, the RMONAR value is compared with the RMONCNT value. Comparison between the counter and the alarm register is performed for those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1, and the RCR1 alarm flag is set when the respective values all match.

The setting range is decimal 01 to 12 + ENB bit. The RTC will not operate normally if any other value is set.

The ENB bit in RMONAR is initialized by a power-on reset. The other fields in RMONAR are not initialized by a power-on or manual reset, or in standby mode.

Bits 6 and 5 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit:	7	6	5	4	3	2	1	0
	ENB	—	—	10-month unit	1-month units			
Initial value:	0	0	0	Undefined	Undefined	Undefined	Undefined	Undefined
R/W:	R/W	R	R	R/W	R/W	R/W	R/W	R/W

11.2.15 RTC Control Register 1 (RCR1)

RCR1 is an 8-bit readable/writable register containing a carry flag and alarm flag, plus flags to enable or disable interrupts for these flags.

The CIE and AIE bits are initialized to 0 by a power-on or manual reset; the value of bits other than CIE and AIE is undefined. In standby mode RCR1 is not initialized, and retains its current value.

Bit:	7	6	5	4	3	2	1	0
	CF	—	—	CIE	AIE	—	—	AF
Initial value:	Undefined	Undefined	Undefined	0	0	Undefined	Undefined	Undefined
R/W:	R/W	R	R	R/W	R/W	R	R	R/W

Bit 7—Carry Flag (CF): This flag is set to 1 on generation of a second counter carry, or a 64 Hz counter carry when the 64 Hz counter is read. The count register value read at this time is not guaranteed, and so the count register must be read again.

Bit 7: CF	Description
0	No second counter carry, or 64 Hz counter carry when 64 Hz counter is read [Clearing condition] When 0 is written to CF
1	Second counter carry, or 64 Hz counter carry when 64 Hz counter is read [Setting conditions] • Generation of a second counter carry, or a 64 Hz counter carry when the 64 Hz counter is read • When 1 is written to CF

Bit 4—Carry Interrupt Enable Flag (CIE): Enables or disables interrupt generation when the carry flag (CF) is set to 1.

Bit 4: CIE	Description
0	Carry interrupt is not generated when CF flag is set to 1 (Initial value)
1	Carry interrupt is generated when CF flag is set to 1

Bit 3—Alarm Interrupt Enable Flag (AIE): Enables or disables interrupt generation when the alarm flag (AF) is set to 1.

Bit 3: AIE	Description	
0	Alarm interrupt is not generated when AF flag is set to 1	(Initial value)
1	Alarm interrupt is generated when AF flag is set to 1	

Bit 0—Alarm Flag (AF): Set to 1 when the alarm time set in those registers among RSECAR, RMINAR, RHRAR, RWKAR, RDAYAR, and RMONAR in which the ENB bit is set to 1 matches the respective counter values.

Bit 0: AF	Description	
0	Alarm registers and counter values do not match [Clearing condition] When 0 is written to AF	(Initial value)
1	Alarm registers and counter values match* [Setting condition] When alarm registers in which the ENB bit is set to 1 and counter values match*	

Note: * Writing 1 does not change the value.

Bits 6, 5, 2, and 1—Reserved. The initial value of these bits is undefined. A write to these bits is invalid, but the write value should always be 0.

11.2.16 RTC Control Register 2 (RCR2)

RCR2 is an 8-bit readable/writable register used for periodic interrupt control, 30-second adjustment, and frequency divider RESET and RTC count control.

RCR2 is basically initialized to H'09 by a power-on reset, except that the value of the PEF bit is undefined. In a manual reset, bits other than RTCEN and START are initialized, while the value of the PEF bit is undefined. In standby mode RCR2 is not initialized, and retains its current value.

Bit:	7	6	5	4	3	2	1	0
	PEF	PES2	PES1	PES0	RTCEN	ADJ	RESET	START
Initial value:	Undefined	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 at the interval specified by bits PES2–PES0. When this flag is set to 1, a periodic interrupt is generated.

Bit 7: PEF	Description
0	Interrupt is not generated at interval specified by bits PES2–PES0 [Clearing condition] When 0 is written to PEF
1	Interrupt is generated at interval specified by bits PES2–PES0 [Setting conditions] • Generation of interrupt at interval specified by bits PES2–PES0 • When 1 is written to PEF

Bits 6 to 4—Periodic Interrupt Enable (PES2–PES0): These bits specify the period for periodic interrupts.

Bit 6: PES2	Bit 5: PES1	Bit 4: PES0	Description
0	0	0	No periodic interrupt generation (Initial value)
		1	Periodic interrupt generated at 1/256-second intervals
	1	0	Periodic interrupt generated at 1/64-second intervals
		1	Periodic interrupt generated at 1/16-second intervals
1	0	0	Periodic interrupt generated at 1/4-second intervals
		1	Periodic interrupt generated at 1/2-second intervals
	1	0	Periodic interrupt generated at 1-second intervals
		1	Periodic interrupt generated at 2-second intervals

Bit 3—Oscillator Enable (RTCEN): Controls the operation of the RTC's crystal oscillator.

Bit 3: RTCEN	Description
0	RTC crystal oscillator is halted
1	RTC crystal oscillator is operated (Initial value)

Bit 2—30-Second Adjustment (ADJ): Used for 30-second adjustment. When 1 is written to this bit, a value up to 29 seconds is rounded down to 00 seconds, and a value of 30 seconds or more is rounded up to 1 minute. The frequency divider circuits (RTC prescaler and R64CNT) are also reset at this time. This bit always returns 0 if read.

Bit 2: ADJ	Description
0	Normal clock operation (Initial value)
1	30-second adjustment performed

Bit 1—Reset (RESET): The frequency divider circuits are initialized by writing 1 to this bit. When 1 is written to the RESET bit, the frequency divider circuits (RTC prescaler and R64CNT) are reset and the RESET bit is automatically cleared to 0 (i.e. does not need to be written with 0).

Bit 1: RESET	Description
0	Normal clock operation (Initial value)
1	Frequency divider circuits are reset

Bit 0—Start Bit (START): Stops and restarts counter (clock) operation.

Bit 0: START	Description
0	Second, minute, hour, day, day-of-week, month, and year counters are stopped*
1	Second, minute, hour, day, day-of-week, month, and year counters operate normally* (Initial value)

Note: * The 64 Hz counter continues to operate unless stopped by means of the RTCEN bit.

11.3 Operation

Examples of the use of the RTC are shown below.

11.3.1 Time Setting Procedures

Figure 11.2 shows examples of the time setting procedures.

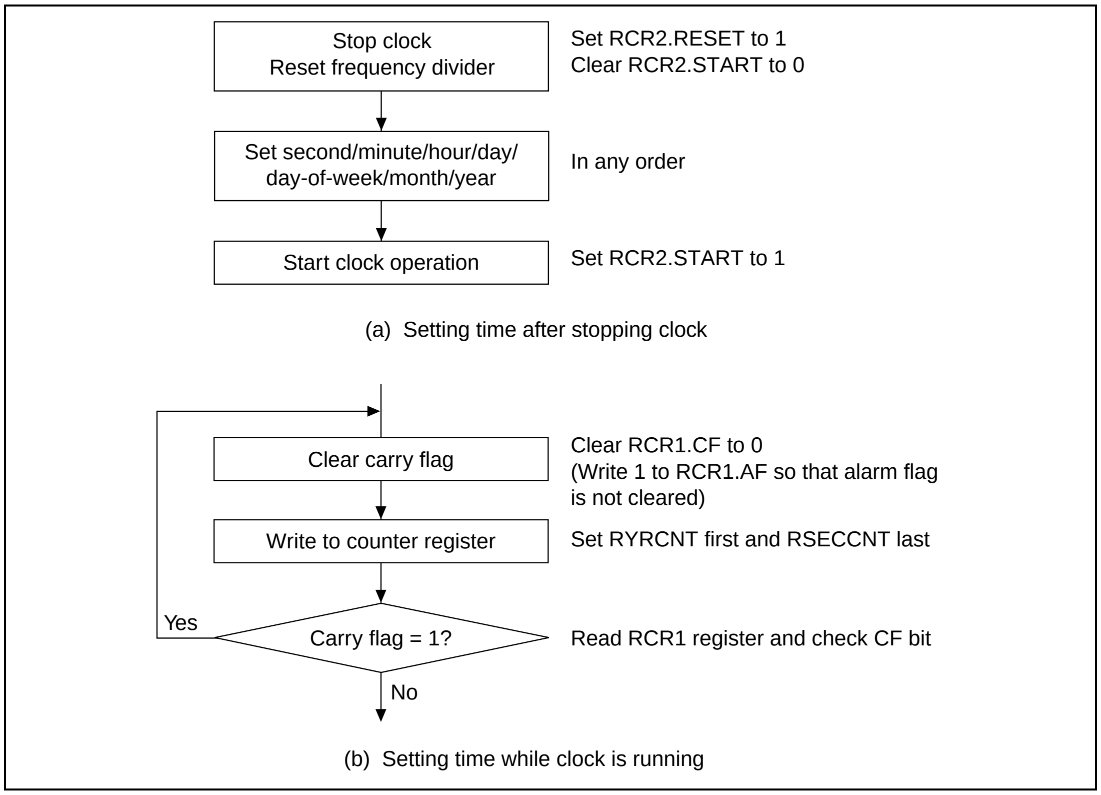

Stop clock
Reset frequency divider
Set RCR2.RESET to 1
Clear RCR2.START to 0
Set second/minute/hour/day/day-of-week/month/year
In any order
Start clock operation
Set RCR2.START to 1
(a) Setting time after stopping clock
Clear carry flag
Clear RCR1.CF to 0
(Write 1 to RCR1.AF so that alarm flag is not cleared)
Write to counter register
Set RYRCNT first and RSECCNT last
Yes
Carry flag = 1?
Read RCR1 register and check CF bit
No
(b) Setting time while clock is running

Figure 11.2 Examples of Time Setting Procedures

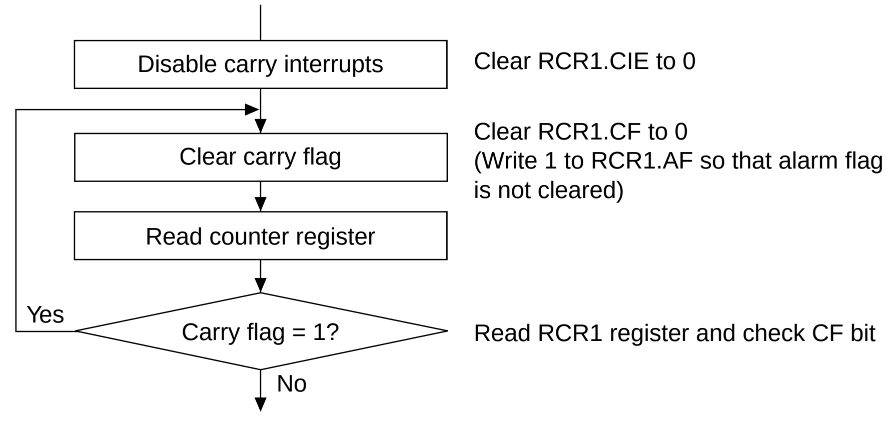
The procedure for setting the time after stopping the clock is shown in (a). The programming for this method is simple, and it is useful for setting all the counters, from second to year.

The procedure for setting the time while the clock is running is shown in (b). This method is useful for modifying only certain counter values (for example, only the second data or hour data). If a carry occurs during the write operation, the write data is automatically updated and there will be an error in the set data. The carry flag should therefore be used to check the write status. If the carry flag (RCR1.CF) is set to 1, the write must be repeated.

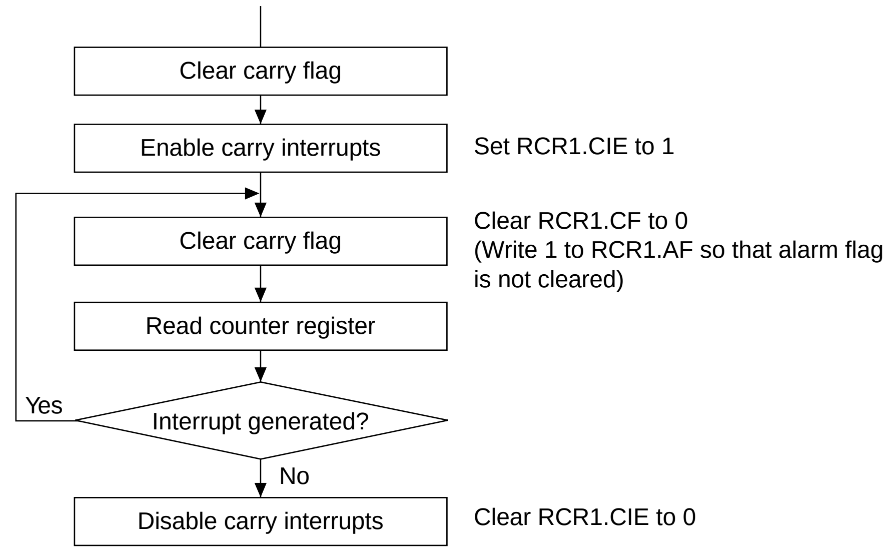
The interrupt function can also be used to determine the carry flag status.

11.3.2 Time Reading Procedures

Figure 11.3 shows examples of the time reading procedures.


Disable carry interrupts
Clear RCR1.CIE to 0
Clear carry flag
Clear RCR1.CF to 0 (Write 1 to RCR1.AF so that alarm flag is not cleared)
Read counter register
Yes
Carry flag = 1?
Read RCR1 register and check CF bit
No

(a) Reading time without using interrupts



(b) Reading time using interrupts

Figure 11.3 Examples of Time Reading Procedures

If a carry occurs while the time is being read, the correct time will not be obtained and the read must be repeated. The procedure for reading the time without using interrupts is shown in (a), and the procedure using carry interrupts in (b). The method without using interrupts is normally used to keep the program simple.

11.3.3 Alarm Function

The use of the alarm function is illustrated in figure 11.4.

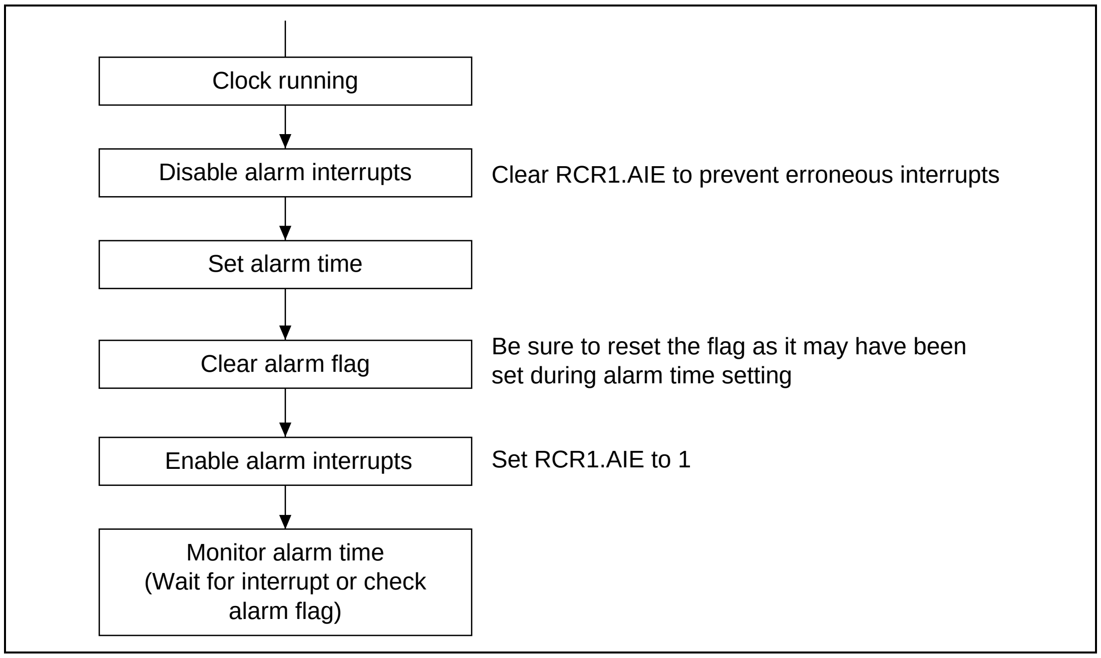

Clock running
Disable alarm interrupts
Clear RCR1.AIE to prevent erroneous interrupts
Set alarm time
Clear alarm flag
Be sure to reset the flag as it may have been set during alarm time setting
Enable alarm interrupts
Set RCR1.AIE to 1
Monitor alarm time (Wait for interrupt or check alarm flag)

Figure 11.4 Example of Use of Alarm Function

An alarm can be generated by the second, minute, hour, day-of-week, day, or month value, or a combination of these. Write 1 to the ENB bit in the alarm registers involved in the alarm setting, and set the alarm time in the lower bits. Write 0 to the ENB bit in registers not involved in the alarm setting.

When the counter and the alarm time match, RCR1.AF is set to 1. Alarm detection can be confirmed by reading this bit, but normally an interrupt is used. If 1 has been written to RCR1.AIE, an alarm interrupt is generated in the event of alarm, enabling the alarm to be detected.

The alarm flag remains set while the counter and alarm time match. If the alarm flag is cleared by writing 0 during this period, it will therefore be set again immediately afterward. This needs to be taken into consideration when writing the program.

11.4 Interrupts

There are three kinds of RTC interrupt: alarm interrupts, periodic interrupts, and carry interrupts.

An alarm interrupt request (ATI) is generated when the alarm flag (AF) in RCR1 is set to 1 while the alarm interrupt enable bit (AIE) is also set to 1.

A periodic interrupt request (PRI) is generated when the periodic interrupt enable bits (PES2–PES0) in RCR2 are set to a value other than 000 and the periodic interrupt flag (PEF) is set to 1.

A carry interrupt request (CUI) is generated when the carry flag (CF) in RCR1 is set to 1 while the carry interrupt enable bit (CIE) is also set to 1.

11.5 Usage Notes

11.5.1 Register Initialization

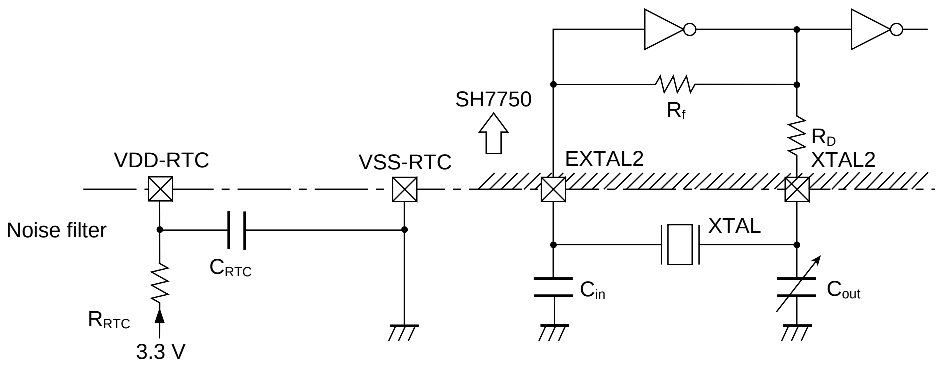
After powering on and making the RCR1 register settings, reset the frequency divider (by setting RCR2.RESET to 1) and make initial settings for all the other registers.

11.5.2 Crystal Oscillator Circuit

Crystal oscillator circuit constants (recommended values) are shown in table 11.3, and the RTC crystal oscillator circuit in figure 11.5.

Table 11.3 Crystal Oscillator Circuit Constants (Recommended Values)

f_{osc}	C_{in}	C_{out}
32.768 kHz	10–22 pF	10–22 pF



Notes:
1. Select either the C_{in} or C_{out} side for the frequency adjustment variable capacitor according to requirements such as the adjustment 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 solid-earth board.
4. The crystal oscillation stabilization 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 and XTAL2) wiring is routed as far away as possible from other power lines (except GND) and signal lines.
7. Insert a noise filter in the RTC power supply.
   The values of C_{RTC} and R_{RTC} depend on the bus and CPU frequency.

Figure 11.5 Example of Crystal Oscillator Circuit Connection

Section 12 Timer Unit (TMU)

12.1 Overview

The SH7750 includes an on-chip 32-bit timer unit (TMU) comprising three 32-bit timer channels (channels 0 to 2).

12.1.1 Features

The TMU has the following features.

- Auto-reload type 32-bit down-counter provided for each channel
- Input capture function provided in channel 2
- Selection of rising edge or falling edge as external clock input edge when external clock is selected or input capture function is used
- 32-bit timer constant register for auto-reload use, readable/writable at any time, and 32-bit down-counter provided for each channel
- Selection of seven counter input clocks for each channel

  External clock (TCLK), on-chip RTC output clock, five internal clocks (Pϕ/4, Pϕ/16, Pϕ/64, Pϕ/256, Pϕ/1024) (Pϕ is the peripheral module clock)
- Each channel can also operate in module standby mode when the on-chip RTC output clock is selected as the counter input clock; that is, timer operation continues even when the clock has been stopped for the TMU.

  Timer count operations using an external or internal clock are only possible when a clock is supplied to the timer unit.
- Synchronous read operation

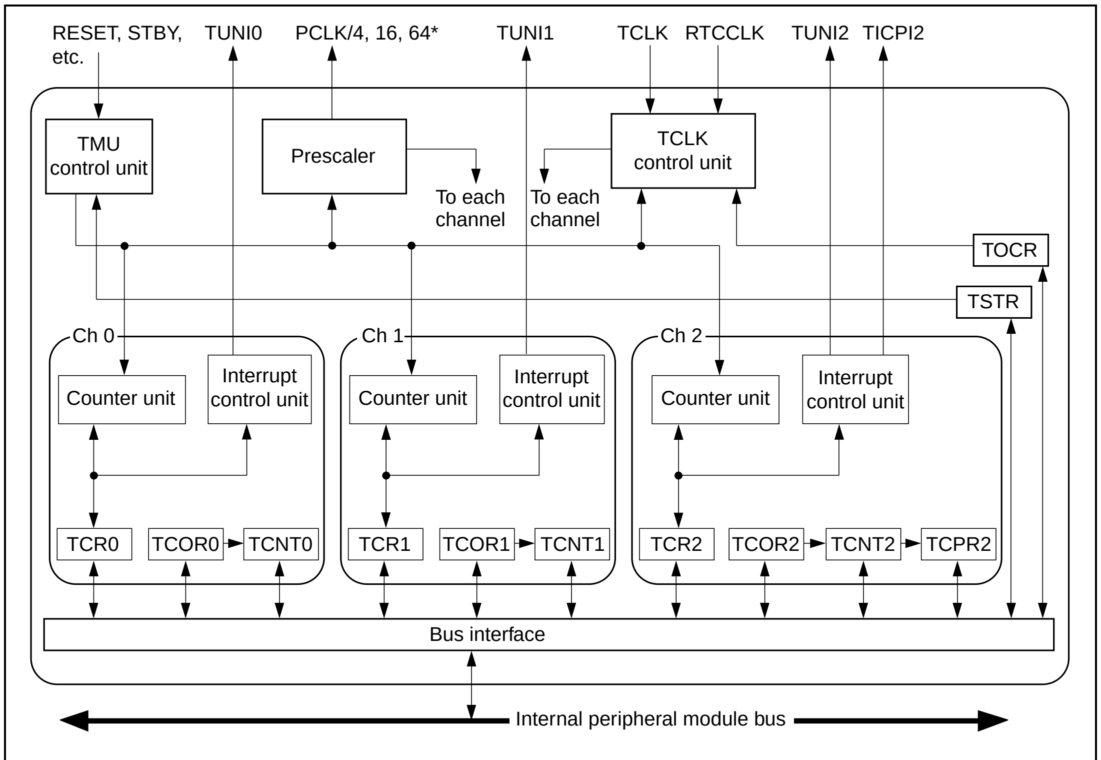
  As the timer counters (TCNT) are serially modified 32-bit registers and the internal peripheral module bus is 16 bits wide, there is a time difference when reading the upper 16 bits and lower 16 bits of TCNT. To prevent counter read value drift due to this time difference, a synchronization circuit is provided that allows simultaneous reading of all 32 bits of the TCNT data.
- Two interrupt sources

  One underflow source (channels 0 to 2) and one input capture source (channel 2)
- DMAC data transfer request capability

  On channel 2, a data transfer request is sent to the DMAC when an input capture interrupt is generated.

12.1.2 Block Diagram

Figure 12.1 shows a block diagram of the TMU.



Note: * Signals with 1/4, 1/16, and 1/64 the Pϕ frequency, supplied to the on-chip peripheral functions.

Figure 12.1 Block Diagram of TMU

12.1.3 Pin Configuration

Table 12.1 shows the TMU pins.

Table 12.1 TMU Pins

Pin Name	Abbreviation	I/O	Function
Clock input/clock output	TCLK	I/O	External clock input pin/input capture control input pin/RTC output pin (shared with RTC)

12.1.4 Register Configuration

Table 12.2 summarizes the TMU registers.

Table 12.2 TMU Registers

Channel	Name	Abbreviation	R/W	Initialization			Initial Value	P4 Address	Area 7 Address	Access Size
				Power-On Reset	Manual Reset	Standby Mode				
Common	Timer output control register	TOCR	R/W	Initialized	Initialized	Held	H'00	H'FFD80000	H'1FD80000	8
	Timer start register	TSTR	R/W	Initialized	Initialized	Initialized*[1]	H'00	H'FFD80004	H'1FD80004	8
0	Timer constant register 0	TCOR0	R/W	Initialized	Initialized	Held	H'FFFFFFFF	H'FFD80008	H'1FD80008	32
	Timer counter 0	TCNT0	R/W	Initialized	Initialized	Held*[2]	H'FFFFFFFF	H'FFD8000C	H'1FD8000C	32
	Timer control register 0	TCR0	R/W	Initialized	Initialized	Held	H'0000	H'FFD80010	H'1FD80010	16
1	Timer constant register 1	TCOR1	R/W	Initialized	Initialized	Held	H'FFFFFFFF	H'FFD80014	H'1FD80014	32
	Timer counter 1	TCNT1	R/W	Initialized	Initialized	Held*[2]	H'FFFFFFFF	H'FFD80018	H'1FD80018	32
	Timer control register 1	TCR1	R/W	Initialized	Initialized	Held	H'0000	H'FFD8001C	H'1FD8001C	16
2	Timer constant register 2	TCOR2	R/W	Initialized	Initialized	Held	H'FFFFFFFF	H'FFD80020	H'1FD80020	32
	Timer counter 2	TCNT2	R/W	Initialized	Initialized	Held*[2]	H'FFFFFFFF	H'FFD80024	H'1FD80024	32
	Timer control register 2	TCR2	R/W	Initialized	Initialized	Held	H'0000	H'FFD80028	H'1FD80028	16
	Input capture register	TCPR2	R	Held	Held	Held	Undefined	H'FFD8002C	H'1FD8002C	32

Notes: 1. Not initialized in module standby mode when the input clock is the on-chip RTC output clock.
2. Counts in module standby mode when the input clock is the on-chip RTC output clock.

12.2 Register Descriptions

12.2.1 Timer Output Control Register (TOCR)

TOCR is an 8-bit readable/writable register that specifies whether external pin TCLK is used as the external clock or input capture control input pin, or as the on-chip RTC output clock output pin.

TOCR is initialized to H'00 by a power-on or manual reset, but is not initialized in standby mode.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	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 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit 0—Timer Clock Pin Control (TCOE): Specifies whether timer clock pin TCLK is used as the external clock or input capture control input pin, or as the on-chip RTC output clock output pin.

Bit 0: TCOE	Description
0	Timer clock pin (TCLK) is used as external clock input or input capture control input pin (Initial value)
1	Timer clock pin (TCLK) is used as on-chip RTC output clock output pin

12.2.2 Timer Start Register (TSTR)

TSTR is an 8-bit readable/writable register that specifies whether the channel 0–2 timer counters (TCNT) are operated or stopped.

TSTR is initialized to H'00 by a power-on or manual reset. In module standby mode, TSTR is not initialized when the input clock selected by each channel is the on-chip RTC output clock (RTCCLK), and is initialized only when the input clock is the external clock (TCLK) or internal clock (Pϕ).

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	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 are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit 2—Counter Start 2 (STR2): Specifies whether timer counter 2 (TCNT2) is operated or stopped.

Bit 2: STR2	Description	
0	TCNT2 count operation is stopped	(Initial value)
1	TCNT2 performs count operation	

Bit 1—Counter Start 1 (STR1): Specifies whether timer counter 1 (TCNT1) is operated or stopped.

Bit 1: STR1	Description	
0	TCNT1 count operation is stopped	(Initial value)
1	TCNT1 performs count operation	

Bit 0—Counter Start 0 (STR0): Specifies whether timer counter 0 (TCNT0) is operated or stopped.

Bit 0: STR0	Description	
0	TCNT0 count operation is stopped	(Initial value)
1	TCNT0 performs count operation	

12.2.3 Timer Constant Registers (TCOR)

The TCOR registers are 32-bit readable/writable registers. There are three TCOR registers, one for each channel.

When a TCNT counter underflows while counting down, the TCOR value is set in that TCNT, which continues counting down from the set value.

The TCOR registers are initialized to H'FFFFFFFF by a power-on or manual reset, but are not initialized and retain their contents in standby mode.

Bit:	31	30	29		2	1	0
				. . . .			
Initial value:	1	1	1		1	1	1
R/W:	R/W	R/W	R/W		R/W	R/W	R/W

12.2.4 Timer Counters (TCNT)

The TCNT registers are 32-bit readable/writable registers. There are three TCNT registers, one for each channel.

Each TCNT counts down on the input clock selected by TPSC2–TPSC0 in the timer control register (TCR).

When a TCNT counter underflows while counting down, the underflow flag (UNF) is set in the corresponding timer control register (TCR). At the same time, the timer constant register (TCOR) value is set in TCNT, and the count-down operation continues from the set value.

As the TCNT registers are serially modified 32-bit registers and the internal peripheral module bus is 16 bits wide, there is a time difference when reading the upper 16 bits and lower 16 bits of TCNT. To prevent counter read value drift due to this time difference, a synchronization circuit is provided. When the upper 16 bits are read, the lower 16 bits are simultaneously stored in a buffer register. After the upper 16 bits are read, the lower 16 bits are read from the buffer register.

The TCNT registers are initialized to H'FFFFFFFF by a power-on or manual reset, but are not initialized and retain their contents in standby mode.

Bit:	31	30	29		2	1	0
				. . . .			
Initial value:	1	1	1		1	1	1
R/W:	R/W	R/W	R/W		R/W	R/W	R/W

When the input clock is the on-chip RTC output clock (RTCCLK), TCNT counts even in module standby mode (that is, when the clock for the TMU is stopped). When the input clock is the external clock (TCLK) or internal clock (Pϕ), TCNT contents are retained in standby mode.

12.2.5 Timer Control Registers (TCR)

The TCR registers are 16-bit readable/writable registers. There are three TCR registers, one for each channel.

Each TCR selects the count clock, specifies the edge when an external clock is selected, and controls interrupt generation when the flag indicating timer counter (TCNT) underflow is set to 1. TCR2 is also used for channel 2 input capture control, and control of interrupt generation in the event of input capture.

The TCR registers are initialized to H'0000 by a power-on or manual reset, but are not initialized in standby mode.

1. Channel 0 and 1 TCR bit configuration

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	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
	—	—	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

2. Channel 2 TCR bit configuration

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

Bit:	7	6	5	4	3	2	1	0
	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 9, 7, and 6 (Channels 0 and 1); Bits 15 to 10 (Channel 2)—Reserved: These bits are always read as 0. A write to these bits is invalid, but the write value should always be 0.

Bit 9—Input Capture Interrupt Flag (ICPF) (Channel 2 Only): Status flag, provided in channel 2 only, that indicates the occurrence of input capture.

Bit 9: ICPF	Description	
0	Input capture has not occurred [Clearing condition] When 0 is written to ICPF	(Initial value)
1	Input capture has occurred [Setting condition] When input capture occurs*	

Note: * Writing 1 does not change the value.

Bit 8—Underflow Flag (UNF): Status flag that indicates the occurrence of underflow.

Bit 8: UNF	Description	
0	TCNT has not underflowed [Clearing condition] When 0 is written to UNF	(Initial value)
1	TCNT has underflowed [Setting condition] When TCNT underflows*	

Note: * Writing 1 does not change the value.

Bits 7 and 6—Input Capture Control (ICPE1, ICPE0) (Channel 2 Only): These bits, provided in channel 2 only, specify whether the input capture function is used, and control enabling or disabling of interrupt generation when the function is used.

When the input capture function is used, a data transfer request is sent to the DMAC in the event of input capture.

When using the input capture function, the TCLK pin must be designated as an input pin with the TCOE bit in the TOCR register. The CKEG bits specify whether the rising edge or falling edge of the TCLK signal is used to set the TCNT2 value in the input capture register (TCPR2).

The TCNT2 value is set in TCPR2 only when the TCR2.ICPF bit is 0. When the TCR2.ICPF bit is 1, TCPR2 is not set in the event of input capture. When input capture occurs, a DMAC transfer request is generated regardless of the value of the TCR2.ICPF bit. However, a new DMAC transfer request is not generated until processing of the previous request is finished.

Bit 7: ICPE1	Bit 6: ICPE0	Description	
0	0	Input capture function is not used	(Initial value)
	1	Reserved (Do not set)	
1	0	Input capture function is used, but interrupt due to input capture (TICPI2) is not enabled Data transfer request is sent to DMAC in the event of input capture	
	1	Input capture function is used, and interrupt due to input capture (TICPI2) is enabled Data transfer request is sent to DMAC in the event of input capture	

Bit 5—Underflow Interrupt Control (UNIE): Controls enabling or disabling of interrupt generation when the UNF status flag is set to 1, indicating TCNT underflow.

Bit 5: UNIE	Description	
0	Interrupt due to underflow (TUNI) is not enabled	(Initial value)
1	Interrupt due to underflow (TUNI) is enabled	

Bits 4 and 3—Clock Edge 1 and 0 (CKEG1, CKEG0): These bits select the external clock input edge when an external clock is selected or the input capture function is used.

Bit 4: CKEG1	Bit 3: CKEG0	Description	
0	0	Count/input capture register set on rising edge	(Initial value)
	1	Count/input capture register set on falling edge	
1	X	Count/input capture register set on both rising and falling edges	

Note: X: 0 or 1 (don’t care)

Bits 2 to 0—Timer Prescaler 2 to 0 (TPSC2–TPSC0): These bits select the TCNT count clock.

When the on-chip RTC output clock is selected as the count clock for a channel, that channel can operate even in module standby mode. When another clock is selected, the channel does not operate in standby mode.

Bit 2: TPSC2	Bit 1: TPSC1	Bit 0: TPSC0	Description	
0	0	0	Counts on Pϕ/4	(Initial value)
		1	Counts on Pϕ/16	
	1	0	Counts on Pϕ/64	
		1	Counts on Pϕ/256	
1	0	0	Counts on Pϕ/1024	
		1	Reserved (Do not set)	
	1	0	Counts on on-chip RTC output clock	
		1	Counts on external clock	

12.2.6 Input Capture Register (TCPR2)

TCPR2 is a 32-bit read-only register for use with the input capture function, provided only in channel 2.

The input capture function is controlled by means of the input capture control bits (ICPE1, ICPE0) and clock edge bits (CKEG1, CKEG0) in TCR2. When input capture occurs, the TCNT2 value is copied into TCPR2. The value is set in TCPR2 only when the ICPF bit in TCR2 is 0.

TCPR2 is not initialized by a power-on or manual reset, or in standby mode.

Bit:	31	30	29	. . . .	2	1	0
Initial value:				Undefined			
R/W:	R	R	R		R	R	R

HITACHI

12.3 Operation

Each channel has a 32-bit timer counter (TCNT) that performs count-down operations, and a 32-bit timer constant register (TCOR). The channels have an auto-reload function that allows cyclic count operations, and can also perform external event counting. Channel 2 also has an input capture function.

12.3.1 Counter Operation

When one of bits STR0–STR2 is set to 1 in the timer start register (TSTR), the timer counter (TCNT) for the corresponding channel starts counting. When TCNT underflows, the UNF flag is set in the corresponding timer control register (TCR). If the UNIE bit in TCR is set to 1 at this time, an interrupt request is sent to the CPU. At the same time, the value is copied from TCOR into TCNT, and the count-down continues (auto-reload function).

Example of Count Operation Setting Procedure: Figure 12.2 shows an example of the count operation setting procedure.

1. Select the count clock with bits TPSC2–TPSC0 in the timer control register (TCR). When an external clock is selected, set the TCLK pin to input mode with the TCOE bit in TOCR, and select the external clock edge with bits CKEG1 and CKEG0 in TCR.
2. Specify whether an interrupt is to be generated on TCNT underflow with the UNIE bit in TCR.
3. When the input capture function is used, set the ICPE bits in TCR, including specification of whether the interrupt function is to be used.
4. Set a value in the timer constant register (TCOR).
5. Set the initial value in the timer counter (TCNT).
6. Set the STR bit to 1 in the timer start register (TSTR) to start the count.

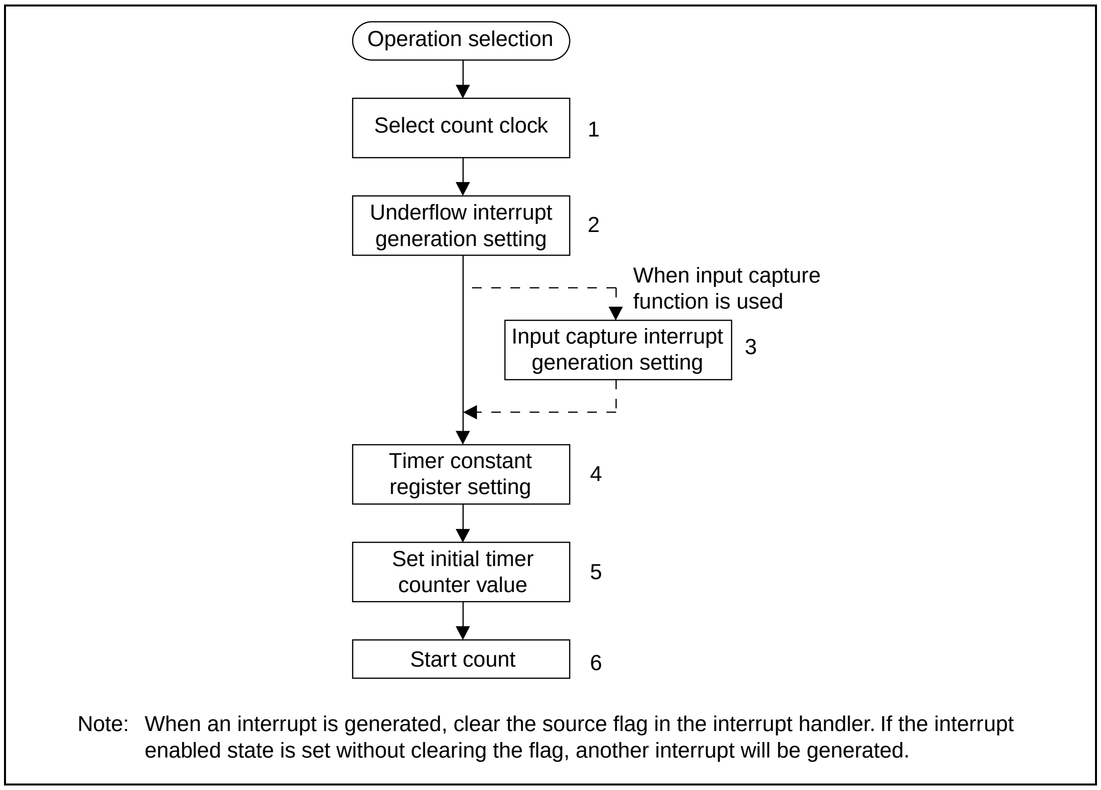

Operation selection
Select count clock
1
Underflow interrupt generation setting
2
When input capture function is used
Input capture interrupt generation setting
3
Timer constant register setting
4
Set initial timer counter value
5
Start count
6
Note: When an interrupt is generated, clear the source flag in the interrupt handler. If the interrupt enabled state is set without clearing the flag, another interrupt will be generated.

Figure 12.2 Example of Count Operation Setting Procedure

Auto-Reload Count Operation: Figure 12.3 shows the TCNT auto-reload operation.

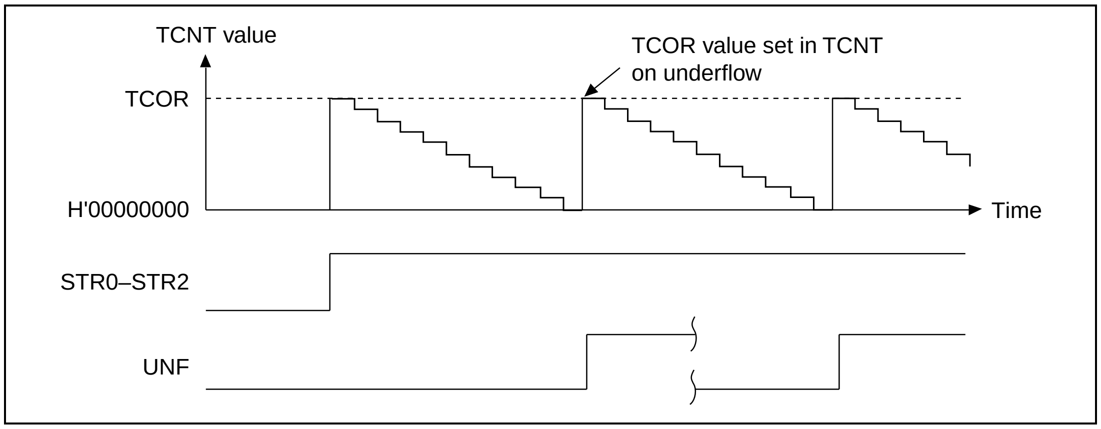


Figure 12.3 TCNT Auto-Reload Operation

TCNT Count Timing:

- Operating on internal clock

  Any of five count clocks (Pϕ/4, Pϕ/16, Pϕ/64, Pϕ/256, or Pϕ/1024) scaled from the peripheral module clock can be selected as the count clock by means of the TPSC2–TPSC0 bits in TCR.

  Figure 12.4 shows the timing in this case.

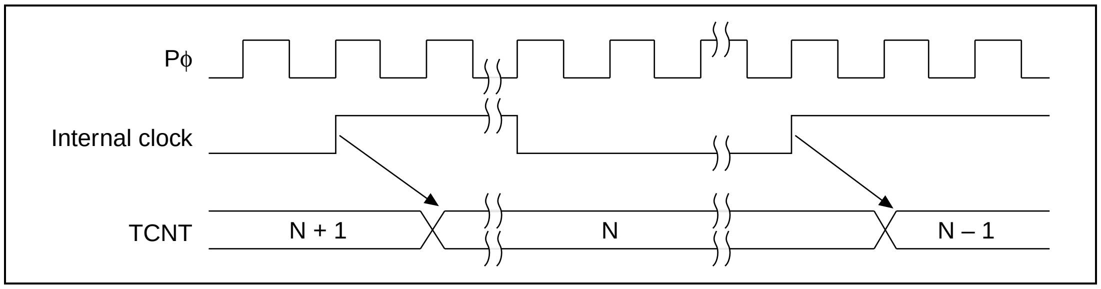


Figure 12.4 Count Timing when Operating on Internal Clock

- Operating on external clock

  External clock pin (TCLK) input can be selected as the timer clock by means of the TPSC2–TPSC0 bits in TCR. The detected edge (rising, falling, or both edges) can be selected with the CKEG1 and CKEG0 bits in TCR.

  Figure 12.5 shows the timing for both-edge detection.

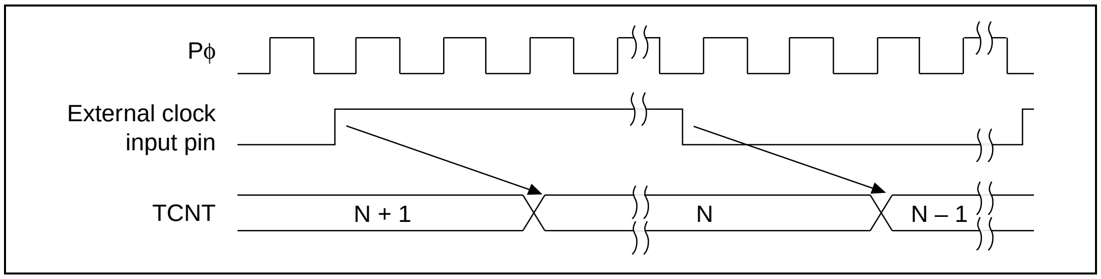


Figure 12.5 Count Timing when Operating on External Clock

- Operating on on-chip RTC output clock

  The on-chip RTC output clock can be selected as the timer clock by means of the TPSC2–TPSC0 bits in TCR. Figure 12.6 shows the timing in this case.

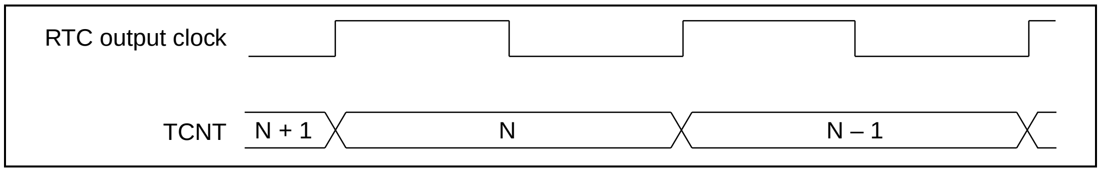


Figure 12.6 Count Timing when Operating on On-Chip RTC Output Clock

12.3.2 Input Capture Function

Channel 2 has an input capture function.

The procedure for using the input capture function is as follows:

1. Use the TCOE bit in the timer output control register (TOCR) to set the TCLK pin to input mode.
2. Use bits TPSC2–TPSC0 in the timer control register (TCR) to set an internal clock or the on-chip RTC output clock as the timer operating clock.
3. Use bits IPCE1 and IPCE0 in TCR to specify use of the input capture function, and whether interrupts are to generated when this function is used.
4. Use bits CKEG1 and CKEG0 in TCR to specify whether the rising or falling edge of the TCLK signal is to be used to set the timer counter (TCNT) value in the input capture register (TCPR2).

This function cannot be used in standby mode.

When input capture occurs, the TCNT2 value is set in TCPR2 only when the ICPF bit in TCR2 is 0. Also, a new DMAC transfer request is not generated until processing of the previous request is finished.

Figure 12.7 shows the operation timing when the input capture function is used (with TCLK rising edge detection).

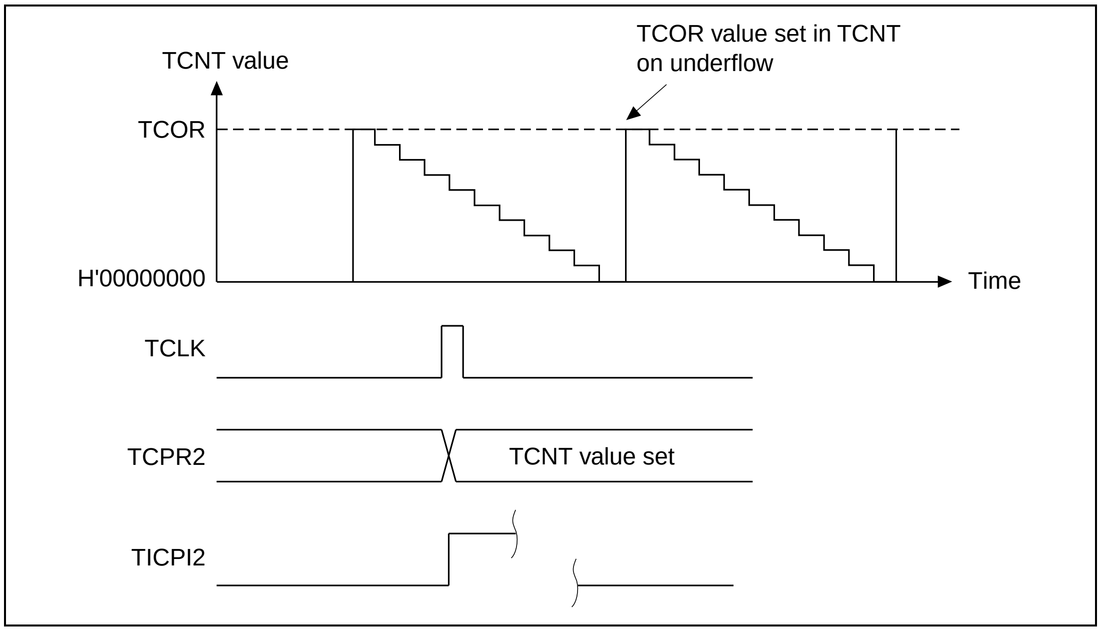


Figure 12.7   Operation Timing when Using Input Capture Function

12.4 Interrupts

There are four TMU interrupt sources, comprising underflow interrupts and the input capture interrupt (when the input capture function is used). Underflow interrupts are generated on channels 0 to 2, and input capture interrupts on channel 2 only.

An underflow interrupt request is generated (for each channel) according to the AND of UNF and the interrupt enable bit (UNIE) in TCR.

When the input capture function is used and an input capture request is generated, an interrupt is requested if the input capture input flag (ICPF) in TCR2 is 1 and the input capture control bits (ICPE1, ICPE0) in TCR2 are 11.

The TMU interrupt sources are summarized in table 12.3.

Table 12.3 TMU Interrupt Sources

Channel	Interrupt Source	Description	Priority
0	TUNI0	Underflow interrupt 0	High
1	TUNI1	Underflow interrupt 1	↑
2	TUNI2	Underflow interrupt 2	↓
	TICPI2	Input capture interrupt 2	Low

12.5 Usage Notes

12.5.1 Register Writes

When performing a register write, timer count operation must be stopped by clearing the start bit (STR0–STR2) for the relevant channel in the timer start register (TSTR).

12.5.2 TCNT Register Reads

When performing a TCNT register read, processing for synchronization with the timer count operation is performed. If a timer count operation and register read processing are performed simultaneously, the TCNT counter value prior to the count-down operation is read by means of the synchronization processing.

12.5.3 Resetting the RTC Frequency Divider

When the on-chip RTC output clock is selected as the count clock, the RTC frequency divider should be reset.

12.5.4 External Clock Frequency

Ensure that the external clock frequency for any channel does not exceed Pϕ/4.

Section 13 Bus State Controller (BSC)

13.1 Overview

The functions of the bus state controller (BSC) include division of the physical address space, and output of control signals in accordance with various types of memory and bus interface specifications. The BSC functions allow DRAM, synchronous DRAM, SRAM, ROM, etc., to be connected directly to the SH7750 without the use of external circuitry, and also support the PCMCIA interface protocol, enabling system design to be simplified and data transfers to be carried out at high speed by a compact system.

13.1.1 Features

The BSC has the following features:

- Physical address space is managed as 7 independent areas
  - Maximum 64 Mbytes for each of areas 0 to 6
  - Bus width of each area can be set in a register (except area 0, which uses an external pin setting)
  - Wait state insertion by $\overline{\text{RDY}}$ pin
  - Wait state insertion can be controlled by program
  - Specification of types of memory connectable to each area
  - Output of control signals allowing direct connection of memory to each area
  - Automatic wait cycle insertion to prevent data bus collisions in case of consecutive memory accesses to different areas, or a read access followed by a write access to the same area
  - Write strobe setup time and hold time periods can be inserted in a write cycle to enable connection to low-speed memory
- Normal memory (SRAM) interface
  - Wait state insertion can be controlled by program
  - Wait state insertion by $\overline{\text{RDY}}$ pin

    Connectable areas: 0 to 6

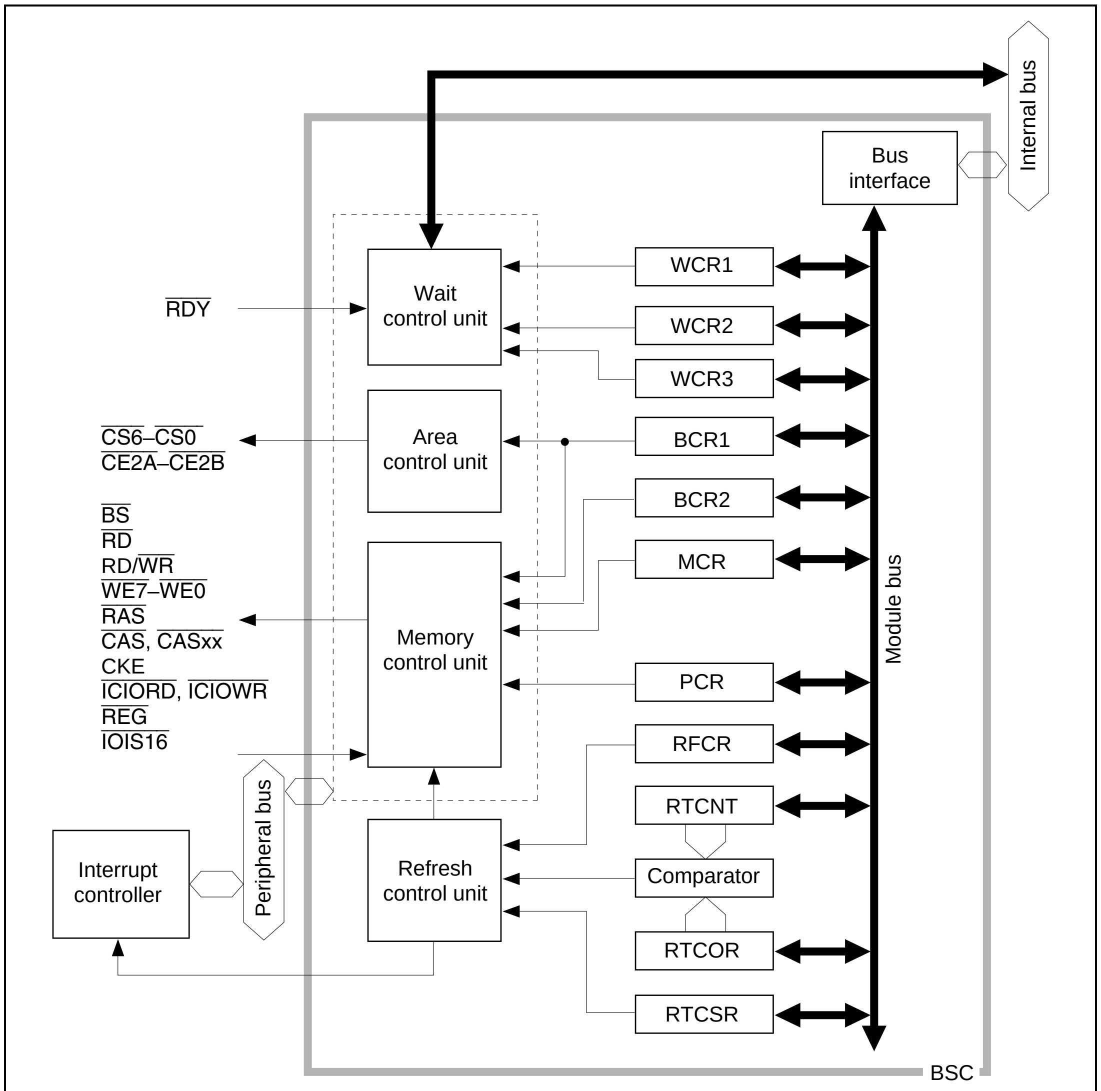
    Settable bus widths: 64, 32, 16, 8
- DRAM interface
  - Row address/column address multiplexing according to DRAM capacity
  - Burst operation (fast page mode, EDO mode)
  - CAS-before-RAS refresh and self-refresh
  - 8-CAS byte control for power-down operation
  - DRAM connection control signal timing can be controlled by register settings

— Consecutive accesses to the same row address
  Connectable areas: 2, 3
  Settable bus widths: 64, 32, 16

- Synchronous DRAM interface
  - Row address/column address multiplexing according to synchronous DRAM capacity
  - Burst operation
  - Auto-refresh and self-refresh
  - Synchronous DRAM connection control signal timing can be controlled by register settings
  - Consecutive accesses to the same row address
    Connectable areas: 2, 3
    Settable bus widths: 64, 32
- Burst ROM interface
  - Wait state insertion can be controlled by program
  - Burst operation, executing the number of transfers set in a register
    Connectable areas: 0, 5, 6
    Settable bus widths: 32, 16, 8
- MPX bus interface
  - Address/data multiplexing
    Connectable areas: 0 to 6
    Settable bus widths: 64, 32
- Byte control SRAM interface
  - SRAM interface with byte control
    Connectable areas: 1, 4
    Settable bus widths: 64, 32, 16
- PCMCIA interface
  - Wait state insertion can be controlled by program
  - Bus sizing function for I/O bus width
- Fine refreshing control
  - Supports refresh operation immediately after self-refresh operation in low-power DRAM by means of refresh counter overflow interrupt function
- Refresh counter can be used as interval timer
  - Interrupt request generated by compare-match
  - Interrupt request generated by refresh counter overflow

13.1.2 Block Diagram

Figure 13.1 shows a block diagram of the BSC.



WCR:	Wait control register	RFCR:	Refresh count register
BCR:	Bus control register	RTCNT:	Refresh timer count register
MCR:	Memory control register	RTCOR:	Refresh time constant register
PCR:	PCMCIA control register	RTCSR:	Refresh timer control/status register

Figure 13.1 Block Diagram of BSC

13.1.3 Pin Configuration

Table 13.1 shows the BSC pin configuration.

Table 13.1 BSC Pins

Name	Signals	I/O	Description
Address bus	A25–A0	O	Address output
Data bus	D63–D52, D51–D32	I/O	Data input/output When port functions are used, D51–D32 cannot be used. Leave open.
Data bus/port	D51–D32/ PORT19– PORT0	I/O	When port functions are not used: data input/output When port functions are used: input/output port (input or output set for each bit by register)
Bus cycle start	$\overline{\mathrm{BS}}$	O	Signal that indicates the start of a bus cycle When using synchronous DRAM: asserted once for a burst transfer For other burst transfers: asserted each data cycle
Chip select 6–0	$\overline{\mathrm{CS6}}$–$\overline{\mathrm{CS0}}$	O	Chip select signals that indicate the area being accessed $\overline{\mathrm{CS5}}$ and $\overline{\mathrm{CS6}}$ are also used as PCMCIA $\overline{\mathrm{CE1A}}$ and $\overline{\mathrm{CE1B}}$
Read/write	RD/$\overline{\mathrm{WR}}$	O	Data bus input/output direction designation signal Also used as the DRAM/synchronous DRAM/PCMCIA write designation signal
Row address strobe	$\overline{\mathrm{RAS}}$	O	$\overline{\mathrm{RAS}}$ signal when using DRAM/synchronous DRAM
Read/column address strobe/ cycle frame	$\overline{\mathrm{RD}}$/$\overline{\mathrm{CASS}}$/ $\overline{\mathrm{FRAME}}$	O	Strobe signal that indicates a read cycle When using synchronous DRAM: $\overline{\mathrm{CAS}}$ signal When using MPX bus: $\overline{\mathrm{FRAME}}$ signal
Data enable 0	$\overline{\mathrm{WE0}}$/$\overline{\mathrm{CAS0}}$/ DQM0	O	When using synchronous DRAM: selection signal for D7–D0 When using DRAM: $\overline{\mathrm{CAS}}$ signal for D7–D0 In other cases: write strobe signal for D7–D0
Data enable 1	$\overline{\mathrm{WE1}}$/$\overline{\mathrm{CAS1}}$/ DQM1	O	When using synchronous DRAM: selection signal for D15–D8 When using DRAM: $\overline{\mathrm{CAS}}$ signal for D15–D8 When using PCMCIA: write strobe signal In other cases: write strobe signal for D15–D8

Table 13.1 BSC Pins (cont)

Name	Signals	I/O	Description
Data enable 2	$\overline{\mathrm{WE2}}$/$\overline{\mathrm{CAS2}}$/ $\overline{\mathrm{DQM2/ICIORD}}$	O	When using synchronous DRAM: selection signal for D23–D16 When using DRAM: $\overline{\mathrm{CAS}}$ signal for D23–D16 When using PCMCIA: $\overline{\mathrm{ICIORD}}$ signal In other cases: write strobe signal for D23–D16
Data enable 3	$\overline{\mathrm{WE3}}$/$\overline{\mathrm{CAS3}}$/ $\overline{\mathrm{DQM3/ICIOWR}}$	O	When using synchronous DRAM: selection signal for D31–D24 When using DRAM: $\overline{\mathrm{CAS}}$ signal for D31–D24 When using PCMCIA: $\overline{\mathrm{ICIOWR}}$ signal In other cases: write strobe signal for D31–D24
Data enable 4	$\overline{\mathrm{WE4}}$/$\overline{\mathrm{CAS4}}$/ DQM4	O	When using synchronous DRAM: selection signal for D39–D32 When using DRAM: $\overline{\mathrm{CAS}}$ signal for D39–D32 In other cases: write strobe signal for D39–D32
Data enable 5	$\overline{\mathrm{WE5}}$/$\overline{\mathrm{CAS5}}$/ DQM5	O	When using synchronous DRAM: selection signal for D47–D40 When using DRAM: $\overline{\mathrm{CAS}}$ signal for D47–D40 In other cases: write strobe signal for D47–D40
Data enable 6	$\overline{\mathrm{WE6}}$/$\overline{\mathrm{CAS6}}$/ DQM6	O	When using synchronous DRAM: selection signal for D55–D48 When using DRAM: $\overline{\mathrm{CAS}}$ signal for D55–D48 In other cases: write strobe signal for D55–D48
Data enable 7	$\overline{\mathrm{WE7}}$/$\overline{\mathrm{CAS7}}$/ DQM7/$\overline{\mathrm{REG}}$	O	When using synchronous DRAM: selection signal for D63–D56 When using DRAM: $\overline{\mathrm{CAS}}$ signal for D63–D56 When using PCMCIA: $\overline{\mathrm{REG}}$ signal In other cases: write strobe signal for D63–D56
Ready	$\overline{\mathrm{RDY}}$	I	Wait state request signal
Area 0 MPX bus specification/16-bit I/O	MD6/$\overline{\mathrm{IOIS16}}$	I	In power-on reset: Designates area 0 bus as MPX bus (1: SRAM, 0: MPX) When using PCMCIA: 16-bit I/O designation signal. Valid only in little-endian mode.
Clock enable	CKE	O	Synchronous DRAM clock enable control signal
Bus release request	$\overline{\mathrm{BREQ}}$/ $\overline{\mathrm{BSACK}}$	I	Bus release request signal/bus acknowledge signal

Table 13.1 BSC Pins (cont)

Name	Signals	I/O	Description
Bus use permission	$\overline{\text{BACK}}$/ $\overline{\text{BSREQ}}$	O	Bus use permission signal/bus request
Area 0 bus width/PCMCIA card select	MD3/$\overline{\text{CE2A}}$*[1] MD4/$\overline{\text{CE2B}}$*[2]	I/O	In power-on reset: external space area 0 bus width specification signal When using PCMCIA: $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$
Endian switchover/ row address strobe	MD5/$\overline{\text{RAS2}}$*[3]	I/O	Endian specification in a power-on reset. $\overline{\text{RAS2}}$ when DRAM is connected to area 2
Master/slave switchover	MD7/TXD	I/O	Indicates master/slave status in a power-on reset. Serial interface TXD
DMAC0 acknowledge signal	DACK0	O	DMAC channel 0 data acknowledge
DMAC1 acknowledge signal	DACK1	O	DMAC channel 1 data acknowledge
Read/column address strobe/ cycle frame 2	$\overline{\text{RD2}}$	O	Same signal as $\overline{\text{RD}}$/$\overline{\text{CASS}}$/$\overline{\text{FRAME}}$ This signal is used when the $\overline{\text{RD}}$/$\overline{\text{CASS}}$/$\overline{\text{FRAME}}$ signal load is heavy.
Read/write 2	RD/$\overline{\text{WR2}}$	O	Same signal as RD/$\overline{\text{WR}}$ This signal is used when the RD/$\overline{\text{WR}}$ signal load is heavy.

Notes: 1. MD3/$\overline{\text{CE2A}}$ input/output switching is performed by BCR1.A56PCM. Output is selected when BCR1.A56PCM = 1.

2. MD4/$\overline{\text{CE2B}}$ input/output switching is performed by BCR1.A56PCM. Output is selected when BCR1.A56PCM = 1.

3. MD5/$\overline{\text{RAS2}}$ input/output switching is performed by BCR1.DRAMTP. Output is selected when BCR1.DRAMTP (2–0) = 101.

13.1.4 Register Configuration

The BSC has the 11 registers shown in table 13.2. In addition, the synchronous DRAM mode register incorporated in synchronous DRAM can also be accessed as an SH7750 register. The functions of these registers include control of direct interfaces to various types of memory, wait states, and refreshing.

Table 13.2 BSC Registers

Name		Abbreviation	R/W	Initial Value	P4 Address	Area 7 Address	Access Size
Bus control register 1		BCR1	R/W	H'0000 0000	H'FF80 0000	H'1F80 0000	32
Bus control register 2		BCR2	R/W	H'3FFC	H'FF80 0004	H'1F80 0004	16
Wait state control register 1		WCR1	R/W	H'7777 7777	H'FF80 0008	H'1F80 0008	32
Wait state control register 2		WCR2	R/W	H'FFFE EFFF	H'FF80 000C	H'1F80 000C	32
Wait state control register 3		WCR3	R/W	H'0777 7777	H'FF80 0010	H'1F80 0010	32
Memory control register		MCR	R/W	H'0000 0000	H'FF80 0014	H'1F80 0014	32
PCMCIA control register		PCR	R/W	H'0000	H'FF80 0018	H'1F80 0018	16
Refresh timer control/status register		RTCSR	R/W	H'0000	H'FF80 001C	H'1F80 001C	16
Refresh timer counter		RTCNT	R/W	H'0000	H'FF80 0020	H'1F80 0020	16
Refresh time constant counter		RTCOR	R/W	H'0000	H'FF80 0024	H'1F80 0024	16
Refresh count register		RFCR	R/W	H'0000	H'FF80 0028	H'1F80 0028	16
Synchronous DRAM mode registers	For area 2	SDMR2	W	—	H'FF90 xxxx*	H'1F90 xxxx	8
	For area 3	SDMR3			H'FF94 xxxx*	H'1F94 xxxx	

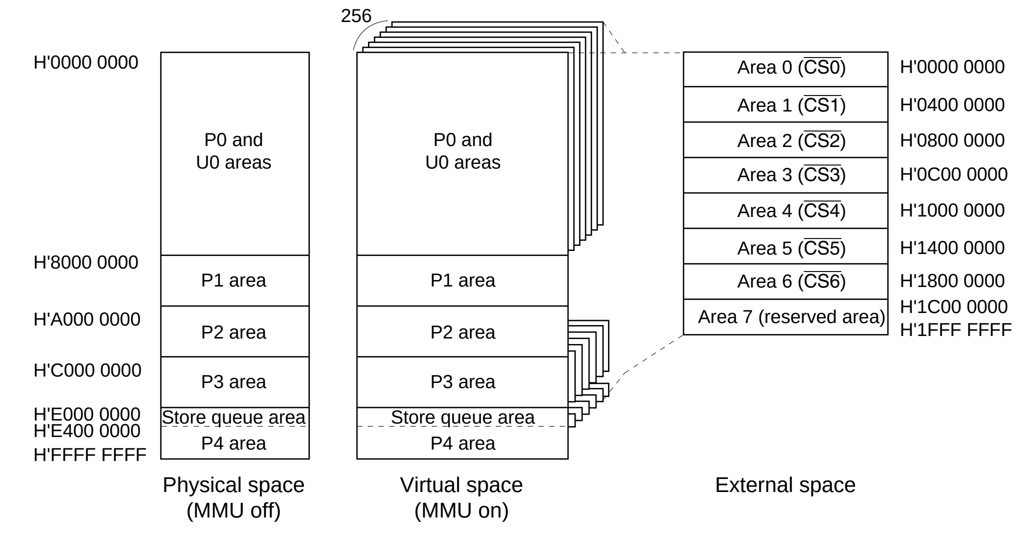
Note: * For details, see section 13.2.8, Synchronous DRAM Mode Registers.

13.1.5 Overview of Areas

Space Divisions: The architecture of the SH7750 provides a 32-bit virtual address space. The virtual space is divided into five areas according to the upper address value. External space comprises a 29-bit address space, divided into eight areas.

The virtual space can be allocated to any external space by means of the memory management unit (MMU). Details are given in section 3, Memory Management Unit (MMU). This section describes the areas into which the external space is divided.

With the SH7750, various kinds of memory or PC cards can be connected to the seven areas of external space as shown in table 13.3, and chip select signals ($\overline{\text{CS0}}$–$\overline{\text{CS6}}$, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$) are output for each of these areas. $\overline{\text{CS0}}$ is asserted when accessing area 0, and $\overline{\text{CS6}}$ when accessing area 6. When DRAM or synchronous DRAM is connected to area 2 or 3, signals such as $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, RD/$\overline{\text{WR}}$, and DQM are also asserted. When the PCMCIA interface is selected for area 5 or 6, $\overline{\text{CE2A}}$/$\overline{\text{CE2B}}$ is asserted in addition to $\overline{\text{CS5}}$/$\overline{\text{CS6}}$ for the byte to be accessed.



Notes: 1. When the MMU is off (MMUCR.AT = 0), the top 3 bits of the 32-bit address are ignored, and memory is mapped onto a fixed 29-bit external space.
2. When the MMU is on (MMUCR.AT = 1), the P0, U0, P3, and store queue areas can be mapped onto any external space using the TLB.
For details, see section 3, Memory Management Unit (MMU).

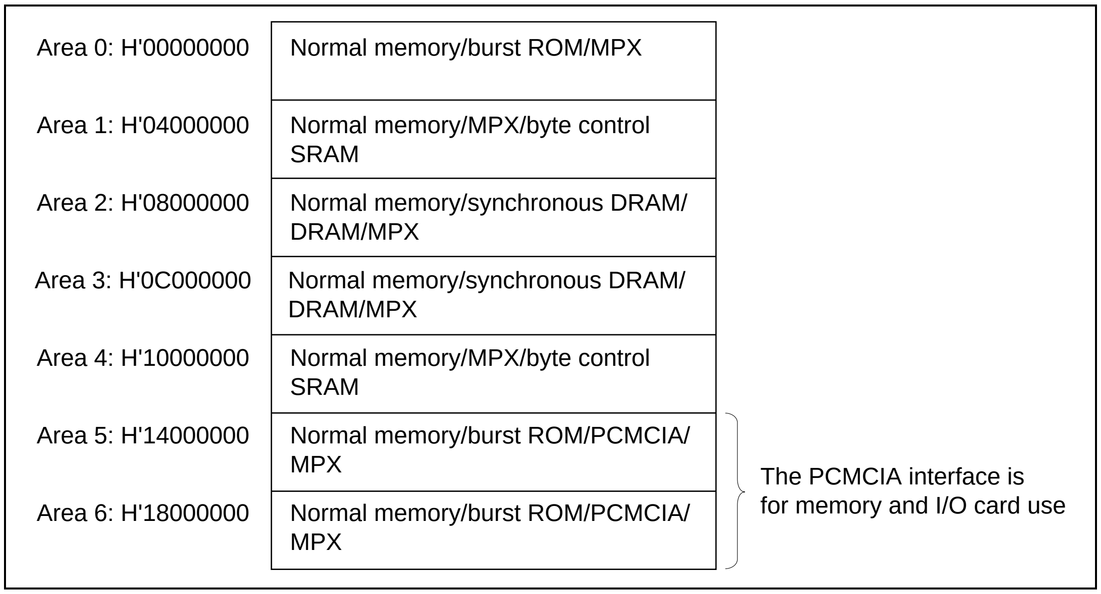
Figure 13.2 Correspondence between Virtual Address Space and External Address Space

HITACHI

Table 13.3 External Address Space Map

Area	External Addresses	Size	Connectable Memory	Settable Bus Widths	Access Size
0	H'00000000–H'03FFFFFF	64 Mbytes	Normal memory	8, 16, 32, 64*[1]	8, 16, 32, 64
			Burst ROM	8, 16, 32*[1]	
			MPX	32, 64*[1]	
1	H'04000000–H'07FFFFFF	64 Mbytes	Normal memory	8, 16, 32, 64*[2]	8, 16, 32, 64
			MPX	32, 64*[2]	
			Byte control SRAM	16, 32, 64*[2]	
2	H'08000000–H'0BFFFFFF	64 Mbytes	Normal memory	8, 16, 32, 64*[2]	8, 16, 32, 64
			Synchronous DRAM	32, 64*[2], *[3]	
			DRAM	16, 32*[2], *[3]	
			MPX	32, 64*[2]	
3	H'0C000000–H'0FFFFFFF	64 Mbytes	Normal memory	8, 16, 32, 64*[2]	8, 16, 32, 64
			Synchronous DRAM	32, 64*[2], *[3]	
			DRAM	16, 32, 64*[2], *[3]	
			MPX	32, 64*[2]	
4	H'10000000–H'13FFFFFF	64 Mbytes	Normal memory	8, 16, 32, 64*[2]	8, 16, 32, 64
			MPX	32, 64*[2]	
			Byte control RAM	16, 32, 64*[2]	
5	H'14000000–H'17FFFFFF	64 Mbytes	Normal memory	8, 16, 32, 64*[2]	8, 16, 32, 64
			MPX	32, 64*[2]	
			Burst ROM	8, 16, 32*[2]	
			PCMCIA	8, 16*[2], *[4]	
6	H'18000000–H'1BFFFFFF	64 Mbytes	Normal memory	8, 16, 32, 64*[2]	8, 16, 32, 64
			MPX	32, 64*[2]	
			Burst ROM	8,16, 32*[2]	
			PCMCIA	8,16*[2], *[4]	
7*[5]	H'1C000000–H'1FFFFFFF	64 Mbytes	—	—	n: 0 to 7

Notes: 1. Memory bus width specified by external pins
2. Memory bus width specified by register
3. With synchronous DRAM interface, bus width is 32 or 64 bits only.
With DRAM interface, bus width is 16 or 32 bits only for area 2, and 16, 32, or 64 bits only for area 3.
4. With PCMCIA interface, bus width is 8 or 16 bits only.
5. Do not access a reserved area, as operation cannot be guaranteed in this case.



Area 0: H'00000000
Normal memory/burst ROM/MPX
Area 1: H'04000000
Normal memory/MPX/byte control SRAM
Area 2: H'08000000
Normal memory/synchronous DRAM/ DRAM/MPX
Area 3: H'0C000000
Normal memory/synchronous DRAM/ DRAM/MPX
Area 4: H'10000000
Normal memory/MPX/byte control SRAM
Area 5: H'14000000
Normal memory/burst ROM/PCMCIA/ MPX
Area 6: H'18000000
Normal memory/burst ROM/PCMCIA/ MPX
The PCMCIA interface is for memory and I/O card use

Figure 13.3 External Space Allocation

Memory Bus Width: In the SH7750, the memory bus width can be set independently for each space. For area 0, a bus size of 8, 16, 32, or 64 bits can be selected in a power-on reset, using external pins. The relationship between the external pins (MD4 and MD3) and the bus width in a power-on reset is shown below.

MD4	MD3	Bus Width
0	0	64 bits
	1	8 bits
1	0	16 bits
	1	32 bits

When normal memory or ROM is used in areas 1 to 6, a bus width of 8, 16, 32, or 64 bits can be selected with bus control register 2 (BCR2). When burst ROM is used, a bus width of 8, 16, or 32 bits can be selected. When byte control SRAM is used, a bus width of 16, 32, or 64 bits can be selected. When the MPX bus is used, a bus width of 32 or 64 bits can be selected. When the DRAM interface is used, a bus width of 16, 32, or 64 bits can be selected with the memory control register (MCR). When the DRAM interface is used for area 2 or 3, a bus width of 16 or 32 bits should be set. For the synchronous DRAM interface, set a bus width of 32 or 64 bits in the MCR register.

When using the PCMCIA interface, set a bus width of 8 or 16 bits.

When using port functions, set a bus width of 8, 16, or 32 bits for all areas.

For details, see section 13.2.2, Bus Control Register 2 (BCR2), and section 13.2.6, Memory Control Register (MCR).

The area 7 address range, H'1C000000 to H'1FFFFFFFF, is a reserved space and must not be used.

13.1.6 PCMCIA Support

The SH7750 supports PCMCIA compliant interface specifications for physical space areas 5 and 6.

The interfaces supported are basically the IC memory card interface and I/O card interface stipulated in JEIDA specifications version 4.2 (PCMCIA2.1).

Physical space areas 5 and 6 support both the IC memory card interface and the I/O card interface.

The PCMCIA interface is supported only in little-endian mode.

Table 13.4 PCMCIA Interface Features

Item	Features
Access	Random access
Data bus	8/16 bits
Memory type	Mask ROM, OTPROM, EPROM, EEPROM, flash memory, SRAM
Common memory capacity	Max. 64 Mbytes
Attribute memory capacity	Max. 64 Mbytes
Others	Dynamic bus sizing for I/O bus width, access to PCMCIA interface from address translation areas

Table 13.5 PCMCIA Support Interfaces

	IC Memory Card Interface			I/O Card Interface			Corresponding SH7750 Pin
Pin	**Signal Name**	**I/O**	**Function**	**Signal Name**	**I/O**	**Function**	
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	$\overline{\text{CE1}}$	I	Card enable	$\overline{\text{CE1}}$	I	Card enable	$\overline{\text{CS5}}$ or $\overline{\text{CS6}}$
8	A10	I	Address	A10	I	Address	A10
9	$\overline{\text{OE}}$	I	Output enable	$\overline{\text{OE}}$	I	Output enable	$\overline{\text{RD}}$
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	$\overline{\text{WE}}/\overline{\text{PGM}}$	I	Write enable	$\overline{\text{WE}}/\overline{\text{PGM}}$	I	Write enable	$\overline{\text{WE1}}$
16	$\overline{\text{RDY}}/\overline{\text{BSY}}$	O	Ready/busy	$\overline{\text{IREQ}}$	O	Interrupt request	Sensed on port
17	VCC		Operating power supply	VCC		Operating power supply	—
18	VPP1		Programming power supply	VPP1		Programming/ peripheral power supply	—
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

Table 13.5 PCMCIA Support Interfaces (cont)

	IC Memory Card Interface			I/O Card Interface			Corresponding SH7750 Pin
Pin	**Signal Name**	**I/O**	**Function**	**Signal Name**	**I/O**	**Function**	
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	$\overline{\text{WP}}$	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	Sensed on port
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	RFSH	I	Refresh request	RFSH	I	Refresh request	Output from port
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
49	A20	I	Address	A20	I	Address	A20
50	A21	I	Address	A21	I	Address	A21
51	VCC		Power supply	VCC		Power supply	—
52	VPP2		Programming power supply	VPP2		Programming/ peripheral power supply	—
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

Table 13.5 PCMCIA Support Interfaces (cont)

	IC Memory Card Interface			I/O Card Interface			Corresponding SH7750 Pin
Pin	**Signal Name**	**I/O**	**Function**	**Signal Name**	**I/O**	**Function**	
57	RFU		Reserved	RFU		Reserved	—
58	RESET	I	Reset	RESET	I	Reset	Output from port
59	$\overline{\text{WAIT}}$	O	Wait request	$\overline{\text{WAIT}}$	O	Wait request	$\overline{\text{RDY}}$
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	$\overline{\text{WE7}}$
62	BVD2	O	Battery voltage detection	$\overline{\text{SPKR}}$	O	Digital speech signal	Sensed on port
63	BVD1	O	Battery voltage detection	$\overline{\text{STSCHG}}$	O	Card status change	Sensed on port
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	Sensed on port
68	GND		Ground	GND		Ground	—

13.2 Register Descriptions

13.2.1 Bus Control Register 1 (BCR1)

Bus control register 1 (BCR1) is a 32-bit readable/writable register that specifies the function, bus cycle status, etc., of each area.

BCR1 is initialized to H'00000000 by a power-on reset, but is not initialized by a manual reset or in standby mode. External memory other than area 0 should not be accessed until register initialization is completed.

Bit:	31	30	29	28	27	26	25	24
Bit name:	ENDIAN	MASTER	A0MPX	—	—	—	IPUP	OPUP
Initial value:	0/1*	0/1*	0/1*	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
Bit name:	—	—	A1MBC	A4MBC	BREQEN	PSHR	MEMMPX	—
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

Bit:	15	14	13	12	11	10	9	8
Bit name:	HIZMEM	HIZCNT	A0BST2	A0BST1	A0BST0	A5BST2	A5BST1	A5BST0
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:	A6BST2	A6BST1	A6BST0	DRAMTP2	DRAMTP1	DRAMTP0	—	A56PCM
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W

Note: * These bits sample external pin values in a power-on reset.

Bit 31—Endian Flag (ENDIAN): Samples the value of the endian specification external pin (MD5) in a power-on reset. The endian mode of all spaces is determined by this bit. ENDIAN is a read-only bit.

Bit 31: ENDIAN	Description
0	In a power-on reset, the endian setting external pin (MD5) is low, designating big-endian mode for the SH7750
1	In a power-on reset, the endian setting external pin (MD5) is high, designating little-endian mode for the SH7750

Bit 30—Master/Slave Flag (MASTER): Samples the value of the master/slave specification external pin (MD7) in a power-on reset. The master/slave status of all spaces is determined by this bit. MASTER is a read-only bit.

Bit 30: MASTER	Description
0	In a power-on reset, the master/slave setting external pin (MD7) is low, designating master mode for the SH7750
1	In a power-on reset, the master/slave setting external pin (MD7) is high, designating slave mode for the SH7750

Bit 29—Area 0 Memory Type (A0MPX): Samples the value of the area 0 memory type specification external pin (MD6) in a power-on reset. The memory type of area 0 is determined by this bit. A0MPX is a read-only bit.

Bit 29: A0MPX	Description
0	In a power-on reset, the external pin specifying the area 0 memory type (MD6) is low, designating the area 0 memory type as normal memory
1	In a power-on reset, the external pin specifying the area 0 memory type (MD6) is high, designating the area 0 memory type as MPX

Bits 28 to 26, 23, 22, 16, and 1—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 25—Control Input Pin Pull-Up Resistor Control (IPUP): Specifies the pull-up resistor status for control input pins (NMI, $\overline{\text{IRL0}}$–$\overline{\text{IRL3}}$, $\overline{\text{BREQ}}$, MD6/$\overline{\text{IOIS16}}$, $\overline{\text{RDY}}$). IPUP is initialized by a power-on reset.

Bit 25: IPUP	Description
0	Pull-up resistor is on for control input pins (NMI, $\overline{\text{IRL0}}$–$\overline{\text{IRL3}}$, $\overline{\text{BREQ}}$, MD6/$\overline{\text{IOIS16}}$, $\overline{\text{RDY}}$) (Initial value)
1	Pull-up resistor is off for control input pins (NMI, $\overline{\text{IRL0}}$–$\overline{\text{IRL3}}$, $\overline{\text{BREQ}}$, MD6/$\overline{\text{IOIS16}}$, $\overline{\text{RDY}}$)

Bit 24—Control Output Pin Pull-Up Resistor Control (OPUP): Specifies the pull-up resistor status for control output pins (A[25:0], $\overline{\text{BS}}$, $\overline{\text{CSn}}$, $\overline{\text{RD}}$, $\overline{\text{WEn}}$, RD/$\overline{\text{WR}}$, $\overline{\text{RAS}}$, $\overline{\text{RAS2}}$, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, $\overline{\text{RD2}}$, RD/$\overline{\text{WR2}}$) when high-impedance. OPUP is initialized by a power-on reset.

Bit 24: OPUP	Description
0	Pull-up resistor is on for control output pins (A[25:0], $\overline{\text{BS}}$, $\overline{\text{CSn}}$, $\overline{\text{RD}}$, $\overline{\text{WEn}}$, RD/$\overline{\text{WR}}$, $\overline{\text{RAS}}$, $\overline{\text{RAS2}}$, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, $\overline{\text{RD2}}$, RD/$\overline{\text{WR2}}$) (Initial value)
1	Pull-up resistor is off for control output pins (A[25:0], $\overline{\text{BS}}$, $\overline{\text{CSn}}$, $\overline{\text{RD}}$, $\overline{\text{WEn}}$, RD/$\overline{\text{WR}}$, $\overline{\text{RAS}}$, $\overline{\text{RAS2}}$, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, $\overline{\text{RD2}}$, RD/$\overline{\text{WR2}}$)

Bit 21—Area 1 SRAM Byte Control Mode (A1MBC): MPX has priority when an MPX bus specification is made. This bit is initialized by a power-on reset.

Bit 21: A1MBC	Description
0	Area 1 SRAM is set to normal mode (Initial value)
1	Area 1 SRAM is set to byte control mode

Bit 20—Area 4 SRAM Byte Control Mode (A4MBC): MPX has priority when an MPX bus specification is made. This bit is initialized by a power-on reset.

Bit 20: A4MBC	Description
0	Area 4 SRAM is set to normal mode (Initial value)
1	Area 4 SRAM is set to byte control mode

Bit 19—BREQ Enable (BREQEN): Indicates whether external requests can be accepted. BREQEN is initialized to the external request acceptance disabled state by a power-on reset. It is ignored in the case of a slave mode startup.

Bit 19: BREQEN	Description
0	External requests are not accepted (Initial value)
1	External requests are accepted

Bit 18—Partial-Sharing Bit (PSHR): Sets partial-sharing mode. PSHR is valid only in the case of a master mode startup.

Bit 18: PSHR	Description
0	Master mode (Initial value)
1	Partial-sharing mode

Bit 17—Area 1 to 6 MPX Bus Specification (MEMMPX): Sets the MPX bus when areas 1 to 6 are set as normal memory (or burst ROM). MEMMPX is initialized by a power-on reset.

Bit 17: MEMMPX	Description
0	Basic interface (or burst ROM interface) is selected when areas 1 to 6 are set as normal memory (or burst ROM) (Initial value)
1	MPX bus interface is selected when areas 1 to 6 are set as normal memory (or burst ROM)

Bit 15—High-Z Control (HIZMEM): Specifies the state of address and other signals (A[25:0], $\overline{\text{BS}}$, $\overline{\text{CSn}}$, RD/$\overline{\text{WR}}$, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, RD/$\overline{\text{WR2}}$) in standby mode.

Bit 15: HIZMEM	Description
0	The A[25:0], $\overline{\text{BS}}$, $\overline{\text{CSn}}$, RD/$\overline{\text{WR}}$, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, and RD/WR2 signals go to high-impedance (High-Z) in standby mode and when the bus is released (Initial value)
1	The A[25:0], $\overline{\text{BS}}$, $\overline{\text{CSn}}$, RD/$\overline{\text{WR}}$, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, and RD/$\overline{\text{WR2}}$ signals drive in standby mode

Bit 14—High-Z Control (HIZCNT): Specifies the state of the $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ signals in standby mode and when the bus is released.

Bit 14: HIZCNT	Description
0	The $\overline{\text{RAS}}$, $\overline{\text{RAS2}}$, $\overline{\text{WEn}}$/$\overline{\text{CASn}}$/DQMn, $\overline{\text{RD}}$/$\overline{\text{CASS}}$/$\overline{\text{FRAME}}$, and $\overline{\text{RD2}}$ signals go to high-impedance (High-Z) in standby mode and when the bus is released (Initial value)
1	The $\overline{\text{RAS}}$, $\overline{\text{RAS2}}$, $\overline{\text{WEn}}$/$\overline{\text{CASn}}$/DQMn, $\overline{\text{RD}}$/$\overline{\text{CASS}}$/$\overline{\text{FRAME}}$, and $\overline{\text{RD2}}$ signals drive in standby mode and when the bus is released

Bits 13 to 11—Area 0 Burst ROM Control (A0BST2–A0BST0): These bits specify whether burst ROM is used in external space area 0. When burst ROM is used, they also specify the number of accesses in a burst. If area 0 is an MPX interface area, these bits are ignored.

Bit 13: A0BST2	Bit 12: A0BST1	Bit 11: A0BST0	Description
0	0	0	Area 0 is accessed as normal memory (Initial value)
		1	Area 0 is accessed as burst ROM (4 consecutive accesses) Can be used with 8-, 16-, or 32-bit bus width
	1	0	Area 0 is accessed as burst ROM (8 consecutive accesses) Can be used with 8-, 16-, or 32-bit bus width
		1	Area 0 is accessed as burst ROM (16 consecutive accesses) Can only be used with 8- or 16-bit bus width. Do not specify for 32-bit bus width
1	0	0	Area 0 is accessed as burst ROM (32 consecutive accesses) Can only be used with 8-bit bus width
		1	Reserved
	1	0	Reserved
		1	Reserved

Bits 10 to 8—Area 5 Burst Enable (A5BST2–A5BST0): These bits specify whether burst ROM is used in external space area 5. When burst ROM is used, they also specify the number of accesses in a burst. If area 5 is an MPX interface area, these bits are ignored.

Bit 10: A5BST2	Bit 9: A5BST1	Bit 8: A5BST0	Description
0	0	0	Area 5 is accessed in normal mode (Initial value)
		1	Area 5 is burst-accessed (4 consecutive accesses) Can be used with 8-, 16-, or 32--bit bus width
	1	0	Area 5 is burst-accessed (8 consecutive accesses) Can be used with 8-, 16-, or 32-bit bus width
		1	Area 5 is burst-accessed (16 consecutive accesses) Can only be used with 8- or 16-bit bus width. Do not specify for 32-bit bus width
1	0	0	Area 5 is burst-accessed (32 consecutive accesses) Can only be used with 8-bit bus width
		1	Reserved
	1	0	Reserved
		1	Reserved

Note: Clear to 0 when PCMCIA is used.

Bits 7 to 5—Area 6 Burst Enable (A6BST2–A6BST0): These bits specify whether burst ROM is used in external space area 6. When burst ROM is used, they also specify the number of accesses in a burst. If area 6 is an MPX interface area, these bits are ignored.

Bit 7: A6BST2	Bit 6: A6BST1	Bit 5: A6BST0	Description
0	0	0	Area 6 is accessed in normal mode (Initial value)
		1	Area 6 is burst-accessed (4 consecutive accesses) Can be used with 8-, 16-, or 32--bit bus width
	1	0	Area 6 is burst-accessed (8 consecutive accesses) Can be used with 8-, 16-, or 32-bit bus width
		1	Area 6 is burst-accessed (16 consecutive accesses) Can only be used with 8- or 16-bit bus width. Do not specify for 32-bit bus width
1	0	0	Area 6 is burst-accessed (32 consecutive accesses) Can only be used with 8-bit bus width
		1	Reserved
	1	0	Reserved
		1	Reserved

Note: Clear to 0 when PCMCIA is used.

Bits 4 to 2—Area 2 and 3 Memory Type (DRAMTP2–DRAMTP0): These bits specify the type of memory connected to external space areas 2 and 3. ROM, SRAM, flash ROM, etc., can be directly connected as normal memory. DRAM and synchronous DRAM can also be directly connected.

Bit 4: DRAMTP2	Bit 3: DRAMTP1	Bit 2: DRAMTP0	Description
0	0	0	Areas 2 and 3 are normal memory or MPX*[1] (Initial value)
		1	Reserved (Cannot be set)
	1	0	Area 2 is normal memory or MPX*[1], area 3 is synchronous DRAM
		1	Areas 2 and 3 are synchronous DRAM
1	0	0	Area 2 is normal memory or MPX*[1], area 3 is DRAM
		1	Areas 2 and 3 are DRAM*[2]
	1	0	Reserved (Cannot be set)
		1	Reserved (Cannot be set)

Note: 1. Selection of normal memory or MPX is determined by the setting of the MEMMPX bit
2. When this mode is selected, 16 or 32 bits should be specified as the bus width for areas 2 and 3. In this mode the MD5 pin is designated for output as the $\overline{\text{RAS2}}$ pin.

Bit 0—Area 5 and 6 Bus Type (A56PCM): Specifies whether external space areas 5 and 6 are accessed as PCMCIA space. The setting of these bits has priority over the MEMMPX and AnBST bit settings.

Bit 0: A56PCM	Description
0	External space areas 5 and 6 are accessed as normal memory (Initial value)
1	External space areas 5 and 6 are accessed as PCMCIA space*

Note: * The MD3 pin is designated for output as the $\overline{\text{CE2A}}$ pin.
The MD4 pin is designated for output as the $\overline{\text{CE2B}}$ pin.

13.2.2 Bus Control Register 2 (BCR2)

Bus control register 2 (BCR2) is a 16-bit readable/writable register that specifies the bus width for each area, and whether a 16-bit port is used.

BCR2 is initialized to H'3FFC by a power-on reset, but is not initialized by a manual reset or in standby mode. External memory other than area 0 should not be accessed until register initialization is completed.

Bit:	15	14	13	12	11	10	9	8
Bit name:	A0SZ1	A0SZ0	A6SZ1	A6SZ0	A5SZ1	A5SZ0	A4SZ1	A4SZ0
Initial value:	0/1*	0/1*	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	A1SZ1	A0SZ0	—	PORTEN
Initial value:	1	1	1	1	1	1	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	—	R/W

Note: * These bits sample the values of the external pins that specify the area 0 bus size.

Bits 15 and 14—Area 0 Bus Width (A0SZ1, A0SZ0): These bits sample the external pins (MD3 and MD4) that specify the bus size in a power-on reset. They are read-only bits.

Bits 2n + 1, 2n—Area n (1 to 6) Bus Width Specification (AnSZ1, AnSZ0): These bits specify the bus width of physical space area n (n = 1 to 6).

(Bit 0): PORTEN	Bit 2n + 1: AnSZ1	Bit 2n: AnSZ0	Description
0	0	0	Bus width is 64 bits (Initial value)
		1	Bus width is 8 bits
	1	0	Bus width is 16 bits
		1	Bus width is 32 bits
1	0	0	Reserved (Setting prohibited)
		1	Bus width is 8 bits
	1	0	Bus width is 16 bits
		1	Bus width is 32 bits

Bit 1—Reserved: This bit is always read as 0, and should only be written with 0.

Bit 0—Port Function Enable (PORTEN): Specifies whether pins D51 to D32 are used as a 20-bit port. When this function is used, a bus width of 8, 16, or 32 bits should be set for all areas.

Bit 0: PORTEN	Description	
0	D51 to D32 are not used as a port	(Initial value)
1	D51 to D32 are used as a port	

13.2.3 Wait Control Register 1 (WCR1)

Wait control register 1 (WCR1) is a 32-bit readable/writable register that specifies the number of idle state insertion cycles for each area. With some kinds of memory, data bus drive does not go off immediately after the read signal from off-chip goes off. As a result, there is a possibility of a data bus collision when consecutive memory accesses are performed on memory in different areas, or when a memory write is performed immediately after a read. In the SH7750, the number of idle cycles set in the WCR1 register are inserted automatically if there is a possibility of this kind of data bus collision.

WCR1 is initialized to H'77777777 by a power-on reset, but is not initialized by a manual reset or in standby mode.

Bit:	31	30	29	28	27	26	25	24
Bit name:	—	DMAIW2	DMAIW1	DMAIW0	—	A6IW2	A6IW1	A6IW0
Initial value:	0	1	1	1	0	1	1	1
R/W:	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
Bit name:	—	A5IW2	A5IW1	A5IW0	—	A4IW2	A4IW1	A4IW0
Initial value:	0	1	1	1	0	1	1	1
R/W:	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
Bit name:	—	A3IW2	A3IW1	A3IW0	—	A2IW2	A2IW1	A2IW0
Initial value:	0	1	1	1	0	1	1	1
R/W:	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	A1IW2	A1IW1	A1IW0	—	A0IW2	A0IW1	A0IW0
Initial value:	0	1	1	1	0	1	1	1
R/W:	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bits 31, 27, 23, 19, 15, 11, 7, and 3—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 30 to 28— DMAIW-DACK Device Inter-Cycle Idle Specification (DMAIW2–DMAIW0): These bits specify the number of idle cycles between bus cycles to be inserted when switching from a DACK device to another space, or from a read access to a write access on the same device. The DMAIW bits are valid only for DMA single address transfer; with DMA dual address transfer, inter-area idle cycles are inserted.

Bits 4n + 2 to 4n—Area n (6 to 0) Inter-Cycle Idle Specification (AnlW2–AnlW0): These bits specify the number of idle cycles between bus cycles to be inserted when switching from external space area n (n = 6 to 0) to another space, or from a read access to a write access in the same space.

DMAIW2/AnlW2	DMAIW1/AnlW1	DMAIW0/AnlW0	Inserted Idle Cycles	
0	0	0	0	
		1	1	
	1	0	2	
		1	3	
1	0	0	6	
		1	9	
	1	0	12	
		1	15	(Initial value)

- Idle Insertion between Accesses

Preceding Cycle	Following Cycle								Same Area	Different Area
	Same Area				Different Area					
	Read		Write		Read		Write		MPX Address Output	MPX Address Output
	CPU	DMA	CPU	DMA	CPU	DMA	CPU	DMA		
Read			M	M	M	M	M	M	M (1)	M (1)
Write					M	M	M	M		M (1)
DMA read (memory → device)			M	M	M	M	M	M	—	M (1)
DMA write (device → memory)	D	D	D	D*	D	D	D	D	—	D (1)

M, D:	WCR1 wait insertion (One cycle inserted in MPX access even if WCR1 is cleared to 0)
M:	Memory setting (area 0 to area 6)
D:	DMA setting
*:	No insertion in consecutive accesses to same device

Note: When synchronous DRAM is used in RAS down mode, set bits DMAIW2–DMAIW0 to 000 and bits A3IW2–A3IW0 to 000.

13.2.4 Wait Control Register 2 (WCR2)

Wait control register 2 (WCR2) is a 32-bit readable/writable register that specifies the number of wait state insertion cycles for each area. It also specifies the data access pitch when performing burst memory access. This enables low-speed memory to be directly connected without using external circuitry.

WCR2 is initialized to H'FFFEEFFF by a power-on reset, but is not initialized by a manual reset or in standby mode.

Bit:	31	30	29	28	27	26	25	24
Bit name:	A6W2	A6W1	A6W0	A6B2	A6B1	A6B0	A5W2	A5W1
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:	A5W0	A5B2	A5B1	A5B0	A4W2	A4W1	A4W0	—
Initial value:	1	1	1	1	1	1	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R

Bit:	15	14	13	12	11	10	9	8
Bit name:	A3W2	A3W1	A3W0	—	A2W2	A2W1	A2W0	A1W2
Initial value:	1	1	1	0	1	1	1	1
R/W:	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	A1W1	A1W0	A0W2	A0W1	A0W0	A0B2	A0B1	A0B0
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 31 to 29—Area 6 Wait Control (A6W2—A6W0): These bits specify the number of wait states to be inserted for external space area 6.

			Description	
			First Cycle	
Bit 31: A6W2	Bit 30: A6W1	Bit 29: A6W0	Inserted Wait States	$\overline{\text{RDY}}$ Pin
0	0	0	0	Ignored
		1	1	Enabled
	1	0	2	Enabled
		1	3	Enabled
1	0	0	6	Enabled
		1	9	Enabled
	1	0	12	Enabled
		1	15 (Initial value)	Enabled

Bits 28 to 26—Area 6 Burst Pitch (A6B2–A6B0): These bits specify the burst pitch in a burst transfer.

			Description	
			Burst Cycle (Excluding First Cycle)	
Bit 28: A6B2	Bit 27: A6B1	Bit 26: A6B0	States Per Data Transfer	$\overline{\text{RDY}}$ Pin
0	0	0	0	Ignored
		1	1	Enabled
	1	0	2	Enabled
		1	3	Enabled
1	0	0	4	Enabled
		1	5	Enabled
	1	0	6	Enabled
		1	7 (Initial value)	Enabled

Bits 25 to 23—Area 5 Wait Control (A5W2–A5W0): These bits specify the number of wait states to be inserted for external space area 5.

			Description	
			First Cycle	
Bit 25: A5W2	Bit 24: A5W1	Bit 23: A5W0	Inserted Wait States	$\overline{\text{RDY}}$ Pin
0	0	0	0	Ignored
		1	1	Enabled
	1	0	2	Enabled
		1	3	Enabled
1	0	0	6	Enabled
		1	9	Enabled
	1	0	12	Enabled
		1	15 (Initial value)	Enabled

Bits 22 to 20—Area 5 Burst Pitch (A5B2–A5B0): These bits specify the burst pitch in a burst transfer.

			Description	
			Burst Cycle (Excluding First Cycle)	
Bit 22: A5B2	Bit 21: A5B1	Bit 20: A5B0	Burst Pitch Per Data Transfer	$\overline{\text{RDY}}$ Pin
0	0	0	0	Ignored
		1	1	Enabled
	1	0	2	Enabled
		1	3	Enabled
1	0	0	4	Enabled
		1	5	Enabled
	1	0	6	Enabled
		1	7 (Initial value)	Enabled

Bits 19 to 17—Area 4 Wait Control (A4W2–A4W0): These bits specify the number of wait states to be inserted for external space area 4.

			Description	
Bit 19: A4W2	**Bit 18: A4W1**	**Bit 17: A4W0**	**Inserted Wait States**	**$\overline{\text{RDY}}$ Pin**
0	0	0	0	Ignored
		1	1	Enabled
	1	0	2	Enabled
		1	3	Enabled
1	0	0	6	Enabled
		1	9	Enabled
	1	0	12	Enabled
		1	15 (Initial value)	Enabled

Bits 16 and 12—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 15 to 13—Area 3 Wait Control (A3W2–A3W0): These bits specify the number of wait states to be inserted for external space area 3. External wait input is only enabled when normal memory is used, and is ignored when DRAM or synchronous DRAM is used.

- When Normal Memory is Used

			Description	
Bit 15: A3W2	**Bit 14: A3W1**	**Bit 13: A3W0**	**Inserted Wait States**	**$\overline{\text{RDY}}$ Pin**
0	0	0	0	Ignored
		1	1	Enabled
	1	0	2	Enabled
		1	3	Enabled
1	0	0	6	Enabled
		1	9	Enabled
	1	0	12	Enabled
		1	15 (Initial value)	Enabled

- When DRAM or Synchronous DRAM is Used*[1]

			Description	
Bit 15: A3W2	**Bit 14: A3W1**	**Bit 13: A3W0**	**DRAM $\overline{\text{CAS}}$ Assertion Width**	**Synchronous DRAM $\overline{\text{CAS}}$ Latency Cycles**
0	0	0	1	Inhibited
		1	2	1*[2]
	1	0	3	2
		1	4	3
1	0	0	7	4*[2]
		1	10	5*[2]
	1	0	13	Inhibited
		1	16	Inhibited

Notes: 1. External wait input is always ignored.
2. Inhibited in RAS down mode.

Bits 11 to 9—Area 2 Wait Control (A2W2–A2W0): These bits specify the number of wait states to be inserted for external space area 2. External wait input is only enabled when normal memory is used, and is ignored when DRAM or synchronous DRAM is used.

- When Normal Memory is Used

			Description	
Bit 11: A2W2	**Bit 10: A2W1**	**Bit 9: A2W0**	**Inserted Wait States**	**$\overline{\text{RDY}}$ Pin**
0	0	0	0	Ignored
		1	1	Enabled
	1	0	2	Enabled
		1	3	Enabled
1	0	0	6	Enabled
		1	9	Enabled
	1	0	12	Enabled
		1	15 (Initial value)	Enabled

- When DRAM or Synchronous DRAM is Used*

			Description	
Bit 11: A2W2	**Bit 10: A2W1**	**Bit 9: A2W0**	**DRAM $\overline{\text{CAS}}$ Assertion Width**	**Synchronous DRAM $\overline{\text{CAS}}$ Latency Cycles**
0	0	0	1	Inhibited
		1	2	1
	1	0	3	2
		1	4	3
1	0	0	7	4
		1	10	5
	1	0	13	Inhibited
		1	16	Inhibited

Note: * External wait input is always ignored.

Bits 8 to 6—Area 1 Wait Control (A1W2–A1W0): These bits specify the number of wait states to be inserted for external space area 1.

			Description	
Bit 8: A1W2	**Bit 7: A1W1**	**Bit 6: A1W0**	**Inserted Wait States**	**$\overline{\text{RDY}}$ Pin**
0	0	0	0	Ignored
		1	1	Enabled
	1	0	2	Enabled
		1	3	Enabled
1	0	0	6	Enabled
		1	9	Enabled
	1	0	12	Enabled
		1	15 (Initial value)	Enabled

Bits 5 to 3—Area 0 Wait Control (A0W2 to A0W0): These bits specify the number of wait states to be inserted for external space area 0.

			Description	
			First Cycle	
Bit 5: A0W2	Bit 4: A0W1	Bit 3: A0W0	Inserted Wait States	$\overline{\text{RDY}}$ Pin
0	0	0	0	Ignored
		1	1	Enabled
	1	0	2	Enabled
		1	3	Enabled
1	0	0	6	Enabled
		1	9	Enabled
	1	0	12	Enabled
		1	15 (Initial value)	Enabled

Bits 2 to 0—Area 0 Burst Pitch (A0B2–A0B0): These bits specify the burst pitch in a burst transfer.

			Description	
			Burst Cycle (Excluding First Cycle)	
Bit 2: A0B2	Bit 1: A0B1	Bit 0: A0B0	Burst Pitch Per Data Transfer	$\overline{\text{RDY}}$ Pin
0	0	0	0	Ignored
		1	1	Enabled
	1	0	2	Enabled
		1	3	Enabled
1	0	0	4	Enabled
		1	5	Enabled
	1	0	6	Enabled
		1	7 (Initial value)	Enabled

• When MPX is Used (Areas 0 to 6)

Bit 4n + 2: AnW2	Bit 4n + 1: AnW1	Bit 4n: AnW0	Description			
			Inserted Wait States			$\overline{\text{RDY}}$ Pin
			1st Data		2nd Data Onward	
			Read	Write		
0	0	0	1	0	0	Enabled
		1		1		Enabled
	1	0	2	2		Enabled
		1	3	3		Enabled
1	0	0	1	0	1	Enabled
		1		1		Enabled
	1	0	2	2		Enabled
		1	3	3		Enabled

(n = 6 to 0)

13.2.5 Wait Control Register 3 (WCR3)

Wait control register 3 (WCR3) is a 32-bit readable/writable register that specifies the cycles inserted in the setup time from the address until assertion of the write strobe, and the data hold time from negation of the strobe, for each area. This enables low-speed memory to be directly connected without using external circuitry.

WCR3 is initialized to H'07777777 by a power-on reset, but is not initialized by a manual reset or in standby mode.

Bit:	31	30	29	28	27	26	25	24
Bit name:	—	—	—	—	—	A6S0	A6H1	A6H0
Initial value:	0	0	0	0	0	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
Bit name:	—	A5S0	A5H1	A5H0	—	A4S0	A4H1	A4H0
Initial value:	0	1	1	1	0	1	1	1
R/W:	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
Bit name:	—	A3S0	A3H1	A3H0	—	A2S0	A2H1	A2H0
Initial value:	0	1	1	1	0	1	1	1
R/W:	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
Bit name:	—	A1S0	A1H1	A0H0	—	A0S0	A0H1	A0H0
Initial value:	0	1	1	1	0	1	1	1
R/W:	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bits 31 to 27, 23, 19, 15, 11, 7, and 3—Reserved: These bits are always read as 0, and should only be written with 0.

Valid only for normal memory and burst ROM:

Bit 4n + 2—Area n (6 to 0) Write Strobe Setup Time (AnS0): Specifies the number of cycles inserted in the setup time from the address until assertion of the read/write strobe.

Bit 4n + 2: AnS0	Waits Inserted in Setup	
0	0	
1	1	(Initial value)

(n = 6 to 0)

Valid only for normal memory and burst ROM:

Bits 4n + 1 and 4n—Area n (6 to 0) Data Hold Time (AnH1, AnH0): When writing, these bits specify the number of cycles to be inserted in the hold time from negation of the write strobe. When reading, they specify the number of cycles to be inserted in the hold time from the data sampling timing.

Bit 4n + 1: AnH1	Bit 4n: AnH0	Waits Inserted in Hold	
0	0	0	
	1	1	
1	0	2	
	1	3	(Initial value)

(n = 6 to 0)

13.2.6 Memory Control Register (MCR)

The memory control register (MCR) is a 32-bit readable/writable register that specifies $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ timing and burst control for DRAM and synchronous DRAM (areas 2 and 3), address multiplexing, and refresh control. This enables DRAM and synchronous DRAM to be directly connected without using external circuitry.

MCR is initialized to H'00000000 by a power-on reset, but is not initialized by a manual reset or in standby mode. Bits RASD, MRSET, TRC2–0, TPC2–0, RCD1–0, TRWL2–0, TRAS2–0, BE, SZ1–0, AMXEXT, AMX2–0, and EDOMODE are written in the initialization following a power-on reset, and should not be modified subsequently. When writing to bits RFSH and RMODE, the same values should be written to the other bits so that they remain unchanged. When using DRAM or synchronous DRAM, areas 2 and 3 should not be accessed until register initialization is completed.

Bit:	31	30	29	28	27	26	25	24
Bit name:	RASD	MRSET	TRC2	TRC1	TRC0	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R	R	R

Bit:	23	22	21	20	19	18	17	16
Bit name:	TCAS	—	TPC2	TPC1	TPC0	—	RCD1	RCD0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R/W	R/W	R/W	R	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
Bit name:	TRWL2	TRWL1	TRWL0	TRAS2	TRAS1	TRAS0	BE	SZ1
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:	SZ0	AMXEXT	AMX2	AMX1	AMX0	RFSH	RMODE	EDO MODE
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 31—RAS Down (RASD): Sets RAS down mode. When RAS down mode is used, set BE to 1. Do not set RAS down mode in slave mode or partial-sharing mode, or when areas 2 and 3 are both designated as synchronous DRAM space.

Bit 31: RASD	Description	
0	Normal mode	(Initial value)
1	RAS down mode	

Note: When synchronous DRAM is used in RAS down mode, set bits DMAIW2–DMAIW0 to 000 and bits A3IW2–A3IW0 to 000.

Bit 30—Mode Register Set (MRSET): Set when a synchronous DRAM mode register setting is used. See Power-On Sequence in section 13.3.5, Synchronous DRAM Interface.

Bit 30: MRSET	Description	
0	All-bank precharge	(Initial value)
1	Mode register setting	

Bits 26 to 24, 22, and 18—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 29 to 27—RAS Precharge Time at End of Refresh (TRC2–TRC0)
(Synchronous DRAM: auto- and self-refresh both enabled; DRAM: auto- and self-refresh both enabled)

Bit 29: TRC2	Bit 28: TRC1	Bit 27: TRC0	RAS Precharge Time Immediately after Refresh	
0	0	0	0	(Initial value)
		1	3	
	1	0	6	
		1	9	
1	0	0	12	
		1	15	
	1	0	18	
		1	21	

Bit 23—CAS Negation Period (TCAS): This bit is valid only when DRAM is connected.

Bit 23: TCAS	CAS Negation Period	
0	1	(Initial value)
1	2	

Bits 21 to 19—RAS Precharge Period (TPC2–TPC0): When the DRAM interface is selected for the connected memory, these bits specify the minimum number of cycles until $\overline{\text{RAS}}$ is asserted again after being negated. When the synchronous DRAM interface is selected, these bits specify the minimum number of cycles until the next bank active command is output after precharging.

			RAS Precharge Time	
Bit 21: TPC2	**Bit 20: TPC1**	**Bit 19: TPC0**	**DRAM**	**Synchronous DRAM**
0	0	0	0	1* (Initial value)
		1	1	2
	1	0	2	3
		1	3	4*
1	0	0	4	5*
		1	5	6*
	1	0	6	7*
		1	7	8*

Note: * Inhibited in RAS down mode.

Bits 17 and 16—RAS-CAS Delay (RCD1, RCD0): When the DRAM interface is selected for the connected memory, these bits set the $\overline{\text{RAS}}$-$\overline{\text{CAS}}$ assertion delay time. When the synchronous DRAM interface is selected, these bits set the bank active-read/write command delay time.

		Description	
Bit 17: RCD1	**Bit 16: RCD0**	**DRAM**	**Synchronous DRAM**
0	0	2 cycles	Reserved (Setting prohibited)
	1	3 cycles	2 cycles
1	0	4 cycles	3 cycles
	1	5 cycles	4 cycles*

Note: * Inhibited in RAS down mode.

Bits 15 to 13—Write Precharge Delay (TRWL2–TRWL0): These bits set the synchronous DRAM write precharge delay time. In auto-precharge mode, they specify the time until the next bank active command is issued after a write cycle. After a write cycle, the next active command is not issued for a period of TPC + TRWL. In RAS down mode, they specify the time until the next precharge command is issued. After a write cycle, the next precharge command is not issued for a period of TRWL. This setting is valid only when synchronous DRAM is connected.

Bit 15: TRWL2	Bit 14: TRWL1	Bit 13: TRWL0	Write Precharge ACT Delay Time
0	0	0	1 (Initial value)
		1	2
	1	0	3*
		1	4*
1	0	0	5*
		1	Reserved (Setting prohibited)
	1	0	Reserved (Setting prohibited)
		1	Reserved (Setting prohibited)

Note: * Inhibited in RAS down mode.

Bits 12 to 10—CAS-Before-RAS Refresh $\overline{\text{RAS}}$ Assertion Period (TRAS2–TRAS0): When the DRAM interface is selected for the connected memory, these bits set the $\overline{\text{RAS}}$ assertion period in CAS-before-RAS refreshing. When the synchronous DRAM interface is selected, the bank active command is not issued for a period of TRC + TRAS after an auto-refresh command is issued.

Bit 12: TRAS2	Bit 11: TRAS1	Bit 10: TRAS0	$\overline{\text{RAS}}$/DRAM Assertion Period	Command Interval after Synchronous DRAM Refresh
0	0	0	2	4 + TRC (Initial value)
		1	3	5 + TRC
	1	0	4	6 + TRC
		1	5	7 + TRC
1	0	0	6	8 + TRC
		1	7	9 + TRC
	1	0	8	10 + TRC
		1	9	11 + TRC

Bit 9—Burst Enable (BE): Specifies whether burst access is performed on DRAM. In synchronous DRAM access, burst access is always performed regardless of the specification of this bit. The DRAM transfer mode depends on EDOMODE.

BE	EDOMODE	8/16/32/64-Bit Transfer	32-Byte Transfer
0	0	Single	Single
	1	Setting prohibited	Setting prohibited
1	0	Single/fast page*	Fast page
	1	EDO	EDO

Note: * In fast page mode, 32-bit or 64-bit transfer with a 16-bit bus, 64-bit transfer with a 32-bit bus.

Bits 8 and 7—Memory Data Size (SZ1, SZ0): These bits specify the memory data size of DRAM and synchronous DRAM. This setting has priority over the BCR2 register setting.

		Description	
Bit 8: SZ1	Bit 7: SZ0	DRAM	SDRAM
0	0	64 bits	64 bits
	1	Reserved (Setting prohibited)	Reserved (Setting prohibited)
1	0	16 bits	Reserved (Setting prohibited)
	1	32 bits	32 bits

Bits 6 to 3—Address Multiplexing (AMXEXT, AMX2–AMX0): These bits specify address multiplexing for DRAM and synchronous DRAM. The actual address shift value is different for the DRAM interface and the synchronous DRAM interface.

- For DRAM Interface:

Bit 6: AMXEXT	Bit 5: AMX2	Bit 4: AMX1	Bit 3: AMX0	Description
				DRAM
0*	0	0	0	8-bit column address product (Initial value)
			1	9-bit column address product
		1	0	10-bit column address product
			1	11-bit column address product
	1	0	0	12-bit column address product
			1	Reserved (Setting prohibited)
		1	0	Reserved (Setting prohibited)
			1	Reserved (Setting prohibited)

Note: * When the DRAM interface is used, clear the AMXEXT bit to 0.

- For Synchronous DRAM Interface:

AMX	AMXEXT	SZ	Synchronous DRAM	BANK
0	0	64	(16M: 512k × 16 bits × 2) × 4	a[22]*
		32	(16M: 512k × 16 bits × 2) × 2	a[21]*
	1	64	(16M: 512k × 16 bits × 2) × 4	a[21]*
		32	(16M: 512k × 16 bits × 2) × 2	a[20]*
1	0	64	(16M: 1M × 8 bits × 2) × 8	a[23]*
		32	(16M: 1M × 8 bits × 2) × 4	a[22]*
	1	64	(16M: 1M × 8 bits × 2) × 8	a[22]*
		32	(16M: 1M × 8 bits × 2) × 4	a[21]*
2	—	64	(64M: 1M × 16 bits × 4) × 4	a[24:23]*
		32	(64M: 1M × 16 bits × 4) × 2	a[23:22]*
3		64	(64M: 2M × 8 bits × 4) × 8	a[25:24]*
		32	(64M: 2M × 8 bits × 4) × 4	a[24:23]*
4		64	(64M: 512k × 32 bits × 4) × 2	a[23:22]*
		32	(64M: 512k × 32 bits × 4) × 1	a[22:21]*
5		64	(64M: 1M × 32 bits × 2) × 2	a[23]*
		32	(64M: 1M × 32 bits × 2) × 1	a[22]*
6		64	Reserved (Setting prohibited)	
		32	Reserved (Setting prohibited)	
7		64	(16M: 256k × 32 bits × 2) × 2	a[21]*
		32	(16M: 256k × 32 bits × 2) × 1	a[20]*

Note: * a[*]: Physical address

Bit 2—Refresh Control (RFSH): Specifies refresh control. Selects whether refreshing is performed for DRAM and synchronous DRAM. When the refresh function is not used, the refresh request cycle generation timer can be used as an interval timer.

Bit 2: RFSH	Description	
0	Refreshing is not performed	(Initial value)
1	Refreshing is performed	

Bit 1—Refresh Mode (RMODE): Specifies whether normal refreshing or self-refreshing is performed when the RFSH bit is set to 1. When the RFSH bit is 1 and this bit is cleared to 0, CAS-before-RAS refreshing or auto-refreshing is performed for DRAM and synchronous DRAM, using the cycle set by refresh-related registers RTCNT, RTCOR, and RTCSR. If a refresh request is issued during an external bus cycle, the refresh cycle is executed when the bus cycle ends. When the RFSH bit is 1 and this bit is set to 1, the self-refresh state is set for DRAM and synchronous DRAM, after waiting for the end of any currently executing external bus cycle. All refresh requests for memory in the self-refresh state are ignored.

Bit 1: RMODE	Description	
0	CAS-before-RAS refreshing is performed (when RFSH = 1)	(Initial value)
1	Self-refreshing is performed (when RFSH = 1)	

Bit 0—EDO Mode (EDOMODE): Used to specify the data sampling timing for data reads when using EDO mode DRAM. The setting of this bit does not affect the operation timing of memory other than DRAM. Set this bit to 1 only when DRAM is used.

13.2.7 PCMCIA Control Register (PCR)

The PCMCIA control register (PCR) is a 16-bit readable/writable register that specifies the $\overline{\text{OE}}$ and $\overline{\text{WE}}$ signal assertion/negation timing for the PCMCIA interface connected to areas 5 and 6. The $\overline{\text{OE}}$ and $\overline{\text{WE}}$ signal assertion width is set by the wait control bits in the WCR2 register.

PCR is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or in standby mode.

Bit:	15	14	13	12	11	10	9	8
Bit name:	A5PCW1	A5PCW0	A6PCW1	A6PCW0	A5TED2	A5TED1	A5TED0	A6TED2
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:	A6TED1	A6TED0	A5TEH2	A5TEH1	A5TEH0	A6TEH2	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

Bits 15 and 14—PCMCIA Wait (A5PCW1, A5PCW0): These bits specify the number of waits to be added to the number of waits specified by WCR2 in a low-speed PCMCIA wait cycle. The setting of these bits is selected when the TC bit is cleared to 0 in the page table entry assistance register (PTEA).

Bit 15: A5PCW1	Bit 14: A5PCW0	Waits Inserted
0	0	0 (Initial value)
	1	15
1	0	30
	1	50

Bits 13 and 12—PCMCIA Wait (A6PCW1, A6PCW0): These bits specify the number of waits to be added to the number of waits specified by WCR2 in a low-speed PCMCIA wait cycle. The setting of these bits is selected when the TC bit is set to 1 in the page table entry assistance register (PTEA).

Bit 13: A6PCW1	Bit 12: A6PCW0	Waits Inserted
0	0	0 (Initial value)
	1	15
1	0	30
	1	50

Bits 11 to 9—Address-$\overline{\text{OE}}$/$\overline{\text{WE}}$ Assertion Delay (A5TED2–A5TED0): These bits set the delay time from address output to $\overline{\text{OE}}$/$\overline{\text{WE}}$ assertion on the connected PCMCIA interface. The setting of these bits is selected when the TC bit is cleared to 0 in PTEA.

Bit 11: A5TED2	Bit 10: A5TED1	Bit 9: A5TED0	Waits Inserted
0	0	0	0 (Initial value)
		1	1
	1	0	2
		1	3
1	0	0	6
		1	9
	1	0	12
		1	15

Bits 8 to 6—Address-$\overline{\text{OE}}$/$\overline{\text{WE}}$ Assertion Delay (A6TED2–A6TED0): These bits set the delay time from address output to $\overline{\text{OE}}$/$\overline{\text{WE}}$ assertion on the connected PCMCIA interface. The setting of these bits is selected when the TC bit is set to 1 in PTEA.

Bit 8: A6TED2	Bit 7: A6TED1	Bit 6: A6TED0	Waits Inserted
0	0	0	0 (Initial value)
		1	1
	1	0	2
		1	3
1	0	0	6
		1	9
	1	0	12
		1	15

Bits 5 to 3—$\overline{\text{OE}}$/$\overline{\text{WE}}$ Negation-Address Delay (A5TEH2–A5TEH0): These bits set the address hold delay time from $\overline{\text{OE}}$/$\overline{\text{WE}}$ negation in a write on the connected PCMCIA interface or in an I/O card read. In the case of a memory card read, the address hold delay time from the data sampling timing is set.The setting of these bits is selected when the TC bit is cleared to 0 in PTEA.

Bit 5: A5TEH2	Bit 4: A5TEH1	Bit 3: A5TEH0	Waits Inserted
0	0	0	0 (Initial value)
		1	1
	1	0	2
		1	3
1	0	0	6
		1	9
	1	0	12
		1	15

Bits 2 to 0—$\overline{\text{OE}}/\overline{\text{WE}}$ Negation-Address Delay (A6TEH2–A6TEH0): These bits set the address hold delay time from $\overline{\text{OE}}/\overline{\text{WE}}$ negation in a write on the connected PCMCIA interface or in an I/O card read. In the case of a memory card read, the address hold delay time from the data sampling timing is set. The setting of these bits is selected when the TC bit is set to 1 in PTEA.

Bit 2: A6TEH2	Bit 1: A6TEH1	Bit 0: A6TEH0	Waits Inserted
0	0	0	0 (Initial value)
		1	1
	1	0	2
		1	3
1	0	0	6
		1	9
	1	0	12
		1	15

13.2.8 Synchronous DRAM Mode Register (SDMR)

The synchronous DRAM mode register (SDMR) is a write-only virtual 16-bit register that is written to via the synchronous DRAM address bus, and sets the mode of the area 2 and area 3 synchronous DRAM.

Settings for the SDMR register must be made before accessing synchronous DRAM.

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

Since the address bus, not the data bus, is used to write to the synchronous DRAM mode register, if the value to be set is “X” and the SDMR register address is “Y”, value “X” is written to the synchronous DRAM mode register by performing a write to address X + Y. When the synchronous DRAM bus width is set to 32 bits, as A0 of the synchronous DRAM is connected to A2 of the SH7750, and A1 of the synchronous DRAM is connected to A3 of the SH7750, the value actually written to the synchronous DRAM is the value of “X” shifted 2 bits to the right.

For example, to write H'0230 to the area 2 SDMR register, arbitrary data is written to address H'FF900000 (address “Y”) + H'08C0 (value “X”) (= H'FF9008C0). As a result, H'0230 is written to the SDMR register. The range of value “X” is H'0000 to H'0FFC.

Similarly, to write H'0230 to the area 3 SDMR register, arbitrary data is written to address H'FF940000 (address “Y”) + H'08C0 (value “X”) (= H'FF9408C0). As a result, H'0230 is written to the SDMR register. The range of value “X” is H'0000 to H'0FFC.

The lower 16 bits of the address are set in the synchronous DRAM mode register.

For a 32-bit bus:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Address					0	0	0	LMODE2	LMODE1	LMODE0	WT	BL2	BL1	BL0		

10 bits set in case of 32-bit bus width

For a 64-bit bus:

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Address				0	0	0	LMODE2	LMODE1	LMODE0	WT	BL2	BL1	BL0			

10 bits set in case of 64-bit bus width

LMODE:	RAS-CAS latency
BL:	Burst length
WT:	Wrap type (0: Sequential)

BL	LMODE
000: Reserved	000: Reserved
001: Reserved	001: 1
010: 4	010: 2
011: 8	011: 3
100: Reserved	100: Reserved
101: Reserved	101: Reserved
110: Reserved	110: Reserved
111: Reserved	111: Reserved

13.2.9 Refresh Timer Control/Status Register (RTSCR)

The refresh timer control/status register (RTSCR) is a 16-bit readable/writable register that specifies the refresh cycle and whether interrupts are to be generated.

RTSCR is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or 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:	—	—	—	—	—	—	—	—

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 to 8—Reserved: These bits are always read as 0. For the write values, see section 13.2.13, Notes on Accessing Refresh Control Registers.

Bit 7—Compare-Match Flag (CMF): Status flag that indicates a match between the refresh timer counter (RTCNT) and refresh time constant register (RTCOR) values.

Bit 7: CMF	Description
0	RTCNT and RTCOR values do not match (Initial value) [Clearing condition] When 0 is written to CMF
1	RTCNT and RTCOR values match [Setting condition] When RTCNT = RTCOR*

Note: * If 1 is written, the original value is retained.

Bit 6—Compare-Match Interrupt Enable (CMIE): Controls generation or suppression of an interrupt request when the CMF flag is set to 1 in RTCSR. Do not set this bit to 1 when CAS-before-RAS refreshing or auto-refreshing is used.

Bit 6: CMIE	Description	
0	Interrupt requests initiated by CMF are disabled	(Initial value)
1	Interrupt requests initiated by CMF are enabled	

Bits 5 to 3—Clock Select Bits (CKS2–CKS0): These bits select the input clock for RTCNT. The base clock is the external bus clock (CKIO). The RTCNT count clock is obtained by scaling CKIO by the specified factor.

Bit 5: CKS2	Bit 4: CKS1	Bit 3: CKS0	Description	
0	0	0	Clock input disabled	(Initial value)
		1	Bus clock (CKIO)/4	
	1	0	CKIO/16	
		1	CKIO/64	
1	0	0	CKIO/256	
		1	CKIO/1024	
	1	0	CKIO/2048	
		1	CKIO/4096	

Bit 2—Refresh Count Overflow Flag (OVF): Status flag that indicates that the number of refresh requests indicated by the refresh count register (RFCR) has exceeded the number specified by the LMTS bit in RTCSR.

Bit 2: OVF	Description	
0	RFCR has not overflowed the count limit indicated by LMTS [Clearing condition] When 0 is written to OVF	(Initial value)
1	RFCR has overflowed the count limit indicated by LMTS [Setting condition] When RFCR overflows the count limit set by LMTS*	

Note: * If 1 is written, the original value is retained.

Bit 1—Refresh Count Overflow Interrupt Enable (OVIE): Controls generation or suppression of an interrupt request when the OVF flag is set to 1 in RTCSR.

Bit 1: OVIE	Description	
0	Interrupt requests initiated by OVF are disabled	(Initial value)
1	Interrupt requests initiated by OVF are enabled	

Bit 0—Refresh Count Overflow Limit Select (LMTS): Specifies the count limit to be compared with the refresh count indicated by the refresh count register (RFCR). If the RFCR register value exceeds the value specified by LMTS, the OVF flag is set.

Bit 0: LMTS	Description	
0	Count limit is 1024	(Initial value)
1	Count limit is 512	

13.2.10 Refresh Timer Counter (RTCNT)

The refresh timer counter (RTCNT) is an 8-bit readable/writable counter that is incremented by the input clock (selected by bits CKS2–CKS0 in the RTCSR register). When the RTCNT counter value matches the RTCOR register value, the CMF bit is set in the RTCSR register and the RTCNT counter is cleared.

RTCNT is initialized to H'0000 by a power-on reset, but continues to count when a manual reset is performed. In standby mode, RTCNT is not initialized, and retains its contents.

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

13.2.11 Refresh Time Constant Register (RTCOR)

The refresh time constant register (RTCOR) is a readable/writable register that specifies the upper limit of the RTCNT counter. The RTCOR register and RTCNT counter values (lower 8 bits) are constantly compared, and when they match the CMF bit is set in the RTCSR register and the RTCNT counter is cleared to 0. If the refresh bit (RFSH) has been set to 1 in the memory control register (MCR) and CAS-before-RAS has been selected as the refresh mode, a memory refresh cycle is generated when the CMF bit is set.

RTCOR is initialized to H'0000 by a power-on reset, but is not initialized, and retains its contents, in a manual reset and 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:	—	—	—	—	—	—	—	—

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 Count Register (RFCR)

The refresh count register (RFCR) is a 10-bit readable/writable counter that counts the number of refreshes by being incremented each time the RTCOR register and RTCNT counter values match. If the RFCR register value exceeds the count limit specified by the LMTS bit in the RTCSR register, the OVF flag is set in the RTCSR register and the RFCR register is cleared.

RFCR is initialized to H'0000 by a power-on reset, but is not initialized, and retains its contents, in a manual reset and 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

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 Notes on Accessing Refresh Control Registers

When the refresh timer control/status register (RTCSR), refresh timer counter (RTCNT), refresh time constant register (RTCOR), and refresh count register (RFCR) are written to, a special code is added to the data to prevent inadvertent rewriting in the event of program runaway, etc. The following procedures should be used for read/write operations.

Writing to RTCSR, RTCNT, RTCOR, and RFCR: A word transfer instruction must always be used when writing to RTCSR, RTCNT, RTCOR, or RFCR. A write cannot be performed with a byte transfer instruction.

When writing to RTCSR, RTCNT, or RTCOR, set B'10100101 in the upper byte and the write data in the lower byte, as shown in figure 13.4. When writing to RFCR, set B'101001 in the 6 bits starting from the MSB in the upper byte, and the write data in the remaining bits.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCSR, RTCNT, RTCOR	1	0	1	0	0	1	0	1	Write data							

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RFCR	1	0	1	0	0	1	Write data									

Figure 13.4 Writing to RTCSR, RTCNT, RTCOR, and RFCR

Reading RTCSR, RTCNT, RTCOR, and RFCR: A 16-bit access must always be used when reading RTCSR, RTCNT, RTCOR, or RFCR. Undefined bits are read as 0.

13.3 Operation

13.3.1 Endian/Access Size and Data Alignment

The SH7750 supports both big-endian mode, in which the most significant byte (MSByte) is at the 0 address end in a string of byte data, and little-endian mode, in which the least significant byte (LSByte) is at the 0 address end. The mode is set by means of the MD5 external pin in a power-on reset, big-endian mode being set if the MD5 pin is low, and little-endian mode if it is high.

A data bus width of 8, 16, 32, or 64 bits can be selected for normal memory, 16, 32, or 64 bits for DRAM, 32 or 64 bits for synchronous DRAM, and 8 or 16 bits for the PCMCIA interface. Data alignment is carried out according to the data bus width and endian mode of each device. Thus, four read operations are needed to read longword data from an 8-bit device. In the SH7750, data alignment and data length conversion between the different interfaces is performed automatically.

The relationship between the endian mode, device data length, and access unit, is shown in tables 13.6 to 13.13.

Table 13.6 (1) 64-Bit External Device/Big-Endian Access and Data Alignment

		Data Bus							
Operation	**No.**	**D63–56**	**D55–48**	**D47–40**	**D39–32**	**D31–24**	**D23–16**	**D15–8**	**D7–0**
Byte, Adr=8n	1	Data 7–0	—	—	—	—	—	—	—
Byte, Adr=8n+1	1	—	Data 7–0	—	—	—	—	—	—
Byte, Adr=8n+2	1	—	—	Data 7–0	—	—	—	—	—
Byte, Adr=8n+3	1	—	—	—	Data 7–0	—	—	—	—
Byte, Adr=8n+4	1	—	—	—	—	Data 7–0	—	—	—
Byte, Adr=8n+5	1	—	—	—	—	—	Data 7–0	—	—
Byte, Adr=8n+6	1	—	—	—	—	—	—	Data 7–0	—
Byte, Adr=8n+7	1	—	—	—	—	—	—	—	Data 7–0
Word, Adr=8n	1	Data 15–8	Data 7–0	—	—	—	—	—	—
Word, Adr=8n+2	1	—	—	Data 15–8	Data 7–0	—	—	—	—
Word, Adr=8n+4	1	—	—	—	—	Data 15–8	Data 7–0	—	—
Word, Adr=8n+6	1	—	—	—	—	—	—	Data 15–8	Data 7–0
Longword, Adr=8n	1	Data 31–24	Data 23–16	Data 15–8	Data 7–0	—	—	—	—
Longword, Adr=8n+4	1	—	—	—	—	Data 31–24	Data 23–16	Data 15–8	Data 7–0
Quadword, Adr=8n	1	Data 63–56	Data 55–48	Data 47–40	Data 39–32	Data 31–24	Data 23–16	Data 15–8	Data 7–0

Table 13.6 (2) 64-Bit External Device/Big-Endian Access and Data Alignment

		Strobe Signals							
Operation	**No.**	$\overline{\text{WE7}}$, $\overline{\text{CAS7}}$, **DQM7**	$\overline{\text{WE6}}$, $\overline{\text{CAS6}}$, **DQM6**	$\overline{\text{WE5}}$, $\overline{\text{CAS5}}$, **DQM5**	$\overline{\text{WE4}}$, $\overline{\text{CAS4}}$, **DQM4**	$\overline{\text{WE3}}$, $\overline{\text{CAS3}}$, **DQM3**	$\overline{\text{WE2}}$, $\overline{\text{CAS2}}$, **DQM2**	$\overline{\text{WE1}}$, $\overline{\text{CAS1}}$, **DQM1**	$\overline{\text{WE0}}$, $\overline{\text{CAS0}}$, **DQM0**
Byte, Adr=8n	1	Asserted							
Byte, Adr=8n+1	1		Asserted						
Byte, Adr=8n+2	1			Asserted					
Byte, Adr=8n+3	1				Asserted				
Byte, Adr=8n+4	1					Asserted			
Byte, Adr=8n+5	1						Asserted		
Byte, Adr=8n+6	1							Asserted	
Byte, Adr=8n+7	1								Asserted
Word, Adr=8n	1	Asserted	Asserted						
Word, Adr=8n+2	1			Asserted	Asserted				
Word, Adr=8n+4	1					Asserted	Asserted		
Word, Adr=8n+6	1							Asserted	Asserted
Longword, Adr=8n	1	Asserted	Asserted	Asserted	Asserted				
Longword, Adr=8n+4	1					Asserted	Asserted	Asserted	Asserted
Quadword, Adr=8n	1	Asserted	Asserted	Asserted	Asserted	Asserted	Asserted	Asserted	Asserted

Table 13.7 32-Bit External Device/Big-Endian Access and Data Alignment

		Data Bus				Strobe Signals			
Operation	**No.**	**D31–D24**	**D23–D16**	**D15–D8**	**D7–D0**	$\overline{\text{WE3}}$, $\overline{\text{CAS3}}$, DQM3	$\overline{\text{WE2}}$, $\overline{\text{CAS2}}$, DQM2	$\overline{\text{WE1}}$, $\overline{\text{CAS1}}$, DQM1	$\overline{\text{WE0}}$, $\overline{\text{CAS0}}$, DQM0
Byte, Adr=4n	1	Data 7–0	—	—	—	Asserted			
Byte, Adr=4n+1	1	—	Data 7–0	—	—		Asserted		
Byte, Adr=4n+2	1	—	—	Data 7–0	—			Asserted	
Byte, Adr=4n+3	1	—	—	—	Data 7–0				Asserted
Word, Adr=4n	1	Data 15–8	Data 7–0	—	—	Asserted	Asserted		
Word, Adr=4n+2	1	—	—	Data 15–8	Data 7–0			Asserted	Asserted
Longword, Adr=4n	1	Data 31–24	Data 23–16	Data 15–8	Data 7–0	Asserted	Asserted	Asserted	Asserted
Quadword	1	Data 63–56	Data 55–48	Data 47–40	Data 39–32	Asserted	Asserted	Asserted	Asserted
	2	Data 31–24	Data 23–16	Data 15–8	Data 7–0	Asserted	Asserted	Asserted	Asserted

Table 13.8 16-Bit External Device/Big-Endian Access and Data Alignment

		Data Bus				Strobe Signals			
Operation	**No.**	**D31–D24**	**D23–D16**	**D15–D8**	**D7–D0**	$\overline{\text{WE3}}$, $\overline{\text{CAS3}}$, DQM3	$\overline{\text{WE2}}$, $\overline{\text{CAS2}}$, DQM2	$\overline{\text{WE1}}$, $\overline{\text{CAS1}}$, DQM1	$\overline{\text{WE0}}$, $\overline{\text{CAS0}}$, DQM0
Byte, Adr=2n	1	—	—	Data 7–0	—			Asserted	
Byte, Adr=2n+1	1	—	—	—	Data 7–0				Asserted
Word	1	—	—	Data 15–8	Data 7–0			Asserted	Asserted
Longword	1	—	—	Data 31–24	Data 23–16			Asserted	Asserted
	2	—	—	Data 15–8	Data 7–0			Asserted	Asserted
Quadword	1	—	—	Data 63–56	Data 55–48			Asserted	Asserted
	2	—	—	Data 47–40	Data 39–32			Asserted	Asserted
	3	—	—	Data 31–24	Data 23–16			Asserted	Asserted
	4	—	—	Data 15–8	Data 7–0			Asserted	Asserted

Table 13.9 8-Bit External Device/Big-Endian Access and Data Alignment

		Data Bus				Strobe Signals			
Operation	**No.**	**D31–D24**	**D23–D16**	**D15–D8**	**D7–D0**	**$\overline{\text{WE3}}$, $\overline{\text{CAS3}}$, DQM3**	**$\overline{\text{WE2}}$, $\overline{\text{CAS2}}$, DQM2**	**$\overline{\text{WE1}}$, $\overline{\text{CAS1}}$, DQM1**	**$\overline{\text{WE0}}$, $\overline{\text{CAS0}}$, DQM0**
Byte	1	—	—	—	Data 7–0				Asserted
Word	1	—	—	—	Data 15–8				Asserted
	2	—	—	—	Data 7–0				Asserted
Longword	1	—	—	—	Data 31–24				Asserted
	2	—	—	—	Data 23–16				Asserted
	3	—	—	—	Data 15–8				Asserted
	4	—	—	—	Data 7–0				Asserted
Quadword	1	—	—	—	Data 63–56				Asserted
	2	—	—	—	Data 55–48				Asserted
	3	—	—	—	Data 47–40				Asserted
	4	—	—	—	Data 39–32				Asserted
	5	—	—	—	Data 31–24				Asserted
	6	—	—	—	Data 23–16				Asserted
	7	—	—	—	Data 15–8				Asserted
	8	—	—	—	Data 7–0				Asserted

Table 13.10 (1) 64-Bit External Device/Little-Endian Access and Data Alignment

		Data Bus							
Operation	**No.**	**D63–56**	**D55–48**	**D47–40**	**D39–32**	**D31–24**	**D23–16**	**D15–8**	**D7–0**
Byte, Adr=8n	1	—	—	—	—	—	—	—	Data 7–0
Byte, Adr=8n+1	1	—	—	—	—	—	—	Data 7–0	—
Byte, Adr=8n+2	1	—	—	—	—	—	Data 7–0	—	—
Byte, Adr=8n+3	1	—	—	—	—	Data 7–0	—	—	—
Byte, Adr=8n+4	1	—	—	—	Data 7–0	—	—	—	—
Byte, Adr=8n+5	1	—	—	Data 7–0	—	—	—	—	—
Byte, Adr=8n+6	1	—	Data 7–0	—	—	—	—	—	—
Byte, Adr=8n+7	1	Data 7–0	—	—	—	—	—	—	—
Word, Adr=8n	1	—	—	—	—	—	—	Data 15–8	Data 7–0
Word, Adr=8n+2	1	—	—			Data 15–8	Data 7–0	—	—
Word, Adr=8n+4	1	—	—	Data 15–8	Data 7–0	—	—	—	—
Word, Adr=8n+6	1	Data 15–8	Data 7–0	—	—	—	—	—	—
Longword, Adr=8n	1	—	—	—	—	Data 31–24	Data 23–16	Data 15–8	Data 7–0
Longword, Adr=8n+4	1	Data 31–24	Data 23–16	Data 15–8	Data 7–0	—	—	—	—
Quadword, Adr=8n	1	Data 63–56	Data 55–48	Data 47–40	Data 39–32	Data 31–24	Data 23–16	Data 15–8	Data 7–0

Table 13.10 (2) 64-Bit External Device/Little-Endian Access and Data Alignment

		Strobe Signals							
Operation	**No.**	**$\overline{\text{WE7}}$, $\overline{\text{CAS7}}$, DQM7**	**$\overline{\text{WE6}}$, $\overline{\text{CAS6}}$, DQM6**	**$\overline{\text{WE5}}$, $\overline{\text{CAS5}}$, DQM5**	**$\overline{\text{WE4}}$, $\overline{\text{CAS4}}$, DQM4**	**$\overline{\text{WE3}}$, $\overline{\text{CAS3}}$, DQM3**	**$\overline{\text{WE2}}$, $\overline{\text{CAS2}}$, DQM2**	**$\overline{\text{WE1}}$, $\overline{\text{CAS1}}$, DQM1**	**$\overline{\text{WE0}}$, $\overline{\text{CAS0}}$, DQM0**
Byte, Adr=8n	1								Asserted
Byte, Adr=8n+1	1							Asserted	
Byte, Adr=8n+2	1						Asserted		
Byte, Adr=8n+3	1					Asserted			
Byte, Adr=8n+4	1				Asserted				
Byte, Adr=8n+5	1			Asserted					
Byte, Adr=8n+6	1		Asserted						
Byte, Adr=8n+7	1	Asserted							
Word, Adr=8n	1							Asserted	Asserted
Word, Adr=8n+2	1					Asserted	Asserted		
Word, Adr=8n+4	1			Asserted	Asserted				
Word, Adr=8n+6	1	Asserted	Asserted						
Longword, Adr=8n	1					Asserted	Asserted	Asserted	Asserted
Longword, Adr=8n+4	1	Asserted	Asserted	Asserted	Asserted				
Quadword, Adr=8n	1	Asserted	Asserted	Asserted	Asserted	Asserted	Asserted	Asserted	Asserted

Table 13.11 32-Bit External Device/Little-Endian Access and Data Alignment

		Data Bus				Strobe Signals			
Operation	**No.**	**D31–D24**	**D23–D16**	**D15–D8**	**D7–D0**	$\overline{\text{WE3}}$, $\overline{\text{CAS3}}$, **DQM3**	$\overline{\text{WE2}}$, $\overline{\text{CAS2}}$, **DQM2**	$\overline{\text{WE1}}$, $\overline{\text{CAS1}}$, **DQM1**	$\overline{\text{WE0}}$, $\overline{\text{CAS0}}$, **DQM0**
Byte, Adr=4n	1		—	—	Data 7–0				Asserted
Byte, Adr=4n+1	1	—	—	Data 7–0	—			Asserted	
Byte, Adr=4n+2	1	—	Data 7–0	—	—		Asserted		
Byte, Adr=4n+3	1	Data 7–0	—	—	—	Asserted			
Word, Adr=4n	1	—	—	Data 15–8	Data 7–0			Asserted	Asserted
Word, Adr=4n+2	1	Data 15–8	Data 7–0	—	—	Asserted	Asserted		
Longword, Adr=4n	1	Data 31–24	Data 23–16	Data 15–8	Data 7–0	Asserted	Asserted	Asserted	Asserted
Quadword	1	Data 31–24	Data 23–16	Data 15–8	Data 7–0	Asserted	Asserted	Asserted	Asserted
	2	Data 63–56	Data 55–48	Data 47–40	Data 39–32	Asserted	Asserted	Asserted	Asserted

Table 13.12 16-Bit External Device/Little-Endian Access and Data Alignment

		Data Bus				Strobe Signals			
Operation	**No.**	**D31–D24**	**D23–D16**	**D15–D8**	**D7–D0**	$\overline{\text{WE3}}$, $\overline{\text{CAS3}}$, **DQM3**	$\overline{\text{WE2}}$, $\overline{\text{CAS2}}$, **DQM2**	$\overline{\text{WE1}}$, $\overline{\text{CAS1}}$, **DQM1**	$\overline{\text{WE0}}$, $\overline{\text{CAS0}}$, **DQM0**
Byte, Adr=2n	1	—	—	—	Data 7–0				Asserted
Byte, Adr=2n+1	1	—	—	Data 7–0	—			Asserted	
Word	1	—	—	Data 15–8	Data 7–0			Asserted	Asserted
Longword	1	—	—	Data 15–8	Data 7–0			Asserted	Asserted
	2	—	—	Data 31–24	Data 23–16			Asserted	Asserted
Quadword	1	—	—	Data 15–8	Data 7–0			Asserted	Asserted
	2	—	—	Data 31–24	Data 23–16			Asserted	Asserted
	3	—	—	Data 47–40	Data 39–32			Asserted	Asserted
	4	—	—	Data 63–56	Data 55–48			Asserted	Asserted

Table 13.13 8-Bit External Device/Little-Endian Access and Data Alignment

		Data Bus				Strobe Signals			
Operation	**No.**	**D31–D24**	**D23–D16**	**D15–D8**	**D7–D0**	$\overline{\text{WE3}}$, $\overline{\text{CAS3}}$, DQM3	$\overline{\text{WE2}}$, $\overline{\text{CAS2}}$, DQM2	$\overline{\text{WE1}}$, $\overline{\text{CAS1}}$, DQM1	$\overline{\text{WE0}}$, $\overline{\text{CAS0}}$, DQM0
Byte	1	—	—	—	Data 7–0				Asserted
Word	1	—	—	—	Data 7–0				Asserted
	2	—	—	—	Data 15–8				Asserted
Longword	1	—	—	—	Data 7–0				Asserted
	2	—	—	—	Data 15–8				Asserted
	3	—	—	—	Data 23–16				Asserted
	4	—	—	—	Data 31–24				Asserted
Quadword	1	—	—	—	Data 7–0				Asserted
	2	—	—	—	Data 15–8				Asserted
	3	—	—	—	Data 23–16				Asserted
	4	—	—	—	Data 31–24				Asserted
	5	—	—	—	Data 39–32				Asserted
	6	—	—	—	Data 47–40				Asserted
	7	—	—	—	Data 55–48				Asserted
	8	—	—	—	Data 63–56				Asserted

13.3.2 Areas

Area 0: For area 0, physical address bits A28 to A26 are 000.

Normal memory such as SRAM, ROM, and MPX, and also burst ROM with a burst function, can be connected to this space.

A bus width of 8, 16, 32, or 64 bits can be selected in a power-on reset by means of external pins MD3 and MD4. For details, see Memory Bus Width in section 13.1.5.

When area 0 space is accessed, the $\overline{\text{CS0}}$ signal is asserted. In addition, the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and write control signals $\overline{\text{WE0}}$ to $\overline{\text{WE7}}$, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A0W2 to A0W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{RDY}}$).

When the burst function is used, the number of burst cycle transfer states is determined in the range 2 to 9 according to the number of waits.

Area 1: For area 1, physical address bits A28 to A26 are 001.

Only normal memory such as SRAM, ROM, MPX, and byte control SRAM can be connected to this space.

A bus width of 8, 16, 32, or 64 bits can be selected with bits A1SZ1 and A1SZ0 in the BCR2 register. When MPX is connected, a bus width of 32 or 64 bits should be selected with bits A1SZ1 and A1SZ0 in the BCR2 register. When byte control SRAM is connected, select a bus width of 16, 32, or 64 bits.

When area 1 space is accessed, the $\overline{\text{CS1}}$ signal is asserted. In addition, the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and write control signals $\overline{\text{WE0}}$ to $\overline{\text{WE7}}$, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A1W2 to A1W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{RDY}}$).

The read/write strobe signal address and $\overline{\text{CS}}$ setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A1S0 and bits A1H1 and A1H0 in the WCR3 register.

Area 2: For area 2, physical address bits A28 to A26 are 010.

Normal memory such as SRAM, ROM, and MPX, and also DRAM and synchronous DRAM, can be connected to this space.

When normal memory is connected, a bus width of 8, 16, 32, or 64 bits can be selected with bits A2SZ1 and A2SZ0 in the BCR2 register. When MPX is connected, a bus width of 32 or 64 bits should be selected with bits A2SZ1 and A2SZ0 in the BCR2 register. When synchronous DRAM is connected, select 32 or 64 bits with the SZ bits in the MCR register. When DRAM is connected to area 2, select a bus width of 16 or 32 bits with the SZ bits in MCR. For details, see Memory Bus Width in section 13.1.5.

When area 2 space is accessed, the $\overline{\text{CS2}}$ signal is asserted.

When normal memory is connected, the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and write control signals $\overline{\text{WE0}}$ to $\overline{\text{WE7}}$, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A2W2 to A2W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{RDY}}$).

The read/write strobe signal address and $\overline{\text{CS}}$ setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A2S0 and bits A2H1 and A2H0 in the WCR3 register.

When synchronous DRAM is connected, the $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ signals, RD/$\overline{\text{WR}}$ signal, and byte control signals DQM0 to DQM7 are asserted, and address multiplexing is performed. $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and data timing control, and address multiplexing control, can be set using the MCR register.

When DRAM is connected, the $\overline{\text{RAS2}}$ signal, $\overline{\text{CAS4}}$ to $\overline{\text{CAS7}}$ signals, and RD/$\overline{\text{WR}}$ signal are asserted, and address multiplexing is performed. $\overline{\text{RAS2}}$, $\overline{\text{CAS}}$, and data timing control, and address multiplexing control, can be set using the MCR register.

Area 3: For area 3, physical address bits A28 to A26 are 011.

Normal memory such as SRAM, ROM, and MPX, and also DRAM and synchronous DRAM, can be connected to this space.

When normal memory is connected, a bus width of 8, 16, 32, or 64 bits can be selected with bits A3SZ1 and A3SZ0 in the BCR2 register. When MPX is connected, a bus width of 32 or 64 bits should be selected with bits A3SZ1 and A3SZ0 in the BCR2 register. When DRAM is connected, 16, 32, or 64 bits can be selected with the SZ bits in the MCR register. When synchronous DRAM is connected, select 32 or 64 bits with the SZ bits in MCR. For details, see Memory Bus Width in section 13.1.5.

When area 3 space is accessed, the $\overline{\text{CS3}}$ signal is asserted.

When normal memory is connected, the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and write control signals $\overline{\text{WE0}}$ to $\overline{\text{WE7}}$, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A3W2 to A3W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{RDY}}$).

The read/write strobe signal address and $\overline{\text{CS}}$ setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A3S0 and bits A3H1 and A3H0 in the WCR3 register.

When synchronous DRAM is connected, the $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ signals, RD/$\overline{\text{WR}}$ signal, and byte control signals DQM0 to DQM7 are asserted, and address multiplexing is performed. When DRAM is connected, the $\overline{\text{RAS}}$ signal, $\overline{\text{CAS0}}$ to $\overline{\text{CAS7}}$ signals, and RD/$\overline{\text{WR}}$ signal are asserted, and address multiplexing is performed. $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and data timing control, and address multiplexing control, can be set using the MCR register.

Area 4: For area 4, physical address bits A28 to A26 are 100.

Normal memory such as SRAM, ROM, MPX, and byte control SRAM can be connected to this space.

A bus width of 8, 16, 32, or 64 bits can be selected with bits A4SZ1 and A4SZ0 in the BCR2 register. When MPX is connected, a bus width of 32 or 64 bits should be selected with bits A4SZ1 and A4SZ0 in the BCR2 register. When byte control SRAM is connected, select a bus width of 16, 32, or 64 bits. For details, see Memory Bus Width in section 13.1.5.

When area 4 space is accessed, the $\overline{\text{CS4}}$ signal is asserted, and the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and write control signals $\overline{\text{WE0}}$ to $\overline{\text{WE7}}$, are also asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A4W2 to A4W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{RDY}}$).

The read/write strobe signal address and $\overline{\text{CS}}$ setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A4S0 and bits A4H1 and A4H0 in the WCR3 register.

Area 5: For area 5, physical address bits A28 to A26 are 101.

Normal memory such as SRAM, ROM, and MPX, and also burst ROM with a burst function, and a PCMCIA interface, can be connected to this space.

When normal memory is connected, a bus width of 8, 16, 32, or 64 bits can be selected with bits A5SZ1 and A5SZ0 in the BCR2 register. When burst ROM is connected, a bus width of 8, 16 or 32 bits can be selected with bits A5SZ1 and A5SZ0 in BCR2. When MPX is connected, a bus width of 32 or 64 bits should be selected with bits A5SZ1 and A5SZ0 in BCR2. When a PCMCIA interface is connected, either 8 or 16 bits should be selected with bits A5SZ1 and A5SZ0 in BCR2. For details, see Memory Bus Width in section 13.1.5.

When area 5 space is accessed with normal memory connected, the $\overline{\text{CS5}}$ signal is asserted. In addition, the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and write control signals $\overline{\text{WE0}}$ to $\overline{\text{WE7}}$, are asserted. When a PCMCIA interface is connected, the $\overline{\text{CE1A}}$ and $\overline{\text{CE2A}}$ signals, the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and the $\overline{\text{WE1}}$, $\overline{\text{WE2}}$, $\overline{\text{WE3}}$, and $\overline{\text{WE7}}$ signals, which can be used as $\overline{\text{WE}}$, $\overline{\text{ICIORD}}$, $\overline{\text{ICIOWR}}$, and $\overline{\text{REG}}$, respectively, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A5W2 to A5W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{RDY}}$).

When the burst function is used, the number of burst cycle transfer states is determined in the range 2 to 9 according to the number of waits.

The read/write strobe signal address and $\overline{\text{CS}}$ setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A5S0 and bits A5H1 and A5H0 in the WCR3 register.

When a PCMCIA interface is used, the address/$\overline{\text{CE1A}}$/$\overline{\text{CE2A}}$ setup and hold times with respect to the read/write strobe signals can be set in the range of 0 to 15 cycles with bits A5TED1 and A5TED0, and bits A5TEH1 and A5TEH0, in the PCR register. In addition, the number of wait cycles can be set in the range 0 to 50 with bits A5PCW1 and A5PCW0. The number of waits set in PCR is added to the number of waits set in WCR2.

Area 6: For area 6, physical address bits A28 to A26 are 110.

Normal memory such as SRAM, ROM, and MPX, and also burst ROM with a burst function, and a PCMCIA interface, can be connected to this space.

When normal memory is connected, a bus width of 8, 16, 32, or 64 bits can be selected with bits A6SZ1 and A6SZ0 in the BCR2 register. When burst ROM is connected, a bus width of 8, 16 or 32 bits can be selected with bits A6SZ1 and A6SZ0 in BCR2. When MPX is connected, a bus width of 32 or 64 bits should be selected with bits A6SZ1 and A6SZ0 in BCR2. When a PCMCIA interface is connected, either 8 or 16 bits should be selected with bits A6SZ1 and A6SZ0 in BCR2. For details, see Memory Bus Width in section 13.1.5.

When area 6 space is accessed with normal memory connected, the $\overline{\text{CS6}}$ signal is asserted. In addition, the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and write control signals $\overline{\text{WE0}}$ to $\overline{\text{WE7}}$, are asserted. When a PCMCIA interface is connected, the $\overline{\text{CE1B}}$ and $\overline{\text{CE2B}}$ signals, the $\overline{\text{RD}}$ signal, which can be used as $\overline{\text{OE}}$, and the $\overline{\text{WE1}}$, $\overline{\text{WE2}}$, $\overline{\text{WE3}}$, and $\overline{\text{WE7}}$ signals, which can be used as $\overline{\text{WE}}$, $\overline{\text{ICIORD}}$, $\overline{\text{ICIOWR}}$, and $\overline{\text{REG}}$, respectively, are asserted.

As regards the number of bus cycles, from 0 to 15 waits can be selected with bits A6W2 to A6W0 in the WCR2 register. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{RDY}}$).

When the burst function is used, the number of burst cycle transfer states is determined in the range 2 to 9 according to the number of waits.

The read/write strobe signal address and $\overline{\text{CS}}$ setup and hold times can be set within a range of 0–1 and 0–3 cycles, respectively, by means of bit A6S0 and bits A6H1 and A6H0 in the WCR3 register.

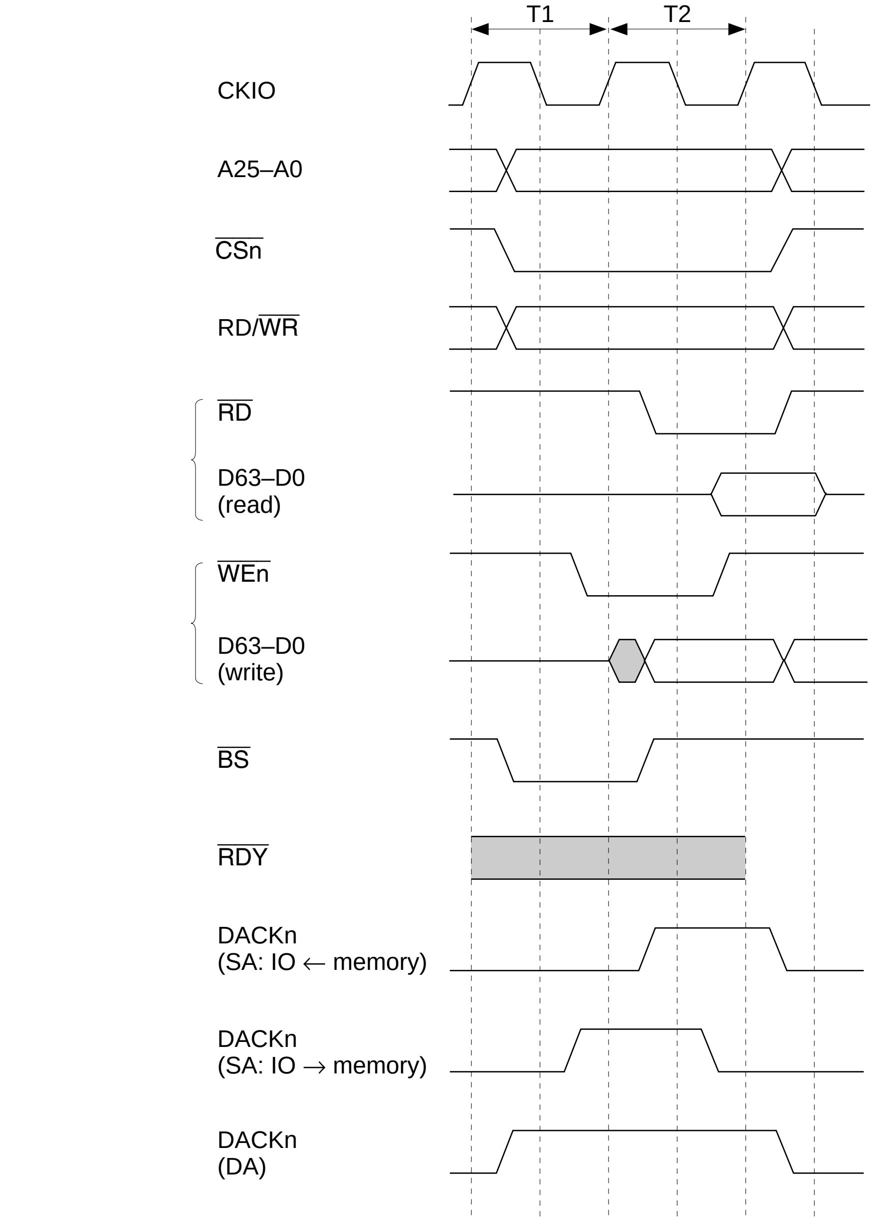
When a PCMCIA interface is used, the address/$\overline{\text{CE1B}}$/$\overline{\text{CE2B}}$ setup and hold times with respect to the read/write strobe signals can be set in the range of 0 to 15 cycles with bits A6TED1 and A6TED0, and bits A6TEH1 and A6TEH0, in the PCR register. In addition, the number of wait cycles can be set in the range 0 to 50 with bits A6PCW1 and A6PCW0. The number of waits set in PCR is added to the number of waits set in WCR2.

13.3.3 Basic Interface

Basic Timing: The basic interface of the SH7750 uses strobe signal output in consideration of the fact that mainly SRAM will be directly connected. Figure 13.5 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 asserted on the T1 rising edge, and negated on the next T2 clock rising edge. Therefore, there is no negation period in case of access at minimum pitch.

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 the case of a 32-bit device, and 16 bits in the 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 operations for cache fill or copy-back follow the set bus width and transfer a total of 32 bytes consecutively. The first access is performed on the data for which there was an access request, and the remaining accesses are performed on the data at the 32-byte boundary. The bus is not released during this transfer.



SA: Single address DMA
DA: Dual address DMA

Figure 13.5 Basic Timing of Basic Interface

Figures 13.6, 13.7, 13.8, and 13.9 show examples of connection to 64-, 32-, 16-, and 8-bit data width SRAM.

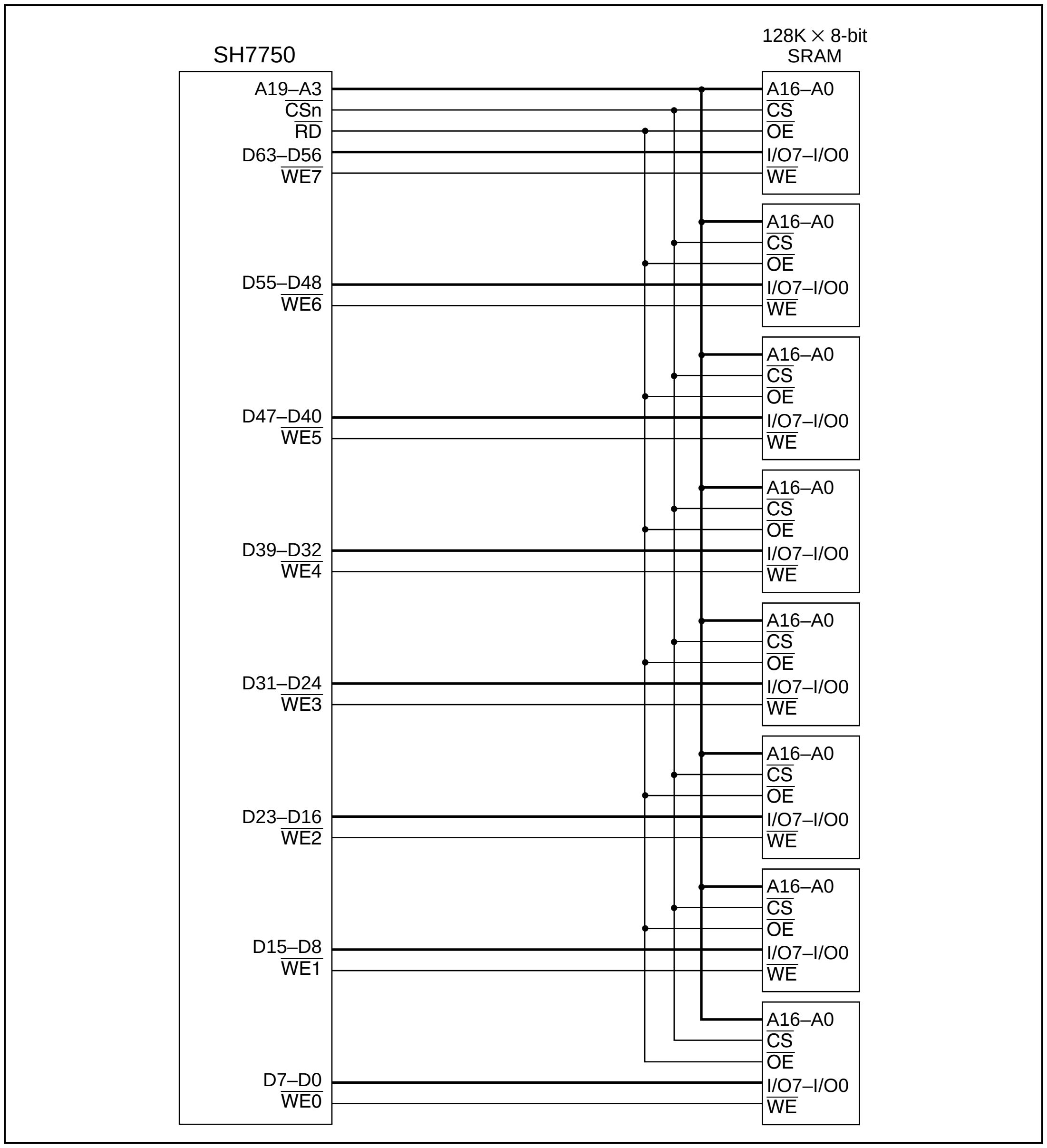


Figure 13.6 Example of 64-Bit Data Width SRAM Connection

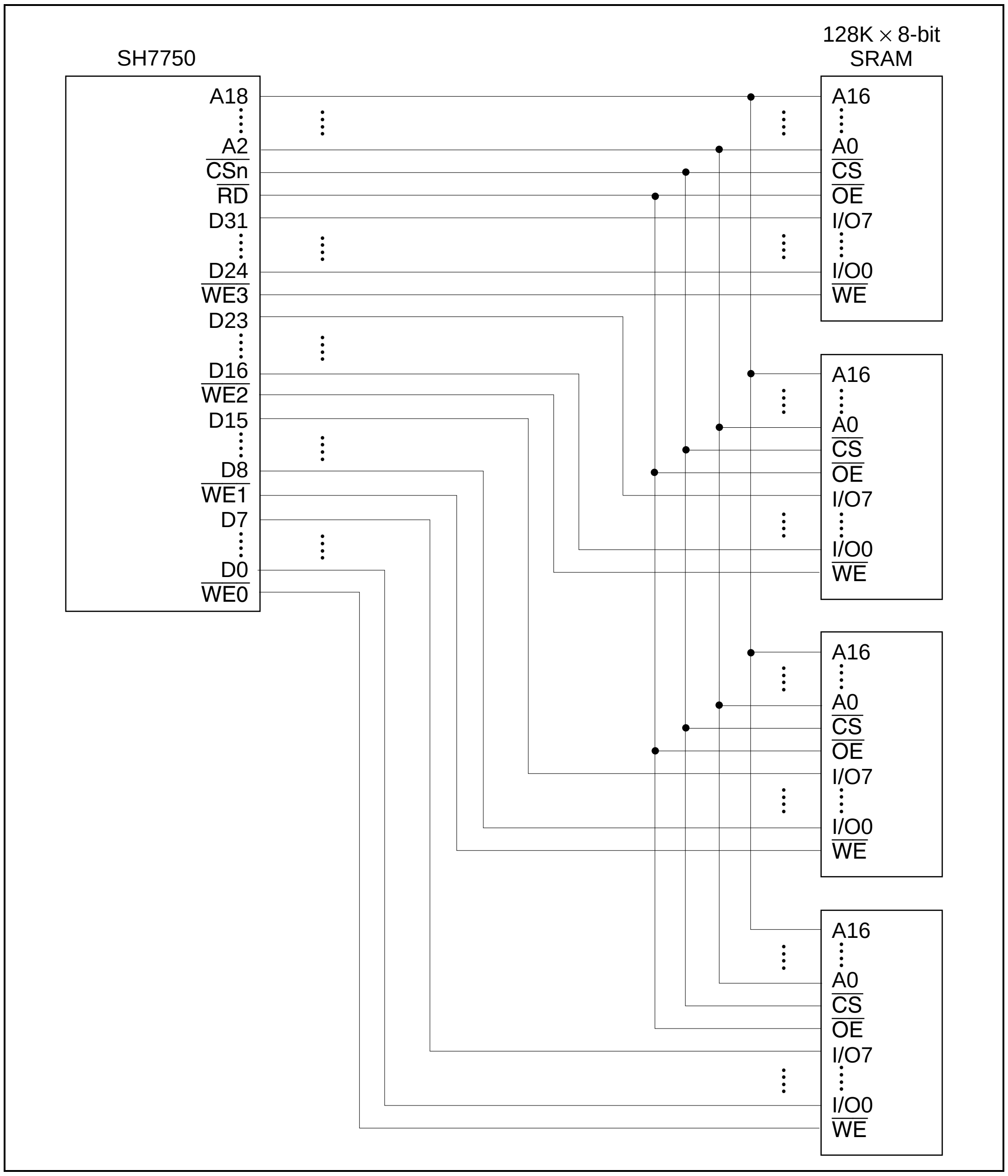


Figure 13.7 Example of 32-Bit Data Width SRAM Connection

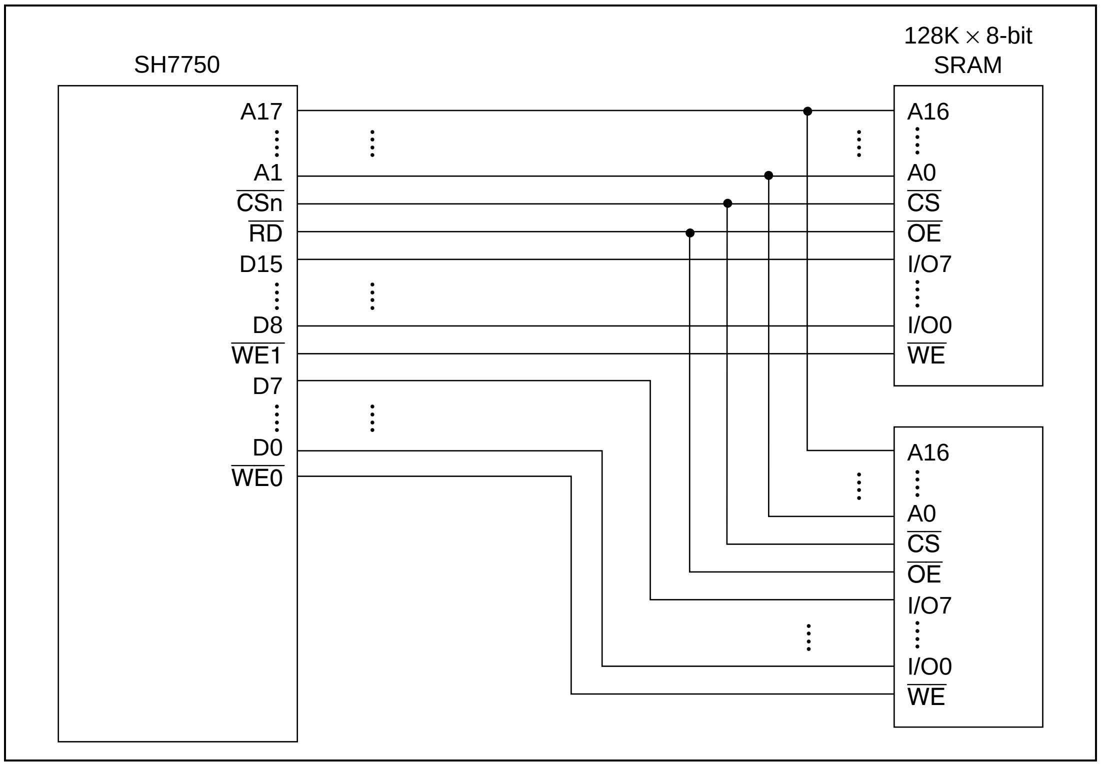


Figure 13.8 Example of 16-Bit Data Width SRAM Connection

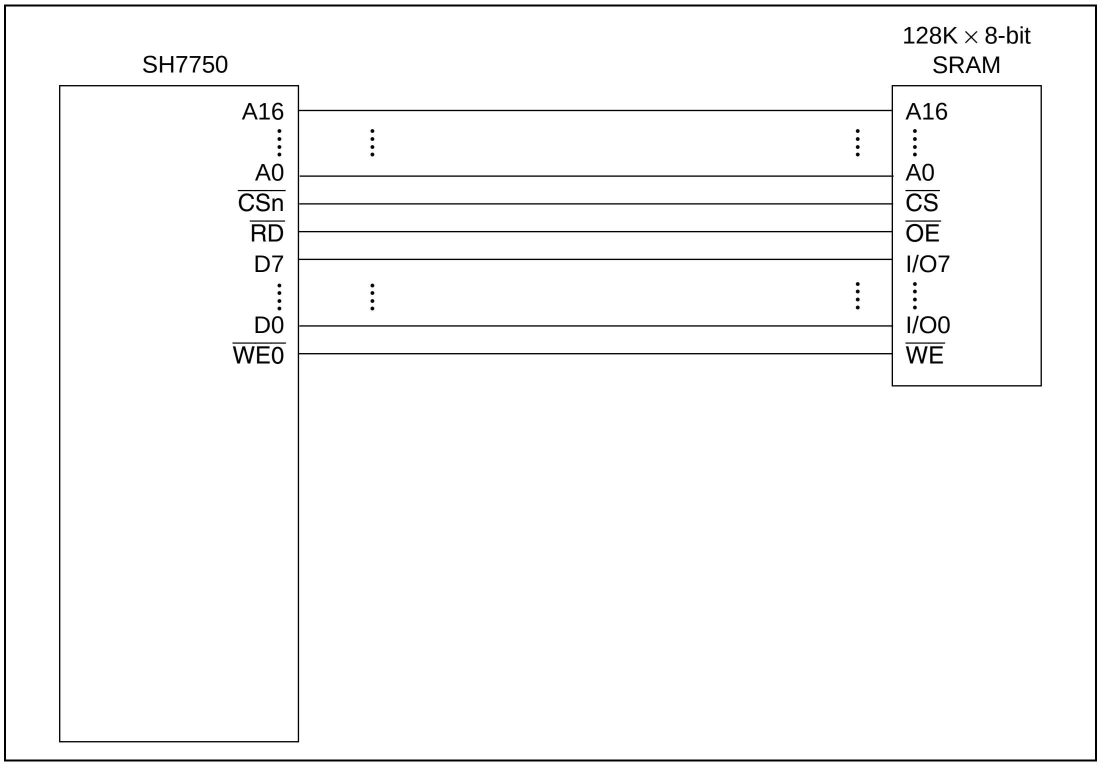


Figure 13.9 Example of 8-Bit Data Width SRAM Connection

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 13.2.4, 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.

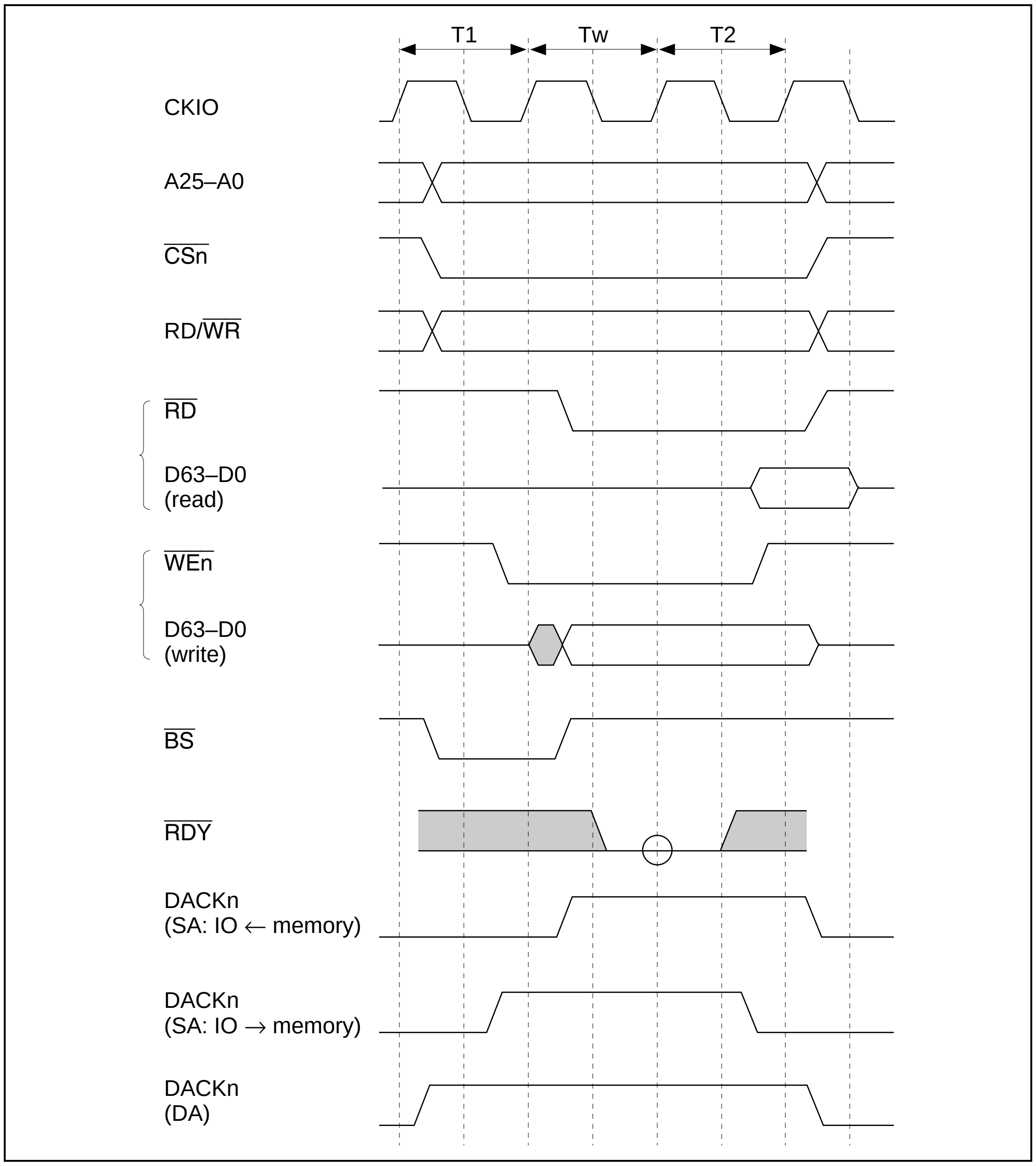


Figure 13.10 Basic Interface Wait Timing (Software Wait Only)

When software wait insertion is specified by WCR2, the external wait input $\overline{\text{RDY}}$ signal is also sampled. $\overline{\text{RDY}}$ signal sampling is shown in figure 13.11. A single-cycle wait is specified as a software wait. Sampling is performed at the transition from the Tw state to the T2 state; therefore, the $\overline{\text{RDY}}$ signal has no effect if asserted in the T1 cycle or the first Tw cycle. The $\overline{\text{RDY}}$ signal is sampled on the rising edge of the clock.

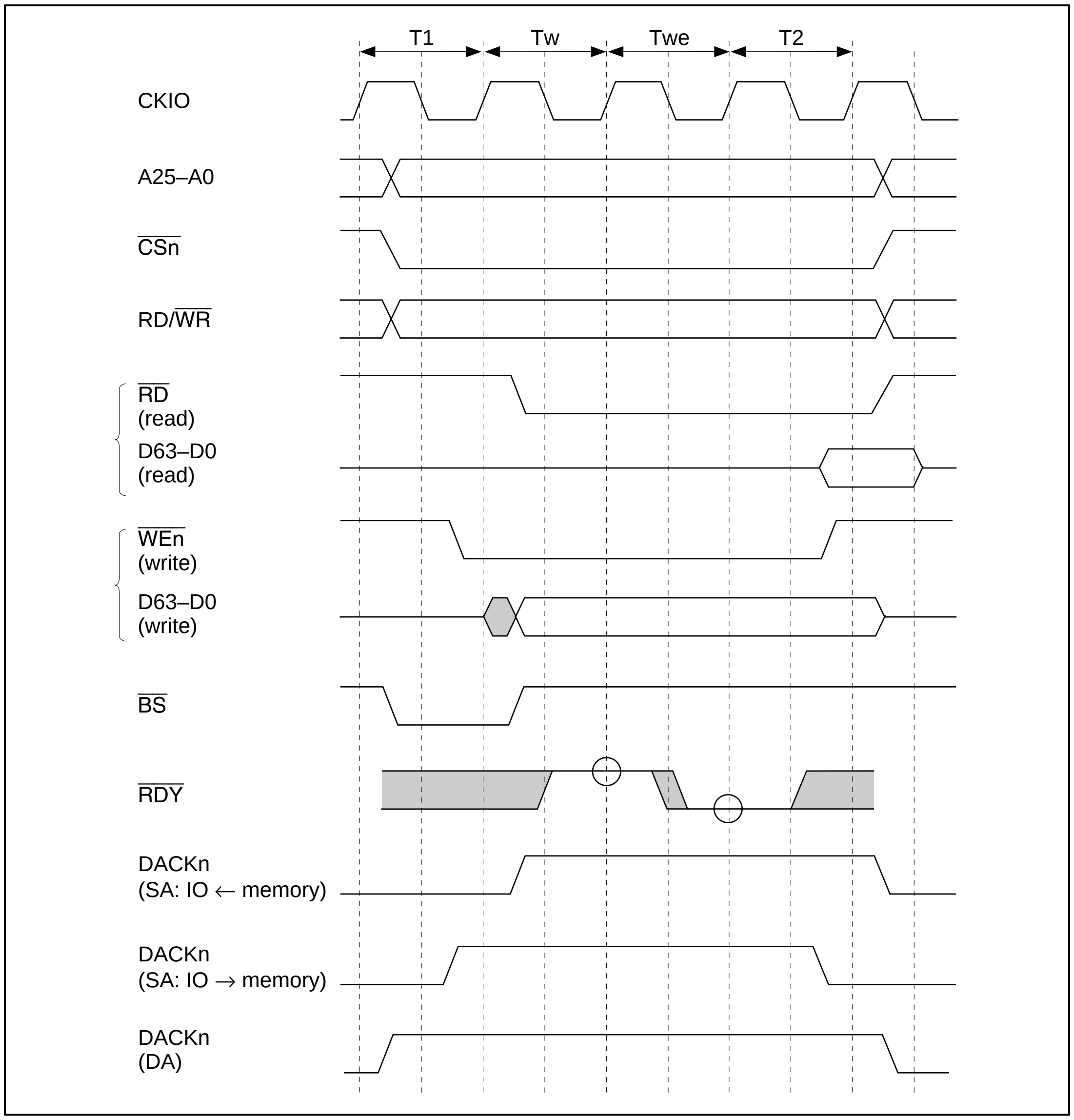


Figure 13.11 Basic Interface Wait State Timing (Wait State Insertion by $\overline{\text{RDY}}$ Signal)

13.3.4 DRAM Interface

Direct Connection of DRAM: 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 DRAM directly to the SH7750.

16, 32, or 64 bits can be selected as the interface data width for area 3 when bits DRAMTP2–0 are set to 100, and 16 or 32 bits can be used for both area 2 and 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 $\overline{\text{RAS}}$, $\overline{\text{CAS0}}$ to $\overline{\text{CAS7}}$, and RD/$\overline{\text{WR}}$. $\overline{\text{CAS2}}$ to $\overline{\text{CAS7}}$ are not used when the data width is 16 bits. When DRAM is connected to areas 2 and 3, the signals for area 2 DRAM connection are $\overline{\text{RAS2}}$, $\overline{\text{CAS4}}$ to $\overline{\text{CAS7}}$, and RD/$\overline{\text{WR}}$, and those for area 3 DRAM connection are $\overline{\text{RAS}}$, $\overline{\text{CAS0}}$ to $\overline{\text{CAS3}}$, and RD/$\overline{\text{WR}}$.

In addition to normal read and write access modes, fast page mode is supported for burst access. For DRAM connected to areas 2 and 3, EDO mode, which enables the DRAM access time to be increased, is supported.

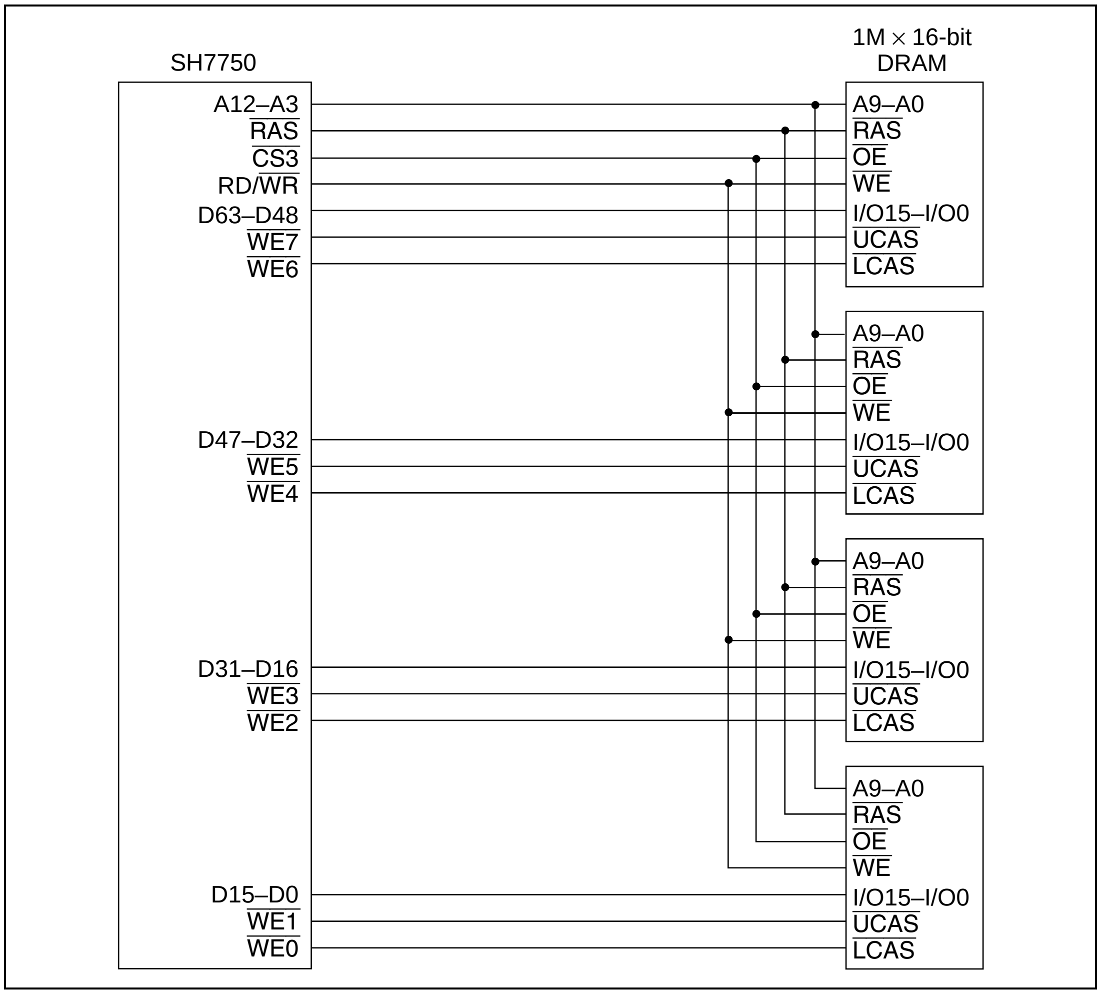


Figure 13.12 Example of DRAM Connection (64-Bit Data Width, Area 3)

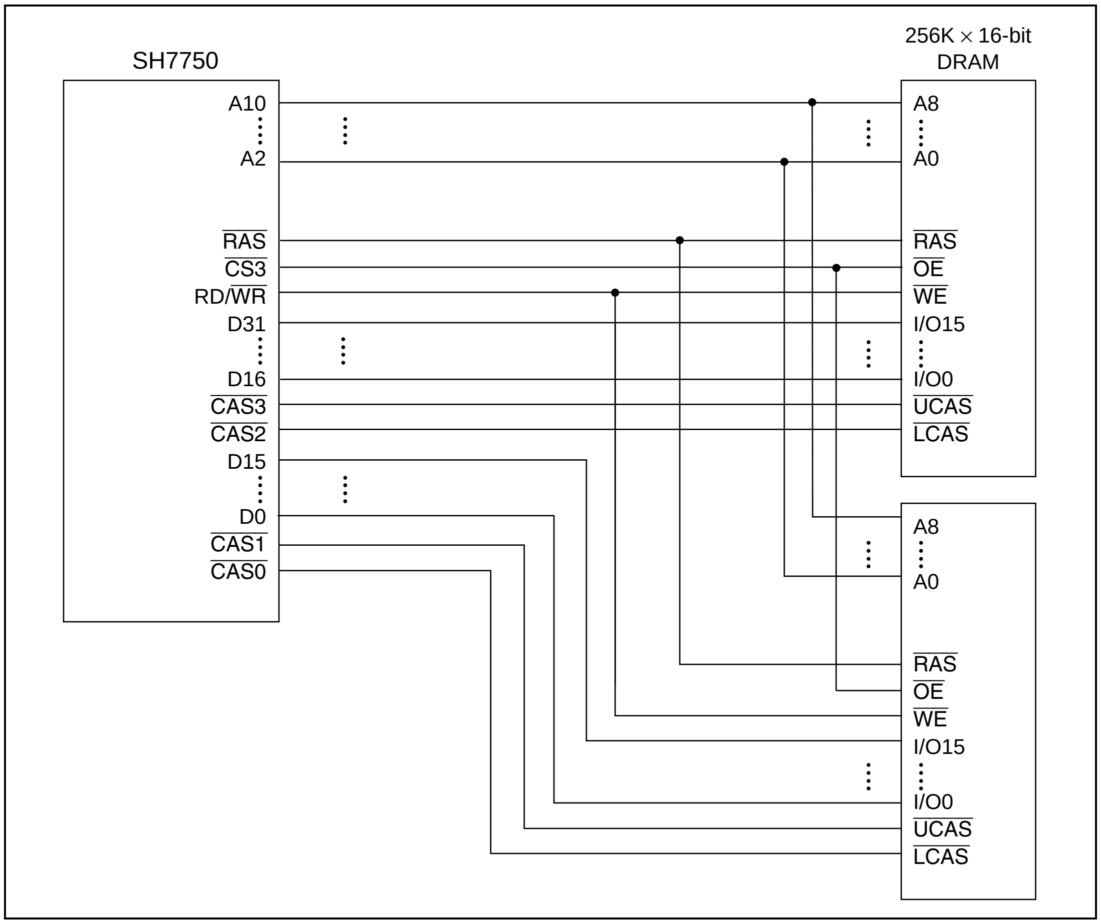


Figure 13.13 Example of DRAM Connection (32-Bit Data Width, Area 3)

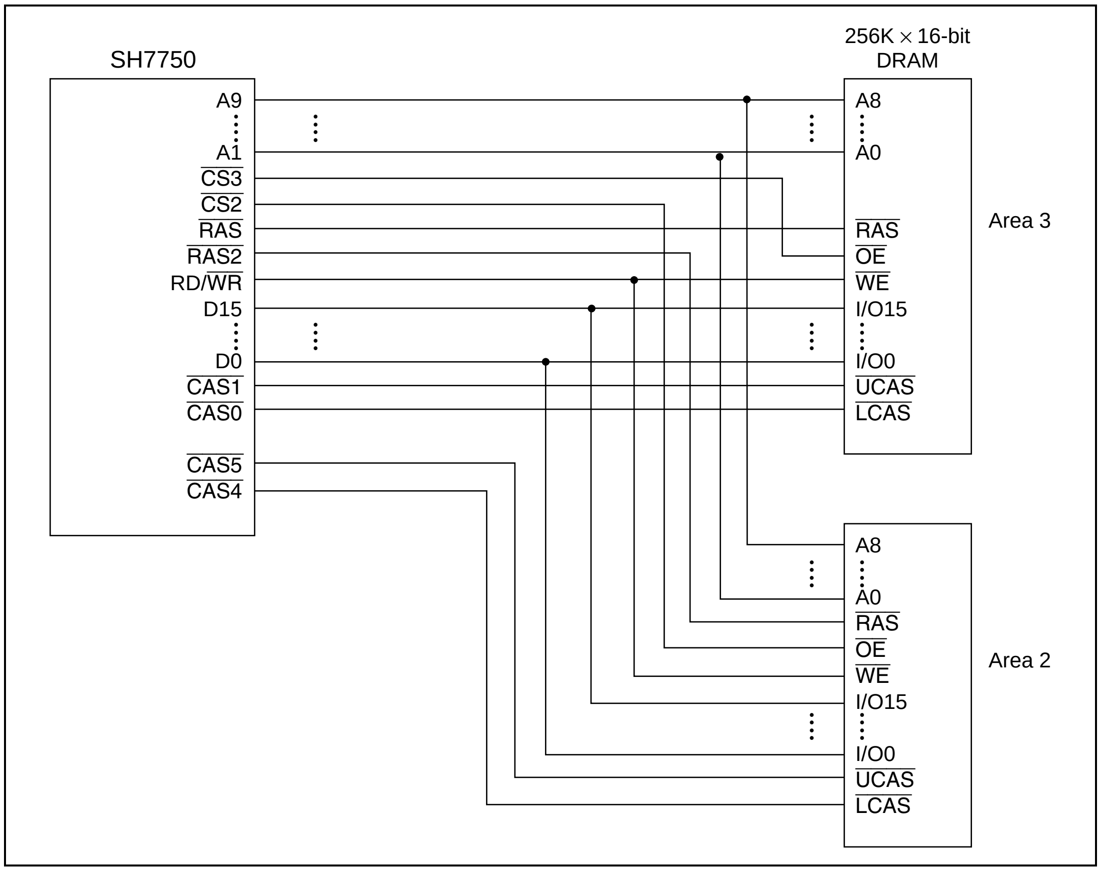


Figure 13.14 Example of DRAM Connection (16-Bit Data Width, Areas 2 and 3)

Address Multiplexing: When area 2 or area 3 is 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 SH7750 without using an external address multiplexer circuit. Any of the five multiplexing methods shown below can be selected, by setting bits AMXEXT and AMX2–0 in MCR for area 2 or 3 DRAM. The relationship between the AMXEXT and AMX2–0 bits and address multiplexing is shown in table 13.14. The address output pins subject to address multiplexing are A17 to A1. The address signals output by pins A25 to A18 are undefined.

Table 13.14 Relationship between AMXEXT and AMX2–0 Bits and Address Multiplexing

Setting				Number of Column Address Bits		External Address Pins				
AMXEXT	**AMX2**	**AMX1**	**AMX0**		**Output Timing**	**A1–A13**	**A14**	**A15**	**A16**	**A17**
0	0	0	0	8 bits	Column address	A1–A13	A14	A15	A16	A17
					Row address	A9–A21	A22	A23	A24	A25
			1	9 bits	Column address	A1–A13	A14	A15	A16	A17
					Row address	A10–A22	A23	A24	A25	A17
		1	0	10 bits	Column address	A1–A13	A14	A15	A16	A17
					Row address	A11–A23	A24	A25	A16	A17
			1	11 bits	Column address	A1–A13	A14	A15	A16	A17
					Row address	A12–A24	A25	A15	A16	A17
	1	0	0	12 bits	Column address	A1–A13	A14	A15	A16	A17
					Row address	A13–A25	A14	A15	A16	A17
Other settings				Reserved	—	—	—	—	—	—

Basic Timing: The basic timing for DRAM access is 4 cycles. This basic timing is shown in figure 13.15. Tpc is the precharge cycle, Tr the $\overline{\text{RAS}}$ assert cycle, Tc1 the $\overline{\text{CAS}}$ assert cycle, and Tc2 the read data latch cycle.

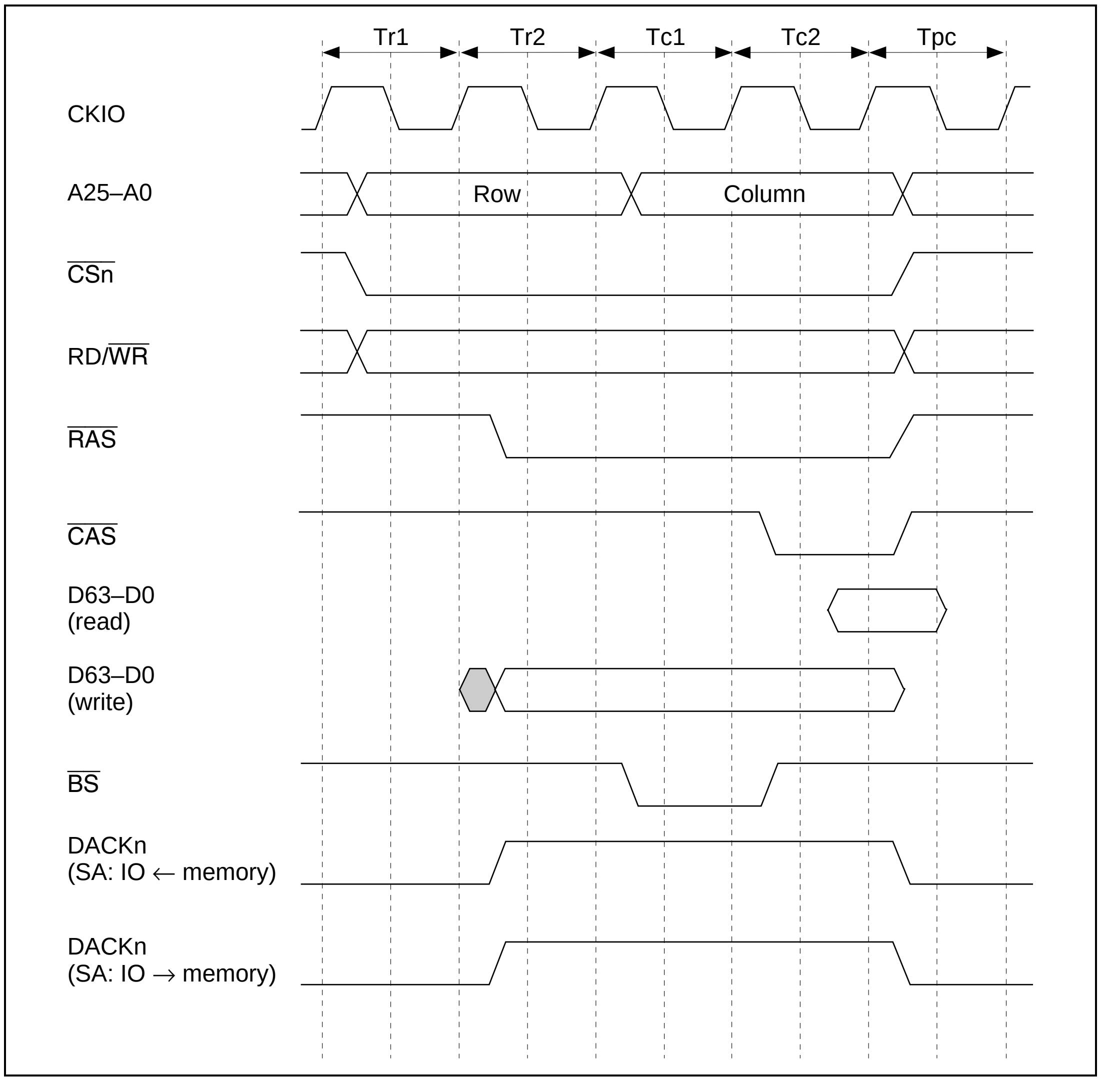


Figure 13.15 Basic DRAM Access Timing

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 and MCR. The timing with state extension using these settings is shown in figure 13.16. Additional Tpc cycles (cycles used to secure the $\overline{\text{RAS}}$ precharge time) can be inserted by means of the TPC bit in MCR, giving from 1 to 7 cycles. The number of cycles from $\overline{\text{RAS}}$ assertion to $\overline{\text{CAS}}$ assertion can be set to between 2 and 5 by inserting Trw cycles by means of the RCD bit in MCR. Also, the number of cycles from $\overline{\text{CAS}}$ assertion to the end of the access can be varied between 1 and 16 according to the setting of A2W2 to A2W0 or A3W2 to A3W0 in WCR2.

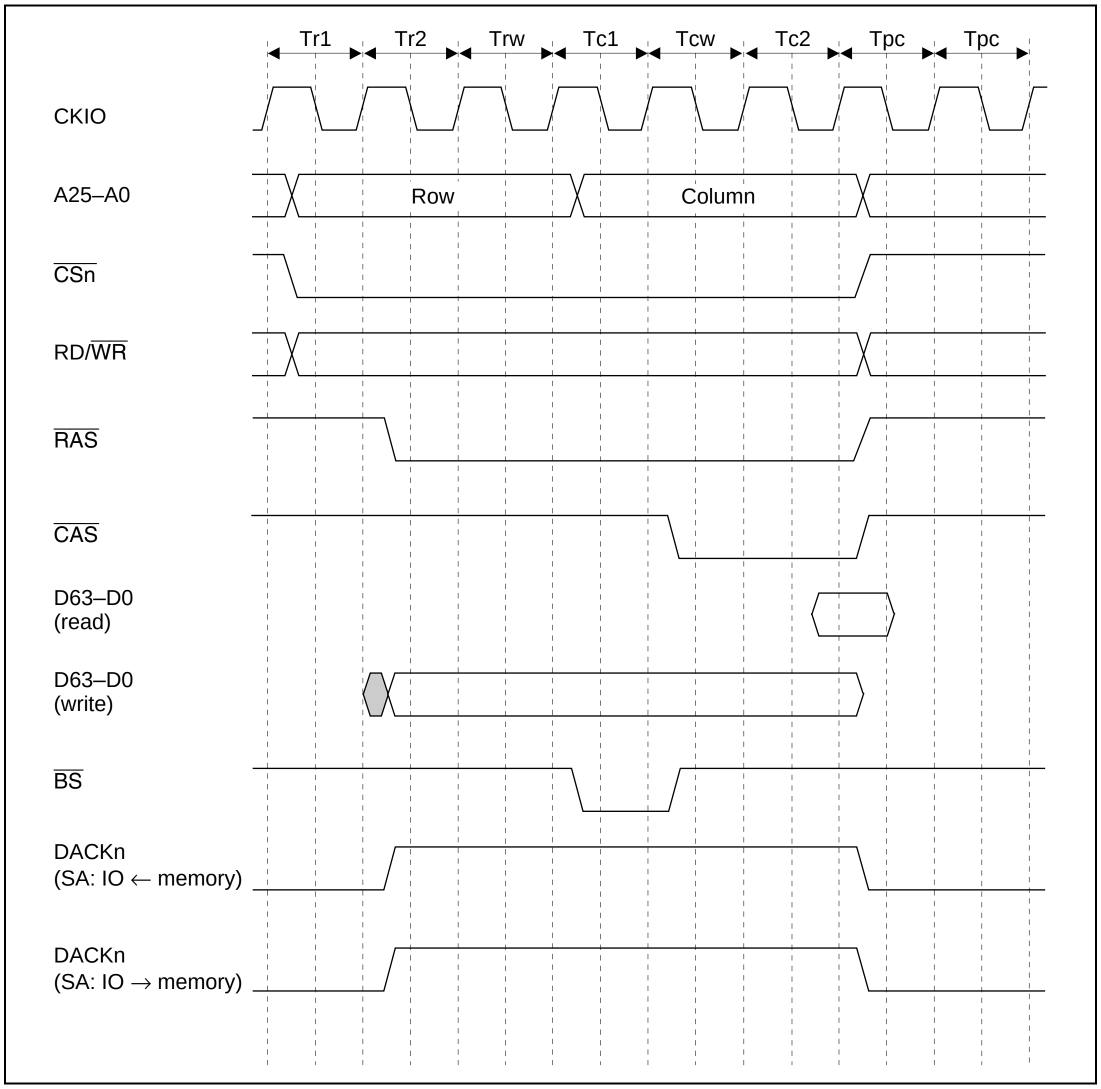


Figure 13.16 DRAM Wait State Timing

Burst Access: In addition to the normal DRAM access mode in which a row address is output in each data access, a fast 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 fast page mode can be selected by means of the burst enable (BE) bit in MCR. The timing for burst access using fast page mode is shown in figure 13.17.

In burst transfer, 4 (longword access) or 32 (cache fill or cache write-back) bytes of data are burst-transferred in the case of a 16-bit bus size. With a 32-bit bus size, 32 bytes of data are burst-transferred (cache fill or cache write-back). In a 32-byte burst transfer (cache fill), the first access comprises a longword that includes the data requiring access. The remaining accesses are performed on 32-byte boundary data that includes the relevant data. In burst transfer (cache write-back), wraparound writing is performed for 32-byte boundary data.

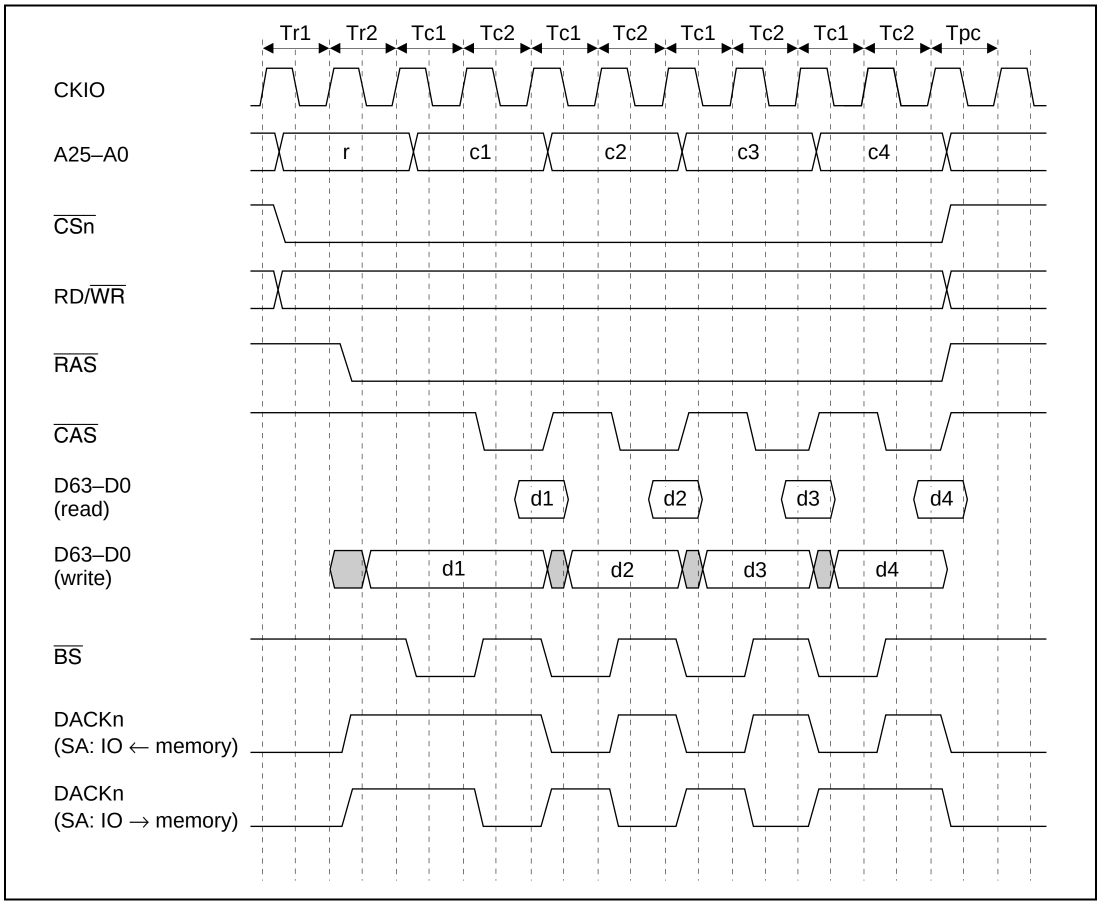


Figure 13.17 DRAM Burst Access Timing

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 SH7750, the EDO mode bit (EDOMODE) in MCR enables either normal access/burst access using fast page mode, or EDO mode normal access/burst access, to be selected for DRAM. When EDO mode is set, BE must be set to 1 in MCR. EDO mode normal access is shown in figure 13.18, and burst access in figure 13.19.

CAS Negation Period: The CAS negation period can be set to 1 or 2 by means of the TCAS bit in the MCR register.

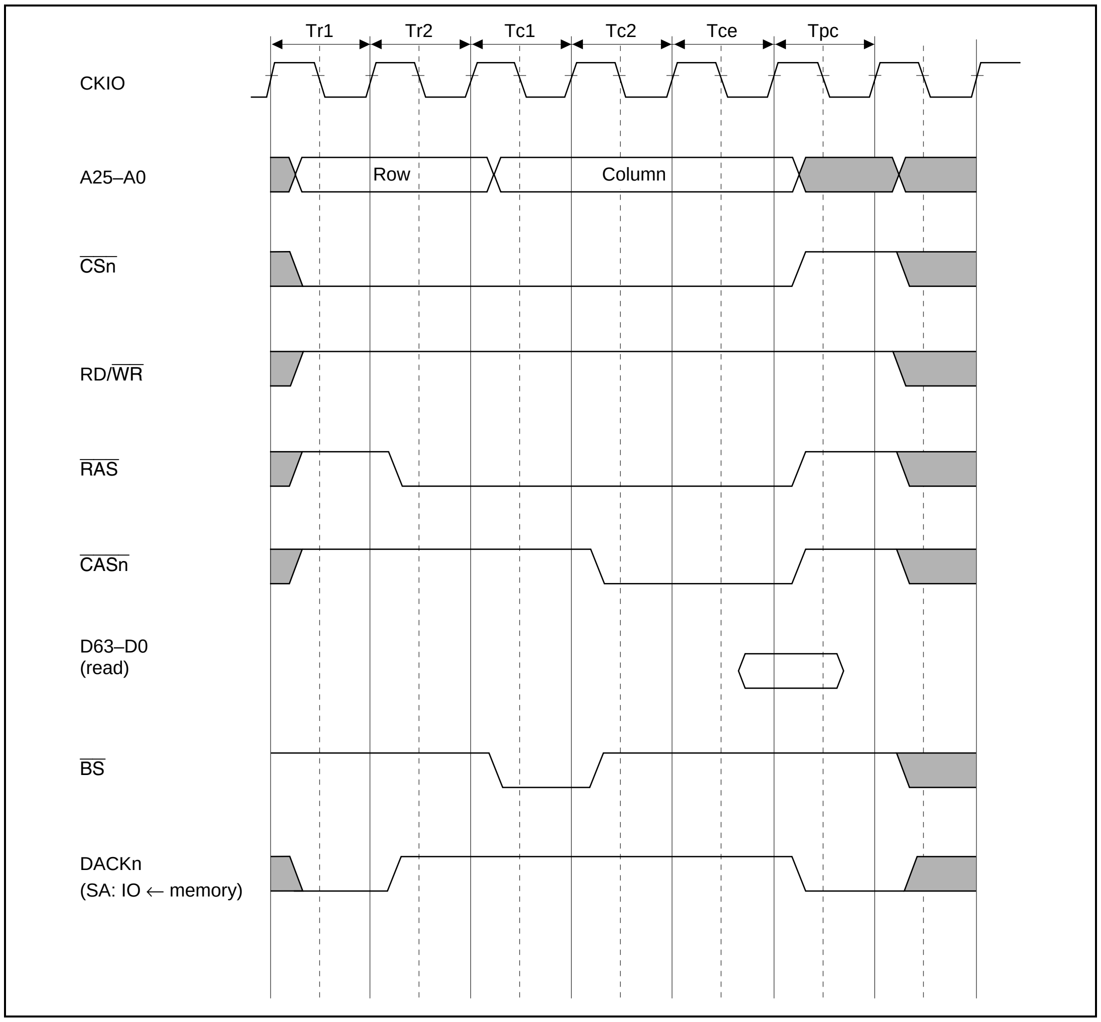


Figure 13.18 DRAM Bus Cycle (EDO Mode, RCD = 0, AnW = 0, TPC = 1)

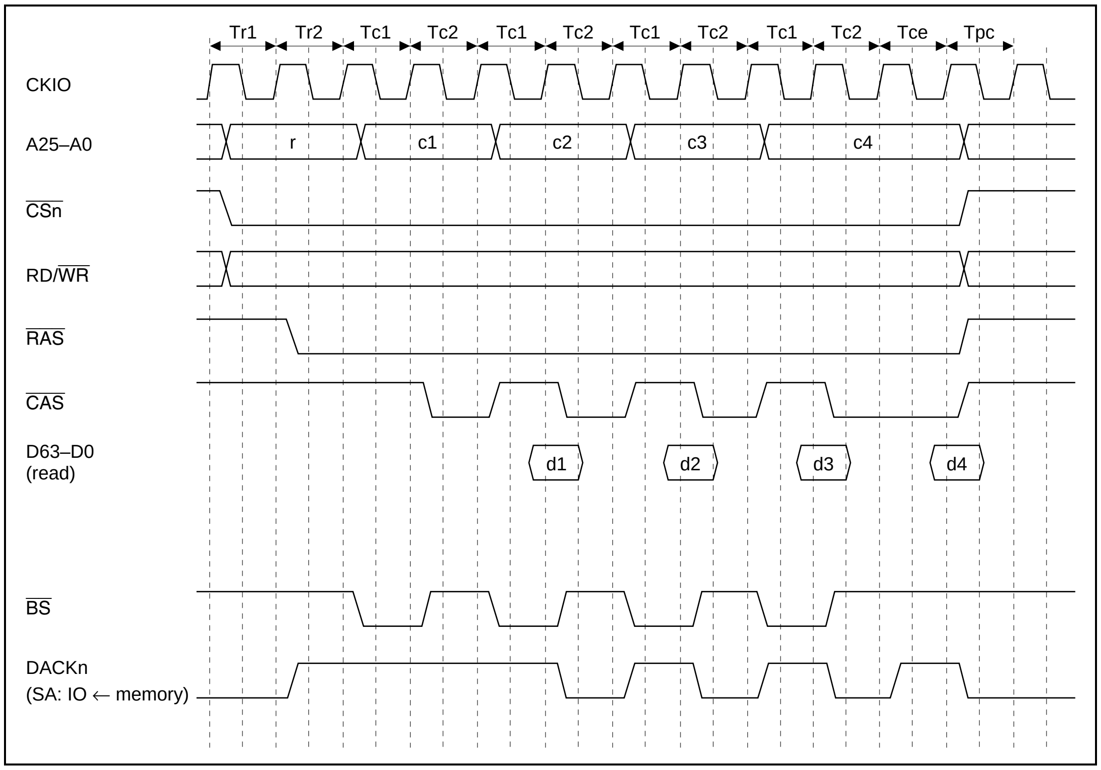


Figure 13.19 Burst Access Timing in DRAM EDO Mode

RAS Down Mode: The SH7750 has an address comparator for detecting row address matches in burst mode. By using this address comparator, and also setting RAS down mode specification bit RASD to 1, it is possible to select RAS down mode, in which $\overline{\text{RAS}}$ remains asserted after the end of an access. When RAS down mode is used, if the refresh cycle is longer than the maximum DRAM $\overline{\text{RAS}}$ assert time, the refresh cycle must be decreased to or below the maximum value of t_{RAS}.

RAS down mode can only be used when DRAM is connected in area 3.

In RAS down mode, in the event of an access to an address with a different row address, an access to a different area, a refresh request, or a bus request, $\overline{\text{RAS}}$ is negated and the necessary operation is performed. When DRAM access is resumed after this, since this is the start of RAS down mode, the operation starts with row address output. Timing charts are shown in figures 13.20 (1), (2), (3), and (4).

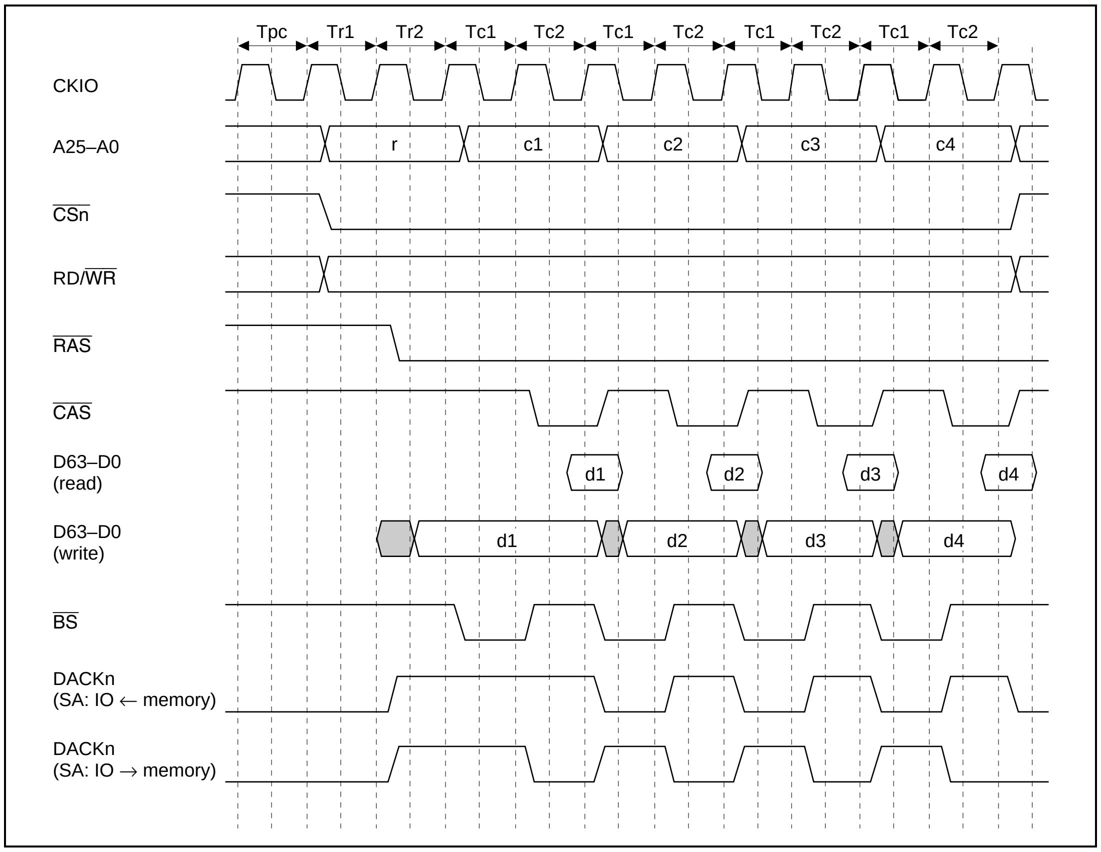


Figure 13.20 (1) DRAM Burst Bus Cycle, RAS Down Mode Start (Fast Page Mode, RCD = 0, Anw = 0)

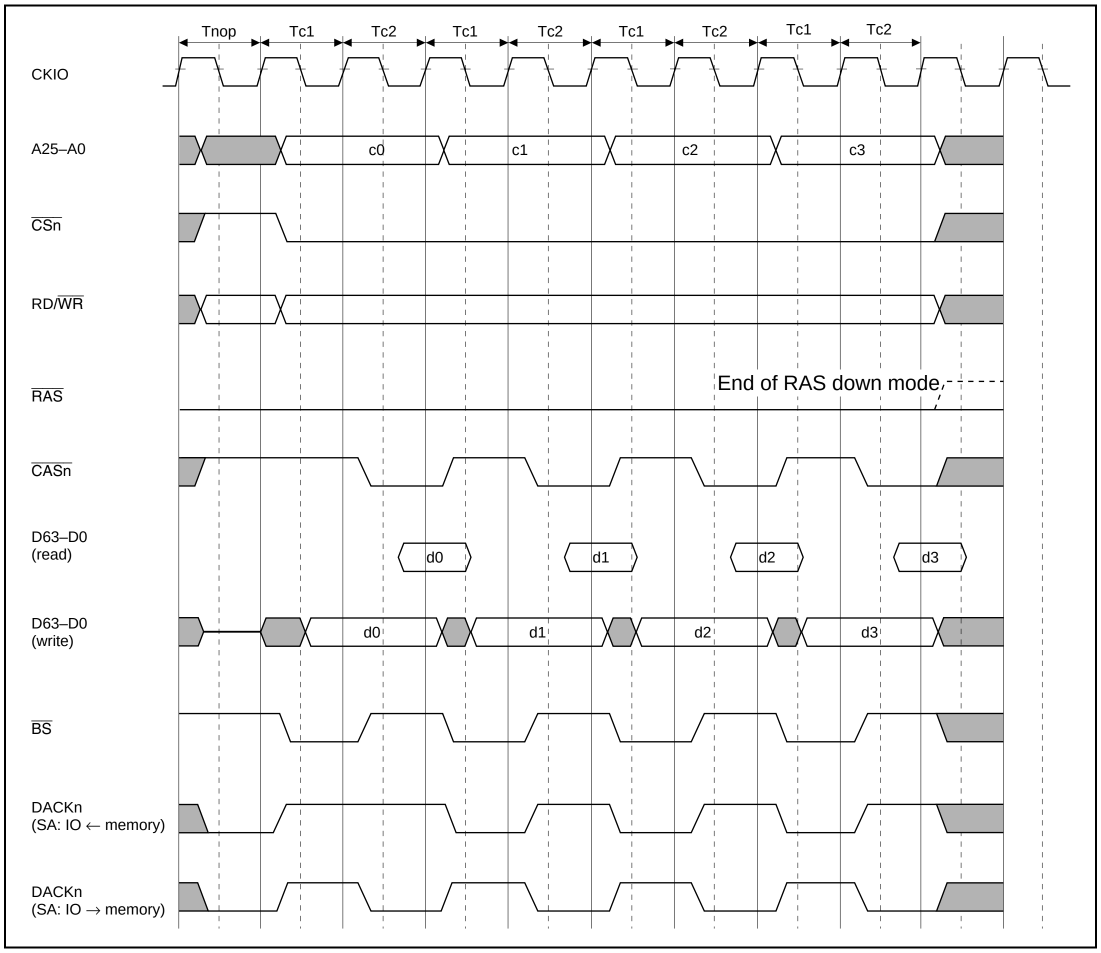


Figure 13.20 (2) DRAM Burst Bus Cycle, RAS Down Mode Continuation (Fast Page Mode, RCD = 0, Anw = 0)

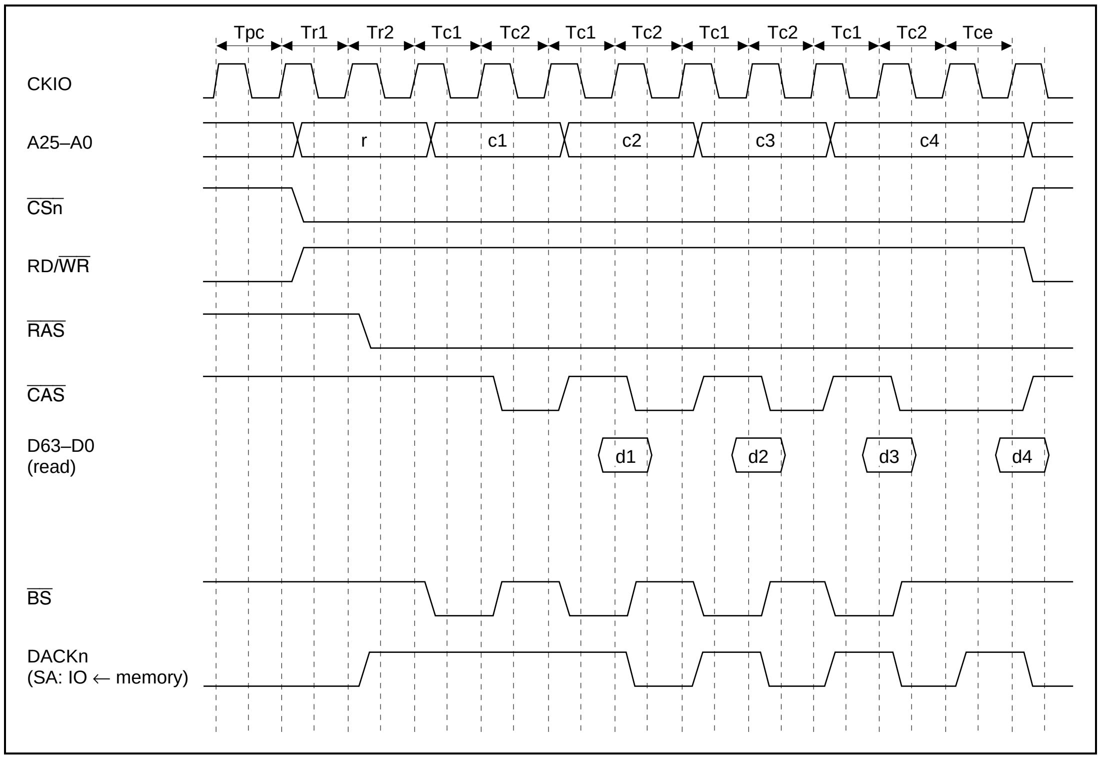


Figure 13.20 (3) DRAM Burst Bus Cycle, RAS Down Mode Start (EDO Mode, RCD = 0, Anw = 0)

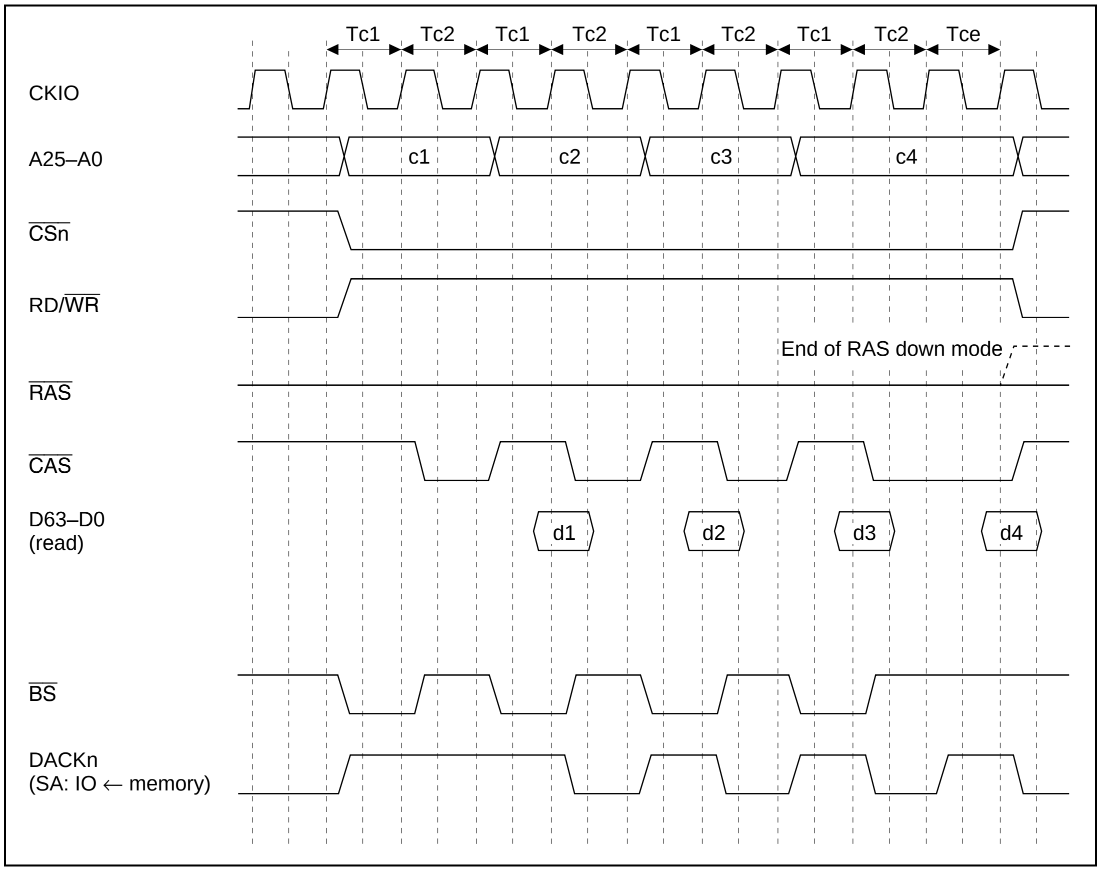


Figure 13.20 (4) DRAM Burst Bus Cycle, RAS Down Mode Continuation (EDO Mode, RCD = 0, Anw = 0)

Refresh Timing: The bus state controller includes a function for controlling DRAM refreshing. Distributed refreshing using a CAS-before-RAS cycle can be performed for DRAM by clearing the RMODE bit to 0 and setting the RFSH bit to 1 in MCR. Self-refresh mode is also supported.

When CAS-before-RAS refresh cycles are executed, refreshing is performed at intervals determined by the input clock selected by bits CKS2–CKS0 in RTCSR, and the value set in RTCOR. The value of bits CKS2–CKS0 in RTCOR should be set so as to satisfy the specification 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 $\overline{\text{BACK}}$ pin goes high. If the SH7750's external bus can be used, CAS-before-RAS refreshing is performed. At the same time, RTCNT is cleared to zero and the count-up is restarted. Figure 13.21 shows the operation of CAS-before-RAS refreshing.

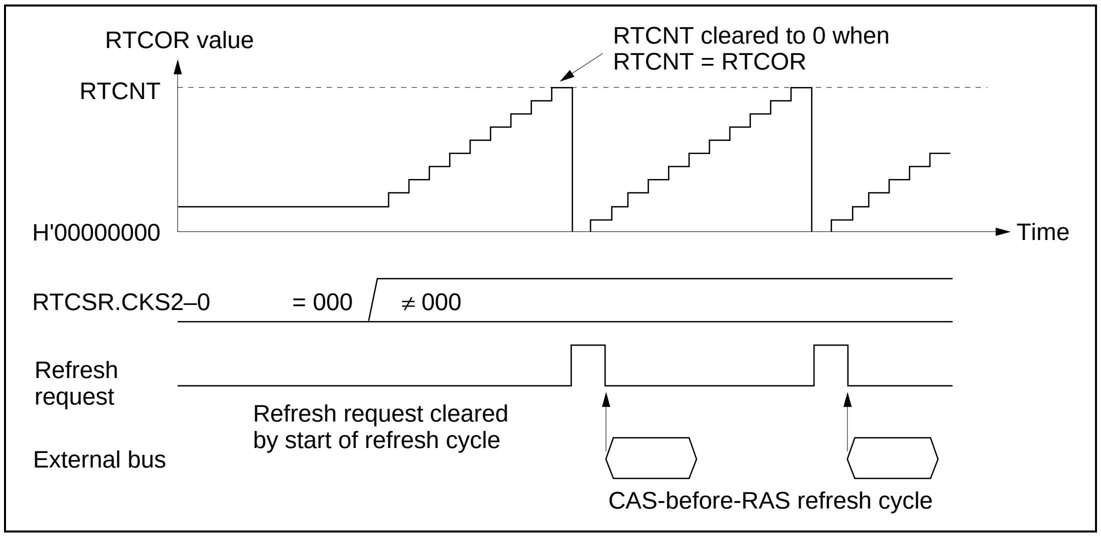


Figure 13.21 CAS-Before-RAS Refresh Operation

Figure 13.22 shows the timing of the CAS-before-RAS refresh cycle.

The number of RAS assert cycles in the refresh cycle is specified by bits TRAS2–TRAS0 in MCR. The specification of the RAS precharge time in the refresh cycle is determined by the setting of bits TRC2–TRC0 in MCR.

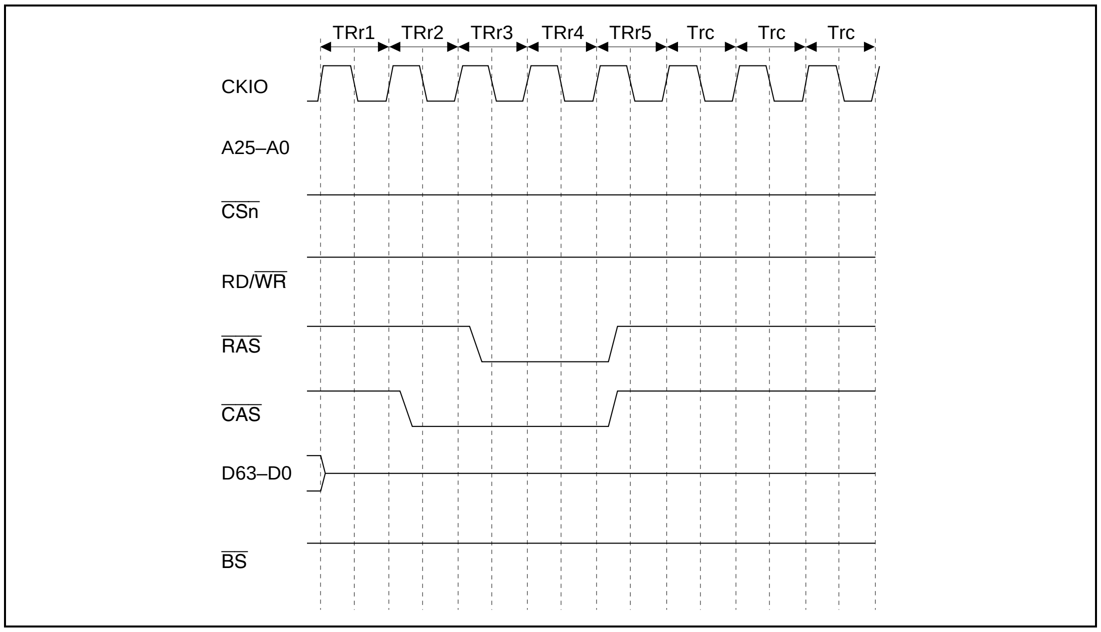


Figure 13.22 DRAM CAS-Before-RAS Refresh Cycle Timing (TRAS = 0, TRC = 1)

The self-refreshing supported by the SH7750 is shown in figure 13.23.

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 bits TRC2–TRC0 in MCR.

DRAMs include low-power products (L versions) with a long refresh cycle time (for example, the HM51W4160AL L version 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 the 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 the OVF, OVIE, and LMTS bits in RTCSR and the refresh controller's refresh count register (RFCR). The necessary procedure is as follows.

1. Normally, set the refresh counter count cycle 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. Re-set the refresh counter count cycle to the requested short cycle (e.g. 1024 cycles/16 ms), set refresh controller overflow interruption, and clear the refresh controller's 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 the case of a manual reset.

Self-refreshing is performed in normal operation, in sleep mode, in standby mode, and in the 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.

As the DRAM $\overline{\text{CAS}}$ signal is multiplexed with $\overline{\text{WEn}}$ for normal memory (SRAM, etc.), access to memory that uses the $\overline{\text{WEn}}$ signals must be disabled during self-refreshing.

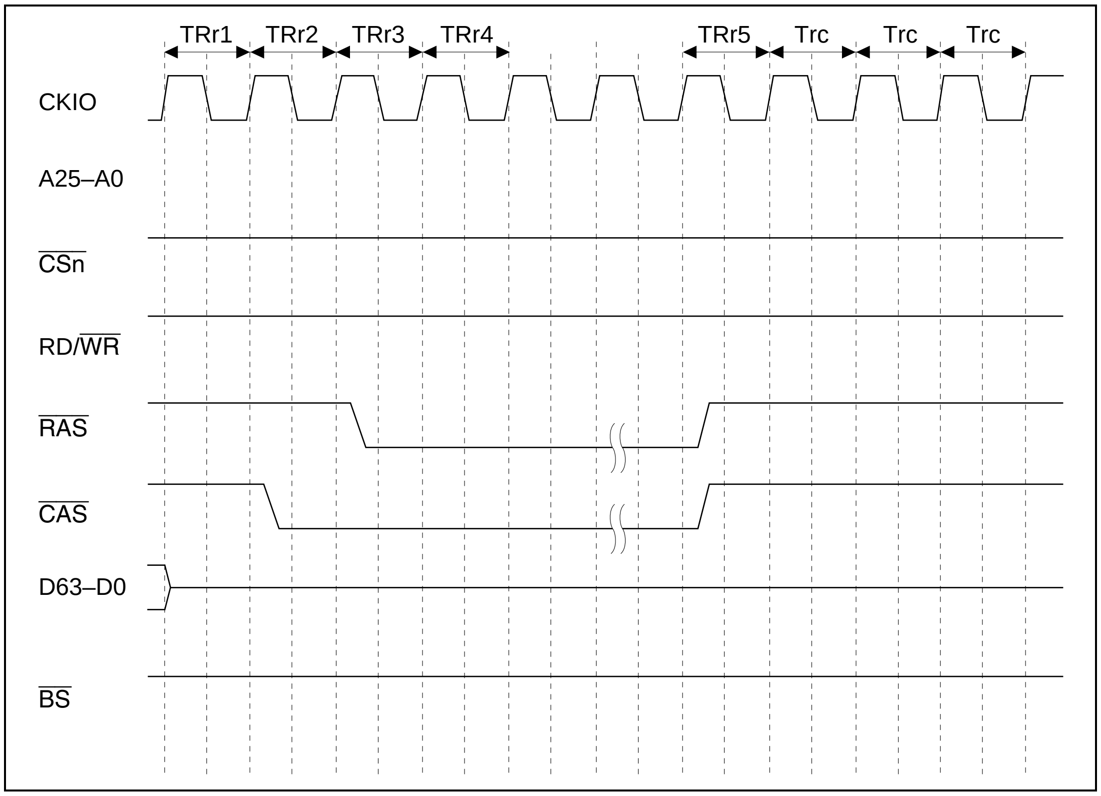


Figure 13.23 DRAM Self-Refresh Cycle Timing

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 at least the prescribed number (usually 8) of 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

Direct Connection of Synchronous DRAM: Since synchronous DRAM can be selected by the $\overline{\text{CS}}$ signal, it can be connected to physical space areas 2 and 3 using $\overline{\text{RAS}}$ and other control signals in common. If the memory type bits (DRAMTP2–0) in BCR1 are set to 010, area 2 is normal memory space and area 3 is synchronous DRAM space; if set to 011, areas 2 and 3 are both synchronous DRAM space.

With the SH7750, burst read/burst write mode is supported as the synchronous DRAM operating mode. The data bus width is 32 or 64 bits, and the SZ size bits in MCR must be set to 00 or 11. The burst enable bit (BE) in MCR is ignored, a 32-byte burst transfer is performed in a cache fill/copy-back cycle, and in a write-through area write or a non-cacheable area read/write, 32-byte data is read even in a single read in order to access synchronous DRAM with a burst read/write access. 32-byte data transfer is also performed in a single write, but DQMn is not asserted when unnecessary data is transferred.

The control signals for direct connection of synchronous DRAM are $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, RD/$\overline{\text{WR}}$, $\overline{\text{CS2}}$ or $\overline{\text{CS3}}$, DQM0 to DQM7, and CKE. All the signals other than $\overline{\text{CS2}}$ and $\overline{\text{CS3}}$ are common to all areas, and signals other than CKE are valid and latched only when $\overline{\text{CS2}}$ or $\overline{\text{CS3}}$ is asserted. Synchronous DRAM can therefore be connected in parallel to a number of areas. CKE is negated (driven low) when the frequency is changed, when the clock is unstable after the clock supply is stopped and restarted, or when self-refreshing is performed, and is always asserted (high) at other times.

Commands for synchronous DRAM are specified by $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, RD/$\overline{\text{WR}}$, and specific address signals. The commands are NOP, auto-refresh (REF), self-refresh (SELF), precharge all banks (PALL), precharge specified bank (PRE), row address strobe bank active (ACTV), read (READ), read with precharge (READA), write (WRIT), write with precharge (WRITA), and mode register setting (MRS).

Byte specification is performed by DQM0 to DQM7. A read/write is performed for the byte for which the corresponding DQM signal is low. When the bus width is 64 bits, in big-endian mode DQM7 specifies an access to address 8n, and DQM0 specifies an access to address 8n + 7. In little-endian mode, DQM7 specifies an access to address 8n + 7, and DQM0 specifies an access to address 8n.

Figures 13.24 and 13.25 show examples of the connection of 16M × 16-bit synchronous DRAMs.

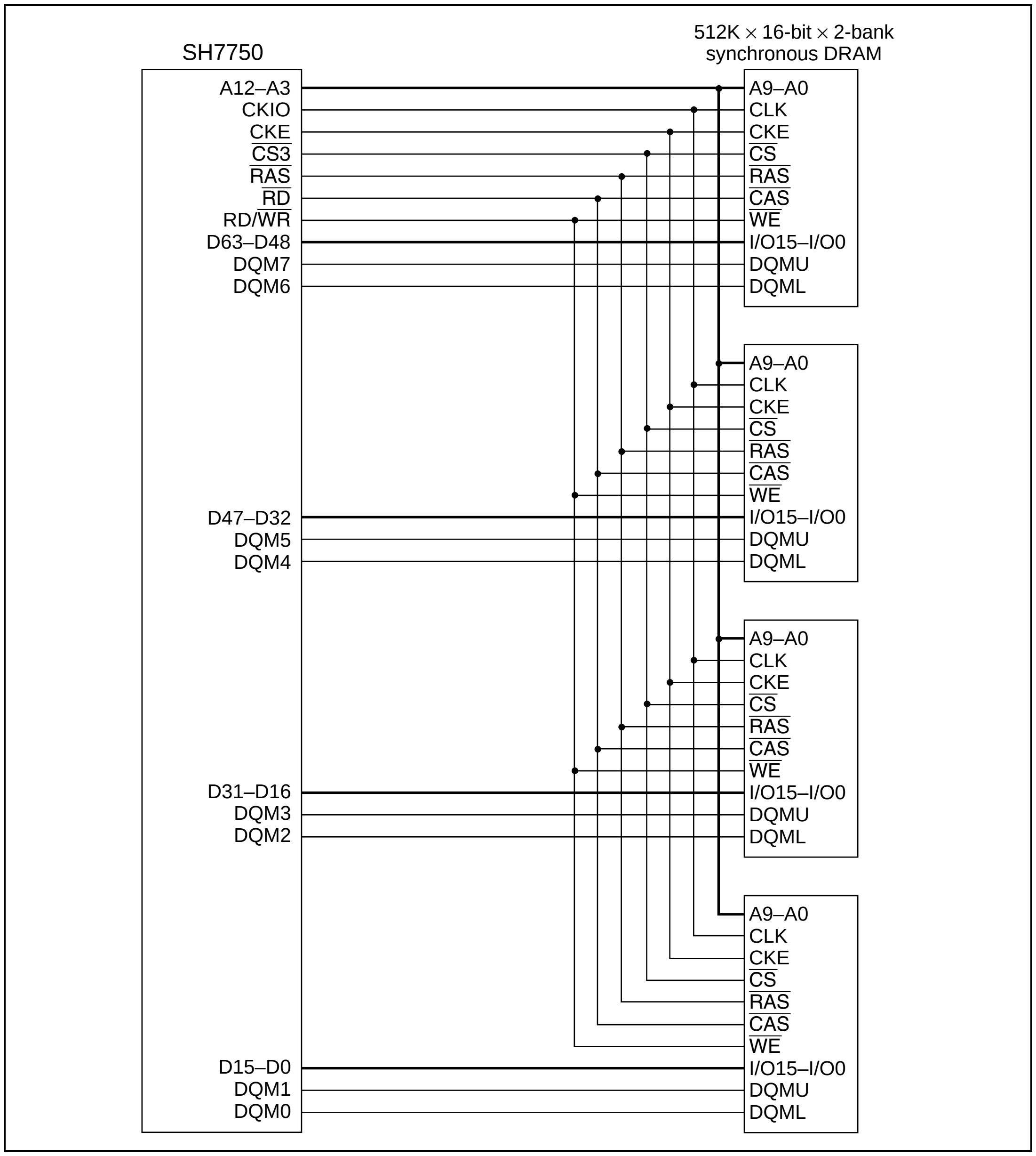


Figure 13.24 Example of 64-Bit Data Width Synchronous DRAM Connection (Area 3)

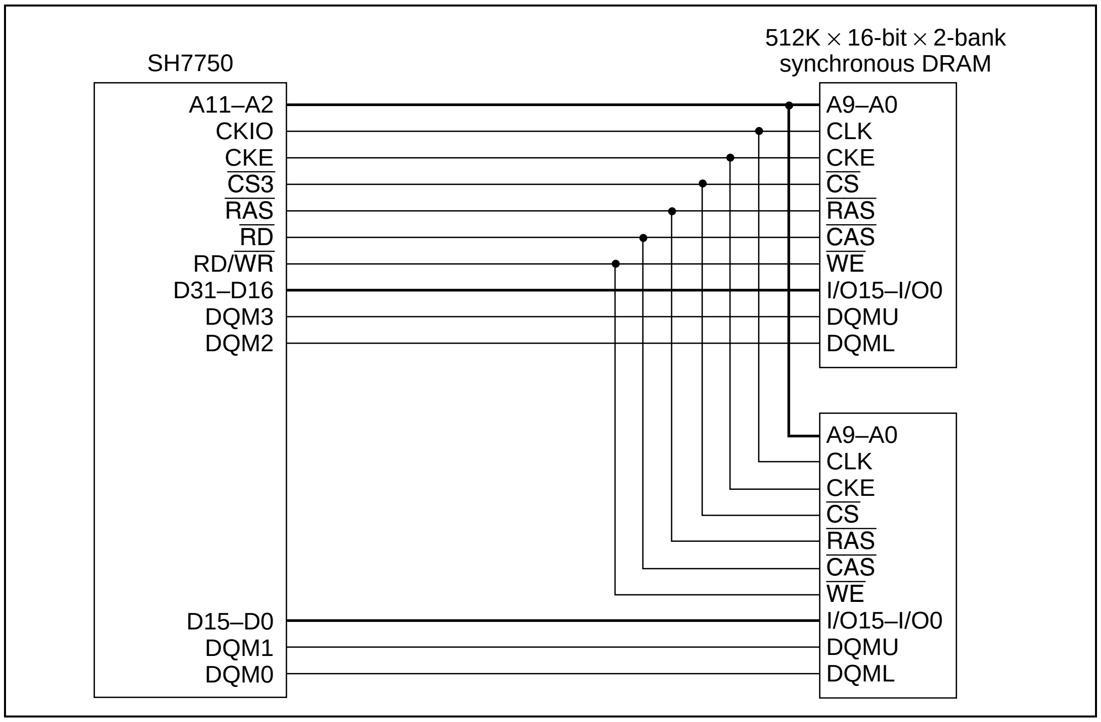


Figure 13.25 Example of 32-Bit Data Width Synchronous DRAM Connection (Area 3)

Address Multiplexing: Synchronous DRAM can be connected without external multiplexing circuitry in accordance with the address multiplex specification bits AMXEXT and AMX2–AMX0 in MCR. Table 13.15 shows the relationship between the address multiplex specification bits and the bits output at the address pins. See Appendix F, Synchronous DRAM Address Multiplexing Tables.

The address signals output at A25–A18, A1, and A0 are undefined.

When A0, the LSB of the synchronous DRAM address, is connected to the SH7750, with a 32-bit bus width it makes a longword address specification. Connection should therefore be made in this order: connect pin A0 of the synchronous DRAM to pin A2 of the SH7750, then connect pin A1 to pin A3.

With a 64-bit bus width, the LSB makes a quadword address specification. Connection should therefore be made in this order: connect pin A0 of the synchronous DRAM to pin A3 of the SH7750, then connect pin A1 to pin A4.

Table 13.15 Example of Correspondence between SH7750 and Synchronous DRAM Address Pins (64-Bit Bus Width, AMX2–AMX0 = 011, AMXEXT = 0)

SH7750 Address Pin			Synchronous DRAM Address Pin	
	RAS Cycle	CAS Cycle		Function
A14	A22	A22	A11	BANK select bank address
A13	A21	H/L	A10	Address precharge setting
A12	A20	0	A9	
A11	A19	0	A8	
A10	A18	A10	A7	
A9	A17	A9	A6	
A8	A16	A8	A5	
A7	A15	A7	A4	
A6	A14	A6	A3	
A5	A13	A5	A2	
A4	A12	A4	A1	
A3	A11	A3	A0	
A2	—	A2	Not used	
A1	—	A1	Not used	
A0	—	A0	Not used	

Burst Read: The timing chart for a burst read is shown in figure 13.26. In the following example it is assumed that four 512K x 16-bit x 2-bank synchronous DRAMs are connected, and a 64-bit data width is used. The burst length is 4. Following the Tr cycle in which ACTV command output is performed, a READA command is issued in the Tc1 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 synchronous DRAM; no new access command can be issued to the same bank during this cycle. In the SH7750, the number of Tpc cycles is determined by the specification of bits TPC2–TPC0 in MCR, and commands are not issued for the same synchronous DRAM during this interval.

The example in figure 13.26 shows the basic cycle. To connect slower synchronous DRAM, the cycle can be extended by setting WCR2 and MCR bits. The number of cycles from the ACTV command output cycle, Tr, to the READA command output cycle, Tc1, can be specified by bits RCD1 and RCD0 in MCR, with a value of 0 to 3 specifying 2 to 4 cycles, respectively. In the case of 2 or more cycles, a Trw cycle, in which an NOP command is issued for the synchronous DRAM, is inserted between the Tr cycle and the Tc cycle. The number of cycles from READA command output cycle Tc1 to the first read data latch cycle, Td1, can be specified as 1 to 5 cycles independently for areas 2 and 3 by means of bits A2W2–A2W0 and A3W2–A3W0 in

WCR2. This number of cycles corresponds to the number of synchronous DRAM CAS latency cycles.

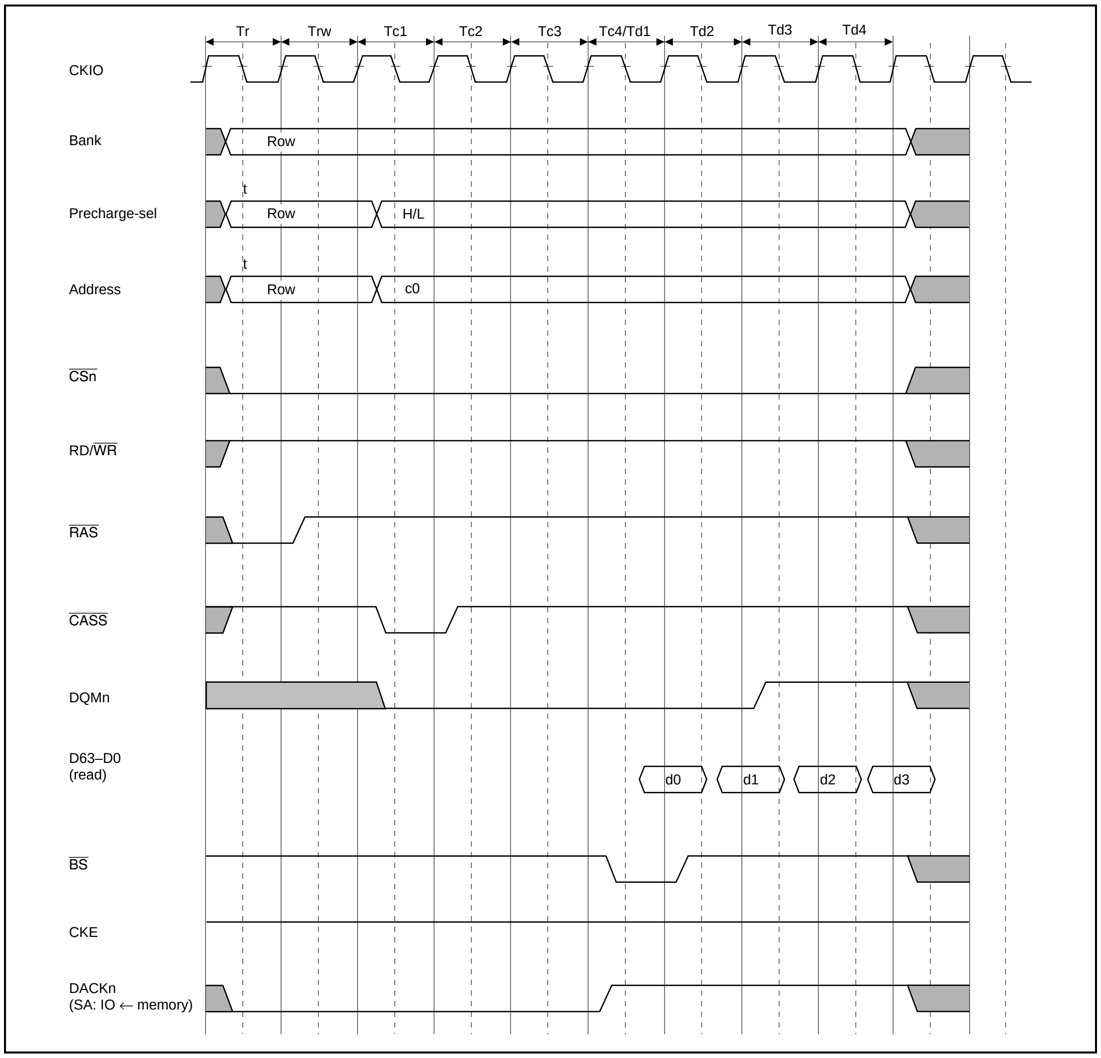


Figure 13.26 Basic Timing for Synchronous DRAM Burst Read

In a synchronous DRAM cycle, the $\overline{\text{BS}}$ signal is asserted for one cycle at the start of the bus cycle. The order of access is as follows: in a fill operation in the event of a cache miss, 64-bit boundary data including the missed data is read first, then 32-byte boundary data including the missed data is read in wraparound mode.

Single Read: With the SH7750, as synchronous DRAM is set to burst read/burst write mode, read data output continues after the required data has been read. To prevent data collisions, after the required data is read in Td1, empty read cycles Td2 to Td4 are performed, and the SH7750 waits for the end of the synchronous DRAM operation. The $\overline{\text{BS}}$ signal is asserted only in Td1.

When the data width is 64 bits, there are 4 burst transfers in a read. In cache-through and other DMA read cycles, of cycles Td1 to Td4, $\overline{\text{BS}}$ is asserted and data latched only in the Td1 cycle.

Since such empty cycles increase the memory access time, and tend to reduce program execution speed and DMA transfer speed, it is important both to avoid unnecessary cache-through area accesses, and to use a data structure that will allow data to be placed at a 32-byte boundary, and to be transferred in 32-byte units, when carrying out DMA transfer with synchronous DRAM specified as the source.

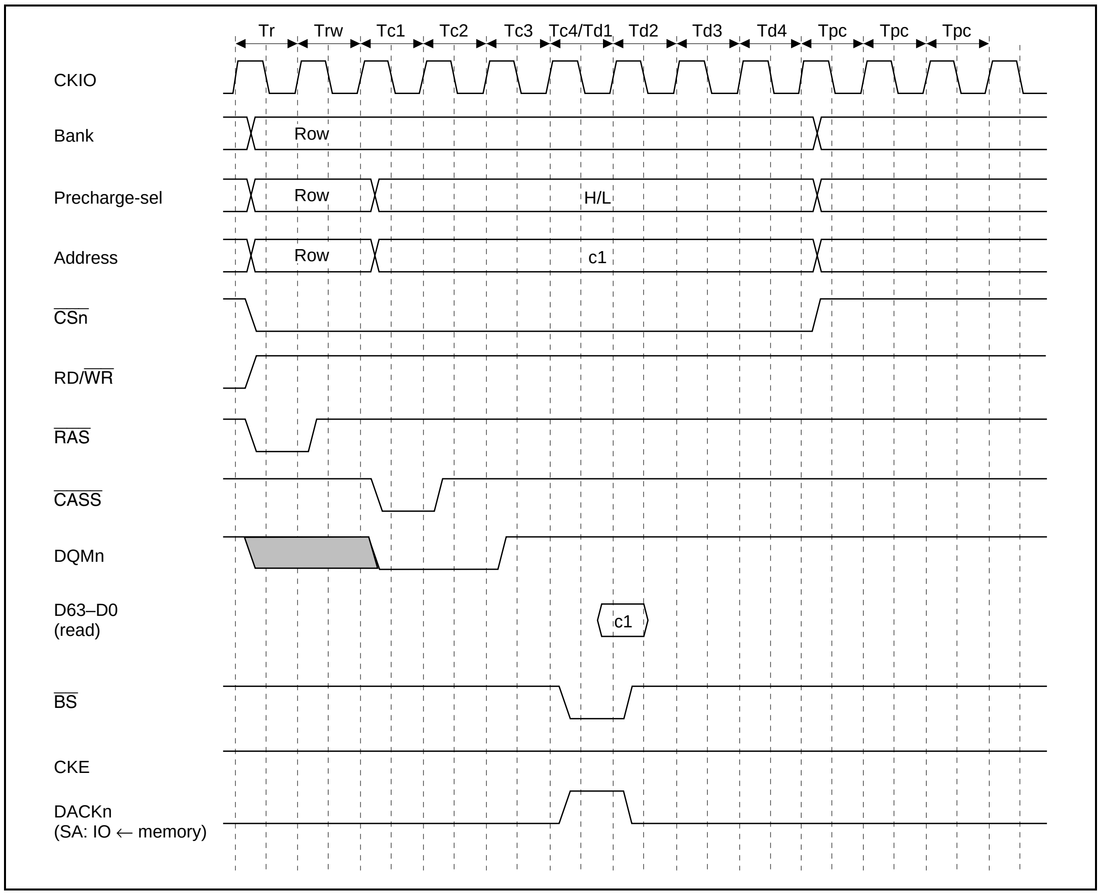


Figure 13.27 Basic Timing for Synchronous DRAM Single Read

Burst Write: The timing chart for a burst write is shown in figure 13.28. In the SH7750, a burst write occurs only in the event of cache copy-back or a 32-byte transfer by the DMAC. In a burst 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 the case of the write with auto-precharge command, precharging of the relevant bank is performed in the synchronous DRAM 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 bits TRWL2–TRWL0 in MCR. 32-byte boundary data is written in wraparound mode.

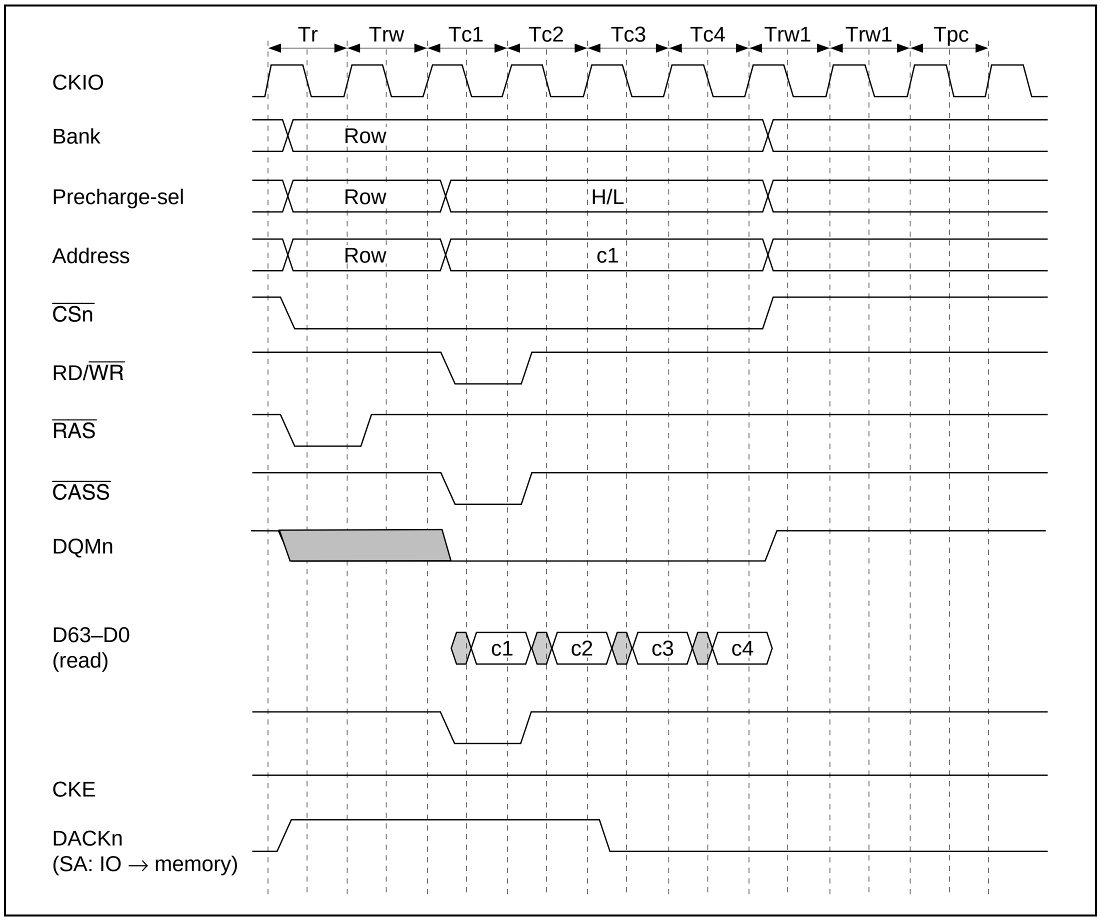


Figure 13.28 Basic Timing for Synchronous DRAM Burst Write

Single Write: The basic timing chart for write access is shown in figure 13.29. 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 the case of a write with auto-precharge, precharging of the relevant bank is performed in the synchronous DRAM 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 bits TRWL2–TRWL0 in MCR.

As the SH7750 supports burst read/burst write operations for synchronous DRAM, a single write requires the same number of cycles as a burst write.

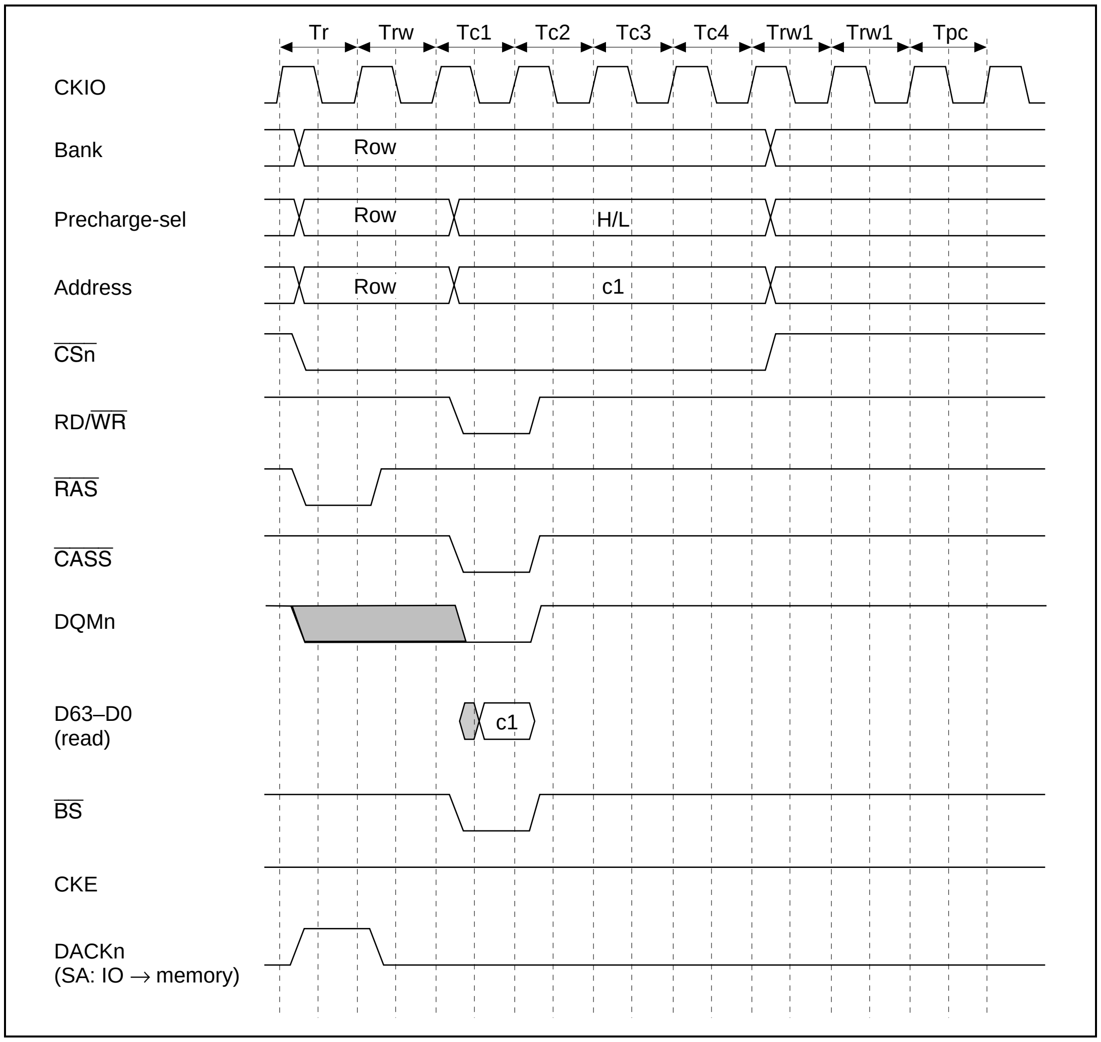


Figure 13.29 Basic Timing for Synchronous DRAM Single Write

RAS Down Mode: The synchronous DRAM 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 DRAM RAS down state. As synchronous DRAM 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 WRIT command. When RAS down 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 bits TPC2–TPC0 in MCR.

There is a limit on t_{RAS}, 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 t_{RAS}. 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 13.30, a burst read cycle for the same row address in figure 13.31, and a burst read cycle for different row addresses in figure 13.32. Similarly, a burst write cycle without auto-precharge is shown in figure 13.33, a burst write cycle for the same row address in figure 13.34, and a burst write cycle for different row addresses in figure 13.35.

When synchronous DRAM is read, there is a 2-cycle latency for the DMQn signal that performs the byte specification. As a result, when the READ command is issued in figure 13.30, if the Tc cycle is executed immediately, the DMQn signal specification for Td1 cycle data output cannot be carried out. Therefore, the CAS latency should not be set to 1.

When RAS down mode is set, if only accesses to the respective banks in area 3 are considered, as long as accesses to the same row address continue, the operation starts with the cycle in figure 13.30 or 13.33, followed by repetition of the cycle in figure 13.31 or 13.34. An access to a different area during this time has no effect. If there is an access to a different row address in the bank active state, after this is detected the bus cycle in figure 13.32 or 13.35 is executed instead of that in figure 13.31 or 13.34. In RAS down mode, too, both banks become inactive after a refresh cycle or after the bus is released as the result of bus arbitration.

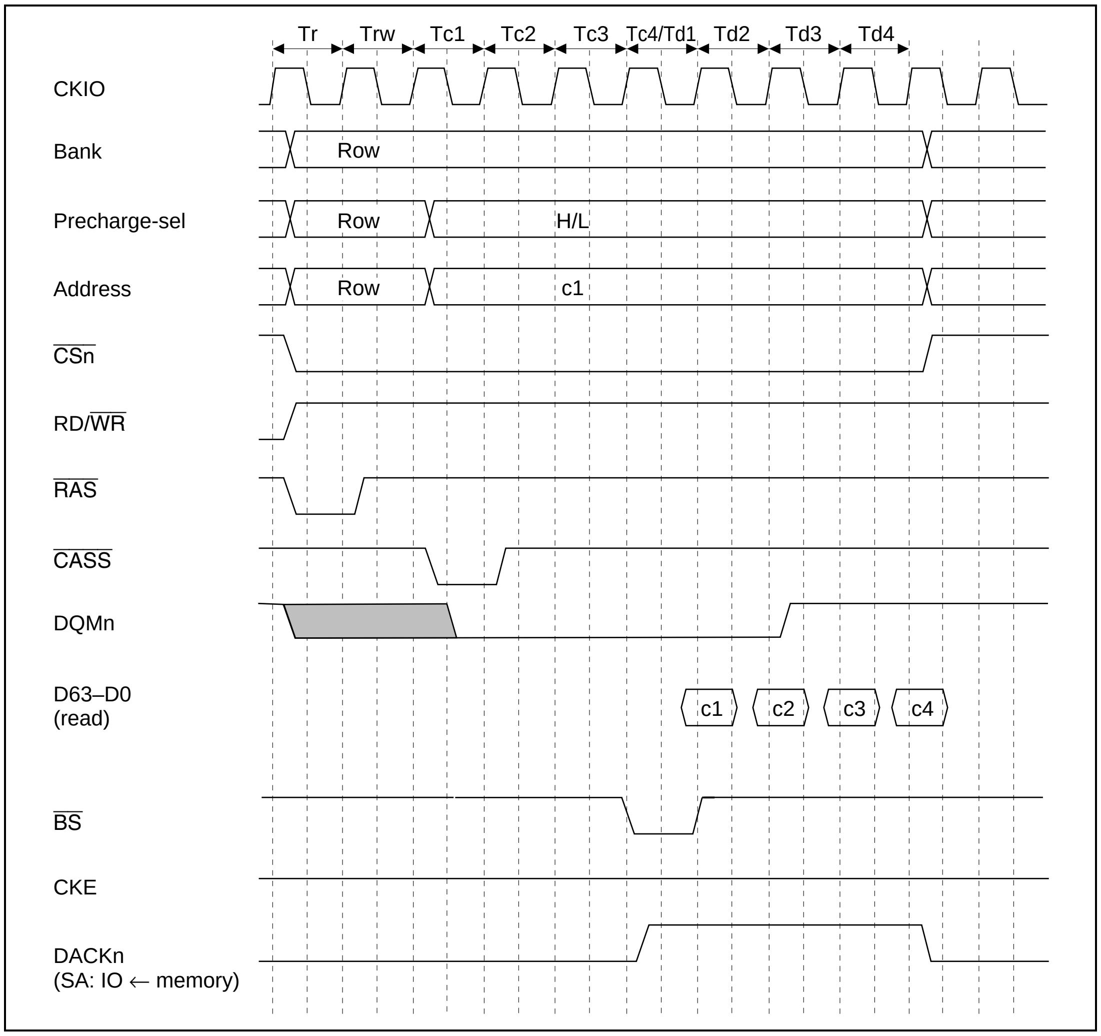


Figure 13.30 Burst Read Timing

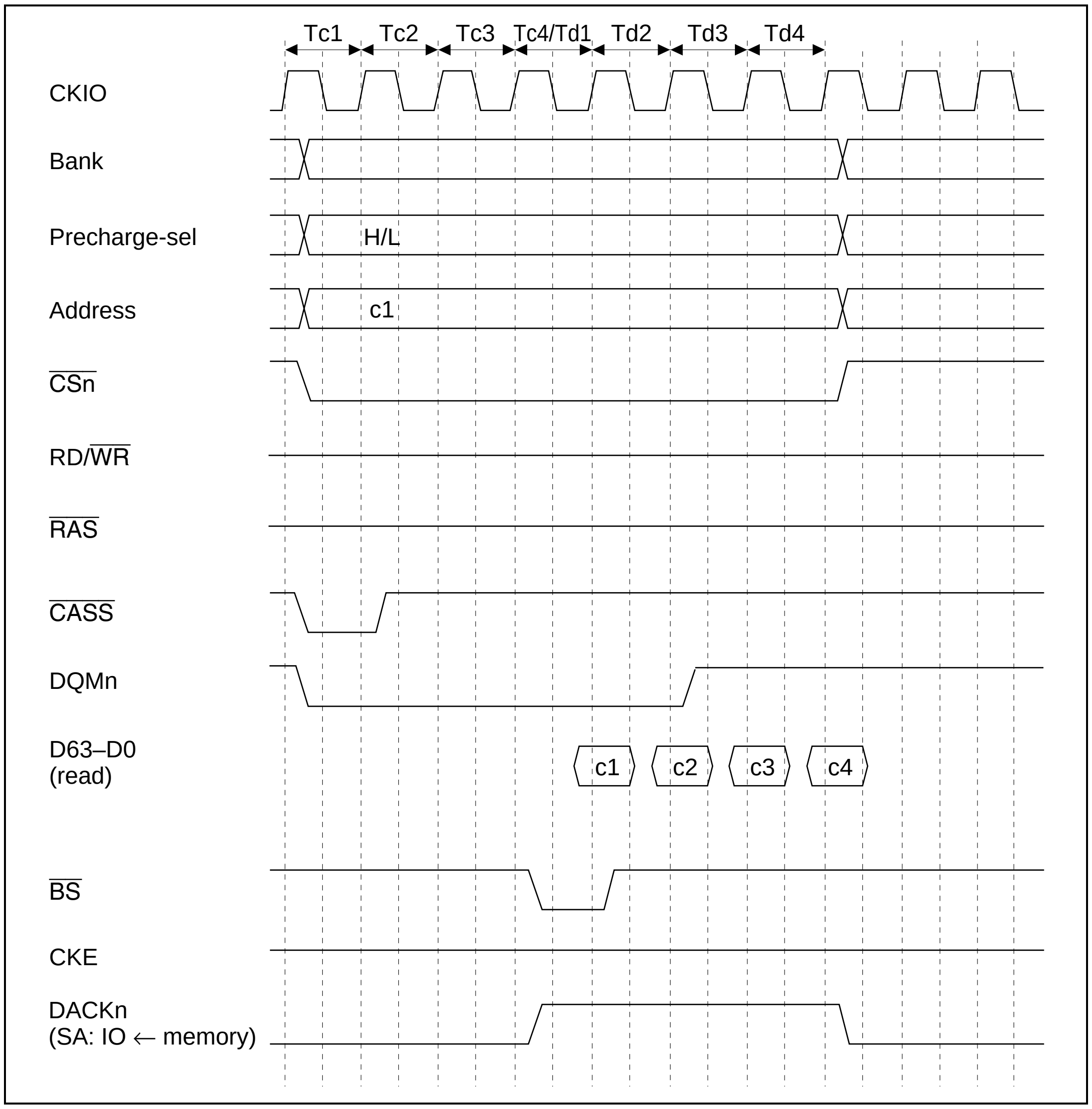


Figure 13.31 Burst Read Timing (RAS Down, Same Row Address)

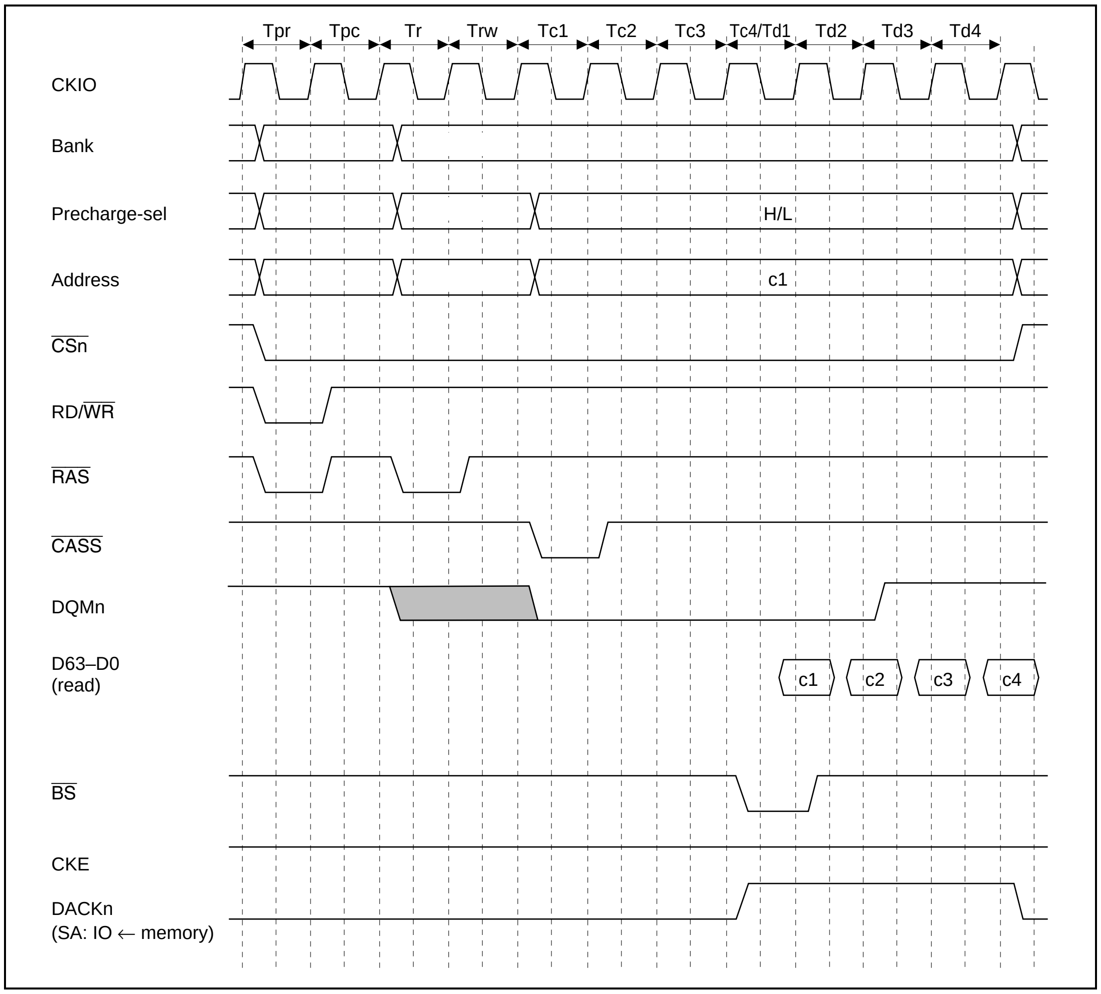


Figure 13.32 Burst Read Timing (RAS Down, Different Row Addresses)

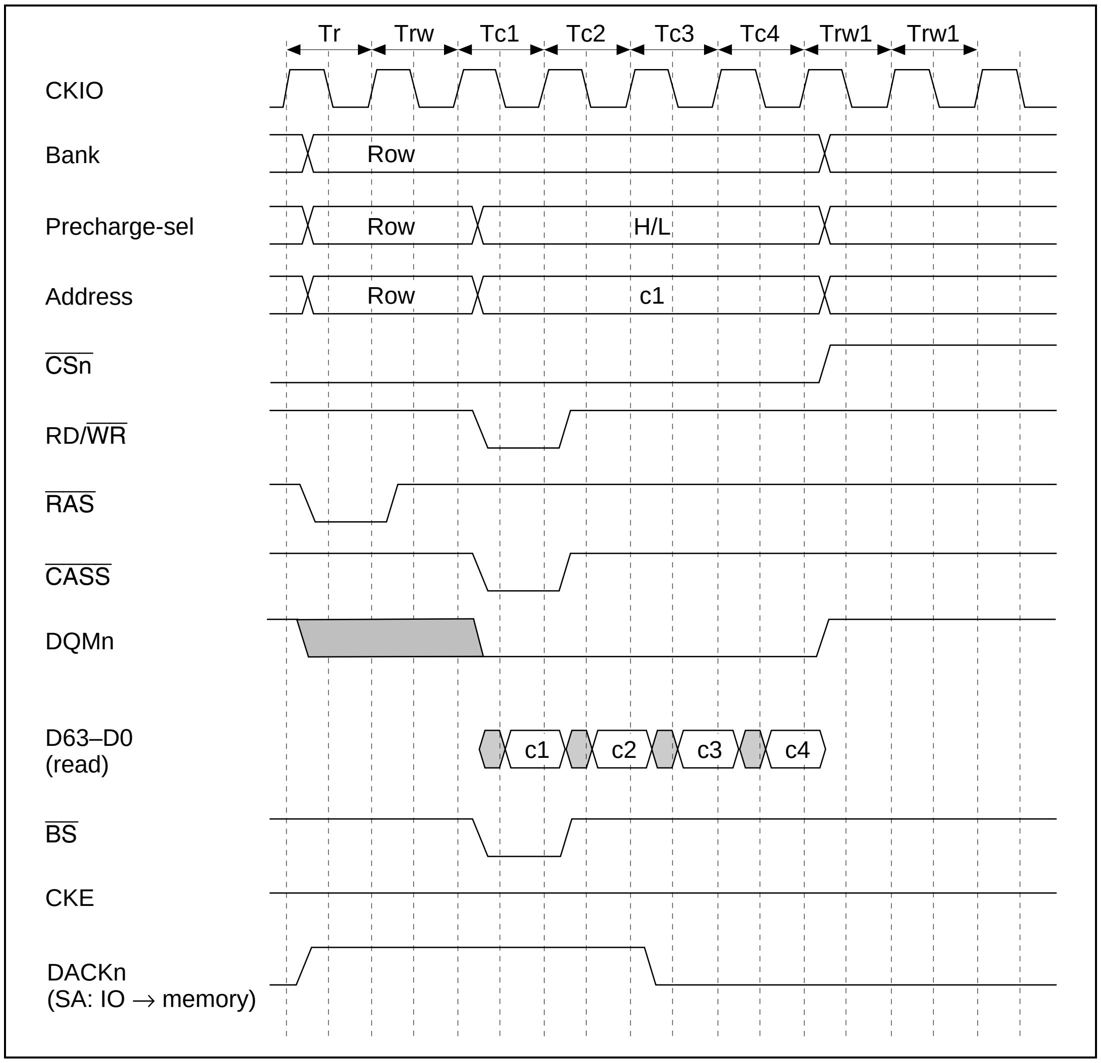
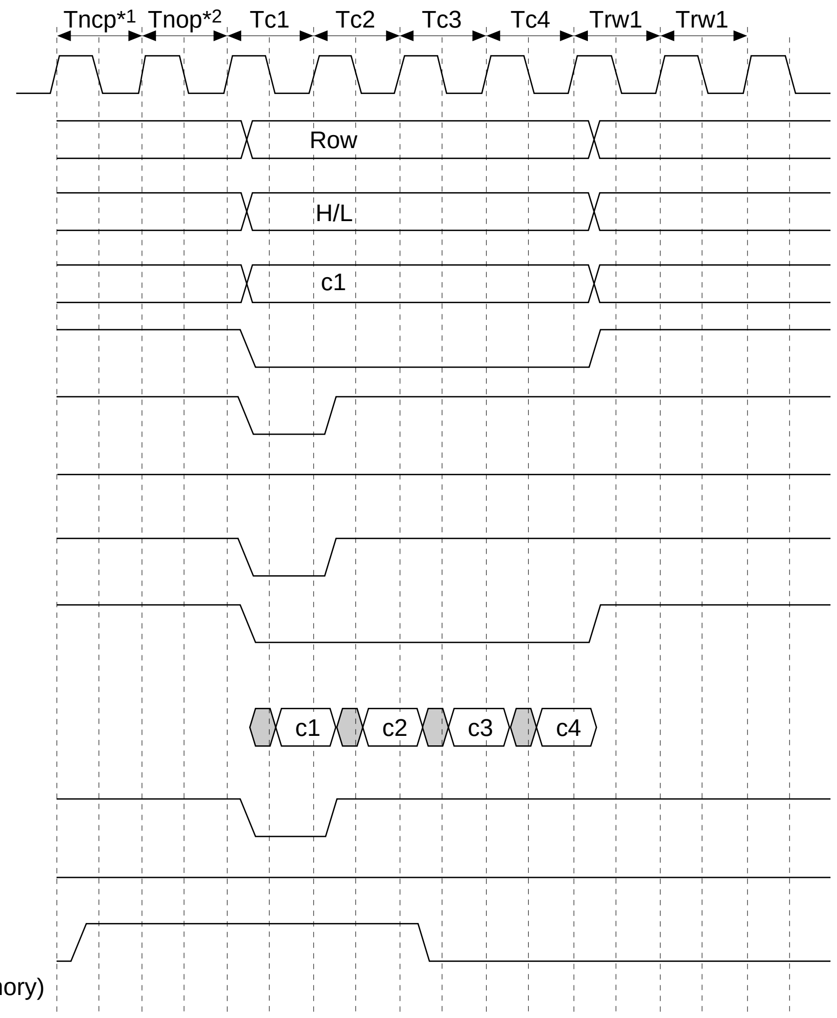


Figure 13.33   Burst Write Timing



Notes: 1. Tncp: DACK output start cycle (inserted only in the case of DACK output)
2. Tnop: Dummy cycle (always inserted)

Figure 13.34 Burst Write Timing (Same Row Address)

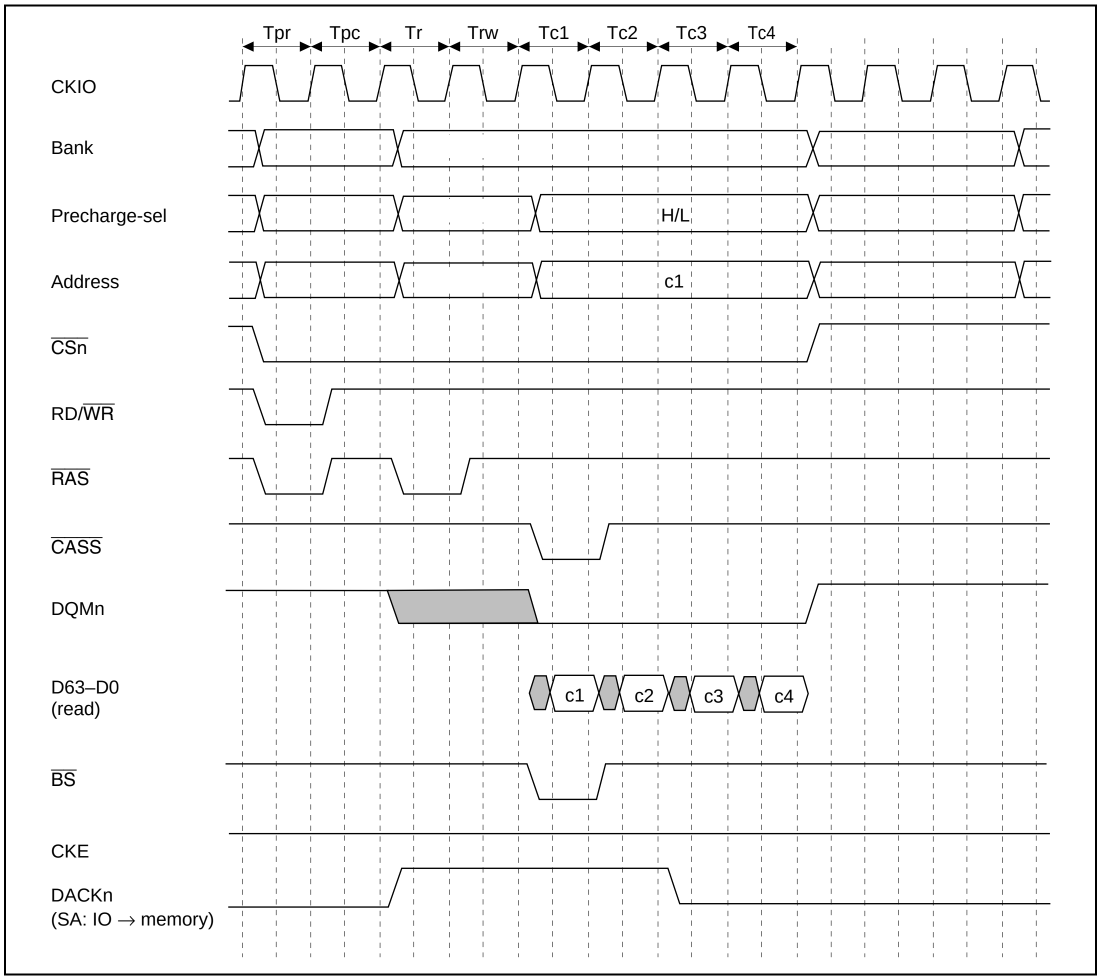


Figure 13.35 Burst Write Timing (Different Row Addresses)

Pipelined Access: When the RASD bit is set to 1 in MCR, pipelined access is performed between an access by the CPU and an access by the DMAC, or in the case of consecutive accesses by the DMAC, to provide faster access to synchronous DRAM. As synchronous DRAM is internally divided into two or four banks, after a READ or WRIT command is issued for one bank it is possible to issue a PRE, ACTV, or other command during the CAS latency cycle or data latch cycle, or during the data write cycle, and so shorten the access cycle.

When a read access is followed by another read access to the same row address, after a READ command has been issued, another READ command is issued before the end of the data latch cycle, so that there is read data on the data bus continuously. When an access is made to another row address and the bank is different, the PRE command or ACTV command can be issued during the CAS latency cycle or data latch cycle. If there are consecutive access requests for different row addresses in the same bank, the PRE command cannot be issued until the last-but-one data latch cycle. If a read access is followed by a write access, it may be possible to issue a PRE or ACT command, depending on the bank and row address, but since the write data is output at the same time as the WRIT command, the PRE, ACTV, and WRIT commands are issued in such a way that one or two empty cycles occur automatically on the data bus. Similarly, with a read access following a write access, or a write access following a write access, the PRE, ACTV, READ, or WRIT command is issued during the data write cycle for the preceding access; however, in the case of different row addresses in the same bank, a PRE command cannot be issued, and so in this case the PRE command is issued following the number of Trwl cycles specified by the TRWL bits in MCR, after the end of the last data write cycle.

Figure 13.36 shows a burst read cycle for a different bank and row address following a preceding burst read cycle.

Pipelined access is enabled only for consecutive access to area 3, and will be discontinued in the event of an access to another area. Pipelined access is also discontinued in the event of a refresh cycle, or bus release due to bus arbitration. The cases in which pipelined access is available are shown in table 13.16. In this table, "DMAC dual" indicates transfer in DMAC dual address mode, and "DMAC single", transfer in DMAC single address mode.

Table 13.16 Cycles in Which Pipelined Access Can Be Used

Preceding Access		Following Access					
		CPU		DMAC Dual		DMAC Single	
		Read	Write	Read	Write	Read	Write
CPU	Read	X	X	O	X	O	O
	Write	X	X	O	X	O	O
DMAC dual	Read	X	X	X	X	X	X
	Write	O	O	O	X	O	O
DMAC single	Read	O	O	X	X	O	O
	Write	O	O	O	X	O	O

O: Pipelined access possible

X: Pipelined access not possible

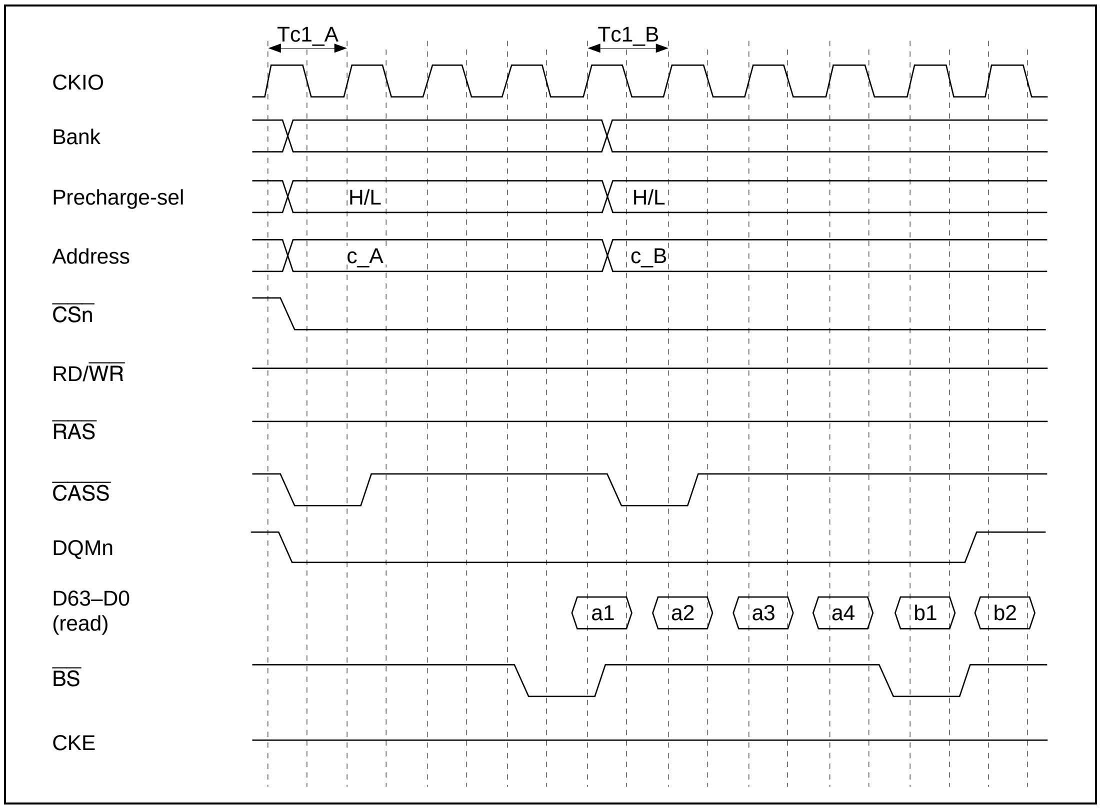


Figure 13.36 Burst Read Cycle for Different Bank and Row Address Following Preceding Burst Read Cycle

Refreshing: The bus state controller is provided with a function for controlling synchronous DRAM refreshing. Auto-refreshing can be performed by clearing the RMODE bit to 0 and setting the RFSH bit to 1 in MCR. If synchronous DRAM 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.

- Auto-Refreshing

  Refreshing is performed at intervals determined by the input clock selected by bits CKS2–CKS0 in RTCSR, and the value set in RTCOR. The value of bits CKS2–CKS0 in RTCOR should be set so as to satisfy the refresh interval specification for the synchronous DRAM used. First make the settings for RTCOR, RTCNT, and the RMODE and RFSH bits in MCR, then make the CKS2–CKS0 setting last of all. 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.

  First, an REF command is issued in the TRr cycle. After the TRr cycle, new command output cannot be performed for the duration of the number of cycles specified by bits TRAS2–TRAS0 in MCR plus the number of cycles specified by bits TRC2–TRC0 in MCR. The TRAS2–TRAS0 and TRC2–TRC0 bits must be set so as to satisfy the synchronous DRAM refresh cycle time specification (active/active command delay time).

  Auto-refreshing is performed in normal operation, in sleep mode, and in the case of a manual reset.

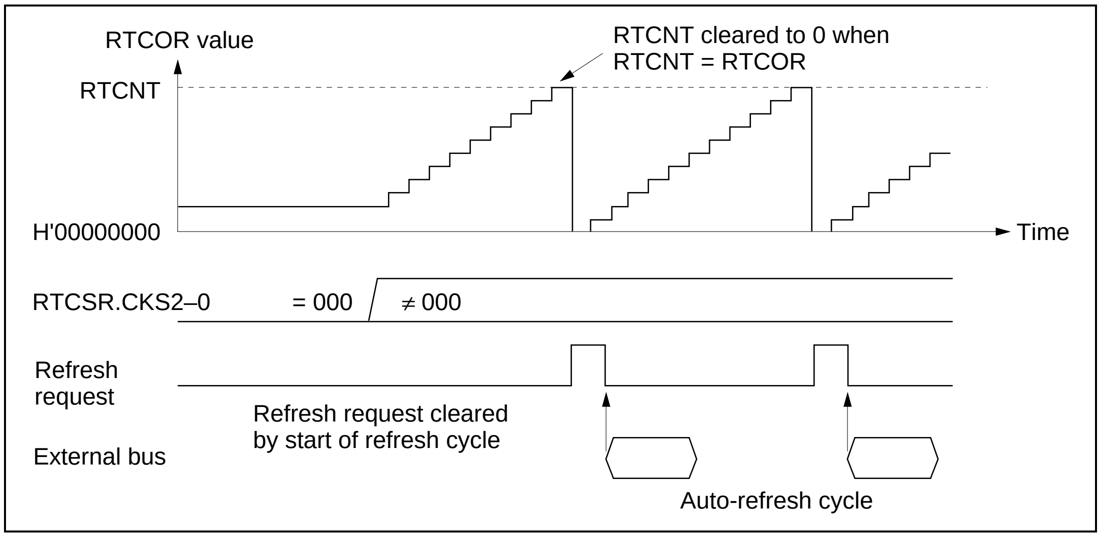


Figure 13.37 Auto-Refresh Operation

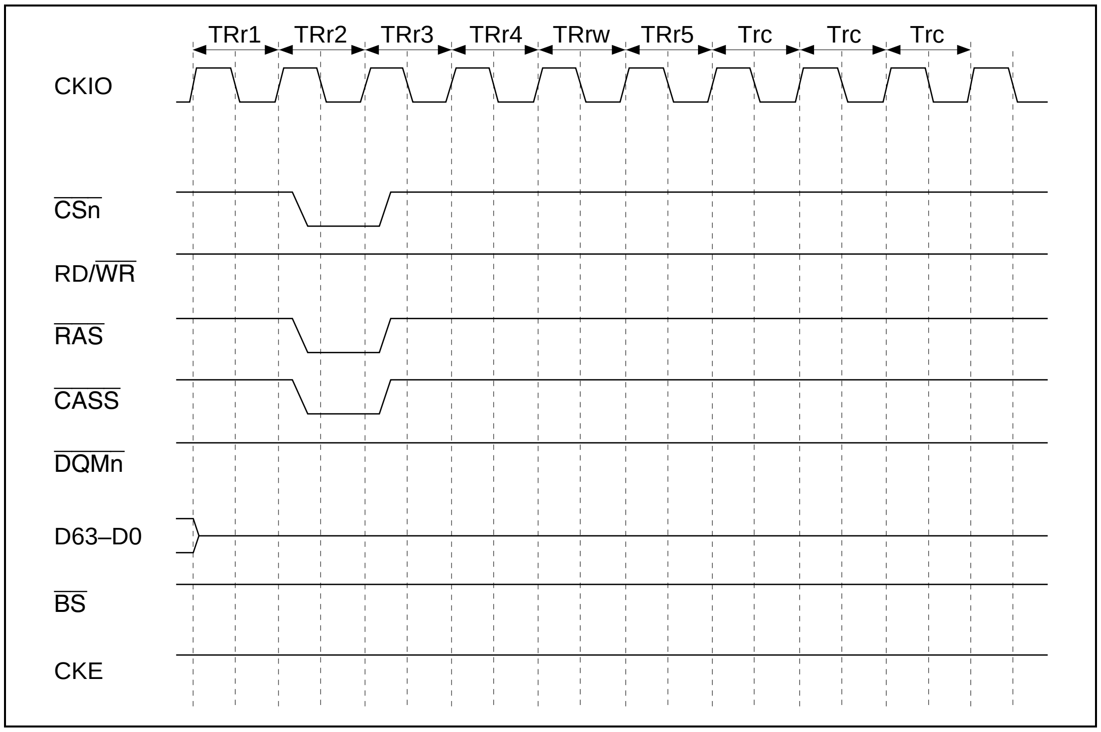


Figure 13.38 Synchronous DRAM Auto-Refresh Timing

- Self-Refreshing

  Self-refresh mode is a kind of standby mode in which the refresh timing and refresh addresses are generated within the synchronous DRAM. 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. Synchronous DRAM 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 bits TRC2–TRC0 in MCR. Self-refresh timing is shown in figure 13.39. 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 SH7750's standby function, and is maintained even after recovery from standby mode other than through a power-on reset.

In the 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 the case of a manual reset.

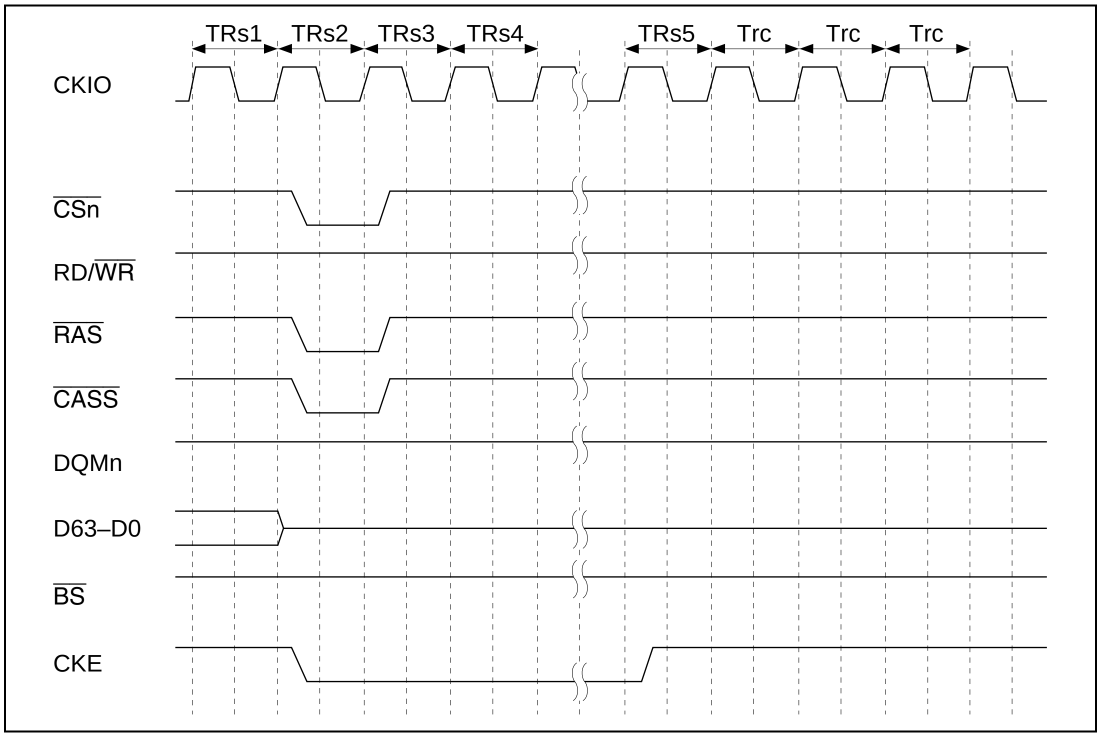


Figure 13.39 Synchronous DRAM Self-Refresh Timing

- 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 $\overline{\text{BACK}}$ pin is negated (driven high). Therefore, normal refreshing can be performed by having the $\overline{\text{BACK}}$ pin monitored by a bus master other than the SH7750 requesting the bus, or the bus arbiter, and returning the bus to the SH7750.

Power-On Sequence: In order to use synchronous DRAM, mode setting must first be performed after powering on. To perform synchronous DRAM initialization correctly, the bus state controller registers must first be set, followed by a write to the synchronous DRAM mode register. In synchronous DRAM 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 synchronous DRAM mode register by performing a write to address H'FF900000 + X for area 2 synchronous DRAM, and to address H'FF940000 + X for area 3 synchronous DRAM. In this operation the data is ignored, but the mode write is performed as a byte-size access. To set burst read/write, CAS latency 1 to 3, wrap type = sequential, and burst length 4 or 8, supported by the SH7750, arbitrary data is written by byte-size access to the following addresses.

Bus Width	CAS Latency	Area 2	Area 3
32	1	FF90004C	FF94004C
	2	FF90008C	FF94008C
	3	FF9000CC	FF9400CC
64	1	FF900090	FF940090
	2	FF900110	FF940110
	3	FF900190	FF940190

The value set in MCR.MRSET is used to select whether a precharge all banks command or a mode register setting command is issued. The timing for the precharge all banks command is shown in figure 13.40 (1), and the timing for the mode register setting command in figure 13.40 (2).

Before mode register, a 200 µs idle time (depending on the memory manufacturer) must be guaranteed after the power required for the synchronous DRAM is turned on. If the reset signal pulse width is greater than this idle time, there is no problem in making the precharge all banks setting immediately.

First, a precharge all banks (PALL) command is issued in the TRp1 cycle by performing a write to address H'FF900000 + X or H'FF940000 + X while MCR.MRSET = 0. Next, the number of dummy auto-refresh cycles specified by the manufacturer (usually 8) or more must be executed. This is 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 change the RTCOR register value 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 synchronous DRAM used for auto-refreshing is not initialized, and so the cycle must always be an auto-refresh cycle. After auto-refreshing has been executed at least the prescribed number of times, a mode register setting command is issued in the TMw1 cycle by setting MCR.MRSET to 1 and performing a write to address H'FF900000 + X or H'FF940000 + X.

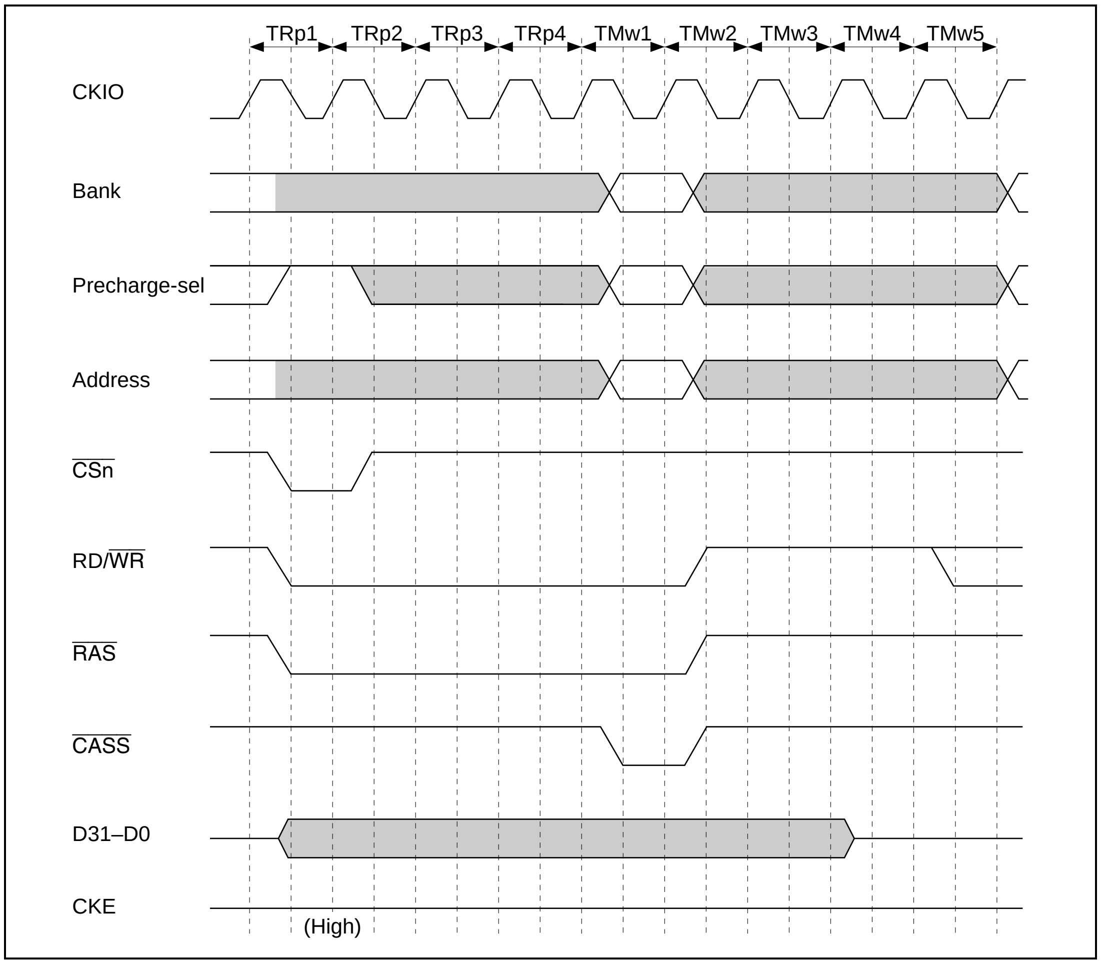


Figure 13.40 (1) Synchronous DRAM Mode Write Timing

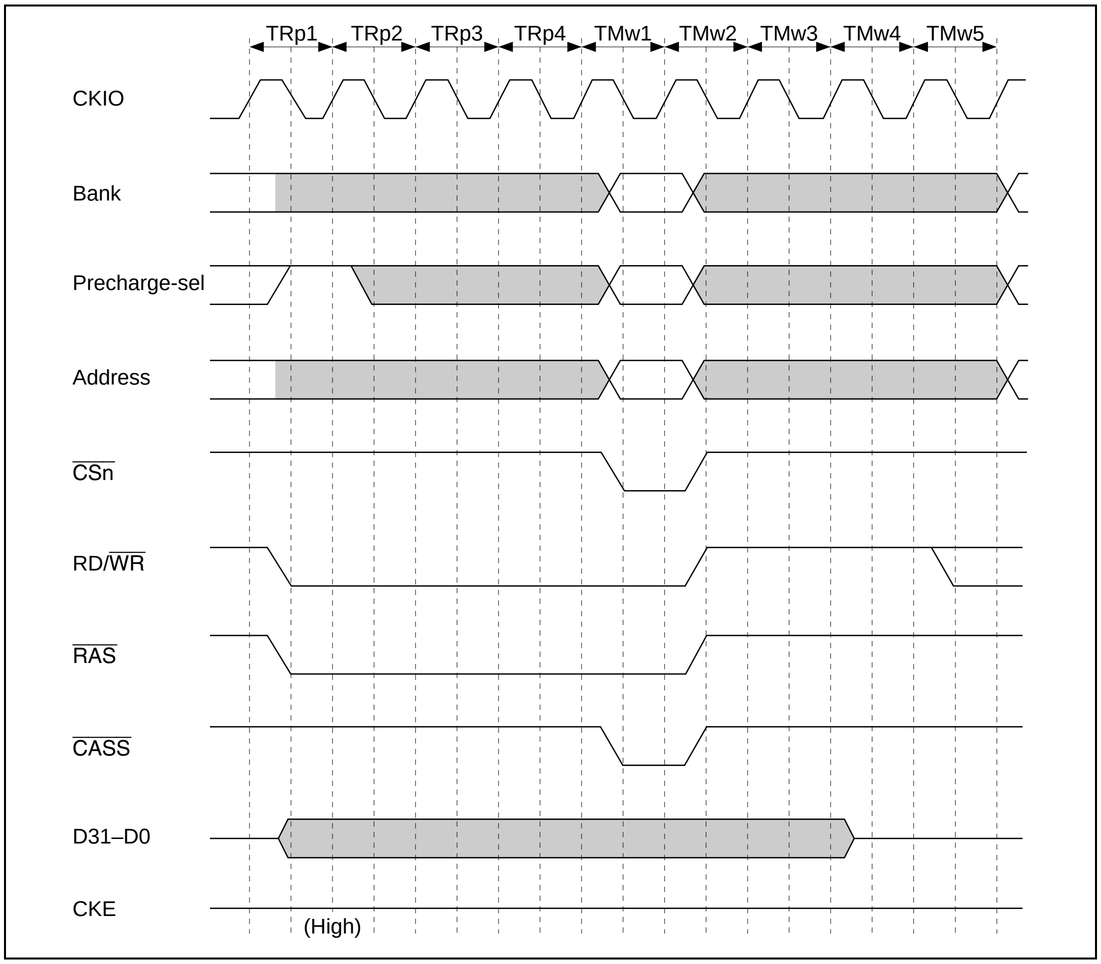


Figure 13.40 (2) Synchronous DRAM Mode Write Timing

13.3.6 Burst ROM Interface

Setting bits A0BST2–A0BST0, A5BST2–A5BST0, and A6BST2–A6BST0 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 burst access function. The timing for burst access to burst ROM is shown in figure 13.41. Two wait cycles are set. Basically, access is performed in the same way as for normal space, but when the first cycle ends, 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, 16, or 32 with bits A0BST2–A0BST0, A5BST2–A5BST0, or A6BST2–A6BST0. When 16-bit ROM is connected, 4, 8, or 16 can be set in the same way. When 32-bit ROM is connected, 4 or 8 can be set.

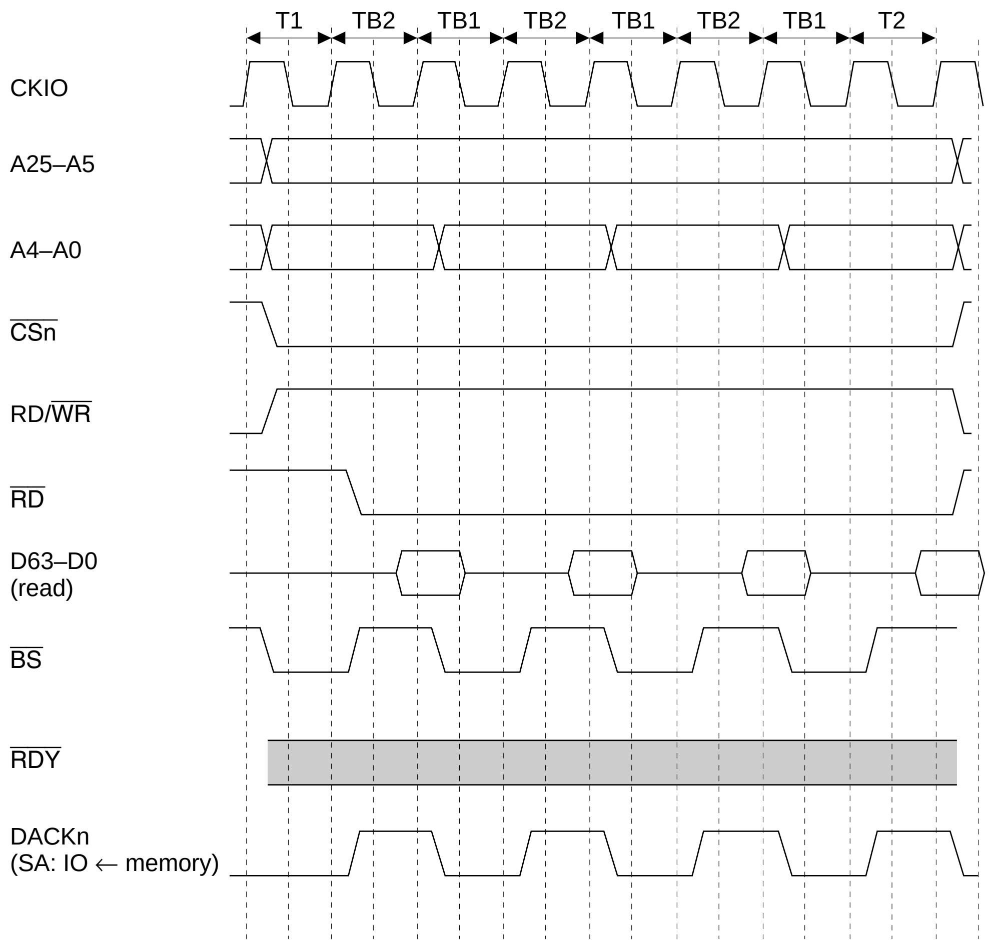
$\overline{\text{RDY}}$ pin sampling is always performed when one or more wait states are set.

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 13.42.

In a ROM write operation, a basic bus cycle (write) is performed.

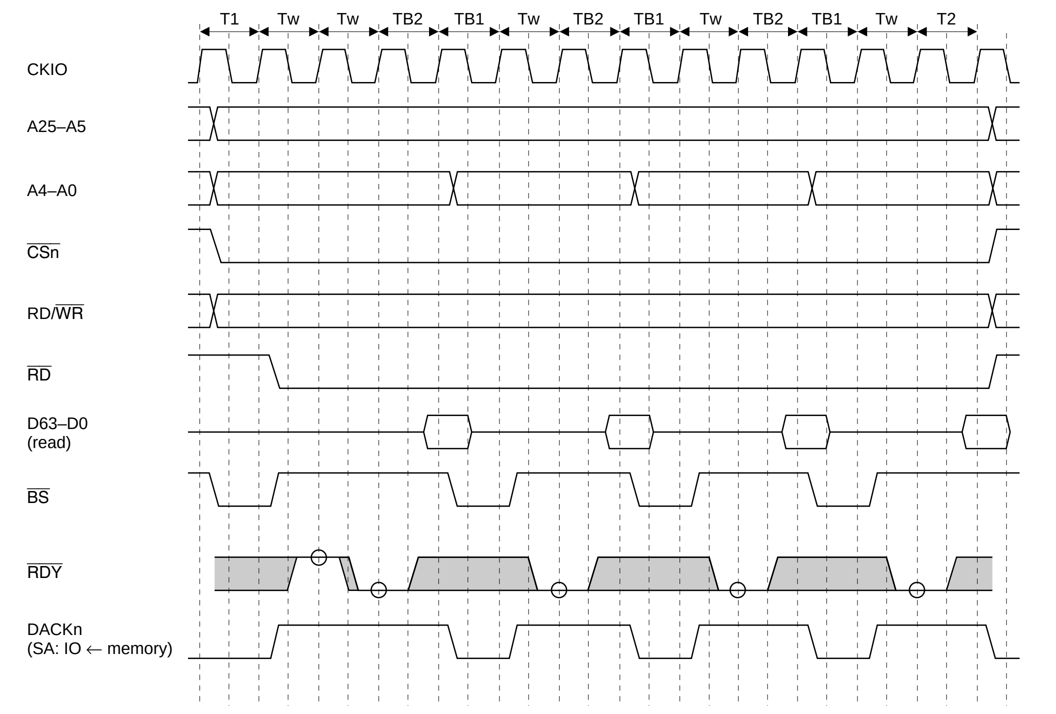
Cache fill or copy-back reads and writes are performed consecutively for a total of 32 bytes according to the set bus width. The first access is performed on the data for which there was an access request, and the remaining accesses are performed on the data at the 32-byte boundary. The bus is not released during this period.

Figure 13.43 shows the timing when a burst ROM setting is made, and setup/hold is specified in WCR3.



Note: For a write cycle, a basic bus cycle (write cycle) is performed.

Figure 13.41 Burst ROM Basic Access Timing



Note: For a write cycle, a basic bus cycle (write cycle) is performed.

Figure 13.42 Burst ROM Wait Access Timing

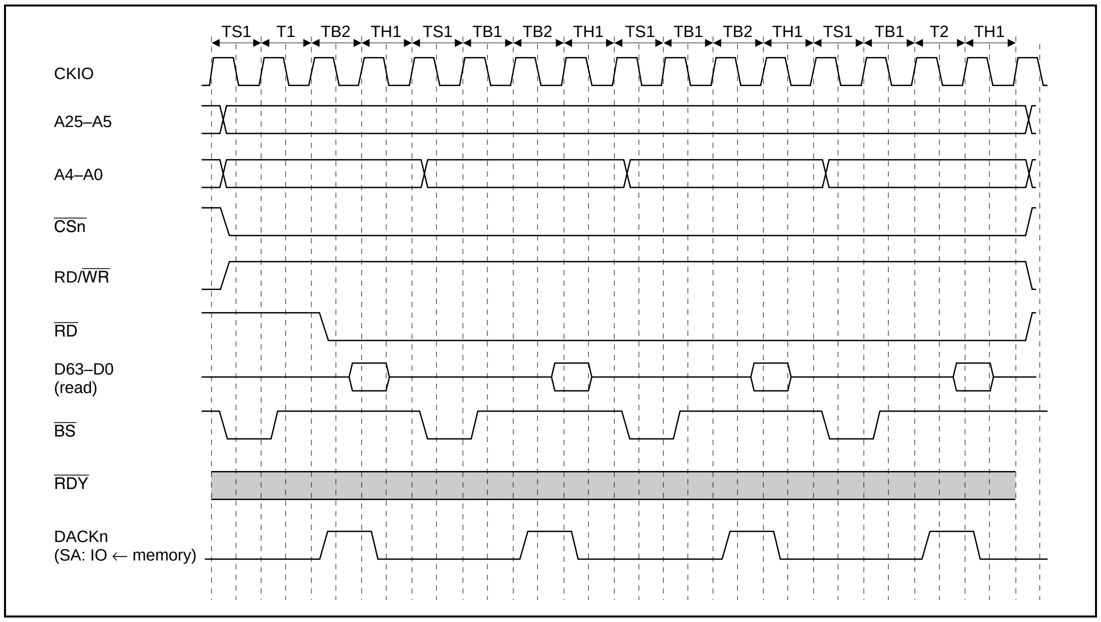


Figure 13.43 Burst ROM Wait Access Timing

13.3.7 PCMCIA Interface

In the SH7750, setting the A56PCM bit in BCR1 to 1 makes the bus interface for external space areas 5 and 6 an IC memory card interface or I/O card interface as stipulated in JEIDA specification version 4.2 (PCMCIA2.1).

Figure 13.44 shows an example of PCMCIA card connection to the SH7750. 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 SH7750's bus interface and the PCMCIA cards.

As operation in big-endian mode is not explicitly stipulated in the JEIDA/PCMCIA specifications, the SH7750 supports only a little-endian mode PCMCIA interface.

The PCMCIA interface can only be accessed when the MMU is used. PCMCIA memory space can be set in MMU page units, and there is a choice of 8-bit common memory, 16-bit common memory, 8-bit attribute memory, 16-bit attribute memory, 8-bit I/O space, 16-bit I/O space, or dynamic bus sizing. The setting is made with bits SA2–SA0 in PTEA.

SA2	SA1	SA0	Description
0	0	0	Reserved (Setting prohibited)
		1	Dynamic I/O bus sizing
	1	0	8-bit I/O space
		1	16-bit I/O space
1	0	0	8-bit common memory
		1	16-bit common memory
	1	0	8-bit attribute memory
		1	16-bit attribute memory

Wait cycles in a bus access can be selected with the TC bit in PTEA. When TC is cleared to 0, bits A5W2–A5W0 in wait control register 2 (WCR2) and bits A5PCW1–A5PCW0, A5TED2–A5TED0, and A5TEH2–A5TEH0 in the PCMCIA control register (PCR) are selected. When TC is set to 1, bits A6W2–A6W0 in WCR2 and bits A6PCW1–A6PCW0, A6TED2–A6TED0, and A6TEH2–A6TEH0 in PCR are selected.

AnPCW1–AnPCW0 specify the number of wait states to be inserted in a low-speed bus cycle; a value of 0, 15, 30, or 50 can be set, and this value is added to the number of wait states for insertion specified by WCR2. AnTED2–AnTED0 can be set to a value from 0 to 15, enabling the address, $\overline{\text{CS}}$, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, and $\overline{\text{REG}}$ setup times with respect to the $\overline{\text{RD}}$ and $\overline{\text{WE1}}$ signals to be secured. AnTEH2–AnTEH0 can also be set to a value from 0 to 15, enabling the address, $\overline{\text{CS}}$, $\overline{\text{CE2A}}$, $\overline{\text{CE2B}}$, and $\overline{\text{REG}}$ write data hold times with respect to the $\overline{\text{RD}}$ and $\overline{\text{WE1}}$ signals to be secured.

Wait cycles between cycles are set with bits A5IW2–A5IW0 and A6IW2–A6IW0 in wait control register 1 (WCR1). The inter-cycle write cycles selected depend only on the area accessed (area 5 or 6): when area 5 is accessed, bits A5IW2–A5IW0 are selected, and when area 6 is accessed, bits A6IW2–A6IW0 are selected.

Cache fill or copy-back reads and writes are performed consecutively for a total of 32 bytes according to the set bus width. The first access is performed on the data for which there was an access request, and the remaining accesses are performed on the data at the 32-byte boundary. The bus is not released during this period.

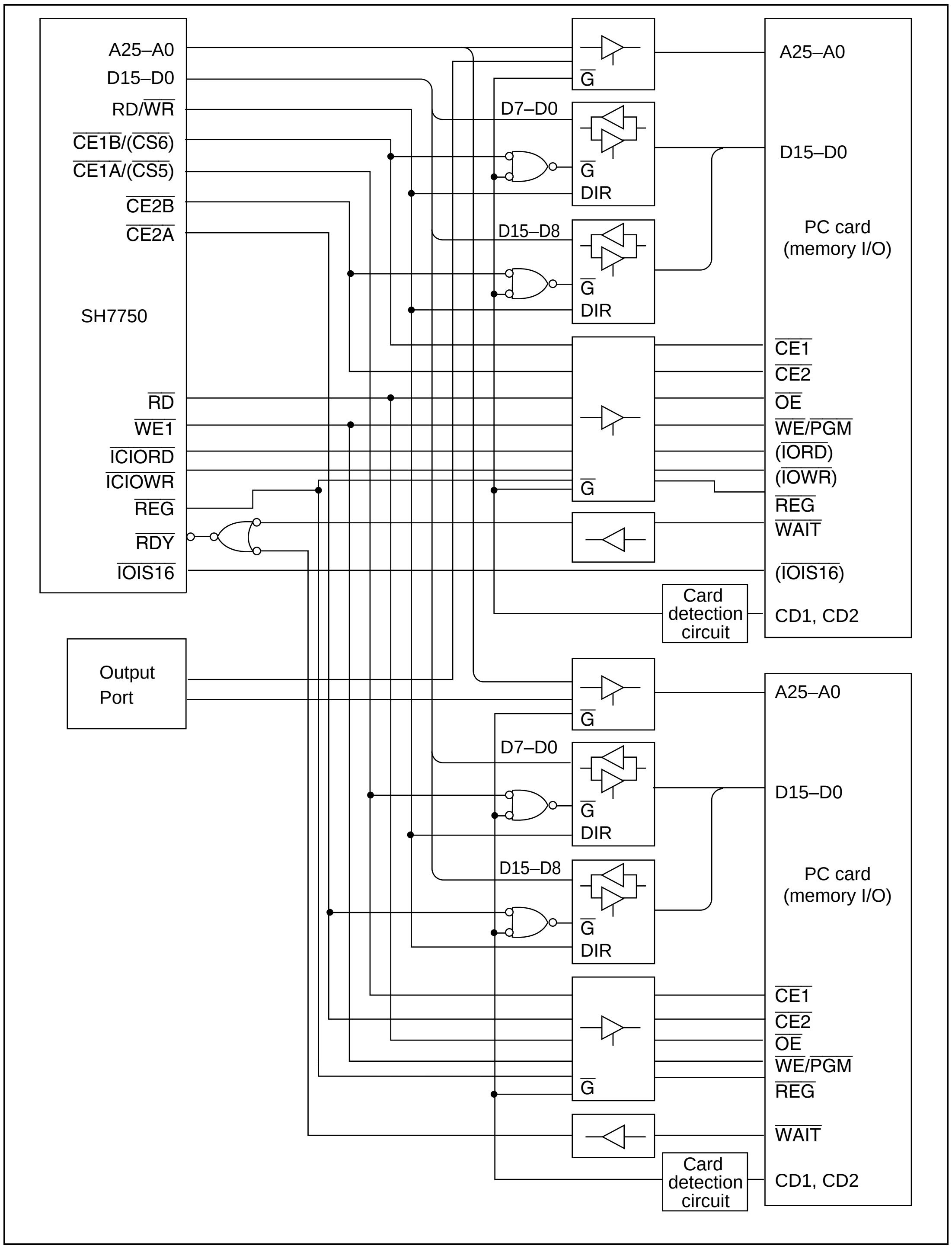


Figure 13.44 Example of PCMCIA Interface

Memory Card Interface Basic Timing: Figure 13.45 shows the basic timing for the PCMCIA IC memory card interface, and figure 13.46 shows the PCMCIA memory bus wait timing.

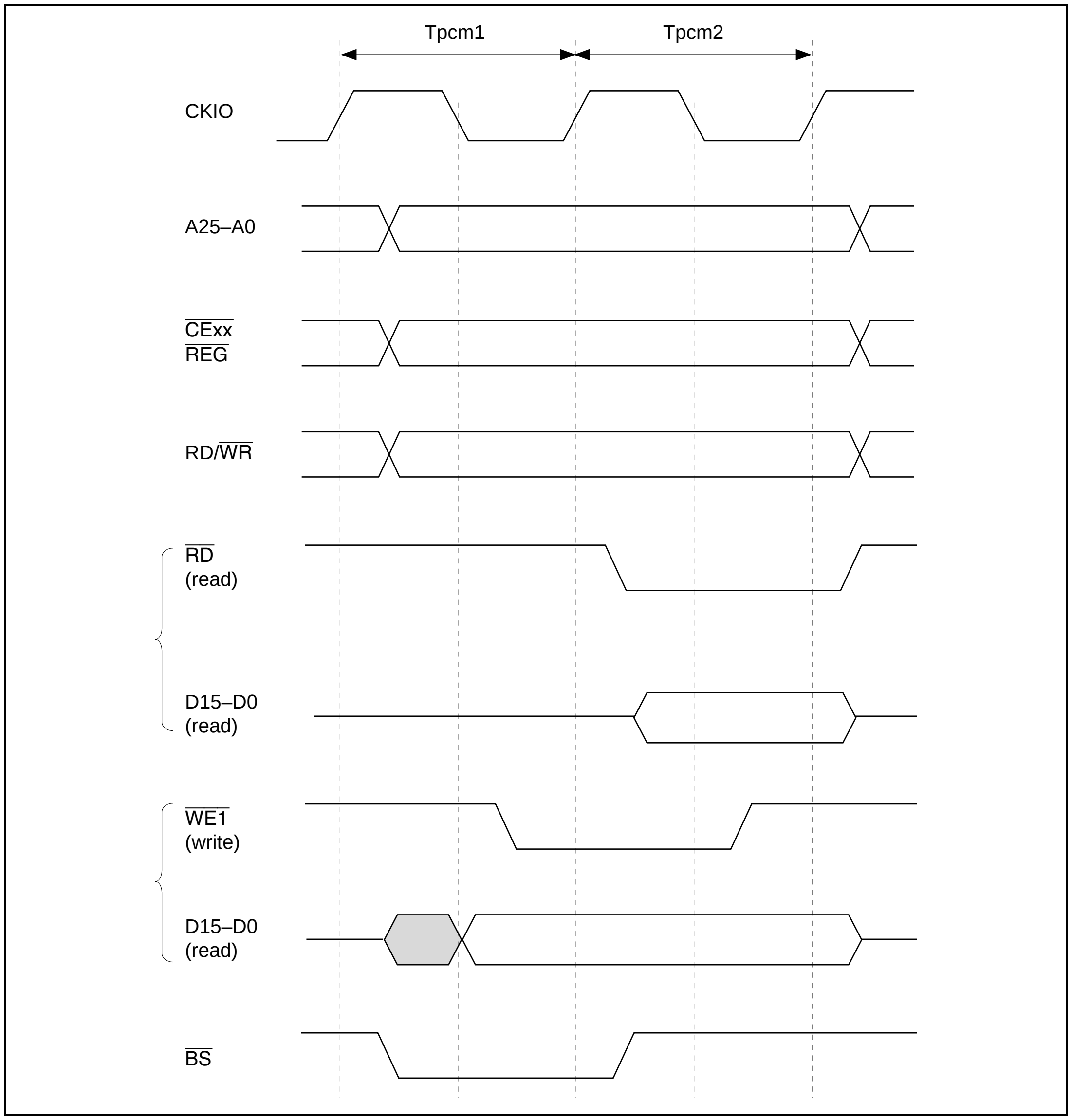


Figure 13.45 Basic Timing for PCMCIA Memory Card Interface

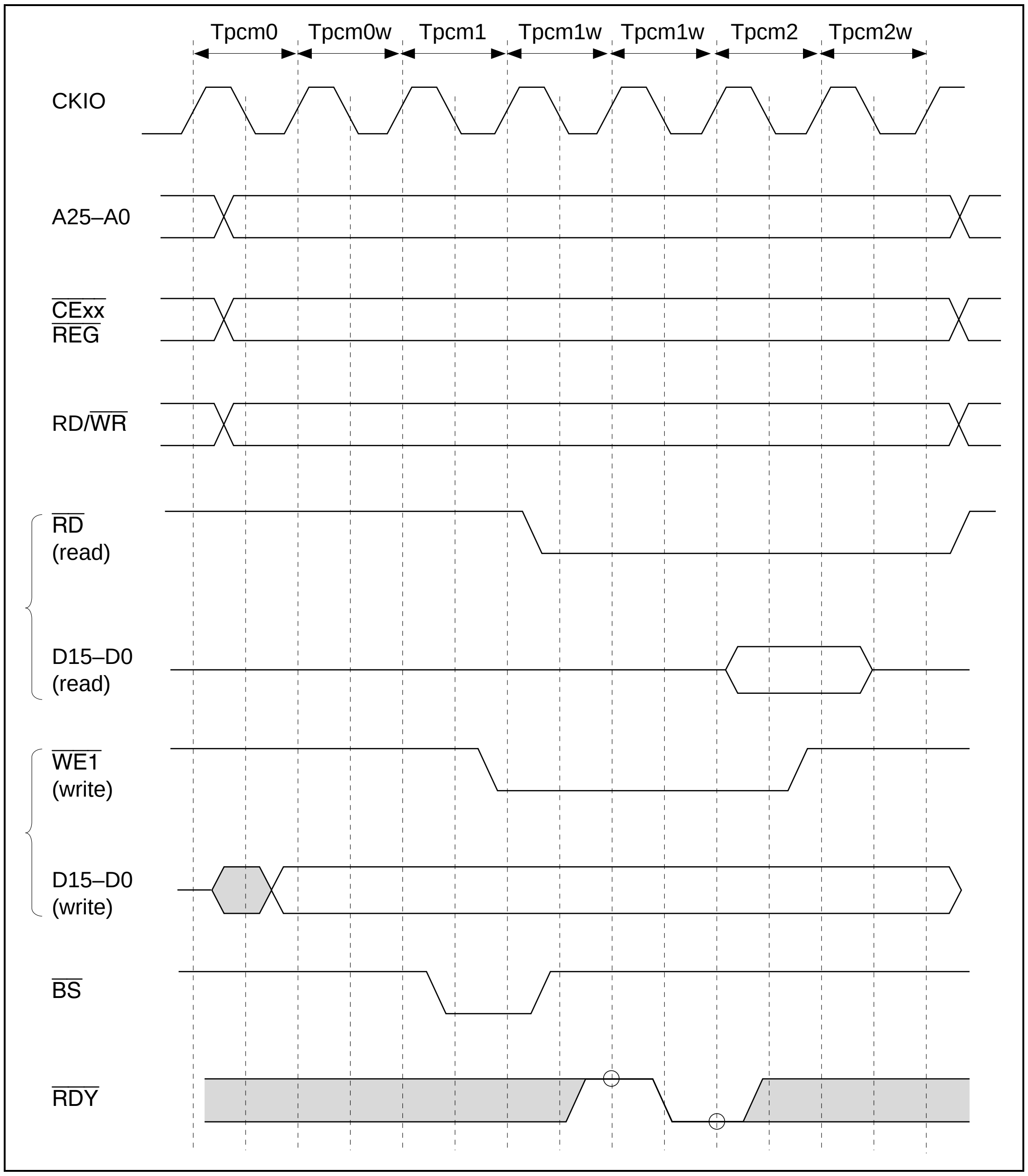


Figure 13.46 Wait Timing for PCMCIA Memory Card Interface

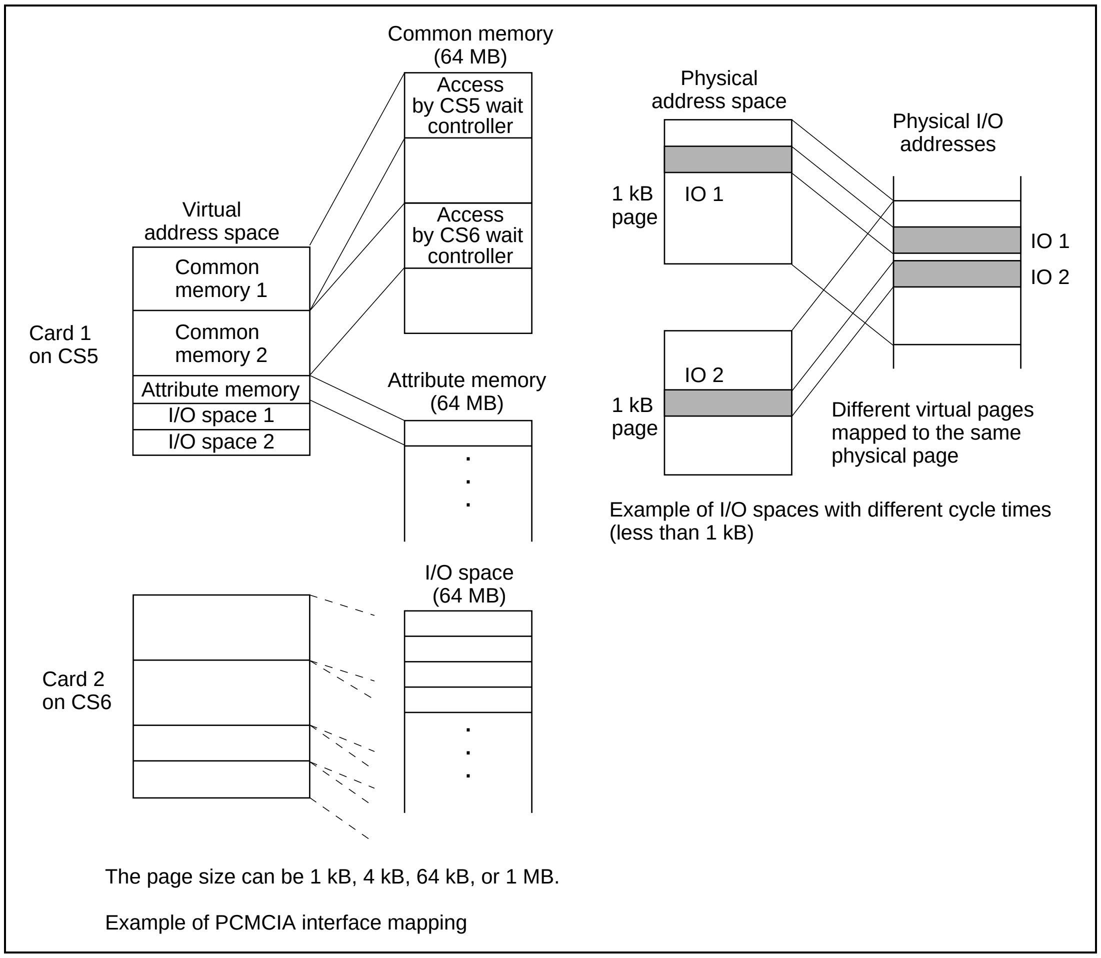

Common memory (64 MB)
Access by CS5 wait controller
Access by CS6 wait controller
Virtual address space
Common memory 1
Common memory 2
Attribute memory
I/O space 1
I/O space 2
Card 1 on CS5
Attribute memory (64 MB)
I/O space (64 MB)
Card 2 on CS6
Physical address space
Physical I/O addresses
1 kB page
IO 1
IO 2
Different virtual pages mapped to the same physical page
Example of I/O spaces with different cycle times (less than 1 kB)
The page size can be 1 kB, 4 kB, 64 kB, or 1 MB.
Example of PCMCIA interface mapping

Figure 13.47 PCMCIA Space Allocation

I/O Card Interface Timing: Figures 13.48 and 13.49 show the timing for the PCMCIA I/O card interface.

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, 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 13.50 shows the basic timing for dynamic bus sizing.

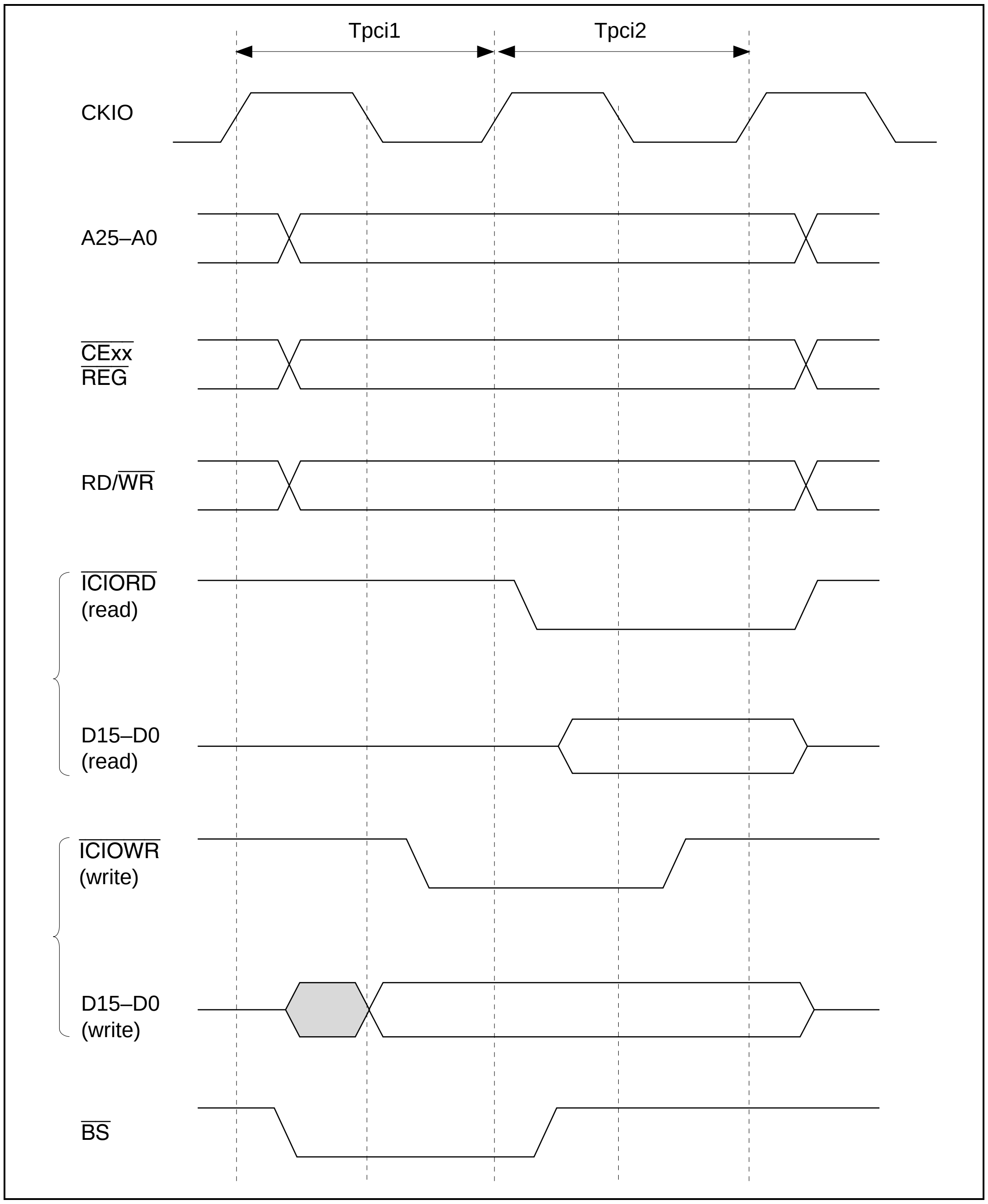


Figure 13.48 Basic Timing for PCMCIA I/O Card Interface

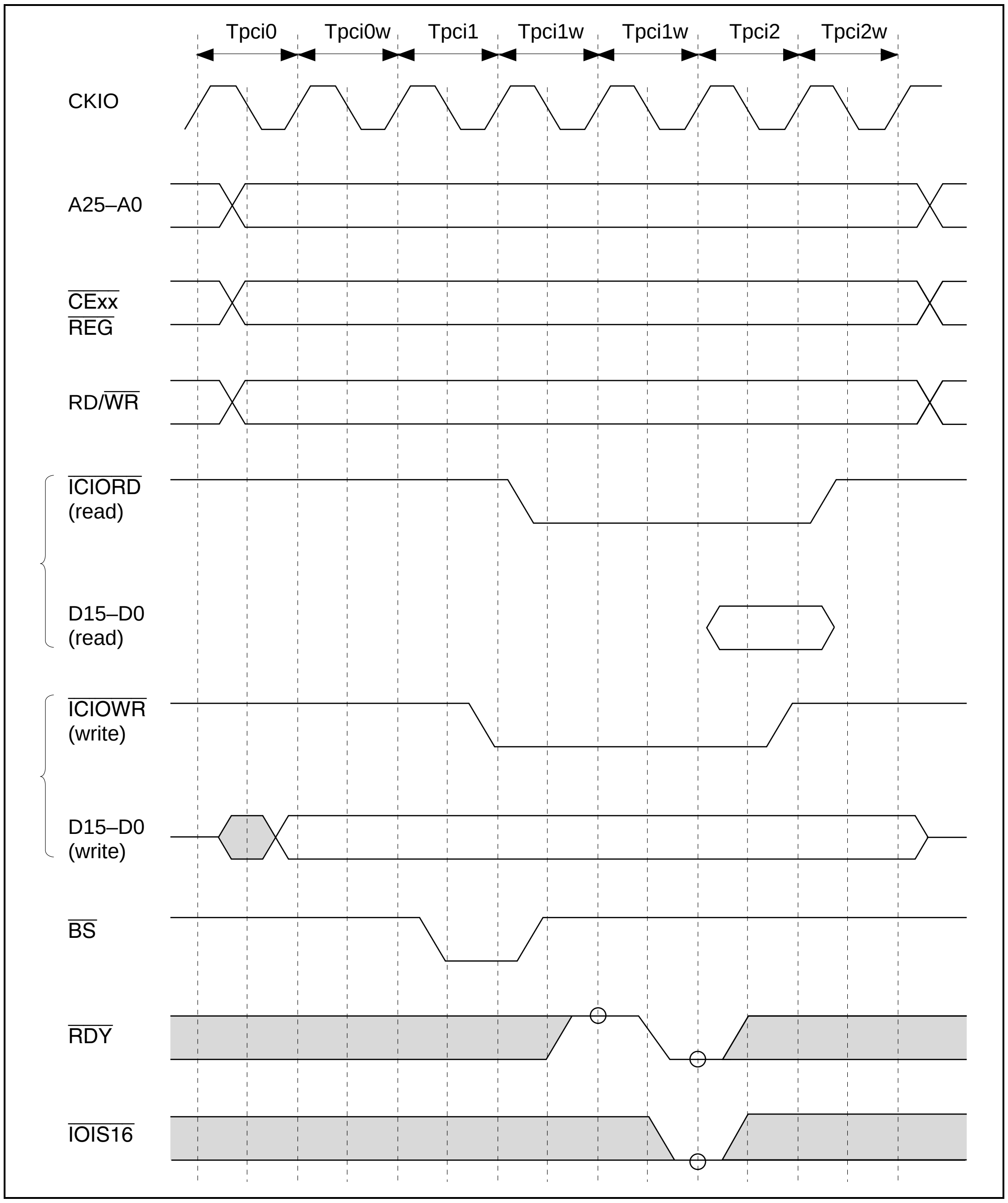


Figure 13.49 Wait Timing for PCMCIA I/O Card Interface

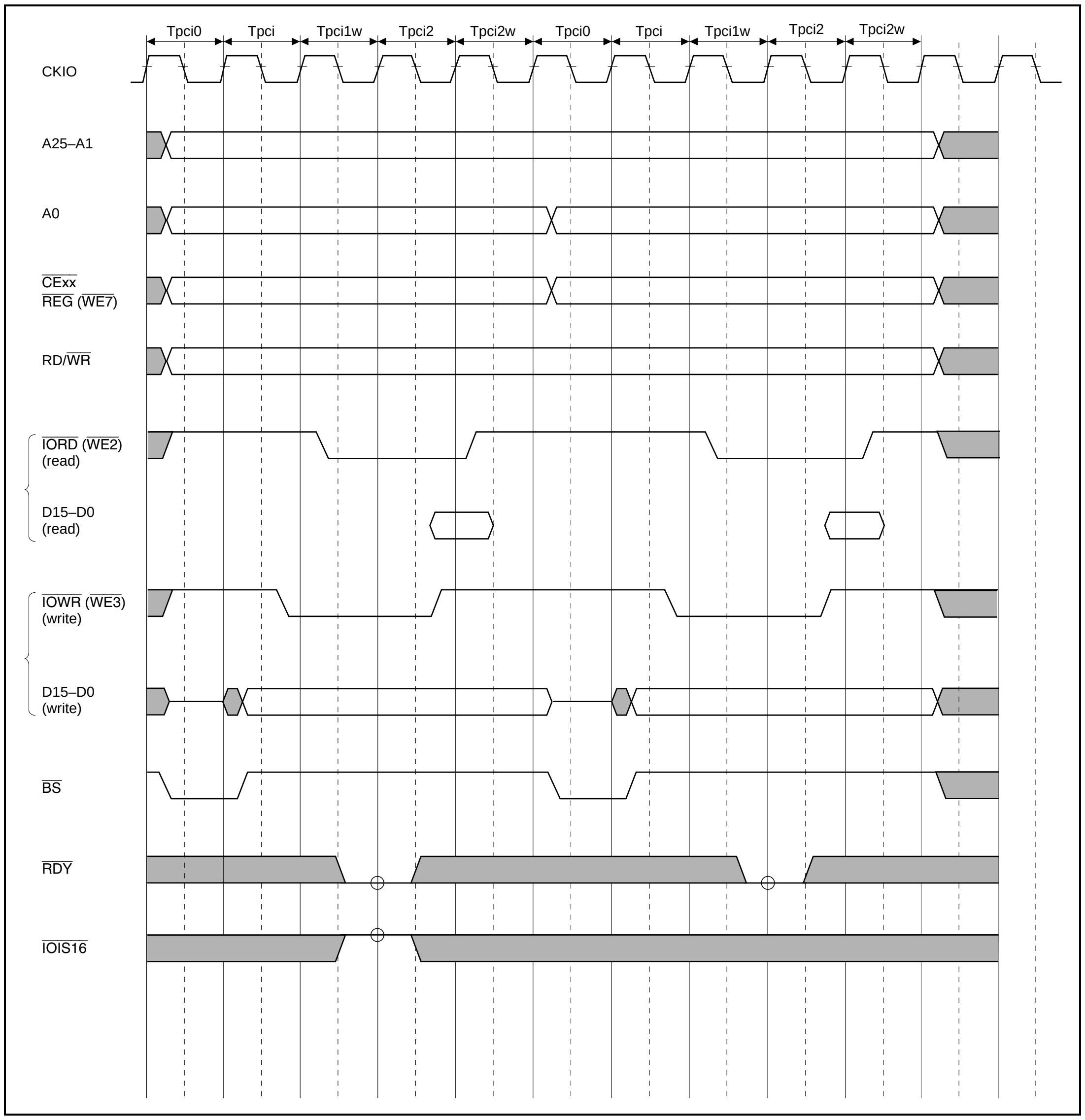


Figure 13.50 Dynamic Bus Sizing Timing for PCMCIA I/O Card Interface

13.3.8 MPX Interface

If the MD6 pin is set to 0 in a power-on reset, the MPX interface for normal memory is selected for area 0. The MPX interface is selected for areas 1 to 6 by means of the MPX bit in BCR1. The MPX interface offers a multiplexed address/data type bus protocol, and permits easy connection to an external memory controller chip that uses a single 32-bit multiplexed address/data bus. The address is output to D25–D0, and the access size to D63–D61.

For details of access sizes and data alignment, see section 13.3.1, Endian/Access Size and Data Alignment.

The address signals output at A25–A0 are undefined.

Cache fill or copy-back reads and writes are performed consecutively for a total of 32 bytes according to the set bus width. The first access is performed on the data for which there was an access request, and the remaining accesses are performed on the data at the 32-byte boundary. The bus is not released during this period.

D63	D62	D61	Access Size
0	0	0	Byte
		1	Word
	1	0	Longword
		1	Quadword
1	X	X	32-byte burst

X: Don't care

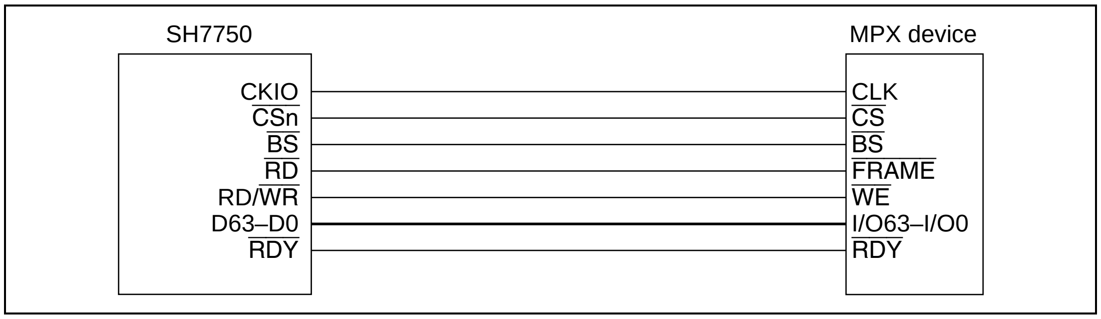


Figure 13.51 Example of 64-Bit Data Width MPX Connection

The MPX interface timing is shown below.

When the MPX interface is used for areas 1 to 6, a bus size of 32 or 64 bits should be specified in BCR2.

For wait control, waits specified by WCR2 and wait insertion by means of the $\overline{\text{RDY}}$ pin can be used.

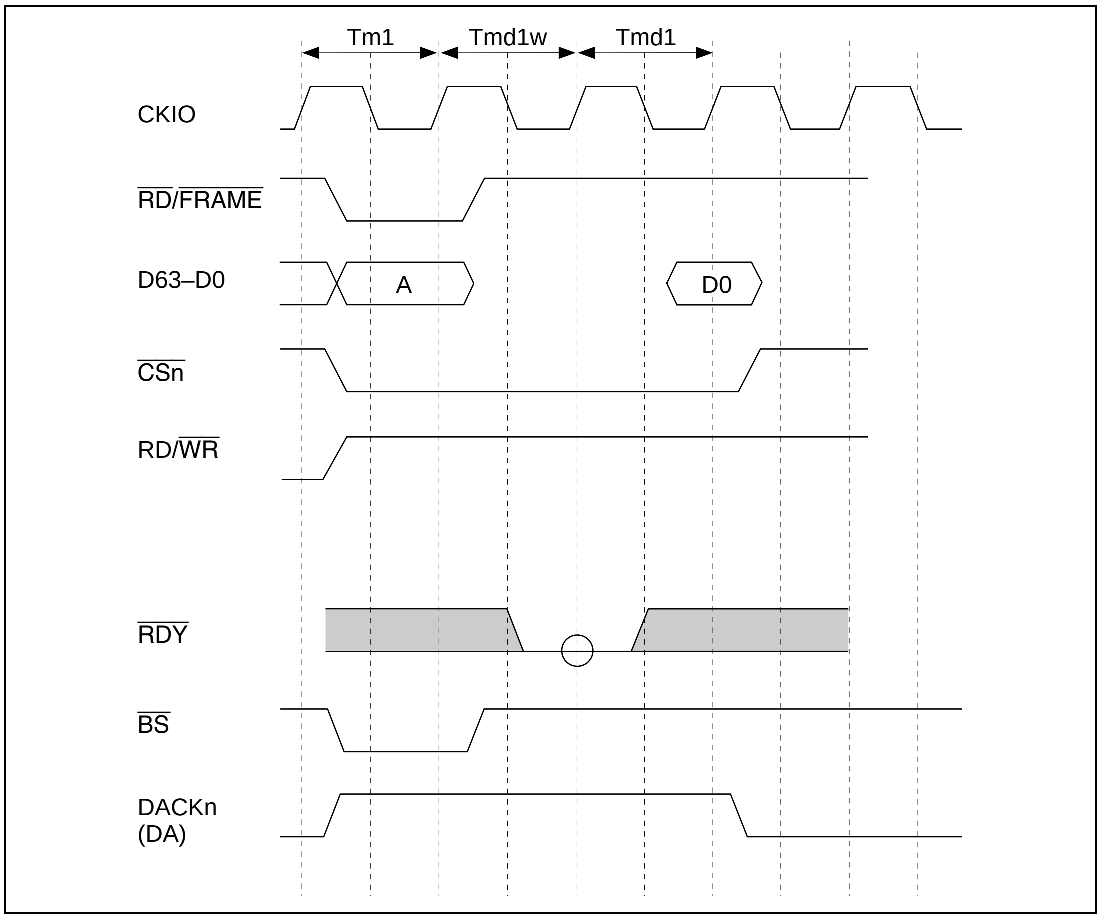


Figure 13.52 MPX Interface Timing 1 (Single Read Cycle, No Wait)

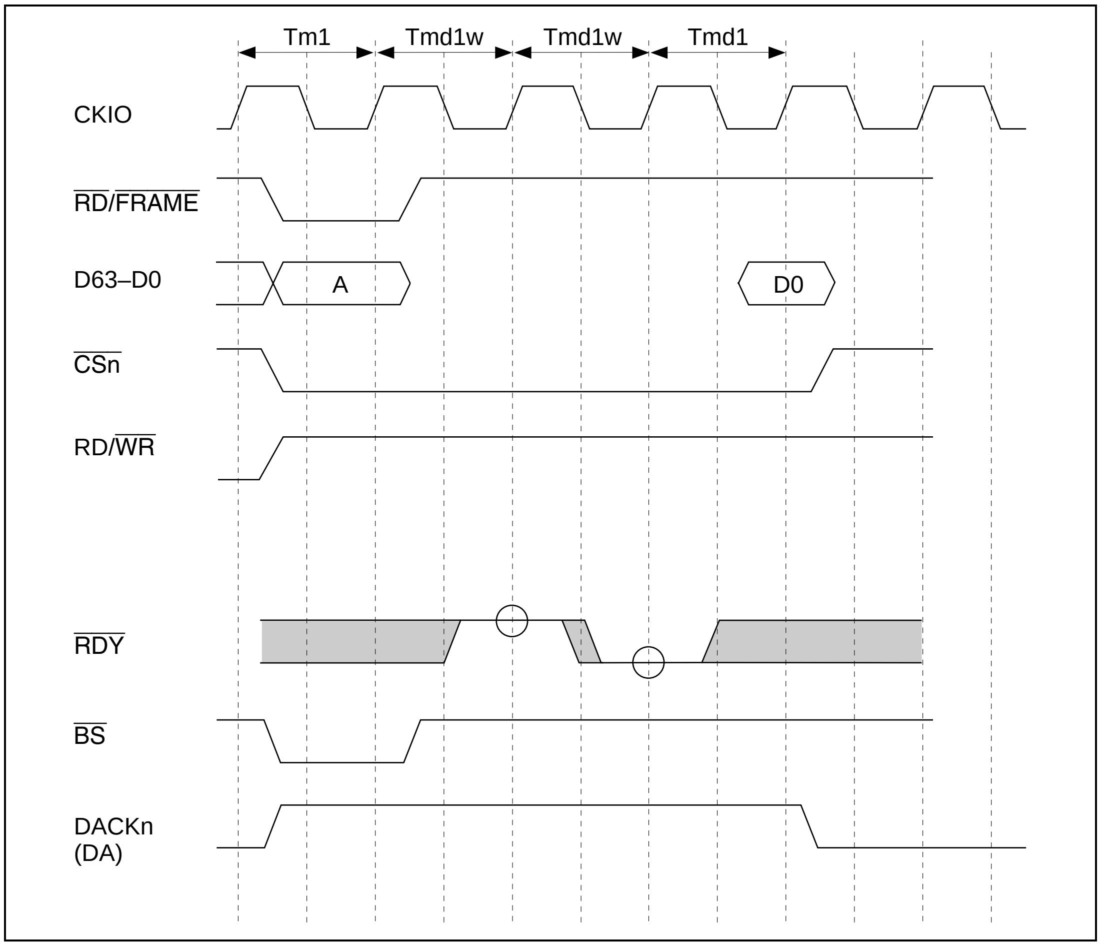


Figure 13.53 MPX Interface Timing 2 (Single Read, One Internal Wait Inserted)

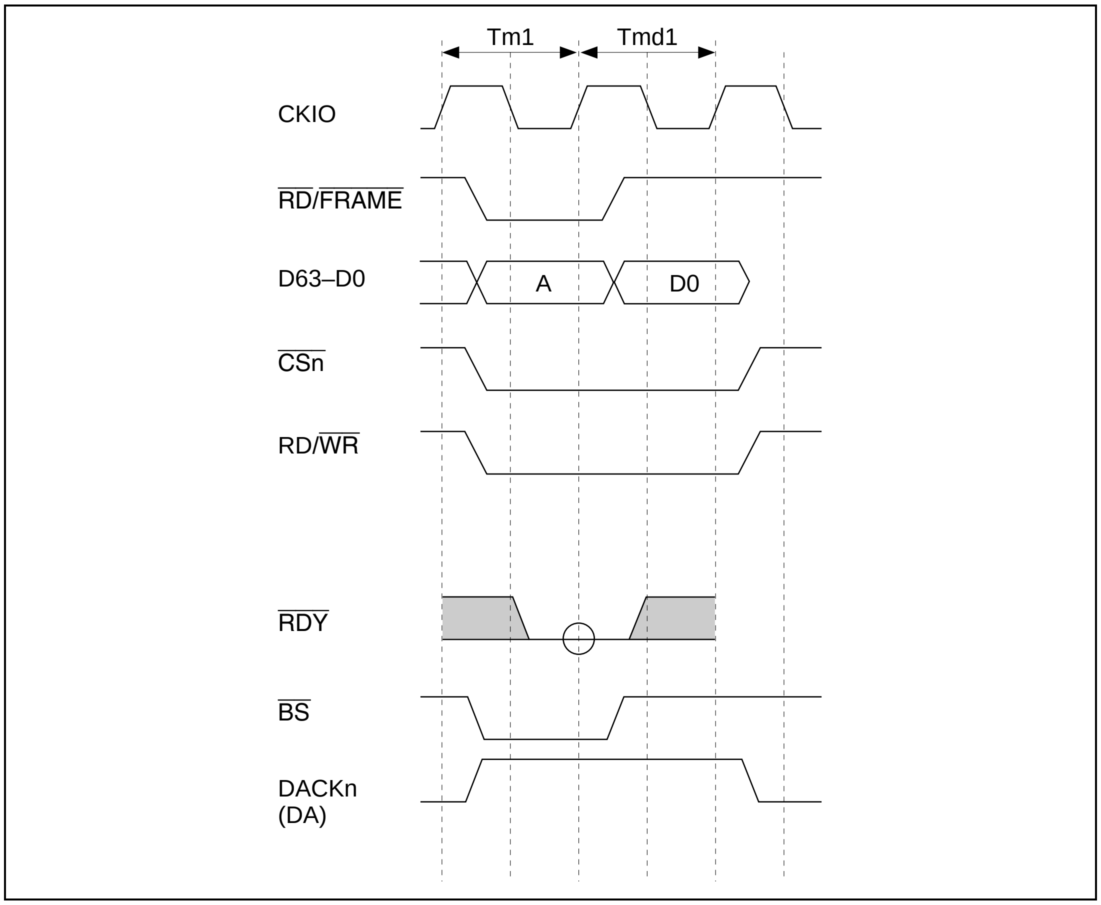


Figure 13.54 MPX Interface Timing 3 (Single Write Cycle, No Wait)

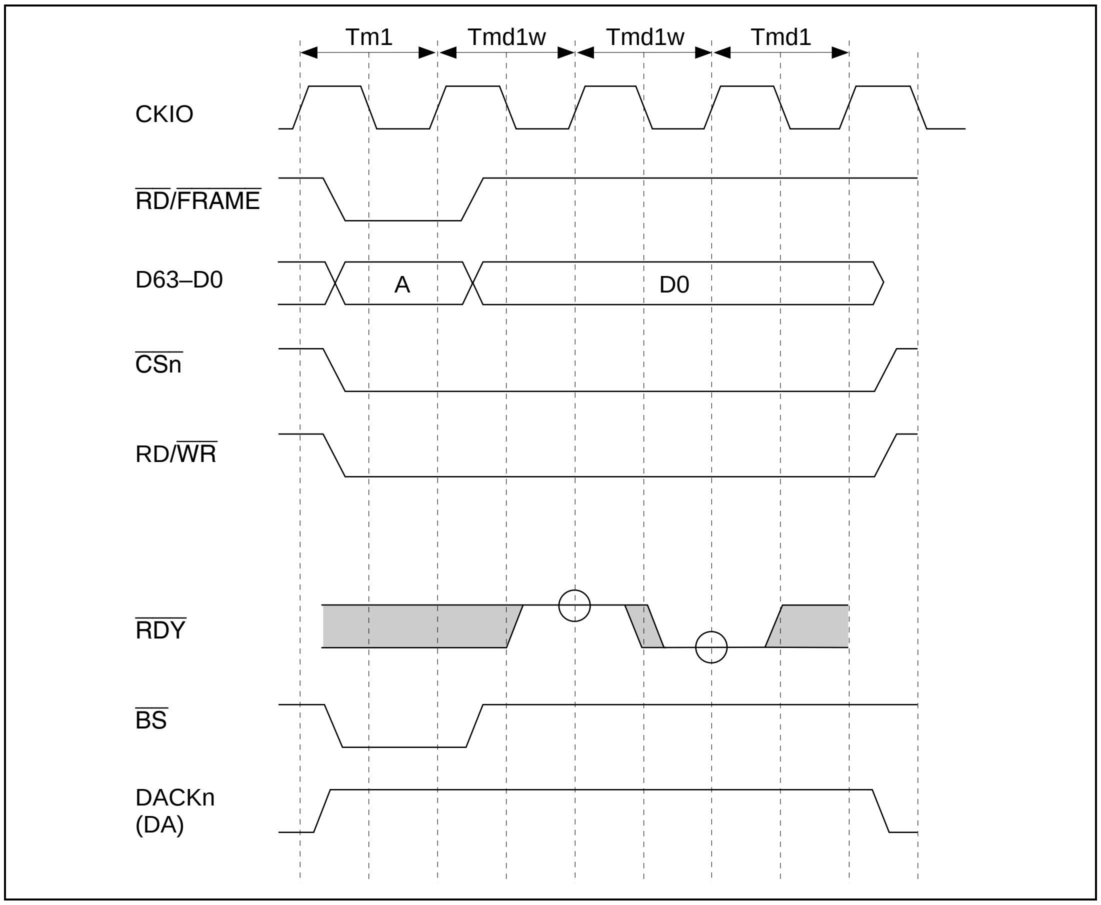


Figure 13.55 MPX Interface Timing 4 (Single Write, One Internal Wait Inserted)

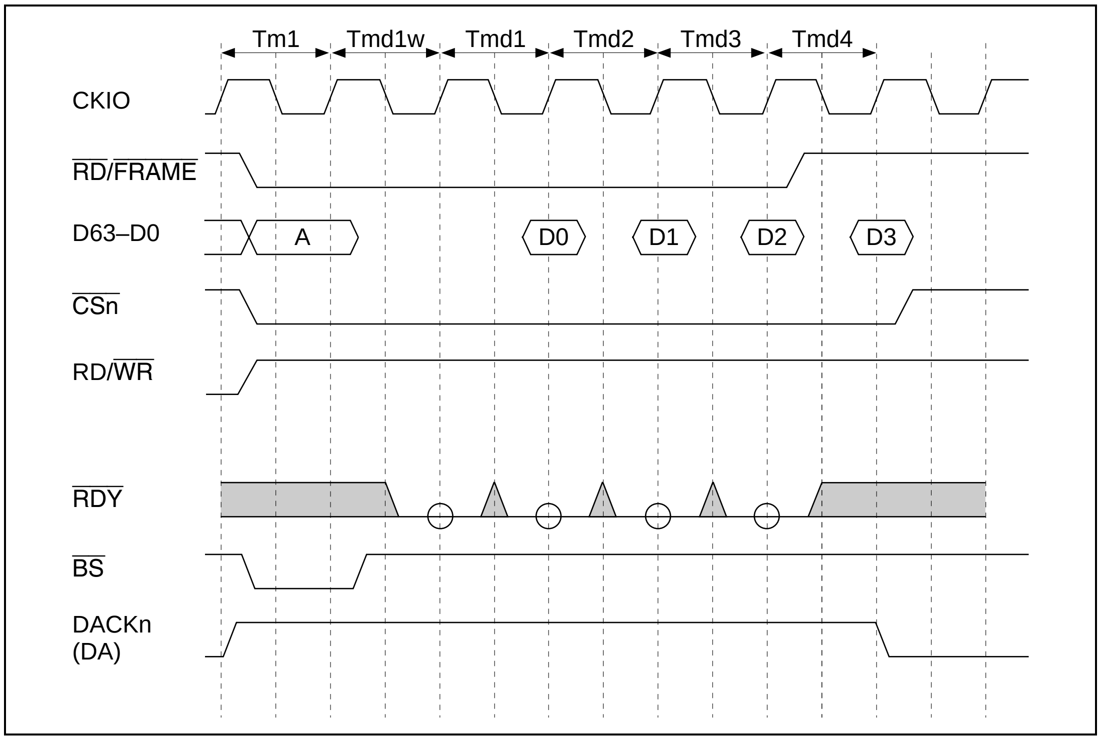


Figure 13.56 MPX Interface Timing 5 (Burst Read Cycle, No Wait)

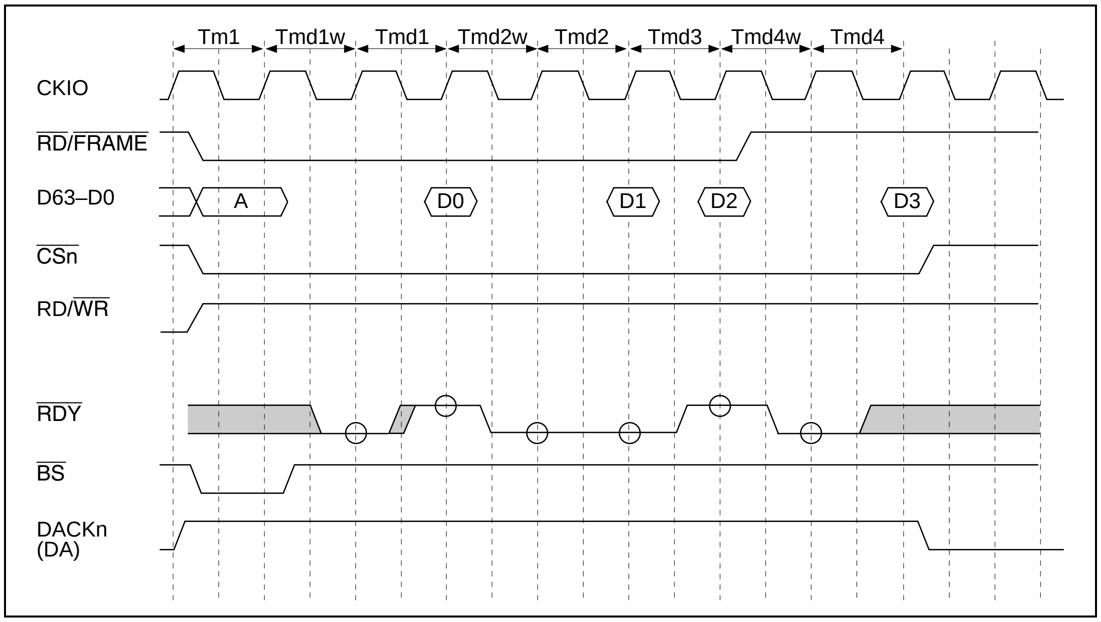


Figure 13.57 MPX Interface Timing 6 (Burst Read Cycle, One Internal Wait Inserted)

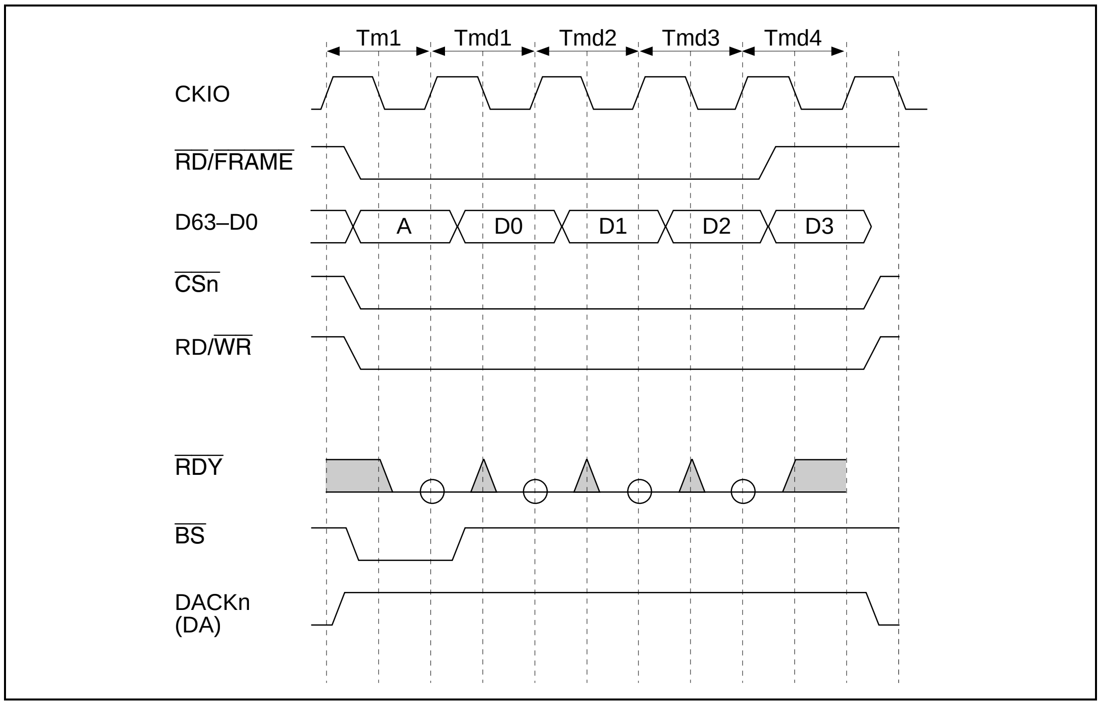


Figure 13.58 MPX Interface Timing 7 (Burst Write Cycle, No Wait)

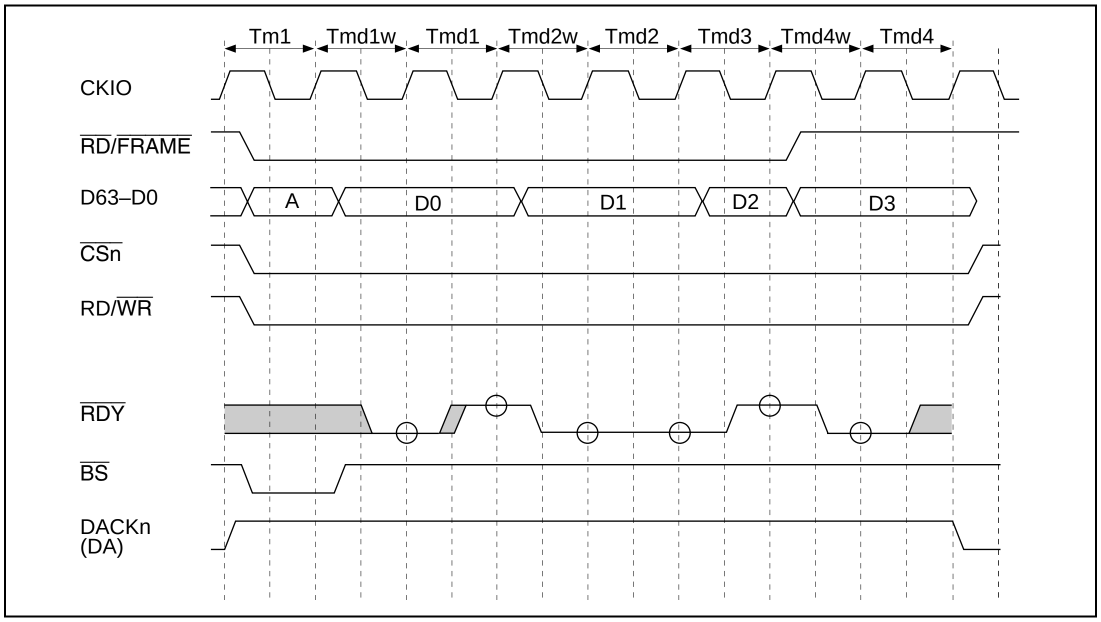


Figure 13.59 MPX Interface Timing 8 (Burst Write Cycle, One Internal Wait Inserted for First Data Only)

13.3.9 Byte Control SRAM

The byte control SRAM interface is a memory interface that outputs a byte select strobe ($\overline{\text{WEn}}$) in both read and write bus cycles. It has 16 bit data pins, and can be directly connected to SRAM which has an upper byte select strobe and lower byte select strobe function such as UB and LB.

Areas 1 and 4 can be designated as byte control SRAM. However, when these areas are set to MPX mode, MPX mode has priority.

The byte control SRAM write timing is the same as for the normal SRAM interface.

In read operations, the $\overline{\text{WEn}}$ pin timing is different. In a read access, only the $\overline{\text{WE}}$ signal for the byte being read is asserted. Assertion is synchronized with the fall of the CKIO clock, as for the $\overline{\text{WE}}$ signal, while negation is synchronized with the rise of the CKIO clock, using the same timing as the $\overline{\text{RD}}$ signal.

Cache fill or copy-back reads and writes are performed consecutively for a total of 32 bytes according to the set bus width. The first access is performed on the data for which there was an access request, and the remaining accesses are performed on the data at the 32-byte boundary. The bus is not released during this period.

Figure 13.60 shows an example of byte control SRAM connection to the SH7750, and figures 13.61 to 13.63 show examples of byte control SRAM bus timing.

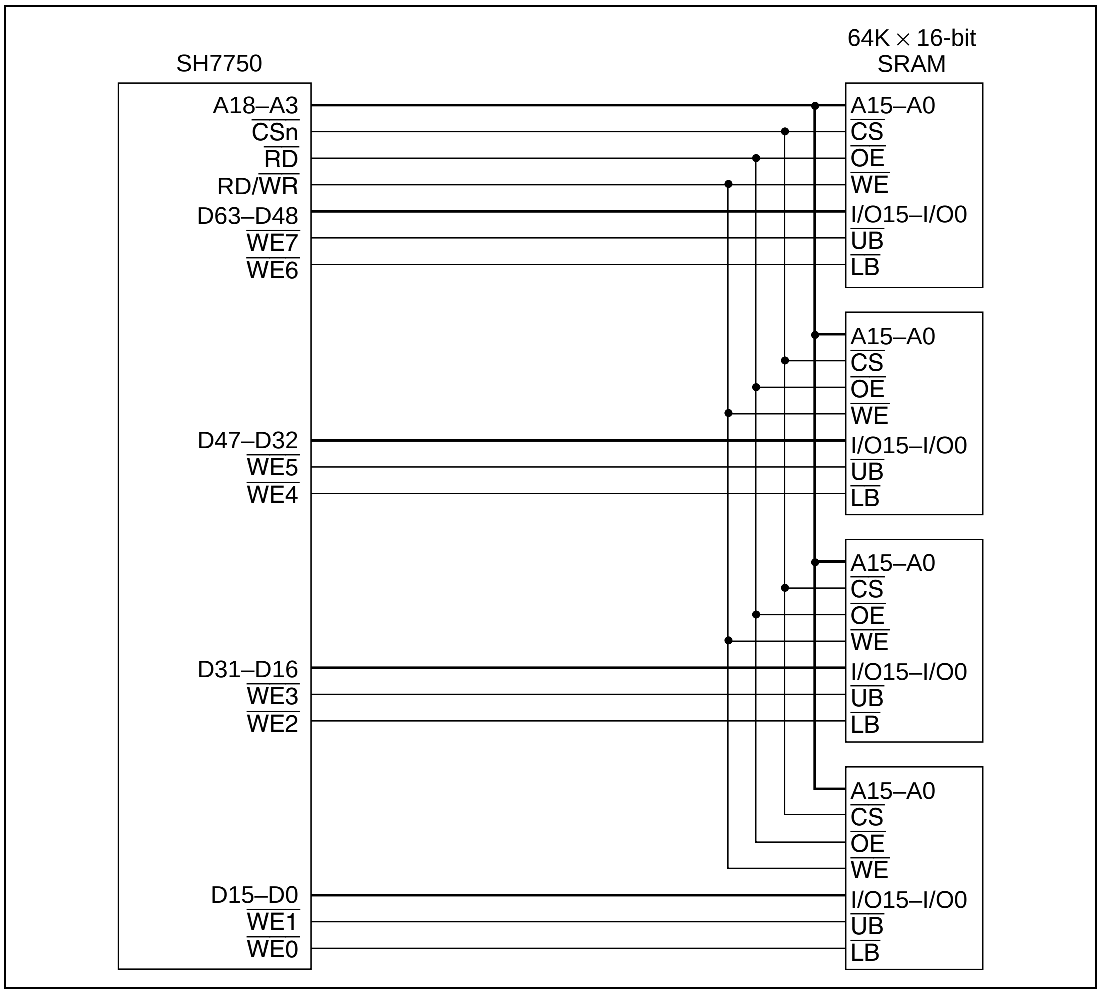


Figure 13.60 Example of 64-Bit Data Width Byte Control SRAM

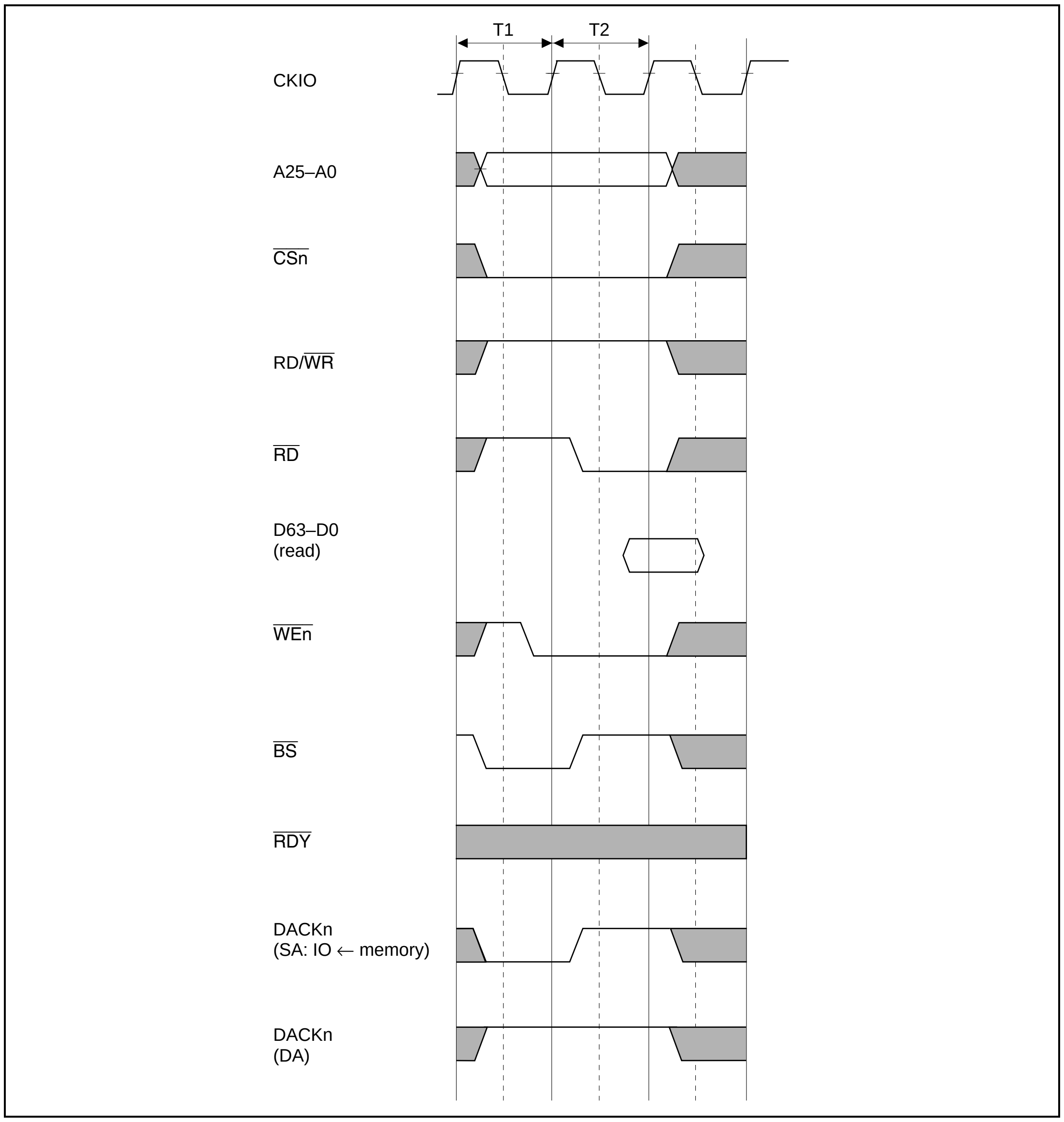


Figure 13.61 Byte Control SRAM Basic Read Cycle (No Wait)

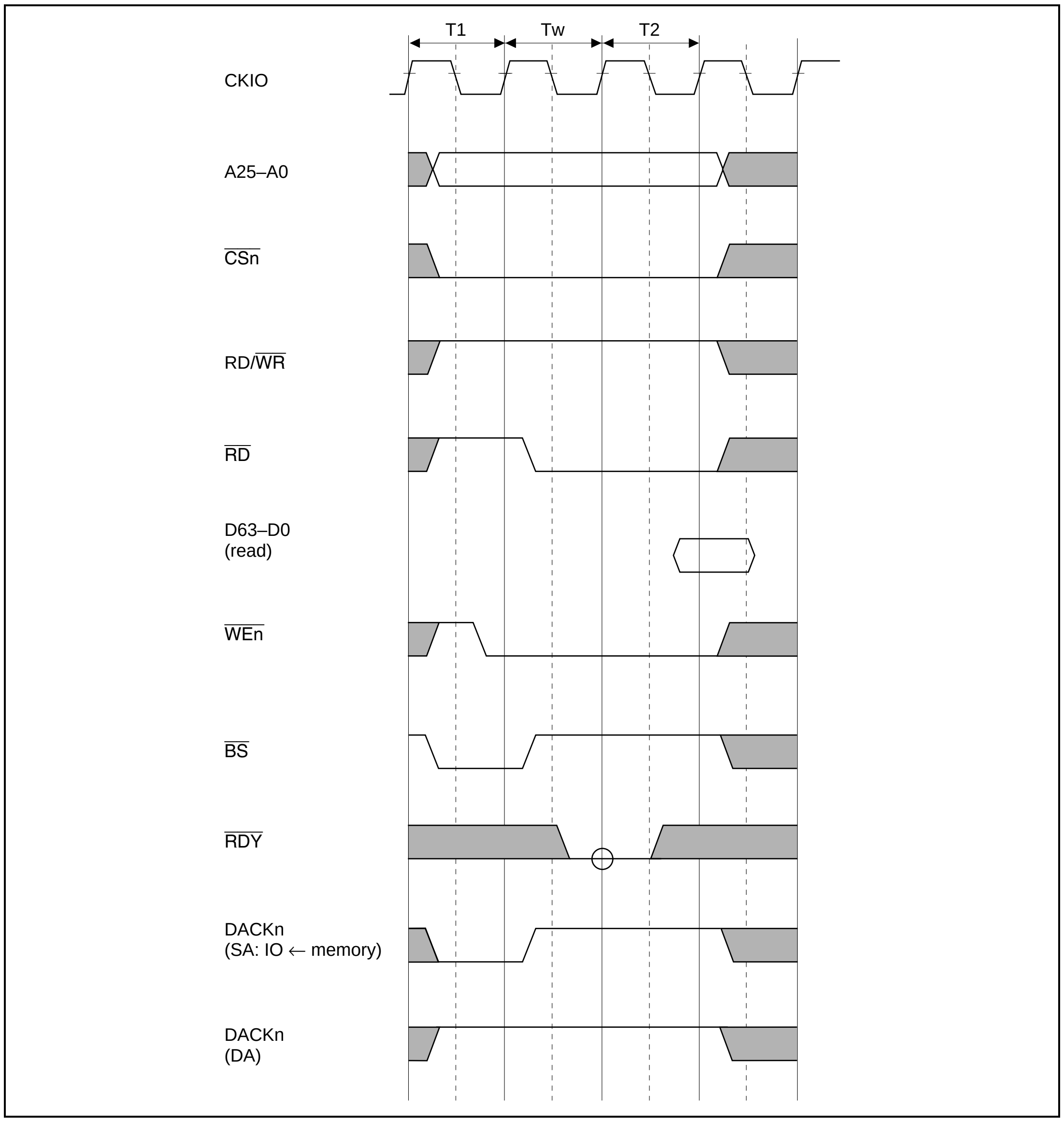


Figure 13.62 Byte Control SRAM Basic Read Cycle (One Internal Wait Cycle)

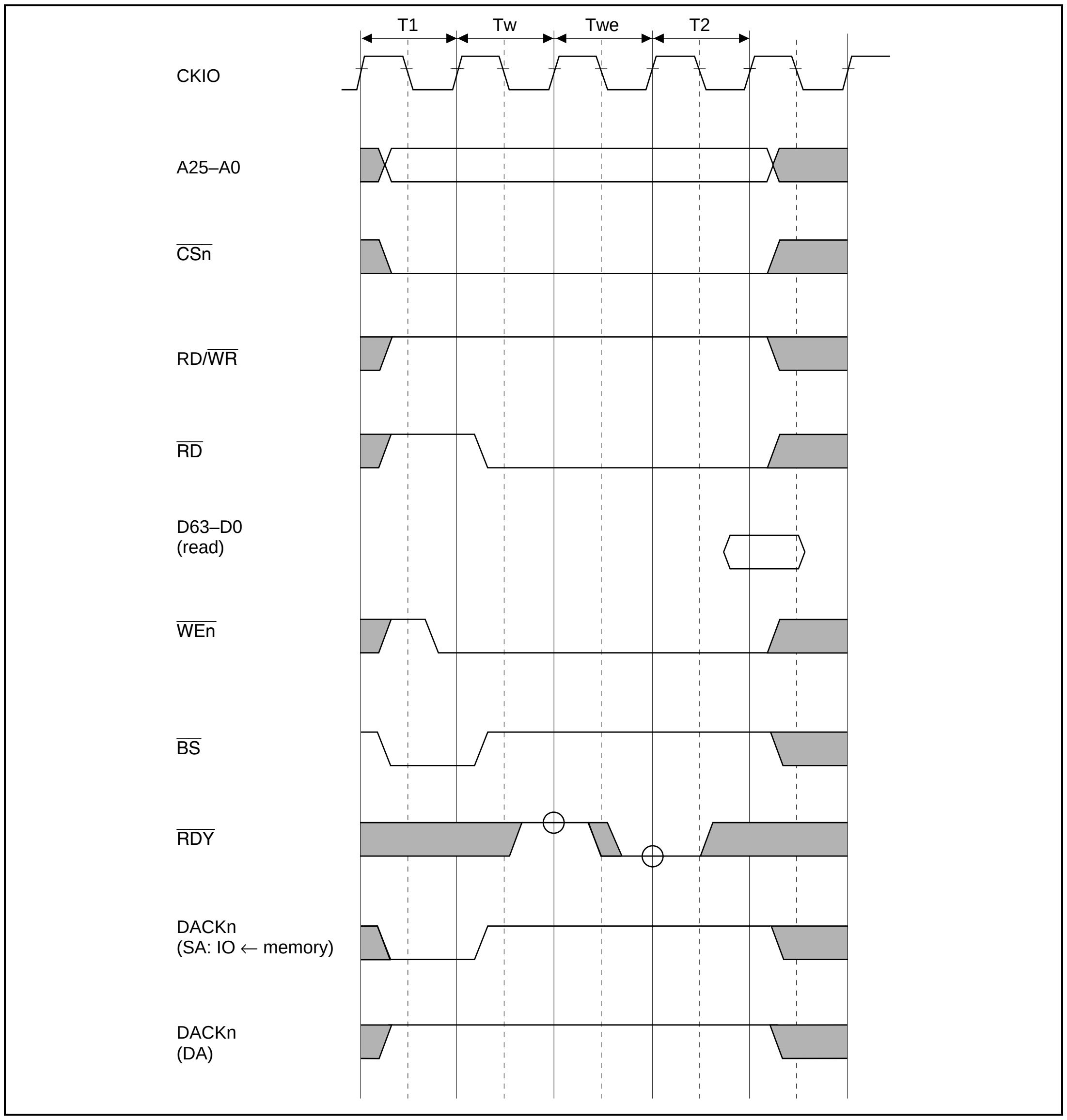


Figure 13.63 Byte Control SRAM Basic Read Cycle (One Internal Wait + One External Wait)

13.3.10 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 the data in the next access, and so resulting 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, and if there is a possibility of a bus collision when the next access is started, inserts a wait cycle before the access cycle to prevent a data collision. Wait cycle insertion consists of inserting idle cycles between access cycles, as shown in section 13.2.3, Wait Control Register (WCR1). When the SH7750 performs consecutive write cycles, the data transfer direction is fixed (from the SH7750 to other memory) and there is no problem. With read accesses to the same area, also, in principle data is output from the same data buffer, and wait cycle insertion is not performed. If there is originally space between accesses, according to the setting of bits AnIW2–AnIW0 (n = 0 to 6) in WCR1, the number of idle cycles inserted is the specified number of idle cycles minus the number of empty cycles.

When bus arbitration is performed, the bus is released after waits are inserted between cycles.

In single address mode DMA transfer, when data transfer is performed from an I/O device to memory the data on the bus is determined by the speed of the I/O device. With a low-speed I/O device, an inter-cycle idle wait equivalent to the output buffer turn-off time must be inserted. Even with high-speed memory, when DMA transfer is considered, it may be necessary to insert an inter-cycle wait to adjust to the speed of a low-speed device, preventing the memory from being used at full speed.

Bits DMAIW2–DMAIW0 in wait control register 1 (WCR1) allow an inter-cycle wait setting to be made when transferring data from an I/O device to memory using single address mode DMA transfer. From 0 to 15 waits can be inserted. The number of waits specified by DMAIW2–DMAIW0 are inserted in single address DMA transfers to all areas.

In dual address mode DMA transfer, the normal inter-cycle wait specified by AnIW2–AnIW0 (n = 0 to 6) is inserted.

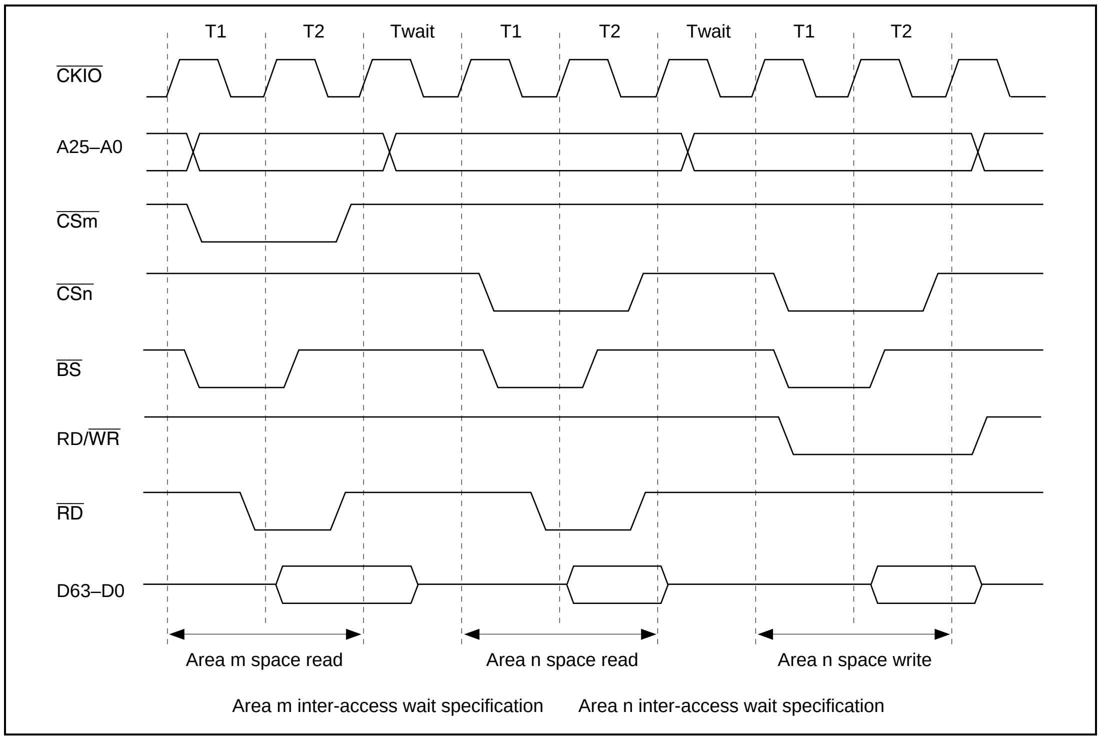


Figure 13.64 Waits between Access Cycles

13.3.11 Bus Arbitration

The SH7750 is provided with a bus arbitration function that grants the bus to an external device when it makes a bus request. Also provided is a bus arbitration function to support the connection of two processors. The purpose of this function is to enable a multiprocessor system to be implemented with a minimum of hardware by connecting the processors in a bus arbitration master and slave arrangement.

There are three bus arbitration modes: master mode, partial-sharing master mode, and slave mode. In master mode the bus is held on a constant basis, and is released to another device in response to a bus request. In slave mode the bus is not held on a constant basis; a bus request is issued each time an external bus cycle occurs, and the bus is released again at the end of the access. In partial-sharing master mode, only area 2 is shared with external devices; slave mode is in effect for area 2, while for other spaces, bus arbitration is not performed and the bus is held constantly. The area in the master mode chip to which area 2 in the partial-sharing master mode chip is allocated is determined by an external circuit.

Master mode and slave mode can be specified by the external mode pins. Partial-sharing master mode is entered from master mode by means of a software setting. See Appendix C, Mode Pin Settings, for the external mode pin settings. In master mode and slave mode, the bus goes to the high-impedance state when not being held, so that it is possible to directly connect the master mode and slave mode chips. In partial-sharing master mode, the bus is constantly driven, and therefore an external buffer is necessary for connection to the master bus. In master mode, it is possible to connect an external device that issues bus requests instead of a slave mode chip. In the following description, an external device that issues bus requests is also referred to as a slave.

The SH7750 has two internal bus masters: the CPU and the DMAC. When synchronous DRAM or DRAM is connected and refresh control is performed, refresh requests constitute a third bus master. In addition to these are bus requests from external devices in master mode. If requests occur simultaneously, priority is given, in high-to-low order, to a bus request from an external device, a refresh request, the DMAC, and the CPU.

To prevent incorrect operation of connected devices when the bus is transferred between master and slave, all bus control signals are negated before the bus is released. When mastership of the bus is received, also, bus control signals begin driving the bus from the negated state. Since signals are driven to the same value by the master and slave exchanging the bus, output buffer collisions can be avoided. By turning off the output buffer on the side releasing the bus, and turning on the output buffer on the side receiving the bus, simultaneously with respect to the bus control signals, it is possible to eliminate the signal high-impedance period. It is not necessary to provide the pull-up resistors usually inserted in these control signal lines to prevent incorrect operation due to external noise in the high-impedance state.

Bus transfer is executed between bus cycles.

When the bus release request signal ($\overline{\text{BREQ}}$) is asserted, the SH7750 releases the bus as soon as the currently executing bus cycle ends, and outputs the bus use permission signal ($\overline{\text{BACK}}$). However, bus release is not performed during a burst transfer for cache fill or write-back, or between a read cycle and write cycle during execution of a TAS instruction. Also, bus arbitration is not performed between bus cycles generated due to the fact that the data bus width is smaller than the access size, such as when a longword access is made to 8-bit memory. When $\overline{\text{BREQ}}$ is negated, $\overline{\text{BACK}}$ is negated and use of the bus is resumed. See Appendix E, Pin Functions, for the pin states when the bus is released.

As the CPU in the SH7750 is connected to cache memory by a dedicated internal bus, reading from cache memory can still be carried out when the bus is being used by another bus master inside or outside the SH7750. When writing from the CPU, an external write cycle is generated when write-through has been set for the cache in the SH7750, or when an access is made to a cache-off area. There is consequently a delay until the bus is returned.

When the SH7750 wants to take back the bus in response to an internal memory refresh request, it negates $\overline{\text{BACK}}$. On receiving the $\overline{\text{BACK}}$ negation, the device that asserted the external bus release request negates $\overline{\text{BREQ}}$ to release the bus. The bus is thereby returned to the SH7750, which then carries out the necessary processing.

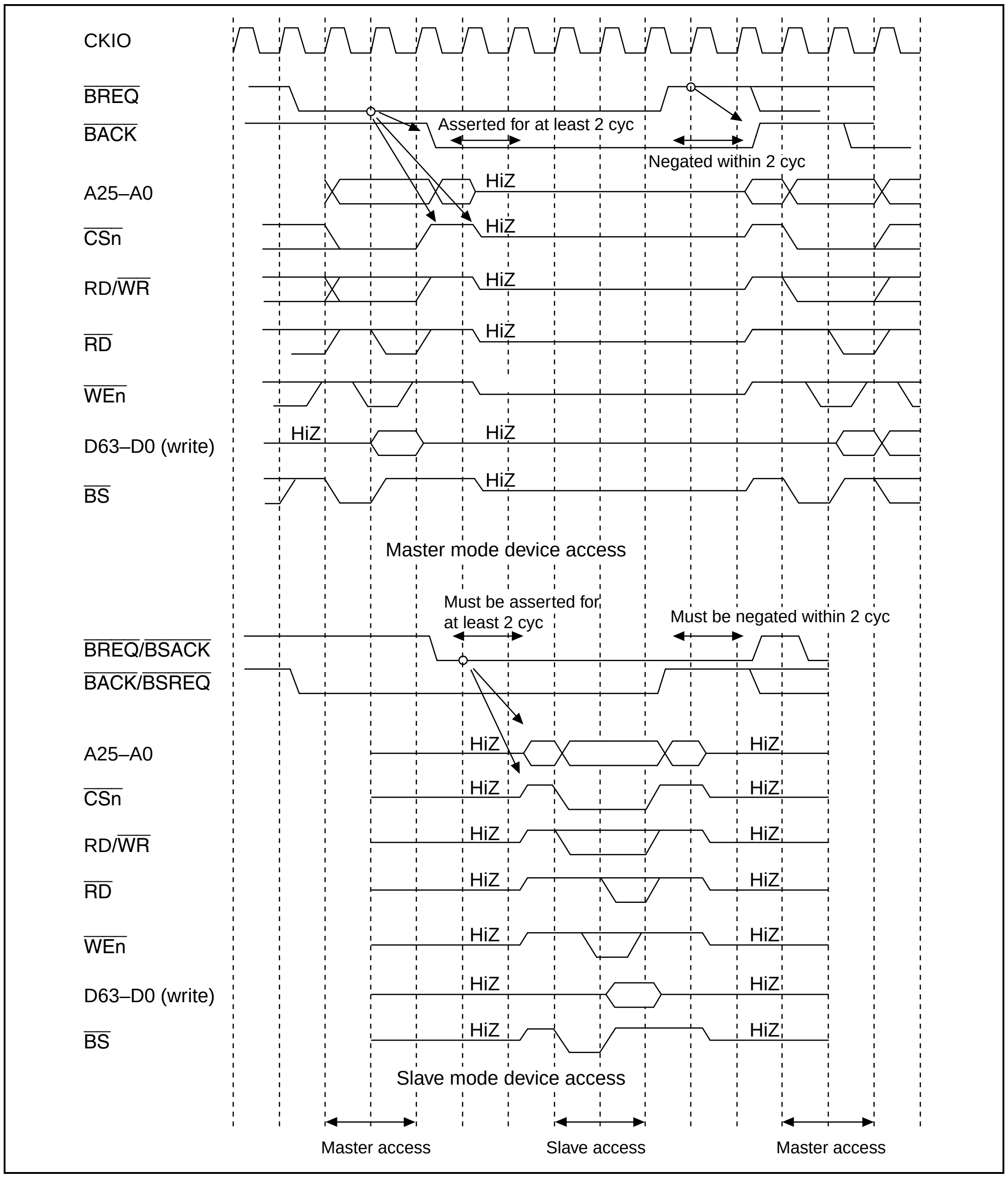


Figure 13.65 Arbitration Sequence

13.3.12 Master Mode

The master mode processor holds the bus itself unless it receives a bus request.

On receiving an assertion (low level) of the bus request signal ($\overline{\mathrm{BREQ}}$) from off-chip, the master mode processor releases the bus and asserts (drives low) the bus use permission signal ($\overline{\mathrm{BACK}}$) as soon as the currently executing bus cycle ends. If a bus release request due to a refresh request has not been issued, on receiving the $\overline{\mathrm{BREQ}}$ negation (high level) indicating that the slave has released the bus, the processor negates (drives high) the $\overline{\mathrm{BACK}}$ signal and resumes use of the bus.

If a bus request is issued due to a memory refresh request in the bus-released state, the processor negates the bus use permission signal ($\overline{\mathrm{BACK}}$), and on receiving the $\overline{\mathrm{BREQ}}$ negation indicating that the slave has released the bus, resumes use of the bus.

When the bus is released, all bus interface related output signals and input/output signals go to the high-impedance state, except for the synchronous DRAM interface CKE signal and bus arbitration $\overline{\mathrm{BACK}}$ signal, and DACK0 and DACK1 which control DMA transfers.

With DRAM, the bus is released after precharging is completed. With synchronous DRAM, also, a precharge command is issued for the active bank and the bus is released after precharging is completed.

The actual bus release sequence is as follows.

First, the bus use permission signal is asserted in synchronization with the rising edge of the clock. The address bus and data bus go to the high-impedance state in synchronization with the next rising edge of the clock after this $\overline{\mathrm{BACK}}$ assertion. At the same time, the bus control signals ($\overline{\mathrm{BS}}$, $\overline{\mathrm{CSn}}$, $\overline{\mathrm{RAS1}}$, $\overline{\mathrm{RAS2}}$, $\overline{\mathrm{WEn}}$, $\overline{\mathrm{RD}}$, RD/$\overline{\mathrm{WR}}$, $\overline{\mathrm{RD2}}$, RD/$\overline{\mathrm{WR2}}$, $\overline{\mathrm{CE2A}}$, and $\overline{\mathrm{CE2B}}$) go to the high-impedance state. These bus control signals are negated no later than one cycle before going to high-impedance. Bus request signal sampling is performed on the rising edge of the clock.

The sequence for re-acquiring the bus from the slave is as follows.

As soon as $\overline{\mathrm{BREQ}}$ negation is detected on the rising edge of the clock, $\overline{\mathrm{BACK}}$ is negated and bus control signal driving is started. Driving of the address bus and data bus starts at the next rising edge of an in-phase clock. The bus control signals are asserted and the bus cycle is actually started, at the earliest, at the clock rising edge at which the address and data signals are driven.

In order to reacquire the bus and start execution of a refresh operation or bus access, the $\overline{\mathrm{BREQ}}$ signal must be negated for at least two cycles.

If a refresh request is generated when $\overline{\mathrm{BACK}}$ has been asserted and the bus has been released, the $\overline{\mathrm{BACK}}$ signal is negated even while the $\overline{\mathrm{BREQ}}$ signal is asserted to request the slave to relinquish the bus. When the SH7750 is used in master mode, consecutive bus accesses may be attempted to reduce the overhead due to arbitration in the case of a slave designed independently by the user. When connecting a slave for which the total duration of consecutive accesses exceeds the refresh cycle, the design should provide for the bus to be released as soon as possible after negation of the $\overline{\text{BACK}}$ signal is detected.

13.3.13 Slave Mode

In slave mode, the bus is normally in the released state, and an external device cannot be accessed unless the bus is acquired through execution of the bus arbitration sequence. In a reset, also, the bus-released state is established and the bus arbitration sequence is started from the reset vector fetch.

To acquire the bus, the slave device asserts (drives low) the $\overline{\text{BSREQ}}$ signal in synchronization with the rising edge of the clock. The bus use permission $\overline{\text{BSACK}}$ signal is sampled for assertion (low level) in synchronization with the rising edge of the clock. When $\overline{\text{BSACK}}$ assertion is detected, the bus control signals and address bus are immediately driven at the negated level. The bus cycle is started at the next rising edge of the clock. The last signal negated at the end of the access cycle is synchronized with the rising edge of the clock. When the bus cycle ends, the $\overline{\text{BSREQ}}$ signal is negated and the release of the bus is reported to the master. On the next rising edge of the clock, the control signals are set to high-impedance.

In order for the slave mode processor to begin access, the $\overline{\text{BSACK}}$ signal must be asserted for at least two cycles.

For a slave access cycle in DRAM or synchronous DRAM, the bus is released on completion of precharging, as in the case of the master.

Refresh control is left to the master mode device, and any refresh control settings made in slave mode are ignored.

Do not use DRAM/synchronous DRAM RAS down mode in slave mode.

Synchronous DRAM mode register settings should be made by the master mode device. Do not use the DMAC’s DDT mode in slave mode.

13.3.14 Partial-Sharing Master Mode

In partial-sharing master mode, area 2 only is shared with other devices, and other areas can be accessed at all times. Partial-sharing master mode can be set by setting master mode with the external mode pins, and setting the PSHR bit to 1 in BCR1 in the initialization procedure in a power-on reset. In a manual reset the bus state controller setting register values are retained, and so need not be set again.

Partial-sharing master mode is designed for use in conjunction with a master mode chip. The partial-sharing master can access a device on the master side via area 2, but the master cannot access a device on the partial-sharing master side.

An address and control signal buffer and a data buffer must be located between the partial-sharing master and the master, and controlled by a buffer control circuit.

The partial-sharing master mode processor uses the following procedure to access area 2. It asserts the $\overline{\text{BSREQ}}$ signal on the rising edge of the clock, and issues a bus request to the master. It samples $\overline{\text{BSACK}}$ on each rising edge of the clock, and on receiving $\overline{\text{BSACK}}$ assertion, starts the access cycle on the next rising edge of the clock. At the end of the access, it negates $\overline{\text{BSREQ}}$ on the rising edge of the clock. Buffer control in an access to an area 2 device by the partial-sharing master is carried out by referencing the $\overline{\text{CS2}}$ signal or $\overline{\text{BSREQ}}$ and $\overline{\text{BSACK}}$ signals on the partial-sharing master side. Permission to use the bus is reported by the $\overline{\text{BSACK}}$ line connected to the partial-sharing master, but the master may also negate the $\overline{\text{BSACK}}$ signal even while the bus is being used, if it needs the bus urgently in order to service a refresh, for example. Consequently, the partial-sharing master has to monitor the $\overline{\text{BSREQ}}$ signal to see whether it can continue to use the bus after detecting $\overline{\text{BSACK}}$ assertion. In the case of the address buffer, after the address buffer is turned on when $\overline{\text{BSACK}}$ assertion is detected, the buffer is kept on until $\overline{\text{BSREQ}}$ is negated, at which point it is turned off. If the turning-off of the buffer used is late, resulting in a collision with the start of an access cycle on the master side, the $\overline{\text{BSREQ}}$ signal output from the partial-sharing master must be routed through a delay circuit as part of the buffer control circuit, and input to the master $\overline{\text{BREQ}}$ signal.

In order for a partial-sharing master mode processor to begin area 2 access, the $\overline{\text{BSACK}}$ signal must be asserted for at least two cycles.

When the bus is released after area 2 has been accessed in partial-sharing master mode, if area 2 is synchronous DRAM, there is a wait of the period required for auto-precharge before bus release is performed.

In partial-sharing master mode, refreshing is not performed for area 2 (refresh requests are ignored).

Do not use DRAM/synchronous DRAM RAS down mode in partial-sharing master mode.

Area 2 synchronous DRAM mode register settings should be made by the master mode device. Set partial-sharing master mode (by setting the PSHR bit to 1 in BCR1) after completion of the area 3 synchronous DRAM mode register settings.

In partial-sharing master mode, DMA transfer should not be performed on area 2, and the DMAC's DDT mode should not be used.

13.3.15 Cooperation between Master and Slave

To enable system resources to be controlled in a harmonious fashion by master and slave, their respective roles must be clearly defined. Before DRAM or synchronous DRAM is used, initialization operations must be carried out. Responsibility must also be assigned when a standby operation is performed to implement the power-down state.

The design of the SH7750 provides for all control, including initialization, refreshing, and standby control, to be carried out by the master mode device. In a dual-processor configuration using direct master/slave connection, all processing except direct access to memory is handled by the master. In a combination of master mode and partial-sharing master mode, the partial-sharing master mode processor performs initialization, refreshing, and standby control for the areas connected to it, with the exception of area 2, while the master performs initialization of the memory connected to it.

If the SH7750 is specified as the master in a power-on reset, it will not accept bus requests from the slave until the $\overline{\text{BREQ}}$ enable bit (BCR1.BREQEN) is set to 1.

To ensure that the slave processor does not access memory requiring initialization before use, such as DRAM and synchronous DRAM, until initialization is completed, write 1 to the $\overline{\text{BREQ}}$ enable bit after initialization ends.

Before setting self-refresh mode in standby mode, etc., write 0 to the $\overline{\text{BREQ}}$ enable bit to invalidate the $\overline{\text{BREQ}}$ signal from the slave. Write 1 to the $\overline{\text{BREQ}}$ enable bit only after the master has performed the necessary processing (refresh settings, etc.) for exiting self-refresh mode.

Section 14 Direct Memory Access Controller (DMAC)

14.1 Overview

The SH7750 includes an on-chip four-channel direct memory access controller (DMAC). The DMAC can be used in place of the CPU to perform high-speed data transfers among external devices equipped with DACK (DMA transfer end notification), external memories, memory-mapped external devices, and on-chip peripheral modules (except the DMAC, BSC, and UBC). Using the DMAC reduces the burden on the CPU and increases the operating efficiency of the chip.

14.1.1 Features

The DMAC has the following features.

- Four channels
- Physical address space
- Choice of 8-bit, 16-bit, 32-bit, 64-bit, or 32-byte transfer data length
- Maximum of 16 M (16,777,216) transfers
- Choice of single or dual address mode
  - Single address mode: Either the transfer source or the transfer destination (peripheral device) is accessed by a DACK signal while the other is accessed by address. One data transfer is completed in one bus cycle.
  - Dual address mode: Both the transfer source and transfer destination are accessed by address. Values set in DMAC internal registers indicate the accessed address for both the transfer source and the transfer destination. Two bus cycles are required for one data transfer.
- Channel functions: Transfer modes that can be set are different for each channel.
  - Channel 0: Single or dual address mode. External requests are accepted.
  - Channel 1: Single or dual address mode. External requests are accepted.
  - Channel 2: Dual address mode only.
  - Channel 3: Dual address mode only.
- Transfer requests: The following three DMAC transfer activation requests are supported.
  - External request: From two $\overline{\text{DREQ}}$ pins. Either low level detection or falling edge detection can be specified. External requests can be accepted on channels 0 and 1 only.
  - Requests from on-chip peripheral modules: Transfer requests from modules such as the SCI and TMU. These can be accepted on all channels.
  - Auto-request: The transfer request is generated automatically within the DMAC.

- Choice of bus mode: Cycle steal mode or burst mode
- Two types of DMAC channel priority ranking:
  - Fixed priority mode: Channel priorities are permanently fixed.
  - Round robin mode: Sets the lowest priority for the channel for which an execution request was last accepted.
- An interrupt request can be sent to the CPU on completion of the specified number of transfers.
- On-demand data transfer mode (DDT mode)

  In this mode, interfacing between an external device and the DMAC is performed using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, and ID [1:0] pins. External requests can be accepted on all four channels.

  For channel 0, data transfer can be carried out with the transfer mode, number of transfers, transfer address (single only), etc., specified by the external device.

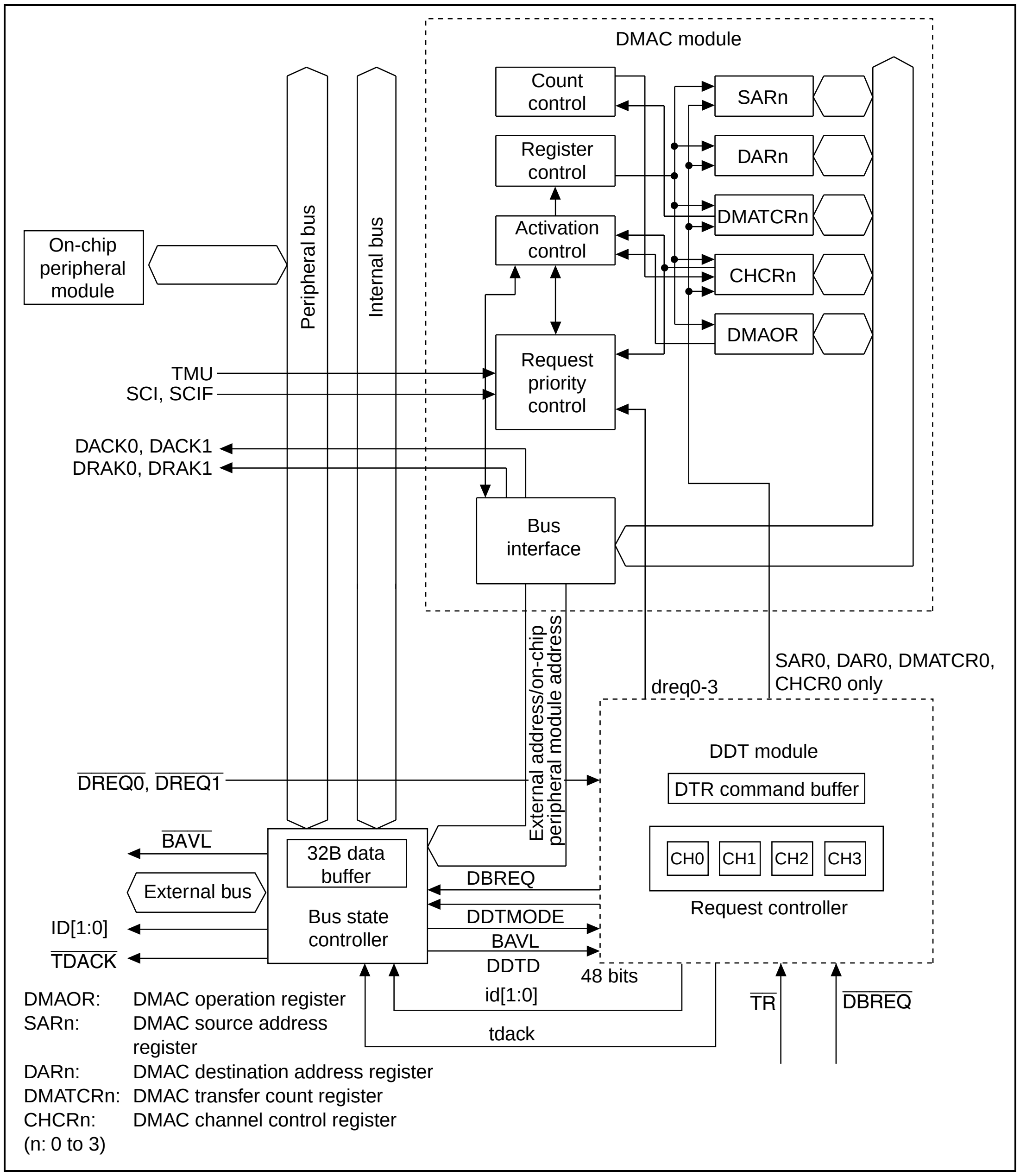
  For channels 1 to 3, when transfer is performed by means of an on-chip peripheral module request or auto-request, the operation is the same as in the normal mode. On these channels, data transfer can be initiated by an external request.

  - Channel 0: Single address mode. External requests are accepted
  - Channel 1: Single or dual address mode. External requests are accepted.
  - Channel 2: Single or dual address mode. External requests are accepted.
  - Channel 3: Single or dual address mode. External requests are accepted.

  In DDT mode, data transfer is carried out using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, and ID [1:0] signals to perform handshaking between the external device and the DMAC.

14.1.2 Block Diagram

Figure 14.1 shows 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
(n: 0 to 3)	

Figure 14.1 Block Diagram of DMAC

14.1.3 Pin Configuration

Tables 14.1 and 14.2 show the DMAC pins.

Table 14.1 DMAC Pins

Channel	Pin Name	Abbreviation	I/O	Function
0	DMA transfer request	$\overline{\text{DREQ0}}$	Input	DMA transfer request input from external device to channel 0
	$\overline{\text{DREQ}}$ acceptance confirmation	DRAK0	Output	Acceptance of request for DMA transfer from channel 0 to external device Notification to external device of start of execution
	DMA transfer end notification	DACK0	Output	Strobe output to external device of DMA transfer request from channel 0 to external device
1	DMA transfer request	$\overline{\text{DREQ1}}$	Input	DMA transfer request input from external device to channel 1
	$\overline{\text{DREQ}}$ acceptance confirmation	DRAK1	Output	Acceptance of request for DMA transfer from channel 1 to external device Notification to external device of start of execution
	DMA transfer end notification	DACK1	Output	Strobe output to external device of DMA transfer request from channel 1 to external device

Table 14.2 DMAC Pins in DDT Mode

Pin Name	Abbreviation	I/O	Function
Data bus request	$\overline{\text{DBREQ}}$ ($\overline{\text{DREQ0}}$)	Input	Data bus release request from external device for DTR format input
Data bus available	$\overline{\text{BAVL}}$ (DRAK0)	Output	Data bus release notification Data bus can be used 2 cycles after $\overline{\text{BAVL}}$ is asserted
Transfer request signal	$\overline{\text{TR}}$ ($\overline{\text{DREQ1}}$)	Input	If asserted 2 cycles after $\overline{\text{BAVL}}$ assertion, DTR format is sent Only $\overline{\text{TR}}$ asserted: DMA request $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$ asserted simultaneously: Direct request to channel 2
DMAC strobe	$\overline{\text{TDACK}}$ (DACK0)	Output	Reply strobe signal for external device from DMAC
Channel number notification	ID [1:0] (DRAK1, DACK1)	Output	Notification of channel number to external device at same time as $\overline{\text{TDACK}}$ output (ID [1] = DRAK1, ID [0] = DACK1)

14.1.4 Register Configuration

Table 14.3 summarizes the DMAC registers. The DMAC has a total of 17 registers: four registers are allocated to each channel, and an additional control register is shared by all four channels.

Table 14.3 DMAC Registers

Channel	Name	Abbreviation	Read/Write	Initial Value	P4 Address	Area 7 Address	Access Size
0	DMA source address register 0	SAR0	R/W*[2]	Undefined	H'FFA00000	H'1FA00000	32
	DMA destination address register 0	DAR0	R/W*[2]	Undefined	H'FFA00004	H'1FA00004	32
	DMA transfer count register 0	DMATCR0	R/W*[2]	Undefined	H'FFA00008	H'1FA00008	32
	DMA channel control register 0	CHCR0	R/W*[1],*[2]	H'00000000	H'FFA0000C	H'1FA0000C	32

Table 14.3 DMAC Registers

Channel	Name	Abbreviation	Read/Write	Initial Value	P4 Address	Area 7 Address	Access Size
1	DMA source address register 1	SAR1	R/W	Undefined	H'FFA00010	H'1FA00010	32
	DMA destination address register 1	DAR1	R/W	Undefined	H'FFA00014	H'1FA00014	32
	DMA transfer count register 1	DMATCR1	R/W	Undefined	H'FFA00018	H'1FA00018	32
	DMA channel control register 1	CHCR1	R/W*[1]	H'00000000	H'FFA0001C	H'1FA0001C	32
2	DMA source address register 2	SAR2	R/W	Undefined	H'FFA00020	H'1FA00020	32
	DMA destination address register 2	DAR2	R/W	Undefined	H'FFA00024	H'1FA00024	32
	DMA transfer count register 2	DMATCR2	R/W	Undefined	H'FFA00028	H'1FA00028	32
	DMA channel control register 2	CHCR2	R/W*[1]	H'00000000	H'FFA0002C	H'1FA0002C	32
3	DMA source address register 3	SAR3	R/W	Undefined	H'FFA00030	H'1FA00030	32
	DMA destination address register 3	DAR3	R/W	Undefined	H'FFA00034	H'1FA00034	32
	DMA transfer count register 3	DMATCR3	R/W	Undefined	H'FFA00038	H'1FA00038	32
	DMA channel control register 3	CHCR3	R/W*[1]	H'00000000	H'FFA0003C	H'1FA0003C	32
Common	DMA operation register	DMAOR	R/W*[1]	H'00000000	H'FFA00040	H'1FA00040	32

Notes: Longword access should be used for all control registers. If a different access width is used, reads will return all 0s and writes will not be possible.

1. Bit 1 of CHCR0–CHCR3 and bits 2 and 1 of DMAOR can only be written with 0 after being read as 1, to clear the flags.
2. In DDT mode, writes from the CPU are masked. Writes from external devices using the DTR format are possible.

14.2 Register Descriptions

14.2.1 DMA Source Address Registers 0–3 (SAR0–SAR3)

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

Bit:	23		0
		. . . .	
Initial value:	—	. . . .	—
R/W:	R/W	. . . .	R/W

DMA source address registers 0–3 (SAR0–SAR3) are 32-bit readable/writable registers that specify the source address of a DMA transfer. These registers have a counter feedback function, and during a DMA transfer they indicate the next source address. In single address mode, the SAR value is ignored when a device with DACK has been specified as the transfer source.

Specify a 16-bit, 32-bit, 64-bit, or 32-byte boundary address when performing a 16-bit, 32-bit, 64-bit, or 32-byte data transfer, respectively. If a different address is specified, an address error will be detected and the DMAC will halt.

The initial value of these registers after a power-on or manual reset is undefined. They retain their values in standby mode and deep sleep mode.

When transfer is performed from memory to an external device in DDT mode, DTR format [31:0] is set in SAR0 [31:0].

14.2.2 DMA Destination Address Registers 0–3 (DAR0–DAR3)

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

Bit:	23	. . . .	0
		. . . .	. . . .
Initial value:	—	. . . .	—
R/W:	R/W	. . . .	R/W

DMA destination address registers 0–3 (DAR0–DAR3) are 32-bit readable/writable registers that specify the destination address of a DMA transfer. These registers have a counter feedback function, and during a DMA transfer they indicate the next destination address. In single address mode, the DAR value is ignored when a device with DACK has been specified as the transfer destination.

Specify a 16-bit, 32-bit, 64-bit, or 32-byte boundary address when performing a 16-bit, 32-bit, 64-bit, or 32-byte data transfer, respectively. If a different address is specified, an address error will be detected and the DMAC will halt.

The initial value of these registers after a power-on or manual reset is undefined. They retain their values in standby mode and deep sleep mode.

When transfer is performed from an external device to memory in DDT mode, DTR format [31:0] is set in DAR0 [31:0].

Note: When a 16-bit, 32-bit, 64-bit, or 32-byte boundary address is specified, take care with the setting of bit 0, bits 1–0, bits 2–0, or bits 4–0, respectively. If an address specification that ignores boundary considerations is made, the DMAC will detect an address error and halt operation on all channels (DMAOR: address error flag AE = 1). The DMAC will also detect an address error and halt if an area 7 address is specified in an external data bus transfer, or if the address of a nonexistent on-chip peripheral module is specified.

14.2.3 DMA Transfer Count Registers 0–3 (DMATCR0–DMATCR3)

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

Bit:	23	22	21	20	19	18	17	16
Initial value:	—	—	—	—	—	—	—	—
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

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

DMA transfer count registers 0–3 (DMATCR0–DMATCR3) are 32-bit readable/writable registers that specify the transfer count for the corresponding channel (byte count, word count, longword count, quadword count, or 32-byte count). Specifying H'000001 gives a transfer count of 1, while H'000000 gives the maximum setting, 16,777,216 (16M) transfers. During DMAC operation, the remaining number of transfers is shown.

Bits 31–24 of these registers are reserved; they are always read as 0, and should only be written with 0.

The initial value of these registers after a power-on or manual reset is undefined. They retain their values in standby mode and deep sleep mode.

In DDT mode, DTR format [55:48] is set in DMATCR0 [7:0]

14.2.4 DMA Channel Control Registers 0–3 (CHCR0–CHCR3)

Bit:	31	30	29	28	27	26	25	24
	SSA2	SSA1	SSA0	STC	DSA2	DSA1	DSA0	DTC
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
	—	—	—	—	DS	RL	AM	AL
Initial value:	0	0	0	0	—	—	—	—
R/W:	R	R	R	R	R/W	(R/W)	R/W	(R/W)

Bit:	15	14	13	12	11	10	9	8
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	TM	TS2	TS1	TS0	—	IE	TE	DE
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R/W	R/(W)	R/W

Note: The TE bit can only be written with 0 after being read as 1, to clear the flag.
The RL, AM, AL, and DS bits may be absent, depending on the channel.

DMA channel control registers 0–3 (CHCR0–CHCR3) are 32-bit readable/writable registers that specify the operating mode, transfer method, etc., for each channel. Bits 31–28 and 27–24 indicate the source address and destination address, respectively; these settings are only valid when the transfer involves the CS5 or CS6 space and the relevant space has been specified as a PCMCIA interface space. In other cases, these bits should be cleared to 0. For details of the PCMCIA interface, see section 13.3.7, PCMCIA Interface, in section 13, Bus State Controller (BSC).

In DDT mode, CHCR0 is set according to the DTR format. (The following settings are fixed: CHCR0 [31:24] = 0, [18:16] = 0, [2] = 0, [1] = 0, [0] = 1)

Bits 18 and 16 are not present in CHCR2 and CHCR3. In CHCR2 and CHCR3, these bits cannot be modified (a write value of 0 should always be used) and are always read as 0.

These registers are initialized to H'00000000 by a power-on or manual reset. They retain their values in standby mode and deep sleep mode.

Bits 31 to 29—Source Address Space Attribute Specification (SSA2–SSA0): These bits specify the space attribute for PCMCIA access. These bits are only valid in the case of page mapping to PCMCIA connected to areas 5 and 6.

Bit 31: SSA2	Bit 30: SSA1	Bit 29: SSA0	Description
0	0	0	Reserved in PCMCIA access (Initial value)
		1	Dynamic bus sizing I/O space
	1	0	8-bit I/O space
		1	16-bit I/O space
1	0	0	8-bit common memory space
		1	16-bit common memory space
	1	0	8-bit attribute memory space
		1	16-bit attribute memory space

Bit 28—Source Address Wait Control Select (STC): Specifies CS5 or CS6 space wait control for PCMCIA access. This bit selects the wait control register in the BSC that performs area 5 and 6 wait cycle control.

Bit 28: STC	Description
0	C5 space wait cycle selection (Initial value) Settings of bits A5W2–A5W0 in wait control register 2 (WCR2), and bits A5PCW1–A5PCW0, A5TED2–A5TED0, and A5TEH2–A5TEH0 in the PCMCIA control register (PCR), are selected
1	C6 space wait cycle selection Settings of bits A6W2–A6W0 in wait control register 2 (WCR2), and bits A6PCW1–A6PCW0, A6TED2–A6TED0, and A6TEH2–A6TEH0 in the PCMCIA control register (PCR), are selected

Note: For details, see section 13.3.7, PCMCIA Interface.

Bits 27 to 25—Destination Address Space Attribute Specification (DSA2–DSA0): These bits specify the space attribute for PCMCIA access. These bits are only valid in the case of page mapping to PCMCIA connected to areas 5 and 6.

Bit 27: DSA2	Bit 26: DSA1	Bit 25: DSA0	Description	
0	0	0	Reserved in PCMCIA access	(Initial value)
		1	Dynamic bus sizing I/O space	
	1	0	8-bit I/O space	
		1	16-bit I/O space	
1	0	0	8-bit common memory space	
		1	16-bit common memory space	
	1	0	8-bit attribute memory space	
		1	16-bit attribute memory space	

Bit 24—Destination Address Wait Control Select (DTC): Specifies CS5 or CS6 space wait cycle control for PCMCIA access. This bit selects the wait control register in the BSC that performs area 5 and 6 wait cycle control.

Bit 24: DTC	Description	
0	C5 space wait cycle selection Settings of bits A5W2–A5W0 in wait control register 2 (WCR2), and bits A5PCW1–A5PCW0, A5TED2–A5TED0, and A5TEH2–A5TEH0 in the PCMCIA control register (PCR), are selected	(Initial value)
1	C6 space wait cycle selection Settings of bits A6W2–A6W0 in wait control register 2 (WCR2), and bits A6PCW1–A6PCW0, A6TED2–A6TED0, and A6TEH2–A6TEH0 in the PCMCIA control register (PCR), are selected	

Note: For details, see section 13.3.7, PCMCIA Interface.

Bits 23 to 20—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 19—$\overline{\text{DREQ}}$ Select (DS): Specifies either low level detection or falling edge detection as the sampling method for the $\overline{\text{DREQ}}$ pin used in external request mode.

In normal DMA mode, this bit is valid only in CHCR0 and CHCR1. In DDT mode, it is valid in CHCR0–CHCR3.

Bit 19: DS	Description	
0	Low level detection	(Initial value)
1	Falling edge detection	

Bit 18—Request Check Level (RL): Selects whether the DRAK signal (that notifies an external device of the acceptance of $\overline{\text{DREQ}}$) is an active-high or active-low output.

This bit is valid only in CHCR0 and CHCR1.

Bit 18: RL	Description	
0	DRAK is an active-high output	(Initial value)
1	DRAK is an active-low output	

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

In normal DMA mode, this bit is valid only in CHCR0 and CHCR1. In DDT mode, it is valid in CHCR1–CHCR3.

Bit 17: AM	Description	
0	DACK is output in read cycle	(Initial value)
1	DACK is output in write cycle	

Bit 16—Acknowledge Level (AL): Specifies the DACK (acknowledge) signal as active-high or active-low.

This bit is valid only in CHCR0 and CHCR1.

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

Bits 15 and 14—Destination Address Mode 1 and 0 (DM1, DM0): These bits specify incrementing/decrementing of the DMA transfer destination address. The specification of these bits is ignored when data is transferred from external memory to an external device in single address mode. For channel 0, in DDT mode, the settings are fixed at DM1 = 0 and DM0 = 1.

Bit 15: DM1	Bit 14: DM0	Description
0	0	Destination address fixed (Initial value)
	1	Destination address incremented (+1 in 8-bit transfer, +2 in 16-bit transfer, +4 in 32-bit transfer, +8 in 64-bit transfer, +32 in 32-byte burst transfer)
1	0	Destination address decremented (–1 in 8-bit transfer, –2 in 16-bit transfer, –4 in 32-bit transfer, –8 in 64-bit transfer, –32 in 32-byte burst transfer)
	1	Setting prohibited

Bits 13 and 12—Source Address Mode 1 and 0 (SM1, SM0): These bits specify incrementing/decrementing of the DMA transfer source address. The specification of these bits is ignored when data is transferred from an external device to external memory in single address mode. For channel 0, in DDT mode the settings are fixed at SM1 = 0 and SM0 = 1.

Bit 13: SM1	Bit 12: SM0	Description
0	0	Source address fixed (Initial value)
	1	Source address incremented (+1 in 8-bit transfer, +2 in 16-bit transfer, +4 in 32-bit transfer, +8 in 64-bit transfer, +32 in 32-byte burst transfer)
1	0	Source address decremented (–1 in 8-bit transfer, –2 in 16-bit transfer, –4 in 32-bit transfer, –8 in 64-bit transfer, –32 in 32-byte burst transfer)
	1	Setting prohibited

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

Bit 11: RS3	Bit 10: RS2	Bit 9: RS1	Bit 8: RS0	Description
0	0	0	0	External request, dual address mode*[1] (external address space → external address space) (Initial value)
			1	Setting prohibited
		1	0	External request, single address mode External address space → external device*[1,]*[3,]*[4]
			1	External request, single address mode External device → external address space*[1,]*[3,]*[4]
	1	0	0	Auto-request (external address space → external address space)*[2]
			1	Auto-request (external address space → on-chip peripheral module)*[2]
		1	0	Auto-request (on-chip peripheral module → external address space)*[2]
			1	Setting prohibited
1	0	0	0	SCI transmit-data-empty interrupt transfer request (external address space → SCTDR1)*[2]
			1	SCI receive-data-full interrupt transfer request (SCRDR1 → external address space)*[2]
		1	0	SCIF transmit-data-empty interrupt transfer request (external address space → SCFTDR2)*[2]
			1	SCIF receive-data-full interrupt transfer request (SCFRDR2 → external address space)*[2]
	1	0	0	TMU channel 2 (input capture interrupt, external address space → external address space)*[2]
			1	TMU channel 2 (input capture interrupt) (external address space → on-chip peripheral module)*[2]
		1	0	TMU channel 2 (input capture interrupt) (on-chip peripheral module → external address space)*[2]
			1	Setting prohibited

Notes: 1. External request specifications are valid only for channels 0 and 1. Requests are not accepted for channels 2 and 3 in normal DMA mode.
2. Dual address mode
3. In DDT mode, selection is possible with the DTR format [60] (R/W bit) specification for channel 0 only.
4. In DDT mode, an external request specification should be made for channels 1, 2, and 3. Only DTR format setting is possible for channel 0.

Bit 7—Transmit Mode (TM): Specifies the bus mode for transfer.

Bit 7: TM	Description
0	Cycle steal mode (Initial value)
1	Burst mode

Setting possible with DTR format [57:55] (MD bits)

Bits 6 to 4—Transmit Size 2 to 0 (TS2–TS0): These bits specify the transfer data size.

Bit 6: TS2	Bit 5: TS1	Bit 4: TS0	Description
0	0	0	Quadword size (64-bit) specification(Initial value)
		1	Byte size (8-bit) specification
	1	0	Word size (16-bit) specification
		1	Longword size (32-bit) specification
1	0	0	32-byte block transfer specification

Setting possible with DTR format [63:61] (SZ bits)

Bit 3—Reserved: This bit is always read as 0, and should only be written with 0.

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

Bit 2: IE	Description
0	Interrupt request not generated after number of transfers specified in DMATCR (Initial value) (CHCR0 only fixed in DDT mode)
1	Interrupt request generated after number of transfers specified in DMATCR

Bit 1—Transfer End (TE): This bit is set to 1 after the number of transfers specified in DMATCR. If the IE bit is set to 1 at this time, an interrupt request (DMTE) is generated.

If data transfer ends before TE is set to 1 (for example, due to an NMI interrupt, address error, or clearing of the DE bit or the DME bit in DMAOR), the TE bit is not set to 1. When this bit is 1, the transfer enabled state is not entered even if the DE bit is set to 1.

Bit 1: TE	Description	
0	Number of transfers specified in DMATCR not completed [Clearing conditions] • When 0 is written to TE after reading TE = 1 • In a power-on or manual reset, and in standby mode	(Initial value)
1	Number of transfers specified in DMATCR completed	

Bit 0—DMAC Enable (DE): Enables operation of the corresponding channel.

Bit 0: DE	Description	
0	Operation of corresponding channel is disabled	(Initial value)
1	Operation of corresponding channel is enabled	

When auto-request is specified (with RS3–RS0), transfer is begun when this bit is set to 1. In the case of an external request or on-chip peripheral module request, transfer is begun when a transfer request is issued after this bit is set to 1. Transfer can be suspended midway by clearing this bit to 0.

Even if the DE bit has been set, transfer is not enabled when TE is 1, when DME in DMAOR is 0, or when the NMIF or AE bit in DMAOR is 1.

For channel 0, in DDT mode this bit is set to 1 when a DTR format is received. DE remains set to 1 even if TE is set to 1. When the mode is switched from DDT mode to normal DMA mode (DDT bit = 0 in DMAOR), the DE bit must be cleared to 0.

14.2.5 DMA Operation Register (DMAOR)

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

Bit:	23	22	21	20	19	18	17	16
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	15	14	13	12	11	10	9	8
	DDT	—	—	—	—	—	PR1	PR0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R	R	R	R	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	AE	NMIF	DME
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/(W)	R/(W)	R/W

Note: The AE and NMIF bits can only be written with 0 after being read as 1, to clear the flags.

DMAOR is a 32-bit readable/writable register that specifies the DMAC transfer mode.

DMAOR is initialized to H'00000000 by a power-on or manual reset. They retain their values in standby mode and deep sleep mode.

Bits 31 to 16—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 15—On-Demand Data Transfer (DDT): Specifies on-demand data transfer mode. When the DDT bit is set to 1, CPU writes to SAR0, DAR0, DMATCR0, and CHCR0 are masked.

Bit 15: DDT	Description	
0	Normal DMA mode	(Initial value)
1	On-demand data transfer mode	

Note: $\overline{\text{BAVL}}$ (DRAK0) is an active-high output in normal DMA mode. When the DDT bit is set to 1, the $\overline{\text{BAVL}}$ pin function is enabled and this pin becomes an active-low output.

Bits 14 to 10—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 9 and 8—Priority Mode 1 and 0 (PR1, PR0): These bits determine the order of priority for channel execution when transfer requests are made for a number of channels simultaneously.

Bit 9: PR1	Bit 8: PR0	Description	
0	0	CH0 > CH1 > CH2 > CH3	(Initial value)
	1	CH0 > CH2 > CH3 > CH1	
1	0	CH2 > CH0 > CH1 > CH3	
	1	Round robin mode	

Bits 7 to 3—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 2—Address Error Flag (AE): Indicates that an address error has occurred during DMA transfer. If this bit is set during data transfer, transfers on all channels are suspended, and an interrupt request (DMAE) is generated. The CPU cannot write 1 to AE. This bit can only be cleared by writing 0 after reading 1.

Bit 2: AE	Description	
0	No address error, DMA transfer enabled [Clearing condition] When 0 is written to AE after reading AE = 1	(Initial value)
1	Address error, DMA transfer disabled [Setting condition] When an address error is caused by the DMAC	

Bit 1—NMI Flag (NMIF): Indicates that NMI has been input. This bit is set regardless of whether or not the DMAC is operating. If this bit is set during data transfer, transfers on all channels are suspended. The CPU cannot write 1 to NMIF. This bit can only be cleared by writing 0 after reading 1.

Bit 1: NMIF	Description	
0	No NMI input, DMA transfer enabled [Clearing condition] When 0 is written to NMIF after reading NMIF = 1	(Initial value)
1	NMI input, DMA transfer disabled [Setting condition] When an NMI interrupt is generated	

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

Even if the DME bit has been set, transfer is not enabled when TE is 1 or DE is 0 in CHCR, or when the NMI or AE bit in DMAOR is 1.

Bit 0: DME	Description	
0	Operation disabled on all channels	(Initial value)
1	Operation enabled on all channels	

14.3 Operation

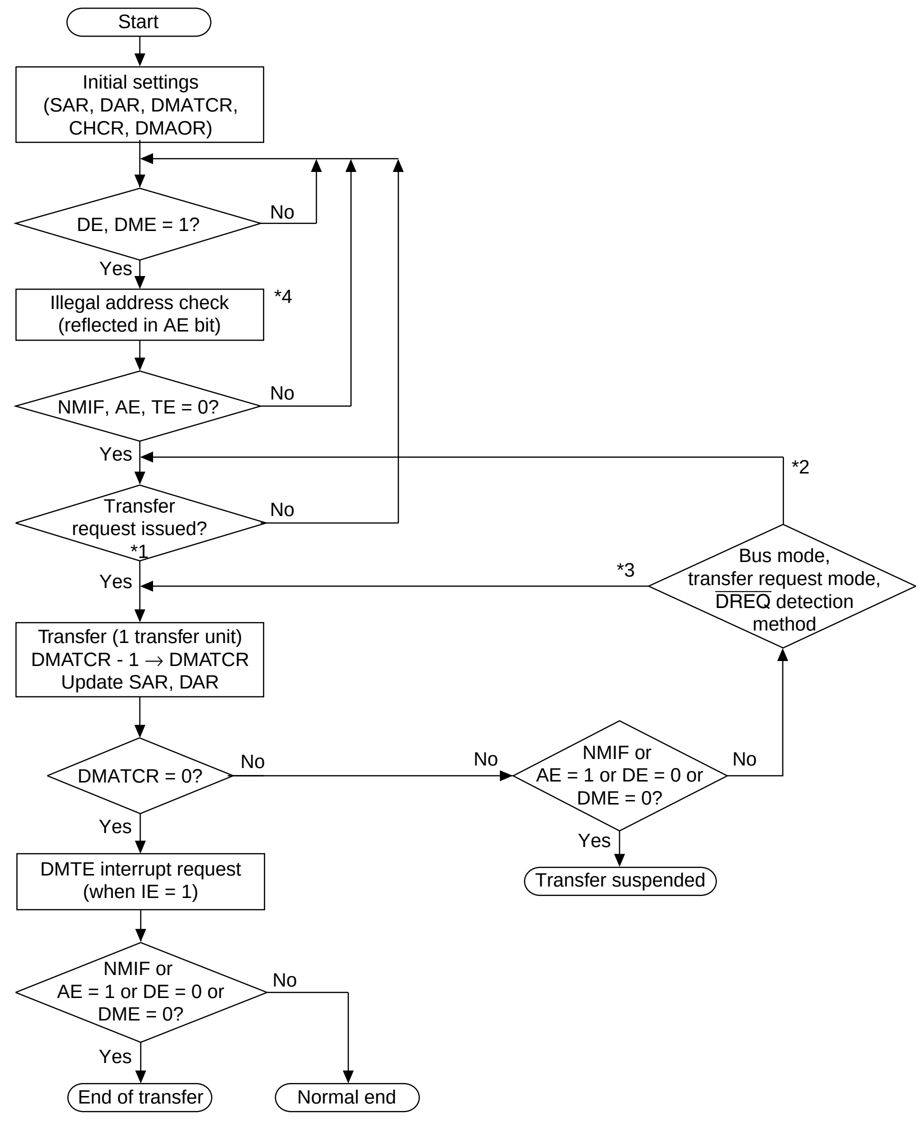
When a DMA transfer request is issued, the DMAC starts the transfer according to the predetermined channel priority order. It ends the transfer when the transfer end conditions are satisfied. Transfers can be requested in three modes: auto-request, external request, and on-chip peripheral module request. There are two modes for DMA transfer: single address mode and dual address mode. Either burst mode or cycle steal mode can be selected as the bus mode.

14.3.1 DMA Transfer Procedure

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

1. The DMAC checks to see if transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0).
2. When a transfer request is issued and transfer has been enabled, the DMAC transfers one transfer unit of data (determined by the setting of TS2–TS0). In auto-request mode, the transfer begins automatically when the DE bit and DME bit are set to 1. The DMATCR value is decremented by 1 for each transfer. The actual transfer flow depends on the address mode and bus mode.
3. When the specified number of transfers have been completed (when the DMATCR value reaches 0), the transfer ends normally. If the IE bit in CHCR is set to 1 at this time, a DMTE interrupt request is sent to the CPU.
4. If a DMAC address error or NMI interrupt occurs, the transfer is suspended. Transfer is also suspended when the DE bit in CHCR or the DME bit in DMAOR is cleared to 0. In the event of an address error, a DMAE interrupt request is forcibly sent to the CPU.

Figure 14.2 shows a flowchart of this procedure.



Start
Initial settings (SAR, DAR, DMATCR, CHCR, DMAOR)
DE, DME = 1?
No
Yes
Illegal address check (reflected in AE bit)
*4
NMIF, AE, TE = 0?
No
Yes
*2
Transfer request issued?
*1
No
Yes
*3
Bus mode, transfer request mode, DREQ detection method
Transfer (1 transfer unit) DMATCR - 1 → DMATCR Update SAR, DAR
DMATCR = 0?
No
No
NMIF or AE = 1 or DE = 0 or DME = 0?
No
Yes
Transfer suspended
Yes
DMTE interrupt request (when IE = 1)
NMIF or AE = 1 or DE = 0 or DME = 0?
No
Yes
End of transfer
Normal end

Notes: 1. In auto-request mode, transfer begins when the NMIF, AE, and TE bits are all 0, and the DE and DME bits are set to 1.
2. $\overline{\text{DREQ}}$ level detection (external request) in burst mode, or cycle steal mode.
3. $\overline{\text{DREQ}}$ edge detection (external request) in burst mode, or auto-request mode in burst mode.
4. An illegal address is detected by comparing bits TS2–TS0 in CHCRn with SARn and DARn.

Figure 14.2 DMAC Transfer Flowchart

14.3.2 DMA Transfer Requests

DMA transfer requests are basically generated at either the data transfer source or destination, but they can also be issued by external devices or on-chip peripheral modules that are neither the source nor the destination.

Transfers can be requested in three modes: auto-request, external request, and on-chip peripheral module request. The transfer request mode is selected by means of bits RS3–RS0 in 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 bit in CHCR0–CHCR3 and the DME bit in the DMA operation register (DMAOR) are set to 1, the transfer begins (so long as the TE bit in CHCR0–CHCR3 and the NMIF and AE bits in DMAOR are all 0).

External Request Mode: In this mode a transfer is performed in response to a transfer request signal ($\overline{\text{DREQ}}$) from an external device. One of the modes shown in table 14.4 should be chosen according to the application system. If DMA transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0), transfer starts when $\overline{\text{DREQ}}$ is input. The DS bit in CHCR0/CHCR1 is used to select either falling edge detection or low level detection for the $\overline{\text{DREQ}}$ signal (level detection when DS = 0, edge detection when DS = 1).

The source of the transfer request does not have to be the data transfer source or destination.

Table 14.4 Selecting External Request Mode with RS Bits

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

On-Chip Peripheral Module Request Mode: In this mode a transfer is performed in response to a transfer request signal (interrupt request signal) from an on-chip peripheral module. As shown in table 14.5, there are seven transfer request signals: input capture interrupts from the timer unit (TMU), and receive-data-full interrupts (RXI) and transmit-data-empty interrupts (TXI) from the two serial communication interfaces (SCI, SCIF). If DMA transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0, AE = 0), transfer starts when a transfer request signal is input.

The source of the transfer request does not have to be the data transfer source or destination. However, when the transfer request is set to RXI (transfer request by SCI/SCIF receive-data-full interrupt), the transfer source must be the SCI/SCIF's receive data register (SCRDR1/SCFRDR2). When the transfer request is set to TXI (transfer request by SCI/SCIF transmit-data-empty interrupt), the transfer destination must be the SCI/SCIF's transmit data register (SCTDR1/SCFTDR2).

Table 14.5 Selecting On-Chip Peripheral Module Request Mode with RS Bits

RS3	RS2	RS1	RS0	DMAC Transfer Request Source	DMAC Transfer Request Signal	Transfer Source	Transfer Destination	Bus Mode
1	0	0	0	SCI transmitter	SCTDR1 (SCI transmit-data-empty transfer request)	External*	SCTDR1	Cycle steal mode
			1	SCI receiver	SCRDR1 (SCI receive-data-full transfer request)	SCRDR1	External*	Cycle steal mode
		1	0	SCIF transmitter	SCFTDR2 (SCIF transmit-data-empty transfer request)	External*	SCFTDR2	Cycle steal mode
			1	SCIF receiver	SCFRDR2 (SCIF receive-data-full transfer request)	SCFRDR2	External*	Cycle steal mode
	1	0	0	TMU channel 2	Input capture occurrence	External*	External*	Burst/cycle steal mode
			1	TMU channel 2	Input capture occurrence	External*	On-chip peripheral	Burst/cycle steal mode
		1	0	TMU channel 2	Input capture occurrence	On-chip peripheral	External*	Burst/cycle steal mode

TMU: Timer unit

SCI: Serial communication interface

SCIF: Serial communication interface with FIFO

Note: * External memory or memory-mapped external device

Note: SCI/SCIF burst transfer setting is prohibited.

To output a transfer request from an on-chip peripheral module, set the DMA transfer request enable bit for that module and output a transfer request signal.

For details, see sections 12, Timer Unit (TMU), 15, Serial Communication Interface (SCI), and 16, Serial Communication Interface with FIFO (SCIF).

When a DMA transfer corresponding to a transfer request signal from an on-chip peripheral module shown in table 14.5 is carried out, the signal is discontinued automatically. This occurs every transfer in cycle steal mode, and in the last transfer in burst mode.

14.3.3 Channel Priorities

If the DMAC receives simultaneous transfer requests on two or more channels, it selects a channel according to a predetermined priority system, either in a fixed mode or round robin mode. The mode is selected with priority bits PR1 and PR0 in the DMA operation register (DMAOR).

Fixed Mode: In this mode, the relative channel priorities remain fixed. The following priority orders are available in fixed mode:

- CH0 > CH1 > CH2 > CH3
- CH0 > CH2 > CH3 > CH1
- CH2 > CH0 > CH1 > CH3

The priority order is selected with bits PR1 and PR0 in DMAOR.

Round Robin Mode: In round robin mode, each time the transfer of one transfer unit (byte, word, longword, quadword, or 32 bytes) ends on a given channel, that channel is assigned the lowest priority level. This is illustrated in figure 14.3. The order of priority in round robin mode immediately after a reset is CH0 > CH1 > CH2 > CH3.

Note: In round robin mode, if no transfer request is accepted for any channel during DMA transfer, the priority order becomes CH0 > CH1 > CH2 > CH3.

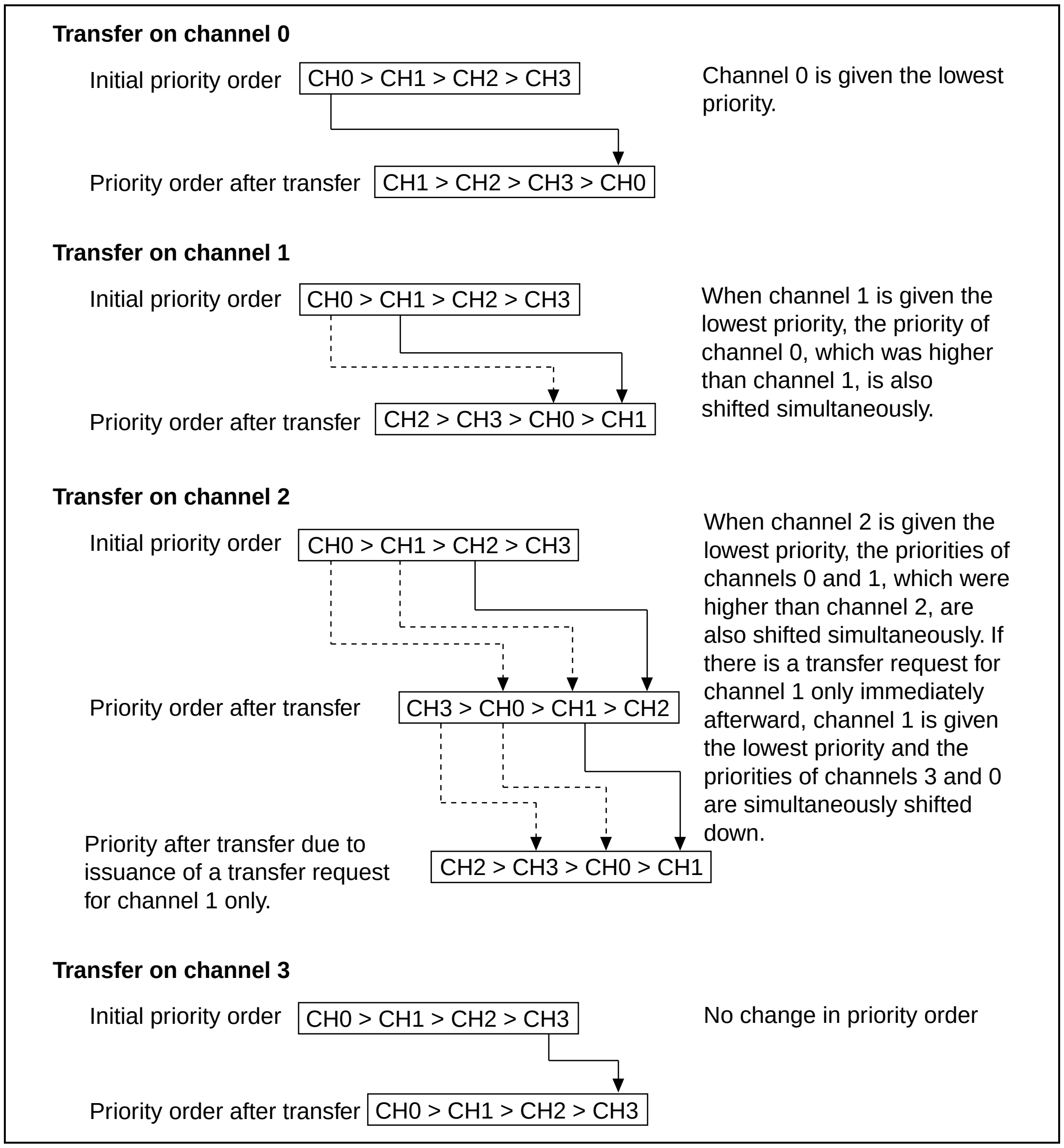

Transfer on channel 0
Initial priority order
CH0 > CH1 > CH2 > CH3
Channel 0 is given the lowest priority.
Priority order after transfer
CH1 > CH2 > CH3 > CH0
Transfer on channel 1
Initial priority order
CH0 > CH1 > CH2 > CH3
When channel 1 is given the lowest priority, the priority of channel 0, which was higher than channel 1, is also shifted simultaneously.
Priority order after transfer
CH2 > CH3 > CH0 > CH1
Transfer on channel 2
Initial priority order
CH0 > CH1 > CH2 > CH3
When channel 2 is given the lowest priority, the priorities of channels 0 and 1, which were higher than channel 2, are also shifted simultaneously. If there is a transfer request for channel 1 only immediately afterward, channel 1 is given the lowest priority and the priorities of channels 3 and 0 are simultaneously shifted down.
Priority order after transfer
CH3 > CH0 > CH1 > CH2
Priority after transfer due to issuance of a transfer request for channel 1 only.
CH2 > CH3 > CH0 > CH1
Transfer on channel 3
Initial priority order
CH0 > CH1 > CH2 > CH3
No change in priority order
Priority order after transfer
CH0 > CH1 > CH2 > CH3

Figure 14.3 Round Robin Mode

Figure 14.4 shows the changes in priority levels when transfer requests are issued simultaneously for channels 0 and 3, and channel 1 receives a transfer request during a transfer on channel 0. The operation of the DMAC in this case is as follows.

1. Transfer requests are issued simultaneously for channels 0 and 3.
2. Since channel 0 has a higher priority level than channel 3, the channel 0 transfer is executed first (channel 3 is on transfer standby).
3. A transfer request is issued for channel 1 during the channel 0 transfer (channels 1 and 3 are on transfer standby).
4. At the end of the channel 0 transfer, channel 0 shifts to the lowest priority level.
5. At this point, channel 1 has a higher priority level than channel 3, so the channel 1 transfer is started (channel 3 is on transfer standby).
6. At the end of the channel 1 transfer, channel 1 shifts to the lowest priority level.
7. The channel 3 transfer is started.
8. At the end of the channel 3 transfer, the channel 3 and channel 2 priority levels are lowered, giving channel 3 the lowest priority.

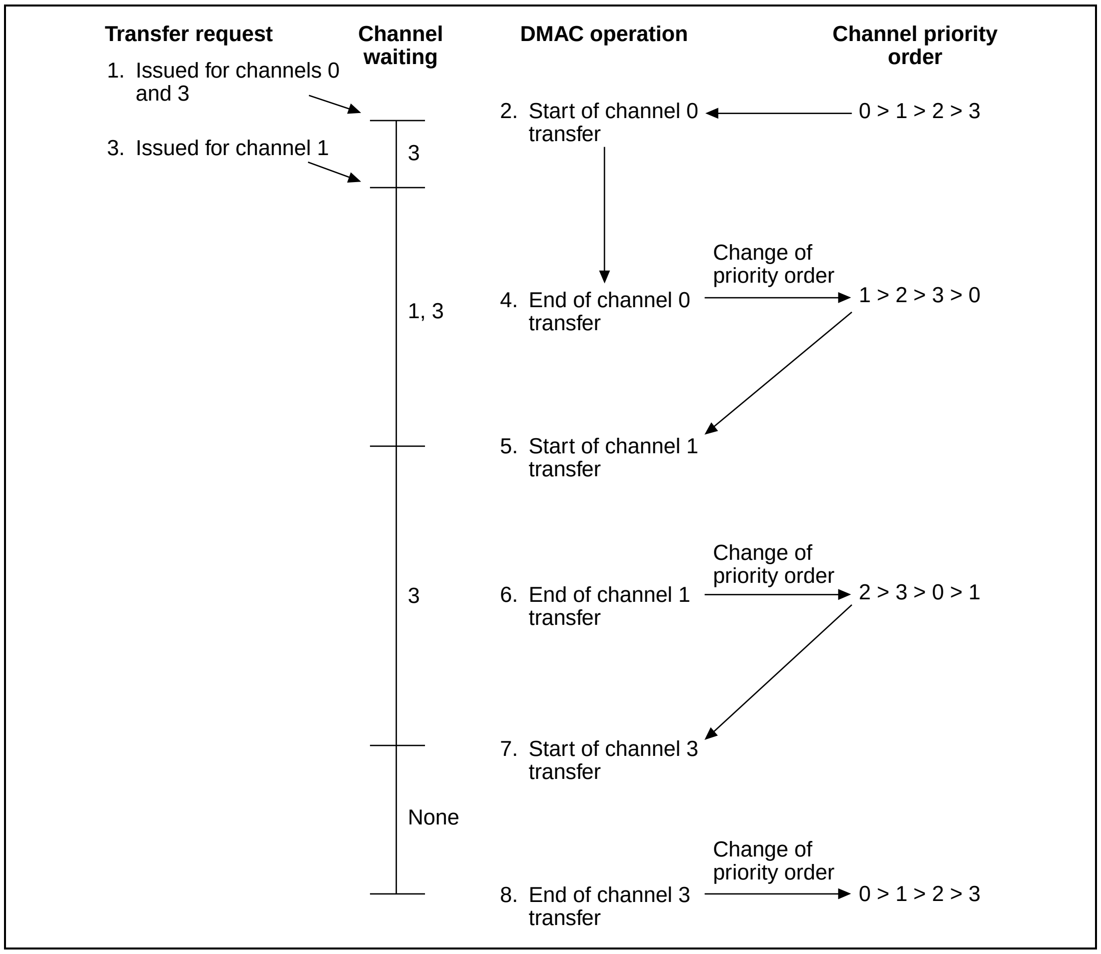


Figure 14.4 Example of Changes in Priority Order in Round Robin Mode

14.3.4 Types of DMA Transfer

The DMAC supports the transfers shown in table 14.6. It can operate in single address mode, in which either the transfer source or the transfer destination is accessed using the acknowledge signal, or in dual address mode, in which both the transfer source and transfer destination addresses are output. The actual transfer operation timing depends on the bus mode, which can be either burst mode or cycle steal mode.

Table 14.6 Supported DMA Transfers

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

Address Modes

Single Address Mode: In single address mode, both the transfer source and the transfer destination are external; one is accessed by the DACK signal and the other by an address. In this mode, the DMAC performs a DMA transfer in one bus cycle by simultaneously outputting the external device strobe signal (DACK) to either the transfer source or transfer destination external device to access it, while outputting an address to the other side of the transfer. Figure 14.5 shows an example of a transfer between external memory and an external device with DACK in which the external device outputs data to the data bus and that data is written to external memory in the same bus cycle.

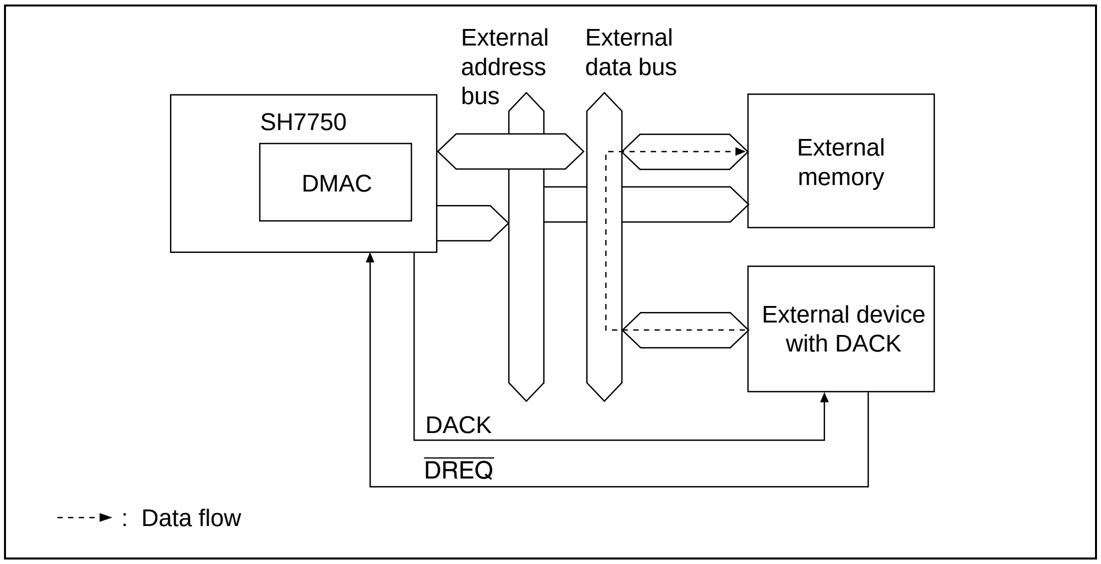
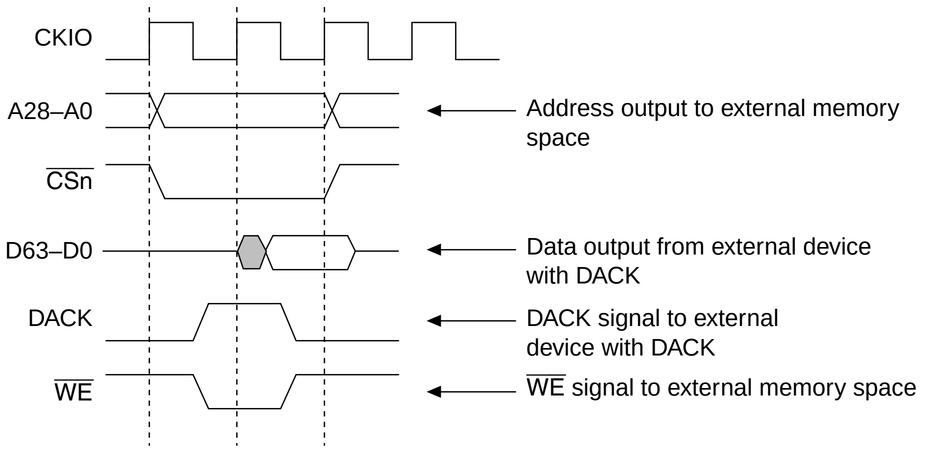


Figure 14.5 Data Flow in Single Address Mode

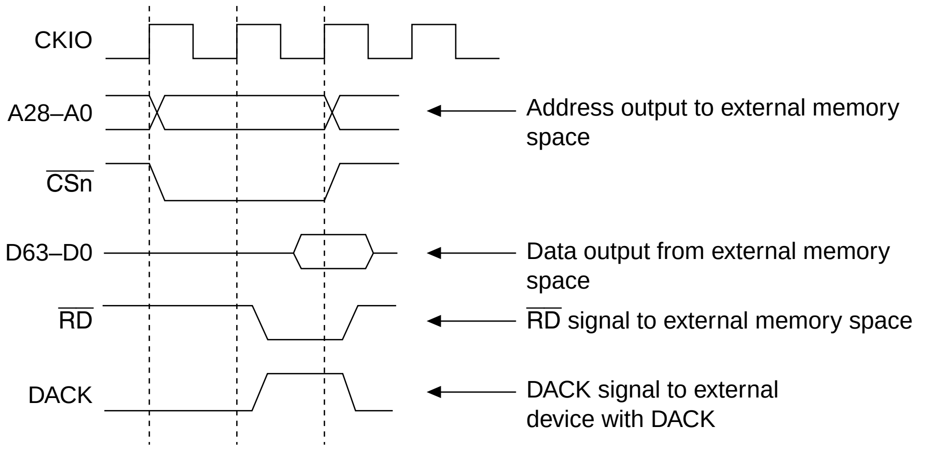
Two types of transfer are possible in single address mode: (1) transfer between an external device with DACK and a memory-mapped external device, and (2) transfer between an external device with DACK and external memory. Only the external request signal ($\overline{\text{DREQ}}$) is used in both these cases.

Figure 14.6 shows the transfer timing for single address mode.

The access timing depends on the type of external memory. For details, see the descriptions of the memory interfaces in section 13, Bus State Controller (BSC).


CKIO
A28–A0
$\overline{\text{CSn}}$
D63–D0
DACK
$\overline{\text{WE}}$
Address output to external memory space
Data output from external device with DACK
DACK signal to external device with DACK
$\overline{\text{WE}}$ signal to external memory space

(a) From external device with DACK to external memory space


CKIO
A28–A0
$\overline{\text{CSn}}$
D63–D0
$\overline{\text{RD}}$
DACK
Address output to external memory space
Data output from external memory space
$\overline{\text{RD}}$ signal to external memory space
DACK signal to external device with DACK

(b) From external memory space to external device with DACK

Figure 14.6 DMA Transfer Timing in Single Address Mode

Dual Address Mode: Dual address mode is used to access both the transfer source and the transfer destination by address. The transfer source and destination can be accessed by either on-chip peripheral module or external address.

In dual address mode, data is read from the transfer source in the data read cycle, and written to the transfer destination in the data write cycle, so that the transfer is executed in two bus cycles. The transfer data is temporarily stored in the data buffer in the bus state controller (BSC).

In a transfer between external memories such as that shown in figure 14.7, data is read from external memory into the BSC's data buffer in the read cycle, then written to the other external memory in the write cycle. Figure 14.8 shows the timing for this operation.

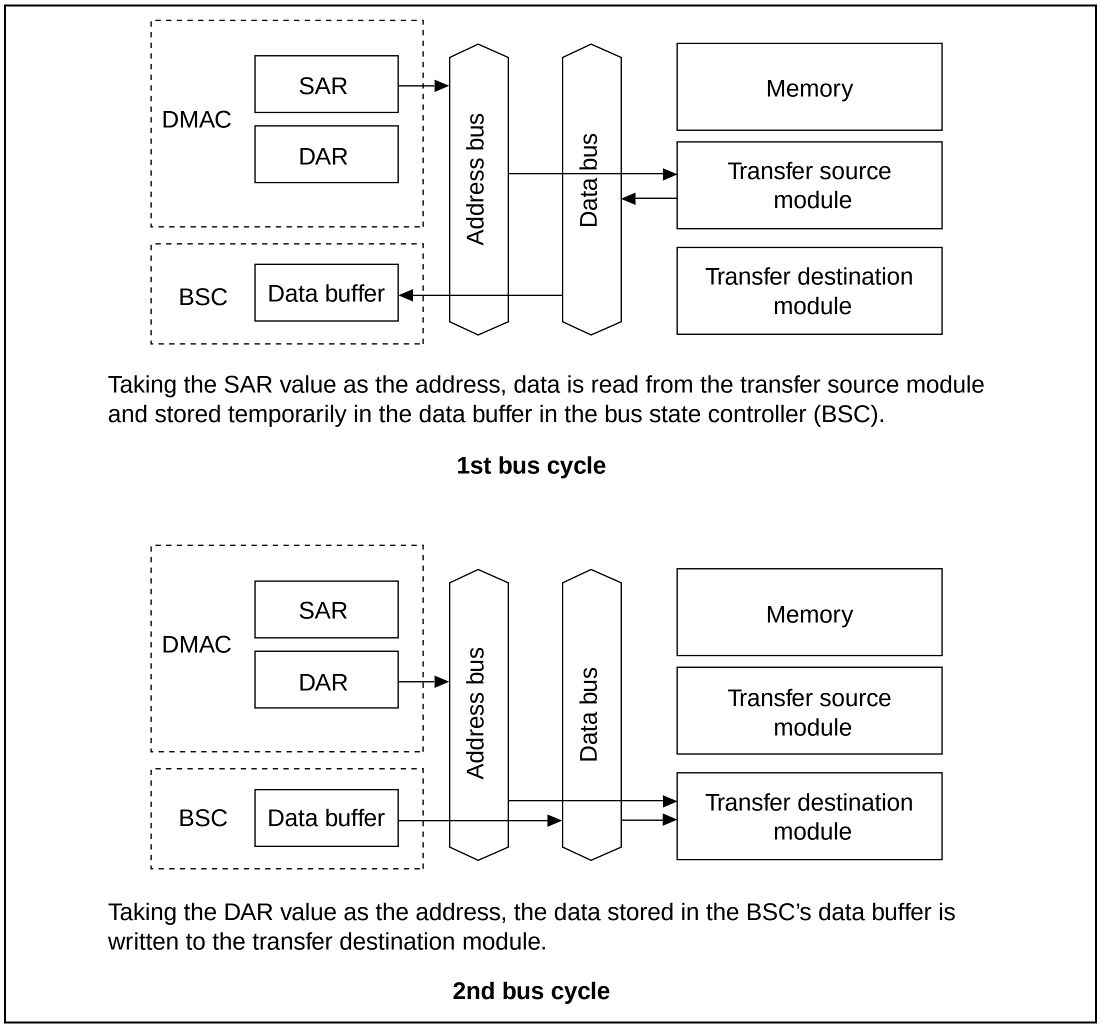

DMAC
SAR
DAR
BSC
Data buffer
Address bus
Data bus
Memory
Transfer source module
Transfer destination module
Taking the SAR value as the address, data is read from the transfer source module and stored temporarily in the data buffer in the bus state controller (BSC).
1st bus cycle
DMAC
SAR
DAR
BSC
Data buffer
Address bus
Data bus
Memory
Transfer source module
Transfer destination module
Taking the DAR value as the address, the data stored in the BSC's data buffer is written to the transfer destination module.
2nd bus cycle

Figure 14.7 Operation in Dual Address Mode

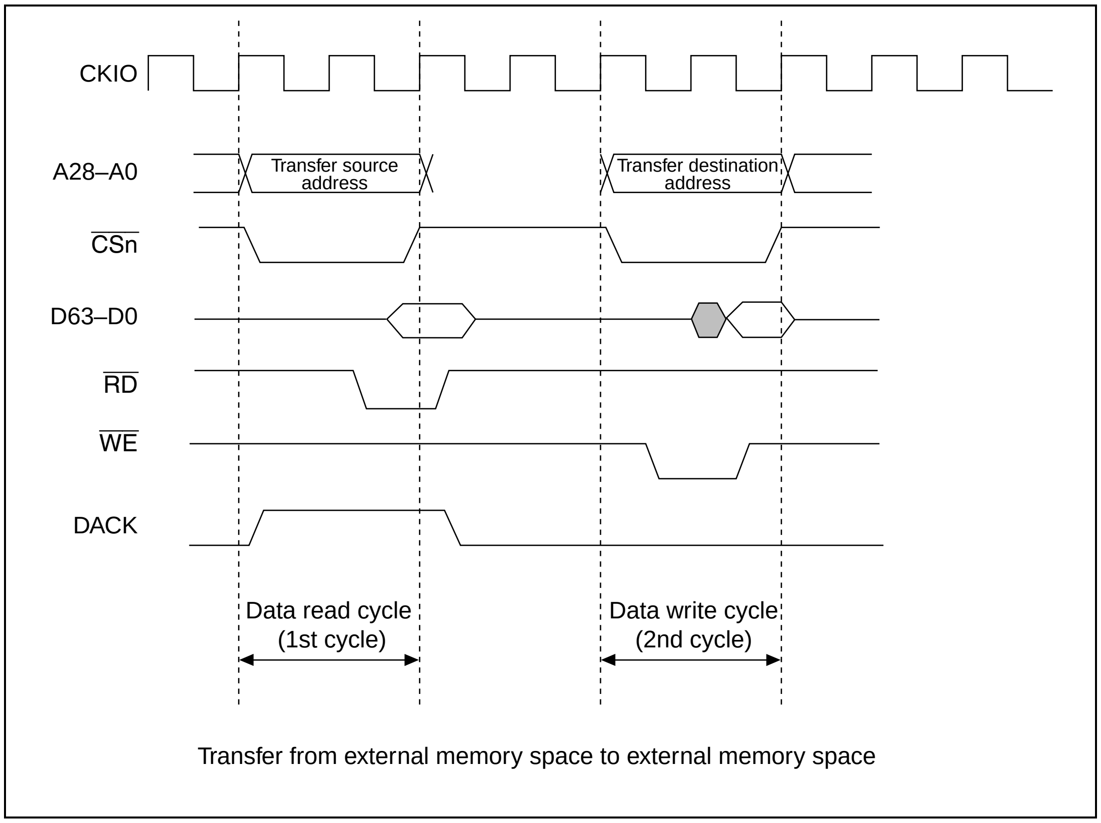

CKIO
A28–A0
Transfer source address
Transfer destination address
$\overline{\text{CSn}}$
D63–D0
$\overline{\text{RD}}$
$\overline{\text{WE}}$
DACK
Data read cycle (1st cycle)
Data write cycle (2nd cycle)
Transfer from external memory space to external memory space

Figure 14.8 Example of Transfer Timing in Dual Address Mode

Bus Modes

There are two bus modes, cycle steal mode and burst mode, selected with the TM bit in CHCR0–CHCR3.

Cycle Steal Mode: In cycle steal mode, the DMAC releases the bus to the CPU at the end of each transfer-unit (8-bit, 16-bit, 32-bit, 64-bit, or 32-byte) transfer. When the next transfer request is issued, the DMAC reacquires the bus from the CPU and carries out another transfer-unit transfer. At the end of this transfer, the bus is again given to the CPU. This is repeated until the transfer end condition is satisfied.

Cycle steal mode can be used with all categories of transfer request source, transfer source, and transfer destination.

Figure 14.9 shows an example of DMA transfer timing in cycle steal mode. The transfer conditions in this example are dual address mode and $\overline{\text{DREQ}}$ level detection.

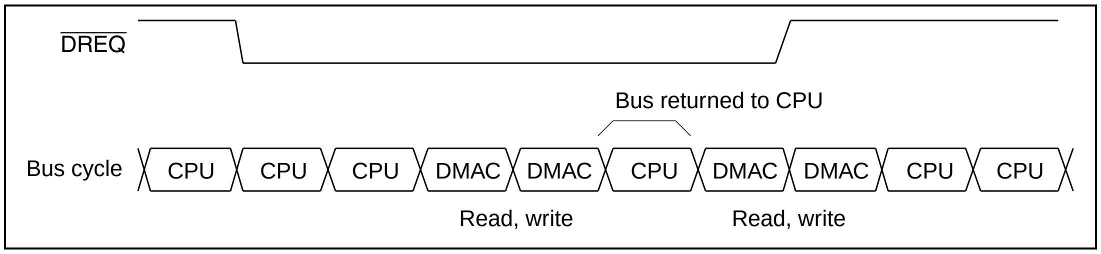


Figure 14.9 Example of DMA Transfer in Cycle Steal Mode

Burst Mode: In burst mode, once the DMAC has acquired the bus it holds the bus and transfers data continuously until the transfer end condition is satisfied. With $\overline{\text{DREQ}}$ low level detection in external request mode, however, when $\overline{\text{DREQ}}$ is driven high the bus passes to another bus master after the end of the DMAC transfer request that has already been accepted, even if the transfer end condition has not been satisfied.

Figure 14.10 shows an example of DMA transfer timing in burst mode. The transfer conditions in this example are single address mode and $\overline{\text{DREQ}}$ level detection.

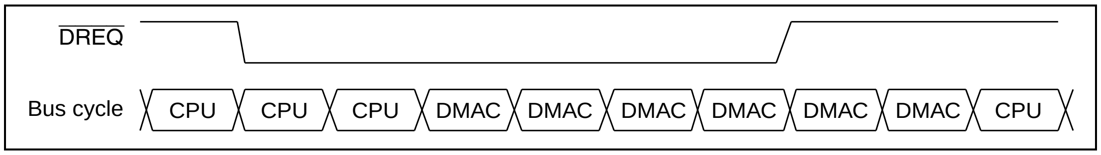


Figure 14.10 Example of DMA Transfer in Burst Mode

Note: Burst mode can be set regardless of the data size. A 32-byte block transfer burst mode setting can also be made.

Relationship between DMA Transfer Type, Request Mode, and Bus Mode

Table 14.7 shows the relationship between the type of DMA transfer, the request mode, and the bus mode.

Table 14.7 Relationship between DMA Transfer Type, Request Mode, and Bus Mode

Address Mode	Type of Transfer	Request Mode	Bus Mode	Transfer Size (Bits)	Usable Channels
Single	External device with DACK and external memory	External	B/C	8/16/32/64/32B	0, 1 (2, 3)*[6]
	External device with DACK and memory-mapped external device	External	B/C	8/16/32/64/32B	0, 1 (2, 3)*[6]
Dual	External memory and external memory	Any*[1]	B/C	8/16/32/64/32B	0, 1, 2, 3*[5,]*[6]
	External memory and memory-mapped external device	Any*[1]	B/C	8/16/32/64/32B	0, 1, 2, 3*[5,]*[6]
	Memory-mapped external device and memory-mapped external device	Any*[1]	B/C	8/16/32/64/32B	0, 1, 2, 3*[5,]*[6]
	External memory and on-chip peripheral module	Any*[2]	B/C*[3]	8/16/32/64*[4]	0, 1, 2, 3*[5,]*[6]
	Memory-mapped external device and on-chip peripheral module	Any*[2]	B/C*[3]	8/16/32/64*[4]	0, 1, 2, 3*[5,]*[6]

32B: 32-byte burst transfer

B: Burst

C: Cycle steal

Notes: 1. External request, auto-request, or on-chip peripheral module request (TMU input capture interrupt request) possible. In the case of an on-chip peripheral module request, it is not possible to specify external memory data transfer with the SCI (SCIF) as the transfer request source.

2. External request, auto-request, or on-chip peripheral module request possible. If the transfer request source is the SCI (SCIF), either the transfer source must be SCRDR1 (SCFRDR2) or the transfer destination must be SCTDR1 (SCFTDR2).

3. When the transfer request source is the SCI (SCIF), only cycle steal mode can be used.

4. Access size permitted for the on-chip peripheral module register that is the transfer source or transfer destination.

5. When the transfer request is an external request, only channels 0 and 1 can be used.

6. In DDT mode, transfer requests can be accepted for all channels from external devices capable of DTR format output.

Bus Mode and Channel Priority Order

When, for example, channel 1 is transferring data in burst mode, and a transfer request is issued to channel 0, which has a higher priority, the channel 0 transfer is started immediately.

If fixed mode has been set for the priority levels (CH0 > CH1), transfer on channel 1 is continued after transfer on channel 0 is completely finished, whether cycle steal mode or burst mode is set for channel 0.

If round robin mode has been set for the priority levels, transfer on channel 1 is restarted after one transfer unit of data is transferred on channel 0, whether cycle steal mode or burst mode is set for channel 0. Channel execution alternates in the order: channel 1 → channel 0 → channel 1 → channel 0.

An example of round robin mode operation is shown in figure 14.11.

Since channel 1 is in burst mode (in the case of edge sensing) regardless of whether fixed mode or round robin mode is set for the priority order, the bus is not released to the CPU until channel 1 transfer ends.

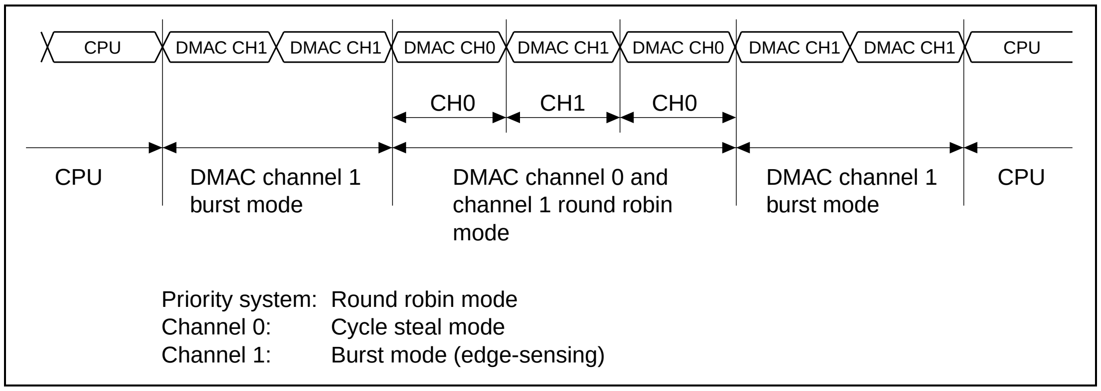

CPU
DMAC CH1
DMAC CH1
DMAC CH0
DMAC CH1
DMAC CH0
DMAC CH1
DMAC CH1
CPU
CH0
CH1
CH0
CPU
DMAC channel 1 burst mode
DMAC channel 0 and channel 1 round robin mode
DMAC channel 1 burst mode
CPU
Priority system: Round robin mode
Channel 0: Cycle steal mode
Channel 1: Burst mode (edge-sensing)

Figure 14.11 Bus Handling with Two DMAC Channels Operating

Note: When channel 1 is in level-sensing burst mode with the settings shown in figure 14.11, the bus is passed to the CPU during a break in requests.

14.3.5 Number of Bus Cycle States and $\overline{\text{DREQ}}$ Pin Sampling Timing

Number of States in Bus Cycle: The number of states in the bus cycle when the DMAC is the bus master is controlled by the bus state controller (BSC) just as it is when the CPU is the bus master. See section 13, Bus State Controller (BSC), for details.

$\overline{\text{DREQ}}$ Pin Sampling Timing: In external request mode, the $\overline{\text{DREQ}}$ pin is sampled at the rising edge of CKIO clock pulses. When $\overline{\text{DREQ}}$ input is detected, a DMAC bus cycle is generated and DMA transfer executed after four CKIO cycles at the earliest.

The second and subsequent $\overline{\text{DREQ}}$ sampling operations are performed one cycle after the start of the first DMAC transfer bus cycle (in the case of single address mode).

DRAK is output for one cycle only, once each time $\overline{\text{DREQ}}$ is detected, regardless of the transfer mode or $\overline{\text{DREQ}}$ detection method. In the case of burst mode edge detection, $\overline{\text{DREQ}}$ is sampled in the first cycle only, and so DRAK is output in the first cycle only .

Operation: Figures 14.12 to 14.23 show the timing in each mode.

1. Cycle Steal Mode

   In cycle steal mode, The $\overline{\text{DREQ}}$ sampling timing differs for dual address mode and single address mode, and for level detection and edge detection of $\overline{\text{DREQ}}$.

   For example, in figure 14.12 (cycle steal mode, dual address mode, level detection), DMAC transfer begins, at the earliest, four CKIO cycles after the first sampling operation. The second sampling operation is performed one cycle after the start of the first DMAC transfer write cycle. If $\overline{\text{DREQ}}$ is not detected at this time, sampling is executed in every subsequent cycle.

   In figure 14.13 (cycle steal mode, dual address mode, edge detection), DMAC transfer begins, at the earliest, five CKIO cycles after the first sampling operation. The second sampling operation begins from the cycle in which the first DMAC transfer read cycle ends. If $\overline{\text{DREQ}}$ is not detected at this time, sampling is executed in every subsequent cycle.

   In figure 14.16 (cycle steal mode, dual address mode, level detection), with SDRAM: row hit read/write transfer using a 64-bit bus width and a 32-byte block as the data size, DMAC transfer begins, at the earliest, four CKIO cycles after the first sampling operation. The second sampling operation is performed in the cycle in which the first DMAC transfer write cycle is begun.

   For details of the timing for various kinds of memory access, see section 13, Bus State Controller (BSC).

   Figure 14.19 shows the case of cycle steal mode, single address mode, and level detection. In this case, too, transfer is started, at the earliest, four CKIO cycles after the first $\overline{\text{DREQ}}$ sampling operation. The second sampling operation is performed one cycle after the start of the first DMAC transfer bus cycle.

Figure 14.20 shows the case of cycle steal mode, single address mode, and edge detection. In this case, transfer is started, at the earliest, five CKIO cycles after the first $\overline{\text{DREQ}}$ sampling operation. The second sampling begins one cycle after the first assertion of DRAK.

In single address mode, the DACK signal is output every DMAC transfer cycle.

2. Burst Mode, Dual Address Mode, Level Detection

   $\overline{\text{DREQ}}$ sampling timing in burst mode using dual address mode and level detection is virtually the same as for cycle steal mode.

   For example, in figure 14.14, DMAC transfer begins, at the earliest, four CKIO cycles after the first sampling operation. The second sampling operation is performed one cycle after the start of the first DMAC transfer write cycle.

   In the case of dual address mode transfer initiated by an external request, the DACK signal can be output in either the read cycle or the write cycle of the DMAC transfer according to the specification of the AM bit in CHCR.

3. Burst Mode, Single Address Mode, Level Detection

   $\overline{\text{DREQ}}$ sampling timing in burst mode using single address mode and level detection is shown in figure 14.21.

   In the example shown in figure 14.21, DMAC transfer begins, at the earliest, four CKIO cycles after the first sampling operation, and the second sampling operation begins one cycle after the start of the first DMAC transfer bus cycle.

   In single address mode, the DACK signal is output every DMAC transfer cycle.

   In figure 14.23, with a 32-byte data size, 64-bit bus width, and SDRAM: row hit write, DMAC transfer begins, at the earliest, six CKIO cycles after the first sampling operation. The second sampling operation begins one cycle after DACK is asserted for the first DMAC transfer.

4. Burst Mode, Dual Address Mode, Edge Detection

   In burst mode using dual address mode and edge detection, $\overline{\text{DREQ}}$ sampling is performed in the first cycle only.

   For example, in the case shown in figure 14.15, DMAC transfer begins, at the earliest, five CKIO cycles after the first sampling operation. DMAC transfer then continues until the end of the number of data transfers set in DMATCR. $\overline{\text{DREQ}}$ is not sampled during this time, and therefore DRAK is output in the first cycle only.

   In the case of dual address mode transfer initiated by an external request, the DACK signal can be output in either the read cycle or the write cycle of the DMAC transfer according to the specification of the AM bit in CHCR.

5. Burst Mode, Single Address Mode, Edge Detection

   In burst mode using single address mode and edge detection, $\overline{\text{DREQ}}$ sampling is performed only in the first cycle.

   For example, in the case shown in figure 14.22, DMAC transfer begins, at the earliest, five cycles after the first sampling operation. DMAC transfer then continues until the end of the number of data transfers set in DMATCR. $\overline{\text{DREQ}}$ is not sampled during this time, and therefore DRAK is output in the first cycle only.

   In single address mode, the DACK signal is output every DMAC transfer cycle.

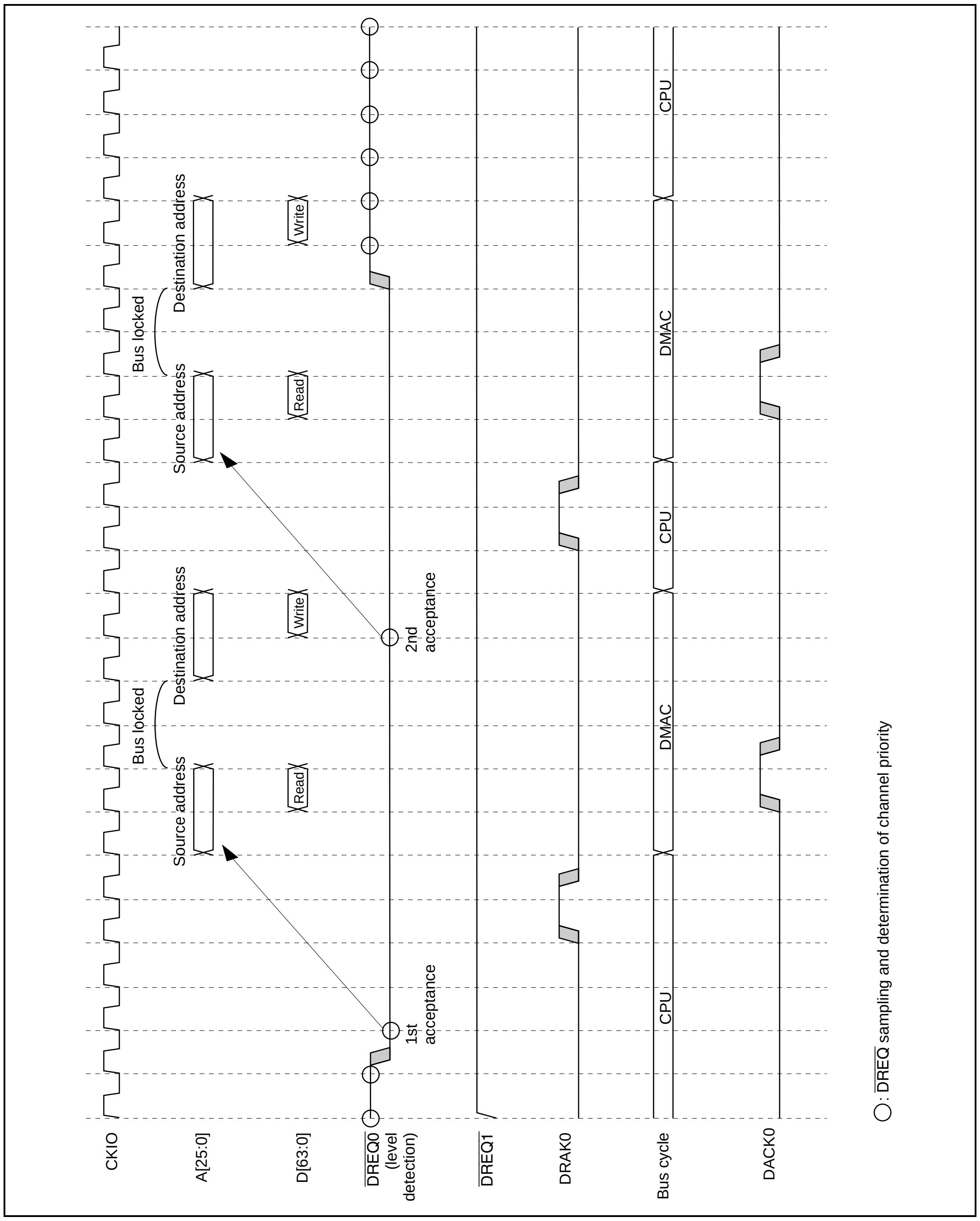

CKIO
A[25:0]
D[63:0]
$\overline{\text{DREQ0}}$ (level detection)
$\overline{\text{DREQ1}}$
DRAK0
Bus cycle
DACK0
Source address
Destination address
Bus locked
Read
Write
1st acceptance
2nd acceptance
CPU
DMAC
○: $\overline{\text{DREQ}}$ sampling and determination of channel priority

Figure 14.12 Dual Address Mode/Cycle Steal Mode
External Bus → External Bus/$\overline{\text{DREQ}}$ (Level Detection), DACK (Read Cycle)

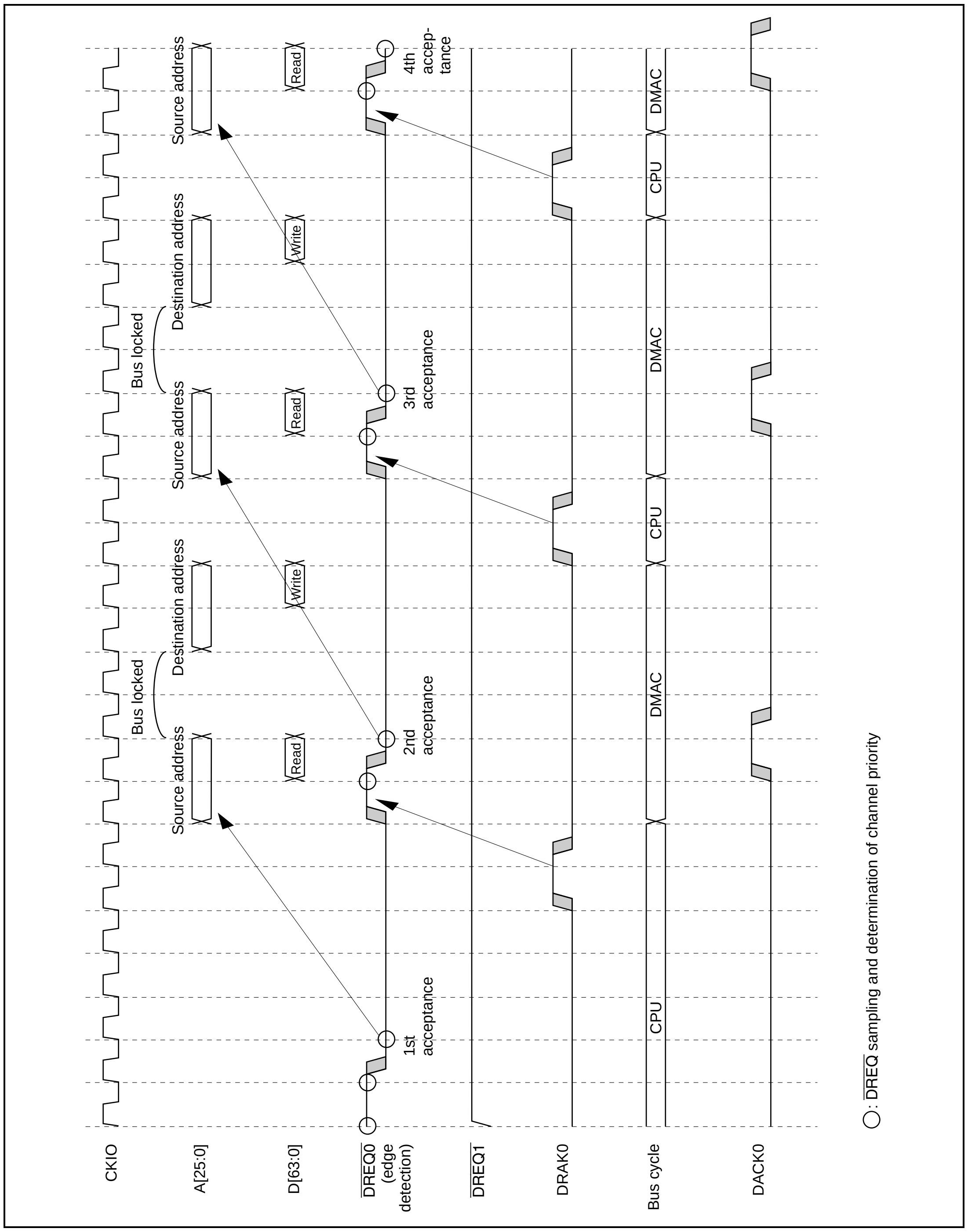


Figure 14.13 Dual Address Mode/Cycle Steal Mode External Bus → External Bus/$\overline{\text{DREQ}}$ (Edge Detection), DACK (Read Cycle)

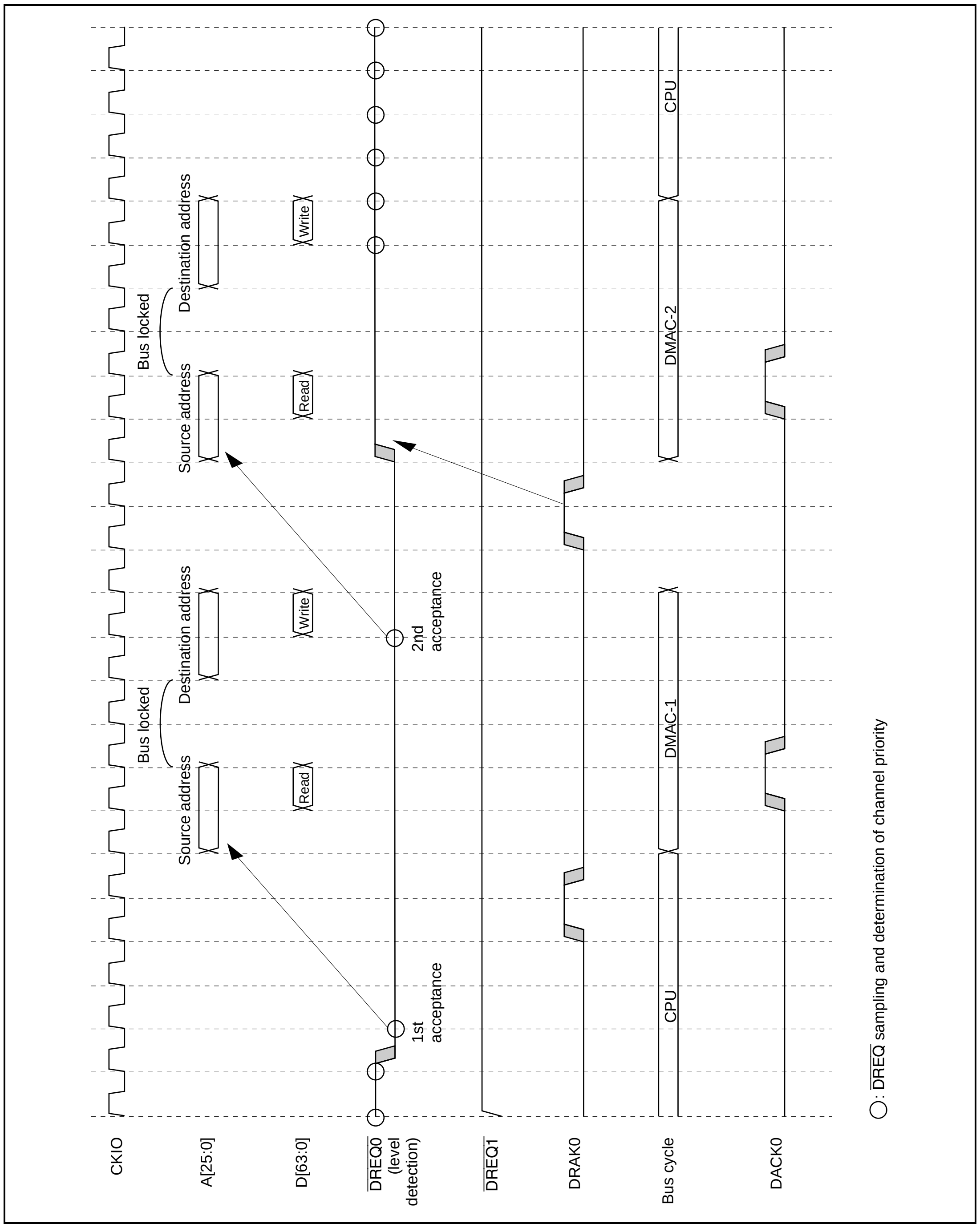


Figure 14.14 Dual Address Mode/Burst Mode
External Bus → External Bus/$\overline{\text{DREQ}}$ (Level Detection), DACK (Read Cycle)

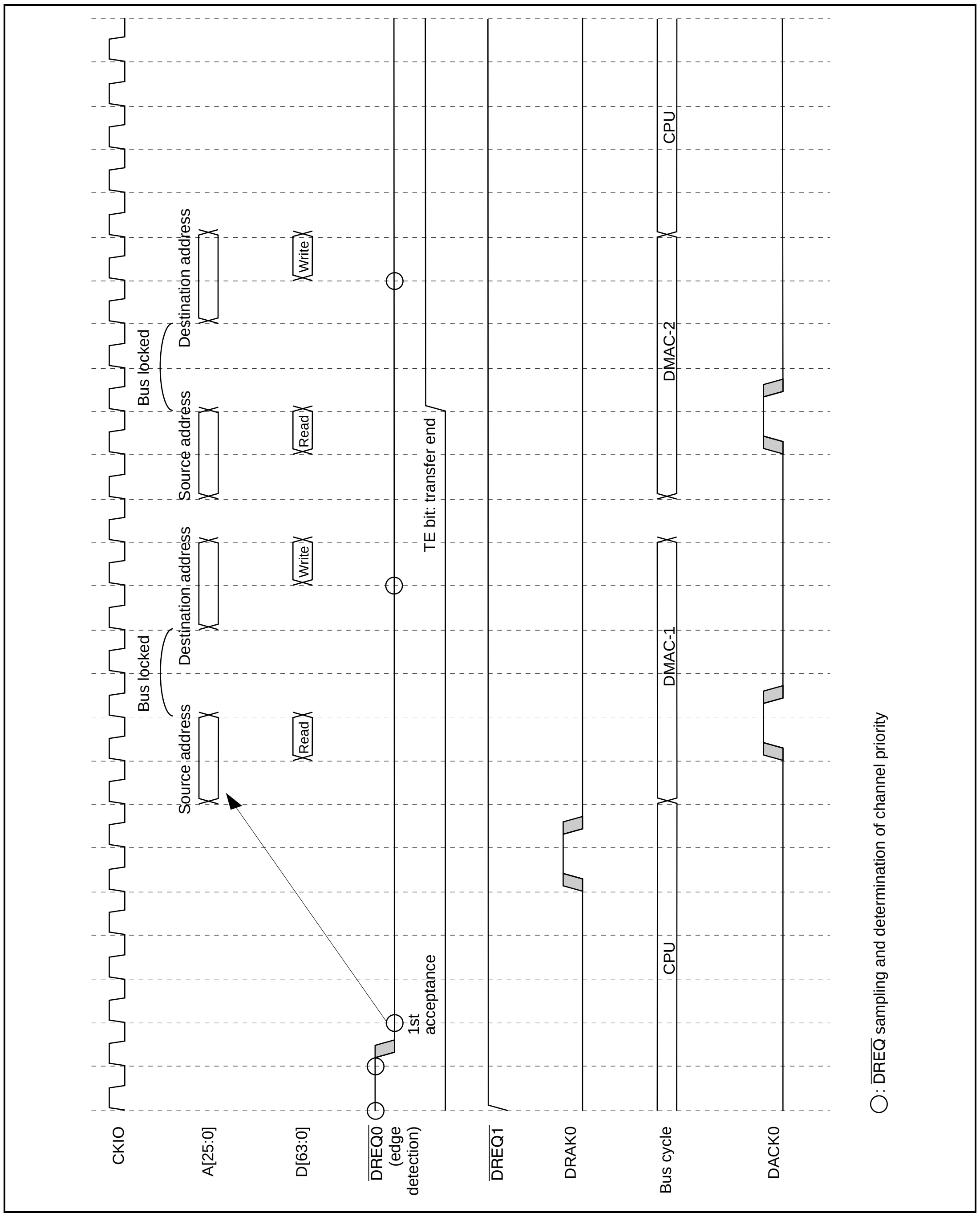


Figure 14.15 Dual Address Mode/Burst Mode
External Bus → External Bus/$\overline{\text{DREQ}}$ (Edge Detection), DACK (Read Cycle)

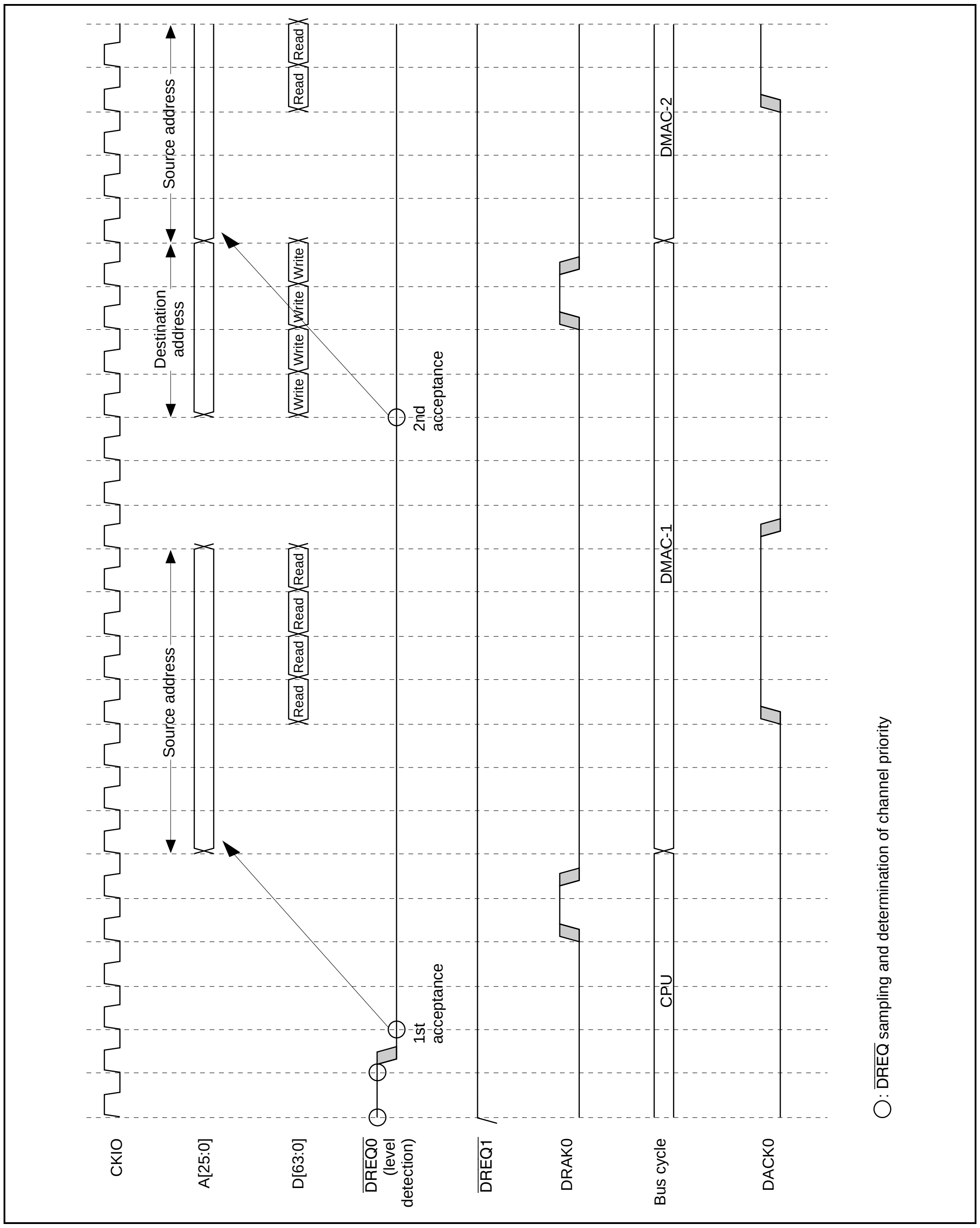


Figure 14.16 Dual Address Mode/Cycle Steal Mode External Bus → External Bus/$\overline{\text{DREQ}}$ (Level Detection)/32-Byte Block Transfer (Bus Width: 64 Bits, SDRAM: Row Hit Read/Write), DACK (Read Cycle)

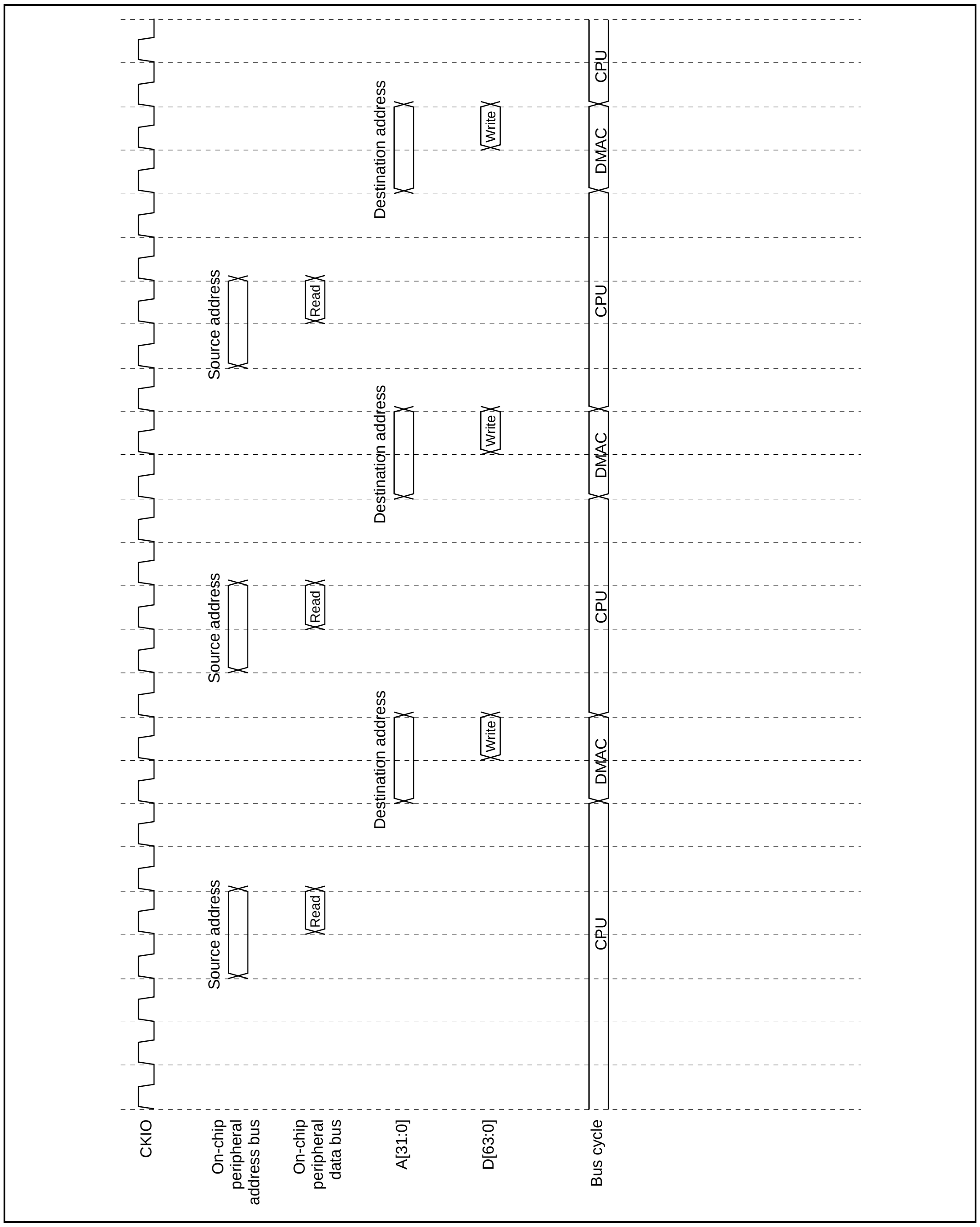


Figure 14.17 Dual Address Mode/Cycle Steal Mode On-Chip SCI (Level Detection) → External Bus

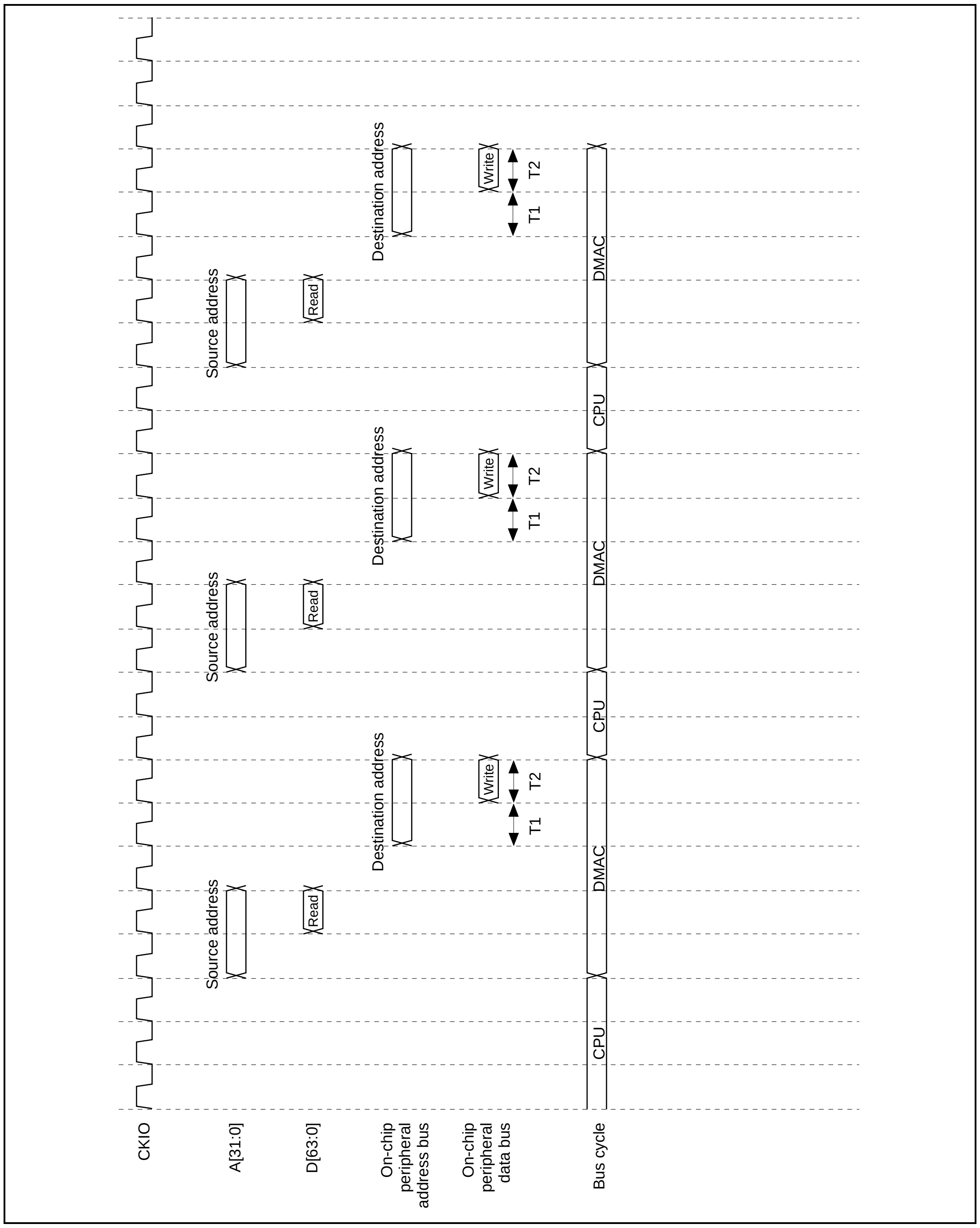


Figure 14.18 Dual Address Mode/Cycle Steal Mode External Bus → On-Chip SCI (Level Detection)

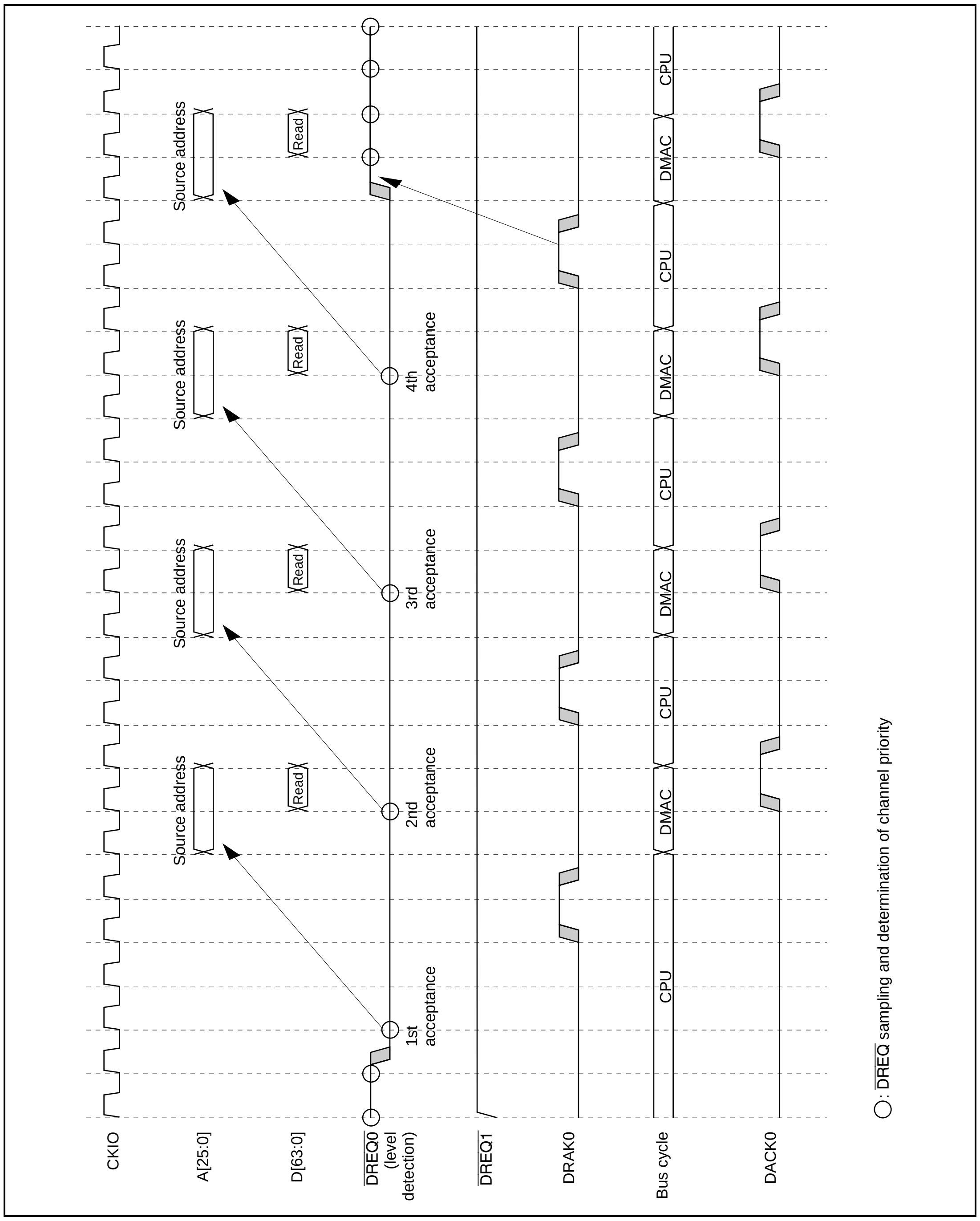


Figure 14.19 Single Address Mode/Cycle Steal Mode External Bus → External Bus/$\overline{\text{DREQ}}$ (Level Detection)

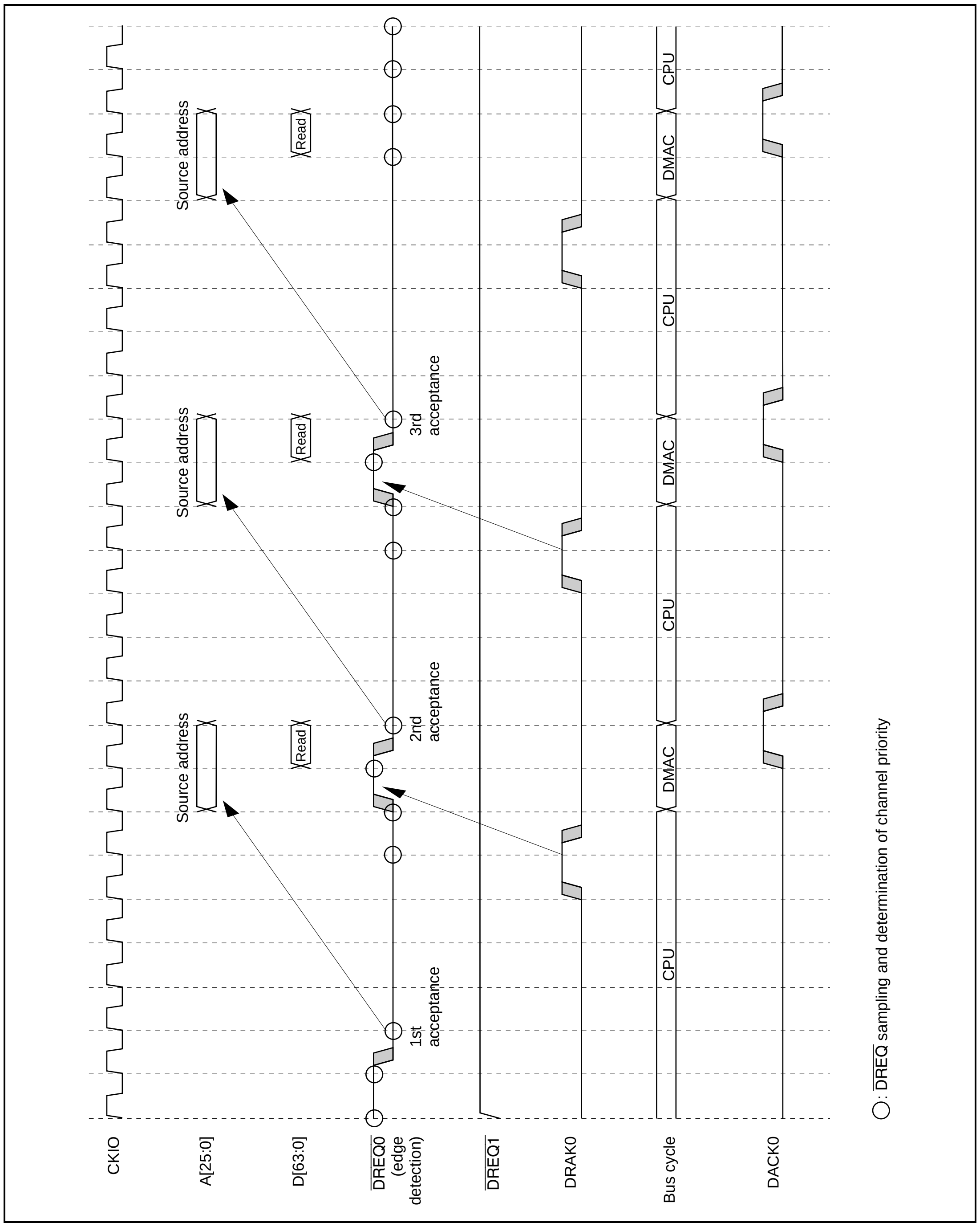


Figure 14.20 Single Address Mode/Cycle Steal Mode External Bus → External Bus/$\overline{\text{DREQ}}$ (Edge Detection)

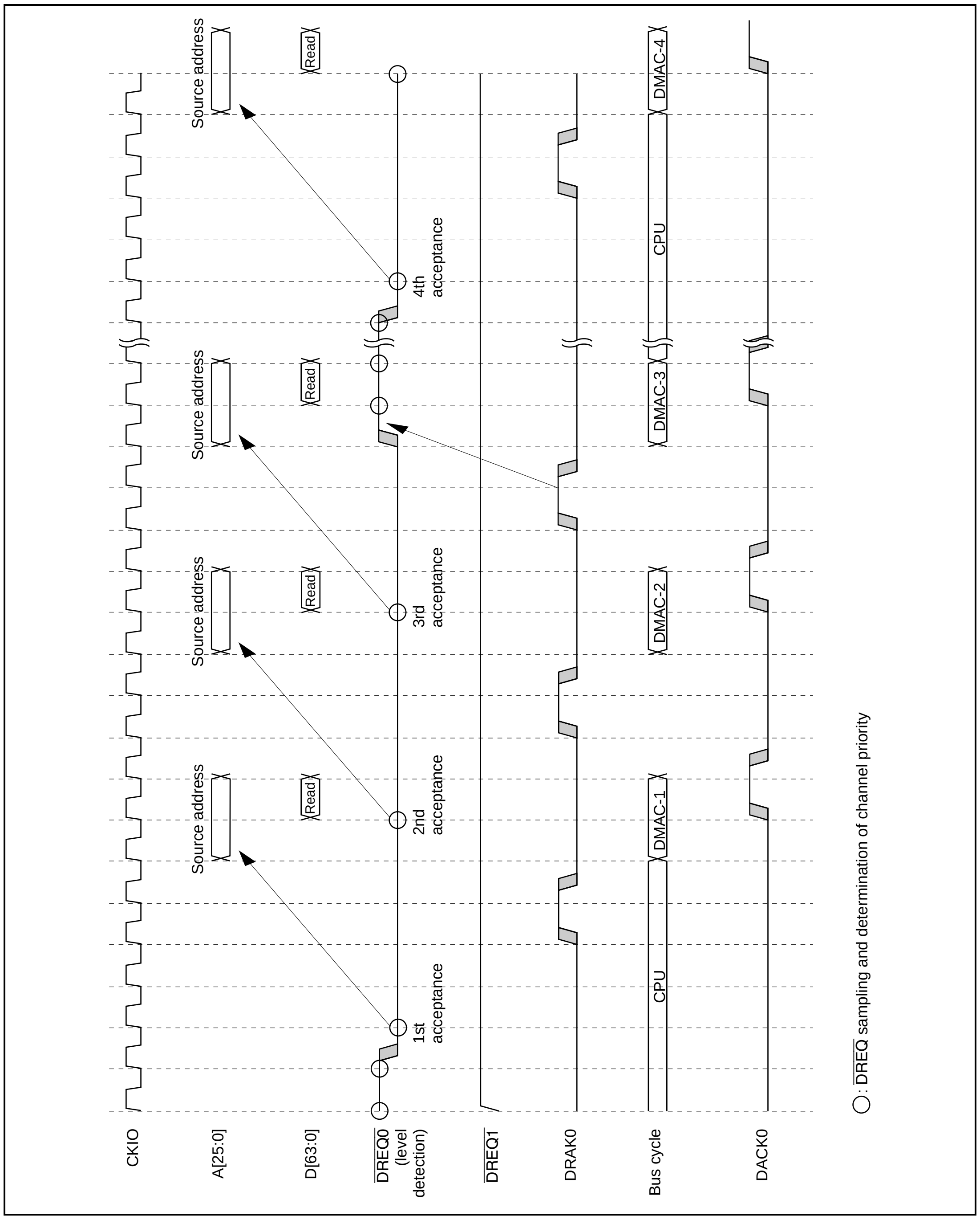


Figure 14.21 Single Address Mode/Burst Mode External Bus → External Bus/$\overline{\text{DREQ}}$ (Level Detection)

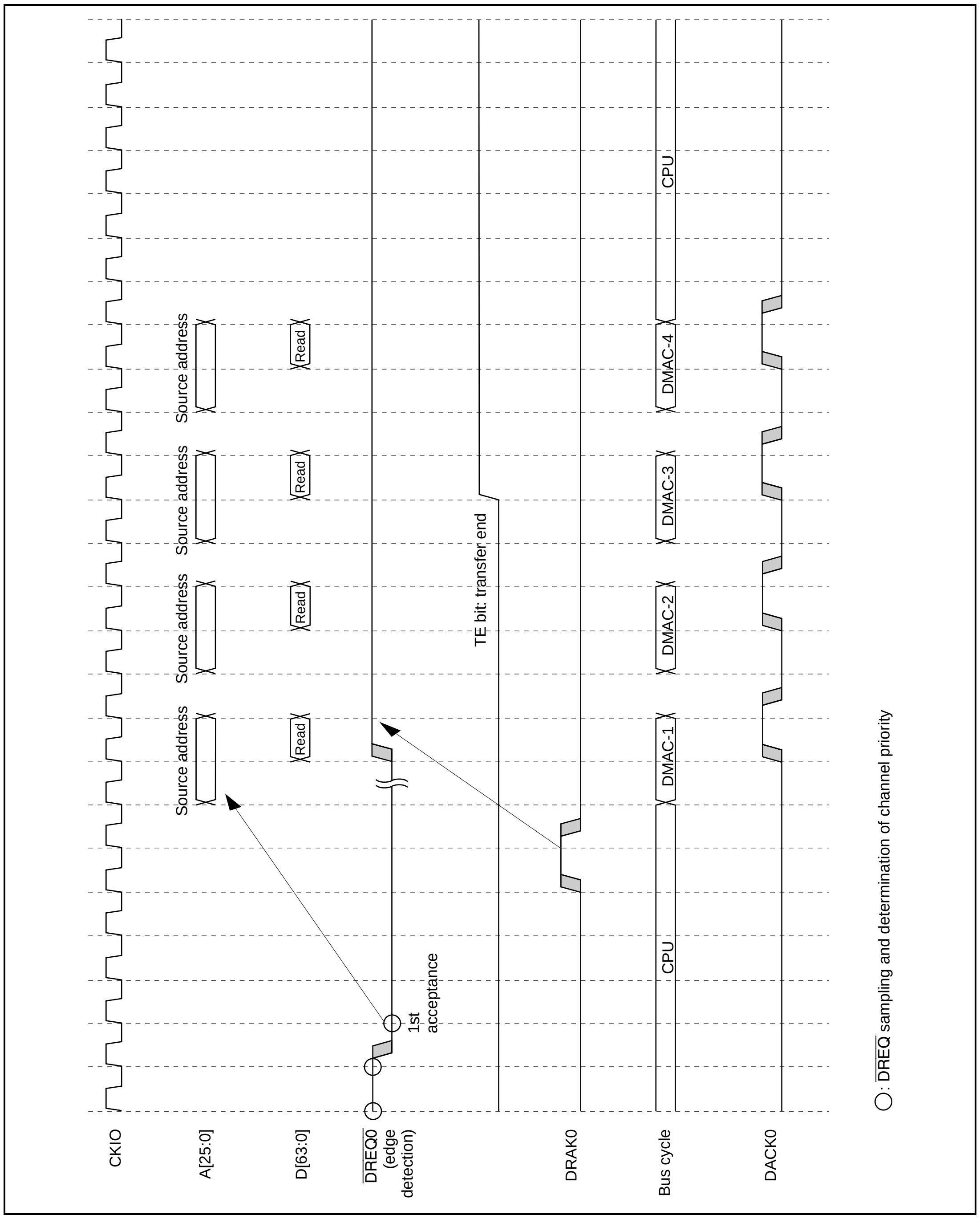


Figure 14.22 Single Address Mode/Burst Mode External Bus → External Bus/$\overline{\text{DREQ}}$ (Edge Detection)

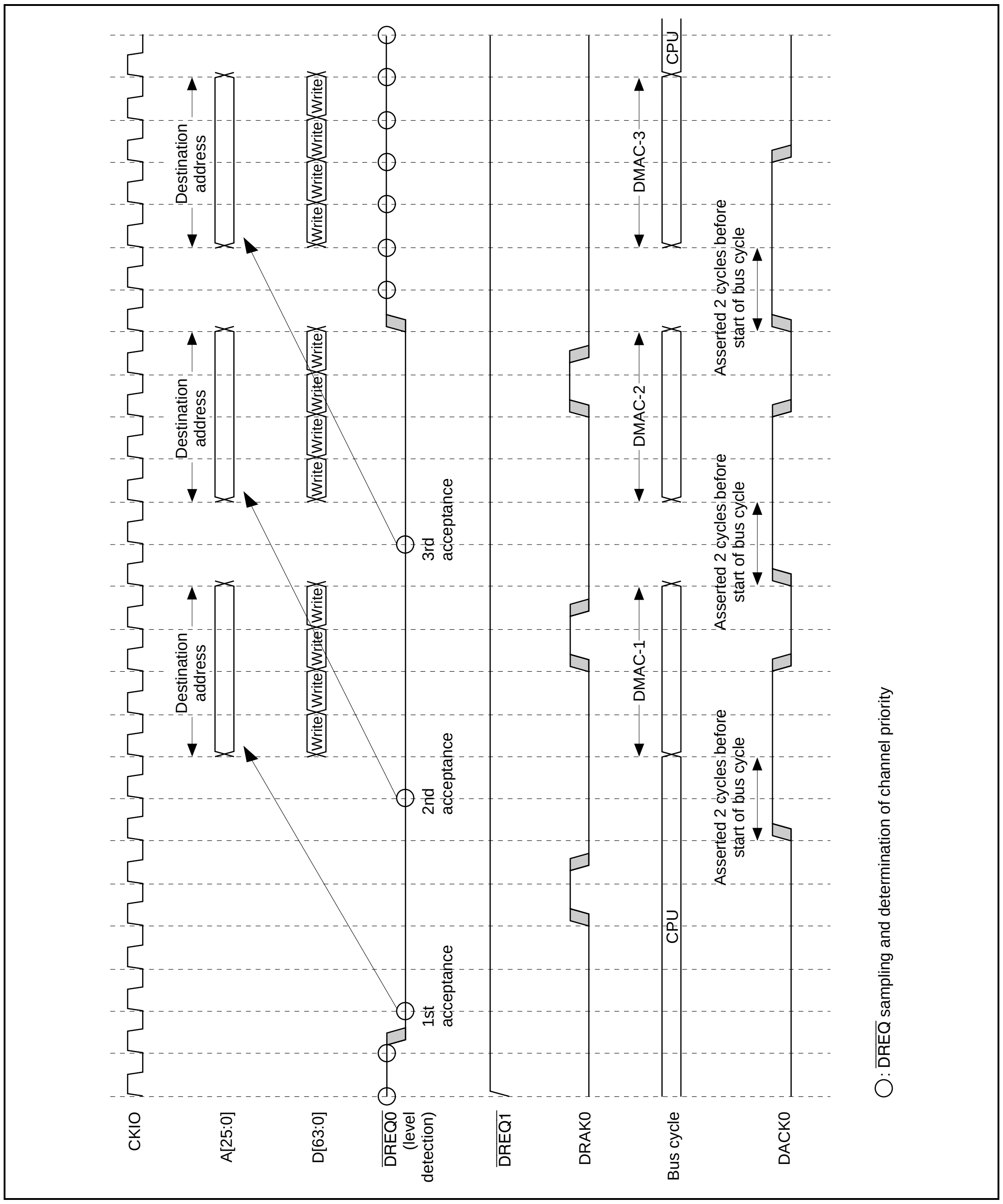


Figure 14.23 Single Address Mode/Burst Mode External Bus → External Bus/$\overline{\text{DREQ}}$ (Level Detection)/32-Byte Block Transfer (Bus Width: 64 Bits, SDRAM: Row Hit Write)

14.3.6 Ending DMA Transfer

The conditions for ending DMA transfer are different for ending on individual channels and for ending on all channels together. Except for the case where transfer ends when the value in the DMA transfer count register (DMATCR) reaches 0, the following conditions apply to ending transfer.

1. Cycle Steal Mode (External Request, On-Chip Peripheral Module Request, Auto-Request)

   When a transfer end condition is satisfied, acceptance of DMAC transfer requests is suspended. The DMAC completes transfer for the transfer requests accepted up to the point at which the transfer end condition was satisfied, then stops.

   In cycle steal mode, the operation is the same for both edge and level transfer request detection.
2. Burst Mode, Edge Detection (External Request, On-Chip Peripheral Module Request, Auto-Request)

   The delay between the point at which a transfer end condition is satisfied and the point at which the DMAC actually stops is the same as in cycle steal mode. In burst mode with edge detection, only the first transfer request activates the DMAC, but the timing of stop request (DE = 0 in CHCR, DME = 0 in DMAOR) sampling is the same as the transfer request sampling timing shown in 4 and 5 under Operation in section 14.3.5. Therefore, a transfer request is regarded as having been issued until a stop request is detected, and the corresponding processing is executed before the DMAC stops.
3. Burst Mode, Level Detection (External Request)

   The delay between the point at which a transfer end condition is satisfied and the point at which the DMAC actually stops is the same as in cycle steal mode. As in the case of burst mode with edge detection, the timing of stop request (DE = 0 in CHCR, DME = 0 in DMAOR) sampling is the same as the transfer request sampling timing shown in 2 and 3 under Operation in section 14.3.5. Therefore, a transfer request is regarded as having been issued until a stop request is detected, and the corresponding processing is executed before the DMAC stops.
4. Transfer Suspension Bus Timing

   Transfer suspension is executed on completion of processing for one transfer unit. In dual address mode transfer, write cycle processing is executed even if a transfer end condition is satisfied during the read cycle, and the transfers covered in 1, 2, and 3 above are also executed before operation is suspended.

Conditions for Ending Transfer on Individual Channels: Transfer ends on the corresponding channel when either of the following conditions is satisfied:

- The value in the DMA transfer count register (DMATCR) reaches 0.
- The DE bit in the DMA channel control register (CHCR) is cleared to 0.

1. End of transfer when DMATCR = 0

   When the DMATCR value reaches 0, DMA transfer ends on the corresponding channel and the transfer end flag (TE) in CHCR is set. If the interrupt enable bit (IE) is set at this time, an interrupt (DMTE) request is sent to the CPU.

   Transfer ending when DMATCR = 0 does not follow the procedures described in 1, 2, 3, and 4 in section 14.3.6.
2. End of transfer when DE = 0 in CHCR

   When the DMA enable bit (DE) in CHCR is cleared, DMA transfer is suspended on the corresponding channel. The TE bit is not set in this case. Transfer ending in this case follows the procedures described in 1, 2, 3, and 4 in section 14.3.6.

Conditions for Ending Transfer Simultaneously on All Channels: Transfer ends on all channels simultaneously when either of the following conditions is satisfied:

- The address error bit (AE) or NMI flag (NMIF) in the DMA operation register (DMAOR) is set.
- The DMA master enable bit (DME) in DMAOR is cleared to 0.

1. End of transfer when AE = 1 in DMAOR

   If the AE bit in DMAOR is set to 1 due to an address error, DMA transfer is suspended on all channels in accordance with the conditions in 1, 2, 3, and 4 in section 14.3.6, and the bus is passed to the CPU. Therefore, when AE is set to 1, the values in the DMA source address register (SAR), DMA destination address register (DAR), and DMA transfer count register (DMATCR) indicate the addresses for the DMA transfer to be performed next and the remaining number of transfers. The TE bit is not set in this case. Before resuming transfer, it is necessary to make a new setting for the channel that caused the address error, then write 0 to the AE bit after first reading 1 from it. Acceptance of external requests is suspended while AE is set to 1, so a DMA transfer request must be reissued when resuming transfer. Acceptance of internal requests is also suspended, so when resuming transfer, the DMA transfer request enable bit for the relevant on-chip peripheral module must be cleared to 0 before the new setting is made.

2. End of transfer when NMIF = 1 in DMAOR

   If the NMIF bit in DMAOR is set to 1 due to an NMI interrupt, DMA transfer is suspended on all channels in accordance with the conditions in 1, 2, 3, and 4 in section 14.3.6, and the bus is passed to the CPU. Therefore, when NMIF is set to 1, the values in the DMA source address register (SAR), DMA destination address register (DAR), and DMA transfer count register (DMATCR) indicate the addresses for the DMA transfer to be performed next and the remaining number of transfers. The TE bit is not set in this case. Before resuming transfer after NMI interrupt handling is completed, 0 must be written to the NMIF bit after first reading 1 from it. As in the case of AE being set to 1, acceptance of external requests is suspended while NMIF is set to 1, so a DMA transfer request must be reissued when resuming transfer. Acceptance of internal requests is also suspended, so when resuming transfer, the DMA transfer request enable bit for the relevant on-chip peripheral module must be cleared to 0 before the new setting is made.

3. End of transfer when DME = 0 in DMAOR

   If the DME bit in DMAOR is cleared to 0, DMA transfer is suspended on all channels in accordance with the conditions in 1, 2, 3, and 4 in section 14.3.6, and the bus is passed to the CPU. The TE bit is not set in this case. When DME is cleared to 0, the values in the DMA source address register (SAR), DMA destination address register (DAR), and DMA transfer count register (DMATCR) indicate the addresses for the DMA transfer to be performed next and the remaining number of transfers. When resuming transfer, DME must be set to 1. Operation will then be resumed from the next transfer.

14.4 Examples of Use

14.4.1 Examples of Transfer between External Memory and an External Device with DACK

Examples of transfer of data in external memory to an external device with DACK using DMAC channel 1 are considered here.

Table 14.8 shows the transfer conditions and the corresponding register settings.

Table 14.8 Conditions for Transfer between External Memory and an External Device with DACK, and Corresponding Register Settings

Transfer Conditions	Register	Set Value
Transfer source: external memory	SAR1	H'0C000000
Transfer source: external device with DACK	DAR1	(Accessed by DACK)
Number of transfers: 32	DMATCR1	H'00000020
Transfer source address: decremented	CHCR1	H'000022A5
Transfer destination address: (setting invalid)		
Transfer request source: external pin ($\overline{\text{DREQ1}}$) edge detection		
Bus mode: burst		
Transfer unit: word		
No interrupt request at end of transfer		
Channel priority order: 2 > 0 > 1 > 3	DMAOR	H'00000201

14.5 On-Demand Data Transfer Mode

14.5.1 Operation

Setting the DDT bit to 1 in DMAOR causes a transition to on-demand data transfer mode (DDT mode). In DDT mode, it is possible to specify direct single address mode transfer to channel 0 via the data bus and DDT module, and simultaneously issue a transfer request, using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, and ID [1:0] signals between an external device and the DMAC. Figure 14.24 shows a block diagram of the DMAC, DDT, BU, and an external device (with $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, and ID [1:0] pins).

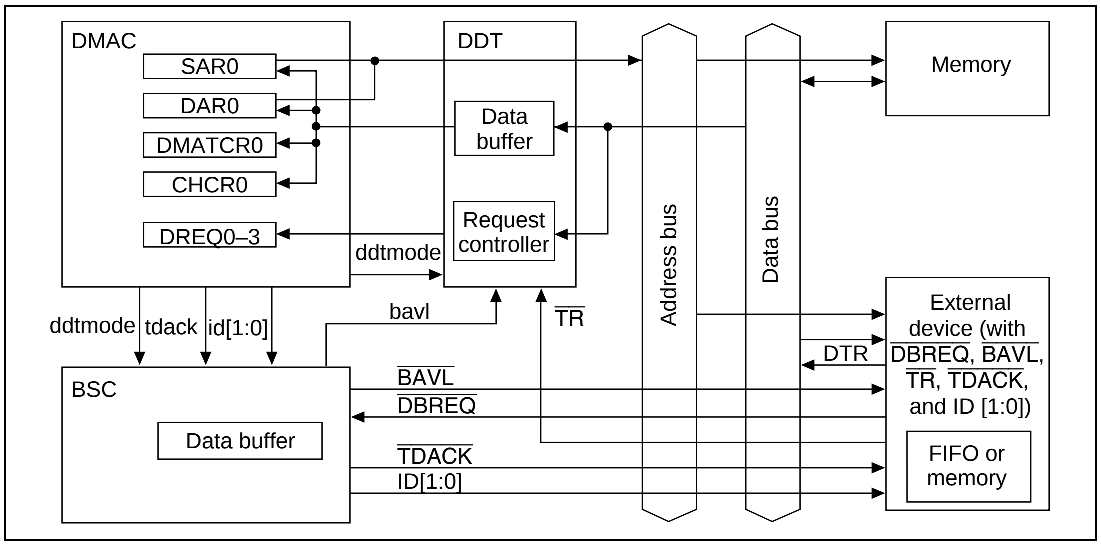


Figure 14.24 On-Demand Transfer Mode Block Diagram

For channels 1 to 3, after making the settings for normal DMA transfer using the CPU, a transfer request can be issued from an external device using the $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, and ID [1:0] signals (handshake protocol using the data bus). A transfer request can also be issued simply by asserting $\overline{\text{TR}}$, without using the external bus (handshake protocol without use of the data bus). For channel 2, after making the DMA transfer settings in the normal way, a transfer request can be issued directly from an external device (with $\overline{\text{DBREQ}}$, $\overline{\text{BAVL}}$, $\overline{\text{TR}}$, $\overline{\text{TDACK}}$, and ID [1:0] pins) by asserting $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$ simultaneously .

In DDT mode, there is a choice of five modes for performing DMA transfer.

1. Normal data transfer mode (channel 0)

   $\overline{\text{BAVL}}$ (the data bus available signal) is asserted in response to $\overline{\text{DBREQ}}$ (the data bus request signal) from an external device. Two CKIO-synchronous cycles after $\overline{\text{BAVL}}$ is asserted, the external data bus drives the data transfer setting command (DTR command) in synchronization with $\overline{\text{TR}}$ (the transfer request signal). The initial settings are then made in the DMAC channel 0 control register, and the DMA transfer is processed.

2. Normal data transfer mode (except channel 0)

   In this mode, the data transfer settings are made in the DMAC from the CPU, and DMA transfer requests only are performed from the external device.

   As in 1 above, $\overline{\text{DBREQ}}$ is asserted from the external device and the external bus is secured, then the DTR command is driven.

   The transfer request channel can be specified by means of the two ID bits in the DTR command.

3. Handshake protocol using the data bus (valid for channel 0 only)

   This mode is only valid for channel 0.

   After the initial settings have been made in the DMAC channel 0 control register, the DDT module asserts a data transfer request for the DMAC by setting the DTR command ID = 00 and MD = 00, and driving the DTR command.

4. Handshake protocol without use of the data bus

   The DDT module includes a function for recording the previously asserted request channel. By using this function, it is possible to assert a transfer request for the channel for which a request was asserted immediately before, by asserting $\overline{\text{TR}}$ only from an external device after a transfer request has once been made to the channel for which an initial setting has been made in the DMAC control register (DTR command and data transfer setting by the CPU in the DMAC).

5. Direct data transfer mode (valid for channel 2 only)

   A data transfer request can be asserted for channel 2 by asserting $\overline{\text{DREQ}}$ and $\overline{\text{TR}}$ simultaneously from an external device after the initial settings have been made in the DMAC channel 2 control register.

Note: For details of the DTR format setting procedure, see Appendix G, SH7750 On-Demand Data Transfer Mode.

14.5.2 Notes on Use of DDT Module

1. The handshake protocol without use of the data bus is always used, except in the case where $\overline{\text{TR}}$ is asserted two cycles after $\overline{\text{BAVL}}$ is asserted (and excluding requests to channel 2 by means of simultaneous assertion of $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$).
2. If a request to channel 2 is asserted by simultaneous assertion of $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$ during execution with the handshake protocol without use of the data bus, it is accepted if there is space in the channel 2 request queue.
3. With the handshake protocol without use of the data bus, a DMA transfer request can be asserted again for the channel for which transfer was requested immediately before by asserting $\overline{\text{TR}}$ only.
4. When channel 0 is operated using the handshake protocol without use of the data bus, MD ≠ 00 should always be transferred as initialization data*. Operation is not guaranteed if the handshake protocol is executed without transferring initialization data.

Note: * Initialization data: MD ≠ 00, ID = 00, SZ, R/W, COUNT, ADDRESS.

5. If only $\overline{\text{TR}}$ is asserted when operating other than with the handshake protocol without use of the data bus, this is ignored by the DDT module (which does not operate).
6. Operation is not guaranteed if the handshake protocol using the data bus is executed for channel 0 without transferring initialization data. (A request only is asserted for the DMAC.)
7. The DDT module is provided with four request queues for each of channels 1 to 3. If a request from an external device is asserted when these request queues are full, it will be ignored. (Channel 0 has a request flag; requests asserted while this flag is set are ignored.)
8. The DDT module uses the following procedure to process ID, MD, and SZ:

   When ID = 00

   a. MD = 00: ID, MD select (handshake with data bus)

   b. MD ≠ 00, SZ = 111: DMAC (CHCR0 DE bit) setting (transfer end request)

   c. MD ≠ 00: ADDRESS, COUNT, MD, RW, SZ, ID select (data transfer to DMAC)

   When ID ≠ 00

   a. Request to channels 1–3 (items other than ID ignored)
9. A data transfer end request (ID = 00, MD ≠ 00, SZ = 111) is not accepted when the channel 0 request flag in the DDT module is set (is not accepted during the bus cycle). Therefore, if the DTR command initialization data settings are ID = 00 and MD = 01 (edge sensing and burst transfer), transfer cannot be halted midway. (Set MD to a value other than 01.)
10. The handshake protocol using the data bus applies only to channel 0 (MD = 00).
11. Except when DTR.ID = 00, data other than DTR.ID is ignored.
12. A channel 0 DMA transfer halt request can be implemented by settings of DTR.ID = 00, DTR.MD ≠ 00, and DTR.SZ = 111. Values set in DMAC control registers, etc., are retained. DMAC register reads are possible, but an execution restart from an external device is not possible.

13. If a request is asserted for a channel other than channel 0 during execution with the handshake protocol using the data bus, and settings of DTR.ID = 00 and DTR.MD = 00 are sent by an external device with the handshake protocol using the data bus after DMA transfer has been executed on that channel, a request to channel 0 is asserted. (Initialization data need not be set when continuing in this way.)
14. $\overline{\text{DBREQ}}$ is already used as a bus arbitration signal, but when a request to channel 2 is asserted by means of simultaneous assertion of $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$, $\overline{\text{DBREQ}}$ is not interpreted as a bus arbitration signal (i.e., $\overline{\text{BAVL}}$ is not asserted by this signal).
15. It takes one cycle for $\overline{\text{DBREQ}}$ to be accepted by the DDT module after being asserted by an external device, but if $\overline{\text{BAVL}}$ is asserted from the BSC at this time, $\overline{\text{BAVL}}$ is not asserted since the $\overline{\text{DBREQ}}$ assertion by the external device is not reported to the BSC.
16. When settings of ID = 00, MD = 10, and SZ = 110 are transferred to the DDT module, the DDT channel 0 request flag and channel 1 to 3 request queues are cleared. (If a transfer request to a particular channel is followed by another request to the same channel while the TE bit in CHCR remains set to 1, request queue clearance is necessary since the DMAC is halted.)
17. When $\overline{\text{TR}}$ only is asserted in the handshake protocol using the data bus while the channel 0 TE flag is set after the end of the last DMA transfer, the TE flag must be cleared.

    If a transfer request is sent by asserting $\overline{\text{TR}}$ only for channel 0 when the channel 0 TE flag is set, the DMAC will freeze. In this case, the flag can be cleared as described in 16 above.
18. After $\overline{\text{DBREQ}}$ is asserted, do not assert $\overline{\text{DBREQ}}$ again until $\overline{\text{BAVL}}$ is asserted, as this will result in a discrepancy between the number of $\overline{\text{DBREQ}}$ and $\overline{\text{BAVL}}$ assertions.
19. Check that DMA transfer is not in progress before modifying the DDT bit in DMAOR. If DMAOR.DDT is cleared to 0 during DMA transfer in DDT mode, the DMAC will freeze. In this case, the flag can be cleared as described in 16 above.

14.6 Usage Notes

1. When modifying SAR0–SAR3, DAR0–DAR3, DMATCR0–DMATCR3, and CHCR0–CHCR3, first clear the DE bit for the relevant channel to 0.
2. The NMIF bit in DMAOR is set when an NMI interrupt is input even if the DMAC is not operating.

   Confirmation method when DMA transfer is not executed correctly:

   Read the NMIF, AE, and DME bits in DMAOR, the DE and TE bits in CHCR0–CHCR3, and DMATCR0–DMATCR3. If NMIF was set before the transfer, the DMATCR transfer count will remain at the set value. If NMIF was set during the transfer, when the DE bit is 1 and the TE bit is 0 in CHCR0–CHCR3, the DMATCR value will indicate the remaining number of transfers.

   Also, the next addresses to be accessed can be found by reading SAR0–SAR3 and DAR0–DAR3. If the AE bit has been set, an address error has occurred. Check the set values in CHCR, SAR, and DAR.
3. Check that DMA transfer is not in progress before making a transition to the module standby state, standby mode, or deep sleep mode.

   Either check that TE = 1 in CHCR0–CHCR3, or clear DME to 0 in DMAOR to terminate DMA transfer. When DME is cleared to 0 in DMAOR, transfer halts at the end of the currently executing DMA bus cycle. Note, therefore, that transfer may not end immediately, depending on the transfer data size. DMA operation is not guaranteed if the module standby state, standby mode, or deep sleep mode is entered without confirming that DMA transfer has ended.
4. Do not specify a DMAC, CCN, BIST, BSC, or UBC control register as the DMAC transfer source or destination.
5. When activating the DMAC, make the SAR, DAR, and DMATCR register settings for the relevant channel before setting DE to 1 in CHCR, or make the register settings with DE cleared to 0 in CHCR, then set DE to 1. It does not matter whether setting of the DME bit to 1 in DMAOR is carried out first or last. To operate the relevant channel, DME and DE must both be set to 1. The DMAC may not operate normally if the SAR, DAR, and DMATCR settings are not made (with the exception of the unused register in single address mode).
6. After the DMATCR count reaches 0 and DMA transfer ends normally, always write 0 to DMATCR even when executing the maximum number of transfers on the same channel.
7. When falling edge detection is used for external requests, keep the external request pin high when making DMAC settings.
8. When using the DMAC in single address mode, set an external address as the address. All channels will halt due to an address error if an on-chip peripheral module address is set.

Section 15 Serial Communication Interface (SCI)

15.1 Overview

The SH7750 is equipped with a single-channel serial communication interface (SCI) and a single-channel serial communication interface with built-in FIFO registers (SCI with FIFO: SCIF).

The SCI can handle both asynchronous and synchronous serial communication. A function is also provided for serial communication between processors (multiprocessor communication function).

The SCI supports a smart card interface conforming to ISO/IEC 7816-3 (Identification Card) as a serial communication interface function for IC card interface use. For details, see section 17, Smart Card Interface.

The SCIF is a dedicated asynchronous communication serial interface with built-in 16-stage FIFO registers for both transmission and reception. For details, see section 16, Serial Communication Interface with FIFO.

15.1.1 Features

SCI features are listed below.

- Choice of synchronous or asynchronous serial communication mode
  - Asynchronous mode

    Serial data communication is executed using an asynchronous system in which synchronization is achieved character by character. Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA). A multiprocessor communication function is also provided that enables serial data communication with a number of processors.

    There is a choice of 12 serial data transfer formats.

Data length:	7 or 8 bits
Stop bit length:	1 or 2 bits
Parity:	Even/odd/none
Multiprocessor bit:	1 or 0
Receive error detection:	Parity, overrun, and framing errors
Break detection:	A break can be detected by reading the RxD pin level directly from the serial port register (SCSPTR1) when a framing error occurs.

— Synchronous mode

  Serial data communication is synchronized with a clock. Serial data communication can be carried out with other chips that have a synchronous communication function.

  There is a single serial data transfer format.

Data length:	8 bits
Receive error detection:	Overrun errors

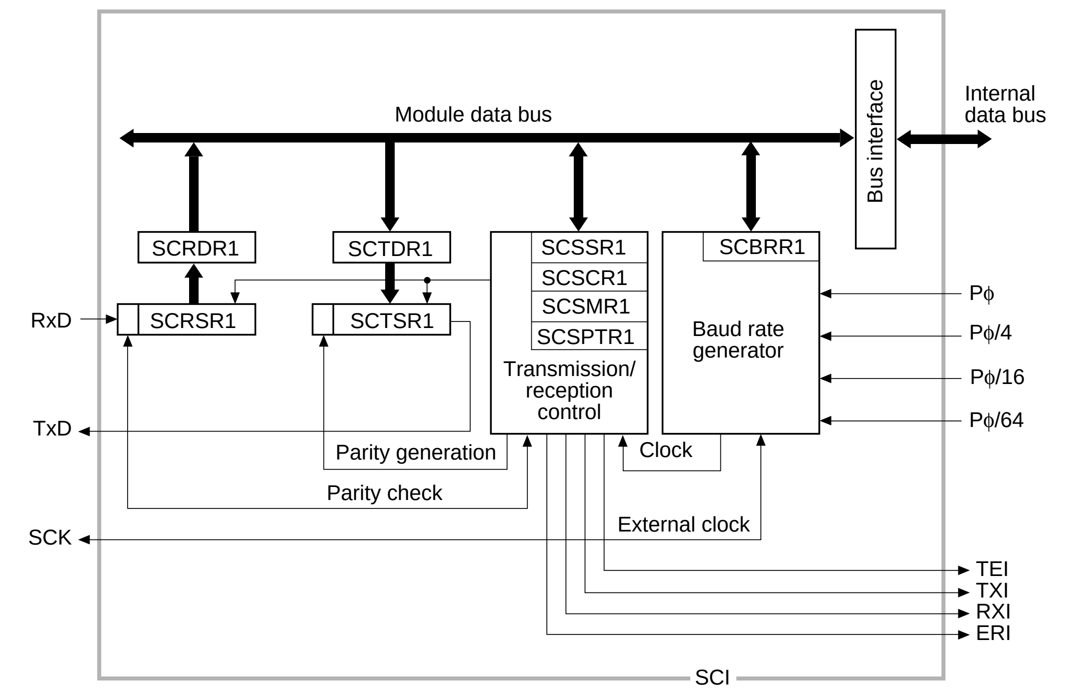
- Full-duplex communication capability

  The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously. Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data.
- On-chip baud rate generator allows any bit rate to be selected.
- Choice of serial clock source: internal clock from baud rate generator or external clock from SCK pin
- Four interrupt sources

  There are four interrupt sources—transmit-data-empty, transmit-end, receive-data-full, and receive-error—that can issue requests independently. The transmit-data-empty interrupt and receive-data-full interrupt can activate the DMA controller (DMAC) to execute a data transfer.
- When not in use, the SCI can be stopped by halting its clock supply to reduce power consumption.

15.1.2 Block Diagram

Figure 15.1 shows a block diagram of the SCI.



SCRSR1:	Receive shift register
SCRDR1:	Receive data register
SCTSR1:	Transmit shift register
SCTDR1:	Transmit data register
SCSMR1:	Serial mode register
SCSCR1:	Serial control register
SCSSR1:	Serial status register
SCBRR1:	Bit rate register
SCSPTR1:	Serial port register

Figure 15.1 Block Diagram of SCI

15.1.3 Pin Configuration

Table 15.1 shows the SCI pin configuration.

Table 15.1 SCI Pins

Pin Name	Abbreviation	I/O	Function
Serial clock pin	MD0/SCK	I/O	Clock input/output
Receive data pin	RxD	Input	Receive data input
Transmit data pin	MD7/TxD	Output	Transmit data output

Note: The serial clock pin and transmit data pin function as mode input pins MD0 and MD7 after a power-on reset. They are made to function as serial pins by performing SCI operation settings with the TE, RE, CKEI, and CKE0 bits in SCSCR1 and the C/$\overline{\text{A}}$ bit in SCSMR1. Break state transmission and detection, can be set in the SCI's SCSPTR1 register.

15.1.4 Register Configuration

The SCI has the internal registers shown in table 15.2. These registers are used to specify asynchronous mode or synchronous mode, the data format, and the bit rate, and to perform transmitter/receiver control.

With the exception of the serial port register, the SCI registers are initialized in standby mode and in the module standby state as well as after a power-on reset or manual reset. When recovering from standby mode or the module standby state, the registers must be set again.

Table 15.2 SCI Registers

Name	Abbreviation	R/W	Initial Value	P4 Address	Area 7 Address	Access Size
Serial mode register	SCSMR1	R/W	H'00	H'FFE00000	H'1FE00000	8
Bit rate register	SCBRR1	R/W	H'FF	H'FFE00004	H'1FE00004	8
Serial control register	SCSCR1	R/W	H'00	H'FFE00008	H'1FE00008	8
Transmit data register	SCTDR1	R/W	H'FF	H'FFE0000C	H'1FE0000C	8
Serial status register	SCSSR1	R/(W)*[1]	H'84	H'FFE00010	H'1FE00010	8
Receive data register	SCRDR1	R	H'00	H'FFE00014	H'1FE00014	8
Serial port register	SCSPTR1	R/W	H'00*[2]	H'FFE0001C	H'1FE0001C	8

Notes: 1. Only 0 can be written, to clear flags.

2. The value of bits 2 and 0 is undefined.

15.2 Register Descriptions

15.2.1 Receive Shift Register (SCRSR1)

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

SCRSR1 is the register used to receive serial data.

The SCI sets serial data input from the RxD pin in SCRSR1 in the order received, starting with the LSB (bit 0), and converts it to parallel data. When one byte of data has been received, it is transferred to SCRDR1 automatically.

SCRSR1 cannot be directly read or written to by the CPU.

15.2.2 Receive Data Register (SCRDR1)

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

SCRDR1 is the register that stores received serial data.

When the SCI has received one byte of serial data, it transfers the received data from SCRSR1 to SCRDR1 where it is stored, and completes the receive operation. SCRSR1 is then enabled for reception.

Since SCRSR1 and SCRDR1 function as a double buffer in this way, it is possible to receive data continuously.

SCRDR1 is a read-only register, and cannot be written to by the CPU.

SCRDR1 is initialized to H'00 by a power-on reset or manual reset, in standby mode, and in the module standby state.

15.2.3 Transmit Shift Register (SCTSR1)

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

SCTSR1 is the register used to transmit serial data.

To perform serial data transmission, the SCI first transfers transmit data from SCTDR1 to SCTSR1, then sends the data to the TxD pin starting with the LSB (bit 0).

When transmission of one byte is completed, the next transmit data is transferred from SCTDR1 to SCTSR1, and transmission started, automatically. However, data transfer from SCTDR1 to SCTSR1 is not performed if the TDRE flag in the serial status register (SCSSR1) is set to 1.

SCTSR1 cannot be directly read or written to by the CPU.

15.2.4 Transmit Data Register (SCTDR1)

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

SCTDR1 is an 8-bit register that stores data for serial transmission.

When the SCI detects that SCTSR1 is empty, it transfers the transmit data written in SCTDR1 to SCTSR1 and starts serial transmission. Continuous serial transmission can be carried out by writing the next transmit data to SCTDR1 during serial transmission of the data in SCTSR1.

SCTDR1 can be read or written to by the CPU at all times.

SCTDR1 is initialized to H'FF by a power-on reset or manual reset, in standby mode, and in the module standby state.

15.2.5 Serial Mode Register (SCSMR1)

Bit:	7	6	5	4	3	2	1	0
	C/$\overline{\text{A}}$	CHR	PE	O/$\overline{\text{E}}$	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

SCSMR1 is an 8-bit register used to set the SCI's serial transfer format and select the baud rate generator clock source.

SCSMR1 can be read or written to by the CPU at all times.

SCSMR1 is initialized to H'00 by a power-on reset or manual reset, in standby mode, and in the module standby state.

Bit 7—Communication Mode (C/$\overline{\text{A}}$): Selects asynchronous mode or synchronous mode as the SCI operating mode.

Bit 7: C/$\overline{\text{A}}$	Description	
0	Asynchronous mode	(Initial value)
1	Synchronous mode	

Bit 6—Character Length (CHR): Selects 7 or 8 bits as the data length in asynchronous mode. In synchronous mode, a fixed data length of 8 bits is used regardless of the CHR setting,

Bit 6: CHR	Description	
0	8-bit data	(Initial value)
1	7-bit data*	

Note: * When 7-bit data is selected, the MSB (bit 7) of SCTDR1 is not transmitted.

Bit 5—Parity Enable (PE): In asynchronous mode, selects whether or not parity bit addition is performed in transmission, and parity bit checking in reception. In synchronous mode, parity bit addition and checking is not performed, regardless of the PE bit setting.

Bit 5: PE	Description	
0	Parity bit addition and checking disabled	(Initial value)
1	Parity bit addition and checking enabled*	

Note: * When the PE bit is set to 1, the parity (even or odd) specified by the O/$\overline{\text{E}}$ bit is added to transmit data before transmission. In reception, the parity bit is checked for the parity (even or odd) specified by the O/$\overline{\text{E}}$ bit.

Bit 4—Parity Mode (O/$\overline{\text{E}}$): Selects either even or odd parity for use in parity addition and checking. The O/$\overline{\text{E}}$ bit setting is only valid when the PE bit is set to 1, enabling parity bit addition and checking, in asynchronous mode. The O/$\overline{\text{E}}$ bit setting is invalid in synchronous mode, and when parity addition and checking is disabled in asynchronous mode.

Bit 4: O/$\overline{\text{E}}$	Description	
0	Even parity*[1]	(Initial value)
1	Odd parity*[2]	

Notes: 1. When even parity is set, parity bit addition is performed in transmission so that the total number of 1-bits in the transmit character plus the parity bit is even. In reception, a check is performed to see if the total number of 1-bits in the receive character plus the parity bit is even.

2. When odd parity is set, parity bit addition is performed in transmission so that the total number of 1-bits in the transmit character plus the parity bit is odd. In reception, a check is performed to see if the total number of 1-bits in the receive character plus the parity bit is odd.

Bit 3—Stop Bit Length (STOP): Selects 1 or 2 bits as the stop bit length in asynchronous mode. The STOP bit setting is only valid in asynchronous mode. If synchronous mode is set, the STOP bit setting is invalid since stop bits are not added.

Bit 3: STOP	Description	
0	1 stop bit*[1]	(Initial value)
1	2 stop bits*[2]	

Notes: 1. In transmission, a single 1-bit (stop bit) is added to the end of a transmit character before it is sent.

2. In transmission, two 1-bits (stop bits) are added to the end of a transmit character before it is sent.

In reception, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, it is treated as a stop bit; if it is 0, it is treated as the start bit of the next transmit character.

Bit 2—Multiprocessor Mode (MP): Selects a multiprocessor format. When a multiprocessor format is selected, the PE bit and O/$\overline{\text{E}}$ bit parity settings are invalid. The MP bit setting is only valid in asynchronous mode; it is invalid in synchronous mode.

For details of the multiprocessor communication function, see section 15.3.3, Multiprocessor Communication Function.

Bit 2: MP	Description	
0	Multiprocessor function disabled	(Initial value)
1	Multiprocessor format selected	

Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0): These bits select the clock source for the on-chip baud rate generator. The clock source can be selected from Pϕ, Pϕ/4, Pϕ/16, and Pϕ/64, according to the setting of bits CKS1 and CKS0.

For the relation between the clock source, the bit rate register setting, and the baud rate, see section 15.2.9, Bit Rate Register (SCBRR1).

Bit 1: CKS1	Bit 0: CKS0	Description	
0	0	Pϕ clock	(Initial value)
	1	Pϕ/4 clock	
1	0	Pϕ/16 clock	
	1	Pϕ/64 clock	

Note: Pϕ: Peripheral clock

15.2.6 Serial Control Register (SCSCR1)

Bit:	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The SCSCR1 register performs enabling or disabling of SCI transfer operations, serial clock output in asynchronous mode, and interrupt requests, and selection of the serial clock source.

SCSCR1 can be read or written to by the CPU at all times.

SCSCR1 is initialized to H'00 by a power-on reset or manual reset, in standby mode, and in the module standby state.

Bit 7—Transmit Interrupt Enable (TIE): Enables or disables transmit-data-empty interrupt (TXI) request generation when serial transmit data is transferred from SCTDR1 to SCTSR1 and the TDRE flag in SCSSR1 is set to 1.

Bit 7: TIE	Description
0	Transmit-data-empty interrupt (TXI) request disabled* (Initial value)
1	Transmit-data-empty interrupt (TXI) request enabled

Note: * TXI interrupt requests can be cleared by reading 1 from the TDRE flag, then clearing it to 0, or by clearing the TIE bit to 0.

Bit 6—Receive Interrupt Enable (RIE): Enables or disables receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request generation when serial receive data is transferred from SCRSR1 to SCRDR1 and the RDRF flag in SCSSR1 is set to 1.

Bit 6: RIE	Description
0	Receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request disabled* (Initial value)
1	Receive-data-full interrupt (RXI) request and receive-error interrupt (ERI) request enabled

Note: * RXI and ERI interrupt requests can be cleared by reading 1 from the RDRF flag, or the FER, PER, or ORER flag, then clearing the flag to 0, or by clearing the RIE bit to 0.

Bit 5—Transmit Enable (TE): Enables or disables the start of serial transmission by the SCI.

Bit 5: TE	Description	
0	Transmission disabled*[1]	(Initial value)
1	Transmission enabled*[2]	

Notes: 1. The TDRE flag in SCSSR1 is fixed at 1.

2. In this state, serial transmission is started when transmit data is written to SCTDR1 and the TDRE flag in SCSSR1 is cleared to 0.

SCSMR1 setting must be performed to decide the transmit format before setting the TE bit to 1.

Bit 4—Receive Enable (RE): Enables or disables the start of serial reception by the SCI.

Bit 4: RE	Description	
0	Reception disabled*[1]	(Initial value)
1	Reception enabled*[2]	

Notes: 1. Clearing the RE bit to 0 does not affect the RDRF, FER, PER, and ORER flags, which retain their states.

2. Serial reception is started in this state when a start bit is detected in asynchronous mode or serial clock input is detected in synchronous mode.

SCSMR1 setting must be performed to decide the receive format before setting the RE bit to 1.

Bit 3—Multiprocessor Interrupt Enable (MPIE): Enables or disables multiprocessor interrupts. The MPIE bit setting is only valid in asynchronous mode when the MP bit in SCSMR1 is set to 1.

The MPIE bit setting is invalid in synchronous mode or when the MP bit is cleared to 0.

Bit 3: MPIE	Description	
0	Multiprocessor interrupts disabled (normal reception performed) [Clearing conditions] • When the MPIE bit is cleared to 0 • When data with MPB = 1 is received	(Initial value)
1	Multiprocessor interrupts enabled* Receive interrupt (RXI) requests, receive-error interrupt (ERI) requests, and setting of the RDRF, FER, and ORER flags in SCSSR1 are disabled until data with the multiprocessor bit set to 1 is received.	

Note: * Receive data transfer from SCRSR1 to SCRDR1, receive error detection, and setting of the RDRF, FER, and ORER flags in SCSSR1, is not performed. When receive data including MPB = 1 is received, the MPB bit in SCSSR1 is set to 1, the MPIE bit is cleared to 0 automatically, and generation of RXI and ERI interrupts (when the TIE and RIE bits in SCSCR1 are set to 1) and FER and ORER flag setting is enabled.

Bit 2—Transmit-End interrupt Enable (TEIE): Enables or disables transmit-end interrupt (TEI) request generation when there is no valid transmit data in SCTDR1 at the time for MSB data transmission.

Bit 2: TEIE	Description	
0	Transmit-end interrupt (TEI) request disabled*	(Initial value)
1	Transmit-end interrupt (TEI) request enabled*	

Note: * TEI interrupt requests can be cleared by reading 1 from the TDRE flag in SCSSR1, then clearing it to 0 and clearing the TEND flag to 0, or by clearing the TEIE bit to 0.

Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0): These bits are used to select the SCI clock source and enable or disable clock output from the SCK pin. The combination of the CKE1 and CKE0 bits determines whether the SCK pin functions as the serial clock output pin or the serial clock input pin.

The setting of the CKE0 bit, however, is only valid for internal clock operation (CKE1 = 0) in asynchronous mode. The CKE0 bit setting is invalid in synchronous mode and in the case of external clock operation (CKE1 = 1). The CKE1 and CKE0 bits must be set before determining the SCI's operating mode with SCSMR1.

For details of clock source selection, see table 15.9 in section 15.3, Operation.

Bit 1: CKE1	Bit 0: CKE0	Description	
0	0	Asynchronous mode	Internal clock/SCK pin functions as input pin (input signal ignored)*1
		Synchronous mode	Internal clock/SCK pin functions as serial clock output*1
	1	Asynchronous mode	Internal clock/SCK pin functions as clock output*2
		Synchronous mode	Internal clock/SCK pin functions as serial clock output
1	0	Asynchronous mode	External clock/SCK pin functions as clock input*3
		Synchronous mode	External clock/SCK pin functions as serial clock input
	1	Asynchronous mode	External clock/SCK pin functions as clock input*3
		Synchronous mode	External clock/SCK pin functions as serial clock input

Notes: 1. Initial value
2. Outputs a clock of the same frequency as the bit rate.
3. Inputs a clock with a frequency 16 times the bit rate.

15.2.7 Serial Status Register (SCSSR1)

Bit:	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial value:	1	0	0	0	0	1	0	0
R/W:	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

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

SCSSR1 is an 8-bit register containing status flags that indicate the operating status of the SCI, and multiprocessor bits.

SCSSR1 can be read or written to by the CPU at all times. However, 1 cannot be written to flags TDRE, RDRF, ORER, PER, and FER. Also note that in order to clear these flags they must be read as 1 beforehand. The TEND flag and MPB flag are read-only flags and cannot be modified.

SCSSR1 is initialized to H'84 by a power-on reset or manual reset, in standby mode, and in the module standby state.

Bit 7—Transmit Data Register Empty (TDRE): Indicates that data has been transferred from SCTDR1 to SCTSR1 and the next serial transmit data can be written to SCTDR1.

Bit 7: TDRE	Description
0	Valid transmit data has been written to SCTDR1 [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 • When data is written to SCTDR1 by the DMAC
1	There is no valid transmit data in SCTDR1 (Initial value) [Setting conditions] • Power-on reset, manual reset, standby mode, or module standby • When the TE bit in SCSCR1 is 0 • When data is transferred from SCTDR1 to SCTSR1 and data can be written to SCTDR1

Bit 6—Receive Data Register Full (RDRF): Indicates that the received data has been stored in SCRDR1.

Bit 6: RDRF	Description
0	There is no valid receive data in SCRDR1 (Initial value) [Clearing conditions] • Power-on reset, manual reset, standby mode, or module standby • When 0 is written to RDRF after reading RDRF = 1 • When data in SCRDR1 is read by the DMAC
1	There is valid receive data in SCRDR1 [Setting condition] When serial reception ends normally and receive data is transferred from SCRSR1 to SCRDR1

Note: SCRDR1 and the RDRF flag are not affected and retain their previous values when an error is detected during reception or when the RE bit in SCSCR1 is cleared to 0.
If reception of the next data is completed while the RDRF flag is still set to 1, an overrun error will occur and the receive data will be lost.

Bit 5—Overrun Error (ORER): Indicates that an overrun error occurred during reception, causing abnormal termination.

Bit 5: ORER	Description
0	Reception in progress, or reception has ended normally*[1] (Initial value) [Clearing conditions] • Power-on reset, manual reset, standby mode, or module standby • When 0 is written to ORER after reading ORER = 1
1	An overrun error occurred during reception*[2] [Setting condition] When the next serial reception is completed while RDRF = 1

Notes: 1. The ORER flag is not affected and retains its previous state when the RE bit in SCSCR1 is cleared to 0.
2. The receive data prior to the overrun error is retained in SCRDR1, and the data received subsequently is lost. Serial reception cannot be continued while the ORER flag is set to 1. In synchronous mode, serial transmission cannot be continued either.

Bit 4—Framing Error (FER): Indicates that a framing error occurred during reception in asynchronous mode, causing abnormal termination.

Bit 4: FER	Description
0	Reception in progress, or reception has ended normally*[1] (Initial value) [Clearing conditions] • Power-on reset, manual reset, standby mode, or module standby • When 0 is written to FER after reading FER = 1
1	A framing error occurred during reception [Setting condition] When the SCI checks whether the stop bit at the end of the receive data is 1 when reception ends, and the stop bit is 0*[2]

Notes: 1. The FER flag is not affected and retains its previous state when the RE bit in SCSCR1 is cleared to 0.
2. In 2-stop-bit mode, only the first stop bit is checked for a value of 1; the second stop bit is not checked. If a framing error occurs, the receive data is transferred to SCRDR1 but the RDRF flag is not set. Serial reception cannot be continued while the FER flag is set to 1.

Bit 3—Parity Error (PER): Indicates that a parity error occurred during reception with parity addition in asynchronous mode, causing abnormal termination.

Bit 3: PER	Description
0	Reception in progress, or reception has ended normally*[1] (Initial value) [Clearing conditions] • Power-on reset, manual reset, standby mode, or module standby • When 0 is written to PER after reading PER = 1
1	A parity error occurred during reception*[2] [Setting condition] When, in reception, the number of 1-bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the O/$\overline{E}$ bit in SCSMR1

Notes: 1. The PER flag is not affected and retains its previous state when the RE bit in SCSCR1 is cleared to 0.
2. If a parity error occurs, the receive data is transferred to SCRDR1 but the RDRF flag is not set. Serial reception cannot be continued while the PER flag is set to 1.

Bit 2—Transmit End (TEND): Indicates that there is no valid data in SCTDR1 when the last bit of the transmit character is sent, and transmission has been ended.

The TEND flag is read-only and cannot be modified.

Bit 2: TEND	Description	
0	Transmission is in progress [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 • When data is written to SCTDR1 by the DMAC	
1	Transmission has been ended [Setting conditions] • Power-on reset, manual reset, standby mode, or module standby • When the TE bit in SCSCR1 is 0 • When TDRE = 1 on transmission of the last bit of a 1-byte serial transmit character	(Initial value)

Bit 1—Multiprocessor Bit (MPB): When reception is performed using a multiprocessor format in asynchronous mode, MPB stores the multiprocessor bit in the receive data.

The MPB flag is read-only and cannot be modified.

Bit 1: MPB	Description	
0	Data with a 0 multiprocessor bit has been received*	(Initial value)
1	Data with a 1 multiprocessor bit has been received	

Note: * Retains its previous state when the RE bit in SCSCR1 is cleared to 0 while using a multiprocessor format.

Bit 0—Multiprocessor Bit Transfer (MPBT): When transmission is performed using a multiprocessor format in asynchronous mode, MPBT stores the multiprocessor bit to be added to the transmit data.

The MPBT bit setting is invalid in synchronous mode, when a multiprocessor format is not used, and when the operation is not transmission.

Unlike transmit data, the MPBT bit is not double-buffered, so it is necessary to check whether transmission has been completed before changing its value.

Bit 0: MPBT	Description	
0	Data with a 0 multiprocessor bit is transmitted	(Initial value)
1	Data with a 1 multiprocessor bit is transmitted	

15.2.8 Serial Port Register (SCSPTR1)

Bit:	7	6	5	4	3	2	1	0
	EIO	—	—	—	SPB1IO	SPB1DT	SPB0IO	SPB0DT
Initial value:	0	0	0	0	0	—	0	—
R/W:	R/W	—	—	—	R/W	R/W	R/W	R/W

SCSPTR1 is an 8-bit readable/writable register that controls input/output and data for the port pins multiplexed with the serial communication interface (SCI) pins. Input data can be read from the RxD pin, output data written to the TxD pin, and breaks in serial transmission/reception controlled, by means of bits 1 and 0. SCK pin data reading and output data writing can be performed by means of bits 3 and 2. Bit 7 controls enabling and disabling of the RXI interrupt.

SCSPTR1 can be read or written to by the CPU at all times. All SCSPTR1 bits except bits 2 and 0 are initialized to 0 by a power-on reset or manual reset; the value of bits 2 and 0 is undefined. SCSPTR1 is not initialized in the module standby state or standby mode.

Bit 7—Error Interrupt Only (EIO): When the EIO bit is 1, an RXI interrupt request is not sent to the CPU even if the RIE bit is set to 1. When the DMAC is used, this setting means that only ERI interrupts are handled by the CPU. The DMAC transfers read data to memory or another peripheral module. This bit specifies enabling or disabling of the RXI interrupt.

Bit 7: EIO	Description
0	The RIE bit enables/disables RXI and ERI interrupts When the RIE bit is 1, RXI and ERI interrupts are sent to INTC(Initial value)
1	When the RIE bit is 1, only ERI interrupts are sent to INTC

Bits 6 to 4—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 3—Serial Port Clock Port I/O (SPB1IO): Specifies serial port SCK pin input/output. When the SCK pin is actually set as a port output pin and outputs the value set by the SPB1DT bit, the C/$\overline{\text{A}}$ bit in SCSMR1 and the CKE1 and CKE0 bits in SCSCR1 should be cleared to 0.

Bit 3: SPB1IO	Description	
0	SPB1DT bit value is not output to the SCK pin	(Initial value)
1	SPB1DT bit value is output to the SCK pin	

Bit 2—Serial Port Clock Port Data (SPB1DT): Specifies the serial port SCK pin input/output data. Input or output is specified by the SPB1IO bit (see the description of bit 3, SPB1IO, for details). When output is specified, the value of the SPB1DT bit is output to the SCK pin. The SCK pin value is read from the SPB1DT bit regardless of the value of the SPB1IO bit. The initial value of this bit after a power-on or manual reset is undefined.

Bit 2: SPB1DT	Description
0	Input/output data is low-level
1	Input/output data is high-level

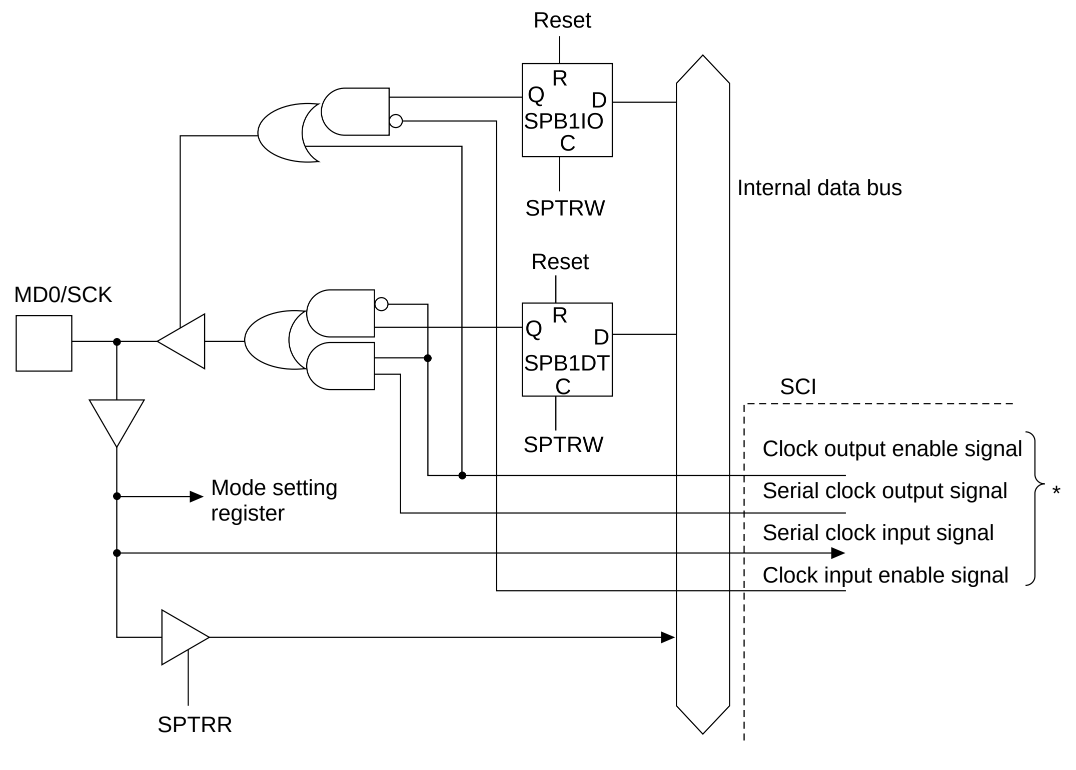
Bit 1—Serial Port Break I/O (SPB0IO): Specifies the serial port TxD pin output condition. When the TxD pin is actually set as a port output pin and outputs the value set by the SPB0DT bit, the TE bit in SCSCR1 should be cleared to 0.

Bit 1: SPB0IO	Description	
0	SPB0DT bit value is not output to the TxD pin	(Initial value)
1	SPB0DT bit value is output to the TxD pin	

Bit 0—Serial Port Break Data (SPB0DT): Specifies the serial port RxD pin input data and TxD pin output data. The TxD pin output condition is specified by the SPB0IO bit (see the description of bit 1, SPB0IO, for details). When the TxD pin is designated as an output, the value of the SPB0DT bit is output to the TxD pin. The RxD pin value is read from the SPB0DT bit regardless of the value of the SPB0IO bit. The initial value of this bit after a power-on or manual reset is undefined.

Bit 0: SPB0DT	Description
0	Input/output data is low-level
1	Input/output data is high-level

SCI I/O port block diagrams are shown in figures 15.2 to 15.4.


Reset
R
Q
D
SPB1IO
C
SPTRW
Internal data bus
Reset
MD0/SCK
R
Q
D
SPB1DT
C
SPTRW
SCI
Clock output enable signal
Serial clock output signal
Serial clock input signal
Clock input enable signal
*
Mode setting register
SPTRR

SPTRW: Write to SPTR
SPTRR: Read SPTR

Note: * Signals that set the SCK pin function as internal clock output or external clock input according to the CKE0 and CKE1 bits in SCSCR1 and the C/$\overline{A}$ bit in SCSMR1.

Figure 15.2 MD0/SCK Pin

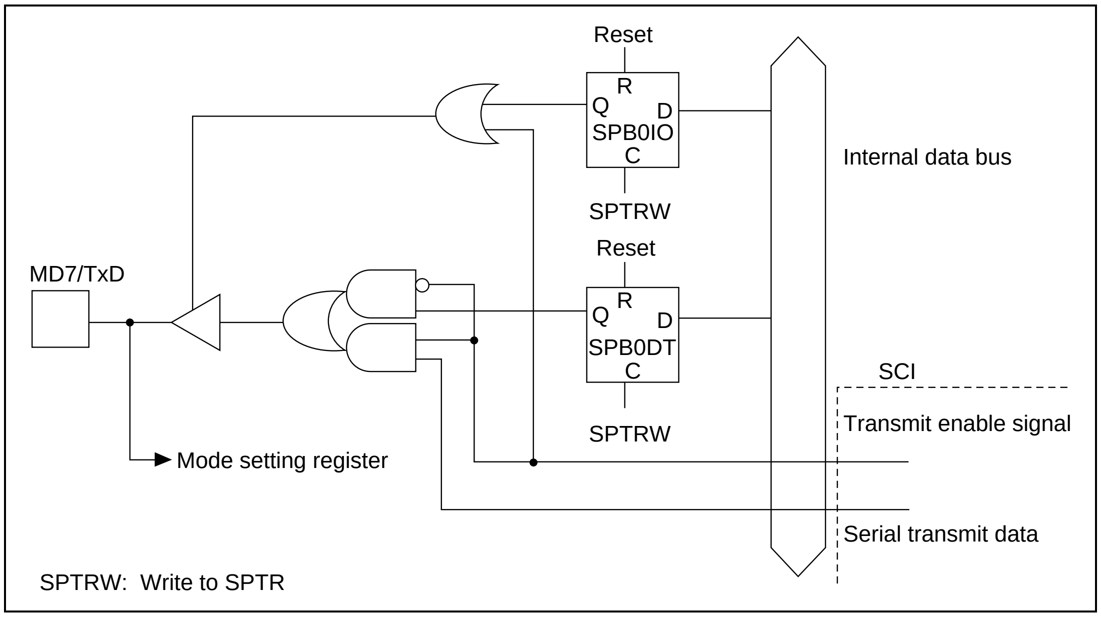


Figure 15.3 MD7/TxD Pin

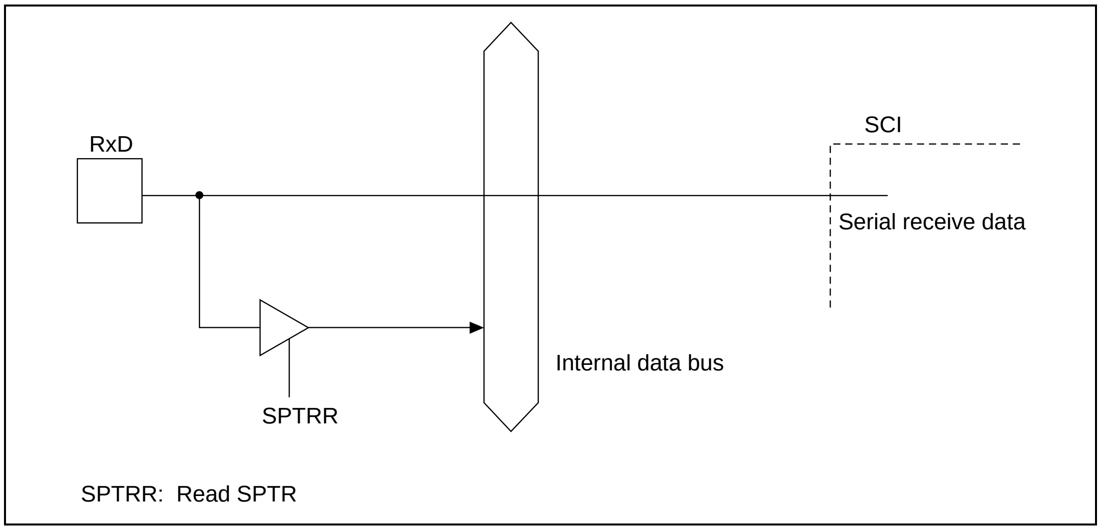


Figure 15.4 RxD Pin

15.2.9 Bit Rate Register (SCBRR1)

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

SCBRR1 is an 8-bit register that sets the serial transfer bit rate in accordance with the baud rate generator operating clock selected by bits CKS1 and CKS0 in SCSMR1.

SCBRR1 can be read or written to by the CPU at all times.

SCBRR1 is initialized to H'FF by a power-on reset or manual reset, in standby mode, and in the module standby state.

The SCBRR1 setting is found from the following equations.

Asynchronous mode:

$$N = \frac{P\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Synchronous mode:

$$N = \frac{P\phi}{8 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Where B: Bit rate (bits/s)
N: SCBRR1 setting for baud rate generator ($0 \le N \le 255$)
Pϕ: Peripheral module operating frequency (MHz)
n: Baud rate generator input clock (n = 0 to 3)
(See the table below for the relation between n and the clock.)

		SCSMR1 Setting	
n	**Clock**	**CKS1**	**CKS0**
0	Pϕ	0	0
1	Pϕ/4	0	1
2	Pϕ/16	1	0
3	Pϕ/64	1	1

The bit rate error in asynchronous mode is found from the following equation:

$$\text{Error (\%)} = \left\{ \frac{P\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

Table 15.3 shows sample SCBRR1 settings in asynchronous mode, and table 15.4 shows sample SCBRR1 settings in synchronous mode.

Table 15.3 Examples of Bit Rates and SCBRR1 Settings in Asynchronous Mode

	Pϕ (MHz)											
	2			**2.097152**			**2.4576**			**3**		
Bit Rate (bits/s)	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**
110	1	141	0.03	1	148	–0.04	1	174	–0.26	1	212	0.03
150	1	103	0.16	1	108	0.21	1	127	0.00	1	155	0.16
300	0	207	0.16	0	217	0.21	0	255	0.00	1	77	0.16
600	0	103	0.16	0	108	0.21	0	127	0.00	0	155	0.16
1200	0	51	0.16	0	54	–0.70	0	63	0.00	0	77	0.16
2400	0	25	0.16	0	26	1.14	0	31	0.00	0	38	0.16
4800	0	12	0.16	0	13	–2.48	0	15	0.00	0	19	–2.34
9600	0	6	–6.99	0	6	–2.48	0	7	0.00	0	9	–2.34
19200	0	2	8.51	0	2	13.78	0	3	0.00	0	4	–2.34
31250	0	1	0.00	0	1	4.86	0	1	22.88	0	2	0.00
38400	0	1	–18.62	0	1	–14.67	0	1	0.00			

	Pϕ (MHz)											
	3.6864			**4**			**4.9152**			**5**		
Bit Rate (bits/s)	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**
110	2	64	0.70	2	70	0.03	2	86	0.31	2	88	–0.25
150	1	191	0.00	1	207	0.16	1	255	0.00	2	64	0.16
300	1	95	0.00	1	103	0.16	1	127	0.00	1	129	0.16
600	0	191	0.00	0	207	0.16	0	255	0.00	1	64	0.16
1200	0	95	0.00	0	103	0.16	0	127	0.00	0	129	0.16
2400	0	47	0.00	0	51	0.16	0	63	0.00	0	64	0.16
4800	0	23	0.00	0	25	0.16	0	31	0.00	0	32	–1.36
9600	0	11	0.00	0	12	0.16	0	15	0.00	0	15	1.73
19200	0	5	0.00	0	6	–6.99	0	7	0.00	0	7	1.73
31250	—	—	—	0	3	0.00	0	4	–1.70	0	4	0.00
38400	0	2	0.00	0	2	8.51	0	3	0.00	0	3	1.73

Legend

Blank: No setting is available.

—: A setting is available but error occurs.

Table 15.3 Examples of Bit Rates and SCBRR1 Settings in Asynchronous Mode (cont)

	Pϕ (MHz)											
	6			**6.144**			**7.37288**			**8**		
Bit Rate (bits/s)	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**	**n**	**N**	**Error (%)**
110	2	106	–0.44	2	108	0.08	2	130	–0.07	2	141	0.03
150	2	77	0.16	2	79	0.00	2	95	0.00	2	103	0.16
300	1	155	0.16	1	159	0.00	1	191	0.00	1	207	0.16
600	1	77	0.16	1	79	0.00	1	95	0.00	1	103	0.16
1200	0	155	0.16	0	159	0.00	0	191	0.00	0	207	0.16
2400	0	77	0.16	0	79	0.00	0	95	0.00	0	103	0.16
4800	0	38	0.16	0	39	0.00	0	47	0.00	0	51	0.16
9600	0	19	–2.34	0	19	0.00	0	23	0.00	0	25	0.16
19200	0	9	–2.34	0	9	0.00	0	11	0.00	0	12	0.16
31250	0	5	0.00	0	5	2.40	0	6	5.33	0	7	0.00
38400	0	4	–2.34	0	4	0.00	0	5	0.00	0	6	–6.99

	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	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

Table 15.3 Examples of Bit Rates and SCBRR1 Settings in Asynchronous Mode (cont)

	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	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

	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 15.4 Examples of Bit Rates and SCBRR1 Settings in Synchronous Mode

	Pϕ (MHz)									
	4		8		16		28.7		30	
Bit Rate (bits/s)	n	N	n	N	n	N	n	N	n	N
10	—	—	—	—	—	—	—	—	—	—
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: As far as possible, the setting should be made so that the error is within 1%.

Legend

Blank:	No setting is available.
—:	A setting is available but error occurs.
*	Continuous transmission/reception is not possible.

Table 15.5 shows the maximum bit rate for various frequencies in asynchronous mode. Tables 15.6 and 15.7 show the maximum bit rates with external clock input.

Table 15.5 Maximum Bit Rate 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 15.6 Maximum Bit Rate with 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 15.7 Maximum Bit Rate with External Clock Input (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

15.3 Operation

15.3.1 Overview

The SCI can carry out serial communication in two modes: asynchronous mode in which synchronization is achieved character by character, and synchronous mode in which synchronization is achieved with clock pulses.

Selection of asynchronous or synchronous mode and the transmission format is made using SCSMR1 as shown in table 15.8. The SCI clock source is determined by a combination of the $C/\overline{A}$ bit in SCSMR1 and the CKE1 and CKE0 bits in SCSCR1, as shown in table 15.9.

- Asynchronous mode
  - Data length: Choice of 7 or 8 bits
  - Choice of parity addition, multiprocessor bit addition, and addition of 1 or 2 stop bits (the combination of these parameters determines the transfer format and character length)
  - Detection of framing, parity, and overrun errors, and breaks, during reception
  - Choice of internal or external clock as SCI clock source

    When internal clock is selected: The SCI operates on the baud rate generator clock and a clock with the same frequency as the bit rate can be output.

    When external clock is selected: A clock with a frequency of 16 times the bit rate must be input (the on-chip baud rate generator is not used).

- Synchronous mode
  - Transfer format: Fixed 8-bit data
  - Detection of overrun errors during reception
  - Choice of internal or external clock as SCI clock source

    When internal clock is selected: The SCI operates on the baud rate generator clock and a serial clock is output off-chip.

    When external clock is selected: The on-chip baud rate generator is not used, and the SCI operates on the input serial clock.

Table 15.8 SCSMR1 Settings for Serial Transfer Format Selection

SCSMR1 Settings						SCI Transfer Format			
Bit 7: C/$\overline{A}$	**Bit 6: CHR**	**Bit 2: MP**	**Bit 5: PE**	**Bit 3: STOP**	**Mode**	**Data Length**	**Multi-processor Bit**	**Parity Bit**	**Stop Bit Length**
0	0	0	0	0	Asynchronous mode	8-bit data	No	No	1 bit
				1					2 bits
			1	0				Yes	1 bit
				1					2 bits
	1		0	0		7-bit data		No	1 bit
				1					2 bits
			1	0				Yes	1 bit
				1					2 bits
	0	1	*	0	Asynchronous mode (multiprocessor format)	8-bit data	Yes	No	1 bit
				1					2 bits
	1			0		7-bit data			1 bit
				1					2 bits
1	*	*	*	*	Synchronous mode	8-bit data	No		None

Note: An asterisk in the table means “Don’t care.”

Table 15.9 SCSMR1 and SCSCR1 Settings for SCI Clock Source Selection

SCSMR1	SCSCR1 Setting			SCI Transmit/Receive Clock	
Bit 7: C/$\overline{A}$	**Bit 1: CKE1**	**Bit 0: CKE0**	**Mode**	**Clock Source**	**SCK Pin Function**
0	0	0	Asynchronous mode	Internal	SCI does not use SCK pin
		1			Outputs clock with same frequency as bit rate
	1	0		External	Inputs clock with frequency of 16 times the bit rate
		1			
1	0	0	Synchronous mode	Internal	Outputs serial clock
		1			
	1	0		External	Inputs serial clock
		1			

15.3.2 Operation in Asynchronous Mode

In asynchronous mode, characters are sent or received, each preceded by a start bit indicating the start of communication and followed by one or two stop bits indicating the end of communication. Serial communication is thus carried out with synchronization established on a character-by-character basis.

Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 15.5 shows the general format for asynchronous serial communication.

In asynchronous serial communication, the transmission line is usually held in the mark state (high level). The SCI monitors the transmission line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication.

One serial communication character consists of a start bit (low level), followed by data (in LSB-first order), a parity bit (high or low level), and finally one or two stop bits (high level).

In asynchronous mode, the SCI performs synchronization at the falling edge of the start bit in reception. The SCI samples the data on the eighth pulse of a clock with a frequency of 16 times the length of one bit, so that the transfer data is latched at the center of each bit.

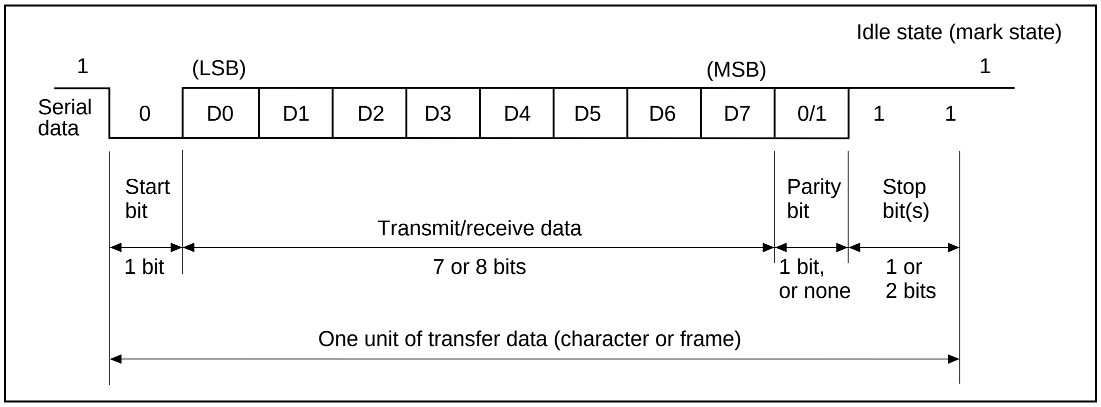


Figure 15.5 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)

Data Transfer Format

Table 15.10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected according to the SCSMR1 setting.

Table 15.10 Serial Transfer Formats (Asynchronous Mode)

SCSMR1 Settings				Serial Transfer Format and Frame Length											
CHR	**PE**	**MP**	**STOP**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
0	0	0	0	S	8-bit data								STOP		
0	0	0	1	S	8-bit data								STOP	STOP	
0	1	0	0	S	8-bit data								P	STOP	
0	1	0	1	S	8-bit data								P	STOP	STOP
1	0	0	0	S	7-bit data							STOP			
1	0	0	1	S	7-bit data							STOP	STOP		
1	1	0	0	S	7-bit data							P	STOP		
1	1	0	1	S	7-bit data							P	STOP	STOP	
0	*	1	0	S	8-bit data								MPB	STOP	
0	*	1	1	S	8-bit data								MPB	STOP	STOP
1	*	1	0	S	7-bit data							MPB	STOP		
1	*	1	1	S	7-bit data							MPB	STOP	STOP	

S:	Start bit
STOP:	Stop bit
P:	Parity bit
MPB:	Multiprocessor bit
Note:	An asterisk in the table means “Don’t care.”

Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input at the SCK pin can be selected as the SCI's serial clock, according to the setting of the $C/\overline{A}$ bit in SCSMR1 and the CKE1 and CKE0 bits in SCSCR1. For details of SCI clock source selection, see table 15.9.

When an external clock is input at the SCK pin, the clock frequency should be 16 times the bit rate used.

When the SCI is operated on an internal clock, the clock can be output from the SCK pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is at the center of each transmit data bit, as shown in figure 15.6.

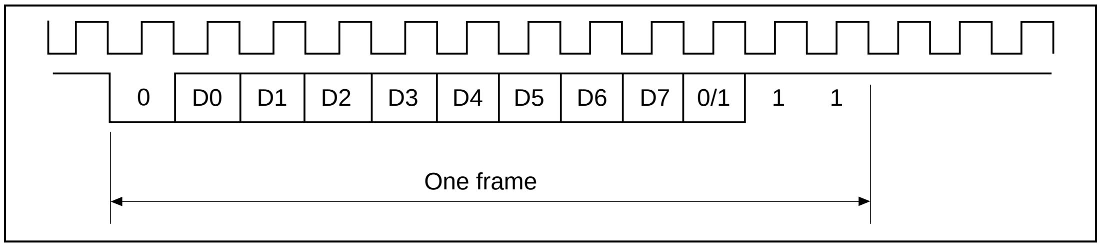


Figure 15.6 Relation between Output Clock and Transfer Data Phase (Asynchronous Mode)

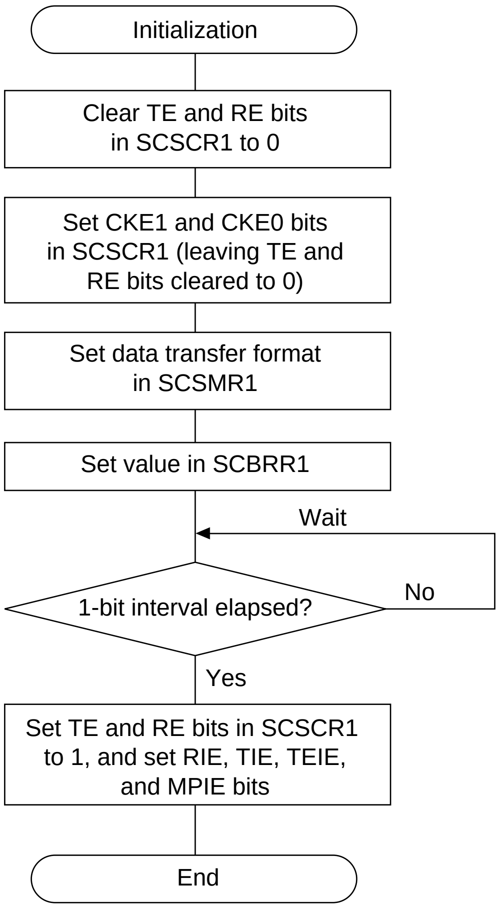
Data Transfer Operations

SCI Initialization (Asynchronous Mode): Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR1 to 0, then initialize the SCI as described below.

When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and SCTSR1 is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of SCRDR1.

When an external clock is used the clock should not be stopped during operation, including initialization, since operation will be unreliable in this case.

Figure 15.7 shows a sample SCI initialization flowchart.



Initialization
Clear TE and RE bits in SCSCR1 to 0
Set CKE1 and CKE0 bits in SCSCR1 (leaving TE and RE bits cleared to 0)
Set data transfer format in SCSMR1
Set value in SCBRR1
Wait
1-bit interval elapsed?
No
Yes
Set TE and RE bits in SCSCR1 to 1, and set RIE, TIE, TEIE, and MPIE bits
End

1. Set the clock selection in SCSCR1.

   Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.

   When clock output is selected in asynchronous mode, it is output immediately after SCSCR1 settings are made.

2. Set the data transfer format in SCSMR1.

3. Write a value corresponding to the bit rate into SCBRR1. (Not necessary if an external clock is used.)

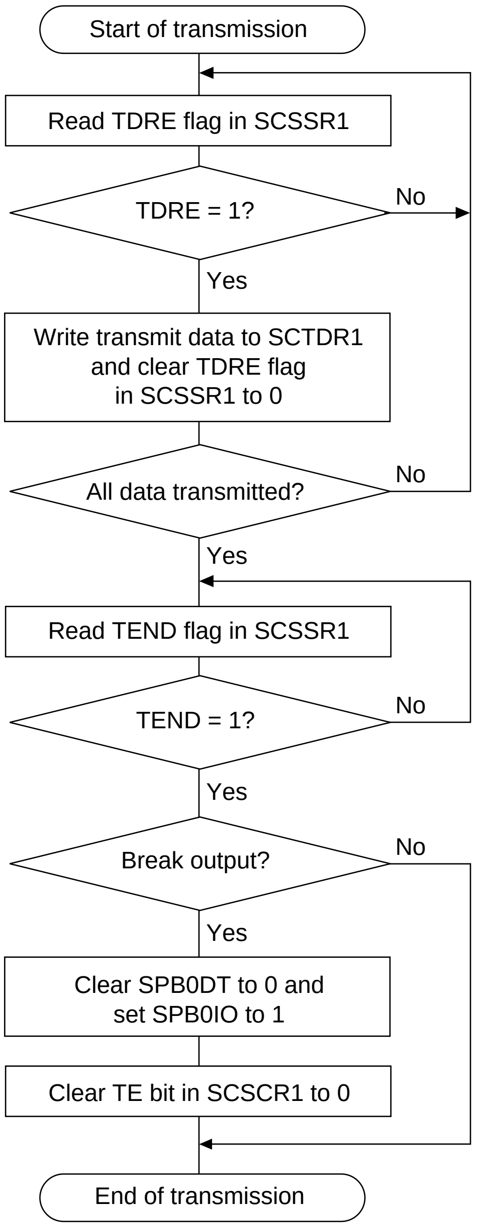
4. Wait at least one bit interval, then set the TE bit or RE bit in SCSCR1 to 1. Also set the RIE, TIE, TEIE, and MPIE bits.

   Setting the TE and RE bits enables the TxD and RxD pins to be used. When transmitting, the SCI will go to the mark state; when receiving, it will go to the idle state, waiting for a start bit.

Figure 15.7 Sample SCI Initialization Flowchart

Serial Data Transmission (Asynchronous Mode): Figure 15.8 shows a sample flowchart for serial transmission.

Use the following procedure for serial data transmission after enabling the SCI for transmission.



1. SCI status check and transmit data write: Read SCSSR1 and check that the TDRE flag is set to 1, then write transmit data to SCTDR1 and clear the TDRE flag to 0.
2. Serial transmission continuation procedure: To continue serial transmission, read 1 from the TDRE flag to confirm that writing is possible, then write data to SCTDR1, and then clear the TDRE flag to 0. (Checking and clearing of the TDRE flag is automatic when the direct memory access controller (DMAC) is activated by a transmit-data-empty interrupt (TXI) request, and data is written to SCTDR1.)
3. Break output at the end of serial transmission: To output a break in serial transmission, clear the SPB0DT bit to 0 and set the SPB0IO bit to 1 in SCSPTR, then clear the TE bit in SCSCR1 to 0.

Figure 15.8 Sample Serial Transmission Flowchart

In serial transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SCSSR1. When TDRE is cleared to 0, the SCI recognizes that data has been written to SCTDR1, and transfers the data from SCTDR1 to SCTSR1.
2. After transferring data from SCTDR1 to SCTSR1, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a transmit-data-empty interrupt (TXI) is generated.

   The serial transmit data is sent from the TxD pin in the following order.

   a. Start bit: One 0-bit is output.
   b. Transmit data: 8-bit or 7-bit data is output in LSB-first order.
   c. Parity bit or multiprocessor bit: One parity bit (even or odd parity), or one multiprocessor bit is output. (A format in which neither a parity bit nor a multiprocessor bit is output can also be selected.)
   d. Stop bit(s): One or two 1-bits (stop bits) are output.
   e. Mark state: 1 is output continuously until the start bit that starts the next transmission is sent.
3. The SCI checks the TDRE flag at the timing for sending the stop bit. If the TDRE flag is cleared to 0, data is transferred from SCTDR1 to SCTSR1, the stop bit is sent, and then serial transmission of the next frame is started.

   If the TDRE flag is set to 1, the TEND flag in SCSSR1 is set to 1, the stop bit is sent, and then the line goes to the mark state in which 1 is output continuously. If the TEIE bit in SCSCR1 is set to 1 at this time, a TEI interrupt request is generated.

Figure 15.9 shows an example of the operation for transmission in asynchronous mode.

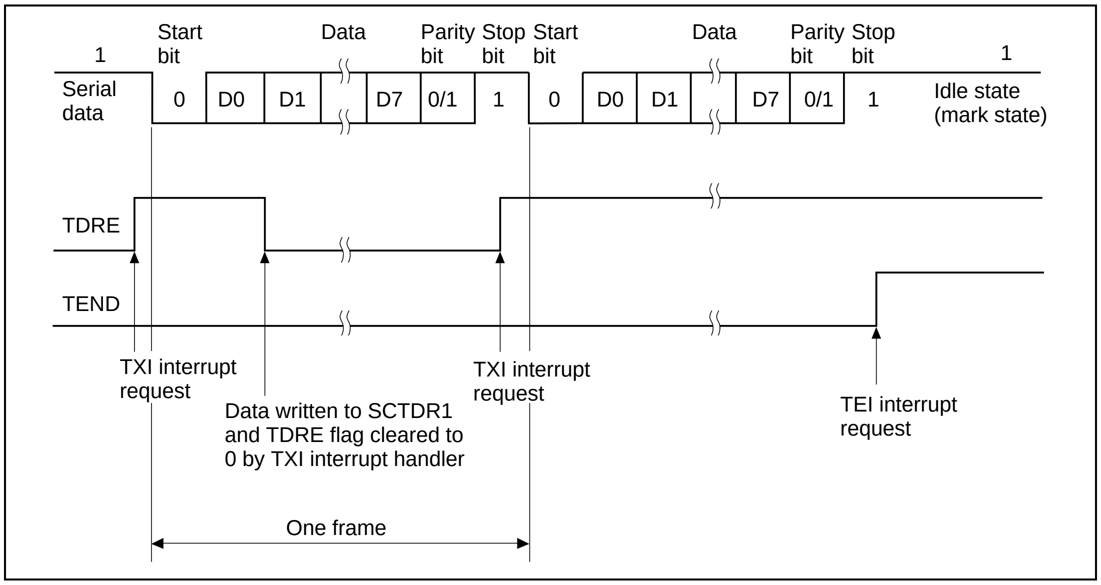
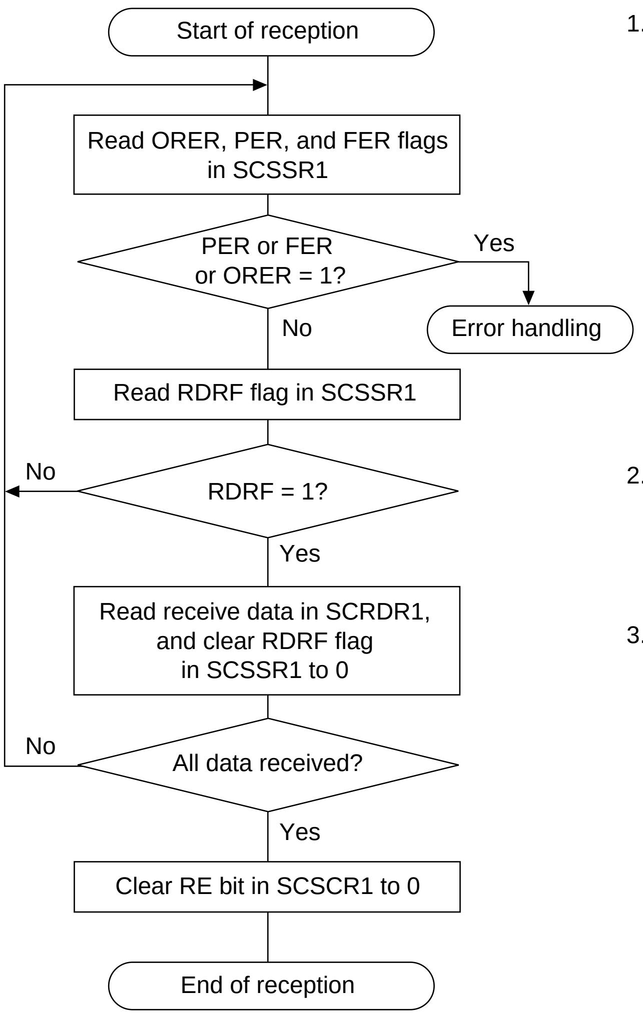


Figure 15.9 Example of Transmit Operation in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit)

Serial Data Reception (Asynchronous Mode): Figure 15.10 shows a sample flowchart for serial reception.

Use the following procedure for serial data reception after enabling the SCI for reception.



1. Receive error handling and break detection: If a receive error occurs, read the ORER, PER, and FER flags in SCSSR1 to identify the error. After performing the appropriate error handling, ensure that the ORER, PER, and FER flags are all cleared to 0. Reception cannot be resumed if any of these flags are set to 1. In the case of a framing error, a break can be detected by reading the value of the RxD pin.
2. SCI status check and receive data read : Read SCSSR1 and check that RDRF = 1, then read the receive data in SCRDR1 and clear the RDRF flag to 0.
3. Serial reception continuation procedure: To continue serial reception, complete zero-clearing of the RDRF flag before the stop bit for the current frame is received. (The RDRF flag is cleared automatically when the direct memory access controller (DMAC) is activated by an RXI interrupt and the SCRDR1 value is read.)

Figure 15.10 Sample Serial Reception Flowchart (1)

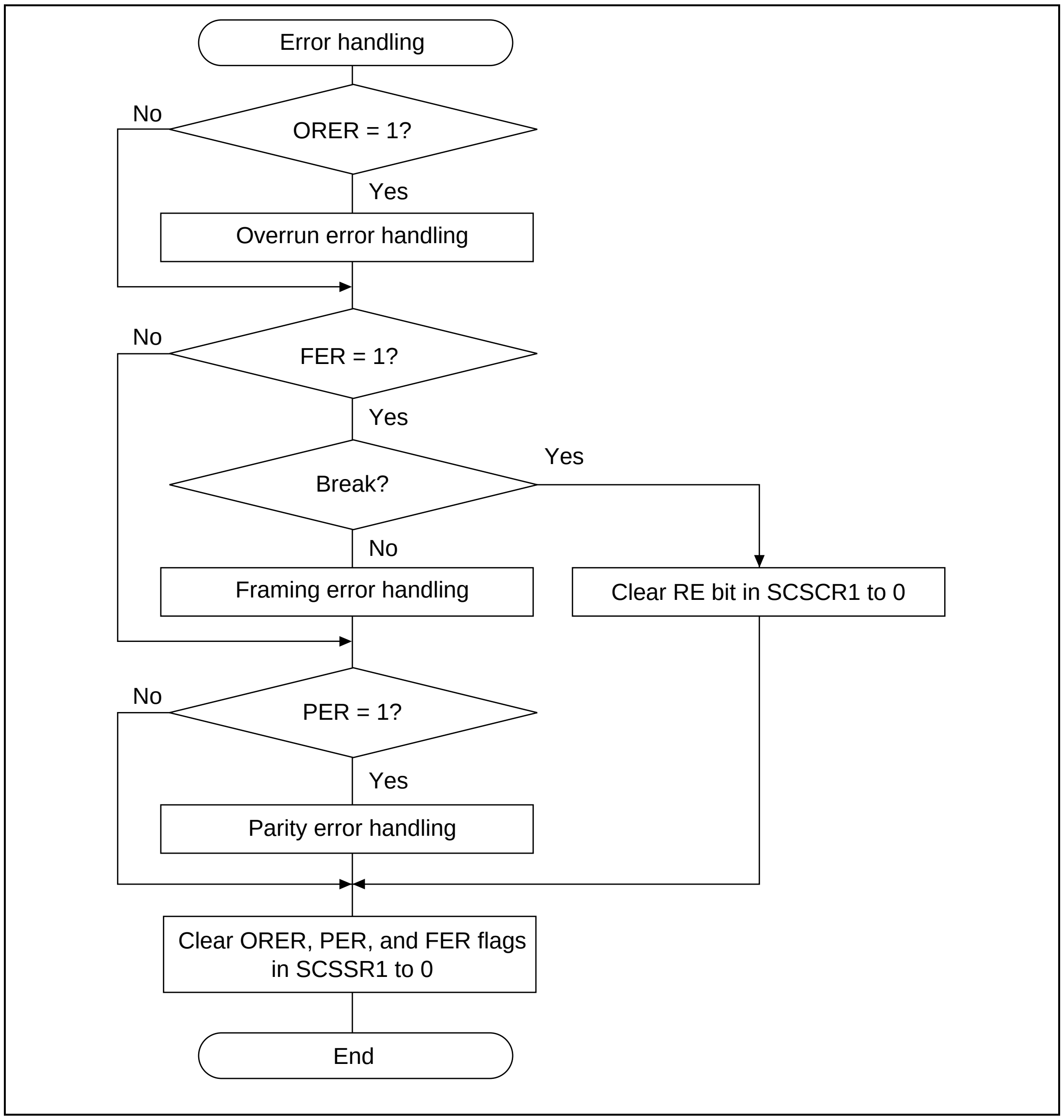


Figure 15.10 Sample Serial Reception Flowchart (2)

In serial reception, the SCI operates as described below.

1. The SCI 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 SCRSR1 in LSB-to-MSB order.
3. The parity bit and stop bit are received.

   After receiving these bits, the SCI carries out the following checks.

   a. Parity check: The SCI checks whether the number of 1-bits in the receive data agrees with the parity (even or odd) set in the O/E bit in SCSMR1.
   b. Stop bit check: The SCI checks whether the stop bit is 1. If there are two stop bits, only the first is checked.
   c. Status check: The SCI checks whether the RDRF flag is 0, indicating that the receive data can be transferred from SCRSR1 to SCRDR1.

   If all the above checks are passed, the RDRF flag is set to 1, and the receive data is stored in SCRDR1.

   If a receive error is detected in the error check, the operation is as shown in table 15.11.

Note: No further receive operations can be performed when a receive error has occurred. Also note that the RDRF flag is not set to 1 in reception, and so the error flags must be cleared to 0.

4. If the EIO bit in SCSPTR1 is cleared to 0 and the RIE bit in SCSCR1 is set to 1 when the RDRF flag changes to 1, a receive-data-full interrupt (RXI) request is generated.

   If the RIE bit in SCSCR1 is set to 1 when the ORER, PER, or FER flag changes to 1, a receive-error interrupt (ERI) request is generated. A receive-data-full request is always output to the DMAC when the RDRF flag changes to 1.

Table 15.11 Receive Error Conditions

Receive Error	Abbreviation	Condition	Data Transfer
Overrun error	ORER	Reception of next data is completed while RDRF flag in SCSSR1 is set to 1	Receive data is not transferred from SCRSR1 to SCRDR1
Framing error	FER	Stop bit is 0	Receive data is transferred from SCRSR1 to SCRDR1
Parity error	PER	Received data parity differs from that (even or odd) set in SCSMR1	Receive data is transferred from SCRSR1 to SCRDR1

Figure 15.11 shows an example of the operation for reception in asynchronous mode.

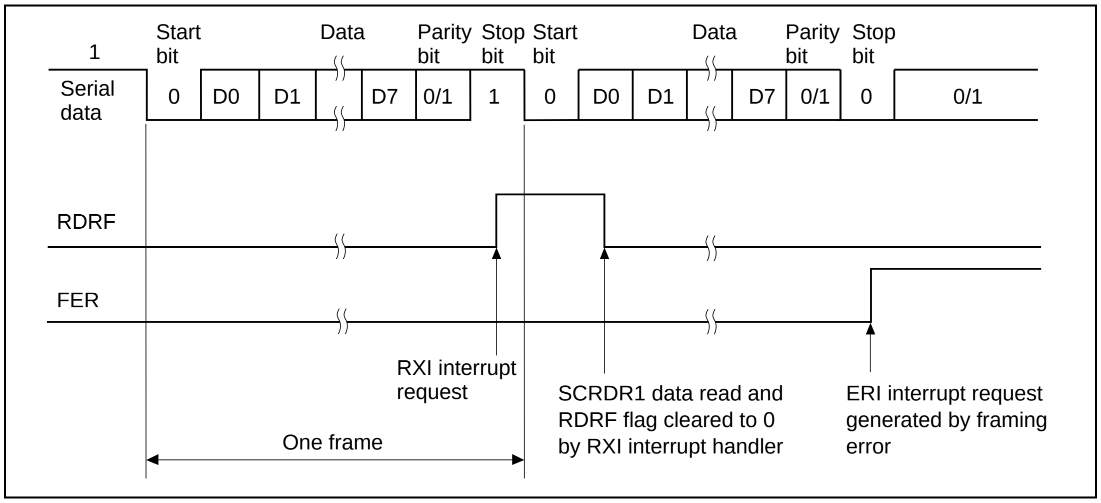


Figure 15.11 Example of SCI Receive Operation (Example with 8-Bit Data, Parity, One Stop Bit)

15.3.3 Multiprocessor Communication Function

The multiprocessor communication function performs serial communication using a multiprocessor format, in which a multiprocessor bit is added to the transfer data, in asynchronous mode. Use of this function enables data transfer to be performed among a number of processors sharing a serial transmission line.

When multiprocessor communication is carried out, each receiving station is addressed by a unique ID code.

The serial communication cycle consists of two cycles: an ID transmission cycle which specifies the receiving station , and a data transmission cycle. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle.

The transmitting station first sends the ID of the receiving station with which it wants to perform serial communication as data with a 1 multiprocessor bit added. It then sends transmit data as data with a 0 multiprocessor bit added.

The receiving station skips the data until data with a 1 multiprocessor bit is sent.

When data with a 1 multiprocessor bit is received, the receiving station compares that data with its own ID. The station whose ID matches then receives the data sent next. Stations whose ID does not match continue to skip the data until data with a 1 multiprocessor bit is again received. In this way, data communication is carried out among a number of processors.

Figure 15.12 shows an example of inter-processor communication using a multiprocessor format.

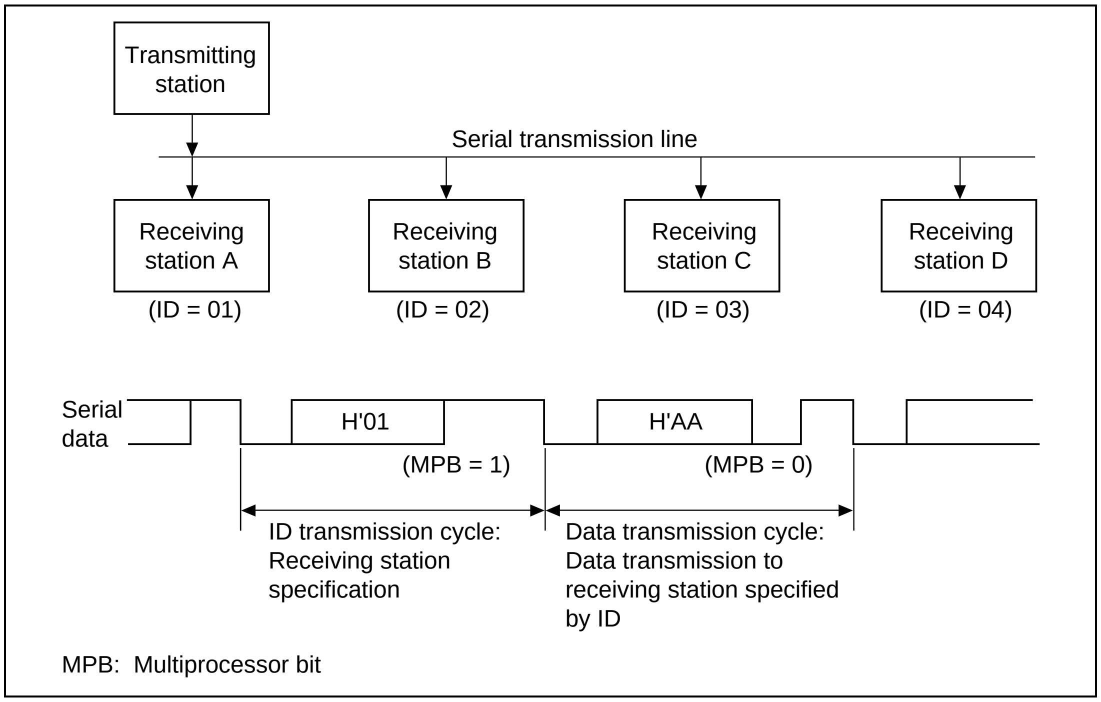


Figure 15.12 Example of Inter-Processor Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)

Data Transfer Formats

There are four data transfer formats. When the multiprocessor format is specified, the parity bit specification is invalid. For details, see table 15.10.

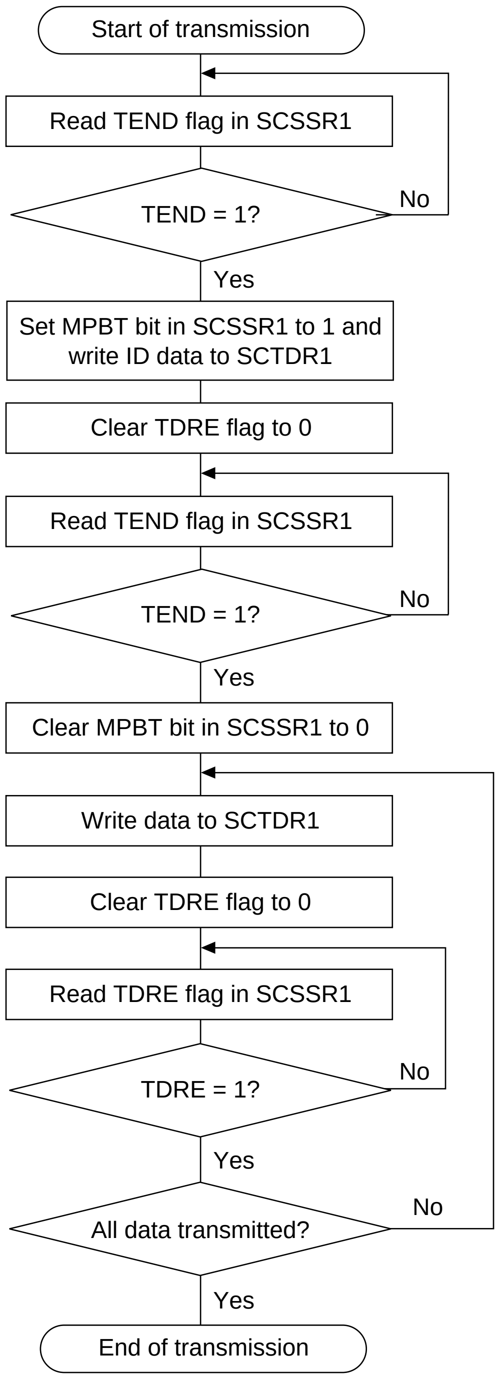
Clock

See the description under Clock in section 15.3.2.

Data Transfer Operations

Multiprocessor Serial Data Transmission: Figure 15.13 shows a sample flowchart for multiprocessor serial data transmission.

Use the following procedure for multiprocessor serial data transmission after enabling the SCI for transmission.



1. SCI status check and ID data write: Read SCSSR1 and check that the TEND flag is set to 1, then set the MPBT bit in SCSSR1 to 1 and write ID data to SCTDR1. Finally, clear the TDRE flag to 0.
2. Preparation for data transfer: Read SCSSR1 and check that the TEND flag is set to 1, then set the MPBT bit in SCSSR1 to 1.
3. Serial data transmission: Write the first transmit data to SCTDR1, then clear the TDRE flag to 0.

   To continue data transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to SCTDR1, and then clear the TDRE flag to 0. (Checking and clearing of the TDRE flag is automatic when the direct memory access controller (DMAC) is activated by a transmit-data-empty interrupt (TXI) request, and data is written to SCTDR1.)

Figure 15.13 Sample Multiprocessor Serial Transmission Flowchart

In serial transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SCSSR1. When TDRE is cleared to 0, the SCI recognizes that data has been written to SCTDR1, and transfers the data from SCTDR1 to SCTSR1.
2. After transferring data from SCTDR1 to SCTSR1, the SCI sets the TDRE flag to 1 and starts transmission.

   The serial transmit data is sent from the TxD pin in the following order.

   a. Start bit: One 0-bit is output.
   b. Transmit data: 8-bit or 7-bit data is output in LSB-first order.
   c. Multiprocessor bit: One multiprocessor bit (MPBT value) is output.
   d. Stop bit(s): One or two 1-bits (stop bits) are output.
   e. Mark state: 1 is output continuously until the start bit that starts the next transmission is sent.
3. The SCI checks the TDRE flag at the timing for sending the stop bit. If the TDRE flag is set to 1, the TEND flag in SCSSR1 is set to 1, the stop bit is sent, and then the line goes to the mark state in which 1 is output. If the TEIE bit in SCSCR1 is set to 1 at this time, a transmit-end interrupt (TEI) request is generated.
4. The SCI monitors the TDRE flag. When TDRE is cleared to 0, the SCI recognizes that data has been written to SCTDR1, and transfers the data from SCTDR1 to SCTSR1.
5. After transferring data from SCTDR1 to SCTSR1, the SCI sets the TDRE flag to 1 and starts transmitting. If the transmit-data-empty interrupt enable bit (TIE bit) in SCSCR1 is set to 1 at this time, a transmit-data-empty interrupt (TXI) request is generated.

   The order of transmission is the same as in step 2.

Figure 15.14 shows an example of SCI operation for transmission using a multiprocessor format.

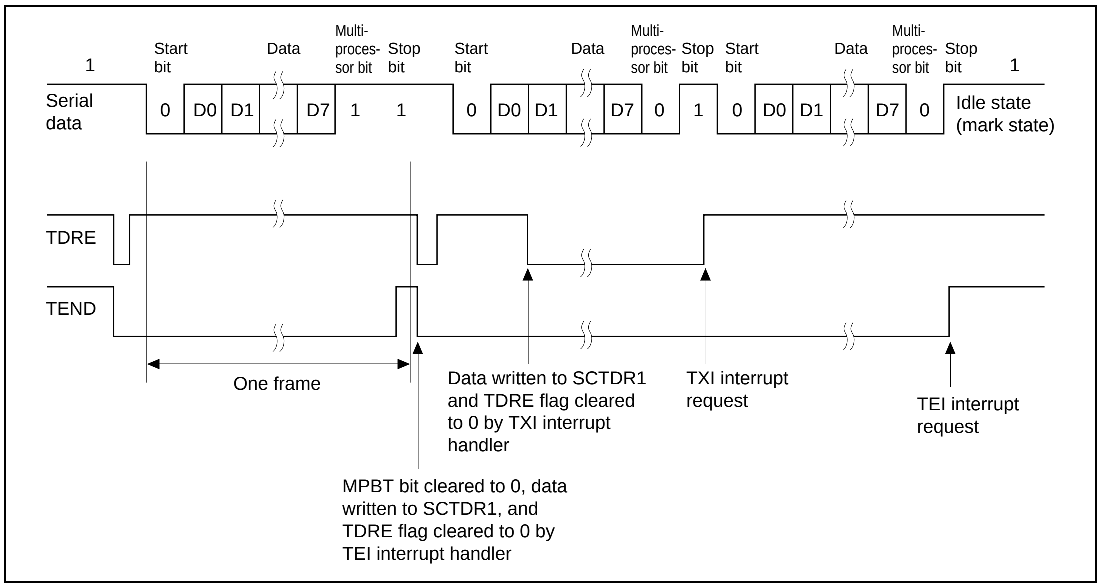
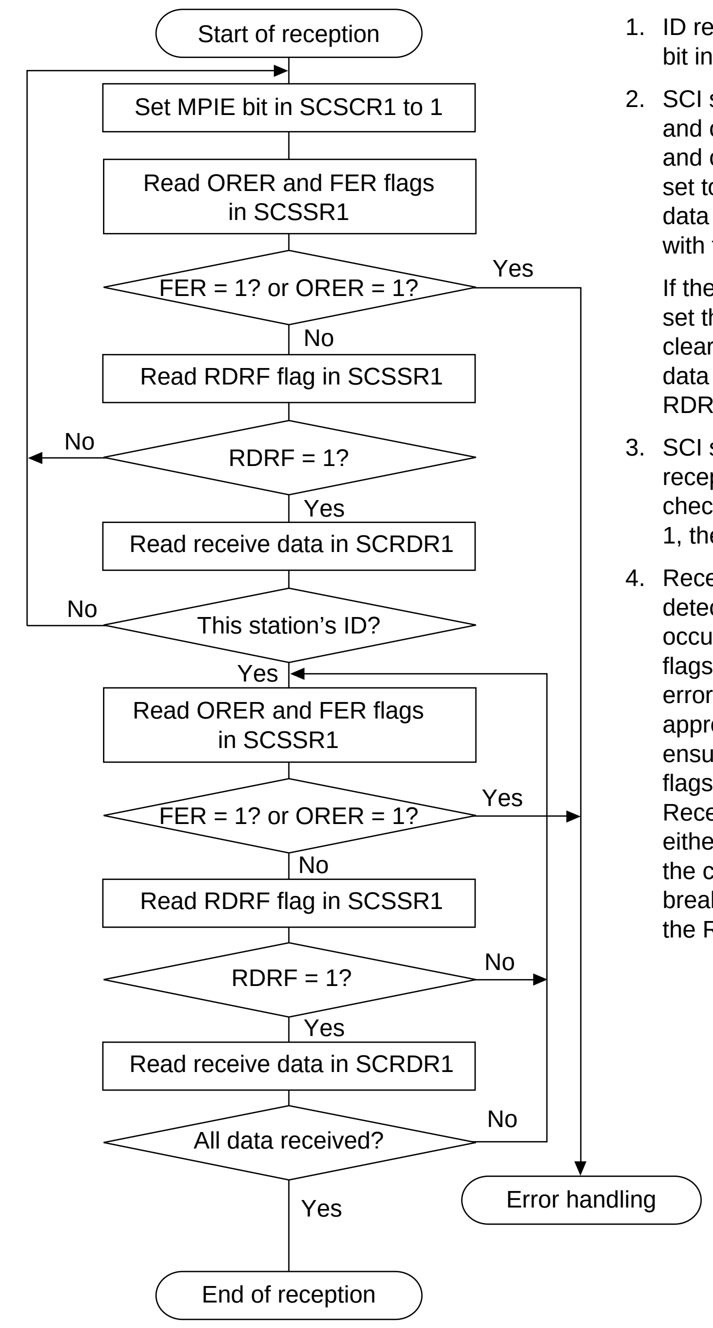


Figure 15.14 Example of SCI Transmit Operation (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)

Multiprocessor Serial Data Reception: Figure 15.15 shows a sample flowchart for multiprocessor serial reception.

Use the following procedure for multiprocessor serial data reception after enabling the SCI for reception.



1. ID reception cycle: Set the MPIE bit in SCSCR1 to 1.
2. SCI status check, ID reception and comparison: Read SCSSR1 and check that the RDRF flag is set to 1, then read the receive data in SCRDR1 and compare it with this station's ID.

   If the data is not this station's ID, set the MPIE bit to 1 again, and clear the RDRF flag to 0. If the data is this station's ID, clear the RDRF flag to 0.
3. SCI status check and data reception: Read SCSSR1 and check that the RDRF flag is set to 1, then read the data in SCRDR1.
4. Receive error handling and break detection: If a receive error occurs, read the ORER and FER flags in SCSSR1 to identify the error. After performing the appropriate error handling, ensure that the ORER and FER flags are all cleared to 0. Reception cannot be resumed if either of these flags is set to 1. In the case of a framing error, a break can be detected by reading the RxD pin value.

Figure 15.15 Sample Multiprocessor Serial Reception Flowchart (1)

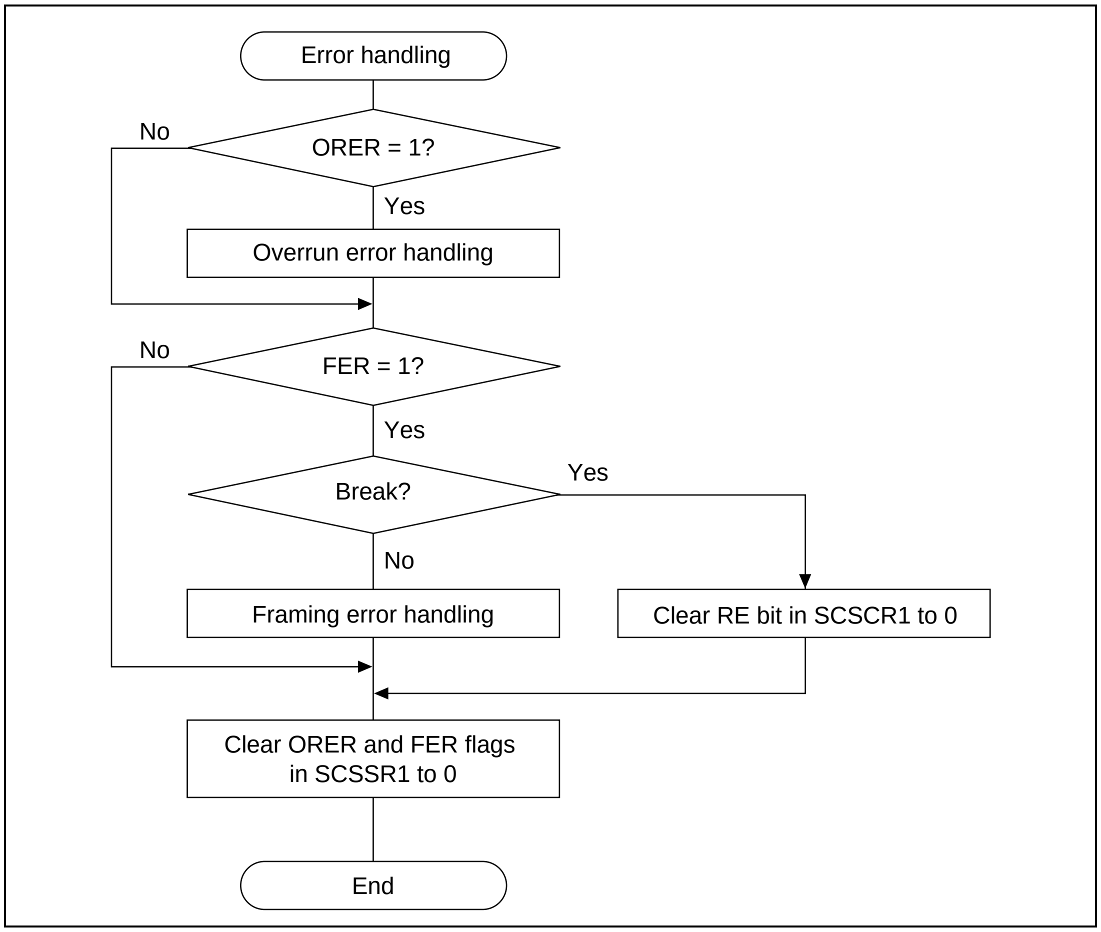
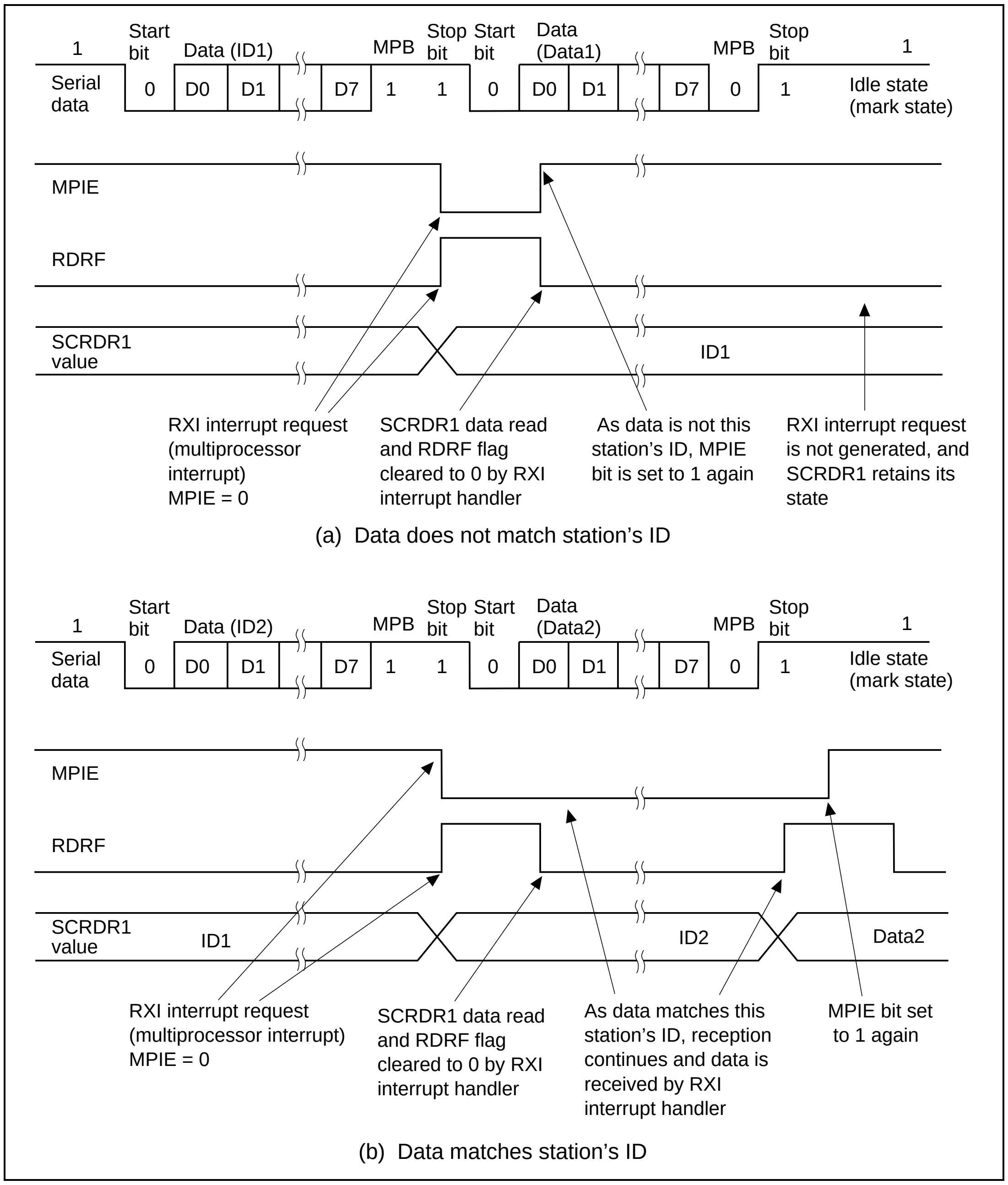


Figure 15.15 Sample Multiprocessor Serial Reception Flowchart (2)

Figure 15.16 shows an example of SCI operation for multiprocessor format reception.



1
Start bit
Data (ID1)
MPB
Stop bit
Start bit
Data (Data1)
MPB
Stop bit
1
Serial data
0
D0
D1
D7
1
1
0
D0
D1
D7
0
1
Idle state (mark state)
MPIE
RDRF
SCRDR1 value
ID1
RXI interrupt request (multiprocessor interrupt) MPIE = 0
SCRDR1 data read and RDRF flag cleared to 0 by RXI interrupt handler
As data is not this station's ID, MPIE bit is set to 1 again
RXI interrupt request is not generated, and SCRDR1 retains its state
(a) Data does not match station's ID
1
Start bit
Data (ID2)
MPB
Stop bit
Start bit
Data (Data2)
MPB
Stop bit
1
Serial data
0
D0
D1
D7
1
1
0
D0
D1
D7
0
1
Idle state (mark state)
MPIE
RDRF
SCRDR1 value
ID1
ID2
Data2
RXI interrupt request (multiprocessor interrupt) MPIE = 0
SCRDR1 data read and RDRF flag cleared to 0 by RXI interrupt handler
As data matches this station's ID, reception continues and data is received by RXI interrupt handler
MPIE bit set to 1 again
(b) Data matches station's ID

Figure 15.16 Example of SCI Receive Operation (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)

In multiprocessor mode serial reception, the SCI operates as described below.

1. The SCI 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 SCRSR1 in LSB-to-MSB order.
3. If the MPIE bit is 1, MPIE is cleared to 0 when a 1 is received in the multiprocessor bit position. If the multiprocessor bit is 0, the MPIE bit is not changed. The value of the multiprocessor bit is transferred to the MPB bit in SCSSR1.
4. If the MPIE bit is 0, RDRF is checked at the stop bit position, and if RDRF is 1 the overrun error bit is set. If the stop bit is not 0, the framing error bit is set. If RDRF is 0, the value in SCRSR1 is transferred to SCRDR1, and if the stop bit is 0, RDRF is set to 1.

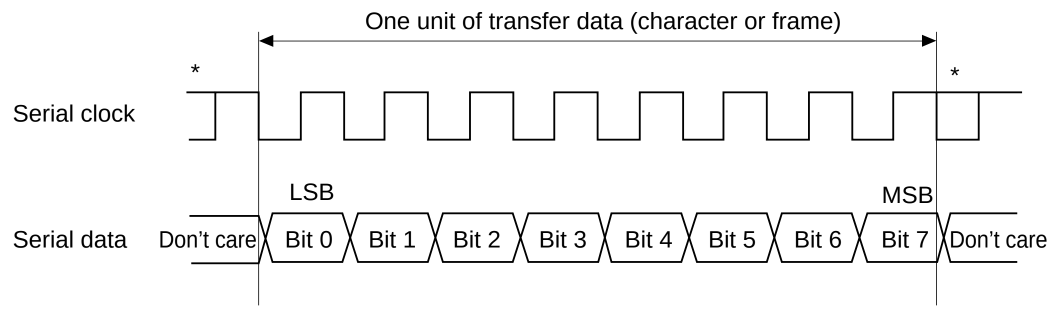
   If MPIE remains set to 1, the SCI ignores the received data.

15.3.4 Operation in Synchronous Mode

In synchronous mode, data is transmitted or received in synchronization with clock pulses, making it suitable for high-speed serial communication.

Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transfer.

Figure 15.17 shows the general format for synchronous serial communication.



Note: * High except in continuous transfer

Figure 15.17 Data Format in Synchronous Communication

HITACHI

In synchronous serial communication, data on the transmission line is output from one falling edge of the serial clock to the next. Data confirmation is guaranteed at the rising edge of the serial clock.

In serial communication, one character consists of data output starting with the LSB and ending with the MSB. After the MSB is output, the transmission line holds the MSB state.

In synchronous mode, the SCI receives data in synchronization with the falling edge of the serial clock.

Data Transfer Format

A fixed 8-bit data format is used. No parity or multiprocessor bits are added.

Clock

Either an internal clock generated by the on-chip baud rate generator or an external serial clock input at the SCK pin can be selected, according to the setting of the C/$\overline{\text{A}}$ bit in SCSMR1 and the CKE1 and CKE0 bits in SCSCR1. For details of SCI clock source selection, see table 15.9.

When the SCI is operated on an internal clock, the serial clock is output from the SCK pin.

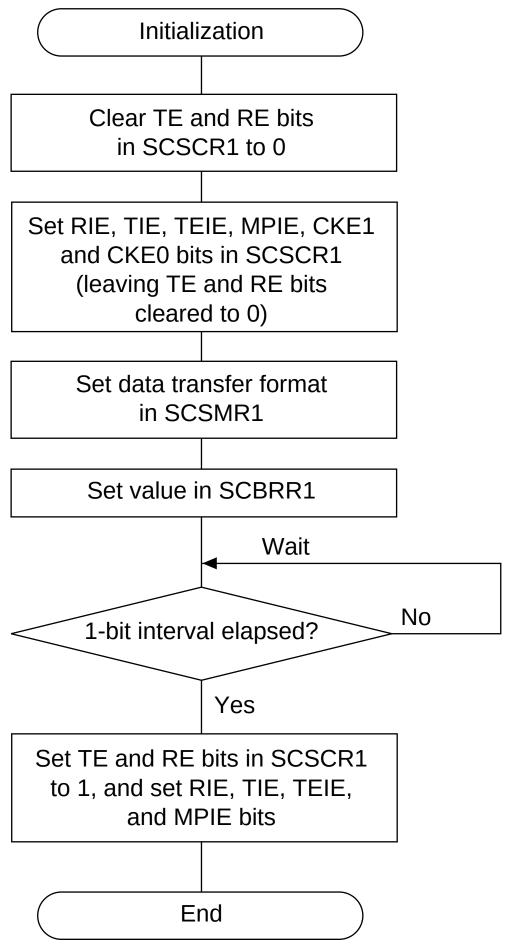
Eight serial clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high. In reception only, if an on-chip clock source is selected, clock pulses are output while RE = 1. When the last data is received, RE should be cleared to 0 before the end of bit 7.

Data Transfer Operations

SCI Initialization (Synchronous Mode): Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR1 to 0, then initialize the SCI as described below.

When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the TDRE flag is set to 1 and SCTSR1 is initialized. Note that clearing the RE bit to 0 does not change the contents of the RDRF, PER, FER, and ORER flags, or the contents of SCRDR1.

Figure 15.18 shows a sample SCI initialization flowchart.


Initialization
Clear TE and RE bits in SCSCR1 to 0
Set RIE, TIE, TEIE, MPIE, CKE1 and CKE0 bits in SCSCR1 (leaving TE and RE bits cleared to 0)
Set data transfer format in SCSMR1
Set value in SCBRR1
Wait
1-bit interval elapsed?
No
Yes
Set TE and RE bits in SCSCR1 to 1, and set RIE, TIE, TEIE, and MPIE bits
End

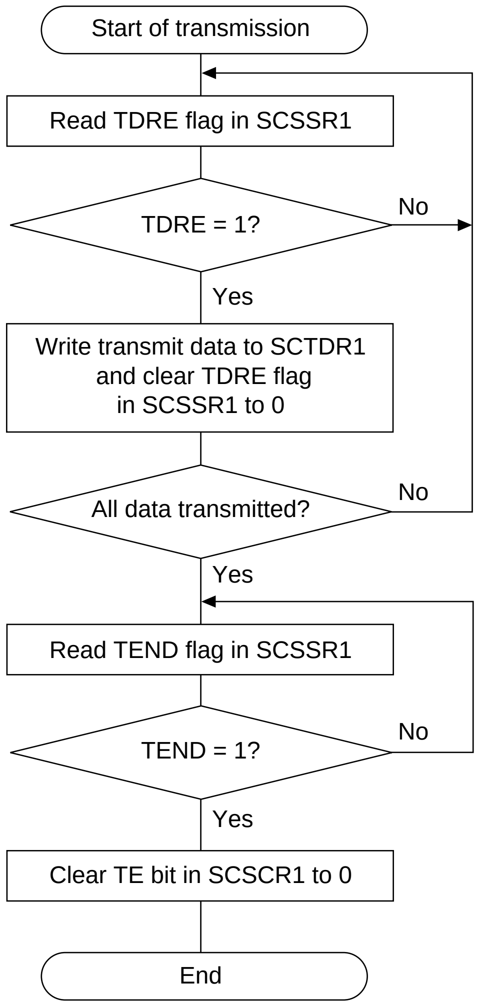
1. Set the clock selection in SCSCR1. Be sure to clear bits RIE, TIE, TEIE, and MPIE, TE and RE, to 0.
2. Set the data transfer format in SCSMR1.
3. Write a value corresponding to the bit rate into SCBRR1. (Not necessary if an external clock is used.)
4. Wait at least one bit interval, then set the TE bit or RE bit in SCSCR1 to 1.

   Also set the RIE, TIE, TEIE, and MPIE bits. Setting the TE and RE bits enables the TxD and RxD pins to be used.

Figure 15.18 Sample SCI Initialization Flowchart

Serial Data Transmission (Synchronous Mode): Figure 15.19 shows a sample flowchart for serial transmission.

Use the following procedure for serial data transmission after enabling the SCI for transmission.



1. SCI status check and transmit data write: Read SCSSR1 and check that the TDRE flag is set to 1, then write transmit data to SCTDR1 and clear the TDRE flag to 0.
2. To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to SCTDR1, and then clear the TDRE flag to 0. (Checking and clearing of the TDRE flag is automatic when the direct memory access controller (DMAC) is activated by a transmit-data-empty interrupt (TXI) request, and data is written to SCTDR1.)

Figure 15.19 Sample Serial Transmission Flowchart

In serial transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SCSSR1. When TDRE is cleared to 0, the SCI recognizes that data has been written to SCTDR1, and transfers the data from SCTDR1 to SCTSR1.
2. After transferring data from SCTDR1 to SCTSR1, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit is set to 1 at this time, a transmit-data-empty interrupt (TXI) request is generated.

   When clock output mode has been set, the SCI outputs 8 serial clock pulses. When use of an external clock has been specified, data is output synchronized with the input clock.

   The serial transmit data is sent from the TxD pin starting with the LSB (bit 0) and ending with the MSB (bit 7).
3. The SCI checks the TDRE flag at the timing for sending the MSB (bit 7).

   If the TDRE flag is cleared to 0, data is transferred from SCTDR1 to SCTSR1, and serial transmission of the next frame is started.

   If the TDRE flag is set to 1, the TEND flag in SCSSR1 is set to 1, the MSB (bit 7) is sent, and the TxD pin maintains its state.

   If the TEIE bit in SCSCR1 is set to 1 at this time, a transmit-end interrupt (TEI) request is generated.
4. After completion of serial transmission, the SCK pin is fixed high.

Figure 15.20 shows an example of SCI operation in transmission.

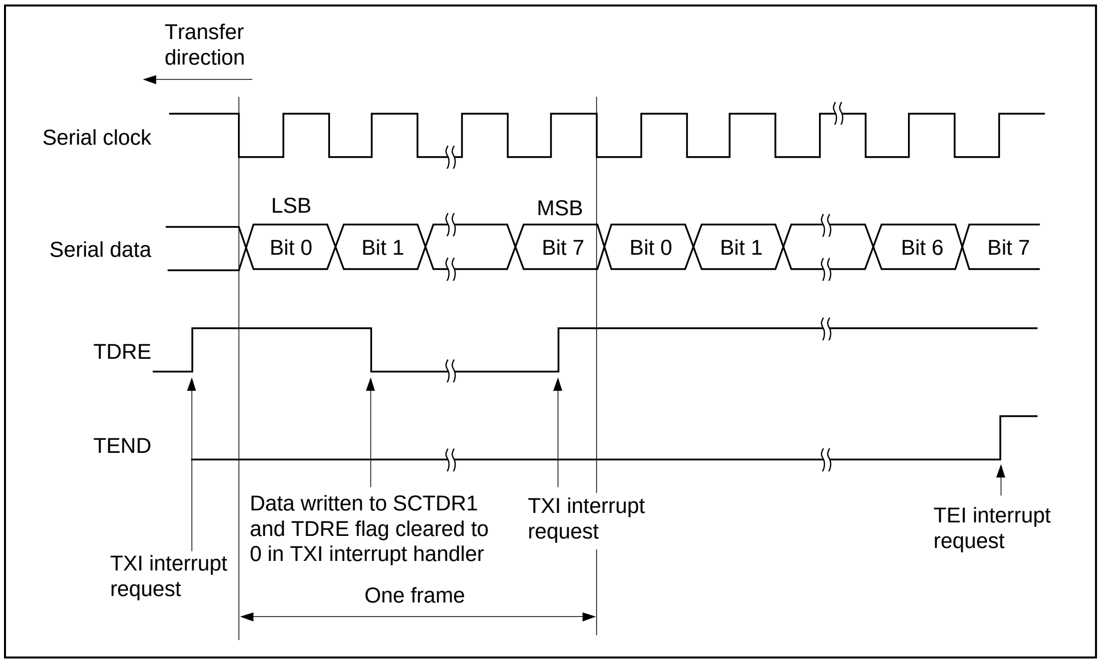
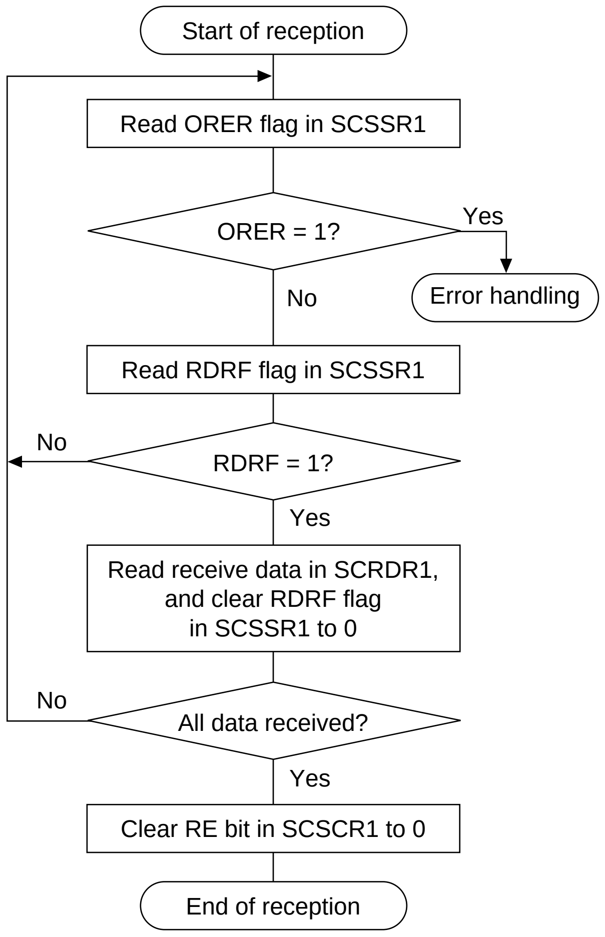


Figure 15.20 Example of SCI Transmit Operation

Serial Data Reception (Synchronous Mode): Figure 15.21 shows a sample flowchart for serial reception.

Use the following procedure for serial data reception after enabling the SCI for reception.

When changing the operating mode from asynchronous to synchronous, be sure to check that the ORER, PER, and FER flags are all cleared to 0. The RDRF flag will not be set if the FER or PER flag is set to 1, and neither transmit nor receive operations will be possible.



1. Receive error handling: If a receive error occurs, read the ORER flag in SCSSR1 , and after performing the appropriate error handling, clear the ORER flag to 0. Transfer cannot be resumed if the ORER flag is set to 1.
2. SCI status check and receive data read: Read SCSSR1 and check that the RDRF flag is set to 1, then read the receive data in SCRDR1 and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.
3. Serial reception continuation procedure: To continue serial reception, finish reading the RDRF flag, reading SCRDR1, and clearing the RDRF flag to 0, before the MSB (bit 7) of the current frame is received. (The RDRF flag is cleared automatically when the direct memory access controller (DMAC) is activated by a receive-data-full interrupt (RXI) request and the SCRDR1 value is read.)

Figure 15.21 Sample Serial Reception Flowchart (1)

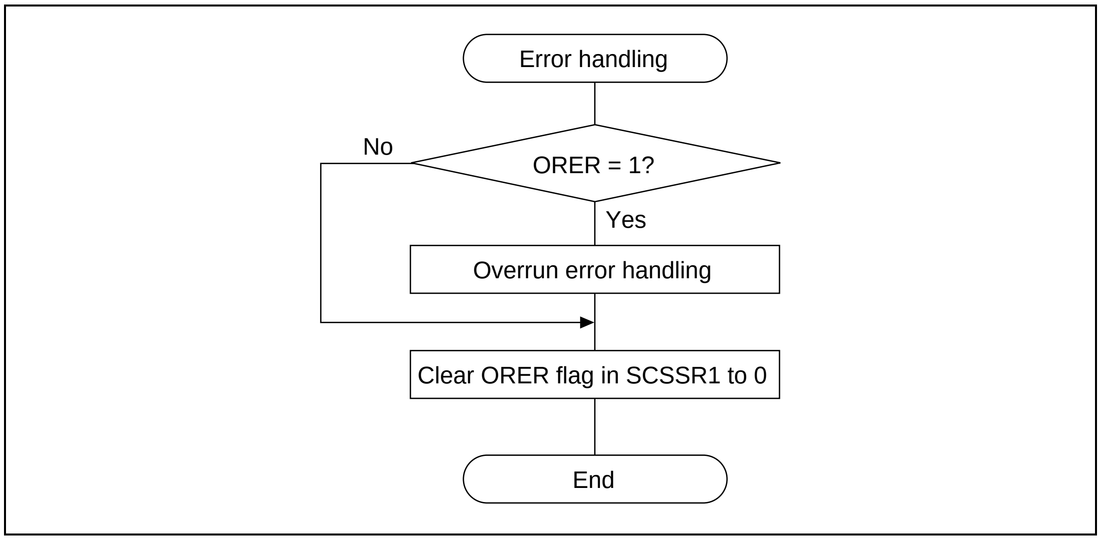


Figure 15.21 Sample Serial Reception Flowchart (2)

In serial reception, the SCI operates as described below.

1. The SCI performs internal initialization in synchronization with serial clock input or output.
2. The received data is stored in SCRSR1 in LSB-to-MSB order.

   After reception, the SCI checks whether the RDRF flag is 0, indicating that the receive data can be transferred from SCRSR1 to SCRDR1.

   If this check is passed, the RDRF flag is set to 1, and the receive data is stored in SCRDR1. If a receive error is detected in the error check, the operation is as shown in table 15.11.

   Neither transmit nor receive operations can be performed subsequently when a receive error has been found in the error check.

   Also, as the RDRF flag is not set to 1 when receiving, the flag must be cleared to 0.
3. If the RIE bit in SCRSR1 is set to 1 when the RDRF flag changes to 1, a receive-data-full interrupt (RXI) request is generated. If the RIE bit in SCRSR1 is set to 1 when the ORER flag changes to 1, a receive-error interrupt (ERI) request is generated.

Figure 15.22 shows an example of SCI operation in reception.

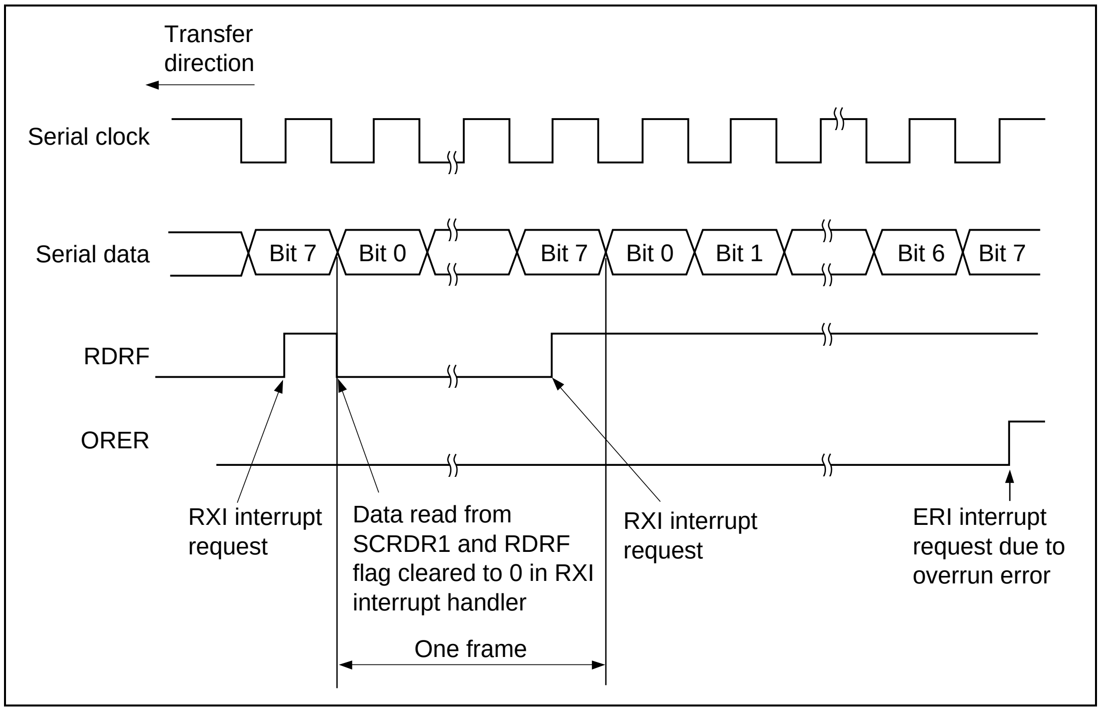
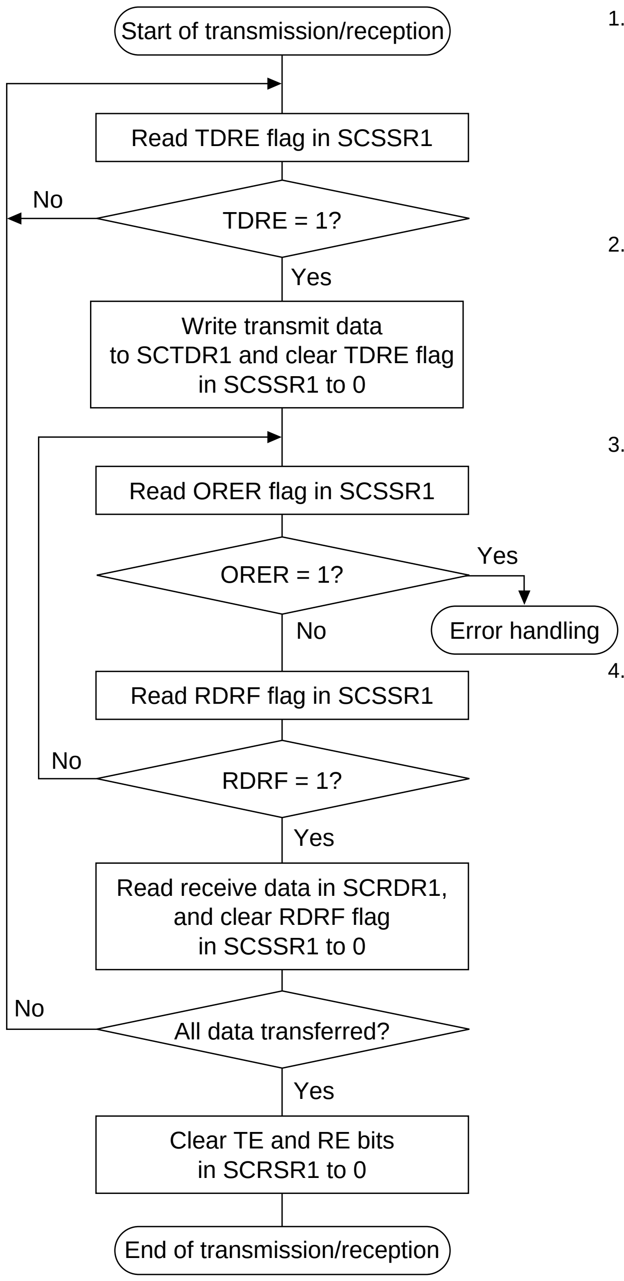


Figure 15.22 Example of SCI Receive Operation

Simultaneous Serial Data Transmission and Reception (Synchronous Mode): Figure 15.23 shows a sample flowchart for simultaneous serial transmit and receive operations.

Use the following procedure for simultaneous serial data transmit and receive operations after enabling the SCI for transmission and reception.


Start of transmission/reception
Read TDRE flag in SCSSR1
TDRE = 1?
No
Yes
Write transmit data to SCTDR1 and clear TDRE flag in SCSSR1 to 0
Read ORER flag in SCSSR1
ORER = 1?
Yes
No
Error handling
Read RDRF flag in SCSSR1
RDRF = 1?
No
Yes
Read receive data in SCRDR1, and clear RDRF flag in SCSSR1 to 0
All data transferred?
No
Yes
Clear TE and RE bits in SCRSR1 to 0
End of transmission/reception

1. SCI status check and transmit data write:
   Read SCSSR1 and check that the TDRE flag is set to 1, then write transmit data to SCTDR1 and clear the TDRE flag to 0. Transition of the TDRE flag from 0 to 1 can also be identified by a TXI interrupt.
2. Receive error handling:
   If a receive error occurs, read the ORER flag in SCSSR1 , and after performing the appropriate error handling, clear the ORER flag to 0. Transmission/reception cannot be resumed if the ORER flag is set to 1.
3. SCI status check and receive data read:
   Read SCSSR1 and check that the RDRF flag is set to 1, then read the receive data in SCRDR1 and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.
4. Serial transmission/reception continuation procedure:
   To continue serial transmission/ reception, finish reading the RDRF flag, reading SCRDR1, and clearing the RDRF flag to 0, before the MSB (bit 7) of the current frame is received. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible, then write data to SCTDR1 and clear the TDRE flag to 0.

   (Checking and clearing of the TDRE flag is automatic when the DMAC is activated by a transmit-data-empty interrupt (TXI) request, and data is written to SCTDR1. Similarly, the RDRF flag is cleared automatically when the DMAC is activated by a receive-data-full interrupt (RXI) request and the SCRDR1 value is read.)

Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE bit and RE bit to 0, then set both these bits to 1.

Figure 15.23 Sample Flowchart for Serial Data Transmission and Reception

15.4 SCI Interrupt Sources and DMAC

The SCI has four interrupt sources: the transmit-end interrupt (TEI) request, receive-error interrupt (ERI) request, receive-data-full interrupt (RXI) request, and transmit-data-empty interrupt (TXI) request.

Table 15.12 shows the interrupt sources and their relative priorities. Individual interrupt sources can be enabled or disabled with the TIE, RIE, and TEIE bits in SCRSR1, and the EIO bit in SCSPTR1. Each kind of interrupt request is sent to the interrupt controller independently.

When the TDRE flag in the serial status register (SCSSR1) is set to 1, a TDR-empty request is generated separately from the interrupt request. A TDR-empty request can activate the direct memory access controller (DMAC) to perform data transfer. The TDRE flag is cleared to 0 automatically when a write to the transmit data register (SCTDR1) is performed by the DMAC.

When the RDRF flag in SCSSR1 is set to 1, an RDR-full request is generated separately from the interrupt request. An RDR-full request can activate the DMAC to perform data transfer.

The RDRF flag is cleared to 0 automatically when a receive data register (SCRDR1) read is performed by the DMAC.

When the ORER, FER, or PER flag in SCSSR1 is set to 1, an ERI interrupt request is generated. The DMAC cannot be activated by an ERI interrupt request. When receive data processing is to be carried out by the DMAC and receive error handling is to be performed by means of an interrupt to the CPU, set the RIE bit to 1 and also set the EIO bit in SCSPTR1 to 1 so that an interrupt error occurs only for a receive error. If the EIO bit is cleared to 0, interrupts to the CPU will be generated even during normal data reception.

When the TEND flag in SCSSR1 is set to 1, a TEI interrupt request is generated. The DMAC cannot be activated by a TEI interrupt request.

A TXI interrupt indicates that transmit data can be written, and a TEI interrupt indicates that the transmit operation has ended.

Table 15.12 SCI Interrupt Sources

Interrupt Source	Description	DMAC Activation	Priority on Reset Release
ERI	Receive error (ORER, FER, or PER)	Not possible	High
RXI	Receive data register full (RDRF)	Possible	↑
TXI	Transmit data register empty (TDRE)	Possible	↓
TEI	Transmit end (TEND)	Not possible	Low

See section 5, Exceptions, for the priority order and relation to non-SCI interrupts.

15.5 Usage Notes

The following points should be noted when using the SCI.

SCTDR1 Writing and the TDRE Flag: The TDRE flag in SCSSR1 is a status flag that indicates that transmit data has been transferred from SCTDR1 to SCTSR1. When the SCI transfers data from SCTDR1 to SCTSR1, the TDRE flag is set to 1.

Data can be written to SCTDR1 regardless of the state of the TDRE flag. However, if new data is written to SCTDR1 when the TDRE flag is cleared to 0, the data stored in SCTDR1 will be lost since it has not yet been transferred to SCTSR1. It is therefore essential to check that the TDRE flag is set to 1 before writing transmit data to SCTDR1.

Simultaneous Multiple Receive Errors: If a number of receive errors occur at the same time, the state of the status flags in SCSSR1 is as shown in table 15.13. If there is an overrun error, data is not transferred from SCRSR1 to SCRDR1, and the receive data is lost.

Table 15.13 SCSSR1 Status Flags and Transfer of Receive Data

	SCSSR1 Status Flags				Receive Data Transfer SCRSR1 → SCRDR1
Receive Errors	**RDRF**	**ORER**	**FER**	**PER**	
Overrun error	1	1	0	0	X
Framing error	0	0	1	0	O
Parity error	0	0	0	1	O
Overrun error + framing error	1	1	1	0	X
Overrun error + parity error	1	1	0	1	X
Framing error + parity error	0	0	1	1	O
Overrun error + framing error + parity error	1	1	1	1	X

O: Receive data is transferred from SCRSR1 to SCRDR1.

X: Receive data is not transferred from SCRSR1 to SCRDR1.

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 the SCI receiver continues to operate in the break state, so if the FER flag is cleared to 0 it will be set to 1 again.

Sending a Break Signal: The input/output condition and level of the TxD pin are determined by bits SPB0IO and SPB0DT in the serial port register (SCSPTR1). This feature can be used to send a break signal.

After the serial transmitter is initialized, the TxD pin function is not selected and the value of the SPB0DT bit substitutes for the mark state until the TE bit is set to 1 (i.e. transmission is enabled). The SPB0IO and SPB0DT bits should therefore be set to 1 (designating output and high level) beforehand.

To send a break signal during serial transmission, clear the SPB0DT 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 its current state, and the TxD pin becomes an output port outputting the value 0.

Receive Error Flags and Transmit Operations (Synchronous Mode Only): Transmission cannot be started when a receive error flag (ORER, PER, or FER) is set to 1, even if the TDRE flag is set to 1. Be sure to clear the receive error flags to 0 before starting transmission.

Note also that the receive error flags are not cleared to 0 by clearing the RE bit to 0.

Receive Data Sampling Timing and Receive Margin in Asynchronous Mode: The SCI operates on a base clock with a frequency of 16 times the bit rate. In reception, the SCI 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 15.24.

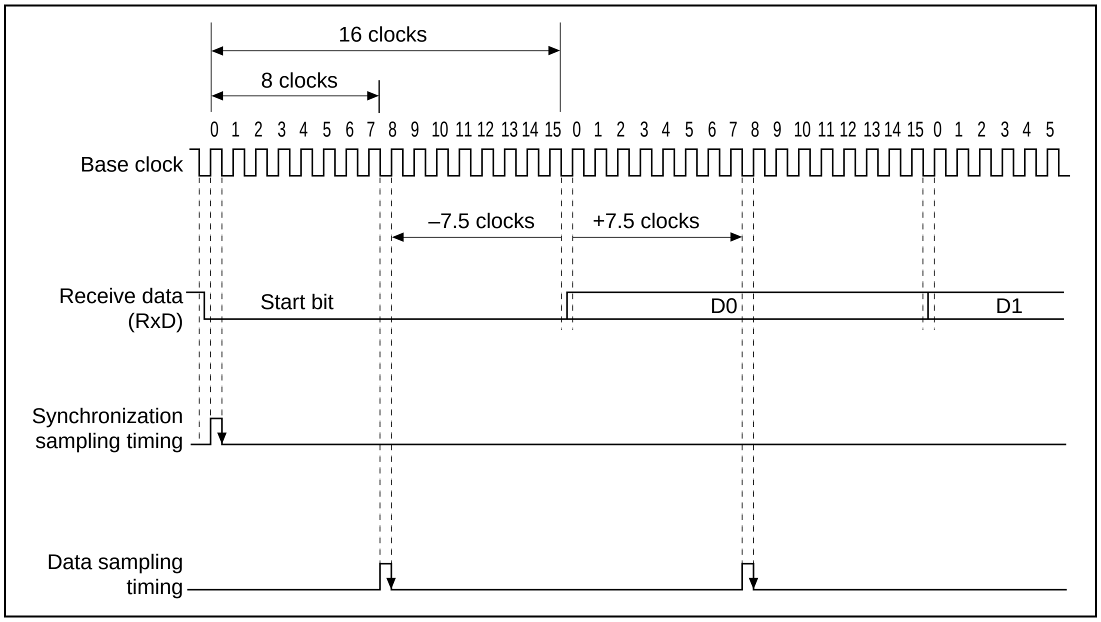


Figure 15.24 Receive Data Sampling Timing in Asynchronous Mode

The receive margin in asynchronous mode can therefore be expressed as shown in equation (1).

$$M = \left| (0.5 - \frac{1}{2N}) - (L - 0.5)\,F - \frac{|\,D - 0.5\,|}{N}\,(1 + F) \right| \times 100\% \quad .... \quad (1)$$

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

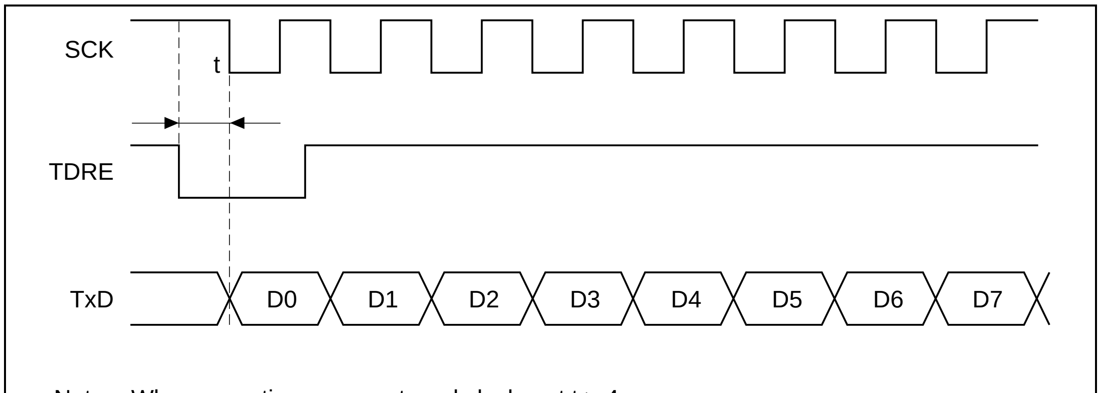
From equation (1), if F = 0 and D = 0.5, the receive margin is 46.875%, as given by equation (2).

When D = 0.5 and F = 0:

$$M = (0.5 - 1/(2 \times 16)) \times 100\% = 46.875\% \quad .... \quad (2)$$

This is a theoretical value. A reasonable margin to allow in system designs is 20% to 30%.

When Using the DMAC: When an external clock source is used as the serial clock, the transmit clock should not be input until at least 5 peripheral operating clock cycles after SCTDR1 is updated by the DMAC. Incorrect operation may result if the transmit clock is input within 4 cycles after SCTDR1 is updated. (See figure 15.25)



Note: When operating on an external clock, set t > 4.

Figure 15.25 Example of Synchronous Transmission by DMAC

When SCRDR1 is read by the DMAC, be sure to set the SCI receive-data-full interrupt (RXI) as the activation source with bits RS3 to RS0 in CHCR.

When Using Synchronous External Clock Mode:

- Do not set TE or RE to 1 until at least 4 peripheral operating clock cycles after external clock SCK has changed from 0 to 1.
- Only set both TE and RE to 1 when external clock SCK is 1.
- In reception, note that if RE is cleared to 0 from 2.5 to 3.5 peripheral operating clock cycles after the rising edge of the RxD D7 bit SCK input, RDRF will be set to 1 but copying to SCRDR1 will not be possible.

When Using Synchronous Internal Clock Mode: In reception, note that if RE is cleared to zero 1.5 peripheral operating clock cycles after the rising edge of the RxD D7 bit SCK output, RDRF will be set to 1 but copying to SCRDR1 will not be possible.

Section 16 Serial Communication Interface with FIFO (SCIF)

16.1 Overview

The SH7750 is equipped with a single-channel serial communication interface with built-in FIFO buffers (Serial Communication Interface with FIFO: SCIF). The SCIF can perform asynchronous serial communication.

Sixteen-stage FIFO registers are provided for both transmission and reception, enabling fast, efficient, and continuous communication.

16.1.1 Features

SCIF features are listed below.

- Asynchronous serial communication

  Serial data communication is executed using an asynchronous system in which synchronization is achieved character by character. Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA).

  There is a choice of 8 serial data transfer formats.

  — Data length: 7 or 8 bits

  — Stop bit length: 1 or 2 bits

  — Parity: Even/odd/none

  — Receive error detection: Parity, framing, and overrun errors

  — Break detection: If the receive data following that in which a framing error occurred is also at the space "0" level, and there is a frame error, a break is detected. When a framing error occurs, a break can also be detected by reading the RxD2 pin level directly from the serial port register (SCSPTR2).
- Full-duplex communication capability

  The transmitter and receiver are independent units, enabling transmission and reception to be performed simultaneously.

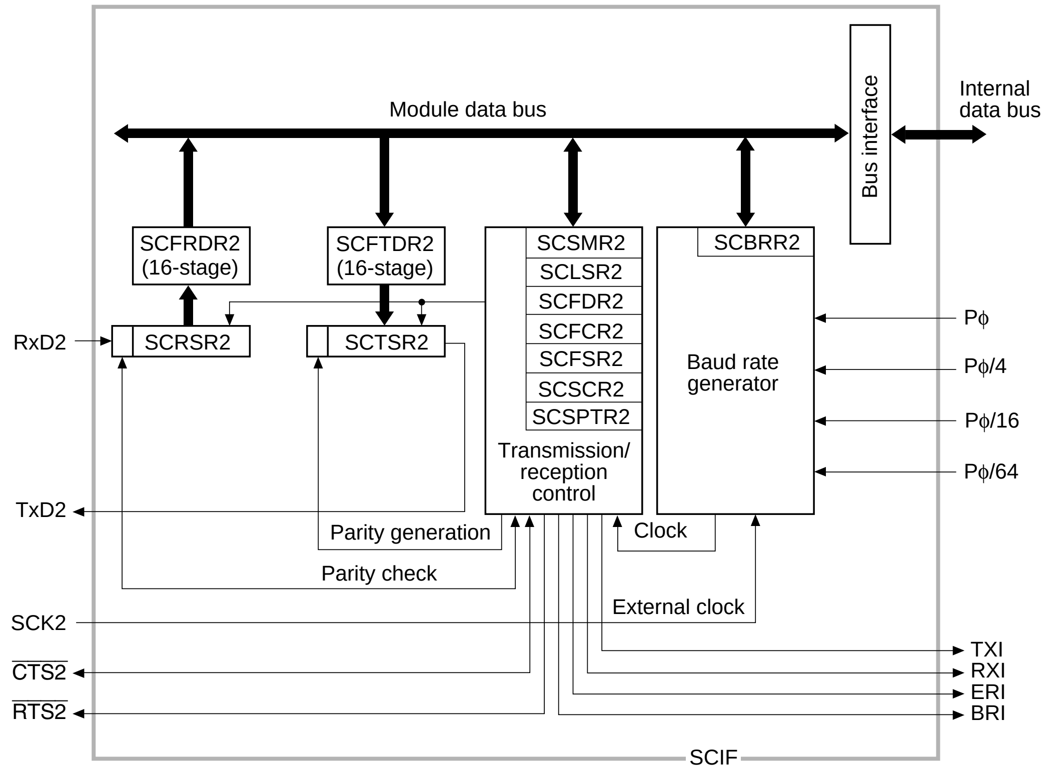
  The transmitter and receiver both have a 16-stage FIFO buffer structure, enabling fast and continuous serial data transmission and reception.
- On-chip baud rate generator allows any bit rate to be selected.
- Choice of serial clock source: internal clock from baud rate generator or external clock from SCK2 pin

- Four interrupt sources

  There are four interrupt sources—transmit-FIFO-data-empty, break, receive-FIFO-data-full, and receive-error—that can issue requests independently.
- The DMA controller (DMAC) can be activated to execute a data transfer by issuing a DMA transfer request in the event of a transmit-FIFO-data-empty or receive-FIFO-data-full interrupt.
- When not in use, the SCIF can be stopped by halting its clock supply to reduce power consumption.
- Modem control functions ($\overline{\text{RTS2}}$ and $\overline{\text{CTS2}}$) are provided.
- The amount of data in the transmit/receive FIFO registers, and the number of receive errors in the receive data in the receive FIFO register, can be ascertained.
- A timeout error (DR) can be detected during reception.

16.1.2 Block Diagram

Figure 16.1 shows a block diagram of the SCIF.



SCRSR2:	Receive shift register	SCFSR2:	Serial status register
SCFRDR2:	Receive FIFO data register	SCBRR2:	Bit rate register
SCTSR2:	Transmit shift register	SCSPTR2:	Serial port register
SCFTDR2:	Transmit FIFO data register	SCFCR2:	FIFO control register
SCSMR2:	Serial mode register	SCFDR2:	FIFO data count register
SCSCR2:	Serial control register	SCLSR2:	Line status register

Figure 16.1 Block Diagram of SCIF

16.1.3 Pin Configuration

Table 16.1 shows the SCIF pin configuration.

Table 16.1 SCIF Pins

Pin Name	Abbreviation	I/O	Function
Serial clock pin	MRESET/SCK2	Input	Clock input
Receive data pin	MD2/RxD2	Input	Receive data input
Transmit data pin	MD1/TxD2	Output	Transmit data output
Modem control pin	$\overline{\text{CTS2}}$	I/O	Transmission enabled
Modem control pin	MD8/$\overline{\text{RTS2}}$	I/O	Transmission request

Note: The MRESET/SCK2 pin functions as the MRESET manual reset pin when a manual reset is executed. The MD1/TxD2, MD2/RxD2, and MD8/$\overline{\text{RTS2}}$ pins function as the MD1, MD2, and MD8 mode input pins after a power-on reset. These pins are made to function as serial pins by performing SCIF operation settings with the TE and RE bits in SCSCR2 and the MCE bit in SCFCR2. Break state transmission and detection can be set in the SCIF's SCSPTR2 register.

16.1.4 Register Configuration

The SCIF has the internal registers shown in table 16.2. These registers are used to specify the data format and bit rate, and to perform transmitter/receiver control.

Table 16.2 SCIF Registers

Name	Abbrevia-tion	R/W	Initial Value	P4 Address	Area 7 Address	Access Size
Serial mode register	SCSMR2	R/W	H'0000	H'FFE80000	H'IFE80000	16
Bit rate register	SCBRR2	R/W	H'FF	H'FFE80004	H'IFE80004	8
Serial control register	SCSCR2	R/W	H'0000	H'FFE80008	H'IFE80008	16
Transmit FIFO data register	SCFTDR2	W	Undefined	H'FFE8000C	H'IFE8000C	8
Serial status register	SCFSR2	R/(W)*[1]	H'0060	H'FFE80010	H'IFE80010	16
Receive FIFO data register	SCFRDR2	R	Undefined	H'FFE80014	H'IFE80014	8
FIFO control register	SCFCR2	R/W	H'0000	H'FFE80018	H'IFE80018	16
FIFO data count register	SCFDR2	R	H'0000	H'FFE8001C	H'IFE8001C	16
Serial port register	SCSPTR2	R/W	H'0000*[2]	H'FFE80020	H'IFE80020	16
Line status register	SCLSR2	R/(W)*[3]	H'0000	H'FFE80024	H'IFE80024	16

Notes: 1. Only 0 can be written, to clear flags. Bits 15 to 8, 3, and 2 are read-only, and cannot be modified.

2. The value of bits 6, 4, and 0 is undefined.

3. Only 0 can be written, to clear flags. Bits 15 to 1 are read-only, and cannot be modified.

16.2 Register Descriptions

16.2.1 Receive Shift Register (SCRSR2)

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

SCRSR2 is the register used to receive serial data.

The SCIF sets serial data input from the RxD2 pin in SCRSR2 in the order received, starting with the LSB (bit 0), and converts it to parallel data. When one byte of data has been received, it is transferred to the receive FIFO register, SCFRDR2, automatically.

SCRSR2 cannot be directly read or written to by the CPU.

16.2.2 Receive FIFO Data Register (SCFRDR2)

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

SCFRDR2 is a 16-stage FIFO register that stores received serial data.

When the SCIF has received one byte of serial data, it transfers the received data from SCRSR2 to SCFRDR2 where it is stored, and completes the receive operation. SCRSR2 is then enabled for reception, and consecutive receive operations can be performed until the receive FIFO register is full (16 data bytes).

SCFRDR2 is a read-only register, and cannot be written to by the CPU.

If a read is performed when there is no receive data in the receive FIFO register, an undefined value will be returned. When the receive FIFO register is full of receive data, subsequent serial data is lost.

The contents of SCFRDR2 are undefined after a power-on reset or manual reset.

16.2.3 Transmit Shift Register (SCTSR2)

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

SCTSR2 is the register used to transmit serial data.

To perform serial data transmission, the SCIF first transfers transmit data from SCFTDR2 to SCTSR2, then sends the data to the TxD2 pin starting with the LSB (bit 0).

When transmission of one byte is completed, the next transmit data is transferred from SCFTDR2 to SCTSR2, and transmission started, automatically.

SCTSR2 cannot be directly read or written to by the CPU.

16.2.4 Transmit FIFO Data Register (SCFTDR2)

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

SCFTDR2 is a 16-stage FIFO register that stores data for serial transmission.

If SCTSR2 is empty when transmit data has been written to SCFTDR2, the SCIF transfers the transmit data written in SCFTDR2 to SCTSR2 and starts serial transmission.

SCFTDR2 is a write-only register, and cannot be read by the CPU.

The next data cannot be written when SCFTDR2 is filled with 16 bytes of transmit data. Data written in this case is ignored.

The contents of SCFTDR2 are undefined after a power-on reset or manual reset.

16.2.5 Serial Mode Register (SCSMR2)

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

Bit:	7	6	5	4	3	2	1	0
	—	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

SCSMR2 is a 16-bit register used to set the SCIF's serial transfer format and select the baud rate generator clock source.

SCSMR2 can be read or written to by the CPU at all times.

SCSMR2 is initialized to H'0000 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bits 15 to 7—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 6—Character Length (CHR): Selects 7 or 8 bits as the asynchronous mode data length.

Bit 6: CHR	Description	
0	8-bit data	(Initial value)
1	7-bit data*	

Note: * When 7-bit data is selected, the MSB (bit 7) of SCFTDR2 is not transmitted.

Bit 5—Parity Enable (PE): Selects whether or not parity bit addition is performed in transmission, and parity bit checking in reception.

Bit 5: PE	Description	
0	Parity bit addition and checking disabled	(Initial value)
1	Parity bit addition and checking enabled*	

Note: * When the PE bit is set to 1, the parity (even or odd) specified by the $O/\overline{E}$ bit is added to transmit data before transmission. In reception, the parity bit is checked for the parity (even or odd) specified by the $O/\overline{E}$ bit.

Bit 4—Parity Mode (O/$\overline{\text{E}}$): Selects either even or odd parity for use in parity addition and checking. The O/$\overline{\text{E}}$ bit setting is only valid when the PE bit is set to 1, enabling parity bit addition and checking. The O/$\overline{\text{E}}$ bit setting is invalid when parity addition and checking is disabled.

Bit 4: O/$\overline{\text{E}}$	Description	
0	Even parity*[1]	(Initial value)
1	Odd parity*[2]	

Notes: 1. When even parity is set, parity bit addition is performed in transmission so that the total number of 1-bits in the transmit character plus the parity bit is even. In reception, a check is performed to see if the total number of 1-bits in the receive character plus the parity bit is even.

2. When odd parity is set, parity bit addition is performed in transmission so that the total number of 1-bits in the transmit character plus the parity bit is odd. In reception, a check is performed to see if the total number of 1-bits in the receive character plus the parity bit is odd.

Bit 3—Stop Bit Length (STOP): Selects 1 or 2 bits as the stop bit length.

Bit 3: STOP	Description	
0	1 stop bit*[1]	(Initial value)
1	2 stop bits*[2]	

Notes: 1. In transmission, a single 1-bit (stop bit) is added to the end of a transmit character before it is sent.

2. In transmission, two 1-bits (stop bits) are added to the end of a transmit character before it is sent.

In reception, only the first stop bit is checked, regardless of the STOP bit setting. If the second stop bit is 1, it is treated as a stop bit; if it is 0, it is treated as the start bit of the next transmit character.

Bit 2—Reserved: This bit is always read as 0, and should only be written with 0.

Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0): These bits select the clock source for the on-chip baud rate generator. The clock source can be selected from Pϕ, Pϕ/4, Pϕ/16, and Pϕ/64, according to the setting of bits CKS1 and CKS0.

For the relation between the clock source, the bit rate register setting, and the baud rate, see section 16.2.8, Bit Rate Register (SCBRR2).

Bit 1: CKS1	Bit 0: CKS0	Description	
0	0	Pϕ clock	(Initial value)
	1	Pϕ/4 clock	
1	0	Pϕ/16 clock	
	1	Pϕ/64 clock	

Note: Pϕ: Peripheral clock

16.2.6 Serial Control Register (SCSCR2)

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

Bit:	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	REIE	—	CKE1	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R	R/W	R

The SCSCR2 register performs enabling or disabling of SCIF transfer operations, and interrupt requests, and selection of the serial clock source.

SCSCR2 can be read or written to by the CPU at all times.

SCSCR2 is initialized to H'0000 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bits 15 to 8, 2, and 0—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 7—Transmit Interrupt Enable (TIE): Enables or disables transmit-FIFO-data-empty interrupt (TXI) request generation when serial transmit data is transferred from SCFTDR2 to SCTSR2, the number of data bytes in the transmit FIFO register falls to or below the transmit trigger set number, and the TDFE flag in the serial status register (SCFSR2) is set to 1.

Bit 7: TIE	Description	
0	Transmit-FIFO-data-empty interrupt (TXI) request disabled*	(Initial value)
1	Transmit-FIFO-data-empty interrupt (TXI) request enabled	

Note: * TXI interrupt requests can be cleared by writing transmit data exceeding the transmit trigger set number to SCFTDR2, reading 1 from the TDFE flag, then clearing it to 0, or by clearing the TIE bit to 0.

Bit 6—Receive Interrupt Enable (RIE): Enables or disables generation of a receive-data-full interrupt (RXI) request when the RDF flag or DR flag in SCFSR2 is set to 1, a receive-error interrupt (ERI) request when the ER flag in SCFSR2 is set to 1, and a break interrupt (BRI) request when the BRK flag in SCFSR2 or the ORER flag in SCLSR2 is set to 1.

Bit 6: RIE	Description	
0	Receive-data-full interrupt (RXI) request, receive-error interrupt (ERI) request, and break interrupt (BRI) request disabled*	(Initial value)
1	Receive-data-full interrupt (RXI) request, receive-error interrupt (ERI) request, and break interrupt (BRI) request enabled	

Note: * An RXI interrupt request can be cleared by reading 1 from the RDF or DR flag, then clearing the flag to 0, or by clearing the RIE bit to 0. ERI and BRI interrupt requests can be cleared by reading 1 from the ER, BRK, or ORER flag, then clearing the flag to 0, or by clearing the RIE and REIE bits to 0.

Bit 5—Transmit Enable (TE): Enables or disables the start of serial transmission by the SCIF.

Bit 5: TE	Description	
0	Transmission disabled	(Initial value)
1	Transmission enabled*	

Note: * Serial transmission is started when transmit data is written to SCFTDR2 in this state.
Serial mode register (SCSMR2) and FIFO control register (SCFCR2) settings must be made, the transmission format decided, and the transmit FIFO reset, before the TE bit is set to 1.

Bit 4—Receive Enable (RE): Enables or disables the start of serial reception by the SCIF.

Bit 4: RE	Description	
0	Reception disabled*1	(Initial value)
1	Reception enabled*2	

Notes: 1. Clearing the RE bit to 0 does not affect the DR, ER, BRK, RDF, FER, PER, and ORER flags, which retain their states.

2. Serial transmission is started when a start bit is detected in this state.

Serial mode register (SCSMR2) and FIFO control register (SCFCR2) settings must be made, the reception format decided, and the receive FIFO reset, before the RE bit is set to 1.

Bit 3—Receive Error Interrupt Enable (REIE): Enables or disables generation of receive-error interrupt (ERI) and break interrupt (BRI) requests. The REIE bit setting is valid only when the RIE bit is 0.

Bit 3: REIE	Description
0	Receive-error interrupt (ERI) and break interrupt (BRI) requests disabled* (Initial value)
1	Receive-error interrupt (ERI) and break interrupt (BRI) requests enabled

Note: * Receive-error interrupt (ERI) and break interrupt (BRI) requests can be cleared by reading 1 from the ER, BRK, or ORER flag, then clearing the flag to 0, or by clearing the RIE and REIE bits to 0. When REIE is set to 1, ERI and BRI interrupt requests will be generated even if RIE is cleared to 0. In DMAC transfer, this setting is made if the interrupt controller is to be notified of ERI and BRI interrupt requests.

Bit 1—Clock Enable 1 (CKE1): Selects the SCIF clock source. The CKE1 bit must be set before determining the SCIF's operating mode with SCSMR2.

Bit 1: CKE1	Description
0	Internal clock/SCK2 pin functions as input pin (input signal ignored)*1
1	External clock/SCK2 pin functions as clock input*2

Notes: 1. Initial value

2. Inputs a clock with a frequency 16 times the bit rate.

16.2.7 Serial Status Register (SCFSR2)

Bit:	15	14	13	12	11	10	9	8
	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

Bit:	7	6	5	4	3	2	1	0
	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: * Only 0 can be written, to clear the flag.

SCFSR2 is a 16-bit register. The lower 8 bits consist of status flags that indicate the operating status of the SCIF, and the upper 8 bits indicate the number of receive errors in the data in the receive FIFO register.

SCFSR2 can be read or written to by the CPU at all times. However, 1 cannot be written to flags ER, TEND, TDFE, BRK, RDF, and DR. Also note that in order to clear these flags they must be read as 1 beforehand. The FER flag and PER flag are read-only flags and cannot be modified.

SCFSR2 is initialized to H'0060 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bits 15 to 12—Number of Parity Errors (PER3–PER0): These bits indicate the number of data bytes in which a parity error occurred in the receive data stored in SCFRDR2.

After the ER bit in SCFSR2 is set, the value indicated by bits 15 to 12 is the number of data bytes in which a parity error occurred.

If all 16 bytes of receive data in SCFRDR2 have parity errors, the value indicated by bits PER3 to PER0 will be 0.

Bits 11 to 8—Number of Framing Errors (FER3–FER0): These bits indicate the number of data bytes in which a framing error occurred in the receive data stored in SCFRDR2.

After the ER bit in SCFSR2 is set, the value indicated by bits 11 to 8 is the number of data bytes in which a framing error occurred.

If all 16 bytes of receive data in SCFRDR2 have framing errors, the value indicated by bits FER3 to FER0 will be 0.

Bit 7—Receive Error (ER): Indicates that a framing error or parity error occurred during reception.*1

Bit 7: ER	Description
0	No framing error or parity error occurred during reception (Initial value) [Clearing conditions] • Power-on reset or manual reset • When 0 is written to ER after reading ER = 1
1	A framing error or parity error occurred during reception [Setting conditions] • When the SCIF checks whether the stop bit at the end of the receive data is 1 when reception ends, and the stop bit is 0*2 • When, in reception, the number of 1-bits in the receive data plus the parity bit does not match the parity setting (even or odd) specified by the $O/\overline{E}$ bit in SCSMR2

Notes: 1. The ER flag is not affected and retains its previous state when the RE bit in SCSCR2 is cleared to 0. When a receive error occurs, the receive data is still transferred to SCFRDR2, and reception continues.

The FER and PER bits in SCFSR2 can be used to determine whether there is a receive error in the data read from SCFRDR2.

2. In 2-stop-bit mode, only the first stop bit is checked for a value of 1; the second stop bit is not checked.

Bit 6—Transmit End (TEND): Indicates that there is no valid data in SCFTDR2 when the last bit of the transmit character is sent, and transmission has been ended.

Bit 6: TEND	Description
0	Transmission is in progress [Clearing conditions] • When transmit data is written to SCFTDR2, and 0 is written to TEND after reading TEND = 1 • When data is written to SCFTDR2 by the DMAC
1	Transmission has been ended (Initial value) [Setting conditions] • Power-on reset or manual reset • When the TE bit in SCSCR2 is 0 • When there is no transmit data in SCFTDR2 on transmission of the last bit of a 1-byte serial transmit character

Bit 5—Transmit FIFO Data Empty (TDFE): Indicates that data has been transferred from SCFTDR2 to SCTSR2, the number of data bytes in SCFTDR2 has fallen to or below the transmit trigger data number set by bits TTRG1 and TTRG0 in the FIFO control register (SCFCR2), and new transmit data can be written to SCFTDR2.

Bit 5: TDFE	Description
0	A number of transmit data bytes exceeding the transmit trigger set number have been written to SCFTDR2 [Clearing conditions] • When transmit data exceeding the transmit trigger set number is written to SCFTDR2, and 0 is written to TDFE after reading TDFE = 1 • When transmit data exceeding the transmit trigger set number is written to SCFTDR2 by the DMAC
1	The number of transmit data bytes in SCFTDR2 does not exceed the transmit trigger set number (Initial value) [Setting conditions] • Power-on reset or manual reset • When the number of SCFTDR2 transmit data bytes falls to or below the transmit trigger set number as the result of a transmit operation*

Note: * As SCFTDR2 is a 16-byte FIFO register, the maximum number of bytes that can be written when TDFE = 1 is 16 - (transmit trigger set number). Data written in excess of this will be ignored.
The number of data bytes in SCFTDR2 is indicated by the upper bits of SCFDR2.

Bit 4—Break Detect (BRK): Indicates that a receive data break signal has been detected.

Bit 4: BRK	Description
0	A break signal has not been received (Initial value) [Clearing conditions] • Power-on reset or manual reset • When 0 is written to BRK after reading BRK = 1
1	A break signal has been received* [Setting condition] When data with a framing error is received, followed by the space “0” level (low level) for at least one frame length

Note: * When a break is detected, the receive data (H'00) following detection is not transferred to SCFRDR2. When the break ends and the receive signal returns to mark “1”, receive data transfer is resumed.

Bit 3—Framing Error (FER): Indicates a framing error in the data read from SCFRDR2.

Bit 3: FER	Description
0	There is no framing error in the receive data read from SCFRDR2 (Initial value) [Clearing conditions] • Power-on reset or manual reset • When there is no framing error in SCFRDR2 read data
1	There is a framing error in the receive data read from SCFRDR2 [Setting condition] When there is a framing error in SCFRDR2 read data

Bit 2—Parity Error (PER): Indicates a parity error in the data read from SCFRDR2.

Bit 2: PER	Description
0	There is no parity error in the receive data read from SCFRDR2 (Initial value) [Clearing conditions] • Power-on reset or manual reset • When there is no parity error in SCFRDR2 read data
1	There is a parity error in the receive data read from SCFRDR2 [Setting condition] When there is a parity error in SCFRDR2 read data

Bit 1—Receive FIFO Data Full (RDF): Indicates that the received data has been transferred from SCRSR2 to SCFRDR2, and the number of receive data bytes in SCFRDR2 is equal to or greater than the receive trigger number set by bits RTRG1 and RTRG0 in the FIFO control register (SCFCR2).

Bit 1: RDF	Description
0	The number of receive data bytes in SCFRDR2 is less than the receive trigger set number (Initial value) [Clearing conditions] • Power-on reset or manual reset • When SCFRDR2 is read until the number of receive data bytes in SCFRDR2 falls below the receive trigger set number, and 0 is written to RDF after reading RDF = 1 • When SCFRDR2 is read by the DMAC until the number of receive data bytes in SCFRDR2 falls below the receive trigger set number
1	The number of receive data bytes in SCFRDR2 is equal to or greater than the receive trigger set number [Setting condition] When SCFRDR2 contains at least the receive trigger set number of receive data bytes*

Note: * SCFRDR2 is a 16-byte FIFO register. When RDF = 1, at least the receive trigger set number of data bytes can be read. If all the data in SCFRDR2 is read and another read is performed, the data value will be undefined. The number of receive data bytes in SCFRDR2 is indicated by the lower bits of SCFDR2.

Bit 0—Receive Data Ready (DR): Indicates that there are fewer than the receive trigger set number of data bytes in SCFRDR2, and no further data has arrived for at least 15 etu after the stop bit of the last data received.

Bit 0: DR	Description
0	Reception is in progress or has ended normally and there is no receive data left in SCFRDR2 (Initial value) [Clearing conditions] • Power-on reset or manual reset • When all the receive data in SCFRDR2 has been read, and 0 is written to DR after reading DR = 1 • When all the receive data in SCFRDR2 has been read by the DMAC
1	No further receive data has arrived [Setting condition] When SCFRDR2 contains fewer than the receive trigger set number of receive data bytes, and no further data has arrived for at least 15 etu after the stop bit of the last data received*

Note: * Equivalent to 1.5 frames with an 8-bit, 1-stop-bit format.
etu: Elementary time unit (time for transfer of 1 bit)

16.2.8 Bit Rate Register (SCBRR2)

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

SCBRR2 is an 8-bit register that sets the serial transfer bit rate in accordance with the baud rate generator operating clock selected by bits CKS1 and CKS0 in SCSMR2.

SCBRR2 can be read or written to by the CPU at all times.

SCBRR2 is initialized to H'FF by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

The SCBRR2 setting is found from the following equation.

Asynchronous mode:

$$N = \frac{P\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Where B: Bit rate (bits/s)
N: SCBRR2 setting for baud rate generator ($0 \le N \le 255$)
Pϕ: Peripheral module operating frequency (MHz)
n: Baud rate generator input clock (n = 0 to 3)
(See the table below for the relation between n and the clock.)

		SCSMR2 Setting	
n	**Clock**	**CKS1**	**CKS0**
0	Pϕ	0	0
1	Pϕ/4	0	1
2	Pϕ/16	1	0
3	Pϕ/64	1	1

The bit rate error in asynchronous mode is found from the following equation:

$$\text{Error (\%)} = \left\{ \frac{P\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

16.2.9 FIFO Control Register (SCFCR2)

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

Bit:	7	6	5	4	3	2	1	0
	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

SCFCR2 performs data count resetting and trigger data number setting for the transmit and receive FIFO registers, and also contains a loopback test enable bit.

SCFCR2 can be read or written to by the CPU at all times.

SCFCR2 is initialized to H'0000 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bits 15 to 8—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 7 and 6—Receive FIFO Data Number Trigger (RTRG1, RTRG0): These bits are used to set the number of receive data bytes that sets the receive data full (RDF) flag in the serial status register (SCFSR2).

The RDF flag is set when the number of receive data bytes in SCFRDR2 is equal to or greater than the trigger set number shown in the following table.

Bit 7: RTRG1	Bit 6: RTRG0	Receive Trigger Number
0	0	1*
	1	4
1	0	8
	1	14

Note: * Initial value

Bits 5 and 4—Transmit FIFO Data Number Trigger (TTRG1, TTRG0): These bits are used to set the number of remaining transmit data bytes that sets the transmit FIFO data register empty (TDFE) flag in the serial status register (SCFSR2). The TDFE flag is set when the number of transmit data bytes in SCFTDR2 is equal to or less than the trigger set number shown in the following table.

Bit 5: TTRG1	Bit 4: TTRG0	Transmit Trigger Number
0	0	8 (8) *
	1	4 (12)
1	0	2 (14)
	1	1 (15)

Note: * Initial value. Figures in parentheses are the number of empty bytes in SCFTDR2 when the flag is set.

Bit 3—Modem Control Enable (MCE): Enables the $\overline{\text{CTS2}}$ and $\overline{\text{RTS2}}$ modem control signals.

Bit 3: MCE	Description	
0	Modem signals disabled*	(Initial value)
1	Modem signals enabled	

Note: * $\overline{\text{CTS2}}$ is fixed at active-0 regardless of the input value, and $\overline{\text{RTS2}}$ output is also fixed at 0.

Bit 2—Transmit FIFO Data Register Reset (TFRST): Invalidates the transmit data in the transmit FIFO data register and resets it to the empty state.

Bit 2: TFRST	Description	
0	Reset operation disabled*	(Initial value)
1	Reset operation enabled	

Note: * A reset operation is performed in the event of a power-on reset or manual reset.

Bit 1—Receive FIFO Data Register Reset (RFRST): Invalidates the receive data in the receive FIFO data register and resets it to the empty state.

Bit 1: RFRST	Description	
0	Reset operation disabled*	(Initial value)
1	Reset operation enabled	

Note: * A reset operation is performed in the event of a power-on reset or manual reset.

Bit 0—Loopback Test (LOOP): Internally connects the transmit output pin (TxD2) and receive input pin (RxD2), and the $\overline{\text{RTS2}}$ pin and $\overline{\text{CTS2}}$ pin, enabling loopback testing.

Bit 0: LOOP	Description	
0	Loopback test disabled	(Initial value)
1	Loopback test enabled	

16.2.10 FIFO Data Count Register (SCFDR2)

SCFDR2 is a 16-bit register that indicates the number of data bytes stored in SCFTDR2 and SCFRDR2.

The upper 8 bits show the number of transmit data bytes in SCFTDR2, and the lower 8 bits show the number of receive data bytes in SCFRDR2.

SCFDR2 can be read by the CPU at all times.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	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

These bits show the number of untransmitted data bytes in SCFTDR2. A value of H'00 indicates that there is no transmit data, and a value of H'10 indicates that SCFTDR2 is full of transmit data.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	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

These bits show the number of receive data bytes in SCFRDR2. A value of H'00 indicates that there is no receive data, and a value of H'10 indicates that SCFRDR2 is full of receive data.

16.2.11 Serial Port Register (SCSPTR2)

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

Bit:	7	6	5	4	3	2	1	0
	RTSIO	RTSDT	CTSIO	CTSDT	—	—	SPB2IO	SPB2DT
Initial value:	0	—	0	—	0	0	0	—
R/W:	R/W	R/W	R/W	R/W	R	R	R/W	R/W

SCSPTR2 is a 16-bit readable/writable register that controls input/output and data for the port pins multiplexed with the serial communication interface (SCIF) pins. Input data can be read from the RxD2 pin, output data written to the TxD2 pin, and breaks in serial transmission/reception controlled, by means of bits 1 and 0. Data can be read from, and output data written to, the $\overline{\text{CTS2}}$ pin by means of bits 5 and 4. Data can be read from, and output data written to, the $\overline{\text{RTS2}}$ pin by means of bits 6 and 7.

SCSPTR2 can be read or written to by the CPU at all times. All SCSPTR2 bits except bits 6, 4, and 0 are initialized to 0 by a power-on reset or manual reset; the value of bits 6, 4, and 0 is undefined. SCSPTR2 is not initialized in standby mode or in the module standby state.

Bits 15 to 8—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 7—Serial Port RTS Port I/O (RTSIO): Specifies the serial port $\overline{\text{RTS2}}$ pin input/output condition. When the $\overline{\text{RTS2}}$ pin is actually set as a port output pin and outputs the value set by the RTSDT bit, the MCE bit in SCFCR2 should be cleared to 0.

Bit 7: RTSIO	Description	
0	RTSDT bit value is not output to $\overline{\text{RTS2}}$ pin	(Initial value)
1	RTSDT bit value is output to $\overline{\text{RTS2}}$ pin	

Bit 6—Serial Port RTS Port Data (RTSDT): Specifies the serial port $\overline{\text{RTS2}}$ pin input/output data. Input or output is specified by the RTSIO bit (see the description of bit 7, RTSIO, for details). In output mode, the RTSDT bit value is output to the $\overline{\text{RTS2}}$ pin. The $\overline{\text{RTS2}}$ pin value is read from the RTSDT bit regardless of the value of the RTSIO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

Bit 6: RTSDT	Description
0	Input/output data is low-level
1	Input/output data is high-level

Bit 5—Serial Port CTS Port I/O (CTSIO): Specifies the serial port $\overline{\text{CTS2}}$ pin input/output condition. When the $\overline{\text{CTS2}}$ pin is actually set as a port output pin and outputs the value set by the CTSDT bit, the MCE bit in SCFCR2 should be cleared to 0.

Bit 5: CTSIO	Description	
0	CTSDT bit value is not output to $\overline{\text{CTS2}}$ pin	(Initial value)
1	CTSDT bit value is output to $\overline{\text{CTS2}}$ pin	

Bit 4—Serial Port CTS Port Data (CTSDT): Specifies the serial port $\overline{\text{CTS2}}$ pin input/output data. Input or output is specified by the CTSIO bit (see the description of bit 5, CTSIO, for details). In output mode, the CTSDT bit value is output to the $\overline{\text{CTS2}}$ pin. The $\overline{\text{CTS2}}$ pin value is read from the CTSDT bit regardless of the value of the CTSIO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

Bit 4: CTSDT	Description
0	Input/output data is low-level
1	Input/output data is high-level

Bits 3 and 2—Reserved: These bits are always read as 0, and should only be written with 0.

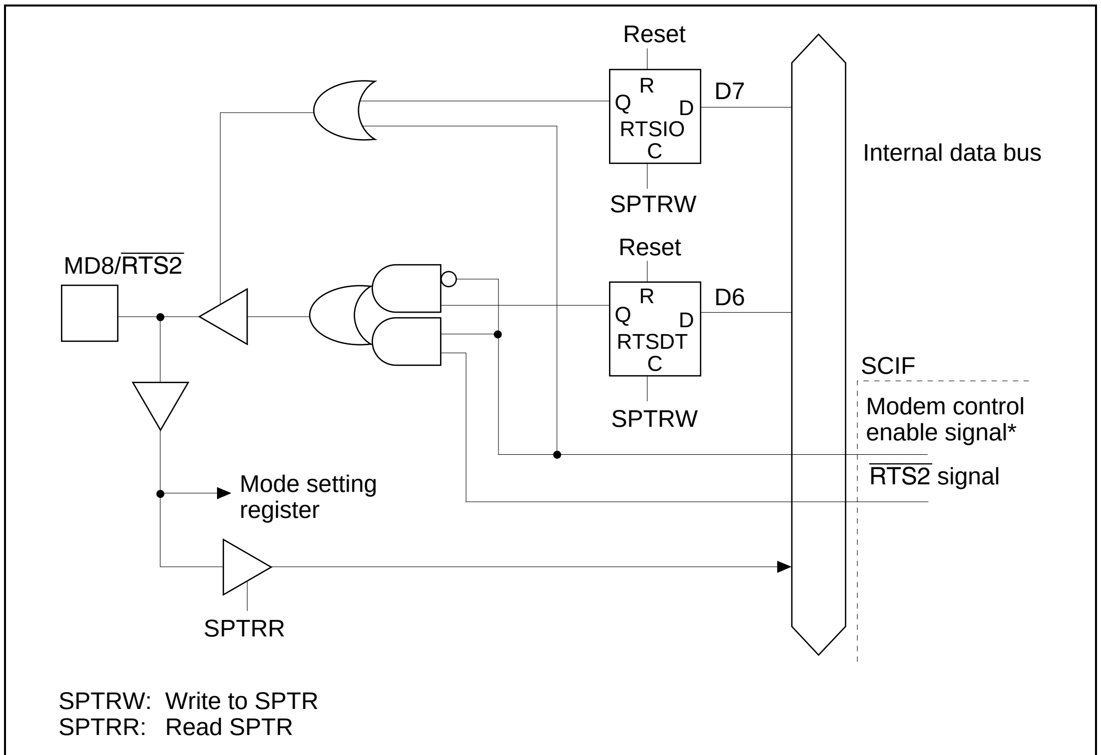
Bit 1—Serial Port Break I/O (SPB2IO): Specifies the serial port TxD2 pin output condition. When the TxD2 pin is actually set as a port output pin and outputs the value set by the SPB2DT bit, the TE bit in SCSCR2 should be cleared to 0.

Bit 1: SPB2IO	Description	
0	SPB2DT bit value is not output to the TxD2 pin	(Initial value)
1	SPB2DT bit value is output to the TxD2 pin	

Bit 0—Serial Port Break Data (SPB2DT): Specifies the serial port RxD2 pin input data and TxD2 pin output data. The TxD2 pin output condition is specified by the SPB2IO bit (see the description of bit 1, SPB2IO, for details). When the TxD2 pin is designated as an output, the value of the SPB2DT bit is output to the TxD2 pin. The RxD2 pin value is read from the SPB2DT bit regardless of the value of the SPB2IO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

Bit 0: SPB2DT	Description
0	Input/output data is low-level
1	Input/output data is high-level

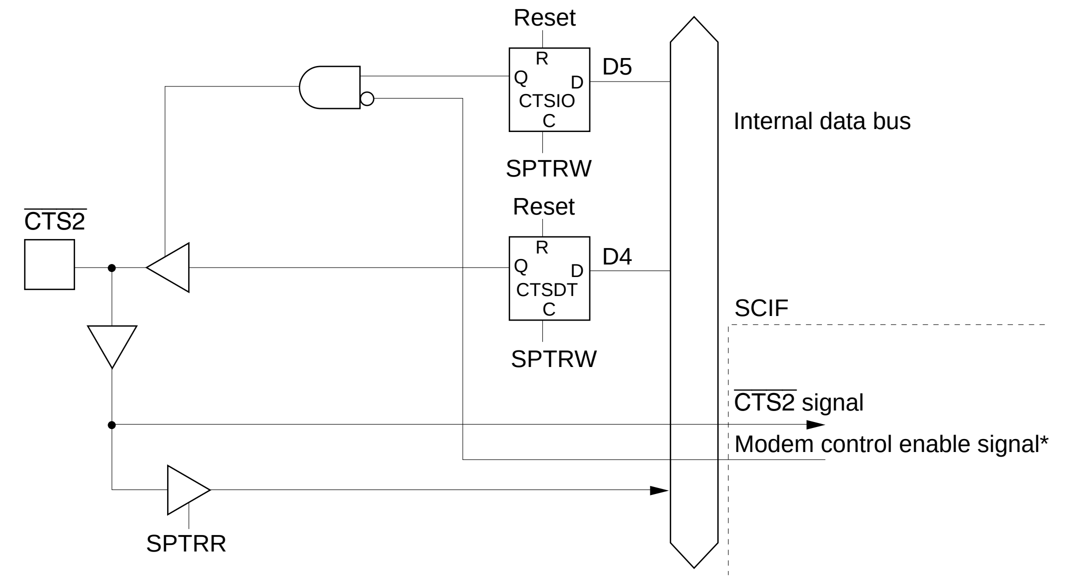
SCIF I/O port block diagrams are shown in figures 16.2 to 16.5.



SPTRW: Write to SPTR
SPTRR: Read SPTR

Note: * The $\overline{RTS2}$ pin function is designated as modem control by the MCE bit in SCFCR2.

Figure 16.2 MD8/$\overline{RTS2}$ Pin



SPTRW: Write to SPTR
SPTRR: Read SPTR

Note: * The $\overline{\text{CTS2}}$ pin function is designated as modem control by the MCE bit in SCFCR2.

Figure 16.3 $\overline{\text{CTS2}}$ Pin

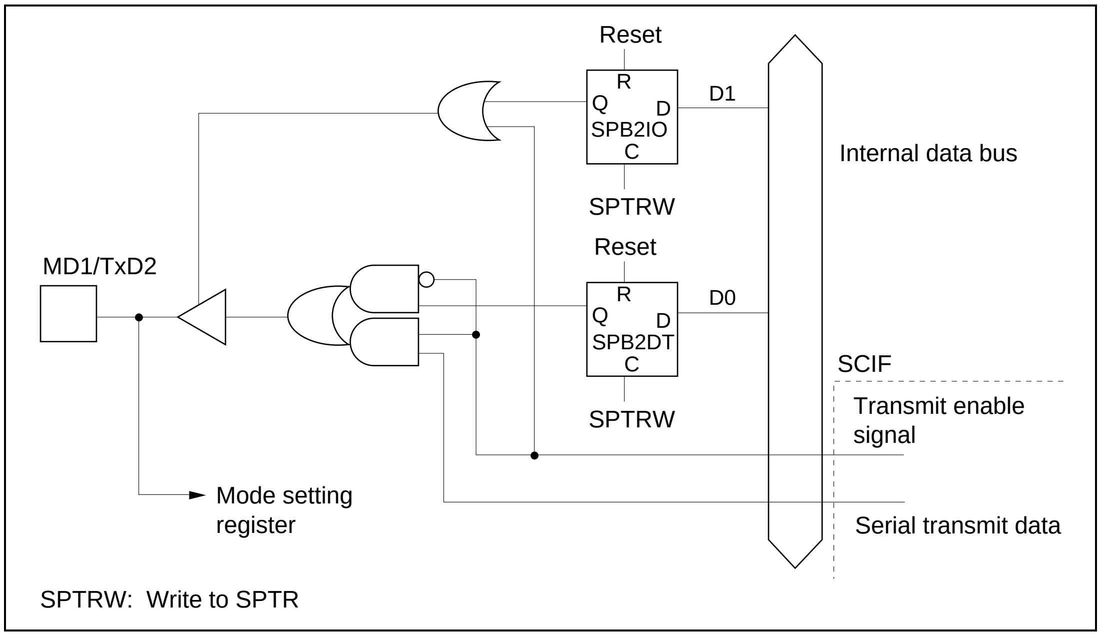


Figure 16.4 MD1/TxD2 Pin

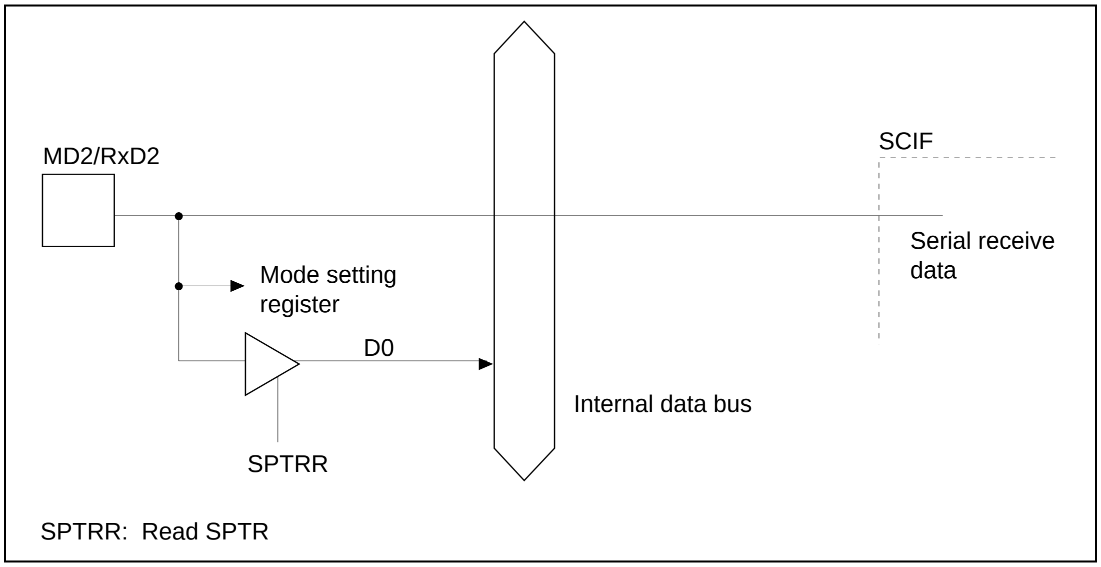


Figure 16.5 MD2/RxD2 Pin

16.2.12 Line Status Register (SCLSR2)

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

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	ORER
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	(R/W)*

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

Bits 15 to 1—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 0—Overrun Error (ORER): Indicates that an overrun error occurred during reception, causing abnormal termination.

Bit 0: ORER	Description
0	Reception in progress, or reception has ended normally*[1] (Initial value) [Clearing conditions] • Power-on reset or manual reset • When 0 is written to ORER after reading ORER = 1
1	An overrun error occurred during reception*[2] [Setting condition] When the next serial reception is completed while the receive FIFO is full

Notes: 1. The ORER flag is not affected and retains its previous state when the RE bit in SCSCR2 is cleared to 0.

2. The receive data prior to the overrun error is retained in SCFRDR2, and the data received subsequently is lost. Serial reception cannot be continued while the ORER flag is set to 1.

16.3 Operation

16.3.1 Overview

The SCIF can carry out serial communication in asynchronous mode, in which synchronization is achieved character by character. See section 15.3.2, Operation in Asynchronous Mode, for details.

Sixteen-stage FIFO buffers are provided for both transmission and reception, reducing the CPU overhead and enabling fast, continuous communication to be performed. $\overline{\text{RTS2}}$ and $\overline{\text{CTS2}}$ signals are also provided as modem control signals.

The transmission format is selected using the serial mode register (SCSMR2), as shown in table 16.3. The SCIF clock source is determined by the CKE1 bit in the serial control register (SCSCR2), as shown in table 16.4.

- Data length: Choice of 7 or 8 bits
- Choice of parity addition and addition of 1 or 2 stop bits (the combination of these parameters determines the transfer format and character length)
- Detection of framing errors, parity errors, receive-FIFO-data-full state, overrun errors, receive-data-ready state, and breaks, during reception
- Indication of the number of data bytes stored in the transmit and receive FIFO registers
- Choice of internal or external clock as SCIF clock source

  When internal clock is selected: The SCIF operates on the baud rate generator clock.

  When external clock is selected: A clock with a frequency of 16 times the bit rate must be input (the on-chip baud rate generator is not used).

Table 16.3 SCSMR2 Settings for Serial Transfer Format Selection

SCSMR2 Settings				SCIF Transfer Format			
Bit 6: CHR	**Bit 5: PE**	**Bit 3: STOP**	**Mode**	**Data Length**	**Multiprocessor Bit**	**Parity Bit**	**Stop Bit Length**
0	0	0	Asynchronous mode	8-bit data	No	No	1 bit
		1					2 bits
	1	0				Yes	1 bit
		1					2 bits
1	0	0		7-bit data		No	1 bit
		1					2 bits
	1	0				Yes	1 bit
		1					2 bits

Table 16.4 SCSCR2 Settings for SCIF Clock Source Selection

SCSCR2 Setting		SCIF Transmit/Receive Clock	
Bit 1: CKE1	**Mode**	**Clock Source**	**SCK2 Pin Function**
0	Asynchronous mode	Internal	SCIF does not use SCK2 pin
1		External	Inputs clock with frequency of 16 times the bit rate

16.3.2 Serial Operation

Data Transfer Format

Table 16.5 shows the data transfer formats that can be used. Any of 8 transfer formats can be selected according to the SCSMR2 settings.

Table 16.5 Serial Transfer Formats

SCSMR2 Settings			Serial Transfer Format and Frame Length											
CHR	**PE**	**STOP**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
0	0	0	S	8-bit data								STOP		
0	0	1	S	8-bit data								STOP	STOP	
0	1	0	S	8-bit data								P	STOP	
0	1	1	S	8-bit data								P	STOP	STOP
1	0	0	S	7-bit data							STOP			
1	0	1	S	7-bit data							STOP	STOP		
1	1	0	S	7-bit data							P	STOP		
1	1	1	S	7-bit data							P	STOP	STOP	

S:	Start bit
STOP:	Stop bit
P:	Parity bit

Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input at the SCK2 pin can be selected as the SCIF's serial clock, according to the setting of the CKE1 bit in SCSCR2. For details of SCIF clock source selection, see table 16.4.

When an external clock is input at the SCK2 pin, the clock frequency should be 16 times the bit rate used.

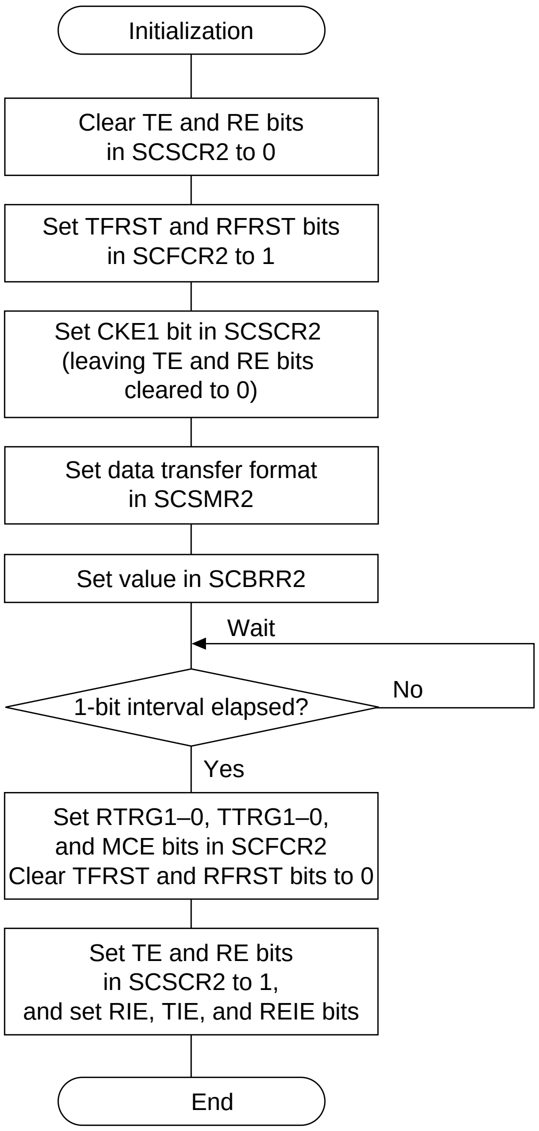
Data Transfer Operations

SCIF Initialization: Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR2 to 0, then initialize the SCIF as described below.

When the transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, SCTSR2 is initialized. Note that clearing the TE and RE bits to 0 does not change the contents of SCFSR2, SCFTDR2, or SCFRDR2. The TE bit should be cleared to 0 after all transmit data has been sent and the TEND flag in SCFSR2 has been set. TEND can also be cleared to 0 during transmission, but the data being transmitted will go to the mark state after the clearance. Before setting TE again to start transmission, the TFRST bit in SCFCR2 should first be set to 1 to reset SCFTDR2.

When an external clock is used the clock should not be stopped during operation, including initialization, since operation will be unreliable in this case.

Figure 16.6 shows a sample SCIF initialization flowchart.


Initialization
Clear TE and RE bits in SCSCR2 to 0
Set TFRST and RFRST bits in SCFCR2 to 1
Set CKE1 bit in SCSCR2 (leaving TE and RE bits cleared to 0)
Set data transfer format in SCSMR2
Set value in SCBRR2
Wait
1-bit interval elapsed?
No
Yes
Set RTRG1–0, TTRG1–0, and MCE bits in SCFCR2 Clear TFRST and RFRST bits to 0
Set TE and RE bits in SCSCR2 to 1, and set RIE, TIE, and REIE bits
End

1. Set the clock selection in SCSCR2.

   Be sure to clear bits RIE and TIE, and bits TE and RE, to 0.

2. Set the data transfer format in SCSMR2.

3. Write a value corresponding to the bit rate into SCBRR2. (Not necessary if an external clock is used.)

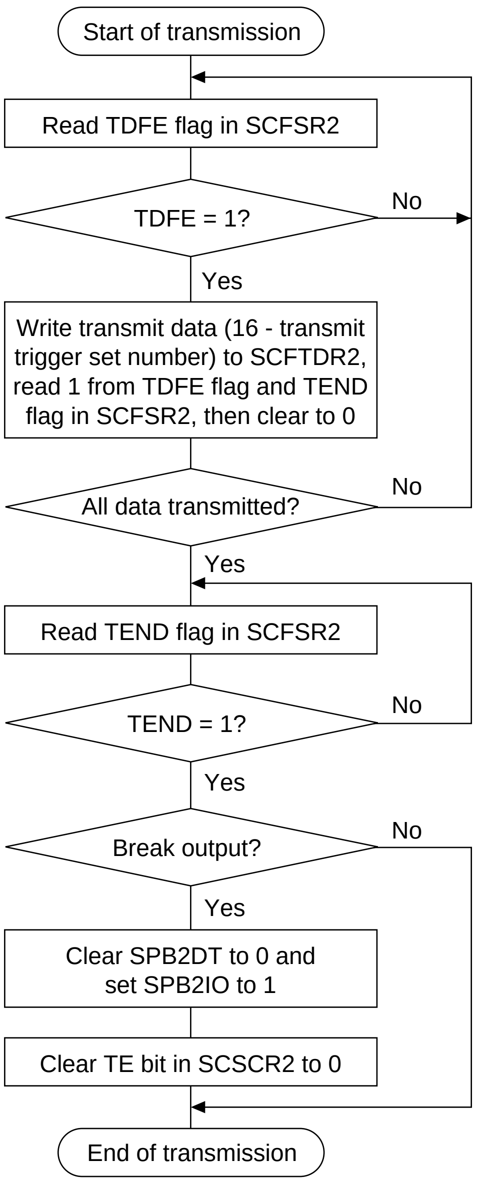
4. Wait at least one bit interval, then set the TE bit or RE bit in SCSCR2 to 1. Also set the RIE, REIE, and TIE bits.

   Setting the TE and RE bits enables the TxD2 and RxD2 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 16.6 Sample SCIF Initialization Flowchart

Serial Data Transmission: Figure 16.7 shows a sample flowchart for serial transmission.

Use the following procedure for serial data transmission after enabling the SCIF for transmission.



Start of transmission
Read TDFE flag in SCFSR2
TDFE = 1?
No
Yes
Write transmit data (16 - transmit trigger set number) to SCFTDR2, read 1 from TDFE flag and TEND flag in SCFSR2, then clear to 0
All data transmitted?
No
Yes
Read TEND flag in SCFSR2
TEND = 1?
No
Yes
Break output?
No
Yes
Clear SPB2DT to 0 and set SPB2IO to 1
Clear TE bit in SCSCR2 to 0
End of transmission

1. SCIF status check and transmit data write:

   Read SCFSR2 and check that the TDFE flag is set to 1, then write transmit data to SCFTDR2, 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 SCFTDR2, and then clear the TDFE flag to 0.

3. Break output at the end of serial transmission:

   To output a break in serial transmission, clear the SPB2DT bit to 0 and set the SPB2IO bit to 1 in SCSPTR2, then clear the TE bit in SCSCR2 to 0.

   In steps 1 and 2, it is possible to ascertain the number of data bytes that can be written from the number of transmit data bytes in SCFTDR2 indicated by the upper 8 bits of SCFDR2.

Figure 16.7 Sample Serial Transmission Flowchart

In serial transmission, the SCIF operates as described below.

1. When data is written into SCFTDR2, the SCIF transfers the data from SCFTDR2 to SCTSR2 and starts transmitting. Confirm that the TDFE flag in the serial status register (SCFSR2) is set to 1 before writing transmit data to SCFTDR2. The number of data bytes that can be written is at least (16 - transmit trigger setting).
2. When data is transferred from SCFTDR2 to SCTSR2 and transmission is started, consecutive transmit operations are performed until there is no transmit data left in SCFTDR2. When the number of transmit data bytes in SCFTDR2 falls to or below the transmit trigger number set in the FIFO control register (SCFCR2), the TDFE flag is set. If the TIE bit in SCSCR2 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 TxD2 pin in the following order.

   a. Start bit: One 0-bit is output.
   b. Transmit data: 8-bit or 7-bit data is output in LSB-first order.
   c. Parity bit: One parity bit (even or odd parity) is output. (A format in which a parity bit is not output can also be selected.)
   d. Stop bit(s): One or two 1-bits (stop bits) are output.
   e. Mark state: 1 is output continuously until the start bit that starts the next transmission is sent.
3. The SCIF checks the SCFTDR2 transmit data at the timing for sending the stop bit. If data is present, the data is transferred from SCFTDR2 to SCTSR2, 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 SCFSR2 is set to 1, the stop bit is sent, and then the line goes to the mark state in which 1 is output.

Figure 16.8 shows an example of the operation for transmission in asynchronous mode.

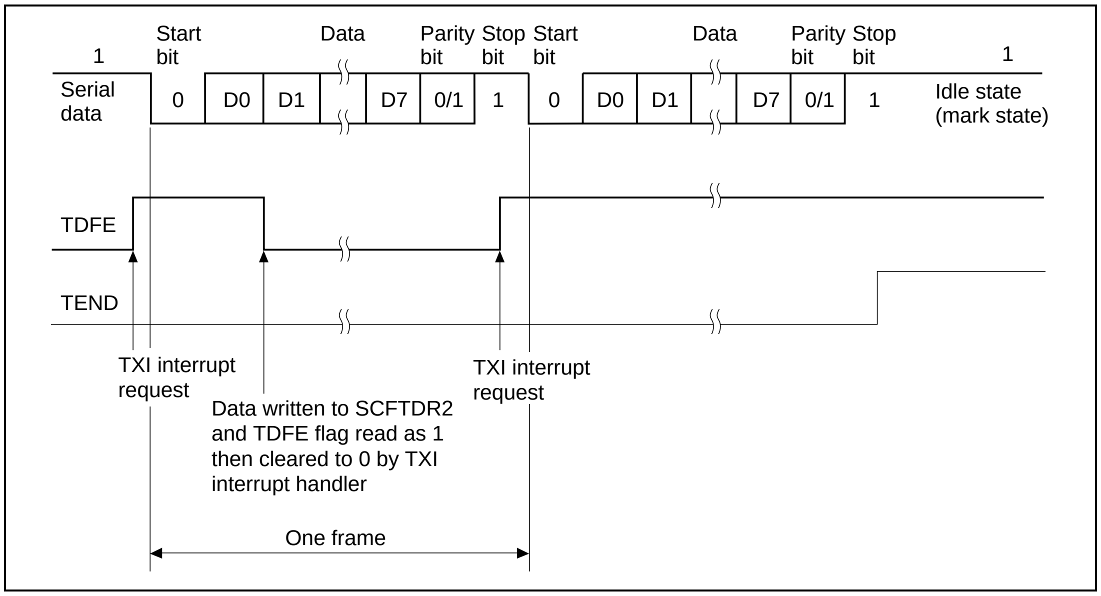


Figure 16.8 Example of Transmit Operation (Example with 8-Bit Data, Parity, One Stop Bit)

4. When modem control is enabled, transmission can be stopped and restarted in accordance with the $\overline{\text{CTS2}}$ input value. When $\overline{\text{CTS2}}$ is set to 1, if transmission is in progress, the line goes to the mark state after transmission of one frame. When $\overline{\text{CTS2}}$ is set to 0, the next transmit data is output starting from the start bit.

   Figure 16.9 shows an example of the operation when modem control is used.

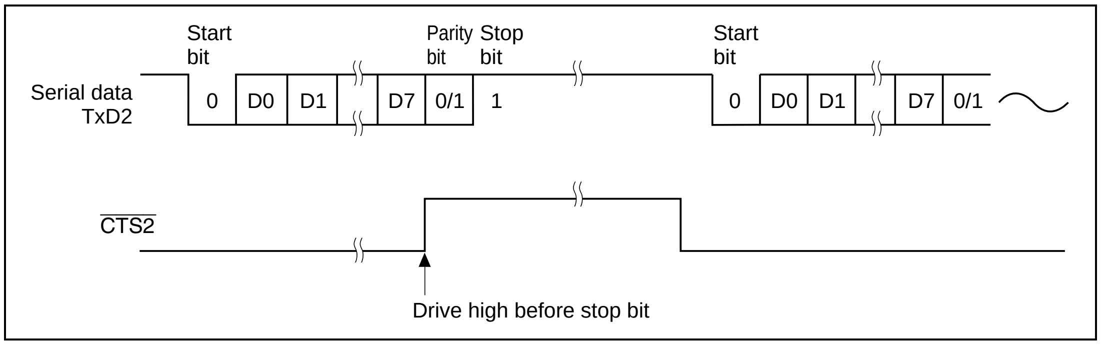
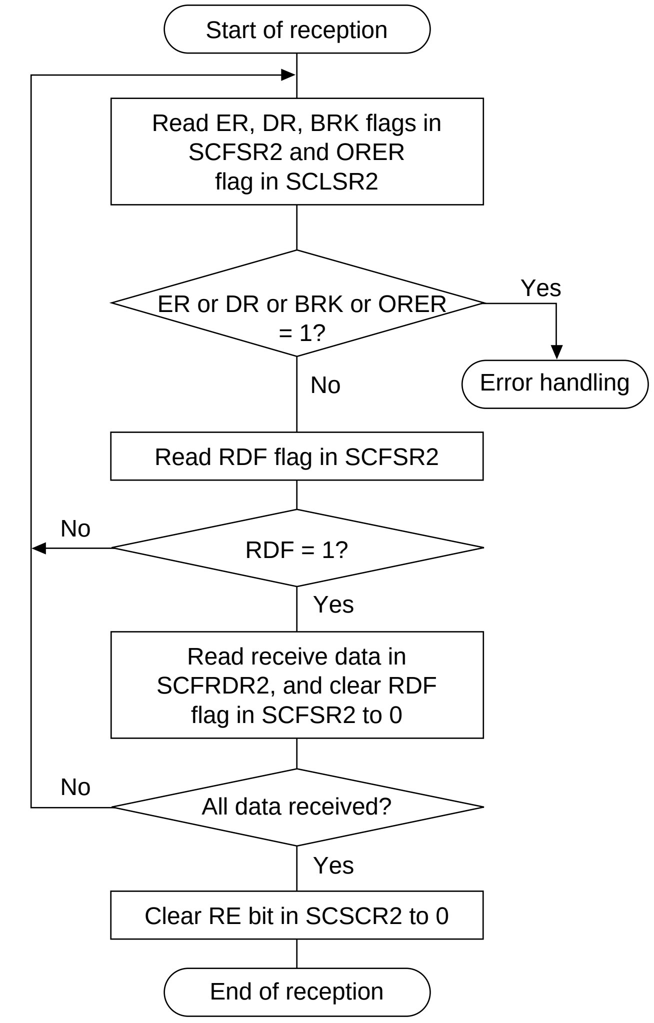


Figure 16.9 Example of Operation Using Modem Control ($\overline{\text{CTS2}}$)

Serial Data Reception: Figure 16.10 shows a sample flowchart for serial reception.

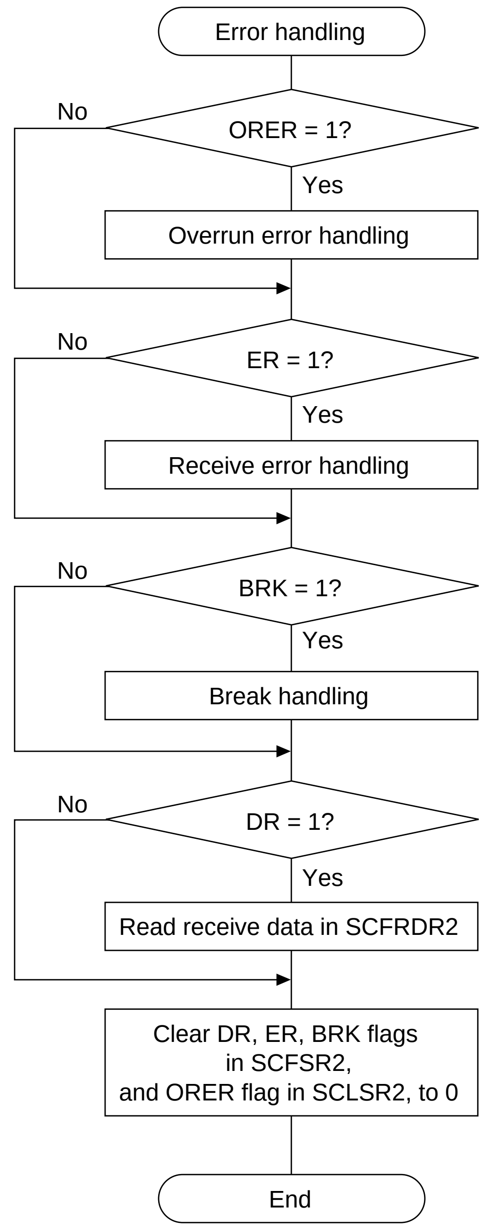
Use the following procedure for serial data reception after enabling the SCIF for reception.



Start of reception
Read ER, DR, BRK flags in SCFSR2 and ORER flag in SCLSR2
ER or DR or BRK or ORER = 1?
Yes
Error handling
No
Read RDF flag in SCFSR2
No
RDF = 1?
Yes
Read receive data in SCFRDR2, and clear RDF flag in SCFSR2 to 0
No
All data received?
Yes
Clear RE bit in SCSCR2 to 0
End of reception

1. Receive error handling and break detection: Read the DR, ER, and BRK flags in SCFSR2, and the ORER flag in SCLSR2, to identify any error, perform the appropriate error handling, then clear the DR, ER, BRK, and ORER flags to 0. In the case of a framing error, a break can also be detected by reading the value of the RxD2 pin.
2. SCIF status check and receive data read : Read SCFSR2 and check that RDF = 1, then read the receive data in SCFRDR2, read 1 from the RDF flag, and then clear the RDF flag to 0. The transition of the RDF flag from 0 to 1 can also be identified by an RXI interrupt.
3. Serial reception continuation procedure: To continue serial reception, read at least the receive trigger set number of receive data bytes from SCFRDR2, read 1 from the RDF flag, then clear the RDF flag to 0. The number of receive data bytes in SCFRDR2 can be ascertained by reading the lower bits of SCFDR2.

Figure 16.10 Sample Serial Reception Flowchart (1)



1. Whether a framing error or parity error has occurred in the receive data read from SCFRDR2 can be ascertained from the FER and PER bits in SCFSR2.
2. When a break signal is received, receive data is not transferred to SCFRDR2 while the BRK flag is set. However, note that the last data in SCFRDR2 is H'00 (the break data in which a framing error occurred is stored).

Figure 16.10 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 SCRSR2 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 (SCRSR2) to SCFRDR2.
   c. Overrun error check: The SCIF checks that the ORER flag is 0, indicating that no overrun error has occurred.
   d. 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 SCFRDR2.

   Note: Reception continues when parity error, framing error occurs.

4. If the RIE bit in SCSCR2 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 or REIE bit in SCSCR2 is set to 1 when the ER flag changes to 1, a receive-error interrupt (ERI) request is generated.

   If the RIE bit or REIE bit in SCSCR2 is set to 1 when the BRK or ORER flag changes to 1, a break reception interrupt (BRI) request is generated.

Figure 16.11 shows an example of the operation for reception.

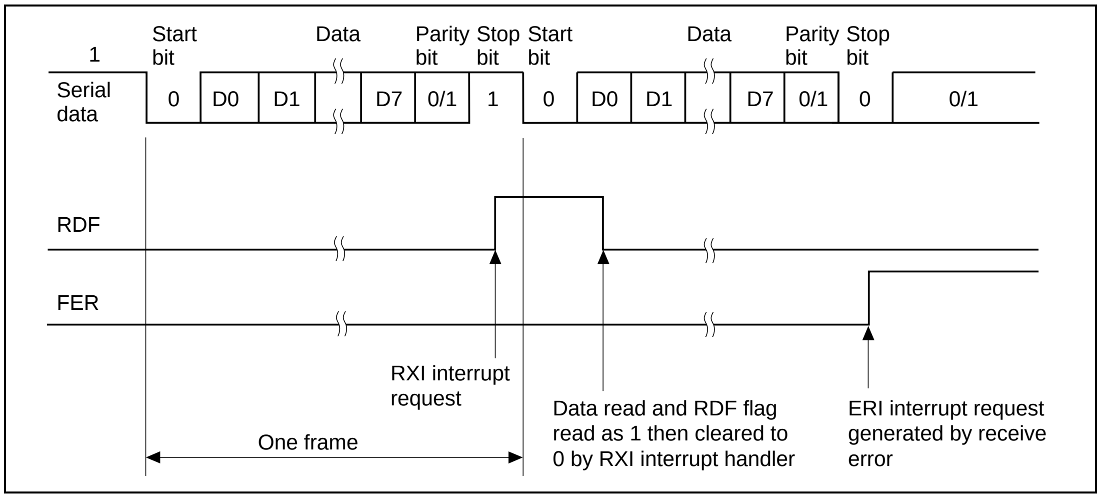


Figure 16.11 Example of SCIF Receive Operation (Example with 8-Bit Data, Parity, One Stop Bit)

5. When modem control is enabled, the $\overline{\text{RTS2}}$ signal is output when SCFRDR2 is empty. When $\overline{\text{RTS2}}$ is 0, reception is possible. When $\overline{\text{RTS2}}$ is 1, this indicates that SCFRDR2 contains 15 or more bytes of data, and there is no free space, reception is not possible.

   Figure 16.12 shows an example of the operation when modem control is used.

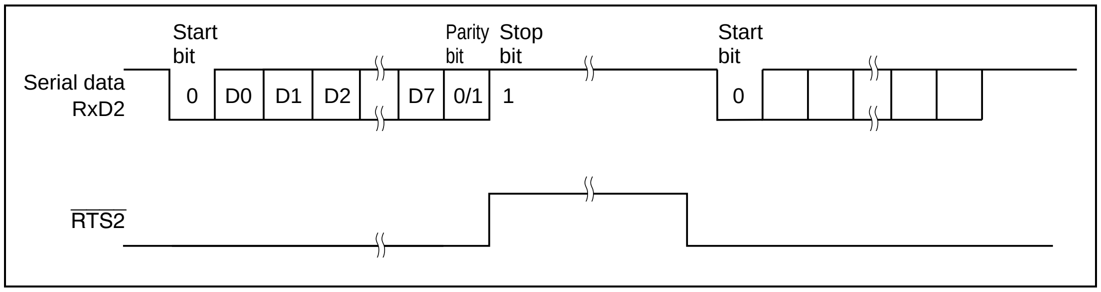


Figure 16.12 Example of Operation Using Modem Control ($\overline{\text{RTS2}}$)

HITACHI

16.4 SCIF Interrupt Sources and the DMAC

The SCIF has four interrupt sources: transmit-FIFO-data-empty interrupt (TXI) request, receive-error interrupt (ERI) request, receive-FIFO-data-full interrupt (RXI) request, and break interrupt (BRI) request.

Table 16.6 shows the interrupt sources and their order of priority. The interrupt sources are enabled or disabled by means of the TIE, RIE, and REIE bits in SCSCR2. A separate interrupt request is sent to the interrupt controller for each of these interrupt sources.

When transmission/reception is carried out using the DMAC, output of interrupt requests to the interrupt controller can be inhibited by clearing the RIE bit in SCSCR2 to 0. By setting the REIE bit to 1 while the RIE bit is cleared to 0, it is possible to output ERI and BRI interrupt requests, but not RXI interrupt requests.

When the TDFE flag in the serial status register (SCFSR2) is set to 1, a transmit-FIFO-data-empty request is generated separately from the interrupt request. A transmit-FIFO-data-empty request can activate the DMAC to perform data transfer.

When the RDF flag or DR flag in SCFSR2 is set to 1, a receive-FIFO-data-full request is generated separately from the interrupt request. A receive-FIFO-data-full request can activate the DMAC to perform data transfer.

When using the DMAC for transmission/reception, set and enable the DMAC before making the SCIF settings. See section 14, Direct Memory Access Controller (DMAC), for details of the DMAC setting procedure.

When the BRK flag in SCFSR2 or the ORER flag in the line status register (SCLSR2) 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 SCFRDR2.

Table 16.6 SCIF 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 FIFO data full flag (RDF) or receive data ready flag (DR)	Possible	↑
BRI	Interrupt initiated by break flag (BRK) or overrun error flag (ORER)	Not possible	↓
TXI	Interrupt initiated by transmit FIFO data empty flag (TDFE)	Possible	Low

See section 5, Exceptions, for priorities and the relationship with non-SCIF interrupts.

16.5 Usage Notes

Note the following when using the SCIF.

SCFTDR2 Writing and the TDFE Flag: The TDFE flag in the serial status register (SCFSR2) is set when the number of transmit data bytes written in the transmit FIFO data register (SCFTDR2) has fallen to or below the transmit trigger number set by bits TTRG1 and TTRG0 in the FIFO control register (SCFCR2). After TDFE is set, transmit data up to the number of empty bytes in SCFTDR2 can be written, allowing efficient continuous transmission.

However, if the number of data bytes written in SCFTDR2 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 SCFTDR2 contains more than the transmit trigger number of transmit data bytes.

The number of transmit data bytes in SCFTDR2 can be found from the upper 8 bits of the FIFO data count register (SCFDR2).

SCFRDR2 Reading and the RDF Flag: The RDF flag in the serial status register (SCFSR2) is set when the number of receive data bytes in the receive FIFO data register (SCFRDR2) has become equal to or greater than the receive trigger number set by bits RTRG1 and RTRG0 in the FIFO control register (SCFCR2). After RDF is set, receive data equivalent to the trigger number can be read from SCFRDR2, allowing efficient continuous reception.

However, if the number of data bytes in SCFRDR2 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 SCFRDR2 can be found from the lower 8 bits of the FIFO data count register (SCFDR2).

Break Detection and Processing: Break signals can be detected by reading the RxD2 pin directly when a framing error (FER) is detected. In the break state the input from the RxD2 pin consists of all 0s, so the FER flag is set and the parity error flag (PER) may also be set.

Although the SCIF stops transferring receive data to SCFRDR2 after receiving a break, the receive operation continues.

Sending a Break Signal: The input/output condition and level of the TxD2 pin are determined by bits SPB2IO and SPB2DT in the serial port register (SCSPTR2). This feature can be used to send a break signal.

After the serial transmitter is initialized, the TxD2 pin function is not selected and the value of the SPB2DT bit substitutes for the mark state until the TE bit is set to 1 (i.e. transmission is enabled). The SPB2IO and SPB2DT bits should therefore be set to 1 (designating output and high level) beforehand.

To send a break signal during serial transmission, clear the SPB2DT 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 its current state, and 0 is output from the TxD2 pin.

Receive Data Sampling Timing and Receive Margin: The SCIF operates on a base clock with a frequency of 16 times the bit 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 16.13.

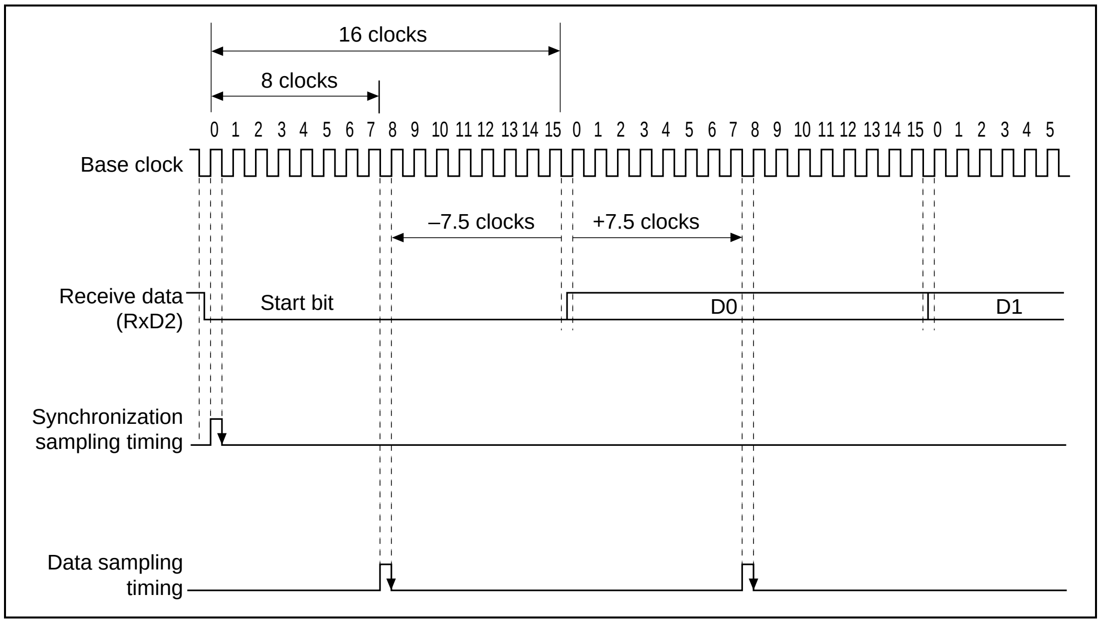


Figure 16.13   Receive Data Sampling Timing in Asynchronous Mode

The receive margin in asynchronous mode can therefore be expressed as shown in equation (1).

$$M = \left| (0.5 - \frac{1}{2N}) - (L - 0.5) F - \frac{| D - 0.5 |}{N} (1 + F) \right| \times 100\% \text{ .... } (1)$$

M:	Receive margin (%)
N:	Ratio of clock frequency to bit rate (N = 16)
D:	Clock duty cycle (D = 0 to 1.0)
L:	Frame length (L = 9 to 12)
F:	Absolute deviation of clock frequency

From equation (1), if F = 0 and D = 0.5, the receive margin is 46.875%, as given by equation (2).

When D = 0.5 and F = 0:

$$M = (0.5 - 1 / (2 \times 16)) \times 100\% = 46.875\% \text{ .... } (2)$$

This is a theoretical value. A reasonable margin to allow in system designs is 20% to 30%.

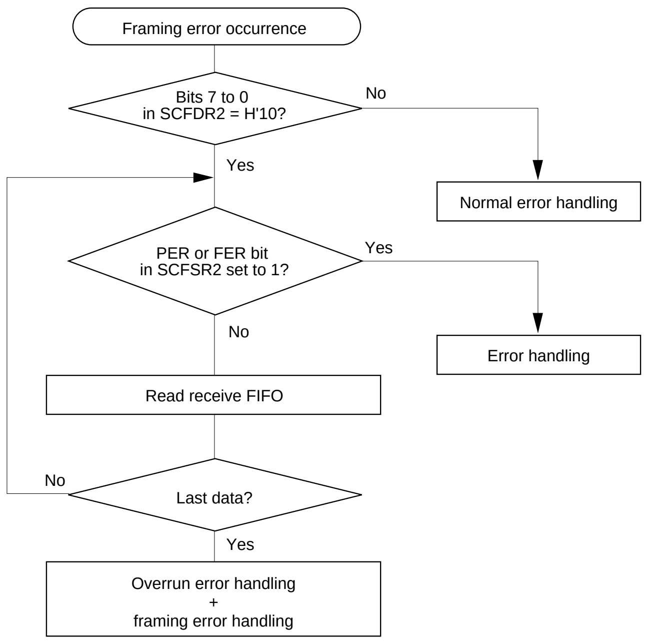
SCK2/MRESET: As the manual reset pin is multiplexed with the SCK2 pin, a manual reset must not be executed while the SCIF is operating in external clock mode.

When Using the DMAC: When using the DMAC for transmission/reception, inhibit output of RXI and TXI interrupt requests to the interrupt controller. If interrupt request output is enabled, interrupt requests to the interrupt controller will be cleared by the DMAC without regard to the interrupt handler.

Serial Ports: Note that, when the SCIF pin value is read using a serial port, the value read will be the value two peripheral clock cycles earlier.

Overrun error flag: SCIF overrun error flag is not set in the case that overrun error and flaming error occurred simultaneously in receiving data, that means 17th byte data which overrun was accompanying with flaming error. In such case, only SCFSR2. ER flag which shows occurrence of flaming error is set. RxFIFO stores data received before the overrun and does not store (i. e. lose) overrun data. SCIF has no bit which corresponds to SCFSR2. FER for the lost data.

In addition to the overrun error handling software routine, exception handler should check co-occurrence of overrun error when a flaming error is occurred and when a co-occurrence is found, it should handle also overrun error (When (i) a overrun error solely occurred without accompanying with other receive error and (ii) when a parity error is accompanied with overrun error, usual overrun error handling can be used. Overrun error handling should rather be done primarily).


Framing error occurrence
Bits 7 to 0 in SCFDR2 = H'10?
No
Yes
Normal error handling
PER or FER bit in SCFSR2 set to 1?
Yes
No
Error handling
Read receive FIFO
No
Last data?
Yes
Overrun error handling + framing error handling

Flow chart:

When flaming error (SCFSR. ER=1) is occurred, bit7 to bit0 should be read out from SCFDR2. If bit7 to bit0 equals H'10, contents of the RxFIFO should be read. When the data received last is not accompanied with flaming error (SCFSR2. FER=0) both overrun error handling and flaming error handling shoud be conducted.

Figure 16.14 Overrun Error Flag

Section 17 Smart Card Interface

17.1 Overview

An IC card (smart card) interface conforming to ISO/IEC 7816-3 (Identification Card) is supported as a serial communication interface (SCI) extension function.

Switching between the normal serial communication interface and the smart card interface is carried out by means of a register setting.

17.1.1 Features

Features of the smart card interface are listed below.

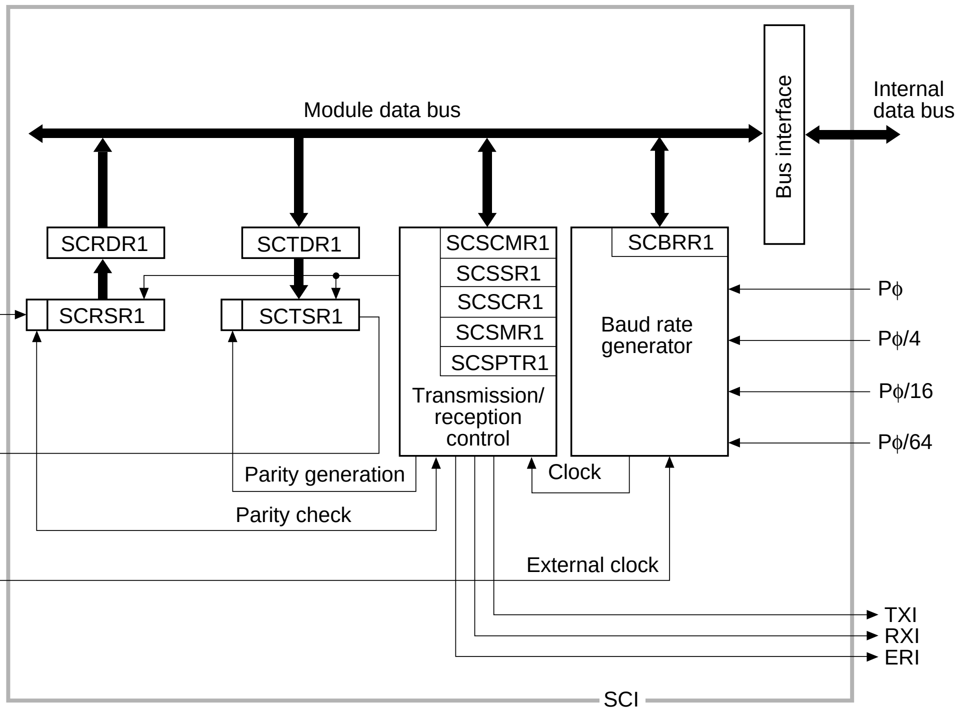
- Asynchronous mode
  - Data length: 8 bits
  - Parity bit generation and checking
  - Transmission of error signal (parity error) in receive mode
  - Error signal detection and automatic data retransmission in transmit mode
  - Direct convention and inverse convention both supported
- On-chip baud rate generator allows any bit rate to be selected
- Three interrupt sources

  There are three interrupt sources—transmit-data-empty, receive-data-full, and transmit/receive error—that can issue requests independently.

  The transmit-data-empty interrupt and receive-data-full interrupt can activate the DMA controller (DMAC) to execute data transfer.

17.1.2 Block Diagram

Figure 17.1 shows a block diagram of the smart card interface.



SCSCMR1:	Smart card mode register
SCRSR1:	Receive shift register
SCRDR1:	Receive data register
SCTSR1:	Transmit shift register
SCTDR1:	Transmit data register
SCSMR1:	Serial mode register
SCSCR1:	Serial control register
SCSSR1:	Serial status register
SCBRR1:	Bit rate register
SCSPTR1:	Serial port register

Figure 17.1 Block Diagram of Smart Card Interface

17.1.3 Pin Configuration

Table 17.1 shows the smart card interface pin configuration.

Table 17.1 Smart Card Interface Pins

Pin Name	Abbreviation	I/O	Function
Serial clock pin	MD0/SCK	I/O	Clock input/output
Receive data pin	RxD	Input	Receive data input
Transmit data pin	MD7/TxD	Output	Transmit data output

17.1.4 Register Configuration

The smart card interface has the internal registers shown in table 17.2. Details of the SCBRR1, SCTDR1, SCRDR1, and SCSPTR1 registers are the same as for the normal SCI function: see the register descriptions in section 15, Serial Communication Interface.

With the exception of the serial port register, the smart card interface registers are initialized in standby mode and in the module standby state as well as by a power-on reset or manual reset. When recovering from standby mode or the module standby state, the registers must be set again.

Table 17.2 Smart Card Interface Registers

Name	Abbreviation	R/W	Initial Value	P4 Address	Area 7 Address	Access Size
Serial mode register	SCSMR1	R/W	H'00	H'FFE00000	H'1FE00000	8
Bit rate register	SCBRR1	R/W	H'FF	H'FFE00004	H'1FE00004	8
Serial control register	SCSCR1	R/W	H'00	H'FFE00008	H'1FE00008	8
Transmit data register	SCTDR1	R/W	H'FF	H'FFE0000C	H'1FE0000C	8
Serial status register	SCSSR1	R/(W)*1	H'84	H'FFE00010	H'1FE00010	8
Receive data register	SCRDR1	R	H'00	H'FFE00014	H'1FE00014	8
Smart card mode register	SCSCMR1	R/W	H'00	H'FFE00018	H'1FE00018	8
Serial port register	SCSPTR1	R/W	H'00*2	H'FFE0001C	H'1FE0001C	8

Notes: 1. Only 0 can be written, to clear flags.
2. The value of bits 2 and 0 is undefined.

17.2 Register Descriptions

Only registers that have been added, and bit functions that have been modified, for the smart card interface are described here.

17.2.1 Smart Card Mode Register (SCSCMR1)

SCSCMR1 is an 8-bit readable/writable register that selects the smart card interface function. SCSCMR1 is initialized to H'00 by a power-on reset or manual reset, in standby mode, and in the module standby state.

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

Bits 7 to 4 and 1—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 3—Smart Card Data Transfer Direction (SDIR): Selects the serial/parallel conversion format.

Bit 3: SDIR	Description	
0	SCTDR1 contents are transmitted LSB-first Receive data is stored in SCRDR1 LSB-first	(Initial value)
1	SCTDR1 contents are transmitted MSB-first Receive data is stored in SCRDR1 MSB-first	

Bit 2—Smart Card Data Invert (SINV): Specifies inversion of the data logic level. This function is used together with the bit 3 function for communication with an inverse convention card. The SINV bit does not affect the logic level of the parity bit. For parity-related setting procedures, see section 17.3.4, Register Settings.

Bit 2: SINV	Description	
0	SCTDR1 contents are transmitted as they are Receive data is stored in SCRDR1 as it is	(Initial value)
1	SCTDR1 contents are inverted before being transmitted Receive data is stored in SCRDR1 in inverted form	

Bit 0—Smart Card Interface Mode Select (SMIF): Enables or disables the smart card interface function.

Bit 0: SMIF	Description	
0	Smart card interface function is disabled	(Initial value)
1	Smart card interface function is enabled	

17.2.2 Serial Mode Register (SCSMR1)

Bit 7 of SCSMR1 has a different function in smart card interface mode.

Bit:	7	6	5	4	3	2	1	0
	GM(C/$\overline{\text{A}}$)	CHR	PE	O/$\overline{\text{E}}$	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—GSM Mode (GM): Sets the smart card interface function to GSM mode.

With the normal smart card interface, this bit is cleared to 0. Setting this bit to 1 selects GSM mode, an additional mode for controlling the timing for setting the TEND flag that indicates completion of transmission, and the type of clock output used. The details of the additional clock output control mode are specified by the CKE1 and CKE0 bits in the serial control register (SCSCR1). In GSM mode, the pulse width is guaranteed when SCK start/stop specifications are made by CKE1 and CKE0.

Bit 7: GM	Description	
0	Normal smart card interface mode operation • The TEND flag is set 12.5 etu after the beginning of the start bit • Clock output on/off control only	(Initial value)
1	GSM mode smart card interface mode operation • The TEND flag is set 11.0 etu after the beginning of the start bit • Clock output on/off and fixed-high/fixed-low control (set in SCSCR1)	

Note: etu: Elementary time unit (time for transfer of 1 bit)

Bits 6 to 0: Operate in the same way as for the normal SCI. See section 15, Serial Communication Interface, for details. With the smart card interface, the following settings should be used: CHR = 0, PE = 1, STOP = 1, MP = 0.

17.2.3 Serial Control Register (SCSCR1)

Bits 1 and 0 of SCSCR1 have a different function in smart card interface mode.

Bit:	7	6	5	4	3	2	1	0
	TIE	RIE	TE	RE	—	—	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

Bits 7 to 4: Operate in the same way as for the normal SCI. See section 15, Serial Communication Interface, for details.

Bits 3 and 2: Not used with the smart card interface.

Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0): These bits specify the function of the SCK pin. In smart card interface mode, an internal clock is always used as the clock source. In smart card interface mode, it is possible to specify a fixed high level or fixed low level for the clock output, in addition to the usual switching between enabling and disabling of the clock output.

GM	CKE1	CKE0	SCK Pin Function
0	0	0	Port I/O pin
		1	Clock output as SCK output pin
	1	0	Invalid setting: must not be used
		1	Invalid setting: must not be used
1	0	0	Output pin with output fixed low
		1	Clock output as output pin
	1	0	Output pin with output fixed high
		1	Clock output as output pin

17.2.4 Serial Status Register (SCSSR1)

Bit 4 of SCSSR1 has a different function in smart card interface mode. Coupled with this, the setting conditions for bit 2 (TEND) are also different.

Bit:	7	6	5	4	3	2	1	0
	TDRE	RDRF	ORER	FER/ ERS	PER	TEND	—	—
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: Operate in the same way as for the normal SCI. See section 15, Serial Communication Interface, for details.

Bit 4—Error Signal Status (ERS): In smart card interface mode, bit 4 indicates the status of the error signal sent back from the receiving side during transmission. Framing errors are not detected in smart card interface mode.

Bit 4: ERS	Description
0	Normal reception, no error signal (Initial value) [Clearing conditions] • Power-on reset, manual reset, standby mode, or module standby • When 0 is written to ERS after reading ERS = 1
1	An error signal has been sent from the receiving side indicating detection of a parity error [Setting condition] • When the low level of the error signal is detected

Note: Clearing the TE bit in SCSCR1 to 0 does not affect the ERS flag, which retains its previous state.

Bit 3—Parity Error (PER): Operates in the same way as for the normal SCI. See section 15, Serial Communication Interface, for details.

Bit 2—Transmit End (TEND): The setting conditions for the TEND flag are as follows.

Bit 2: TEND	Description
0	Transmission in progress [Clearing condition] • When 0 is written to TDRE after reading TDRE = 1
1	Transmission has been ended (Initial value) [Setting conditions] • Power-on reset, manual reset, standby mode, or module standby • When the TE bit in SCSCR1 is 0 and the FER/ERS bit is also 0 • When the GM bit in SCSMR1 is 0, and TDRE = 1 and FER/ERS = 0 (normal transmission) 2.5 etu after transmission of a 1-byte serial character • When the GM bit in SCSMR1 is 1, and TDRE = 1 and FER/ERS = 0 (normal transmission) 1.0 etu after transmission of a 1-byte serial character

etu: Elementary Time Unit

Bits 1 and 0: Not used with the smart card interface.

17.3 Operation

17.3.1 Overview

The main functions of the smart card interface are as follows.

- One frame consists of 8-bit data plus a parity bit.
- In transmission, a guard time of at least 2 etu (elementary time unit: the time for transfer of one bit) is left between the end of the parity bit and the start of the next frame.
- If a parity error is detected during reception, a low error signal level is output for a 1-etu period 10.5 etu after the start bit.
- If an error signal is detected during transmission, the same data is transmitted automatically after the elapse of 2 etu or longer.
- Only asynchronous communication is supported; there is no synchronous communication function.

17.3.2 Pin Connections

Figure 17.2 shows a schematic diagram of smart card interface related pin connections.

In communication with an IC card, since both transmission and reception are carried out on a single data transmission line, the TxD pin and RxD pin should be connected outside the chip. The data transmission line should be pulled up on the V_{CC} power supply side with a resistor. The TxD pin is multiplexed with MD7, so caution is required in a reset.

When the clock generated on the smart card interface is used by an IC card, the SCK pin output is input to the CLK pin of the IC card. No connection is needed if the IC card uses an internal clock.

Chip port output is used as the reset signal.

Other pins must normally be connected to the power supply or ground.

Note: If an IC card is not connected, and both TE and RE are set to 1, closed transmission/reception is possible, enabling self-diagnosis to be carried out.

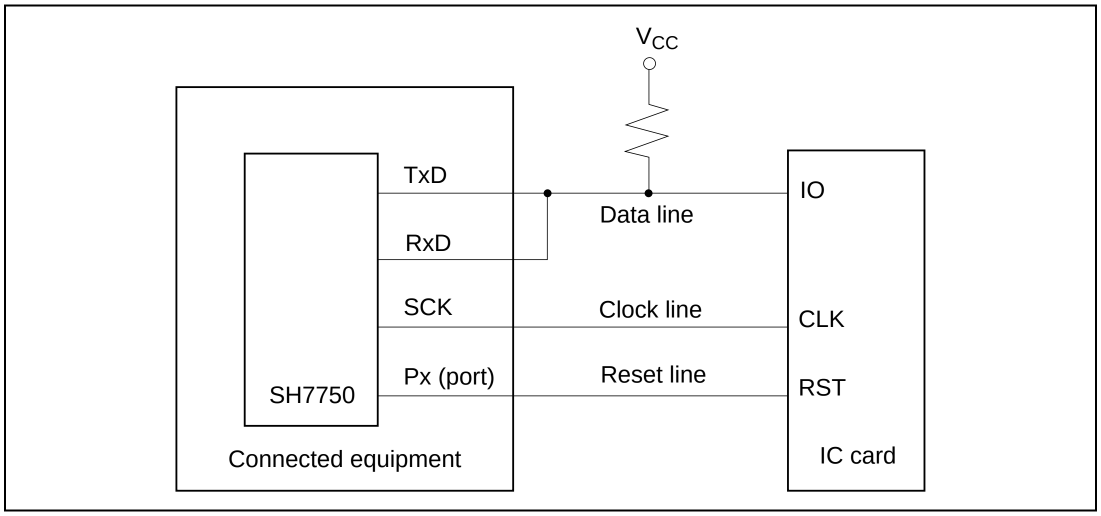
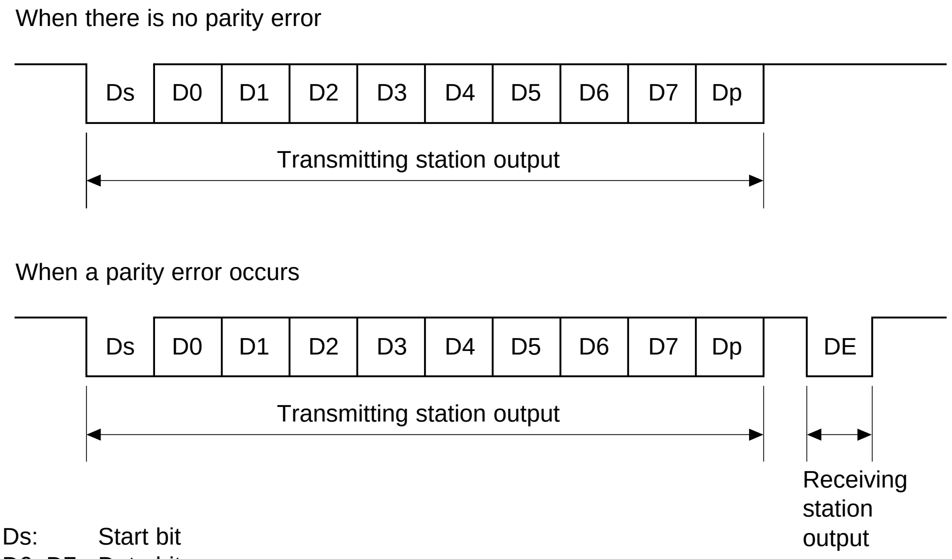


Figure 17.2 Schematic Diagram of Smart Card Interface Pin Connections

17.3.3 Data Format

Figure 17.3 shows the smart card interface data format. In reception in this mode, a parity check is carried out on each frame, and if an error is detected an error signal is sent back to the transmitting side to request retransmission of the data. If an error signal is detected during transmission, the same data is retransmitted.


When there is no parity error
Ds
D0
D1
D2
D3
D4
D5
D6
D7
Dp
Transmitting station output
When a parity error occurs
Ds
D0
D1
D2
D3
D4
D5
D6
D7
Dp
DE
Transmitting station output
Receiving station output

Ds:	Start bit
D0–D7:	Data bits
Dp:	Parity bit
DE:	Error signal

Figure 17.3 Smart Card Interface Data Format

The operation sequence is as follows.

1. When the data line is not in use it is in the high-impedance state, and is fixed high with a pull-up resistor.
2. The transmitting station starts transmission of one frame of data. The data frame starts with a start bit (Ds, low-level), followed by 8 data bits (D0 to D7) and a parity bit (Dp).
3. With the smart card interface, the data line then returns to the high-impedance state. The data line is pulled high with a pull-up resistor.
4. The receiving station carries out a parity check.

   If there is no parity error and the data is received normally, the receiving station waits for reception of the next data.

   If a parity error occurs, however, the receiving station outputs an error signal (DE, low-level) to request retransmission of the data. After outputting the error signal for the prescribed length of time, the receiving station places the signal line in the high-impedance state again. The signal line is pulled high again by a pull-up resistor.
5. If the transmitting station does not receive an error signal, it proceeds to transmit the next data frame.

   If it receives an error signal, however, it returns to step 2 and retransmits the erroneous data.

17.3.4 Register Settings

Table 17.3 shows a bit map of the registers used by the smart card interface. Bits indicated as 0 or 1 must be set to the value shown. The setting of other bits is described below.

Table 17.3 Smart Card Interface Register Settings

	Bit							
Register	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
SCSMR1	GM	0	1	$O/\overline{E}$	1	0	CKS1	CKS0
SCBRR1	BRR7	BRR6	BRR5	BRR4	BRR3	BRR2	BRR1	BRR0
SCSCR1	TIE	RIE	TE	RE	0	0	CKE1	CKE0
SCTDR1	TDR7	TDR6	TDR5	TDR4	TDR3	TDR2	TDR1	TDR0
SCSSR1	TDRE	RDRF	ORER	FER/ERS	PER	TEND	0	0
SCRDR1	RDR7	RDR6	RDR5	RDR4	RDR3	RDR2	RDR1	RDR0
SCSCMR1	—	—	—	—	SDIR	SINV	—	SMIF
SCSPTR1	EIO	—	—	—	SPB1IO	SPB1DT	SPB0IO	SPB0DT

Note: A dash indicates an unused bit.

Serial Mode Register (SCSMR1) Settings: The GM bit is used to select the timing of TEND flag setting, and, together with the CKE1 and CKE0 bits in the serial control register (SCSCR1), to select the clock output state.

The O/$\overline{\text{E}}$ bit is cleared to 0 if the IC card is of the direct convention type, and set to 1 if of the inverse convention type.

Bits CKS1 and CKS0 select the clock source of the on-chip baud rate generator. See section 17.3.5, Clock.

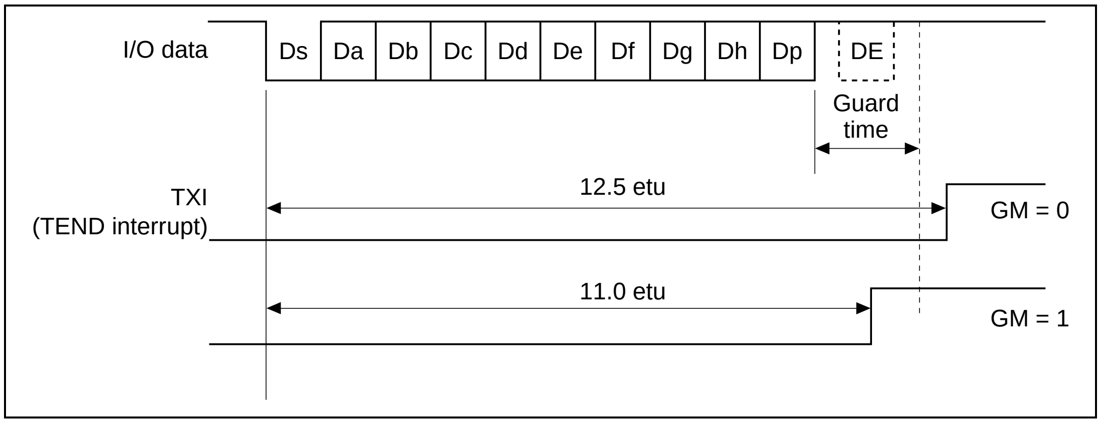


Figure 17.4 TEND Generation Timing

Bit Rate Register (SCBRR1) Setting: SCBRR1 is used to set the bit rate. See section 17.3.5, Clock, for the method of calculating the value to be set.

Serial Control Register (SCSCR1) Settings: The function of the TIE, RIE, TE, and RE bits is the same as for the normal SCI. See section 15, Serial Communication Interface, for details.

The CKE1 and CKE0 bits specify the clock output state. See section 17.3.5, Clock, for details.

Smart Card Mode Register (SCSCMR1) Settings: The SDIR bit and SINV bit are both cleared to 0 if the IC card is of the direct convention type, and both set to 1 if of the inverse convention type.

The SMIF bit is set to 1 when the smart card interface is used.

Figure 17.5 shows examples of register settings and the waveform of the start character for the two types of IC card (direct convention and inverse convention).

With the direct convention type, the logic 1 level corresponds to state Z and the logic 0 level to state A, and transfer is performed in LSB-first order. The start character data in this case is H'3B. The parity bit is 1 since even parity is stipulated for the smart card.

With the inverse convention type, the logic 1 level corresponds to state A and the logic 0 level to state Z, and transfer is performed in MSB-first order. The start character data in this case is H'3F. The parity bit is 0, corresponding to state Z, since even parity is stipulated for the smart card.

Inversion specified by the SINV bit applies only to the data bits, D7 to D0. For parity bit inversion, the O/$\overline{\text{E}}$ bit in SCSMR1 is set to odd parity mode. (This applies to both transmission and reception).

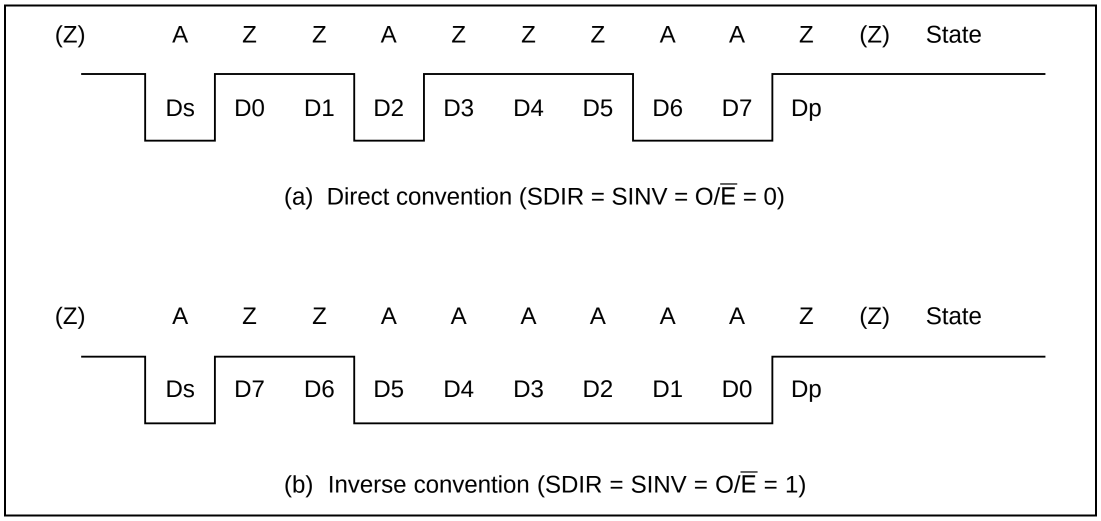

(Z)
A
Z
Z
A
Z
Z
Z
A
A
Z
(Z)
State
Ds
D0
D1
D2
D3
D4
D5
D6
D7
Dp
(a) Direct convention (SDIR = SINV = O/E̅ = 0)
(Z)
A
Z
Z
A
A
A
A
A
A
Z
(Z)
State
Ds
D7
D6
D5
D4
D3
D2
D1
D0
Dp
(b) Inverse convention (SDIR = SINV = O/E̅ = 1)

Figure 17.5 Sample Start Character Waveforms

17.3.5 Clock

Only an internal clock generated by the on-chip baud rate generator can be used as the transmit/receive clock for the smart card interface. The bit rate is set with the bit rate register (SCBRR1) and the CKS1 and CKS0 bits in the serial mode register (SCSMR1). The equation for calculating the bit rate is shown below. Table 17.5 shows some sample bit rates.

If clock output is selected with CKE0 set to 1, a clock with a frequency of 372 times the bit rate is output from the SCK pin.

$$B = \frac{P\phi}{1488 \times 2^{2n-1} \times (N + 1)} \times 10^6$$

Where: N = Value set in SCBRR1 ($0 \le N \le 255$)
B = Bit rate (bits/s)
Pϕ = Peripheral module operating frequency (MHz)
n = 0 to 3 (See table 17.4)

Table 17.4 Values of n and Corresponding CKS1 and CKS0 Settings

n	CKS1	CKS0
0	0	0
1	0	1
2	1	0
3	1	1

Table 17.5 Examples of Bit Rate B (bits/s) for Various SCBRR1 Settings (When n = 0)

	Pϕ (MHz)						
N	**7.1424**	**10.00**	**10.7136**	**14.2848**	**25.0**	**33.0**	**50.0**
0	9600.0	13440.9	14400.0	19200.0	33602.2	44354.8	67204.3
1	4800.0	6720.4	7200.0	9600.0	16801.1	22177.4	33602.2
2	3200.0	4480.3	4800.0	6400.0	11200.7	14784.9	22401.4

Note: Bit rates are rounded to one decimal place.

The method of calculating the value to be set in the bit rate register (SCBRR1) from the peripheral module operating frequency and bit rate is shown below. Here, N is an integer in the range $0 \leq N \leq 255$, and the smaller error is specified.

$$N = \frac{P\phi}{1488 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Table 17.6 Examples of SCBRR1 Settings for Bit Rate B (bits/s) (When n = 0)

	Pϕ (MHz)													
	7.1424		**10.00**		**10.7136**		**14.2848**		**25.00**		**33.00**		**50.00**	
Bits/s	**N**	**Error**	**N**	**Error**	**N**	**Error**	**N**	**Error**	**N**	**Error**	**N**	**Error**	**N**	**Error**
9600	0	0.00	1	30.00	1	25.00	1	8.99	3	14.27	4	8.22	6	0.01

Table 17.7 Maximum Bit Rate at Various Frequencies (Smart Card Interface Mode)

Pϕ (MHz)	**Maximum Bit Rate (bits/s)**	**N**	**n**
7.1424	19200	0	0
10.00	26882	0	0
10.7136	28800	0	0
16.00	43010	0	0
20.00	53763	0	0
25.0	67204	0	0
30.0	80645	0	0
33.0	88710	0	0
50.0	67204	0	0

The bit rate error is given by the following equation:

$$\text{Error (\%)} = \left\{ \frac{P\phi}{1488 \times 2^{2n-1} \times B \times (N+1)} \right\} \times 10^6 - 1 \times 100$$

Table 17.8 shows the relationship between the smart card interface transmit/receive clock register settings and the output state.

Table 17.8 Register Settings and SCK Pin State

	Register Values				SCK Pin	
Setting	**SMIF**	**GM**	**CKE1**	**CKE0**	**Output**	**State**
1*[1]	1	0	0	0	Port	Determined by setting of SPB1IO and SPB1DT bits in SCSPTR1
	1	0	0	1	⎺⎽⎺⎽⎺⎽	SCK (serial clock) output state
2*[2]	1	1	0	0	Low output	Low-level output state
	1	1	0	1	⎺⎽⎺⎽⎺⎽	SCK (serial clock) output state
3*[2]	1	1	1	0	High output	High-level output state
	1	1	1	1	⎺⎽⎺⎽⎺⎽	SCK (serial clock) output state

Notes: 1. The SCK output state changes as soon as the CKE0 bit setting is changed.
Clear the CKE1 bit to 0.
2. Stopping and starting the clock by changing the CKE0 bit setting does not affect the clock duty cycle.

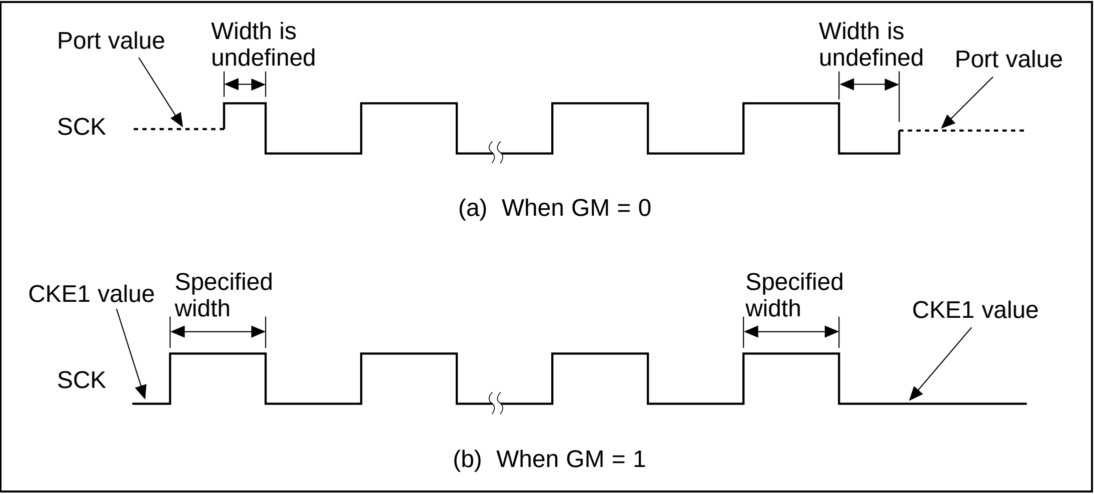


Figure 17.6 Difference in Clock Output According to GM Bit Setting

17.3.6 Data Transfer Operations

Initialization: Before transmitting and receiving data, the smart card interface must be initialized as described below. Initialization is also necessary when switching from transmit mode to receive mode, or vice versa. Figure 17.7 shows a sample initialization processing flowchart.

1. Clear the TE and RE bits in the serial control register (SCSCR1) to 0.
2. Clear error flags FER/ERS, PER, and ORER in the serial status register (SCSSR1) to 0.
3. Set the GM bit, parity bit (O/$\overline{\text{E}}$), and baud rate generator select bits (CKS1 and CKS0) in the serial mode register (SCSMR1). 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 (SCSCMR1).

   When the SMIF bit is set to 1, the TxD pin and RxD pin both go to the high-impedance state.
5. Set the value corresponding to the bit rate in the bit rate register (SCBRR1).
6. Set the clock source select bits (CKE1 and CKE0) in SCSCR1. Clear the TIE, RIE, TE, RE, MPIE, and TEIE bits to 0.

   If the CKE0 bit is set to 1, the clock is output from the SCK pin.
7. Wait at least one bit interval, then set the TIE, RIE, TE, and RE bits in SCSCR1. Do not set the TE bit and RE bit at the same time, except for self-diagnosis.

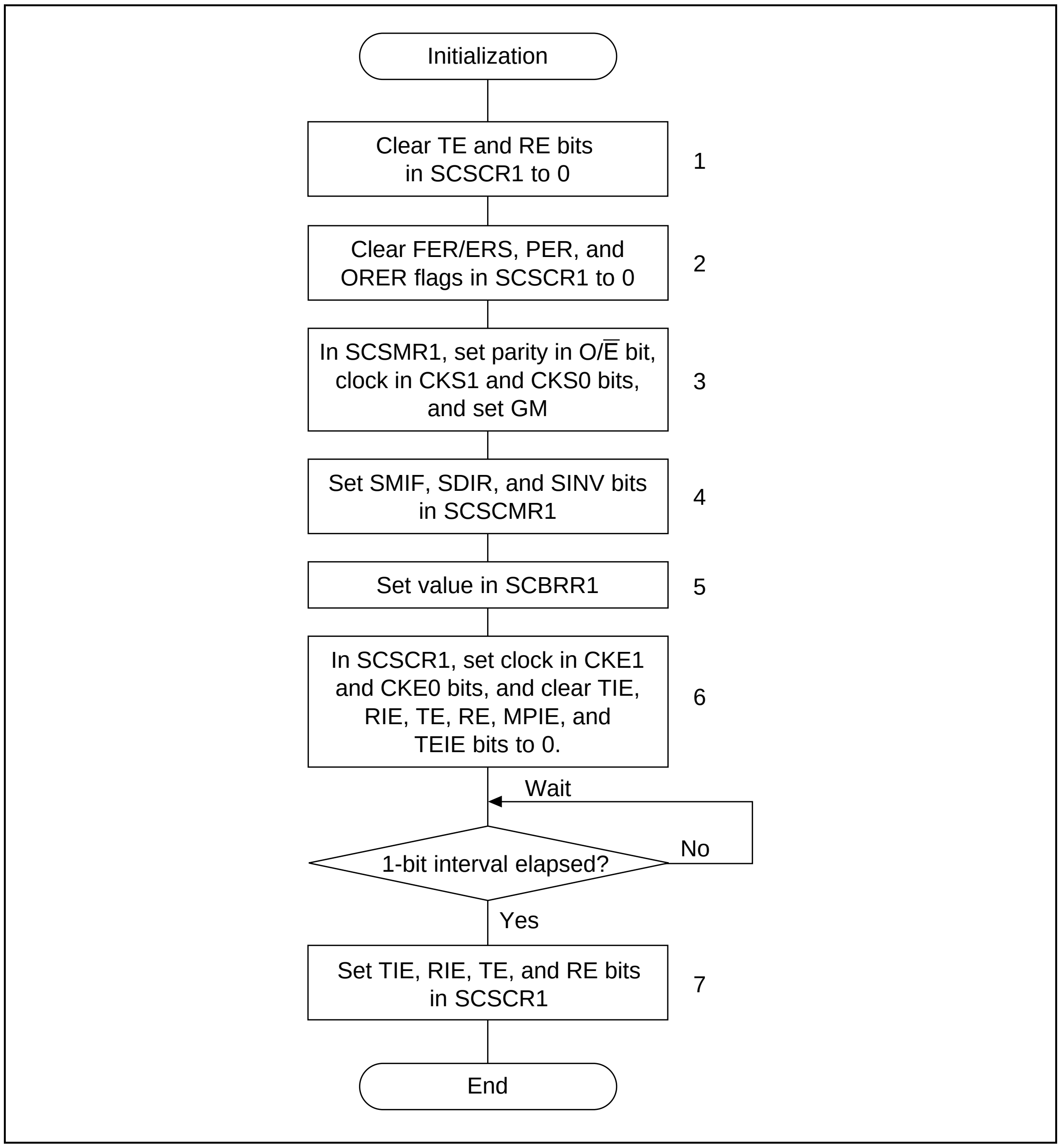

Initialization
Clear TE and RE bits in SCSCR1 to 0
1
Clear FER/ERS, PER, and ORER flags in SCSCR1 to 0
2
In SCSMR1, set parity in O/$\overline{E}$ bit, clock in CKS1 and CKS0 bits, and set GM
3
Set SMIF, SDIR, and SINV bits in SCSCMR1
4
Set value in SCBRR1
5
In SCSCR1, set clock in CKE1 and CKE0 bits, and clear TIE, RIE, TE, RE, MPIE, and TEIE bits to 0.
6
Wait
1-bit interval elapsed?
No
Yes
Set TIE, RIE, TE, and RE bits in SCSCR1
7
End

Figure 17.7 Sample Initialization Flowchart

Serial Data Transmission: As data transmission in smart card mode involves error signal sampling and retransmission processing, the processing procedure is different from that for the normal SCI. Figure 17.8 shows a sample transmission processing flowchart.

1. Perform smart card interface mode initialization as described in Initialization above.
2. Check that the FER/ERS error flag in SCSSR1 is cleared to 0.
3. Repeat steps 2 and 3 until it can be confirmed that the TEND flag in SCSSR1 is set to 1.
4. Write the transmit data to SCTDR1, clear the TDRE flag to 0, and perform the transmit operation. The TEND flag is cleared to 0.
5. To continue transmitting data, go back to step 2.
6. To end transmission, clear the TE bit to 0.

With the above processing, interrupt handling is possible.

If transmission ends and the TEND flag is set to 1 while the TIE bit is set to 1 and interrupt requests are enabled, a transmit-data-empty interrupt (TXI) request will be generated. If an error occurs in transmission and the ERS flag is set to 1 while the RIE bit is set to 1 and interrupt requests are enabled, a transmit/receive-error interrupt (ERI) request will be generated. See Interrupt Operation below for details.

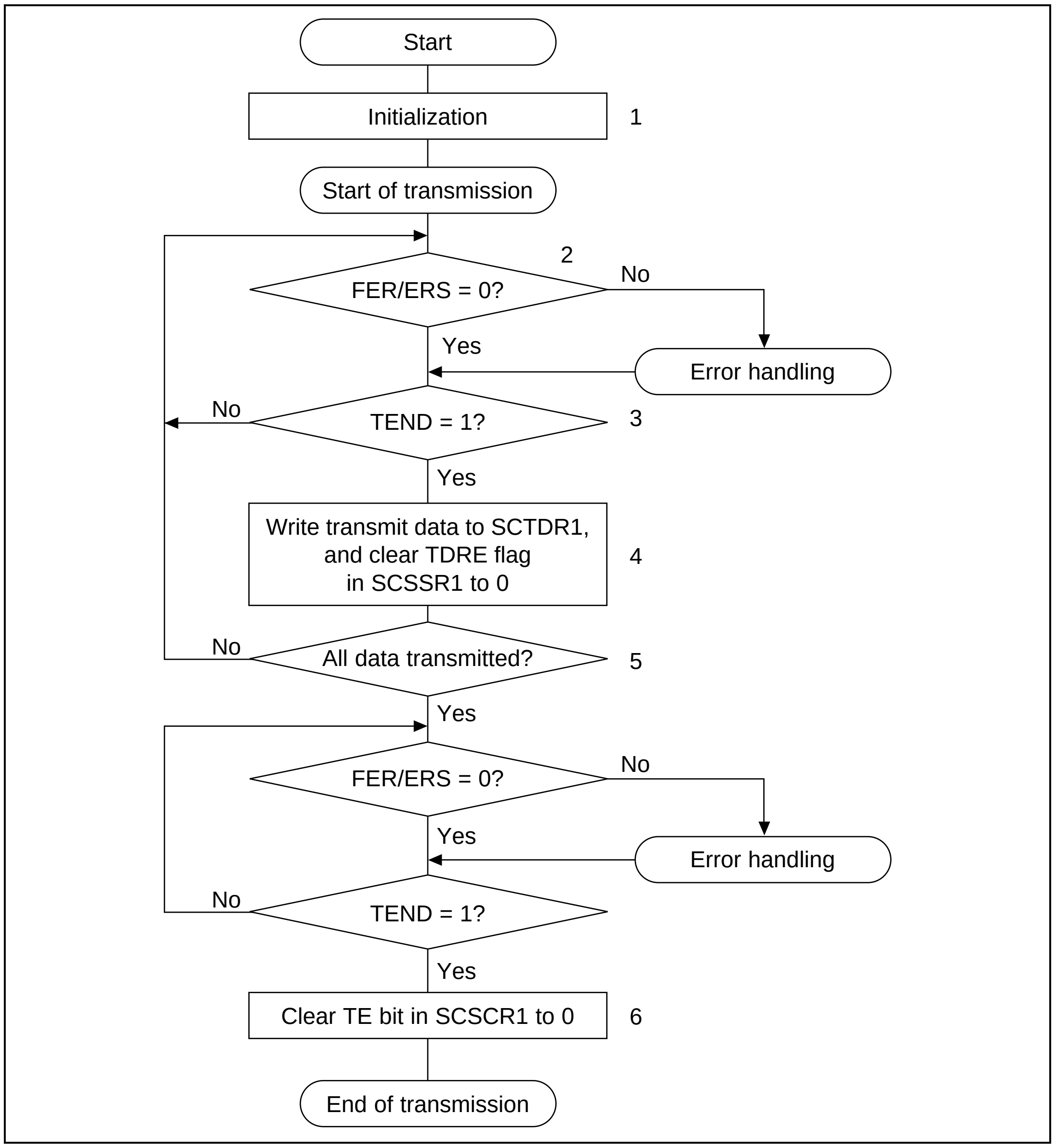


Figure 17.8 Sample Transmission Processing Flowchart

Serial Data Reception: Data reception in smart card mode uses the same processing procedure as for the normal SCI. Figure 17.9 shows a sample reception processing flowchart.

1. Perform smart card interface mode initialization as described in Initialization above.
2. Check that the ORER flag and PER flag in SCSSR1 are cleared to 0. If either is set, perform the appropriate receive error handling, then clear both the ORER and the PER flag to 0.
3. Repeat steps 2 and 3 until it can be confirmed that the RDRF flag is set to 1.
4. Read the receive data from SCRDR1.
5. To continue receiving data, clear the RDRF flag to 0 and go back to step 2.
6. To end reception, clear the RE bit to 0.

With the above processing, interrupt handling is possible.

If reception ends and the RDRF flag is set to 1 while the RIE bit is set to 1 and interrupt requests are enabled, a receive-data-full interrupt (RXI) request will be generated. If an error occurs in reception and either the ORER flag or the PER flag is set to 1, a transmit/receive-error interrupt (ERI) request will be generated.

See Interrupt Operation below for details.

If a parity error occurs during reception and the PER flag is set to 1, the received data is still transferred to SCRDR1, and therefore this data can be read.

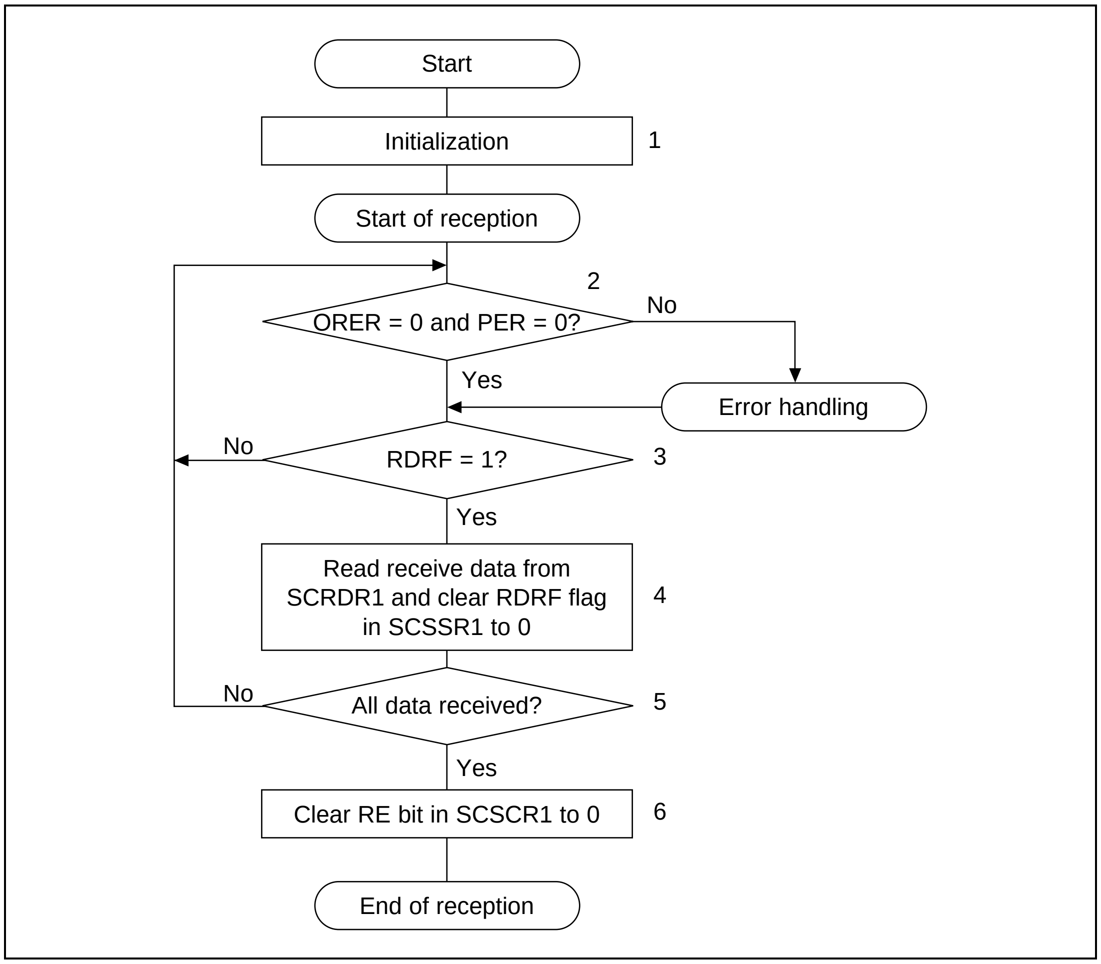


Figure 17.9 Sample Reception Processing Flowchart

Mode Switching Operation: When switching from receive mode to transmit mode, first confirm that the receive operation has been completed, then start from initialization, clearing RE to 0 and setting TE to 1. The RDRF flag or the PER and ORER flags can be used to check that the receive operation has been completed.

When switching from transmit mode to receive mode, first confirm that the transmit operation has been completed, then start from initialization, clearing TE to 0 and setting RE to 1. The TEND flag can be used to check that the transmit operation has been completed.

Interrupt Operation: There are three interrupt sources in smart card interface mode, generating transmit-data-empty interrupt (TXI) requests, transmit/receive-error interrupt (ERI) requests, and receive-data-full interrupt (RXI) requests. The transmit-end interrupt (TEI) request cannot be used in this mode.

When the TEND flag in SCSSR1 is set to 1, a TXI interrupt request is generated.

When the RDRF flag in SCSSR1 is set to 1, an RXI interrupt request is generated.

When any of flags ORER, PER, and FER/ERS in SCSSR1 is set to 1, an ERI interrupt request is generated. The relationship between the operating states and interrupt sources is shown in table 17.9.

Table 17.9 Smart Card Mode Operating States and Interrupt Sources

Operating State		Flag	Mask Bit	Interrupt Source
Transmit mode	Normal operation	TEND	TIE	TXI
	Error	FER/ERS	RIE	ERI
Receive mode	Normal operation	RDRF	RIE	RXI
	Error	PER, ORER	RIE	ERI

Data Transfer Operation by DMAC: In smart card mode, as with the normal SCI, transfer can be carried out using the DMAC. In a transmit operation, when the TEND flag in SCSSR1 is set to 1, a TXI interrupt is requested. If the TXI request is designated beforehand as a DMAC activation source, the DMAC will be activated by the TXI request, and transfer of the transmit data will be carried out. The TEND flag is automatically cleared to 0 when data transfer is performed by the DMAC. In the event of an error, the SCI retransmits the same data automatically. The TEND flag remains cleared to 0 during this time, and the DMAC is not activated. Thus, the number of bytes specified by the SCI and DMAC are transmitted automatically, including retransmission following an error. However, the ERS flag is not cleared automatically when an error occurs, and therefore the RIE bit should be set to 1 beforehand so that an ERI request will be generated in the event of an error, and the ERS flag will be cleared.

In a receive operation, an RXI interrupt request is generated when the RDRF flag in SCSSR1 is set to 1. If the RXI request is designated beforehand as a DMAC activation source, the DMAC will be activated by the RXI request, and transfer of the receive data will be carried out.. The RDRF flag is cleared to 0 automatically when data transfer is performed by the DMAC. If an error occurs, an error flag is set but the RDRF flag is not. The DMAC is not activated, but instead, an ERI interrupt request is sent to the CPU. The error flag must therefore be cleared.

When performing data transfer using the DMAC, it is essential to set and enable the DMAC before carrying out SCI settings. For details of the DMAC setting procedures, see section 14, Direct Memory Access Controller (DMAC).

17.4 Usage Notes

The following points should be noted when using the SCI as a smart card interface.

(1) Receive Data Sampling Timing and Receive Margin

In asynchronous mode, the SCI operates on a base clock with a frequency of 372 times the transfer rate. In reception, the SCI 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 186th base clock pulse. The timing is shown in figure 17.10.

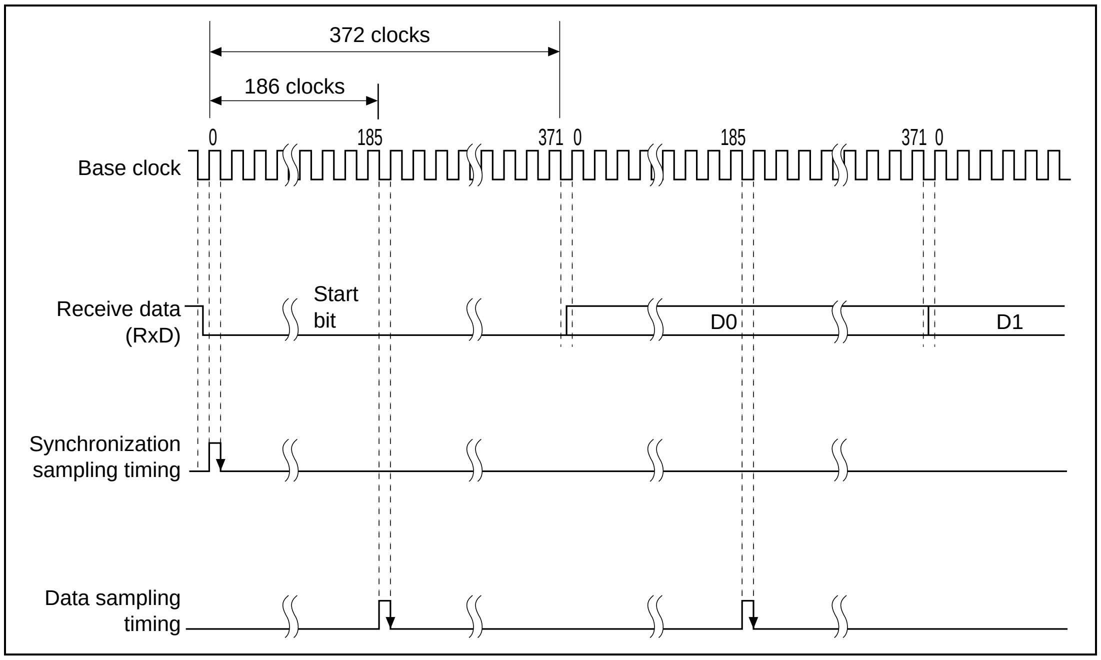


Figure 17.10 Receive Data Sampling Timing in Smart Card Mode

The receive margin in smart card mode can therefore be expressed as shown in the following equation.

$$M = \left| (0.5 - \frac{1}{2N}) - (L - 0.5)\,F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100\%$$

M: Receive margin (%)
N: Ratio of clock frequency to bit rate (N = 372)
D: Clock duty cycle (D = 0 to 1.0)
L: Frame length (L =10)
F: Absolute deviation of clock frequency

From the above equation, if F = 0 and D = 0.5, the receive margin is 49.866%, as given by the following equation.

When D = 0.5 and F = 0:

$$M = (0.5 - 1/2 \times 372) \times 100\% = 49.866\%$$

(2) Retransfer Operations

Retransfer operations are performed by the SCI in receive mode and transmit mode as described below.

Retransfer Operation when SCI is in Receive Mode: Figure 17.11 illustrates the retransfer operation when the SCI is in receive mode.

1. If an error is found when the received parity bit is checked, the PER bit in SCSSR1 is automatically set to 1. If the RIE bit in SCSCR1 is enabled at this time, an ERI interrupt request is generated. The PER bit in SCSSR1 should be cleared to 0 before the next parity bit is sampled.
2. The RDRF bit in SCSSR1 is not set for a frame in which an error has occurred.
3. If an error is found when the received parity bit is checked, the PER bit in SCSSR1 is not set to 1.
4. If no error is found when the received parity bit is checked, the receive operation is judged to have been completed normally, and the RDRF bit in SCSSR1 is automatically set to 1. If the RIE bit in SCSCR1 is enabled at this time, an RXI interrupt request is generated.
5. When a normal frame is received, the pin retains the high-impedance state at the timing for error signal transmission.

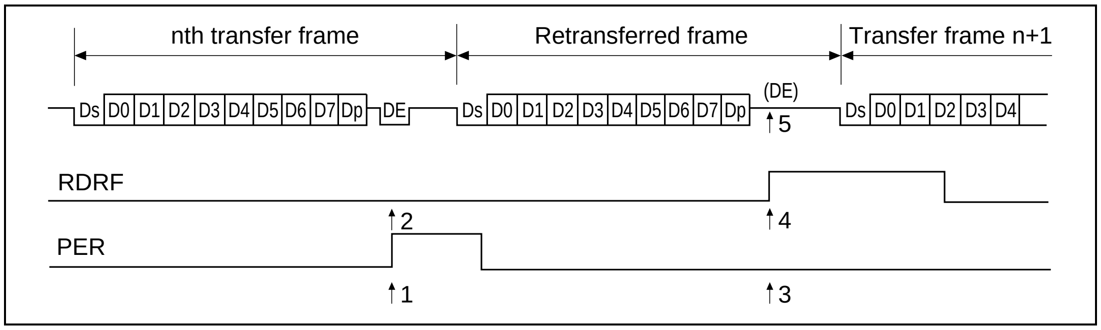


Figure 17.11 Retransfer Operation in SCI Receive Mode

Retransfer Operation when SCI is in Transmit Mode: Figure 17.12 illustrates the retransfer operation when the SCI is in transmit mode.

1. If an error signal is sent back from the receiving side after transmission of one frame is completed, the FER/ERS bit in SCSSR1 is set to 1. If the RIE bit in SCSCR1 is enabled at this time, an ERI interrupt request is generated. The FER/ERS bit in SCSSR1 should be cleared to 0 before the next parity bit is sampled.
2. The TEND bit in SCSSR1 is not set for a frame for which an error signal indicating an error is received.
3. If an error signal is not sent back from the receiving side, the FER/ERS bit in SCSSR1 is not set.
4. If an error signal is not sent back from the receiving side, transmission of one frame, including a retransfer, is judged to have been completed, and the TEND bit in SCSSR1 is set to 1. If the TIE bit in SCSCR1 is enabled at this time, a TXI interrupt request is generated.

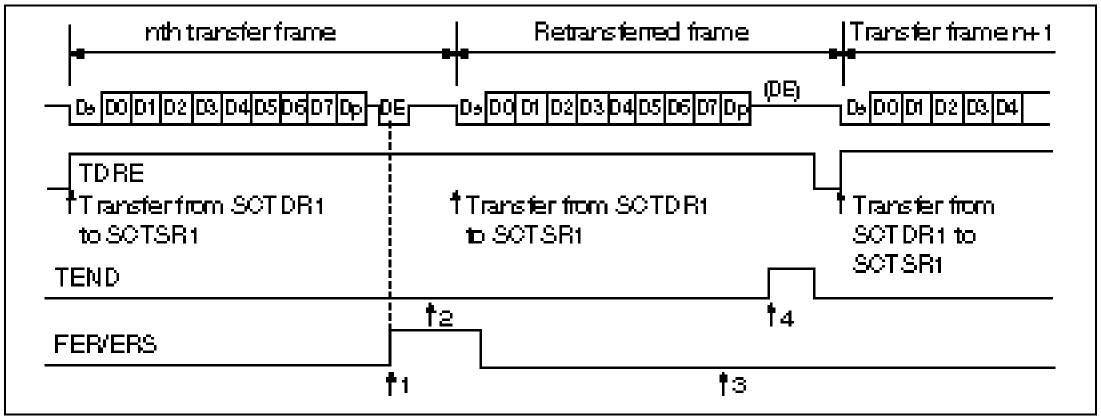

nth transfer frame
Retransferred frame
Transfer frame n+1
Ds D0 D1 D2 D3 D4 D5 D6 D7 Dp DE
Ds D0 D1 D2 D3 D4 D5 D6 D7 Dp (DE)
Ds D0 D1 D2 D3 D4
TDRE
Transfer from SCTDR1 to SCTSR1
Transfer from SCTDR1 to SCTSR1
Transfer from SCTDR1 to SCTSR1
TEND
2
4
FER/ERS
1
3

Figure 17.12 Retransfer Operation in SCI Transmit Mode

(3) Standby Mode and Clock

When switching between smart card interface mode and standby mode, the following procedures should be used to maintain the clock duty cycle.

Switching from Smart Card Interface Mode to Standby Mode:

1. Set the SBP1IO and SBP1DT bits in SCSPTR1 to the values for the fixed output state in standby mode.
2. Write 0 to the TE and RE bits in the serial control register (SCSCR1) to stop transmit/receive operations. At the same time, set the CKE1 bit to the value for the fixed output state in standby mode.
3. Write 0 to the CKE0 bit in SCSCR1 to stop the clock.
4. Wait for one serial clock cycle. During this period, the duty cycle is preserved and clock output is fixed at the specified level.
5. Write H'00 to the serial mode register (SCSMR1) and smart card mode register (SCSMR1).
6. Make the transition to the standby state.

Returning from Standby Mode to Smart Card Interface Mode:

7. Clear the standby state.
8. Set the CKE1 bit in SCSCR1 to the value for the fixed output state at the start of standby (the current SCK pin state).
9. Set smart card interface mode and output the clock. Clock signal generation is started with the normal duty cycle.

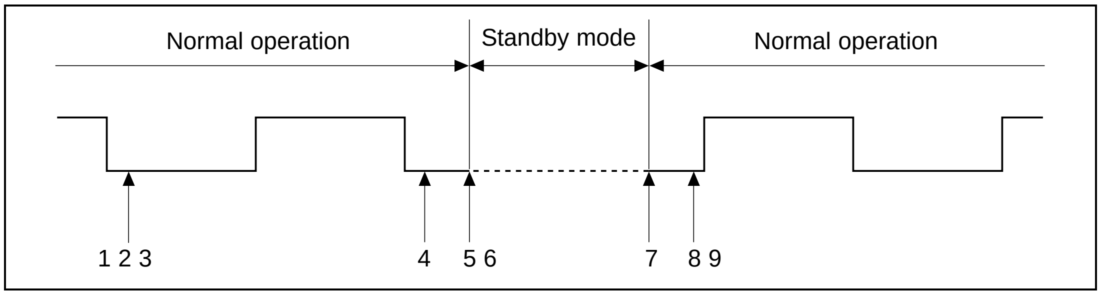


Figure 17.13 Procedure for Stopping and Restarting the Clock

(4) Power-On and Clock

The following procedure should be used to secure the clock duty cycle after powering on.

1. The initial state is port input and high impedance. Use pull-up or pull-down resistors to fix the potential.
2. Fix at the output specified by the CKE1 bit in the serial control register (SCSCR1).
3. Set the serial mode register (SCSMR1) and smart card mode register (SCSCMR1), and switch to smart card mode operation.
4. Set the CKE0 bit in SCSCR1 to 1 to start clock output.

Section 18 I/O Ports

18.1 Overview

The SH7750 has a 20-bit general-purpose I/O port, SCI I/O port, and SCIF I/O port.

18.1.1 Features

The features of the general-purpose I/O port are as follows:

- 20-bit I/O port with input/output direction independently specifiable for each bit
- Pull-up can be specified independently for each bit.
- Interrupt input is possible for 16 of the 20 I/O port bits.
- Use or non-use of the I/O port can be selected with the PORTEN bit in bus control register 2 (BCR2).

The features of the SCI I/O port are as follows:

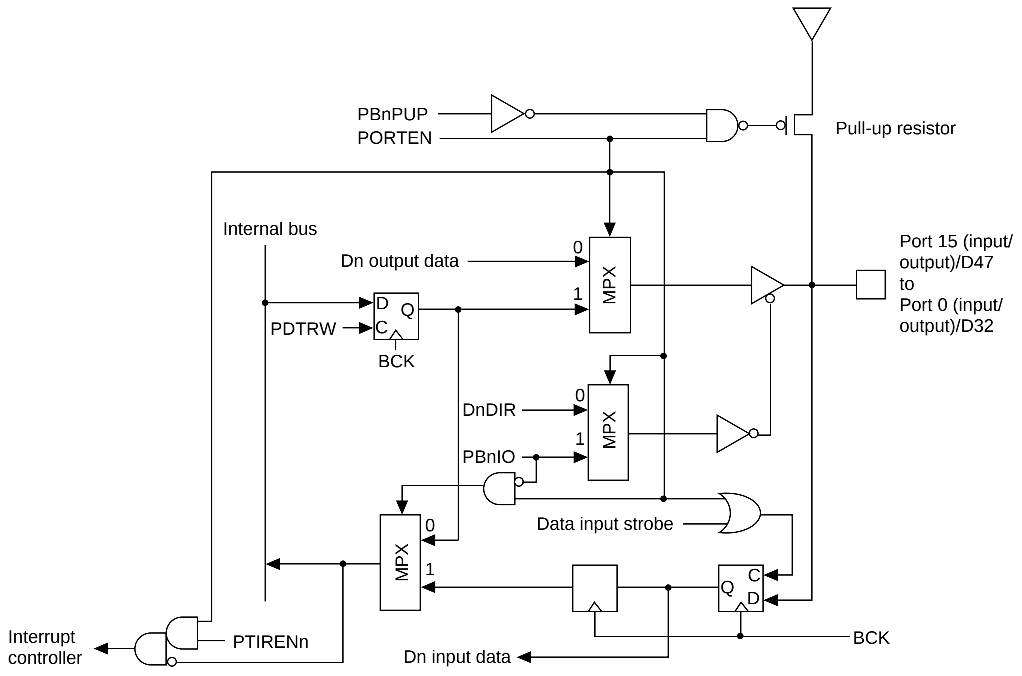
- Data can be output when the I/O port is designated for output and SCI enabling has not been set. This allows break function transmission.
- The RxD pin value can be read at all times, allowing break state detection.
- SCK pin control is possible when the I/O port is designated for output and SCI enabling has not been set.
- The SCK pin value can be read at all times.

The features of the SCIF I/O port are as follows:

- Data can be output when the I/O port is designated for output and SCIF enabling has not been set. This allows break function transmission.
- The RxD2 pin value can be read at all times, allowing break state detection.
- $\overline{\text{CTS2}}$ and $\overline{\text{RTS2}}$ pin control is possible when the I/O port is designated for output and SCIF enabling has not been set.
- The $\overline{\text{CTS2}}$ and $\overline{\text{RTS2}}$ pin values can be read at all times.

18.1.2 Block Diagrams

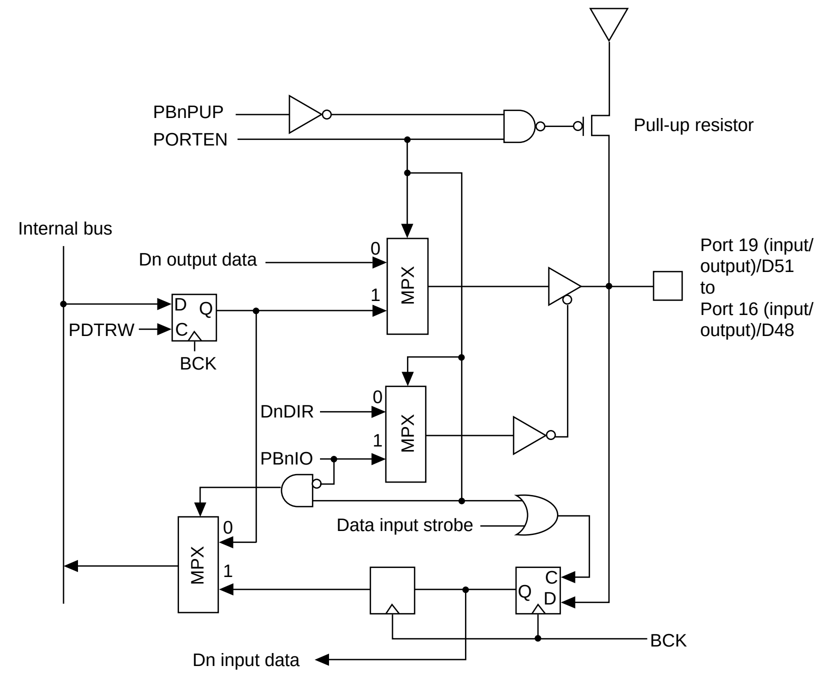
Figure 18.1 shows a block diagram of the 16-bit general-purpose I/O port.



PORTEN	0: Port not available	1: Port available
PBnPuP	0: Pull-up	1: Pull-up off
DnDIR	0: Input	1: Output
PBnIO	0: Input	1: Output
PTIRENn	0: Interrupt input disabled	1: Interrupt input enabled

Figure 18.1 16-Bit Port

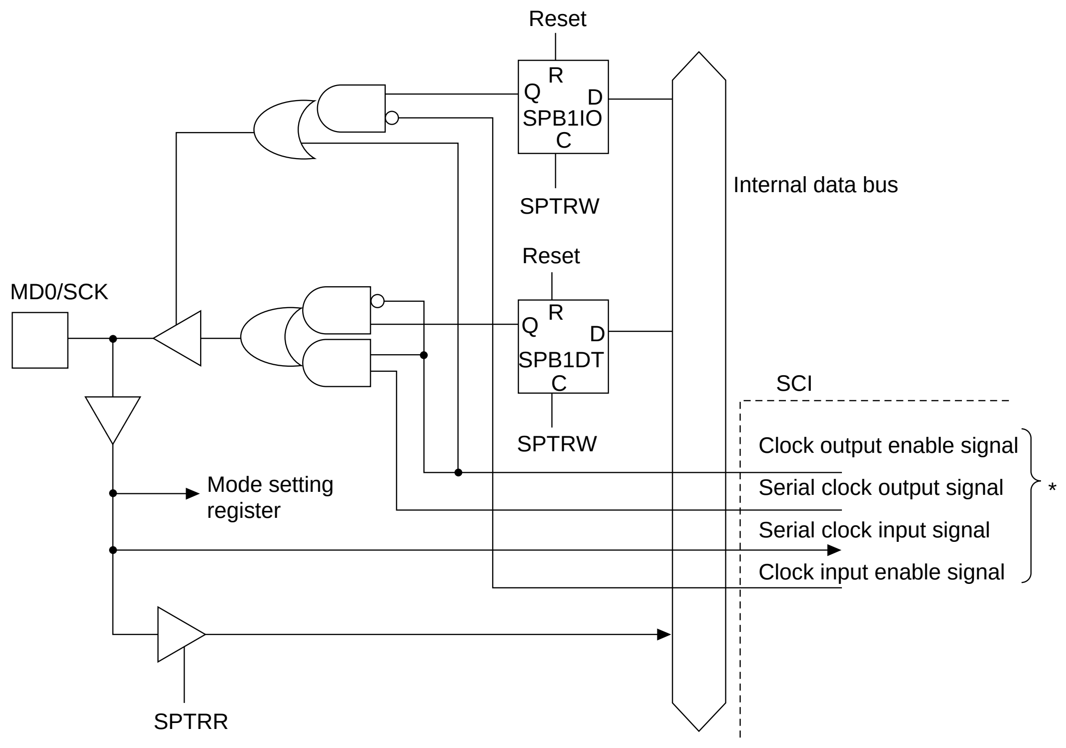
Figure 18.2 shows a block diagram of the 4-bit general-purpose I/O port.



PORTEN	0: Port not available	1: Port available
PBnPuP	0: Pull-up	1: Pull-up off
DnDIR	0: Input	1: Output
PBnIO	0: Input	1: Output

Figure 18.2 4-Bit Port

SCI I/O port block diagrams are shown in figures 18.3 to 18.5.


Reset
R
Q
D
SPB1IO
C
SPTRW
Internal data bus
Reset
MD0/SCK
R
Q
D
SPB1DT
C
SPTRW
SCI
Clock output enable signal
Mode setting register
Serial clock output signal
*
Serial clock input signal
Clock input enable signal
SPTRR

SPTRW: Write to SPTR
SPTRR: Read SPTR

Note: * Signals that set the SCK pin function as internal clock output or external clock input according to the CKE0 and CKE1 bits in SCSCR1 and the C/$\overline{A}$ bit in SCSMR1.

Figure 18.3 MD0/SCK Pin

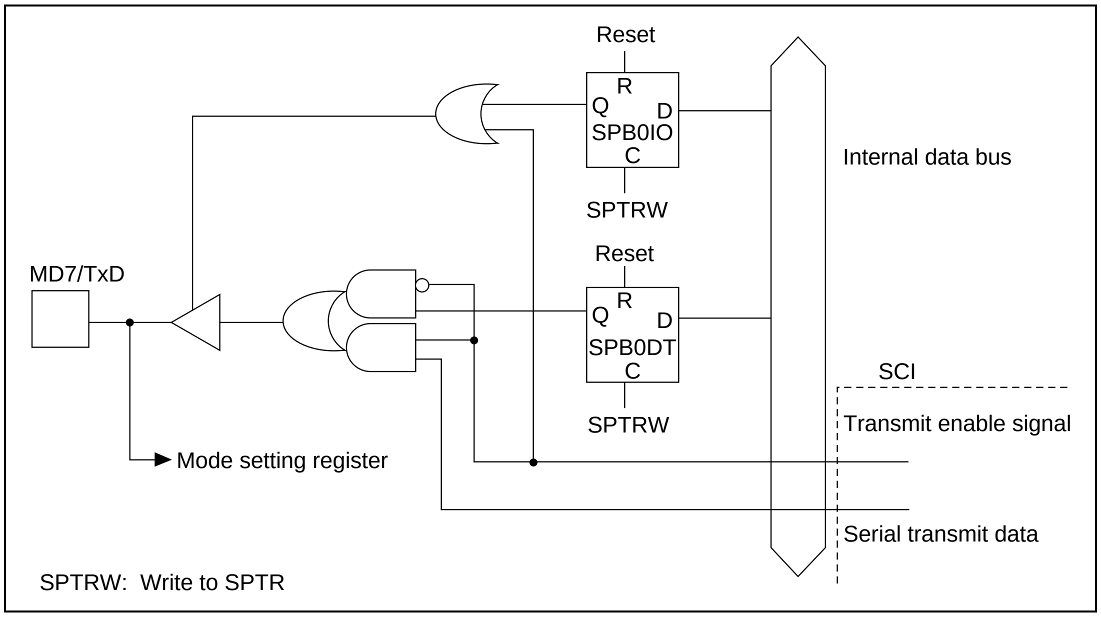


Figure 18.4 MD7/TxD Pin

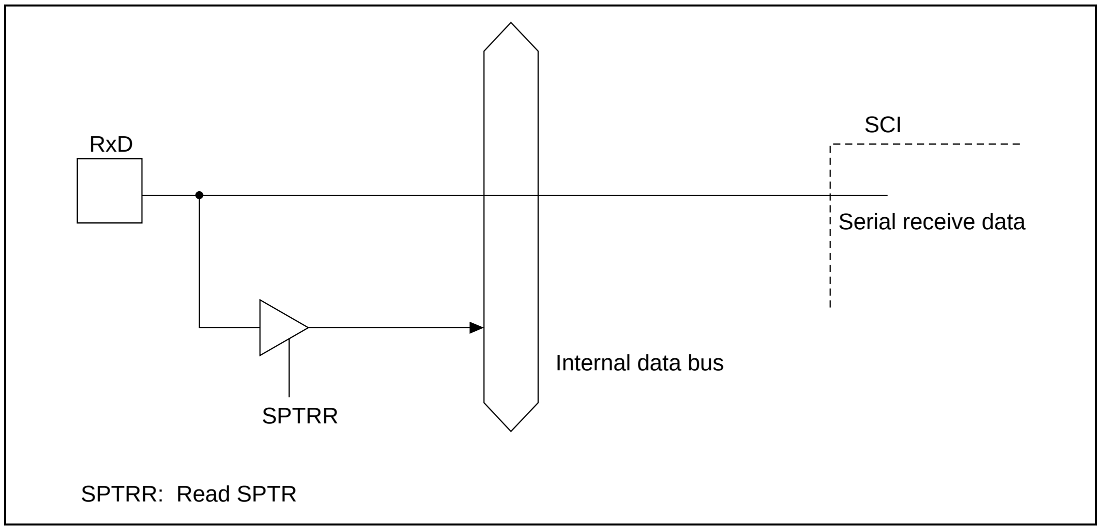


Figure 18.5 RxD Pin

SCIF I/O port block diagrams are shown in figures 18.6 to 18.9.

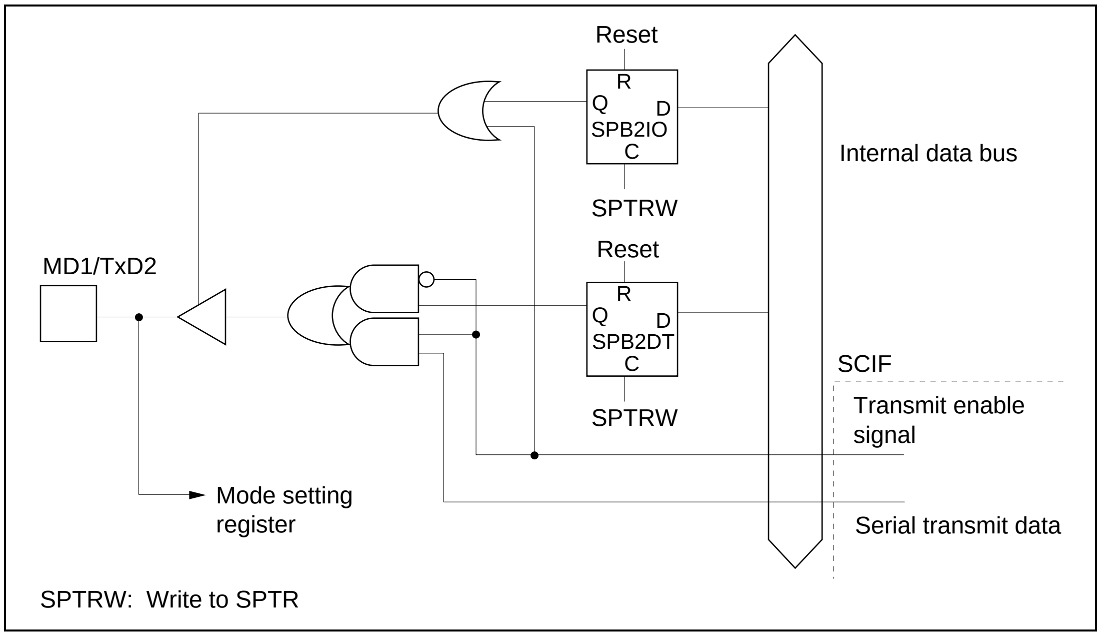


Figure 18.6 MD1/TxD2 Pin

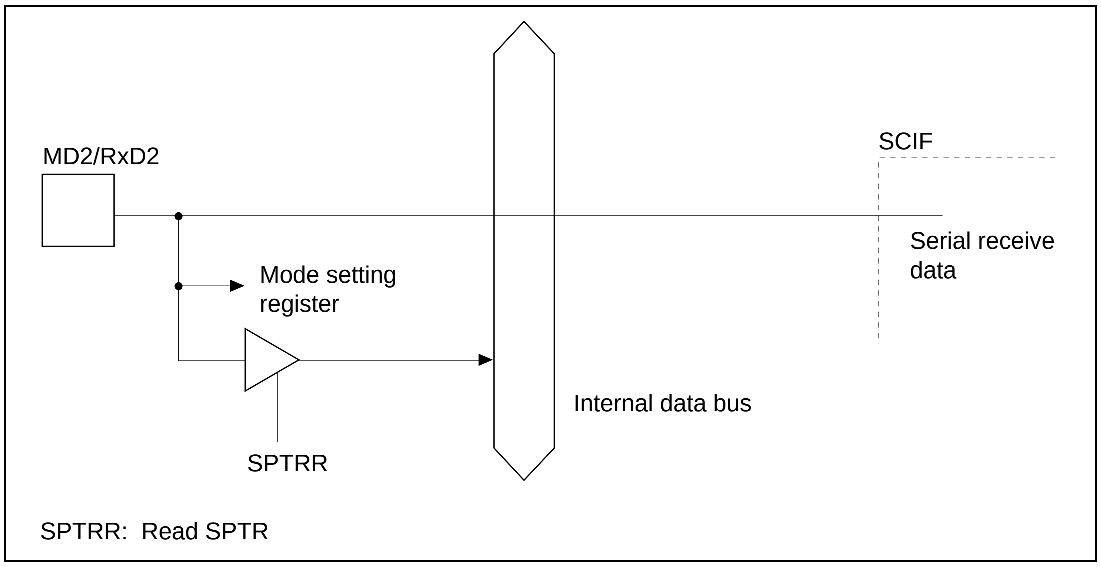
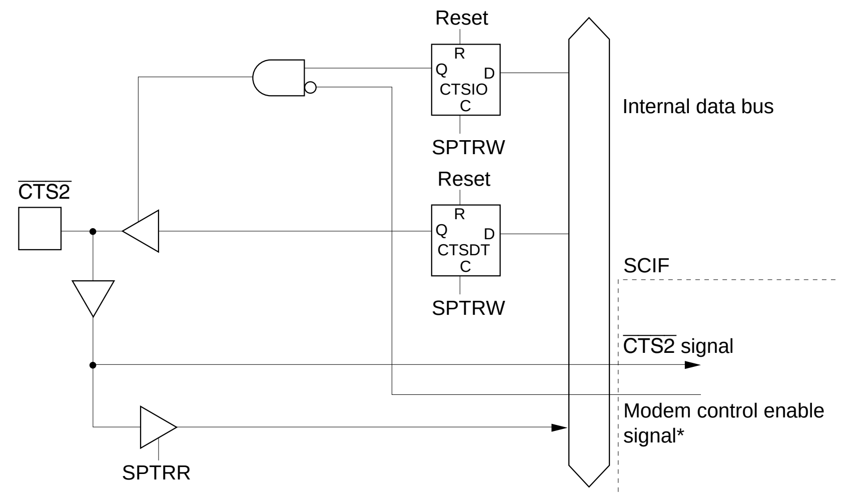


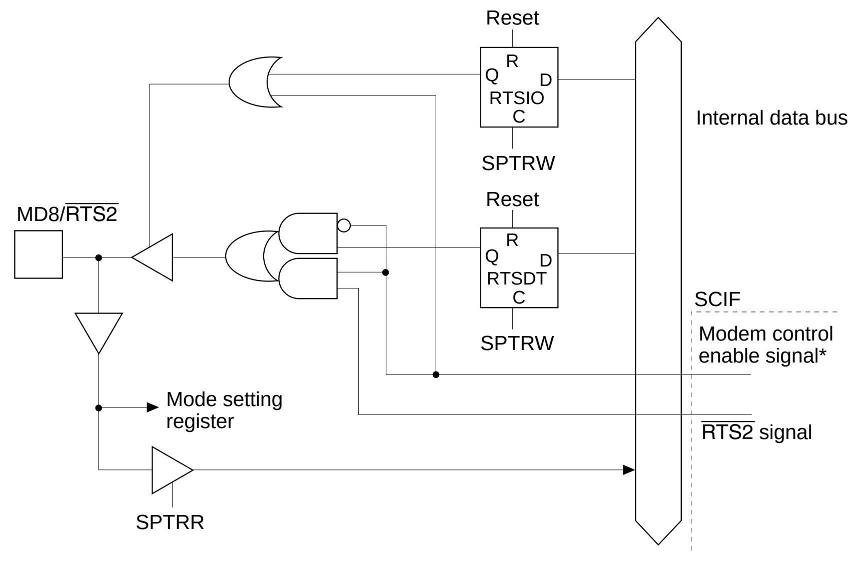
Figure 18.7 MD2/RxD2 Pin



SPTRW: Write to SPTR
SPTRR: Read SPTR

Note: * MCE bit in SCFCR2: signal that designates modem control as the $\overline{\text{CTS2}}$ pin function.

Figure 18.8 $\overline{\text{CTS2}}$ Pin



Reset
R
Q
D
RTSIO
C
SPTRW
Reset
R
Q
D
RTSDT
C
SPTRW
Internal data bus
MD8/$\overline{\text{RTS2}}$
SCIF
Modem control enable signal*
Mode setting register
$\overline{\text{RTS2}}$ signal
SPTRR

SPTRW: Write to SPTR
SPTRR: Read SPTR

Note: * MCE bit in SCFCR2: signal that designates modem control as the $\overline{\text{RTS2}}$ pin function.

Figure 18.9 MD8/$\overline{\text{RTS2}}$ Pin

18.1.3 Pin Configuration

Table 18.1 shows the 20-bit general-purpose I/O port pin configuration.

Table 18.1 20-Bit General-Purpose I/O Port Pins

Pin Name	Signal	I/O	Function
Port 19 pin	PORT19	I/O	I/O port
Port 18 pin	PORT18	I/O	I/O port
Port 17 pin	PORT17	I/O	I/O port
Port 16 pin	PORT16	I/O	I/O port
Port 15 pin	PORT15	I/O*	I/O port / GPIO interrupt
Port 14 pin	PORT14	I/O*	I/O port / GPIO interrupt
Port 13 pin	PORT13	I/O*	I/O port / GPIO interrupt
Port 12 pin	PORT12	I/O*	I/O port / GPIO interrupt
Port 11 pin	PORT11	I/O*	I/O port / GPIO interrupt
Port 10 pin	PORT10	I/O*	I/O port / GPIO interrupt
Port 9 pin	PORT9	I/O*	I/O port / GPIO interrupt
Port 8 pin	PORT8	I/O*	I/O port / GPIO interrupt
Port 7 pin	PORT7	I/O*	I/O port / GPIO interrupt
Port 6 pin	PORT6	I/O*	I/O port / GPIO interrupt
Port 5 pin	PORT5	I/O*	I/O port / GPIO interrupt
Port 4 pin	PORT4	I/O*	I/O port / GPIO interrupt
Port 3 pin	PORT3	I/O*	I/O port / GPIO interrupt
Port 2 pin	PORT2	I/O*	I/O port / GPIO interrupt
Port 1 pin	PORT1	I/O*	I/O port / GPIO interrupt
Port 0 pin	PORT0	I/O*	I/O port / GPIO interrupt

Note: * When port pins are used as GPIO interrupts, they must be set to input mode. The input setting can be made in the PCTRA register.

Table 18.2 shows the SCI I/O port pin configuration.

Table 18.2 SCI I/O Port Pins

Pin Name	Abbreviation	I/O	Function
Serial clock pin	MD0/SCK	I/O	Clock input/output
Receive data pin	RxD	Input	Receive data input
Transmit data pin	MD7/TxD	Output	Transmit data output

Note: Pins MD0/SCK and MD7/TxD function as mode input pins MD0 and MD7 after a power-on reset. They are made to function as serial pins by performing SCI operation settings with the TE, RE, CKEI, and CKE0 bits in SCSCR1 and the C/$\overline{\text{A}}$ bit in SCSMR1. Break state transmission and detection can be performed by means of a setting in the SCI's SCSPTR1 register.

Table 18.3 shows the SCIF I/O port pin configuration.

Table 18.3 SCIF I/O Port Pins

Pin Name	Abbreviation	I/O	Function
Serial clock pin	MRESET/SCK2	Input	Clock input
Receive data pin	MD2/RxD2	Input	Receive data input
Transmit data pin	MD1/TxD2	Output	Transmit data output
Modem control pin	$\overline{\text{CTS2}}$	I/O	Transmission enabled
Modem control pin	MD8/$\overline{\text{RTS2}}$	I/O	Transmission request

Note: The MRESET/SCK2 pin functions as the MRESET manual reset pin when a manual reset is executed. The MD1/TxD2, MD2/RxD2, and MD8/$\overline{\text{RTS2}}$ pins function as the MD1, MD2, and MD8 mode input pins after a power-on reset. These pins are made to function as serial pins by performing SCIF operation settings with the TE and RE bits in SCSCR2 and the MCE bit in SCFCR2. Break state transmission and detection can be set in the SCIF's SCSPTR2 register.

18.1.4 Register Configuration

The 20-bit general-purpose I/O port, SCI I/O port, and SCIF I/O port have seven registers, as shown in table 18.4.

Table 18.4 I/O Port Registers

Name	Abbreviation	R/W	Initial Value*	P4 Address	Area 7 Address	Access Size
Port control register A	PCTRA	R/W	H'00000000	H'FF80002C	H'1F80002C	32
Port data register A	PDTRA	R/W	Undefined	H'FF800030	H'1F800030	16
Port control register B	PCTRB	R/W	H'00000000	H'FF800040	H'1F800040	32
Port data register B	PDTRB	R/W	Undefined	H'FF800044	H'1F800044	16
GPIO interrupt control register	GPIOIC	R/W	H'00000000	H'FF800048	H'1F800048	16
Serial port register	SCSPTR1	R/W	Undefined	H'FFE0001C	H'1FE0001C	8
Serial port register	SCSPTR2	R/W	Undefined	H'FFE80020	H'1FE80020	16

Note: * Initialized by a power-on reset.

18.2 Register Descriptions

18.2.1 Port Control Register A (PCTRA)

Port control register A (PCTRA) is a 32-bit readable/writable register that controls the input/output direction and pull-up for each bit in the 16-bit port (port 15 pin to port 0 pin). As the initial value of port data register A (PDTRA) is undefined, all the bits in the 16-bit port should be set to output with PCTRA after writing a value to the PDTRA register.

PCTRA is initialized to H'00000000 by a power-on reset. It is not initialized by a manual reset or in standby mode, and retains its contents.

Bit:	31	30	29	28	27	26	25	24
	PB15PUP	PB15IO	PB14PUP	PB14IO	PB13PUP	PB13IO	PB12PUP	PB12IO
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
	PB11PUP	PB11IO	PB10PUP	PB10IO	PB9PUP	PB9IO	PB8PUP	PB8IO
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
	PB7PUP	PB7IO	PB6PUP	PB6IO	PB5PUP	PB5IO	PB4PUP	PB4IO
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
	PB3PUP	PB3IO	PB2PUP	PB2IO	PB1PUP	PB1IO	PB0PUP	PB0IO
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 2n + 1 (n = 0–15)—Port Pull-Up Control (PBnPUP): Specifies whether each bit in the 16-bit port is to be pulled up with a built-in resistor. Pull-up is automatically turned off for a port pin set to output by bit PBnIO.

Bit 2n + 1: PBnPUP	Description	
0	Bit m (m = 0–15) of 16-bit port is pulled up	(Initial value)
1	Bit m (m = 0–15) of 16-bit port is not pulled up	

Bit 2n (n = 0–15)—Port I/O Control (PBnIO): Specifies whether each bit in the 16-bit port is an input or an output.

Bit 2n: PBnIO	Description	
0	Bit m (m = 0–15) of 16-bit port is an input	(Initial value)
1	Bit m (m = 0–15) of 16-bit port is an output	

18.2.2 Port Data Register A (PDTRA)

Port data register A (PDTRA) is a 16-bit readable/writable register used as a data latch for each bit in the 16-bit port. When a bit is set as an output, the value written to the PDTRA register is output from the external pin. When a value is read from the PDTRA register while a bit is set as an input, the external pin value sampled on the external bus clock is read. When a bit is set as an output, the value written to the PDTRA register is read.

PDTR is not initialized by a power-on or manual reset, or in standby mode, and retains its contents.

Bit:	15	14	13	12	11	10	9	8
	PB15DT	PB14DT	PB13DT	PB12DT	PB11DT	PB10DT	PB9DT	PB8DT
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
	PB7DT	PB6DT	PB5DT	PB4DT	PB3DT	PB2DT	PB1DT	PB0DT
Initial value:	—	—	—	—	—	—	—	—
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

18.2.3 Port Control Register B (PCTRB)

Port control register B (PCTRB) is a 32-bit readable/writable register that controls the input/output direction and pull-up for each bit in the 4-bit port (port 19 pin to port 16 pin). As the initial value of port data register B (PDTRB) is undefined, each bit in the 4-bit port should be set to output with PCTRB after writing a value to the PDTRB register.

PCTRB is initialized to H'00000000 by a power-on reset. It is not initialized by a manual reset or in standby mode, and retains its contents.

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

Bit:	23	22	21	20	19	18	17	16
	—	—	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

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

Bit:	7	6	5	4	3	2	1	0
	PB19PUP	PB19IO	PB18PUP	PB18IO	PB17PUP	PB17IO	PB16PUP	PB16IO
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 2n + 1 (n = 0–3)—Port Pull-Up Control (PBnPUP): Specifies whether each bit in the 4-bit port is to be pulled up with a built-in resistor. Pull-up is automatically turned off for a port pin set to output by bit PBnIO.

Bit 2n + 1: PBnPUP	Description	
0	Bit m (m = 16–19) of 4-bit port is pulled up	(Initial value)
1	Bit m (m = 16–19) of 4-bit port is not pulled up	

Bit 2n (n = 0–3)—Port I/O Control (PBnIO): Specifies whether each bit in the 4-bit port is an input or an output.

Bit 2n: PBnIO	Description	
0	Bit m (m = 16–19) of 4-bit port is an input	(Initial value)
1	Bit m (m = 16–19) of 4-bit port is an output	

18.2.4 Port Data Register B (PDTRB)

Port data register B (PDTRB) is a 16-bit readable/writable register used as a data latch for each bit in the 4-bit port. When a bit is set as an output, the value written to the PDTRB register is output from the external pin. When a value is read from the PDTRB register while a bit is set as an input, the external pin value sampled on the external bus clock is read. When a bit is set as an output, the value written to the PDTRB register is read.

PDTRB is not initialized by a power-on or manual reset, or in standby mode, and retains its contents.

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

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	PB19DT	PB18DT	PB17DT	PB16DT
Initial value:	0	0	0	0	—	—	—	—
R/W:	R	R	R	R	R/W	R/W	R/W	R/W

18.2.5 GPIO Interrupt Control Register (GPIOIC)

The GPIO interrupt control register (GPIOIC) is a 16-bit readable/writable register that performs 16-bit interrupt input control.

GPIOIC is initialized to H'0000 by a power-on reset. It is not initialized by a manual reset or in standby mode, and retains its contents.

GPIO interrupts are active-low level interrupts. Bit-by-bit masking is possible, and the OR of all the bits set as GPIO interrupts is used for interrupt detection. Which bits interrupts are input to can be identified by reading the PDTRA register.

Bit:	15	14	13	12	11	10	9	8
	PTIREN15	PTIREN14	PTIREN13	PTIREN12	PTIREN11	PTIREN10	PTIREN9	PTIREN8
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
	PTIREN7	PTIREN6	PTIREN5	PTIREN4	PTIREN3	PTIREN2	PTIREN1	PTIREN0
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 n (n = 0–15)—Port Interrupt Enable (PTIRENn): Specifies whether interrupt input is performed for each bit.

Bit n: PTIRENn	Description
0	Port m (m = 0–15) of 16-bit port is used as a normal I/O port (Initial value)
1	Port m (m = 0–15) of 16-bit port is used as a GPIO interrupt*

Note: * When using an interrupt, set the corresponding port to input in the PCTRA register before making the PTIRENn setting.

18.2.6 Serial Port Register (SCSPTR1)

Bit:	7	6	5	4	3	2	1	0
	EIO	—	—	—	SPB1IO	SPB1DT	SPB0IO	SPB0DT
Initial value:	0	0	0	0	0	—	0	—
R/W:	R/W	—	—	—	R/W	R/W	R/W	R/W

The serial port register (SCSPTR1) is an 8-bit readable/writable register that controls input/output and data for the port pins multiplexed with the serial communication interface (SCI) pins. Input data can be read from the RxD pin, output data written to the TxD pin, and breaks in serial transmission/reception controlled, by means of bits 1 and 0. SCK pin data reading and output data writing can be performed by means of bits 3 and 2. Bit 7 controls enabling and disabling of the RXI interrupt.

SCSPTR1 can be read or written to by the CPU at all times. All SCSPTR1 bits except bits 2 and 0 are initialized to 0 by a power-on reset or manual reset; the value of bits 2 and 0 is undefined. SCSPTR1 is not initialized in the module standby state or standby mode.

Bit 7—Error Interrupt Only (EIO): See section 15.2.8, Serial Port Register (SCSPTR1).

Bits 6 to 4—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 3—Serial Port Clock Port I/O (SPB1IO): Specifies serial port SCK pin input/output. When the SCK pin is actually set as a port output pin and outputs the value set by the SPB1DT bit, the C/$\overline{\text{A}}$ bit in SCSMR1 and the CKE1 and CKE0 bits in SCSCR1 should be cleared to 0.

Bit 3: SPB1IO	Description	
0	SPB1DT bit value is not output to the SCK pin	(Initial value)
1	SPB1DT bit value is output to the SCK pin	

Bit 2—Serial Port Clock Port Data (SPB1DT): Specifies the serial port SCK pin input/output data. Input or output is specified by the SPB1IO bit (see the description of bit 3, SPB1IO, for details). When output is specified, the value of the SPB1DT bit is output to the SCK pin. The SCK pin value is read from the SPB1DT bit regardless of the value of the SPB1IO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

Bit 2: SPB1DT	Description
0	Input/output data is low-level
1	Input/output data is high-level

Bit 1—Serial Port Break I/O (SPB0IO): Specifies the serial port TxD pin output condition. When the TxD pin is actually set as a port output pin and outputs the value set by the SPB0DT bit, the TE bit in SCSCR1 should be cleared to 0.

Bit 1: SPB0IO	Description	
0	SPB0DT bit value is not output to the TxD pin	(Initial value)
1	SPB0DT bit value is output to the TxD pin	

Bit 0—Serial Port Break Data (SPB0DT): Specifies the serial port RxD pin input data and TxD pin output data. The TxD pin output condition is specified by the SPB0IO bit (see the description of bit 1, SPB0IO, for details). When the TxD pin is designated as an output, the value of the SPB0DT bit is output to the TxD pin. The RxD pin value is read from the SPB0DT bit regardless of the value of the SPB0IO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

Bit 0: SPB0DT	Description
0	Input/output data is low-level
1	Input/output data is high-level

18.2.7 Serial Port Register (SCSPTR2)

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

Bit:	7	6	5	4	3	2	1	0
	RTSIO	RTSDT	CTSIO	CTSDT	—	—	SPB2IO	SPB2DT
Initial value:	0	—	0	—	0	0	0	—
R/W:	R/W	R/W	R/W	R/W	R	R	R/W	R/W

The serial port register (SCSPTR2) is a 16-bit readable/writable register that controls input/output and data for the port pins multiplexed with the serial communication interface (SCIF) pins. Input data can be read from the RxD2 pin, output data written to the TxD2 pin, and breaks in serial transmission/reception controlled, by means of bits 1 and 0. $\overline{\text{CTS2}}$ pin data reading and output data writing can be performed by means of bits 5 and 4, and $\overline{\text{RTS2}}$ pin data reading and output data writing by means of bits 7 and 6.

SCSPTR2 can be read or written to by the CPU at all times. All SCSPTR2 bits except bits 6, 4, and 0 are initialized to 0 by a power-on reset or manual reset; the value of bits 6, 4, and 0 is undefined. SCSPTR2 is not initialized in standby mode or in the module standby state.

Bits 15 to 8—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 7—Serial Port RTS Port I/O (RTSIO): Specifies serial port $\overline{\text{RTS2}}$ pin input/output. When the $\overline{\text{RTS2}}$ pin is actually set as a port output pin and outputs the value set by the RTSDT bit, the MCE bit in SCFCR2 should be cleared to 0.

Bit 7: RTSIO	Description	
0	RTSDT bit value is not output to the $\overline{\text{RTS2}}$ pin	(Initial value)
1	RTSDT bit value is output to the $\overline{\text{RTS2}}$ pin	

Bit 6—Serial Port RTS Port Data (RTSDT): Specifies the serial port $\overline{\text{RTS2}}$ pin input/output data. Input or output is specified by the RTSIO pin (see the description of bit 7, RTSIO, for details). When the $\overline{\text{RTS2}}$ pin is designated as an output, the value of the RTSDT bit is output to the $\overline{\text{RTS2}}$ pin. The $\overline{\text{RTS2}}$ pin value is read from the RTSDT bit regardless of the value of the RTSIO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

Bit 6: RTSDT	Description
0	Input/output data is low-level
1	Input/output data is high-level

Bit 5—Serial Port CTS Port I/O (CTSIO): Specifies serial port $\overline{\text{CTS2}}$ pin input/output. When the $\overline{\text{CTS2}}$ pin is actually set as a port output pin and outputs the value set by the CTSDT bit, the MCE bit in SCFCR2 should be cleared to 0.

Bit 5: CTSIO	Description	
0	CTSDT bit value is not output to the $\overline{\text{CTS2}}$ pin	(Initial value)
1	CTSDT bit value is output to the $\overline{\text{CTS2}}$ pin	

Bit 4—Serial Port CTS Port Data (CTSDT): Specifies the serial port $\overline{\text{CTS2}}$ pin input/output data. Input or output is specified by the CTSIO pin (see the description of bit 5, CTSIO, for details). When the $\overline{\text{CTS2}}$ pin is designated as an output, the value of the CTSDT bit is output to the $\overline{\text{CTS2}}$ pin. The $\overline{\text{CTS2}}$ pin value is read from the CTSDT bit regardless of the value of the CTSIO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

Bit 4: CTSDT	Description
0	Input/output data is low-level
1	Input/output data is high-level

Bits 3 and 2—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 1—Serial Port Break I/O (SPB2IO): Specifies the serial port TxD2 pin output condition. When the TxD2 pin is actually set as a port output pin and outputs the value set by the SPB2DT bit, the TE bit in SCSCR2 should be cleared to 0.

Bit 1: SPB2IO	Description	
0	SPB2DT bit value is not output to the TxD2 pin	(Initial value)
1	SPB2DT bit value is output to the TxD2 pin	

Bit 0—Serial Port Break Data (SPB2DT): Specifies the serial port RxD2 pin input data and TxD2 pin output data. The TxD2 pin output condition is specified by the SPB2IO bit (see the description of bit 1, SPB2IO, for details). When the TxD2 pin is designated as an output, the value of the SPB2DT bit is output to the TxD2 pin. The RxD2 pin value is read from the SPB2DT bit regardless of the value of the SPB2IO bit. The initial value of this bit after a power-on reset or manual reset is undefined.

Bit 0: SPB2DT	Description
0	Input/output data is low-level
1	Input/output data is high-level

Section 19 Interrupt Controller (INTC)

19.1 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 handle interrupt requests according to user-set priority.

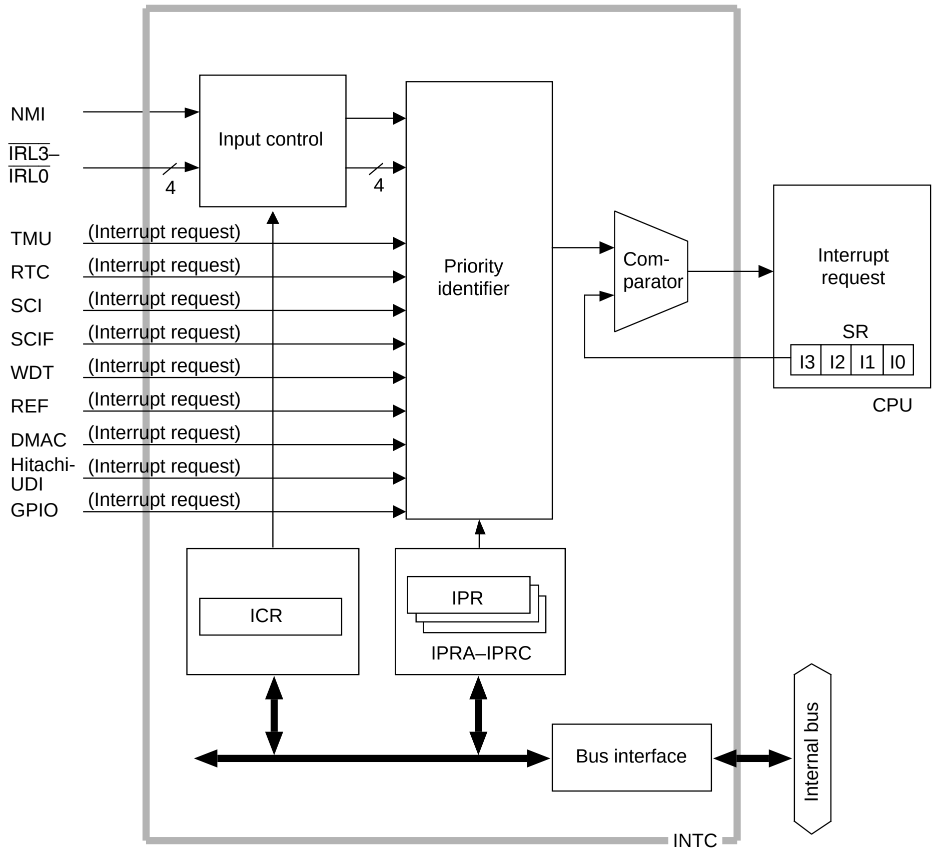
19.1.1 Features

The INTC has the following features.

- Fifteen interrupt priority levels can be set
  By setting the three interrupt priority registers, the priorities of on-chip peripheral module interrupts can be selected from 15 levels for different request sources.
- NMI noise canceler function
  The NMI input level bit indicates the NMI pin state. The pin state can be checked by reading this bit in the interrupt exception handler, enabling it to be used as a noise canceler.
- NMI request masking when SR.BL bit is set
  It is possible to select whether or not NMI requests are to be masked when the SR.BL bit is set.

19.1.2 Block Diagram

Figure 19.1 shows a block diagram of the INTC.



TMU:	Timer unit
RTC:	Realtime clock unit
SCI:	Serial communication interface
SCIF:	Serial communication interface with FIFO
WDT:	Watchdog timer
REF:	Memory refresh controller section of the bus state controller
DMAC:	Direct memory access controller
Hitachi-UDI:	Hitachi-UDI unit
GPIO:	I/O port
ICR:	Interrupt control register
IPRA–IPRC:	Interrupt priority registers A–C
SR:	Status register

Figure 19.1 Block Diagram of INTC

19.1.3 Pin Configuration

Table 19.1 shows the INTC pin configuration.

Table 191 INTC Pins

Pin Name	Abbreviation	I/O	Function
Nonmaskable interrupt input pin	NMI	Input	Input of nonmaskable interrupt request signal
Interrupt input pins	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$	Input	Input of interrupt request signals (maskable by I3–I0 in SR)

19.1.4 Register Configuration

The INTC has the registers shown in table 19.2.

Table 19.2 INTC Registers

Name	Abbreviation	R/W	Initial Value*1	P4 Address	Area 7 Address	Access Size
Interrupt control register	ICR	R/W	*2	H'FFD00000	H'1FD00000	16
Interrupt priority register A	IPRA	R/W	H'0000	H'FFD00004	H'1FD00004	16
Interrupt priority register B	IPRB	R/W	H'0000	H'FFD00008	H'1FD00008	16
Interrupt priority register C	IPRC	R/W	H'0000	H'FFD0000C	H'1FD0000C	16

Notes: 1. Initialized by a power-on reset or manual reset.
2. H'8000 when the NMI pin is high, H'0000 when the NMI pin is low.

19.2 Interrupt Sources

There are three types of interrupt sources: NMI, RL, and on-chip peripheral modules. Each interrupt has a priority level (16–0), with level 16 as the highest and level 1 as the lowest. When level 0 is set, the interrupt is masked and interrupt requests are ignored.

19.2.1 NMI Interrupt

The NMI interrupt has the highest priority level of 16. It is always accepted unless the BL bit in the status register in the CPU is set to 1. In sleep or standby mode, the interrupt is accepted even if the BL bit is set to 1.

A setting can also be made to have the NMI interrupt accepted even if the BL bit is set to 1.

Input from the NMI pin is edge-detected. The NMI edge select bit (NMIE) in the interrupt control register (ICR) is used to select either rising or falling edge. When the NMIE bit in the ICR register is modified, the NMI interrupt is not detected for a maximum of 6 bus clock cycles after the modification.

NMI interrupt exception handling does not affect the interrupt mask level bits (I3–I0) in the status register (SR).

19.2.2 IRL Interrupts

IRL interrupts are input by level at pins $\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$. The priority level is the level indicated by pins $\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$. An $\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ 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).

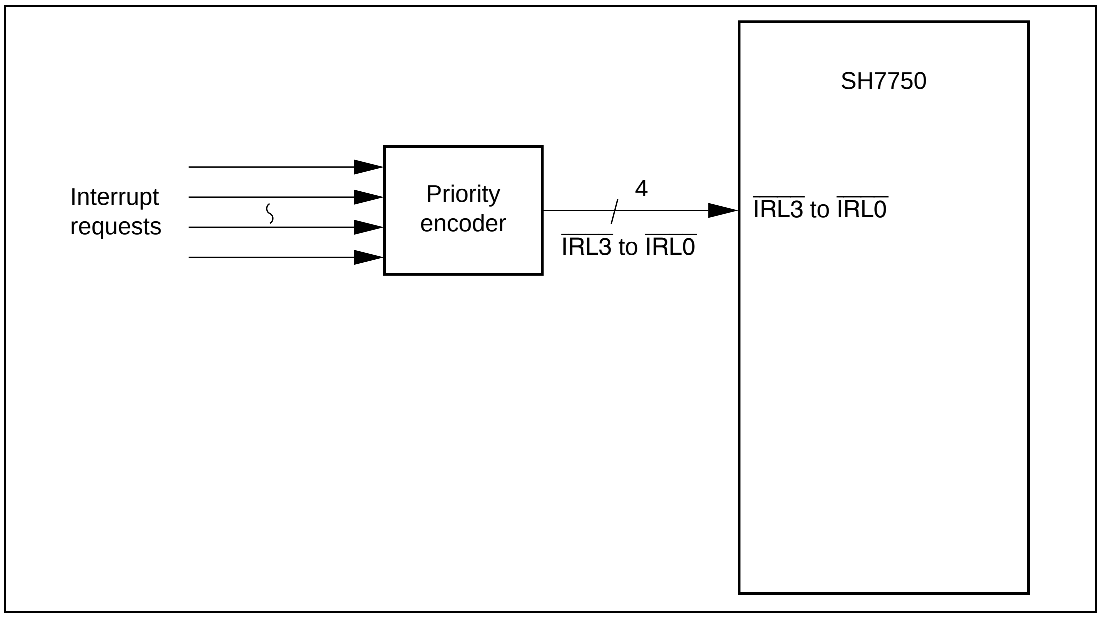

SH7750
Interrupt requests
Priority encoder
4
$\overline{\text{IRL3}}$ to $\overline{\text{IRL0}}$
$\overline{\text{IRL3}}$ to $\overline{\text{IRL0}}$

Figure 19.2 Example of IRL Interrupt Connection

Table 19.3 $\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ Pins and Interrupt Levels

$\overline{\text{IRL3}}$	$\overline{\text{IRL2}}$	$\overline{\text{IRL1}}$	$\overline{\text{IRL0}}$	**Interrupt Priority Level**	**Interrupt Request**
0	0	0	0	15	Level 15 interrupt request
			1	14	Level 14 interrupt request
		1	0	13	Level 13 interrupt request
			1	12	Level 12 interrupt request
	1	0	0	11	Level 11 interrupt request
			1	10	Level 10 interrupt request
		1	0	9	Level 9 interrupt request
			1	8	Level 8 interrupt request
1	0	0	0	7	Level 7 interrupt request
			1	6	Level 6 interrupt request
		1	0	5	Level 5 interrupt request
			1	4	Level 4 interrupt request
	1	0	0	3	Level 3 interrupt request
			1	2	Level 2 interrupt request
		1	0	1	Level 1 interrupt request
			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 bus clock cycle remain unchanged for three consecutive cycles, so that no transient level on the IRL pin change is detected. In standby mode, as the bus clock is stopped, noise cancellation is performed using the 32.768 kHz clock for the RTC instead. When the RTC is not used, therefore, 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 handling 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 IRL interrupt handling.

Pins $\overline{\text{IRL0}}$–$\overline{\text{IRL3}}$ can be used for four independent interrupt requests by setting the IRLM bit to 1 in the ICR register.

Table 19.4 $\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ Pins and Interrupt Levels (When IRLM = 1)

$\overline{\text{IRL3}}$	$\overline{\text{IRL2}}$	$\overline{\text{IRL1}}$	$\overline{\text{IRL0}}$	Interrupt Priority Level	Interrupt Request
1/0	1/0	1/0	0	13	IRL0
1/0	1/0	0	1	10	IRL1
1/0	0	1	1	7	IRL2
0	1	1	1	4	IRL3

19.2.3 On-Chip Peripheral Module Interrupts

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

- Hitachi-UDI unit (Hitachi-UDI)
- Direct memory access controller (DMAC)
- Timer unit (TMU)
- Realtime clock (RTC)
- Serial communication interface (SCI)
- Serial communication interface with FIFO (SCIF)
- Bus state controller (BSC)
- Watchdog timer (WDT)
- I/O port (GPIO)

Not every interrupt source is assigned a different interrupt vector, bus sources are reflected in the interrupt event register (INTEVT), so it is easy to identify sources by using the INTEVT register value as a branch offset in the exception handling routine.

A priority level from 15 to 0 can be set for each module by means of interrupt priority registers A to C (IPRA–IPRC).

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

On-chip peripheral module interrupt source flag and interrupt enable flag updating should only be carried out when the BL bit in the status register (SR) is set to 1. To prevent acceptance of an erroneous interrupt from an interrupt source that should have been updated, first read the on-chip peripheral register containing the relevant flag, then clear the BL bit to 0. This will secure the necessary timing internally. When updating a number of flags, there is no problem if only the register containing the last flag updated is read.

If flag updating is performed while the BL bit is cleared to 0, the program may jump to the interrupt handling routine when the INTEVT register value is 0. In this case, interrupt handling is initiated due to the timing relationship between the flag update and interrupt request recognition within the chip. Processing can be continued without any problem by executing an RTE instruction.

19.2.4 Interrupt Exception Handling and Priority

Table 19.5 lists the codes for the interrupt event register (INTEVT), and the order of interrupt priority. Each interrupt source is assigned a unique INTEVT code. The start address of the interrupt handler is common to each interrupt source. This is why, for instance, the value of INTEVT is used as an offset at the start of the interrupt handler and branched to in order to identify the interrupt source.

The order of priority of the on-chip peripheral modules is specified as desired by setting priority levels from 0 to 15 in interrupt priority registers A to C (IPRA–IPRC). The order of priority of the on-chip peripheral modules is set to 0 by a reset.

When the priorities for multiple interrupt sources are set to the same level and such interrupts are generated simultaneously, they are handled according to the default priority order shown in table 19.5.

Updating of interrupt priority registers A to C should only be carried out when the BL bit in the status register (SR) is set to 1. To prevent erroneous interrupt acceptance, first read one of the interrupt priority registers, then clear the BL bit to 0. This will secure the necessary timing internally.

Table 19.5 Interrupt Exception Handling Sources and Priority Order

Interrupt Source		INTEVT Code	Interrupt Priority (Initial Value)	IPR (Bit Numbers)	Priority within IPR Setting Unit	Default Priority
NMI		H'1C0	16	—	—	High
IRL	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = 0	H'200	15	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = 1	H'220	14	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = 2	H'240	13	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = 3	H'260	12	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = 4	H'280	11	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = 5	H'2A0	10	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = 6	H'2C0	9	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = 7	H'2E0	8	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = 8	H'300	7	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = 9	H'320	6	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = A	H'340	5	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = B	H'360	4	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = C	H'380	3	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = D	H'3A0	2	—	—	
	$\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ = E	H'3C0	1	—	—	
	IRL0	H'240	13	—	—	
	IRL1	H'2A0	10	—	—	
	IRL2	H'300	7	—	—	
	IRL3	H'360	4	—	—	
Hitachi-UDI	Hitachi-UDI	H'600	15–0 (0)	IPRC (3–0)	—	
GPIO	GPIOI	H'620	15–0 (0)	IPRC (15–12)	—	
DMAC	DMTE0	H'640	15–0 (0)	IPRC (11–8)	High	
	DMTE1	H'660				
	DMTE2	H'680				
	DMTE3	H'6A0				
	DMAE	H'6C0			Low	
TMU0	TUNI0	H'400	15–0 (0)	IPRA (15–12)	—	
TMU1	TUNI1	H'420	15–0 (0)	IPRA (11–8)	—	
TMU2	TUNI2	H'440	15–0 (0)	IPRA (7–4)	High	
	TICPI2	H'460			Low	Low

Table 19.5 Interrupt Exception Handling Sources and Priority Order (cont)

Interrupt Source		INTEVT Code	Interrupt Priority (Initial Value)	IPR (Bit Numbers)	Priority within IPR Setting Unit	Default Priority
RTC	ATI	H'480	15–0 (0)	IPRA (3–0)	High ↕ Low	High
	PRI	H'4A0				
	CUI	H'4C0				
SCI1	ERI	H'4E0	15–0 (0)	IPRB (7–4)	High	
	RXI	H'500				
	TXI	H'520				
	TEI	H'540			Low	
SCIF	ERI	H'700	15–0 (0)	IPRC (7–4)	High	
	RXI	H'720				
	BRI	H'740				
	TXI	H'760			Low	
WDT	ITI	H'560	15–0 (0)	IPRB (15–12)	—	
REF	RCMI	H'580	15–0 (0)	IPRB (11–8)	High	
	ROVI	H'5A0			Low	Low

Note: TUNI0–TUNI2: Underflow interrupts
TICPI2: Input capture interrupt
ATI: Alarm interrupt
PRI: Periodic interrupt
CUI: Carry-up interrupt
ERI: Receive-error interrupt
RXI: Receive-data-full interrupt
TXI: Transmit-data-empty interrupt
TEI: Transmit-end interrupt
BRI: Break interrupt request
ITI: Interval timer interrupt
RCMI: Compare-match interrupt
ROVI: Refresh counter overflow interrupt
Hitachi-UDI: Hitachi-UDI interrupt
GPIOI: I/O port interrupt
DMTE0–DMTE3: DMAC transfer end interrupts
DMAE: DMAC address error interrupt

19.3 Register Descriptions

19.3.1 Interrupt Priority Registers A to C (IPRA–IPRC)

Interrupt priority registers A to C (IPRA–IPRC) are 16-bit readable/writable registers that set priority levels from 0 to 15 for on-chip peripheral module interrupts. These registers are initialized to H'0000 by a reset. They are 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 19.6 shows the relationship between the interrupt request sources and the IPRA–IPRC register bits.

Table 19.6 Interrupt Request Sources and IPRA–IPRC Registers

	Bits			
Register	**15–12**	**11–8**	**7–4**	**3–0**
Interrupt priority register A	TMU0	TMU1	TMU2	RTC
Interrupt priority register B	WDT	REF*[1]	SCI1	Reserved*[2]
Interrupt priority register C	GPIO	DMAC	SCIF	Hitachi-UDI

Notes: 1. REF is the memory refresh unit in the bus state controller (BSC). See section 13, Bus State Controller (BSC), for details.

2. Reserved bits: These bits are always read as 0 and should always be written with 0.

As shown in table 19.6, four on-chip peripheral modules are assigned to each register. Interrupt priority levels are established by setting a value from H'F (1111) to H'0 (0000) in each of the four-bit groups: 15–12, 11–8, 7–4, and 3–0. Setting H'F designates priority level 15 (the highest level), and setting H'0 designates priority level 0 (requests are masked).

19.3.2 Interrupt Control Register (ICR)

The interrupt control register (ICR) is a 16-bit register that sets the input signal detection mode for external interrupt input pin NMI and indicates the input signal level at the NMI pin. This register is initialized by a power-on reset or manual reset. It is not initialized in standby mode.

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

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

Note: * 1 when NMI pin input is high, 0 when low.

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. It cannot be modified.

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

Bit 14—NMI Interrupt Mask (MAI): Specifies whether or not all interrupts are to be masked while the NMI pin input level is low, irrespective of the CPU's SR.BL bit.

Bit 14: MAI	Description	
0	Interrupts enabled even while NMI pin is low	(Initial value)
1	Interrupts disabled while NMI pin is low*	

Note: * NMI interrupts are accepted in normal operation and in sleep mode.
In standby mode, all interrupts are masked, and standby is not cleared, while the NMI pin is low.

Bit 9—NMI Block Mode (NMIB): Specifies whether an NMI request is to be held pending or detected immediately while the SR.BL bit is set to 1.

Bit 9: NMIB	Description
0	NMI interrupt requests held pending while SR.BL bit is set to 1 (Initial value)
1	NMI interrupt requests detected while SR.BL bit is set to 1

Notes: 1. If interrupt requests are enabled while SR.BL = 1, the previous exception information will be lost, and so must be saved beforehand.
2. This bit is cleared automatically by NMI acceptance.

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

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

Bit 7—IRL Pin Mode (IRLM): Specifies whether pins $\overline{\text{IRL3}}$–$\overline{\text{IRL0}}$ are to be used as level-encoded interrupt requests or as four independent interrupt requests.

Bit 7: IRLM	Description
0	$\overline{\text{IRL}}$ pins used as level-encoded interrupt requests (Initial value)
1	$\overline{\text{IRL}}$ pins used as four independent interrupt requests

Bits 13 to 10 and 6 to 0—Reserved: These bits are always read as 0, and should only be written with 0.

19.4 INTC Operation

19.4.1 Interrupt Operation Sequence

The sequence of operations when an interrupt is generated is described below. Figure 19.3 shows a flowchart of the operations.

1. The interrupt request sources send interrupt request signals to the interrupt controller.
2. The interrupt controller selects the highest-priority interrupt from the interrupt requests sent, according to the priority levels set in interrupt priority registers A to C (IPRA–IPRC). Lower-priority interrupts are held pending. If two of these interrupts have the same priority level, or if multiple interrupts occur within a single module, the interrupt with the highest priority according to table 19.5, Interrupt Exception Handling Sources and Priority Order, 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 that the level in bits I3–I0, the interrupt controller accepts the interrupt and sends an interrupt request signal to the CPU.
4. The CPU accepts an interrupt at a break between instructions.
5. The interrupt source code is set in the interrupt event register (INTEVT).
6. The status register (SR) and program counter (PC) are saved to SSR and SPC, respectively.
7. The block bit (BL), mode bit (MD), and register bank bit (RB) in SR are set to 1.
8. The CPU jumps to the start address of the interrupt handler (the sum of the value set in the vector base register (VBR) and H'00000600).

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 handling routine for the particular 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 SH7750.

2. The interrupt source flag should be cleared in the interrupt handler. To ensure that an interrupt request that should have been cleared is not inadvertently accepted again, read the interrupt source flag after it has been cleared, then wait for the interval shown in table 19.7 (Time for priority decision and SR mask bit comparison) before clearing the BL bit or executing an RTE instruction.

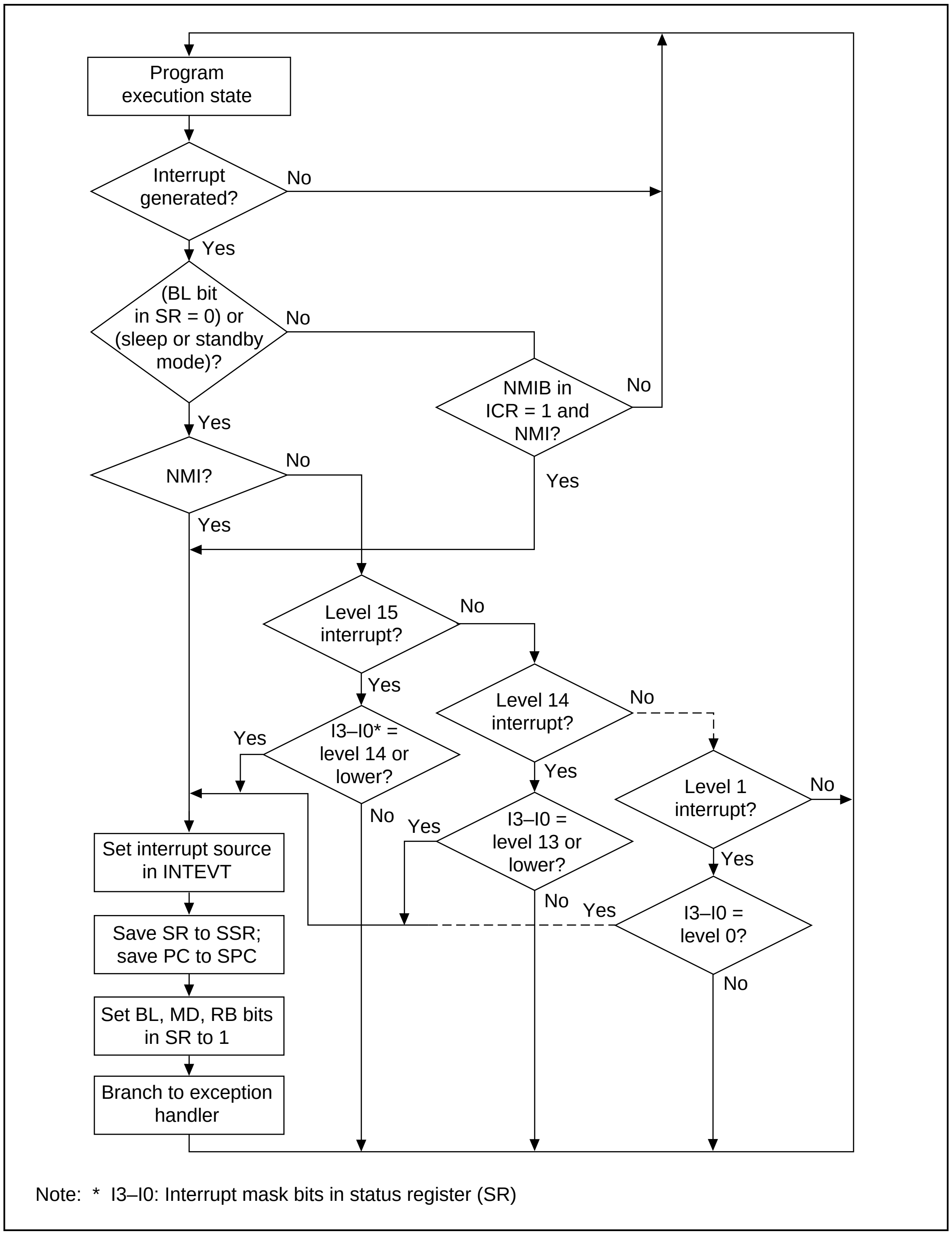

Program execution state
Interrupt generated?
No
Yes
(BL bit in SR = 0) or (sleep or standby mode)?
No
Yes
NMIB in ICR = 1 and NMI?
No
Yes
NMI?
No
Yes
Level 15 interrupt?
No
Yes
Level 14 interrupt?
No
Yes
I3–I0* = level 14 or lower?
Yes
No
Level 1 interrupt?
No
Yes
I3–I0 = level 13 or lower?
Yes
No
I3–I0 = level 0?
Yes
No
Set interrupt source in INTEVT
Save SR to SSR; save PC to SPC
Set BL, MD, RB bits in SR to 1
Branch to exception handler
Note: * I3–I0: Interrupt mask bits in status register (SR)

Figure 19.3 Interrupt Operation Flowchart

19.4.2 Multiple Interrupts

When handling multiple interrupts, interrupt handling should include the following procedures:

1. Branch to a specific interrupt handler corresponding to a code set in the INTEVT register. The code in INTEVT can be used as a branch-offset for branching to the specific handler.
2. Clear the interrupt source in the corresponding interrupt handler.
3. Save SPC and SSR to the stack.
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. Set the BL bit in SR to 1.
7. Restore SSR and SPC from memory.
8. 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. This enables the interrupt response time to be shortened for urgent processing.

19.4.3 Interrupt Masking with MAI Bit

By setting the MAI bit to 1 in the ICR register, it is possible to mask interrupts while the NMI pin is low, irrespective of the BL and IMASK bits in the SR register.

- In normal operation and sleep mode

  All interrupts are masked while the NMI pin is low. However, an NMI interrupt only is generated by a transition at the NMI pin.
- In standby mode

  All interrupts are masked while the NMI pin is low, and an NMI interrupt is not generated by a transition at the NMI pin. Therefore, standby cannot be cleared by an NMI interrupt while the MAI bit is set to 1.

19.5 Interrupt Response Time

The time from generation of an interrupt request until interrupt exception handling is performed and fetching of the first instruction of the exception handler is started (the interrupt response time) is shown in table 19.7.

Table 19.7 Interrupt Response Time

		Number of States			
Item		**NMI**	**RL**	**Peripheral Modules**	**Notes**
Time for priority decision and SR mask bit comparison		1Icyc + 4Bcyc	1Icyc + 7Bcyc	1Icyc + 2Bcyc	
Wait time until end of sequence being executed by CPU		S – 1 (≥ 0) × Icyc	S – 1 (≥ 0) × Icyc	S – 1 (≥ 0) × Icyc	
Time from interrupt exception handling (save of SR and PC) until fetch of first instruction of exception handler is started		4 × Icyc	4 × Icyc	4 × Icyc	
Response time	Total	5Icyc + 4Bcyc + (S – 1)Icyc	5Icyc + 7Bcyc + (S – 1)Icyc	5Icyc + 2Bcyc + (S – 1)Icyc	
	Minimum case	13Icyc	19Icyc	9Icyc	When Icyc: Bcyc = 2:1
	Maximum case	36 + S Icyc	60 + S Icyc	20 + S Icyc	When Icyc: Bcyc = 8:1

Icyc:	One cycle of internal clock supplied to CPU, etc.
Bcyc:	One CKIO cycle
S:	Latency of instruction

Section 20 User Break Controller (UBC)

20.1 Overview

The user break controller (UBC) provides functions that simplify program debugging. When break conditions are set in the UBC, a user break interrupt is generated according to the contents of the bus cycle generated by the CPU. This function makes it easy to design an effective self-monitoring debugger, enabling programs to be debugged with the chip alone, without using an in-circuit emulator.

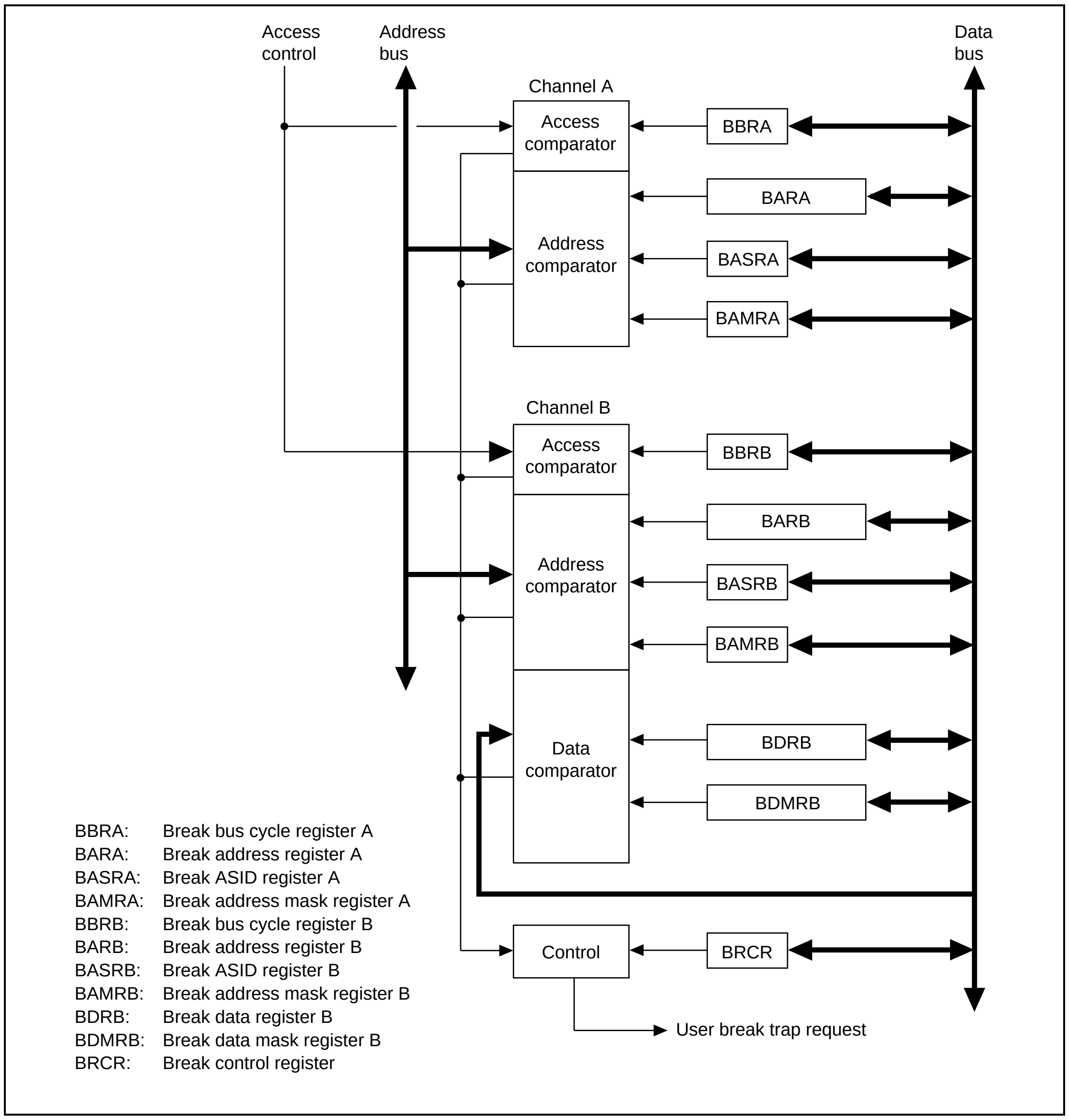
20.1.1 Features

The UBC has the following features.

- Two break channels (A and B)
  User break interrupts can be generated on independent conditions for channels A and B, or on sequential conditions (sequential break setting: channel A → channel B).
- The following can be set as break compare conditions:
  - Address (selection of 32-bit virtual address and ASID for comparison):
    Address: All bits compared/lower 10 bits masked/lower 12 bits masked/lower 16 bits masked/lower 20 bits masked/all bits masked
    ASID: All bits compared/all bits masked
  - Data (channel B only, 32-bit mask capability)
  - Bus cycle: Instruction access/operand access
  - Read/write
  - Operand size: Byte/word/longword/quadword
- An instruction access cycle break can be effected before or after the instruction is executed.

20.1.2 Block Diagram

Figure 20.1 shows a block diagram of the UBC.



Access control
Address bus
Data bus
Channel A
Access comparator
Address comparator
BBRA
BARA
BASRA
BAMRA
Channel B
Access comparator
Address comparator
Data comparator
BBRB
BARB
BASRB
BAMRB
BDRB
BDMRB
Control
BRCR
User break trap request
BBRA: Break bus cycle register A
BARA: Break address register A
BASRA: Break ASID register A
BAMRA: Break address mask register A
BBRB: Break bus cycle register B
BARB: Break address register B
BASRB: Break ASID register B
BAMRB: Break address mask register B
BDRB: Break data register B
BDMRB: Break data mask register B
BRCR: Break control register

Figure 20.1 Block Diagram of User Break Controller

Table 20.1 shows the UBC registers.

Table 20.1 UBC Registers

Name	Abbreviation	R/W	Initial Value	P4 Address	Area 7 Address	Access Size
Break address register A	BARA	R/W	Undefined	H'FF200000	H'1F200000	32
Break address mask register A	BAMRA	R/W	Undefined	H'FF200004	H'1F200004	8
Break bus cycle register A	BBRA	R/W	H'0000	H'FF200008	H'1F200008	16
Break ASID register A	BASRA	R/W	Undefined	H'FF000014	H'1F000014	8
Break address register B	BARB	R/W	Undefined	H'FF20000C	H'1F20000C	32
Break address mask register B	BAMRB	R/W	Undefined	H'FF200010	H'1F200010	8
Break bus cycle register B	BBRB	R/W	H'0000	H'FF200014	H'1F200014	16
Break ASID register B	BASRB	R/W	Undefined	H'FF000018	H'1F000018	8
Break data register B	BDRB	R/W	Undefined	H'FF200018	H'1F200018	32
Break data mask register B	BDMRB	R/W	Undefined	H'FF20001C	H'1F20001C	32
Break control register	BRCR	R/W	H'0000*	H'FF200020	H'1F200020	16

Note: * Some bits are not initialized. See section 20.2.12, Break Control Register (BRCR), for details.

20.2 Register Descriptions

20.2.1 Access to UBC Control Registers

The access size must be the same as the control register size. If the sizes are different, a write will not be effected in a UBC register write operation, and a read operation will return an undefined value. UBC control register contents cannot be transferred to a floating-point register using a floating-point memory load instruction.

When a UBC control register is updated, use either of the following methods to make the updated value valid:

1. Execute an RTE instruction after the memory store instruction that updated the register. The updated value will be valid from the RTE instruction jump destination onward.
2. Execute instructions requiring 5 states for execution after the memory store instruction that updated the register. As the SH7750 executes two instructions in parallel and a minimum of 0.5 state is required for execution of one instruction, 11 instructions must be inserted. The updated value will be valid from the 6th state onward.

20.2.2 Break Address Register A (BARA)

Bit:	31	30	29	28	27	26	25	24
	BAA31	BAA30	BAA29	BAA28	BAA27	BAA26	BAA25	BAA24
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
	BAA23	BAA22	BAA21	BAA20	BAA19	BAA18	BAA17	BAA16
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
	BAA15	BAA14	BAA13	BAA12	BAA11	BAA10	BAA9	BAA8
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
	BAA7	BAA6	BAA5	BAA4	BAA3	BAA2	BAA1	BAA0
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: *: Undefined

Break address register A (BARA) is a 32-bit readable/writable register that specifies the virtual address used in the channel A break conditions. BARA is not initialized by a power-on reset or manual reset.

Bits 31 to 0—Break Address A31 to A0 (BAA31–BAA0): These bits hold the virtual address (bits 31–0) used in the channel A break conditions.

20.2.3 Break ASID Register A (BASRA)

Bit:	7	6	5	4	3	2	1	0
	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

Note: *: Undefined

Break ASID register A (BASRA) is an 8-bit readable/writable register that specifies the ASID used in the channel A break conditions. BASRA is not initialized by a power-on reset or manual reset.

Bits 7 to 0—Break ASID A7 to A0 (BASA7–BASA0): These bits hold the ASID (bits 7–0) used in the channel A break conditions.

20.2.4 Break Address Mask Register A (BAMRA)

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	BAMA2	BASMA	BAMA1	BAMA0
Initial value:	0	0	0	0	*	*	*	*
R/W:	R	R	R	R	R/W	R/W	R/W	R/W

Note: *: Undefined

Break address mask register A (BAMRA) is an 8-bit readable/writable register that specifies which bits are to be masked in the break ASID set in BASRA and the break address set in BARA. BAMRA is not initialized by a power-on reset or manual reset.

Bits 7 to 4—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 2—Break ASID Mask A (BASMA): Specifies whether all bits of the channel A break ASID (BASA7–BASA0) are to be masked.

Bit 2: BASMA	Description
0	All BASRA bits are included in break conditions
1	No BASRA bits are included in break conditions

Bits 3, 1, and 0—Break Address Mask A2 to A0 (BAMA2–BAMA0): These bits specify which bits of the channel A break address (BAA31–BAA0) set in BARA are to be masked.

Bit 3: BAMA2	Bit 1: BAMA1	Bit 0: BAMA0	Description
0	0	0	All BARA bits are included in break conditions
		1	Lower 10 bits of BARA are masked, and not included in break conditions
	1	0	Lower 12 bits of BARA are masked, and not included in break conditions
		1	All BARA bits are masked, and not included in break conditions
1	0	0	Lower 16 bits of BARA are masked, and not included in break conditions
		1	Lower 20 bits of BARA are masked, and not included in break conditions
	1	*	Reserved (cannot be set)

Note: *: Don’t care

20.2.5 Break Bus Cycle Register A (BBRA)

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

Bit:	7	6	5	4	3	2	1	0
	—	SZA2	IDA1	IDA0	RWA1	RWA0	SZA1	SZA0
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

Break bus cycle register A (BBRA) is a 16-bit readable/writable register that sets three conditions—(1) instruction access/operand access, (2) read/write, and (3) operand size—from among the channel A break conditions.

BBRA is initialized to H'0000 by a power-on reset. It retains its value in standby mode.

Bits 15 to 7—Reserved: These bits are always read as 0, and should only be written with 0.

Bits 5 and 4—Instruction Access/Operand Access Select A (IDA1, IDA0): These bits specify whether an instruction access cycle or an operand access cycle is used as the bus cycle in the channel A break conditions.

Bit 5: IDA1	Bit 4: IDA0	Description
0	0	Condition comparison is not performed (Initial value)
	1	Instruction access cycle is used as break condition
1	0	Operand access cycle is used as break condition
	1	Instruction access cycle or operand access cycle is used as break condition

Bits 3 and 2—Read/Write Select A (RWA1, RWA0): These bits specify whether a read cycle or write cycle is used as the bus cycle in the channel A break conditions.

Bit 3: RWA1	Bit 2: RWA0	Description
0	0	Condition comparison is not performed (Initial value)
	1	Read cycle is used as break condition
1	0	Write cycle is used as break condition
	1	Read cycle or write cycle is used as break condition

Bits 6, 1, and 0—Operand Size Select A (SZA2–SZA0): These bits select the operand size of the bus cycle used as a channel A break condition.

Bit 6: SZA2	Bit 1: SZA1	Bit 0: SZA0	Description
0	0	0	Operand size is not included in break conditions (Initial value)
		1	Byte access is used as break condition
	1	0	Word access is used as break condition
		1	Longword access is used as break condition
1	0	0	Quadword access is used as break condition
		1	Reserved (cannot be set)
	1	*	Reserved (cannot be set)

Note: *: Don't care

20.2.6 Break Address Register B (BARB)

BARB is the channel B break address register. The bit configuration is the same as for BARA.

20.2.7 Break ASID Register B (BASRB)

BASRB is the channel B break ASID register. The bit configuration is the same as for BASRA.

20.2.8 Break Address Mask Register B (BAMRB)

BAMRB is the channel B break address mask register. The bit configuration is the same as for BAMRA.

20.2.9 Break Data Register B (BDRB)

Bit:	31	30	29	28	27	26	25	24
	BDB31	BDB30	BDB29	BDB28	BDB27	BDB26	BDB25	BDB24
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
	BDB23	BDB22	BDB21	BDB20	BDB19	BDB18	BDB17	BDB16
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
	BDB15	BDB14	BDB13	BDB12	BDB11	BDB10	BDB9	BDB8
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
	BDB7	BDB6	BDB5	BDB4	BDB3	BDB2	BDB1	BDB0
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: *: Undefined

Break data register B (BDRB) is a 32-bit readable/writable register that specifies the data (bits 31–0) to be used in the channel B break conditions. BDRB is not initialized by a power-on reset or manual reset.

Bits 31 to 0—Break Data B31 to B0 (BDB31–BDB0): These bits hold the data (bits 31–0) to be used in the channel B break conditions.

20.2.10 Break Data Mask Register B (BDMRB)

Bit:	31	30	29	28	27	26	25	24
	BDMB31	BDMB30	BDMB29	BDMB28	BDMB27	BDMB26	BDMB25	BDMB24
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
	BDMB23	BDMB22	BDMB21	BDMB20	BDMB19	BDMB18	BDMB17	BDMB16
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
	BDMB15	BDMB14	BDMB13	BDMB12	BDMB11	BDMB10	BDMB9	BDMB8
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
	BDMB7	BDMB6	BDMB5	BDMB4	BDMB3	BDMB2	BDMB1	BDMB0
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: *: Undefined

Break data mask register B (BDMRB) is a 32-bit readable/writable register that specifies which bits of the break data set in BDRB are to be masked. BDMRB is not initialized by a power-on reset or manual reset.

Bits 31 to 0—Break Data Mask B31 to B0 (BDMB31–BDMB0): These bits specify whether the corresponding bit of the channel B break data (BDB31–BDB0) set in BDRB is to be masked.

Bit 31–0: BDMBn	Description
0	Channel B break data bit BDBn is included in break conditions
1	Channel B break data bit BDBn is masked, and not included in break conditions

n = 31 to 0

Note: When the data bus value is included in the break conditions, the operand size should be specified. When byte size is specified, set the same data in bits 15–8 and 7–0 of BDRB and BDMRB.

20.2.11 Break Bus Cycle Register B (BBRB)

BBRB is the channel B bus break register. The bit configuration is the same as for BBRA.

20.2.12 Break Control Register (BRCR)

Bit:	15	14	13	12	11	10	9	8
	CMFA	CMFB	—	—	—	PCBA	—	—
Initial value:	0	0	0	0	0	*	0	0
R/W:	R/W	R/W	R	R	R	R/W	R	R

Bit:	7	6	5	4	3	2	1	0
	DBEB	PCBB	—	—	SEQ	—	—	UBDE
Initial value:	*	*	0	0	*	0	0	0
R/W:	R/W	R/W	R	R	R/W	R	R	R/W

Note: *: Undefined

The break control register (BRCR) is a 16-bit readable/writable register that specifies (1) whether channels A and B are to be used as two independent channels or in a sequential condition, (2) whether the break is to be effected before or after instruction execution, (3) whether the BDRB register is to be included in the channel B break conditions, and (4) whether the user break debug function is to be used. BRCR also contains condition match flags. The CMFA, CMFB, and UBDE bits in BRCR are initialized to 0 by a power-on reset, but retain their value in standby mode. The value of the PCBA, DBEB, PCBB, and SEQ bits is undefined after a power-on reset or manual reset, so these bits should be initialized by software as necessary.

Bit 15—Condition Match Flag A (CMFA): Set to 1 when a break condition set for channel A is satisfied. This flag is not cleared to 0 (to confirm that the flag is set again after once being set, it should be cleared with a write.)

Bit 15: CMFA	Description	
0	Channel A break condition is not matched	(Initial value)
1	Channel A break condition match has occurred	

Bit 14—Condition Match Flag B (CMFB): Set to 1 when a break condition set for channel B is satisfied. This flag is not cleared to 0 (to confirm that the flag is set again after once being set, it should be cleared with a write.)

Bit 14: CMFB	Description	
0	Channel B break condition is not matched	(Initial value)
1	Channel B break condition match has occurred	

Bits 13 to 11—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 10—Instruction Access Break Select A (PCBA): Specifies whether a channel A instruction access cycle break is to be effected before or after the instruction is executed. This bit is not initialized by a power-on reset or manual reset.

Bit 10: PCBA	**Description**
0	Channel A PC break is effected before instruction execution
1	Channel A PC break is effected after instruction execution

Bits 9 and 8—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 7—Data Break Enable B (DBEB): Specifies whether the data bus condition is to be included in the channel B break conditions. This bit is not initialized by a power-on reset or manual reset.

Bit 7: DBEB	Description
0	Data bus condition is not included in channel B conditions
1	Data bus condition is included in channel B conditions

Note: When the data bus is included in the break conditions, bits IDB1–0 in break bus cycle register B (BBRB) should be set to 10 or 11.

Bit 6—PC Break Select B (PCBB): Specifies whether a channel B instruction access cycle break is to be effected before or after the instruction is executed. This bit is not initialized by a power-on reset or manual reset.

Bit 6: PCBB	Description
0	Channel B PC break is effected before instruction execution
1	Channel B PC break is effected after instruction execution

Bits 5 and 4—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 3—Sequence Condition Select (SEQ): Specifies whether the conditions for channels A and B are to be independent or sequential. This bit is not initialized by a power-on reset or manual reset.

Bit 3: SEQ	Description
0	Channel A and B comparisons are performed as independent conditions
1	Channel A and B comparisons are performed as sequential conditions (channel A $\rightarrow$ channel B)

Bits 2 and 1—Reserved: These bits are always read as 0, and should only be written with 0.

Bit 0—User Break Debug Enable (UBDE): Specifies whether the user break debug function (see section 20.4, User Break Debug Support Function) is to be used.

Bit 0: UBDE	Description	
0	User break debug function is not used	(Initial value)
1	User break debug function is used	

20.3 Operation

20.3.1 Explanation of Terms Relating to Accesses

An instruction access is an access that obtains an instruction. An operand access is any memory access for the purpose of instruction execution. For example, the access to address PC+disp×2+4 in the instruction MOV.W @(disp,PC), Rn (an access very close to the program counter) is an operand access. The fetching of an instruction from the branch destination when a branch instruction is executed is also an instruction access. As the term “data” is used to distinguish data from an address, the term “operand access” is used in this section.

In the SH7750, all operand accesses are treated as either read accesses or write accesses. The following instructions require special attention:

- PREF, OCBP, and OCBWB instructions: Treated as read accesses.
- MOVCA and OCBI instructions: Treated as write accesses.
- TAS instruction: Treated as one read access and one write access.

The operand accesses for the PREF, OCBP, OCBWB, and OCBI instructions are accesses with no access data.

The SH7750 handles all operand accesses as having a data size. The data size can be byte, word, longword, or quadword. The operand data size for the PREF, OCBP, OCBWB, MOVCA, and OCBI instructions is treated as longword.

20.3.2 Explanation of Terms Relating to Instruction Intervals

In this section, "1 (2, 3, ...) instruction(s) after...", as a measure of the distance between two instructions, is defined as follows. A branch is counted as an interval of two instructions.

- Example of sequence of instructions with no branch:

```
100   Instruction A (0 instructions after instruction A)
102   Instruction B (1 instruction after instruction A)
104   Instruction C (2 instructions after instruction A)
106   Instruction D (3 instructions after instruction A)
```

- Example of sequence of instructions with a branch (however, the example of a sequence of instructions with no branch should be applied when the branch destination of a delayed branch instruction is the instruction itself + 4):

```
       100   Instruction A: BT/S L200 (0 instructions after instruction A)
       102   Instruction B (1 instruction after instruction A, 0 instructions after instruction
             B)
L200   200   Instruction C (3 instructions after instruction A, 2 instructions after instruction
             B)
       202   Instruction D (4 instructions after instruction A, 3 instructions after instruction
             B)
```

20.3.3 User Break Operation Sequence

The sequence of operations from setting of break conditions to user break exception handling is described below.

1. Specify pre- or post-execution breaking in the case of an instruction access, inclusion or exclusion of the data bus value in the break conditions in the case of an operand access, and use of independent or sequential channel A and B break conditions, in the break control register (BRCR). Set the break addresses in the break address registers for each channel (BARA, BARB), the ASIDs corresponding to the break space in the break ASID registers (BASRA, BASRB), and the address and ASID masking methods in the break address mask registers (BAMRA, BAMRB). If the data bus value is to be included in the break conditions, also set the break data in the break data register (BDRB) and the data mask in the break data mask register (BDMRB).

2. Set the break bus conditions in the break bus cycle registers (BBRA, BBRB). If even one of the BBRA/BBRB instruction access/operand access select (ID bit) and read/write select groups (RW bit) is set to 00, a user break interrupt will not be generated on the corresponding channel. Make the BBRA and BBRB settings after all other break-related register settings have been completed. If breaks are enabled with BBRA/BBRB while the

break address, data, or mask register, or the break control register is in the initial state after a reset, a break may be generated inadvertently.

3. The operation when a break condition is satisfied depends on the BL bit (in the CPU's SR register). When the BL bit is 0, exception handling is started and the condition match flag (CMFA/CMFB) for the respective channel is set for the matched condition. When the BL bit is 1, the condition match flag (CMFA/CMFB) for the respective channel is set for the matched condition but exception handling is not started.

   The condition match flags (CMFA, CMFB) are set by a branch condition match, but are not reset. Therefore, a memory store instruction should be used on the BRCR register to clear the flags to 0. See section 20.3.6, Condition Match Flag Setting, for the exact setting conditions for the condition match flags.

4. When sequential condition mode has been selected, and the channel B condition is matched after the channel A condition has been matched, a break is effected at the instruction at which the channel B condition was matched. See section 20.3.8, Contiguous A and B Settings for Sequential Conditions, for the operation when the channel A condition match and channel B condition match occur close together. With sequential conditions, only the channel B condition match flag is set. When sequential condition mode has been selected, if it is wished to clear the channel A match when the channel A condition has been matched but the channel B condition has not yet been matched, this can be done by writing 0 to the SEQ bit in the BRCR register.

20.3.4 Instruction Access Cycle Break

1. When an instruction access/read/word setting is made in the break bus cycle register (BBRA/BBRB), an instruction access cycle can be used as a break condition. In this case, breaking before or after execution of the relevant instruction can be selected with the PCBA/PCBB bit in the break control register (BRCR). When an instruction access cycle is used as a break condition, clear the LSB of the break address registers (BARA, BARB) to 0. A break will not be generated if this bit is set to 1.

2. When a pre-execution break is specified, the break is effected when it is confirmed that the instruction is to be fetched and executed. Therefore, an overrun-fetched instruction (an instruction that is fetched but not executed when a branch or exception occurs) cannot be used in a break. However, if a TLB miss or TLB protection violation exception occurs at the time of the fetch of an instruction subject to a break, the break exception handling is carried out first. The instruction TLB exception handling is performed when the instruction is re-executed (see section 5.4, Exception Types and Priorities). Also, since a delayed branch instruction and the delay slot instruction are executed as a single instruction, if a pre-execution break is specified for a delay slot instruction, the break will be effected before execution of the delayed branch instruction. However, a pre-execution break cannot be specified for the delay slot instruction for an RTE instruction.

3. With a pre-execution break, the instruction set as a break condition is executed, then a break interrupt is generated before the next instruction is executed. When a post-execution break is set for a delayed branch instruction, the delay slot is executed and the break is effected before execution of the instruction at the branch destination (when the branch is made) or the instruction two instructions ahead of the branch instruction (when the branch is not made).

4. When an instruction access cycle is set for channel B, break data register B (BDRB) is ignored in judging whether there is an instruction access match. Therefore, a break condition specified by the DBEB bit in BRCR is not executed.

20.3.5 Operand Access Cycle Break

1. In the case of an operand access cycle break, the bits included in address bus comparison vary as shown below according to the data size specification in the break bus cycle register (BBRA/BBRB).

Data Size	Address Bits Compared
Quadword (100)	Address bits A31–A3
Longword (011)	Address bits A31–A2
Word (010)	Address bits A31–A1
Byte (001)	Address bits A31–A0
Not included in condition (000)	In quadword access, address bits A31–A3
	In longword access, address bits A31–A2
	In word access, address bits A31–A1
	In byte access, address bits A31–A0

2. When data value is included in break conditions in channel B

   When a data value is included in the break conditions, set the DBEB bit in the break control register (BRCR) to 1. In this case, break data register B (BDRB) and break data mask register B (BDMRB) settings are necessary in addition to the address condition. A user break interrupt is generated when all three conditions—address, ASID, and data—are matched. When a quadword access occurs, the 64-bit access data is divided into an upper 32 bits and lower 32 bits, and interpreted as two 32-bit data units. A break is generated if either of the 32-bit data units satisfies the data match condition.

   Set the IDB1–0 bits in break bus cycle register B (BBRB) to 10 or 11. When byte data is specified, the same data should be set in the two bytes comprising bits 15–8 and bits 7–0 in 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.

3. When the DBEB bit in the break control register (BRCR) is set to 1, a break is not generated by an operand access with no access data (an operand access in a PREF, OCBP, OCBWB, or OCBI instruction).

20.3.6 Condition Match Flag Setting

1. Instruction access with post-execution condition, or operand access

   The flag is set when execution of the instruction that causes the break is completed. As an exception to this, however, in the case of an instruction with more than one operand access the flag may be set on detection of the match condition alone, without waiting for execution of the instruction to be completed.

   Example 1:

   100 BT L200 (branch performed)

   102 Instruction (operand access break on channel A) → flag not set

   Example 2:

   110 FADD (FPU exception)

   112 Instruction (operand access break on channel A) → flag not set

2. Instruction access with pre-execution condition

   The flag is set when the break match condition is detected.

   Example 1:

   110 Instruction (pre-execution break on channel A) → flag set

   112 Instruction (pre-execution break on channel B) → flag not set

   Example 2:

   110 Instruction (pre-execution break on channel B, instruction access TLB miss) → flag set

20.3.7 Program Counter (PC) Value Saved

1. When instruction access (pre-execution) is set as a break condition, the program counter (PC) value saved to SPC in user break interrupt handling is the address of the instruction at which the break condition match occurred. In this case, a user break interrupt is generated and the fetched instruction is not executed.
2. When instruction access (post-execution) is set as a break condition, the program counter (PC) value saved to SPC in user break interrupt handling is the address of the instruction to be executed after the instruction at which the break condition match occurred. In this case, the fetched instruction is executed, and a user break interrupt is generated before execution of the next instruction.
3. When an instruction access (post-execution) break condition is set for a delayed branch instruction, the delay slot instruction is executed and a user break is effected before execution of the instruction at the branch destination (when the branch is made) or the instruction two instructions ahead of the branch instruction (when the branch is not made). In this case, the PC value saved to SPC is the address of the branch destination (when the branch is made) or the instruction following the delay slot instruction (when the branch is not made).

4. When operand access (address only) is set as a break condition, the address of the instruction to be executed after the instruction at which the condition match occurred is saved to SPC.
5. When operand access (address + data) is set as a break condition, execution of the instruction at which the condition match occurred is completed. A user break interrupt is generated before execution of instructions from one instruction later to four instructions later. It is not possible to specify at which instruction, from one later to four later, the interrupt will be generated. The start address of the instruction after the instruction for which execution is completed at the point at which user break interrupt handling is started is saved to SPC. If an instruction between one instruction later and four instructions later causes another exception, control is performed as follows. Designating the exception caused by the break as exception 1, and the exception caused by an instruction between one instruction later and four instructions later as exception 2, the fact that memory updating and register updating that essentially cannot be performed by exception 2 cannot be performed is guaranteed irrespective of the existence of exception 1. The program counter value saved is the address of the first instruction for which execution is suppressed. Whether exception 1 or exception 2 is used for the exception jump destination and the value written to the exception register (EXPEVT/INTEVT) is not guaranteed. However, if exception 2 is from a source not synchronized with an instruction (external interrupt or peripheral module interrupt), exception 1 is used for the exception jump destination and the value written to the exception register (EXPEVT/INTEVT).

20.3.8 Contiguous A and B Settings for Sequential Conditions

When channel A match and channel B match timings are close together, a sequential break may not be guaranteed. Rules relating to the guaranteed range are given below.

1. Instruction access matches on both channel A and channel B

Instruction B is 0 instructions after instruction A	Equivalent to setting the same address. Do not use this setting.
Instruction B is 1 instruction after instruction A	Sequential operation is not guaranteed.
Instruction B is 2 or more instructions after instruction A	Sequential operation is guaranteed.

2. Instruction access match on channel A, operand access match on channel B

Instruction B is 0 or 1 instruction after instruction A	Sequential operation is not guaranteed.
Instruction B is 2 or more instructions after instruction A	Sequential operation is guaranteed.

3. Operand access match on channel A, instruction access match on channel B

Instruction B is 0 to 3 instructions after instruction A	Sequential operation is not guaranteed.
Instruction B is 4 or more instructions after instruction A	Sequential operation is guaranteed.

4. Operand access matches on both channel A and channel B

   Do not make a setting such that a single operand access will match the break conditions of both channel A and channel B. There are no other restrictions. For example, sequential operation is guaranteed even if two accesses within a single instruction match channel A and channel B conditions in turn.

20.3.9 Usage Notes

1. Do not execute a post-execution instruction access break for the SLEEP instruction.
2. Do not make an operand access break setting between 1 and 3 instructions before a SLEEP instruction.
3. The value of the BL bit referenced in a user break exception depends on the break setting, as follows.
   a. Pre-execution instruction access break: The BL bit value before the executed instruction is referenced.
   b. Post-execution instruction access break: The OR of the BL bit values before and after the executed instruction is referenced.
   c. Operand access break (address/data): The BL bit value after the executed instruction is referenced.
   d. In the case of an instruction that modifies the BL bit

SL.BL	Pre-Execution Instruction Access	Post-Execution Instruction Access	Pre-Execution Instruction Access	Post-Execution Instruction Access	Operand Access (Address/Data)
$0 \rightarrow 0$	A	A	A	A	A
$1 \rightarrow 0$	M	M	M	M	A
$0 \rightarrow 1$	A	M	A	M	M
$1 \rightarrow 1$	M	M	M	M	M

A: Accepted
M: Masked

e. In the case of an RTE delay slot

   The BL bit value before execution of a delay slot instruction is the same as the BL bit value before execution of an RTE instruction. The BL bit value after execution of a delay slot instruction is the same as the first BL bit value for the first instruction executed on returning by means of an RTE instruction (the same as the value of the BL bit in SSR before execution of the RTE instruction).

f. If an interrupt or exception is accepted with the BL bit cleared to 0, the value of the BL bit before execution of the first instruction of the exception handling routine is 1.

4. If channels A and B both match independently at virtually the same time, and, as a result, the SPC value is the same for both user break interrupts, only one user break interrupt is generated, but both the CMFA bit and the CMFB bit are set. For example:

   110 Instruction (post-execution instruction break on channel A) → SPC = 112, CMFA = 1

   112 Instruction (pre-execution instruction break on channel B) → SPC = 112, CMFB = 1

5. The PCBA or PCBB bit in BRCR is invalid for an instruction access break setting.

6. When the SEQ bit in BRCR is 1, the internal sequential break state is initialized by a channel B condition match. For example: A → A → B (user break generated) → B (no break generated)

7. In the event of contention between a re-execution type exception and a post-execution break in a multistep instruction, the re-execution type exception is generated. In this case, the CMF bit may or may not be set to 1 when the break condition occurs.

8. A post-execution break is classified as a completion type exception. Consequently, in the event of contention between a completion type exception and a post-execution break, the post-execution break is suppressed in accordance with the priorities of the two events. For example, in the case of contention between a TRAPA instruction and a post-execution break, the user break is suppressed. However, in this case, the CMF bit is set by the occurrence of the break condition.

20.4 User Break Debug Support Function

The user break debug support function enables the processing used in the event of a user break exception to be changed. When a user break exception occurs, if the UBDE bit is set to 1 in the BRCR register, the DBR register value will be used as the branch destination address instead of [VBR + offset]. The value of R15 is saved in the SGR register regardless of the value of the UBDE bit in the BRCR register or the kind of exception event. A flowchart of the user break debug support function is shown in figure 20.2.

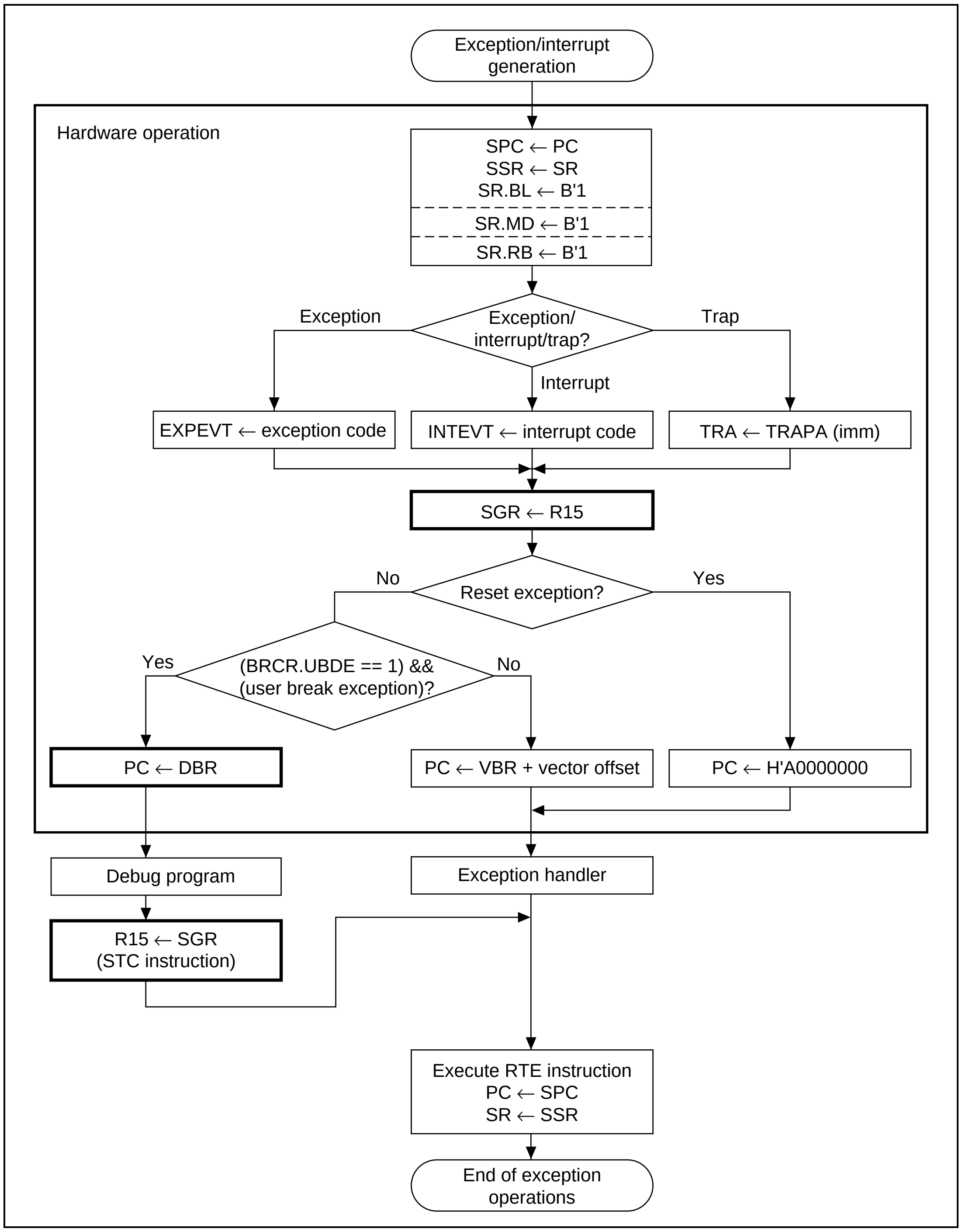


Figure 20.2 User Break Debug Support Function Flowchart

20.5 Examples of Use

Instruction Access Cycle Break Condition Settings

- Register settings: BASRA = H'80 / BARA = H'00000404 / BAMRA = H'00 / BBRA = H'0014 / BASRB = H'70 / BARB = H'00008010 / BAMRB = H'01 / BBRB = H'0014 / BDRB = H'00000000 / BDMRB = H'00000000 / BRCR = H'0400

  Conditions set: Independent channel A/channel B mode
  - Channel A: ASID: H'80 / address: H'00000404 / address mask: H'00
    Bus cycle: instruction access (post-instruction-execution), read (operand size not included in conditions)
  - Channel B: ASID: H'70 / address: H'00008010 / address mask: H'01
    Data: H'00000000 / data mask: H'00000000
    Bus cycle: instruction access (pre-instruction-execution), read (operand size not included in conditions)

  A user break is generated after execution of the instruction at address H'00000404 with ASID = H'80, or before execution of an instruction at addresses H'00008000–H'000083FE with ASID = H'70.

- Register settings: BASRA = H'80 / BARA = H'00037226 / BAMRA = H'00 / BBRA = H'0016 / BASRB = H'70 / BARB = H'0003722E / BAMRB = H'00 / BBRB = H'0016 / BDRB = H'00000000 / BDMRB = H'00000000 / BRCR = H'0008

  Conditions set: Channel A $\rightarrow$ channel B sequential mode
  - Channel A: ASID: H'80 / address: H'00037226 / address mask: H'00
    Bus cycle: instruction access (pre-instruction-execution), read, word
  - Channel B: ASID: H'70 / address: H'0003722E / address mask: H'00
    Data: H'00000000 / data mask: H'00000000
    Bus cycle: instruction access (pre-instruction-execution), read, word

  The instruction at address H'00037266 with ASID = H'80 is executed, then a user break is generated before execution of the instruction at address H'0003722E with ASID = H'70.

- Register settings: BASRA = H'80 / BARA = H'00027128 / BAMRA = H'00 / BBRA = H'001A / BASRB = H'70 / BARB = H'00031415 / BAMRB = H'00 / BBRB = H'0014 / BDRB = H'00000000 / BDMRB = H'00000000 / BRCR = H'0000

  Conditions set: Independent channel A/channel B mode

  — Channel A: ASID: H'80 / address: H'00027128 / address mask: H'00
    Bus cycle: CPU, instruction access (pre-instruction-execution), write, word

  — Channel B: ASID: H'70 / address: H'00031415 / address mask: H'00
    Data: H'00000000 / data mask: H'00000000
    Bus cycle: CPU, instruction access (pre-instruction-execution), read (operand size not included in conditions)

  A user break interrupt is not generated on channel A since the instruction access is not a write cycle.
  A user break interrupt is not generated on channel B since instruction access is performed on an even address.

Operand Access Cycle Break Condition Settings

- Register settings: BASRA = H'80 / BARA = H'00123456 / BAMRA = H'00 / BBRA = H'0024 / BASRB = H'70/ BARB = H'000ABCDE / BAMRB = H'02 / BBRB = H'002A / BDRB = H'0000A512 / BDMRB = H'00000000 / BRCR = H'0080

  Conditions set: Independent channel A/channel B mode

  — Channel A: ASID: H'80 / address: H'00123456 / address mask: H'00
    Bus cycle: operand access, read (operand size not included in conditions)

  — Channel B: ASID: H'70 / address: H'000ABCDE / address mask: H'02
    Data: H'0000A512 / data mask: H'00000000
    Bus cycle: operand access, write, word
    Data break enabled

  On channel A, a user break interrupt is generated in the event of a longword read at address H'00123454, a word read at address H'00123456, or a byte read at address H'00123456, with ASID = H'80.
  On channel B, a user break interrupt is generated when H'A512 is written by word access to any address from H'000AB000 to H'000ABFFE with ASID = H'70.

Section 21 Hitachi User Debug Interface (Hitachi-UDI)

21.1 Overview

21.1.1 Features

The Hitachi user debug interface (Hitachi-UDI) is a serial input/output interface conforming to JTAG, IEEE 1149.1, and IEEE Standard Test Access Port and Boundary-Scan Architecture. The SH7750's Hitachi-UDI does not support boundary-scan, but is used for emulator connection. The functions of this interface should not be used when using an emulator. Refer to the emulator manual for the method of connecting the emulator. The Hitachi-UDI uses six pins (TCK, TMS, TD, TDO, $\overline{\text{TRST}}$, and $\overline{\text{ASEBRK}}$/BRKACK). The pin functions and serial transfer protocol conform to the JTAG specifications.

21.1.2 Block Diagram

Figure 21.1 shows a block diagram of the Hitachi-UDI. The TAP (test access port) controller and control registers are reset independently of the chip reset pin by driving the $\overline{\text{TRST}}$ pin low or setting TMS to 1 and applying TCK for at least five clock cycles. The other circuits are reset and initialized in an ordinary reset. The Hitachi-UDI circuit has four internal registers: SDBPR, SDIR, SDDRH, and SDDRL (these last two together designated SDDR). The SDBPR register supports the JTAG bypass mode, SDIR is the command register, and SDDR is the data register. SDIR can be accessed directly from the TDI and TDO pins.

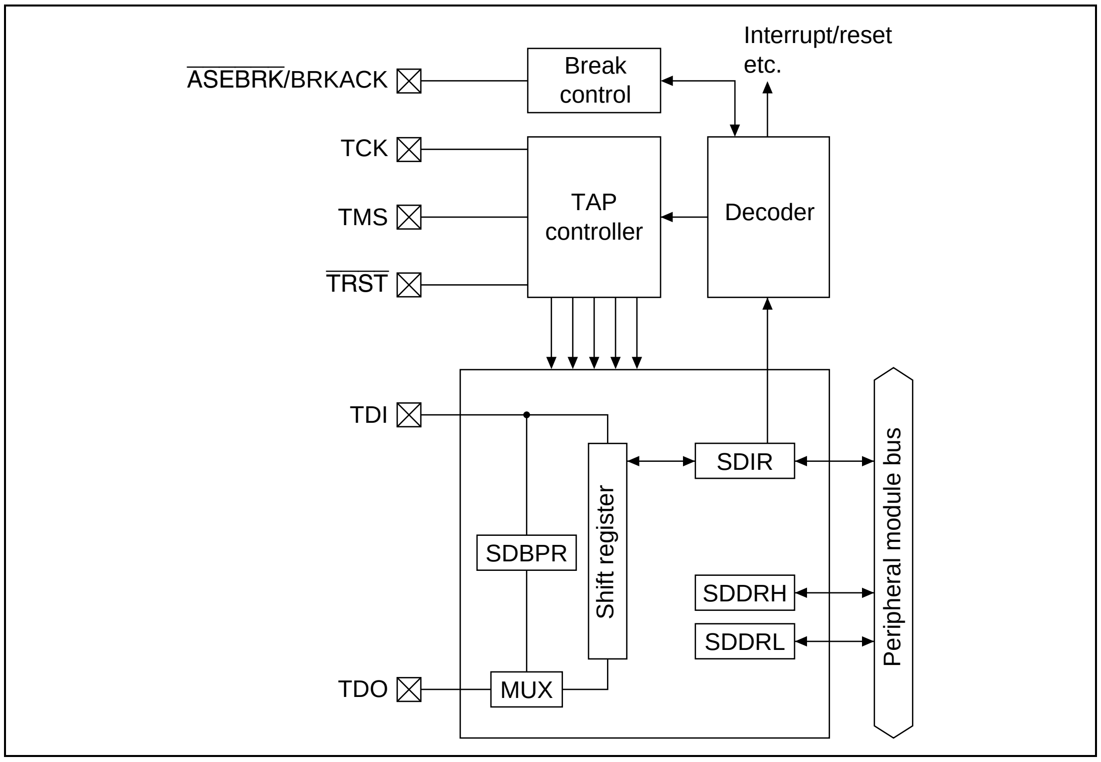


Figure 21.1 Block Diagram of Hitachi-UDI Circuit

21.1.3 Pin Configuration

Table 21.1 shows the Hitachi-UDI pin configuration.

Table 21.1 Hitachi-UDI Pins

Pin Name	Abbreviation	I/O	Function	When Not Used
Clock pin	TCK	Input	Same as the JTAG serial clock input pin. Data is transferred from data input pin TDI to the Hitachi-UDI circuit, and data is read from data output pin TDO, in synchronization with this signal.	Open*[1]
Mode pin	TMS	Input	The mode select input pin. Changing this signal in synchronization with TCK determines the meaning of the data input from TDI. The protocol conforms to the JTAG (IEEE Std 1149.1) specification.	Open*[1]
Reset pin	$\overline{\text{TRST}}$	Input	The input pin that resets the Hitachi-UDI. This signal is received asynchronously with respect to TCK, and effects a reset of the JTAG interface circuit when low. $\overline{\text{TRST}}$ must be driven low for a certain period when powering on, regardless of whether or not JTAG is used. This differs from the IEEE specification.	Fix at ground*[2]
Data input pin	TDI	Input	The data input pin. Data is sent to the Hitachi-UDI circuit by changing this signal in synchronization with TCK.	Open*[1]
Data output pin	TDO	Output	The data output pin. Data is sent to the Hitachi-UDI circuit by reading this signal in synchronization with TCK.	Open
Emulator pin	$\overline{\text{ASEBRK}}$/ BRKACK	Input/ output	Dedicated emulator pin	Open*[1]

Notes: 1. Pulled up inside the chip. When designing a board that allows use of an emulator, or when using interrupts and resets via the Hitachi-UDI, there is no problem in connecting a pullup resistance externally.

2. When designing a board that enables the use of an emulator, or when using interrupts and resets via the Hitachi-UDI, drive $\overline{\text{TRST}}$ low for a period overlapping $\overline{\text{RESET}}$ at power-on, and also provide for control by $\overline{\text{TRST}}$ alone.

The maximum frequency of TCK (TMS, TDI, TDO) is 20 MHz. Make the TCK or SH7750 CPG setting so that the TCK frequency is lower than that of the SH7750's on-chip peripheral module clock.

21.1.4 Register Configuration

Table 21.2 shows the Hitachi-UDI registers. Except for SDBPR, these registers are mapped in the control register space and can be referenced by the CPU.

Table 21.2 Hitachi-UDI Registers

		CPU Side				Hitachi-UDI Side		
Name	**Abbre-viation**	**R/W**	**P4 Address**	**Area 7 Address**	**Access Size**	**R/W**	**Access Size**	**Initial Value***
Instruction register	SDIR	R	H'FFF00000	H'1FF00000	16	R/W	16	H'FFFF
Data register H	SDDR/ SDDRH	R/W	H'FFF00008	H'1FF00008	32/16	—	32	Undefined
Data register L	SDDRL	R/W	H'FFF0000A	H'1FF0000A	16	—	—	Undefined
Bypass register	SDBPR	—	—	—	—	R/W	1	Undefined

Note: * Initialized when the $\overline{\text{TRST}}$ pin goes low or when the TAP is in the Test-Logic-Reset state.

21.2 Register Descriptions

21.2.1 Instruction Register (SDIR)

The instruction register (SDIR) is a 16-bit register that can only be read by the CPU. In the initial state, bypass mode is set. The value (command) is set from the serial input pin (TDI). SDIR is initialized by the $\overline{\text{TRST}}$ pin or in the TAP Test-Logic-Reset state. When this register is written to from the Hitachi-UDI, writing is possible regardless of the CPU mode. However, if a read is performed by the CPU while writing is in progress, it may not be possible to read the correct value. In this case, SDIR should be read twice, and then read again if the read values do not match. Operation is undefined if a reserved command is set in this register.

Bit:	15	14	13	12	11	10	9	8
	TI3	TI2	TI1	TI0	—	—	—	—
Initial value:	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R	R	R

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

Bits 15 to 12—Test Instruction Bits (TI3–TI0)

Bit 15: TI3	Bit 14: TI2	Bit 13: TI1	Bit 12: TI0	Description
0	0	—	—	Reserved
	1	0	—	Reserved
		1	0	Hitachi-UDI reset negate
			1	Hitachi-UDI reset assert
1	0	0	—	Reserved
		1	—	Hitachi-UDI interrupt
	1	0	—	Reserved
		1	0	Reserved
			1	Bypass mode (Initial value)

Bits 11 to 0—Reserved: These bits are always read as 1, and should only be written with 1.

21.2.2 Data Register (SDDR)

The data register (SDDR) is a 32-bit register, comprising the two 16-bit registers SDDRH and SDDRL, that can be read and written to by the CPU. The value in this register is not initialized by a $\overline{\text{TRST}}$ or CPU reset.

Bit:	31	30	29	28	27	26	25	24
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
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
Initial value:	*	*	*	*	*	*	*	*
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

Note: *: Undefined

Bits 31 to 0—DR Data: These bits store the SDDR value.

21.2.3 Bypass Register (SDBPR)

The bypass register (SDBPR) is a one-bit register that cannot be accessed by the CPU. When bypass mode is set in SDIR, SDBPR is connected between the TDI pin and TDO pin of the Hitachi-UDI.

21.3 Operation

21.3.1 TAP Control

Figure 21.2 shows the internal states of the TAP control circuit. These conform to the state transitions specified by JTAG.

- The transition condition is the TMS value at the rising edge of TCK.
- The TDI value is sampled at the rising edge of TCK, and shifted at the falling edge.
- The TDO value changes at the falling edge of TCK. When not in the Shift-DR or Shift-IR state, TDO is in the high-impedance state.
- In a transition to $\overline{\text{TRST}}$ = 0, a transition is made to the Test-Logic-Reset state asynchronously with respect to TCK.

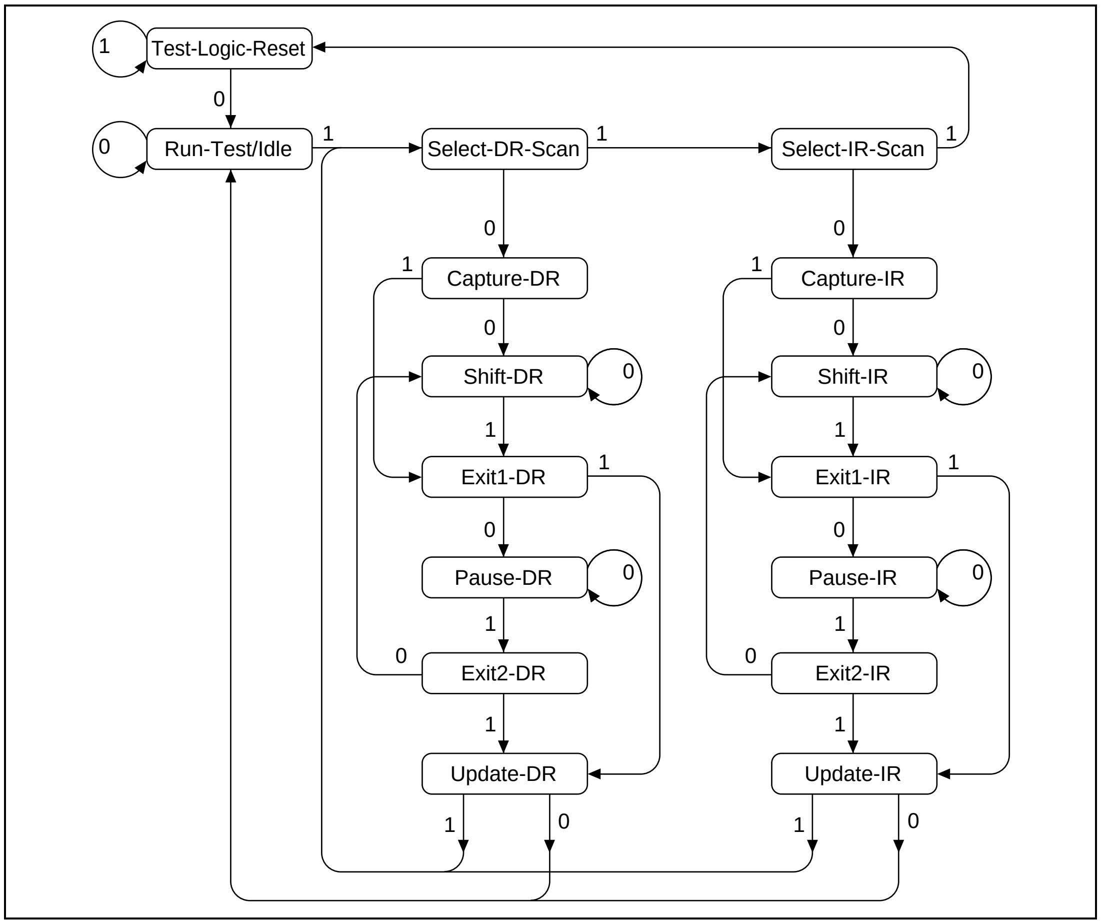


Figure 21.2 TAP Control State Transition Diagram

21.3.2 Hitachi-UDI Reset

A power-on reset is effected by an SDIR command. A reset is effected by sending a Hitachi-UDI reset assert command, and then sending a Hitachi-UDI reset negate command, from the Hitachi-UDI pin (see figure 21.3). The interval required between the Hitachi-UDI reset assert command and the Hitachi-UDI reset negate command is the same as the length of time the reset pin is held low in order to effect a power-on reset.

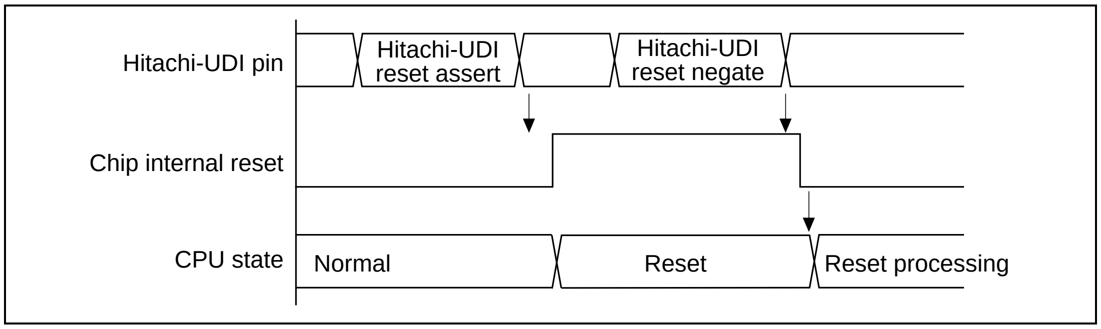


Figure 21.3 Hitachi-UDI Reset

21.3.3 Hitachi-UDI Interrupt

The Hitachi-UDI interrupt function generates an interrupt by setting a command value in SDIR from the Hitachi-UDI. The Hitachi-UDI interrupt is of general exception/interrupt operation type, with a branch to an address based on VBR and return effected by means of an RTE instruction. The exception code stored in control register INTEVT in this case is H'600. The priority of the Hitachi-UDI interrupt can be controlled with bits 3 to 0 of control register IPRC.

The Hitachi-UDI interrupt request signal is asserted for about eight SH7750 on-chip peripheral clock cycles after the command is set. The number of assertion cycles is determined by the ratio of TCK to the on-chip peripheral clock frequency. As the assertion period is limited, the CPU may sometimes miss a request. The Hitachi-UDI interrupt command automatically changes to the bypass command immediately after being set.

21.3.4 Bypass

The Hitachi-UDI pins can be set to the bypass mode specified by JTAG by setting a command in SDIR from the Hitachi-UDI.

21.4 Usage Notes

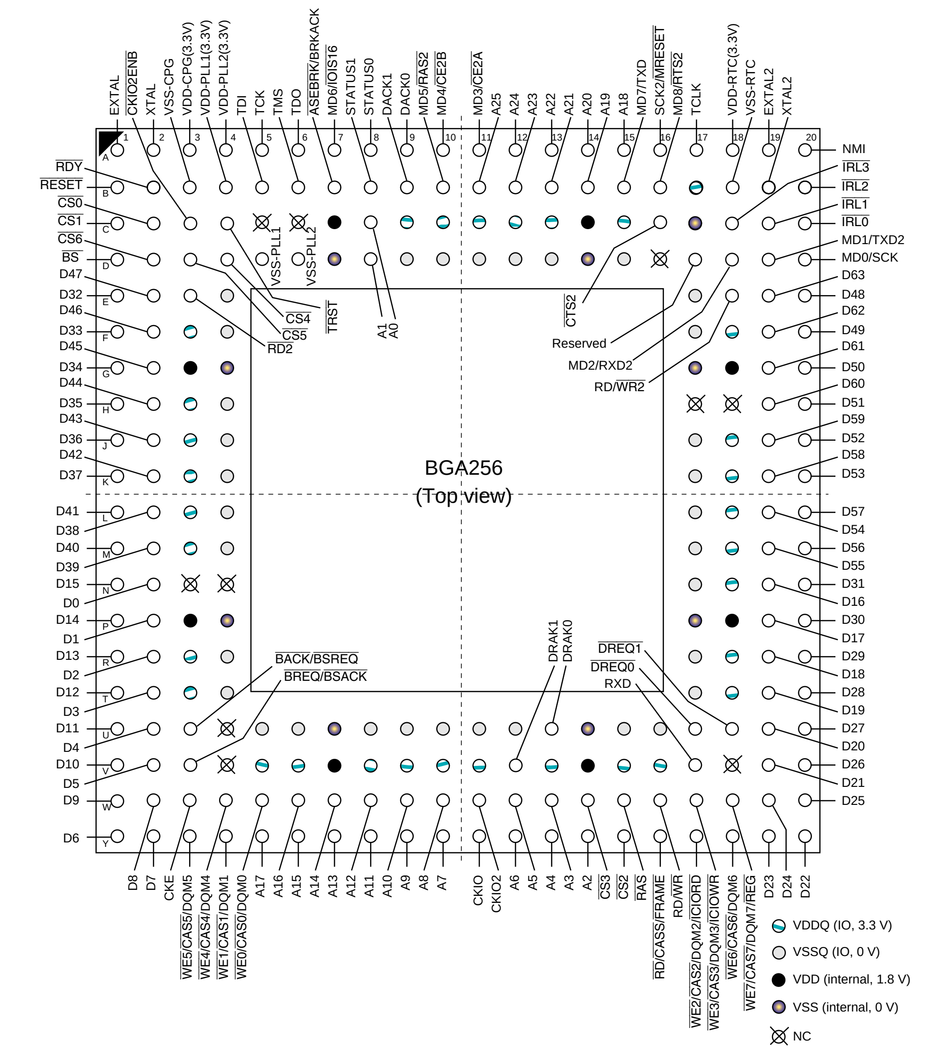
1. SDIR Command

   Once an SDIR command has been set, it remains unchanged until initialization by asserting $\overline{\text{TRST}}$ or placing the TAP in the Test-Logic-Reset state, or until another command (other than a Hitachi-UDI interrupt command) is written from the Hitachi-UDI.
2. SDIR Commands in Sleep Mode

   Sleep mode is cleared by a Hitachi-UDI interrupt or Hitachi-UDI reset, and these exception requests are accepted in this mode. In standby mode, neither a Hitachi-UDI interrupt nor a Hitachi-UDI reset is accepted..
3. The Hitachi-UDI is used for emulator connection. Therefore, Hitachi-UDI functions cannot be used when an emulator is used.
4. The SH7750’s Hitachi-UDI pins must not be connected to a boundary-scan signal loop on the board.

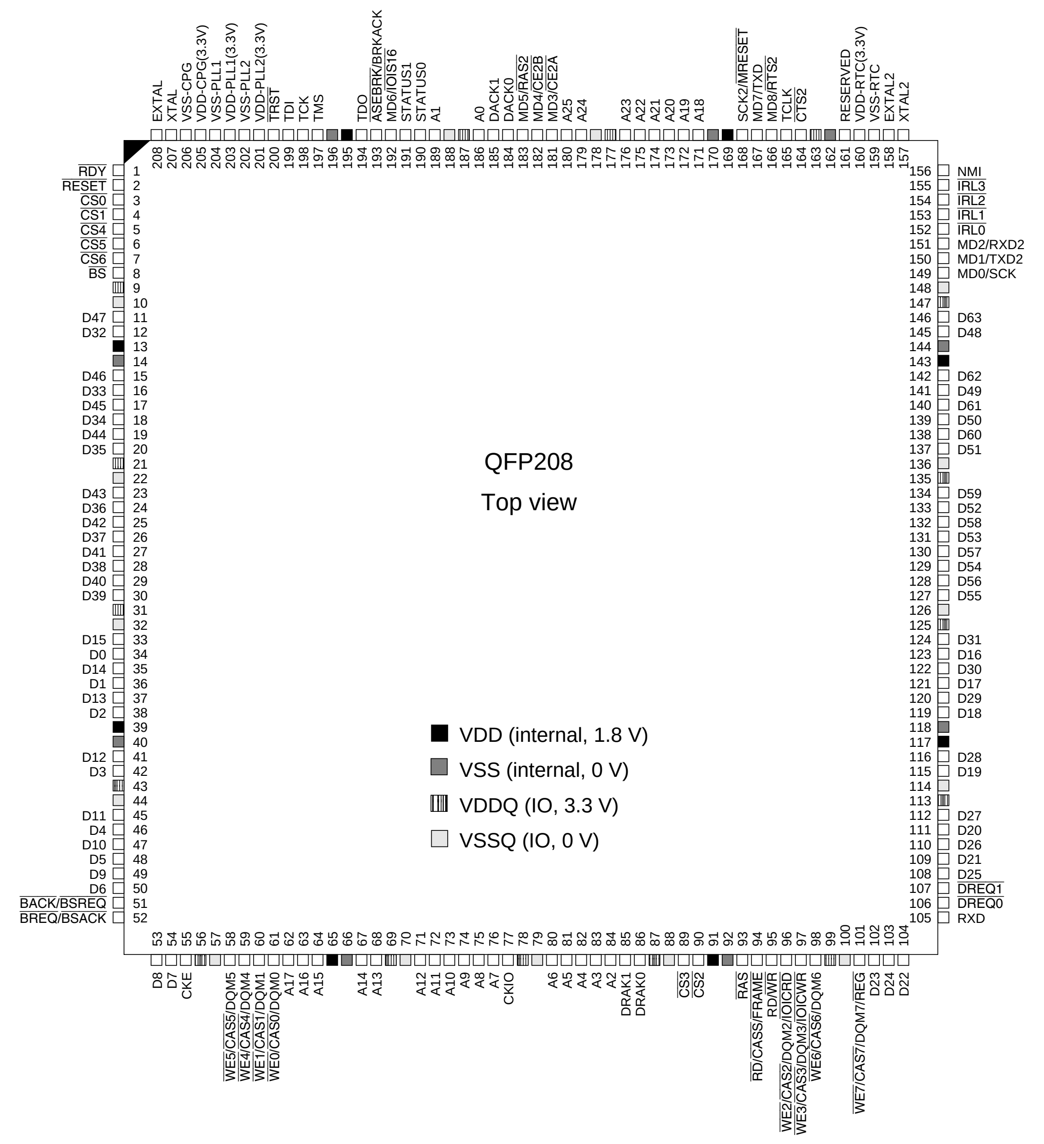
Section 22 Pin Description

22.1 Pin Arrangement



Note: Power must be supplied to the on-chip PLL power supply pins (VDD-PLL1, VDD-PLL2, VSS-PLL1, VSS-PLL2, VDD-CPG, VSS-CPG, VDD-RTC, and VSS-RTC) regardless of whether or not the PLL circuits, crystal resonator, and RTC are used.

Figure 22.1 Pin Arrangement (256-Pin BGA)



Note: Power must be supplied to the on-chip PLL power supply pins (VDD-PLL1, VDD-PLL2, VSS-PLL1, VSS-PLL2, VDD-CPG, VSS-CPG, VDD-RTC, and VSS-RTC) regardless of whether or not the PLL circuits, crystal resonator, and RTC are used.

Figure 22.2 Pin Arrangement (208-Pin QFP)

22.2 Pin Functions

22.2.1 Pin Functions (256-Pin BGA)

Table 22.1 Pin Functions

No.	Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
						SRAM	**DRAM**	**SDRAM**	**PCMCIA**	**MPX**
1	B2	$\overline{RDY}$	I	Bus ready		$\overline{RDY}$			$\overline{RDY}$	$\overline{RDY}$
2	B1	$\overline{RESET}$	I	Reset					$\overline{RESET}$	
3	C2	$\overline{CS0}$	O	Chip select 0		$\overline{CS0}$				$\overline{CS0}$
4	C1	$\overline{CS1}$	O	Chip select 1		$\overline{CS1}$				$\overline{CS1}$
5	D4	$\overline{CS4}$	O	Chip select 4		$\overline{CS4}$				$\overline{CS4}$
6	D3	$\overline{CS5}$	O	Chip select 5		$\overline{CS5}$			$\overline{CE1A}$	$\overline{CS5}$
7	D2	$\overline{CS6}$	O	Chip select 6		$\overline{CS6}$			$\overline{CE1B}$	$\overline{CS6}$
8	D1	$\overline{BS}$	O	Bust start		($\overline{BS}$)	($\overline{BS}$)	($\overline{BS}$)	($\overline{BS}$)	($\overline{BS}$)
9	E4	VSSQ	Power	IO GND (0 V)						
10	E3	$\overline{RD2}$	O	= $\overline{RD}$/$\overline{CASS}$/ $\overline{FRAME}$		$\overline{OE}$		$\overline{CAS}$	$\overline{OE}$	$\overline{FRAME}$
11	F3	VDDQ	Power	IO VDD (3.3 V)						
12	F4	VSSQ	Power	IO GND (0 V)						
13	E2	D47	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
14	E1	D32	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
15	G3	VDD	Power	Internal VDD (1.8 V)						
16	G4	VSS	Power	Internal GND (0 V)						
17	F2	D46	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
18	F1	D33	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
19	H3	VDDQ	Power	IO VDD (3.3 V)						
20	H4	VSSQ	Power	IO GND (0 V)						
21	G2	D45	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
22	G1	D34	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
23	H2	D44	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
24	H1	D35	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
25	J3	VDDQ	Power	IO VDD (3.3 V)						
26	J4	VSSQ	Power	IO GND (0 V)						
27	J2	D43	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
28	J1	D36	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)

Table 22.1 Pin Functions (cont)

No.	Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
						SRAM	DRAM	SDRAM	PCMCIA	MPX
29	K2	D42	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
30	K1	D37	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
31	K3	VDDQ	Power	IO VDD (3.3 V)						
32	K4	VSSQ	Power	IO GND (0 V)						
33	L1	D41	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
34	L2	D38	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
35	M1	D40	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
36	M2	D39	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
37	L3	VDDQ	Power	IO VDD (3.3 V)						
38	L4	VSSQ	Power	IO GND (0 V)						
39	N1	D15	I/O	Data						A15
40	N2	D0	I/O	Data						A0
41	P1	D14	I/O	Data						A14
42	P2	D1	I/O	Data						A1
43	M3	VDDQ	Power	IO VDD (3.3 V)						
44	M4	VSSQ	Power	IO GND (0 V)						
45	R1	D13	I/O	Data						A13
46	R2	D2	I/O	Data						A2
47	P3	VDD	Power	Internal VDD (1.8 V)						
48	P4	VSS	Power	Internal GND (0 V)						
49	T1	D12	I/O	Data						A12
50	T2	D3	I/O	Data						A3
51	R3	VDDQ	Power	IO VDD (3.3 V)						
52	R4	VSSQ	Power	IO GND (0 V)						
53	U1	D11	I/O	Data						A11
54	U2	D4	I/O	Data						A4
55	V1	D10	I/O	Data						A10
56	V2	D5	I/O	Data						A5
57	T3	VDDQ	Power	IO VDD (3.3 V)						
58	T4	VSSQ	Power	IO GND (0 V)						
59	W1	D9	I/O	Data						A9
60	Y1	D6	I/O	Data						A6

Table 22.1 Pin Functions (cont)

No.	Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
						SRAM	DRAM	SDRAM	PCMCIA	MPX
61	U3	$\overline{\text{BACK}}$/ $\overline{\text{BSREQ}}$	O	Bus acknowledge/ bus request						
62	V3	$\overline{\text{BREQ}}$/ $\overline{\text{BSACK}}$	I	Bus request/bus acknowledge						
63	W2	D8	I/O	Data						A8
64	Y2	D7	I/O	Data						A7
65	W3	CKE	O	Clock output enable				CKE		
66	V5	VDDQ	Power	IO VDD (3.3 V)						
67	U5	VSSQ	Power	IO GND (0 V)						
68	Y3	$\overline{\text{WE5}}$/$\overline{\text{CAS5}}$/ DQM5	O	D47–D40 select signal		$\overline{\text{WE5}}$	$\overline{\text{CAS5}}$	DQM5		
69	W4	$\overline{\text{WE4}}$/$\overline{\text{CAS4}}$/ DQM4	O	D39–D32 select signal		$\overline{\text{WE4}}$	$\overline{\text{CAS4}}$	DQM4		
70	Y4	$\overline{\text{WE1}}$/$\overline{\text{CAS1}}$/ DQM1	O	D15–D8 select signal		$\overline{\text{WE1}}$	$\overline{\text{CAS1}}$	DQM1	$\overline{\text{WE1}}$	
71	W5	$\overline{\text{WE0}}$/$\overline{\text{CAS0}}$/ DQM0	O	D7–D0 select signal		$\overline{\text{WE0}}$	$\overline{\text{CAS0}}$	DQM0		
72	Y5	A17	O	Address						
73	V6	VDDQ	Power	IO VDD (3.3 V)						
74	U6	VSSQ	Power	IO GND (0 V)						
75	W6	A16	O	Address						
76	Y6	A15	O	Address						
77	V7	VDD	Power	Internal VDD (1.8 V)						
78	U7	VSS	Power	Internal GND (0 V)						
79	W7	A14	O	Address						
80	Y7	A13	O	Address						
81	V8	VDDQ	Power	IO VDD (3.3 V)						
82	U8	VSSQ	Power	IO GND (0 V)						
83	V4	NC								
84	W8	A12	O	Address						
85	Y8	A11	O	Address						
86	W9	A10	O	Address						

Table 22.1 Pin Functions (cont)

No.	Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
						SRAM	DRAM	SDRAM	PCMCIA	MPX
87	V9	VDDQ	Power	IO VDD (3.3 V)						
88	U9	VSSQ	Power	IO GND (0 V)						
89	Y9	A9	O	Address						
90	W10	A8	O	Address						
91	Y10	A7	O	Address						
92	Y11	CKIO	O	Clock output				CKIO		
93	V10	VDDQ	Power	IO VDD (3.3 V)						
94	U10	VSSQ	Power	IO GND (0 V)						
95	W11	CKIO2	O	= CKIO*				CKIO		
96	Y12	A6	O	Address						
97	W12	A5	O	Address						
98	Y13	A4	O	Address						
99	V11	VDDQ	Power	IO VDD (3.3 V)						
100	U11	VSSQ	Power	IO GND (0 V)						
101	W13	A3	O	Address						
102	Y14	A2	O	Address						
103	V12	DRAK1	O	DMAC1 request acknowledge						
104	U13	DRAK0	O	DMAC0 request acknowledge						
105	V13	VDDQ	Power	IO VDD (3.3 V)						
106	U12	VSSQ	Power	IO GND (0 V)						
107	W14	$\overline{CS3}$	O	Chip select 3		$\overline{CS3}$	($\overline{CS3}$)	$\overline{CS3}$		$\overline{CS3}$
108	Y15	$\overline{CS2}$	O	Chip select 2		$\overline{CS2}$	($\overline{CS2}$)	$\overline{CS2}$		$\overline{CS2}$
109	V14	VDD	Power	Internal VDD (1.8 V)						
110	U14	VSS	Power	Internal GND (0 V)						
111	W15	$\overline{RAS}$	O	$\overline{RAS}$			$\overline{RAS}$	$\overline{RAS}$		
112	Y16	$\overline{RD}$/$\overline{CASS}$/ $\overline{FRAME}$	O	Read/$\overline{CAS}$/ $\overline{FRAME}$		$\overline{OE}$		$\overline{CAS}$	$\overline{OE}$	$\overline{FRAME}$
113	V15	VDDQ	Power	IO VDD (3.3 V)						
114	U15	VSSQ	Power	IO GND (0 V)						

Note: * CKIO2 is not connected to PLL2.

Table 22.1 Pin Functions (cont)

No.	Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
						SRAM	DRAM	SDRAM	PCMCIA	MPX
115	W16	RD/$\overline{\text{WR}}$	O	Read/write			RD/$\overline{\text{WR}}$	RD/$\overline{\text{WR}}$		RD/$\overline{\text{WR}}$
116	Y17	$\overline{\text{WE2}}$/$\overline{\text{CAS2}}$/ DQM2/ $\overline{\text{ICIORD}}$	O	D23–D16 select signal		$\overline{\text{WE2}}$	$\overline{\text{CAS2}}$	DQM2	$\overline{\text{ICIORD}}$	
117	W17	$\overline{\text{WE3}}$/$\overline{\text{CAS3}}$/ DQM3/ $\overline{\text{ICIOWR}}$	O	D31–D24 select signal		$\overline{\text{WE3}}$	$\overline{\text{CAS3}}$	DQM3	$\overline{\text{ICIOWR}}$	
118	Y18	$\overline{\text{WE6}}$/$\overline{\text{CAS6}}$/ DQM6	O	D55–D48 select signal		$\overline{\text{WE6}}$	$\overline{\text{CAS6}}$	DQM6		
119	V16	VDDQ	Power	IO VDD (3.3 V)						
120	U16	VSSQ	Power	IO GND (0 V)						
121	W18	$\overline{\text{WE7}}$/$\overline{\text{CAS7}}$/ DQM7/$\overline{\text{REG}}$	O	D63–D56 select signal		$\overline{\text{WE7}}$	$\overline{\text{CAS7}}$	DQM7	$\overline{\text{REG}}$	
122	Y19	D23	I/O	Data						A23
123	W19	D24	I/O	Data						A24
124	Y20	D22	I/O	Data						A22
125	V17	RXD	I	SCI data input						
126	U17	$\overline{\text{DREQ0}}$	I	Request from DMAC0						
127	U18	$\overline{\text{DREQ1}}$	I	Request from DMAC1						
128	W20	D25	I/O	Data						A25
129	T18	VDDQ	Power	IO VDD (3.3 V)						
130	T17	VSSQ	Power	IO GND (0 V)						
131	V19	D21	I/O	Data						A21
132	V20	D26	I/O	Data						
133	U19	D20	I/O	Data						A20
134	U20	D27	I/O	Data						
135	R18	VDDQ	Power	IO VDD (3.3 V)						
136	R17	VSSQ	Power	IO GND (0 V)						
137	T19	D19	I/O	Data						A19
138	T20	D28	I/O	Data						
139	P18	VDD	Power	Internal VDD (1.8 V)						
140	P17	VSS	Power	Internal GND (0 V)						
141	R19	D18	I/O	Data						A18

Table 22.1 Pin Functions (cont)

No.	Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
						SRAM	DRAM	SDRAM	PCMCIA	MPX
142	R20	D29	I/O	Data						
143	N18	VDDQ	Power	IO VDD (3.3 V)						
144	N17	VSSQ	Power	IO GND (0 V)						
145	P19	D17	I/O	Data						A17
146	P20	D30	I/O	Data						
147	N19	D16	I/O	Data						A16
148	N20	D31	I/O	Data						
149	M18	VDDQ	Power	IO VDD (3.3 V)						
150	M17	VSSQ	Power	IO GND (0 V)						
151	M19	D55	I/O	Data						
152	M20	D56	I/O	Data						
153	L19	D54	I/O	Data						
154	L20	D57	I/O	Data						
155	L18	VDDQ	Power	IO VDD (3.3 V)						
156	L17	VSSQ	Power	IO GND (0 V)						
157	K20	D53	I/O	Data						
158	K19	D58	I/O	Data						
159	J20	D52	I/O	Data						
160	J19	D59	I/O	Data						
161	K18	VDDQ	Power	IO VDD (3.3 V)						
162	K17	VSSQ	Power	IO GND (0 V)						
163	H20	D51	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
164	H19	D60	I/O	Data						
165	G20	D50	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
166	G19	D61	I/O	Data						ACCSIZE0
167	J18	VDDQ	Power	IO VDD (3.3 V)						
168	J17	VSSQ	Power	IO GND (0 V)						
169	F20	D49	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
170	F19	D62	I/O	Data						ACCSIZE1
171	G18	VDD	Power	Internal VDD (1.8 V)						
172	G17	VSS	Power	Internal GND (0 V)						
173	E20	D48	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)

Table 22.1 Pin Functions (cont)

No.	Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
						SRAM	**DRAM**	**SDRAM**	**PCMCIA**	**MPX**
174	E19	D63	I/O	Data						ACCSIZE2
175	F18	VDDQ	Power	IO VDD (3.3 V)						
176	F17	VSSQ	Power	IO GND (0 V)						
177	E17	VSSQ	Power	IO GND (0 V)						
178	E18	RD/$\overline{\text{WR2}}$	O	= RD/$\overline{\text{WR}}$			RD/$\overline{\text{WR}}$	RD/$\overline{\text{WR}}$		RD/$\overline{\text{WR}}$
179	D20	MD0/SCK	I/O	Mode/SCI clock	MD0	SCK	SCK	SCK	SCK	SCK
180	D19	MD1/TXD2	I/O	Mode SCIF data output	MD1	TXD2	TXD2	TXD2	TXD2	TXD2
181	D18	MD2/RXD2	I	Mode/SCIF data input	MD2	RXD2	RXD2	RXD2	RXD2	RXD2
182	C20	$\overline{\text{IRL0}}$	I	Interrupt 0						
183	C19	$\overline{\text{IRL1}}$	I	Interrupt 1						
184	B20	$\overline{\text{IRL2}}$	I	Interrupt 2						
185	C18	$\overline{\text{IRL3}}$	I	Interrupt 3						
186	A20	NMI	I	Nonmaskable interrupt						
187	B19	XTAL2	O	RTC crystal resonator pin						
188	A19	EXTAL2	I	RTC crystal resonator pin						
189	B18	VSS-RTC	Power	RTC GND (0 V)						
190	A18	VDD-RTC	Power	RTC VDD (3.3 V)						
191	D17	Reserved	I	Pull up to 3.3. V						
192	C17	VSS	Power	Internal GND (0 V)						
193	B17	VDDQ	Power	IO VDD (3.3 V)						
194	C16	$\overline{\text{CTS2}}$	I/O	SCIF data control (CTS)						
195	A17	TCLK	I/O	RTC/TMU clock						
196	B16	MD8/$\overline{\text{RTS2}}$	I/O	Mode/SCIF data control (RTS)	MD8	$\overline{\text{RTS2}}$	$\overline{\text{RTS2}}$	$\overline{\text{RTS2}}$	$\overline{\text{RTS2}}$	$\overline{\text{RTS2}}$
197	C15	VDDQ	Power	IO VDD (3.3 V)						

Table 22.1 Pin Functions (cont)

No.	Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
						SRAM	**DRAM**	**SDRAM**	**PCMCIA**	**MPX**
198	D15	VSSQ	Power	IO GND (0 V)						
199	B15	MD7/TXD	I/O	Mode/SCI data output	MD7	TXD	TXD	TXD	TXD	TXD
200	A16	SCK2/ $\overline{\text{MRESET}}$	I	SCIF clock/ manual reset	$\overline{\text{MRESET}}$	SCK2	SCK2	SCK2	SCK2	SCK2
201	C14	VDD	Power	Internal VDD (1.8 V)						
202	D14	VSS	Power	Internal GND (0 V)						
203	A15	A18	O	Address						
204	B14	A19	O	Address						
205	C13	VDDQ	Power	IO VDD (3.3 V)						
206	D13	VSSQ	Power	IO GND (0 V)						
207	A14	A20	O	Address						
208	B13	A21	O	Address						
209	A13	A22	O	Address						
210	B12	A23	O	Address						
211	C12	VDDQ	Power	IO VDD (3.3 V)						
212	D12	VSSQ	Power	IO GND (0 V)						
213	A12	A24	O	Address						
214	B11	A25	O	Address						
215	A11	MD3/$\overline{\text{CE2A}}$	I/O	Mode/ PCMCIA-CE	MD3				$\overline{\text{CE2A}}$	
216	A10	MD4/$\overline{\text{CE2B}}$	I/O	Mode/ PCMCIA-CE	MD4				$\overline{\text{CE2B}}$	
217	C11	VDDQ	Power	IO VDD (3.3 V)						
218	D11	VSSQ	Power	IO GND (0 V)						
219	B10	MD5/$\overline{\text{RAS2}}$	I/O	Mode/$\overline{\text{RAS}}$ (DRAM)	MD5		$\overline{\text{RAS2}}$			
220	A9	DACK0	O	DMAC0 bus acknowledge						
221	B9	DACK1	O	DMAC1 bus acknowledge						
222	C8	A0	O	Address						
223	C10	VDDQ	Power	IO VDD (3.3 V)						
224	D10	VSSQ	Power	IO GND (0 V)						

Table 22.1 Pin Functions (cont)

No.	Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
						SRAM	DRAM	SDRAM	PCMCIA	MPX
225	D8	A1	O	Address						
226	A8	STATUS0	O	Status						
227	B8	STATUS1	O	Status						
228	A7	MD6/ $\overline{\text{IOIS16}}$	I	Mode/$\overline{\text{IOIS16}}$ (PCMCIA)	MD6				$\overline{\text{IOIS16}}$	
229	C9	VDDQ	Power	IO VDD (3.3 V)						
230	D9	VSSQ	Power	IO GND (0 V)						
231	B7	$\overline{\text{ASEBRK}}$/ BRKACK	I/O	Pin break/ acknowledge (Hitachi-UDI)						
232	A6	TDO	O	Data out (Hitachi-UDI)						
233	C7	VDD	Power	Internal VDD (1.8 V)						
234	D7	VSS	Power	Internal GND (0 V)						
235	B6	TMS	I	Mode (Hitachi-UDI)						
236	A5	TCK	I	Clock (Hitachi-UDI)						
237	B5	TDI	I	Data in (Hitachi-UDI)						
238	C4	$\overline{\text{TRST}}$	I	Reset (Hitachi-UDI)						
239	C3	$\overline{\text{CKIO2ENB}}$	I	CKIO2, $\overline{\text{RD2}}$, RD/$\overline{\text{WR2}}$ enable						
240	C6	NC								
241	A4	VDD-PLL2	Power	PLL2 VDD (3.3V)						
242	D6	VSS-PLL2	Power	PLL2 GND (0V)						
243	B4	VDD-PLL1	Power	PLL1 VDD (3.3V)						
244	D5	VSS-PLL1	Power	PLL1 GND (0V)						
245	A3	VDD-CPG	Power	CPG VDD (3.3V)						
246	B3	VSS-CPG	Power	CPG GND (0V)						
247	A2	XTAL	O	Crystal resonator						

Table 22.1 Pin Functions (cont)

						Memory Interface				
No.	Pin No.	Pin Name	I/O	Function	Reset	SRAM	DRAM	SDRAM	PCMCIA	MPX
248	A1	EXTAL	I	External input clock/crystal resonator						
249	C5	NC								
250	D16	NC								
251	H17	NC								
252	H18	NC								
253	N3	NC								
254	N4	NC								
255	U4	NC								
256	V18	NC								

I:	Input
O:	Output
I/O:	Input/output
Power:	Power supply

Notes: 1. The VDDQ (3.3. V), VSSQ, VDD (1.8 V), and VSS pins must all be connected to the system power supply, and power must be supplied continuously. Even if only the RTC is operating (in standby mode), power must be supplied to all VDDQ, VSSQ, VDD, and VSS pins, in the same way as for VDD-RTC and VSS-RTC.

2. Power must be supplied to VDD-PLL1/2 and VSS-PLL1/2 regardless of whether or not the on-chip PLL circuits are used.
3. Power must be supplied to VDD-CPG and VSS-CPG regardless of whether or not the on-chip crystal resonator is used.
4. Power must be supplied to VDD-RTC and VSS-RTC regardless of whether or not the on-chip RTC is used.
5. VSSQ, VSS, VSS-RTC, VSS-PLL1/2, and VSS-CPG are connected inside the package.

22.2.2 Pin Functions (208-Pin QFP)

Table 22.2 Pin Functions

Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
					SRAM	**DRAM**	**SDRAM**	**PCMCIA**	**MPX**
1	$\overline{RDY}$	I	Bus ready		$\overline{RDY}$			$\overline{RDY}$	$\overline{RDY}$
2	$\overline{RESET}$	I	Reset					$\overline{RESET}$	
3	$\overline{CS0}$	O	Chip select 0		$\overline{CS0}$				$\overline{CS0}$
4	$\overline{CS1}$	O	Chip select 1		$\overline{CS1}$				$\overline{CS1}$
5	$\overline{CS4}$	O	Chip select 4		$\overline{CS4}$				$\overline{CS4}$
6	$\overline{CS5}$	O	Chip select 5		$\overline{CS5}$			$\overline{CE1A}$	$\overline{CS5}$
7	$\overline{CS6}$	O	Chip select 6		$\overline{CS6}$			$\overline{CE1B}$	$\overline{CS6}$
8	$\overline{BS}$	O	Bust start		($\overline{BS}$)	($\overline{BS}$)	($\overline{BS}$)	($\overline{BS}$)	($\overline{BS}$)
9	VDDQ	Power	IO VDD (3.3 V)						
10	VSSQ	Power	IO GND (0 V)						
11	D47	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
12	D32	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
13	VDD	Power	Internal VDD (1.8 V)						
14	VSS	Power	Internal GND (0 V)						
15	D46	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
16	D33	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
17	D45	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
18	D34	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
19	D44	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
20	D35	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
21	VDDQ	Power	IO VDD (3.3 V)						
22	VSSQ	Power	IO GND (0 V)						
23	D43	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
24	D36	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
25	D42	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
26	D37	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
27	D41	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
28	D38	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
29	D40	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)
30	D39	I/O	Data/port		(Port)	(Port)	(Port)	(Port)	(Port)

Table 22.2 Pin Functions (cont)

Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
					SRAM	DRAM	SDRAM	PCMCIA	MPX
31	VDDQ	Power	IO VDD (3.3 V)						
32	VSSQ	Power	IO GND (0 V)						
33	D15	I/O	Data						A15
34	D0	I/O	Data						A0
35	D14	I/O	Data						A14
36	D1	I/O	Data						A1
37	D13	I/O	Data						A13
38	D2	I/O	Data						A2
39	VDD	Power	Internal VDD (1.8 V)						
40	VSS	Power	Internal GND (0 V)						
41	D12	I/O	Data						A12
42	D3	I/O	Data						A3
43	VDDQ	Power	IO VDD (3.3 V)						
44	VSSQ	Power	IO GND (0 V)						
45	D11	I/O	Data						A11
46	D4	I/O	Data						A4
47	D10	I/O	Data						A10
48	D5	I/O	Data						A5
49	D9	I/O	Data						A9
50	D6	I/O	Data						A6
51	$\overline{\text{BACK}}$/ $\overline{\text{BSREQ}}$	O	Bus acknowledge/ bus request						
52	$\overline{\text{BREQ}}$/ $\overline{\text{BSACK}}$	I	Bus request/bus acknowledge						
53	D8	I/O	Data						A8
54	D7	I/O	Data						A7
55	CKE	O	Clock output enable				CKE		
56	VDDQ	Power	IO VDD (3.3 V)						
57	VSSQ	Power	IO GND (0 V)						
58	$\overline{\text{WE5}}$/$\overline{\text{CAS5}}$/ DQM5	O	D47–D40 select signal		$\overline{\text{WE5}}$	$\overline{\text{CAS5}}$	DQM5		

Table 22.2 Pin Functions (cont)

Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
					SRAM	**DRAM**	**SDRAM**	**PCMCIA**	**MPX**
59	$\overline{\text{WE4}}$/$\overline{\text{CAS4}}$/ DQM4	O	D39–D32 select signal		$\overline{\text{WE4}}$	$\overline{\text{CAS4}}$	DQM4		
60	$\overline{\text{WE1}}$/$\overline{\text{CAS1}}$/ DQM1	O	D15–D8 select signal		$\overline{\text{WE1}}$	$\overline{\text{CAS1}}$	DQM1	$\overline{\text{WE1}}$	
61	$\overline{\text{WE0}}$/$\overline{\text{CAS0}}$/ DQM0	O	D7–D0 select signal		$\overline{\text{WE0}}$	$\overline{\text{CAS0}}$	DQM0		
62	A17	O	Address						
63	A16	O	Address						
64	A15	O	Address						
65	VDD	Power	Internal VDD (1.8 V)						
66	VSS	Power	Internal GND (0 V)						
67	A14	O	Address						
68	A13	O	Address						
69	VDDQ	Power	IO VDD (3.3 V)						
70	VSSQ	Power	IO GND (0 V)						
71	A12	O	Address						
72	A11	O	Address						
73	A10	O	Address						
74	A9	O	Address						
75	A8	O	Address						
76	A7	O	Address						
77	CKIO	O	Clock output				CKIO		
78	VDDQ	Power	IO VDD (3.3 V)						
79	VSSQ	Power	IO GND (0 V)						
80	A6	O	Address						
81	A5	O	Address						
82	A4	O	Address						
83	A3	O	Address						
84	A2	O	Address						
85	DRAK1	O	DMAC1 request acknowledge						
86	DRAK0	O	DMAC0 request acknowledge						
87	VDDQ	Power	IO VDD (3.3 V)						

Table 22.2 Pin Functions (cont)

Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
					SRAM	DRAM	SDRAM	PCMCIA	MPX
88	VSSQ	Power	IO GND (0 V)						
89	$\overline{CS3}$	O	Chip select 3		$\overline{CS3}$	($\overline{CS3}$)	$\overline{CS3}$		$\overline{CS3}$
90	$\overline{CS2}$	O	Chip select 2		$\overline{CS2}$	($\overline{CS2}$)	$\overline{CS2}$		$\overline{CS2}$
91	VDD	Power	Internal VDD (1.8 V)						
92	VSS	Power	Internal GND (0 V)						
93	$\overline{RAS}$	O	$\overline{RAS}$			$\overline{RAS}$	$\overline{RAS}$		
94	$\overline{RD}$/$\overline{CASS}$/ $\overline{FRAME}$	O	Read/$\overline{CAS}$/ $\overline{FRAME}$		$\overline{OE}$		$\overline{CAS}$	$\overline{OE}$	$\overline{FRAME}$
95	RD/$\overline{WR}$	O	Read/write			RD/$\overline{WR}$	RD/$\overline{WR}$		RD/$\overline{WR}$
96	$\overline{WE2}$/$\overline{CAS2}$/ DQM2/ $\overline{ICIORD}$	O	D23–D16 select signal		$\overline{WE2}$	$\overline{CAS2}$	DQM2	$\overline{ICIORD}$	
97	$\overline{WE3}$/$\overline{CAS3}$/ DQM3/ $\overline{ICIOWR}$	O	D31–D24 select signal		$\overline{WE3}$	$\overline{CAS3}$	DQM3	$\overline{ICIOWR}$	
98	$\overline{WE6}$/$\overline{CAS6}$/ DQM6	O	D55–D48 select signal		$\overline{WE6}$	$\overline{CAS6}$	DQM6		
99	VDDQ	Power	IO VDD (3.3 V)						
100	VSSQ	Power	IO GND (0 V)						
101	$\overline{WE7}$/$\overline{CAS7}$/ DQM7/$\overline{REG}$	O	D63–D56 select signal		$\overline{WE7}$	$\overline{CAS7}$	DQM7	$\overline{REG}$	
102	D23	I/O	Data						A23
103	D24	I/O	Data						A24
104	D22	I/O	Data						A22
105	RXD	I	SCI data input						
106	$\overline{DREQ0}$	I	Request from DMAC0						
107	$\overline{DREQ1}$	I	Request from DMAC1						
108	D25	I/O	Data						A25
109	D21	I/O	Data						A21
110	D26	I/O	Data						
111	D20	I/O	Data						A20
112	D27	I/O	Data						
113	VDDQ	Power	IO VDD (3.3 V)						

Table 22.2 Pin Functions (cont)

Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
					SRAM	DRAM	SDRAM	PCMCIA	MPX
114	VSSQ	Power	IO GND (0 V)						
115	D19	I/O	Data						A19
116	D28	I/O	Data						
117	VDD	Power	Internal VDD (1.8 V)						
118	VSS	Power	Internal GND (0 V)						
119	D18	I/O	Data						A18
120	D29	I/O	Data						
121	D17	I/O	Data						A17
122	D30	I/O	Data						
123	D16	I/O	Data						A16
124	D31	I/O	Data						
125	VDDQ	Power	IO VDD (3.3 V)						
126	VSSQ	Power	IO GND (0 V)						
127	D55	I/O	Data						
128	D56	I/O	Data						
129	D54	I/O	Data						
130	D57	I/O	Data						
131	D53	I/O	Data						
132	D58	I/O	Data						
133	D52	I/O	Data						
134	D59	I/O	Data						
135	VDDQ	Power	IO VDD (3.3 V)						
136	VSSQ	Power	IO GND (0 V)						
137	D51	I/O	Data						
138	D60	I/O	Data						
139	D50	I/O	Data						
140	D61	I/O	Data						ACCSIZE0
141	D49	I/O	Data						
142	D62	I/O	Data						ACCSIZE1
143	VDD	Power	Internal VDD (1.8 V)						
144	VSS	Power	Internal GND (0 V)						

Table 22.2 Pin Functions (cont)

Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
					SRAM	**DRAM**	**SDRAM**	**PCMCIA**	**MPX**
145	D48	I/O	Data						
146	D63	I/O	Data						ACCSIZE2
147	VDDQ	Power	IO VDD (3.3 V)						
148	VSSQ	Power	IO GND (0 V)						
149	MD0/SCK	I/O	Mode/SCI clock	MD0	SCK	SCK	SCK	SCK	SCK
150	MD1/TXD2	I/O	Mode SCIF data output	MD1	TXD2	TXD2	TXD2	TXD2	TXD2
151	MD2/RXD2	I	Mode/SCIF data input	MD2	RXD2	RXD2	RXD2	RXD2	RXD2
152	$\overline{\text{IRL0}}$	I	Interrupt 0						
153	$\overline{\text{IRL1}}$	I	Interrupt 1						
154	$\overline{\text{IRL2}}$	I	Interrupt 2						
155	$\overline{\text{IRL3}}$	I	Interrupt 3						
156	NMI	I	Nonmaskable interrupt						
157	XTAL2	O	RTC crystal resonator pin						
158	EXTAL2	I	RTC crystal resonator pin						
159	VSS-RTC	Power	RTC GND (0 V)						
160	VDD-RTC	Power	RTC VDD (3.3 V)						
161	Reserved	I	Pull up to 3.3. V						
162	VSS	Power	Internal GND (0 V)						
163	VDDQ	Power	IO VDD (3.3 V)						
164	$\overline{\text{CTS2}}$	I/O	SCIF data control (CTS)						
165	TCLK	I/O	RTC/TMU clock						
166	MD8/$\overline{\text{RTS2}}$	I/O	Mode/SCIF data control (RTS)	MD8	$\overline{\text{RTS2}}$	$\overline{\text{RTS2}}$	$\overline{\text{RTS2}}$	$\overline{\text{RTS2}}$	$\overline{\text{RTS2}}$
167	MD7/TXD	I/O	Mode/SCI data output	MD7	TXD	TXD	TXD	TXD	TXD
168	SCK2/ $\overline{\text{MRESET}}$	I	SCIF clock/ manual reset	$\overline{\text{MRESET}}$	SCK2	SCK2	SCK2	SCK2	SCK2

Table 22.2 Pin Functions (cont)

Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
					SRAM	DRAM	SDRAM	PCMCIA	MPX
169	VDD	Power	Internal VDD (1.8 V)						
170	VSS	Power	Internal GND (0 V)						
171	A18	O	Address						
172	A19	O	Address						
173	A20	O	Address						
174	A21	O	Address						
175	A22	O	Address						
176	A23	O	Address						
177	VDDQ	Power	IO VDD (3.3 V)						
178	VSSQ	Power	IO GND (0 V)						
179	A24	O	Address						
180	A25	O	Address						
181	MD3/$\overline{\text{CE2A}}$	I/O	Mode/ PCMCIA-CE	MD3				$\overline{\text{CE2A}}$	
182	MD4/$\overline{\text{CE2B}}$	I/O	Mode/ PCMCIA-CE	MD4				$\overline{\text{CE2B}}$	
183	MD5/$\overline{\text{RAS2}}$	I/O	Mode/$\overline{\text{RAS}}$ (DRAM)	MD5		$\overline{\text{RAS2}}$			
184	DACK0	O	DMAC0 bus acknowledge						
185	DACK1	O	DMAC1 bus acknowledge						
186	A0	O	Address						
187	VDDQ	Power	IO VDD (3.3 V)						
188	VSSQ	Power	IO GND (0 V)						
189	A1	O	Address						
190	STATUS0	O	Status						
191	STATUS1	O	Status						
192	MD6/ $\overline{\text{IOIS16}}$	I	Mode/$\overline{\text{IOIS16}}$ (PCMCIA)	MD6				$\overline{\text{IOIS16}}$	
193	$\overline{\text{ASEBRK}}$/ BRKACK	I/O	Pin break/ acknowledge (Hitachi-UDI)						
194	TDO	O	Data out (Hitachi-UDI)						

Table 22.2 Pin Functions (cont)

Pin No.	Pin Name	I/O	Function	Reset	Memory Interface				
					SRAM	DRAM	SDRAM	PCMCIA	MPX
195	VDD	Power	Internal VDD (1.8 V)						
196	VSS	Power	Internal GND (0 V)						
197	TMS	I	Mode (Hitachi-UDI)						
198	TCK	I	Clock (Hitachi-UDI)						
199	TDI	I	Data in (Hitachi-UDI)						
200	$\overline{\text{TRST}}$	I	Reset (Hitachi-UDI)						
201	VDD-PLL2	Power	PLL2 VDD (3.3V)						
202	VSS-PLL2	Power	PLL2 GND (0V)						
203	VDD-PLL1	Power	PLL1 VDD (3.3V)						
204	VSS-PLL1	Power	PLL1 GND (0V)						
205	VDD-CPG	Power	CPG VDD (3.3V)						
206	VSS-CPG	Power	CPG GND (0V)						
207	XTAL	O	Crystal resonator						
208	EXTAL	I	External input clock/crystal resonator						

I:	Input
O:	Output
I/O:	Input/output
Power:	Power supply

Notes:
1. The VDDQ (3.3. V), VSSQ, VDD (1.8 V), and VSS pins must all be connected to the system power supply, and power must be supplied continuously. Even if only the RTC is operating (in standby mode), power must be supplied to all VDDQ, VSSQ, VDD, and VSS pins, in the same way as for VDD-RTC and VSS-RTC.
2. Power must be supplied to VDD-PLL1/2 and VSS-PLL1/2 regardless of whether or not the on-chip PLL circuits are used.
3. Power must be supplied to VDD-CPG and VSS-CPG regardless of whether or not the on-chip crystal resonator is used.
4. Power must be supplied to VDD-RTC and VSS-RTC regardless of whether or not the on-chip RTC is used.
5. With the QFP package, VSSQ, VSS, VSS-RTC, VSS-PLL1/2, and VSS-CPG are not connected inside the package.
6. The $\overline{\text{RD2}}$, RD/$\overline{\text{WR2}}$, CKIO2, and $\overline{\text{CKIO2ENB}}$ pins are not provided on the QFP package.
7. With the QFP package, the maximum external bus operating frequency is 83 MHz.

Section 23 Electrical Characteristics

23.1 Absolute Maximum Ratings

Table 23.1 Absolute Maximum Ratings

Item	Symbol	Value	Unit
I/O, PLL, RTC power supply voltage	V_{DDQ} $V_{DD-PLL1/2}$, V_{DD-RTC}, V_{DD-CPG}	–0.3 to 4.2	V
Internal power supply voltage	V_{DD}	–0.3 to 2.5	V
Input voltage	V_{in}	–0.3 to V_{DDQ} + 0.3	V
Operating temperature	T_{opr}	–20 to 75	°C
Storage temperature	T_{stg}	–55 to 125	°C

Note: Permanent damage to the chip may result if the maximum ratings are exceeded. V_{DD} (1.8 V) should be input after input of V_{DDQ}, $V_{DD-PLL1/2}$, V_{DD-RTC}, and V_{DD-CPG} (3.3 V).

23.2 DC Characteristics

Table 23.2 DC Characteristics

(Ta = –20 to +75°C)

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Power supply voltage		V_{DDQ} $V_{DD-PLL1/2}$ V_{DD-CPG} V_{DD-RTC}	3.0	3.3	3.6	V	Normal mode, sleep mode, standby mode
		V_{DD}	1.6	1.8	2.0		Normal mode, sleep mode, standby mode
Current dissipation	Normal operation	I_{DD}	—	840	—	mA	V_{DDQ}, $V_{DD-PLL1/2}$, V_{DD-RTC}, V_{DD-CPG} = 3.3 V V_{DD} = 1.8 V *1 f = 200 MHz *2 f = 100 MHz *3 f = 50 MHz
			—	420	—		
			—	210	—		
	Sleep mode		—	150*1	—		
			—	80*2	—		
			—	40*3	—		
	Standby mode		—	TBD	—	μA	Ta = 25°C (RTC on)
			—	TBD	—		Ta > 50°C (RTC on)
			—	TBD	—		Ta = 25°C (RTC off)
			—	TBD	—		Ta > 50°C (RTC off)
Current dissipation	Normal operation	I_{DDQ}	—	160*1	—	mA	V_{DDQ}, $V_{DD-PLL1/2}$, V_{DD-RTC}, V_{DD-CPG} = 3.3 V V_{DD} = 1.8 V *1 f = 200 MHz, t_{cyc} = 100 MHz *2 f = 100 MHz, t_{cyc} = 50 MHz *3 f = 50 MHz, t_{cyc} = 25 MHz
			—	80*2	—		
			—	40*3	—		
	Sleep mode		—	40*1	—		
			—	20*2	—		
			—	10*3	—		
	Standby mode		—	TBD	—	μA	Ta = 25°C (RTC on)
			—	TBD	—		Ta > 50°C (RTC on)
			—	TBD	—		Ta = 25°C (RTC off)
			—	TBD	—		Ta > 50°C (RTC off)

Table 23.2 DC Characteristics (cont)

(Ta = –20 to +75°C)

Item		Symbol	Min	Typ	Max	Unit	Test Conditions
Input voltage	$\overline{\text{RESET}}$, NMI, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$/BRKACK	V_{IH}	$V_{DDQ} \times 0.9$	—	V_{DDQ} + 0.3	V	
	Other input pins		2.0	—	V_{DDQ} + 0.3		
	$\overline{\text{RESET}}$, NMI, $\overline{\text{TRST}}$, $\overline{\text{ASEBRK}}$/BRKACK	V_{IL}	–0.3	—	$V_{DDQ} \times 0.1$		
	Other input pins		–0.3	—	$V_{DDQ} \times 0.2$		
Output voltage	All output pins	V_{OH}	2.4	—	—	V	
		V_{OL}	—	—	0.55		
Pull-up resistance	Port pins	R_{pull}	20	60	180	kΩ	
Pin capacitance	All pins	C_L	—	—	10	pF	

Notes: 1. Connect $V_{DD-PLL1/2}$, V_{DD-RTC}, and V_{DD-CPG} to V_{DDQ}, and V_{SS-CPG}, $V_{SS-PLL1/2}$, and $V_{SSQ-RTC}$ to GND, regardless of whether or not the PLL circuits and RTC are used.

2. The current dissipation values are for V_{IH} min = V_{DDQ} – 0.5 V and V_{IL} max = 0.5 V with all output pins unloaded.

3. To reduce the leakage current in standby mode, the RTC must be turned on.

4. I_{DDQ} is the sum of the V_{DDQ}, $V_{DD-PLL1/2}$, V_{DD-RTC}, and V_{DD-CPG} 3.3 V system currents.

Table 23.3 Permissible Output Currents

(Ta = –20 to +75°C)

Item	Symbol	Min	Typ	Max	Unit
Permissible output low current (per pin)	I_{OL}	—	—	2	mA
Permissible output low current (total)	ΣI_{OL}	—	—	120	
Permissible output high current (per pin)	$-I_{OH}$	—	—	2	
Permissible output high current (total)	$\Sigma(-I_{OH})$	—	—	40	

Note: To protect chip reliability, do not exceed the output current values in table 23.3.

23.3 AC Characteristics

In principle, SH7750 input should be synchronous. Unless specified otherwise, ensure that the setup time and hold times for each input signal are observed.

Table 23.4 Clock Timing

Item		Symbol	Min	Typ	Max	Unit	Notes
Operating frequency	CPU, FPU, cache, TLB	f	1	—	200	MHz	
	External bus		1	—	100		
	Peripheral modules		1	—	50		

23.3.1 Clock and Control Signal Timing

Table 23.5 Clock and Control Signal Timing

(V_{DDQ} = 3.0 to 3.6 V, V_{DD} = typ. 1.8 V, T_a = –20 to +75°C, C_L = 30 pF)

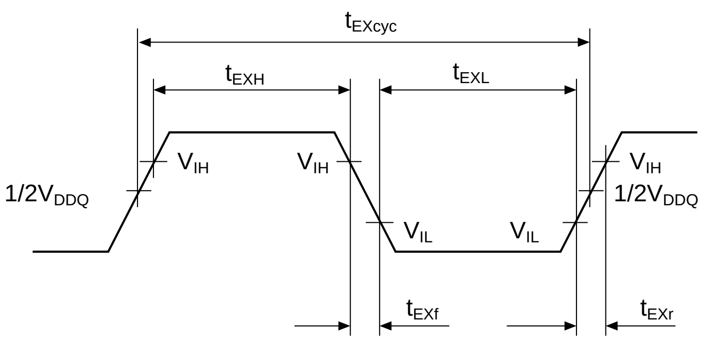
Item			Symbol	Min	Max	Unit	Figure
EXTAL clock input frequency	PLL1, 2 operating	1/2 divider operating	f_{EX}	16	66.7	MHz	
		1/2 divider not operating	f_{EX}	8	33.3		
	PLL1, 2 not operating	1/2 divider operating	f_{EX}	2	66.7		
		1/2 divider not operating	f_{EX}	1	33.3		
EXTAL clock input cycle time			t_{EXcyc}	15	1000	ns	23.1
EXTAL clock input low-level pulse width			t_{EXL}	3.5		ns	23.1
EXTAL clock input high-level pulse width			t_{EXH}	3.5		ns	23.1
EXTAL clock output rise time			t_{EXr}		4	ns	23.1
EXTAL clock input fall time			t_{EXf}		4	ns	23.1
CKIO clock output	PLL2 operating		f_{OP}	25	100	MHz	
	PLL2 not operating		f_{OP}	1	100	MHz	
CKIO clock output cycle time			t_{cyc}	10	1000	ns	23.2
CKIO clock output low-level pulse width			t_{CKOL}	1	—	ns	23.2
CKIO clock output high-level pulse width			t_{CKOH}	1	—	ns	23.2
CKIO clock output rise time			t_{CKOr}	—	4	ns	23.2
CKIO clock output fall time			t_{CKOf}	—	4	ns	23.2

Table 23.5 Clock and Control Signal Timing (cont)

(V_{DDQ} = 3.0 to 3.6 V, V_{DD} = typ. 1.8 V, T_a = –20 to +75°C, C_L = 30 pF)

Item	Symbol	Min	Max	Unit	Figure
Power-on oscillation settling time	t_{OSC1}	10	—	ms	23.3, 23.5
Power-on oscillation settling time/mode settling	t_{OSCMD}	10	—	ms	23.3, 23.5
SCK2 reset setup time	t_{SCK2RS}	20	—	ns	23.11
SCK2 reset hold time	t_{SCK2RH}	20	—	ns	23.3, 23.5, 23.11
MD reset setup time	t_{MDRS}	3	—	t_{cyc}	23.12
MD reset hold time	t_{MDRH}	20	—	ns	23.3, 23.5, 23.12
$\overline{\text{RESET}}$ assert time	t_{RESW}	20	—	t_{cyc}	23.3, 23.4, 23.5, 23.6, 23.11
PLL synchronization settling time	t_{PLL}	200	—	µs	23.9, 23.10
Standby return oscillation settling time 1	t_{OSC2}	10	—	ms	23.4, 23.6
Standby return oscillation settling time 2	t_{OSC3}	5	—	ms	23.7
Standby return oscillation settling time 3	t_{OSC4}	5	—	ms	23.8
IRL interrupt determination time (RTC used, standby mode)	t_{IRLSTB}	—	200	µs	23.10
$\overline{\text{TRST}}$ reset hold time	t_{TRSTRH}	0		ns	23.3, 23.5

Note: When a crystal resonator is connected to EXTAL and XTAL, the maximum frequency is 33.3 MHz. When a 3rd overtone crystal resonator is used, an external tank circuit is necessary.



Note: When the clock is input from the EXTAL pin

Figure 23.1 EXTAL Clock Input Timing

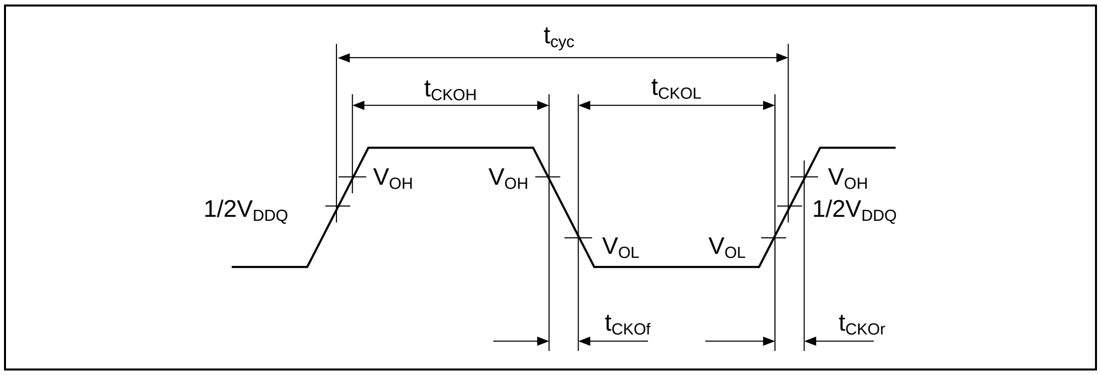
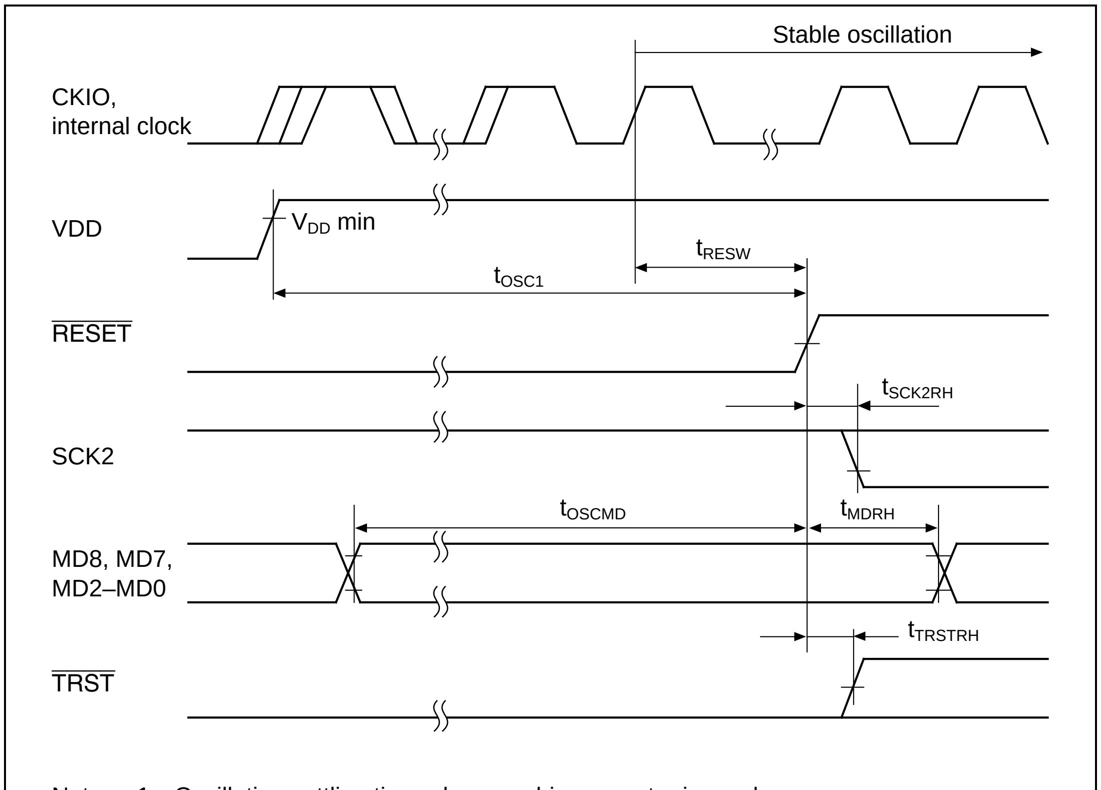


Figure 23.2 CKIO Clock Output Timing



Notes: 1. Oscillation settling time when on-chip resonator is used
2. PLL2 not operating

Figure 23.3 Power-On Oscillation Settling Time

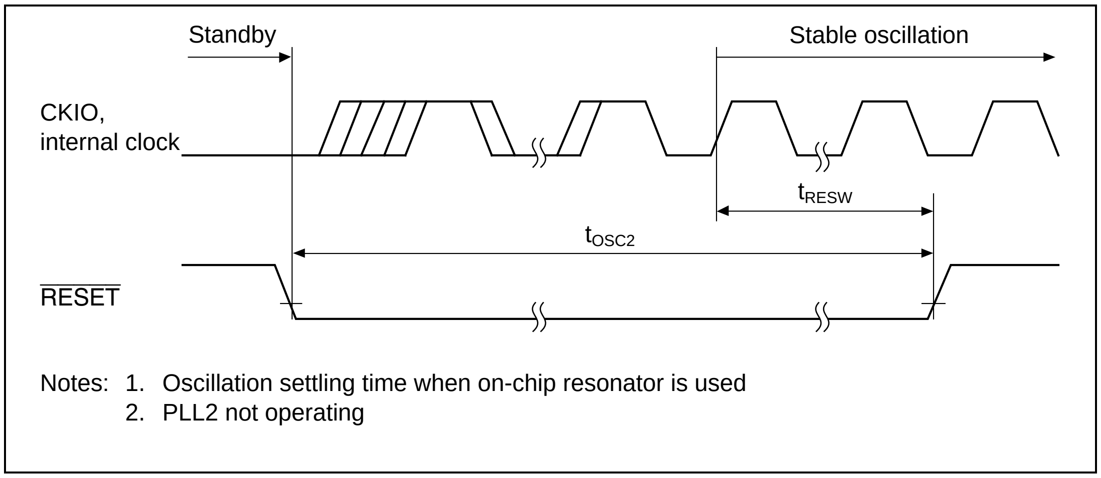

Standby
Stable oscillation
CKIO,
internal clock
t_{RESW}
t_{OSC2}
$\overline{\text{RESET}}$
Notes: 1. Oscillation settling time when on-chip resonator is used
2. PLL2 not operating

Figure 23.4 Standby Return Oscillation Settling Time (Return by $\overline{\text{RESET}}$)

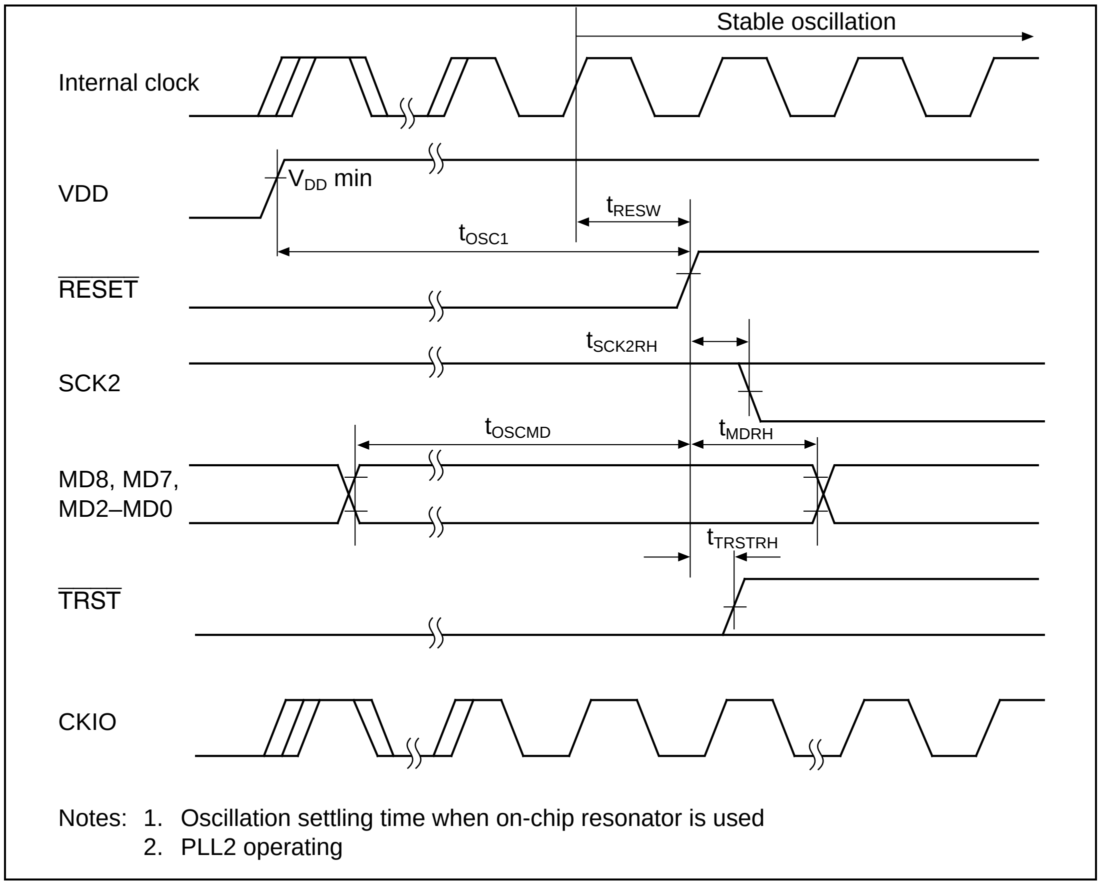
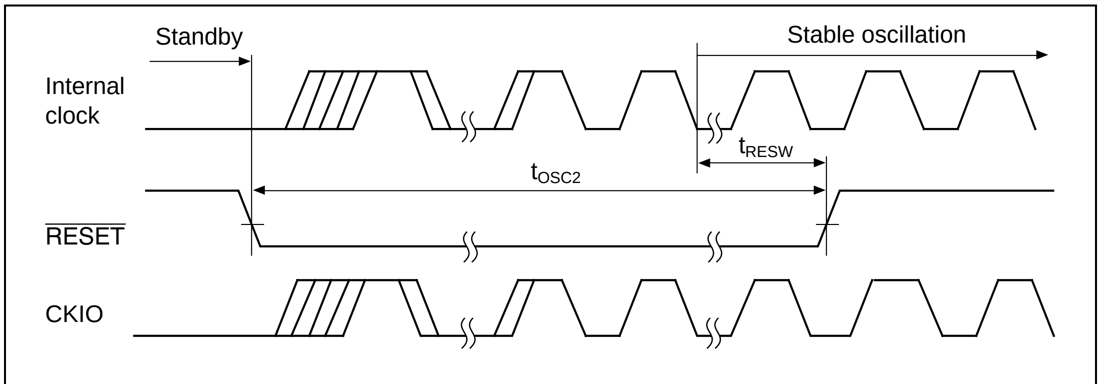
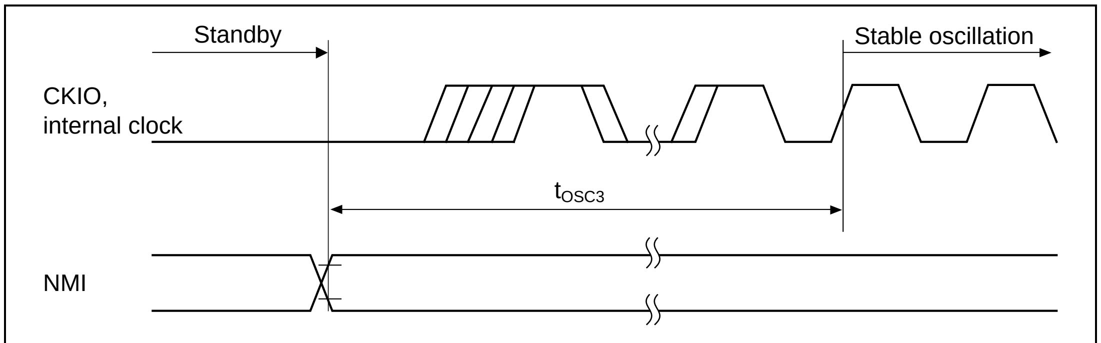


Figure 23.5 Power-On Oscillation Settling Time



Notes: 1. Oscillation settling time when on-chip resonator is used
2. PLL2 operating

Figure 23.6 Standby Return Oscillation Settling Time (Return by $\overline{\text{RESET}}$)



Note: Oscillation settling time when on-chip resonator is used

Figure 23.7 Standby Return Oscillation Settling Time (Return by NMI)

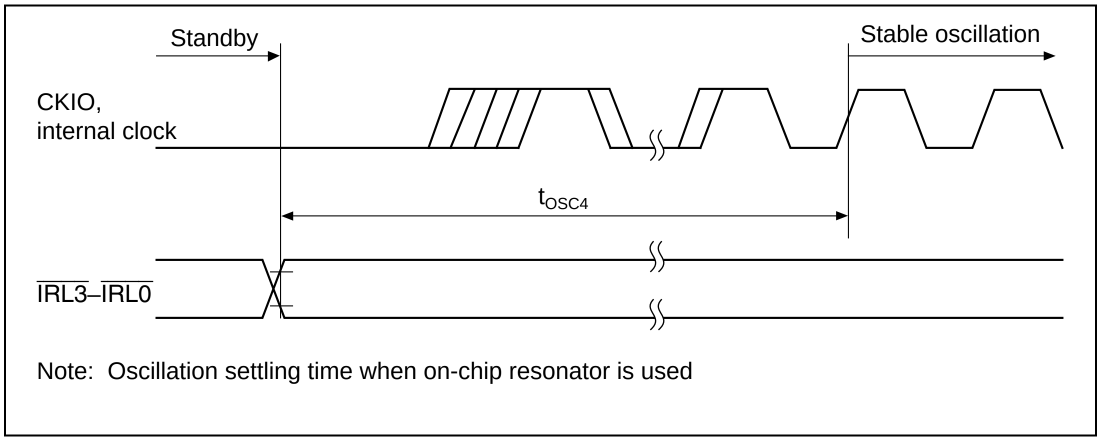

Standby
Stable oscillation
CKIO,
internal clock
t_{OSC4}
$\overline{IRL3}$–$\overline{IRL0}$
Note: Oscillation settling time when on-chip resonator is used

Figure 23.8 Standby Return Oscillation Settling Time (Return by $\overline{IRL3}$–$\overline{IRL0}$)

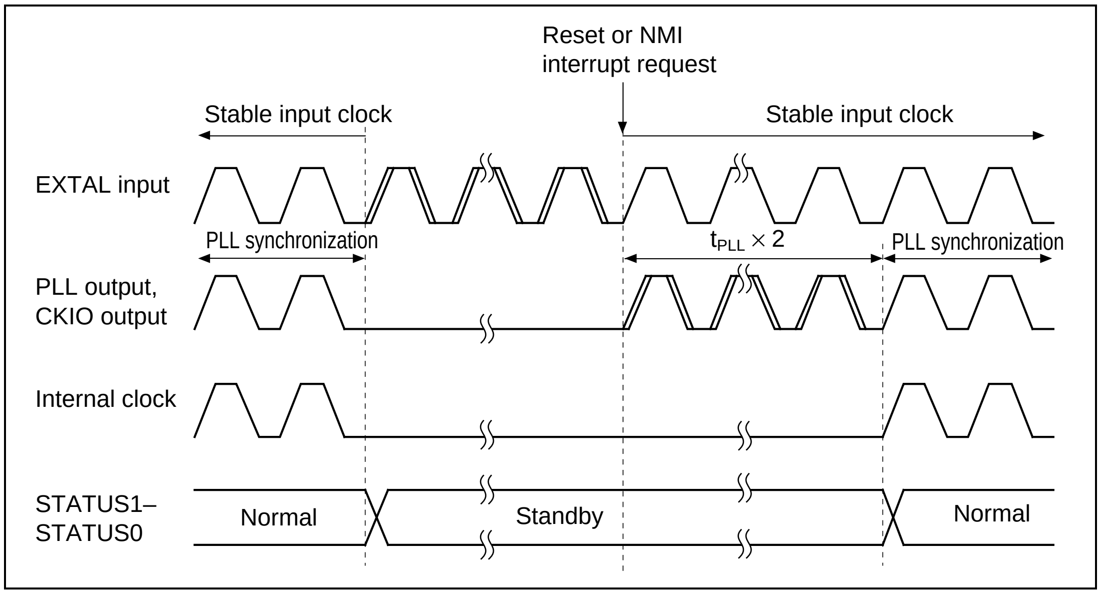


Figure 23.9 PLL Synchronization Settling Time in Case of $\overline{RESET}$ or NMI Interrupt

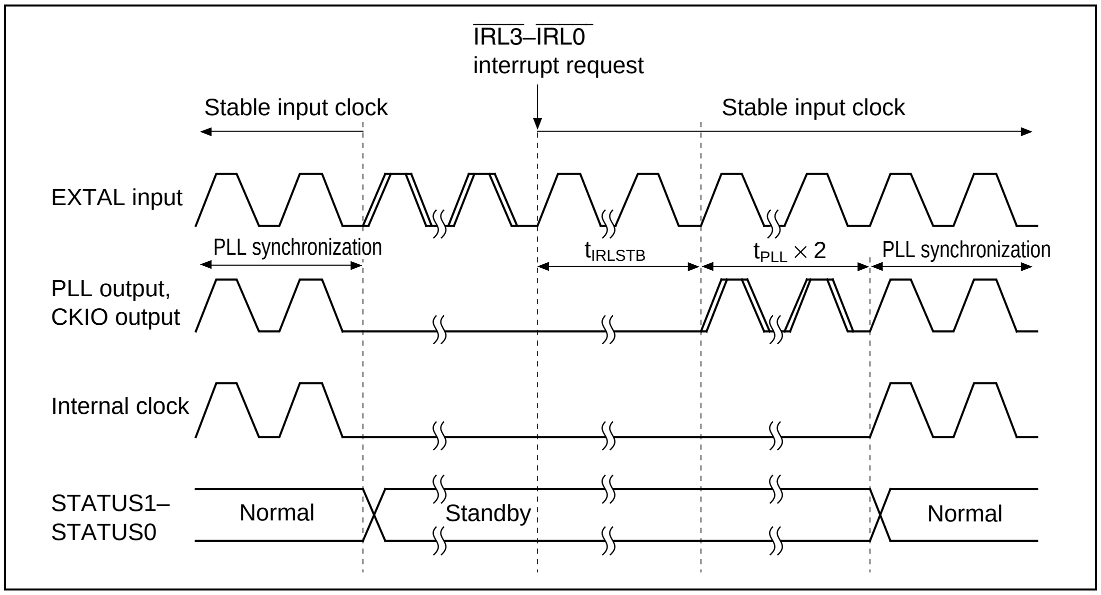


Figure 23.10 PLL Synchronization Settling Time in Case of IRL Interrupt

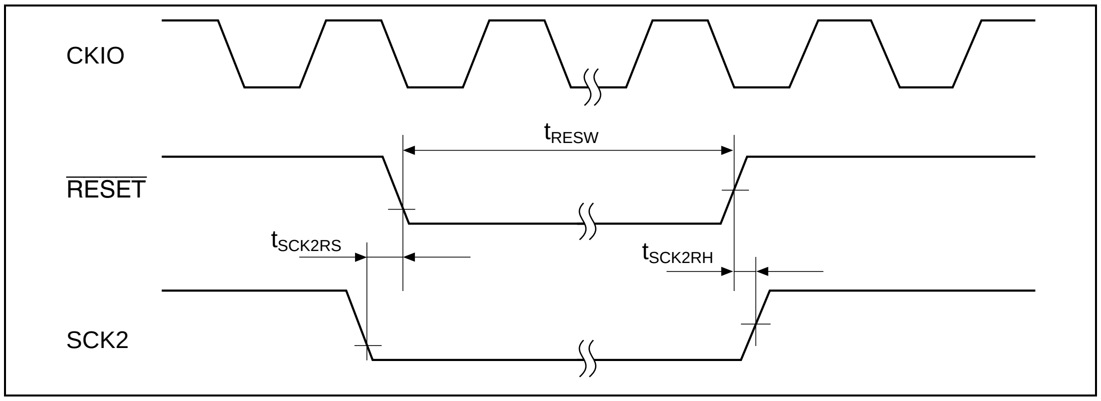


Figure 23.11 Manual Reset Input Timing

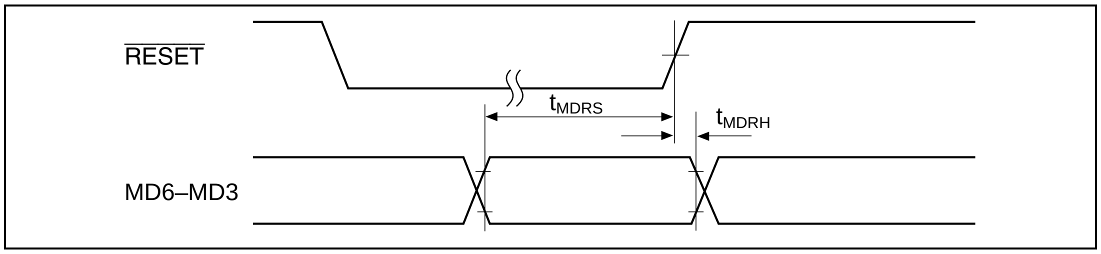


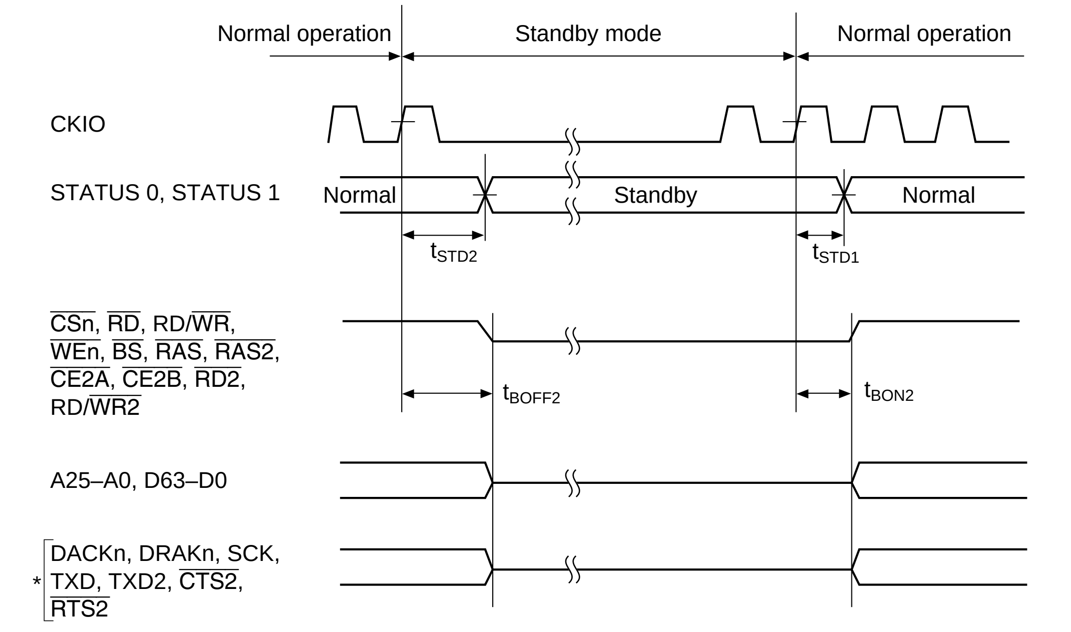
Figure 23.12 Mode Input Timing

23.3.2 Control Signal Timing

Table 23.6 Control Signal Timing

(V_{DDQ} = 3.0 to 3.6 V, V_{DD} = typ. 1.8 V, T_a = –20 to +75°C, C_L = 30 pF, PLL2 on)

		66 MHz		83 MHz		100 MHz				
Item	**Symbol**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Unit**	**Figure**	**Note**
$\overline{\text{BREQ}}$ setup time	t_{BREQS}	2	—	2	—	2	—	ns		BGA
		3.5	—	1.5	—	—	—	ns		QFP
$\overline{\text{BREQ}}$ hold time	t_{BREQH}	1.5	—	1.5	—	1.5	—	ns		
$\overline{\text{BACK}}$ delay time	t_{BACKD}	—	10	—	8	—	6	ns		
Bus tri-state delay time	t_{BOFF1}	—	15	—	12	—	10	ns		
Bus tri-state delay time to standby mode	t_{BOFF2}	—	2	—	2	—	2	t_{cyc}	23.13	
Bus buffer on time	t_{BON1}	—	15	—	12	—	10	ns		
Bus buffer on time from standby	t_{BON2}	—	1	—	1	—	1	t_{cyc}	23.13	
STATUS0/1 delay time	t_{STD1}	—	11	—	9	—	7	ns	23.13	
STATUS0/1 delay time to standby	t_{STD2}	—	2	—	2	—	2	t_{cyc}	23.13	


Normal operation
Standby mode
Normal operation
CKIO
STATUS 0, STATUS 1
Normal
Standby
Normal
t_{STD2}
t_{STD1}
$\overline{CSn}$, $\overline{RD}$, RD/$\overline{WR}$, $\overline{WEn}$, $\overline{BS}$, $\overline{RAS}$, $\overline{RAS2}$, $\overline{CE2A}$, $\overline{CE2B}$, $\overline{RD2}$, RD/$\overline{WR2}$
t_{BOFF2}
t_{BON2}
A25–A0, D63–D0
* DACKn, DRAKn, SCK, TXD, TXD2, $\overline{CTS2}$, $\overline{RTS2}$

Note: * When the PHZ bit in STBCR is set to 1, these pins go to the high-impedance state (except for pins being used as port pins, which retain their port state).

Figure 23.13 Pin Drive Timing for Standby Mode

23.3.3. Bus Timing

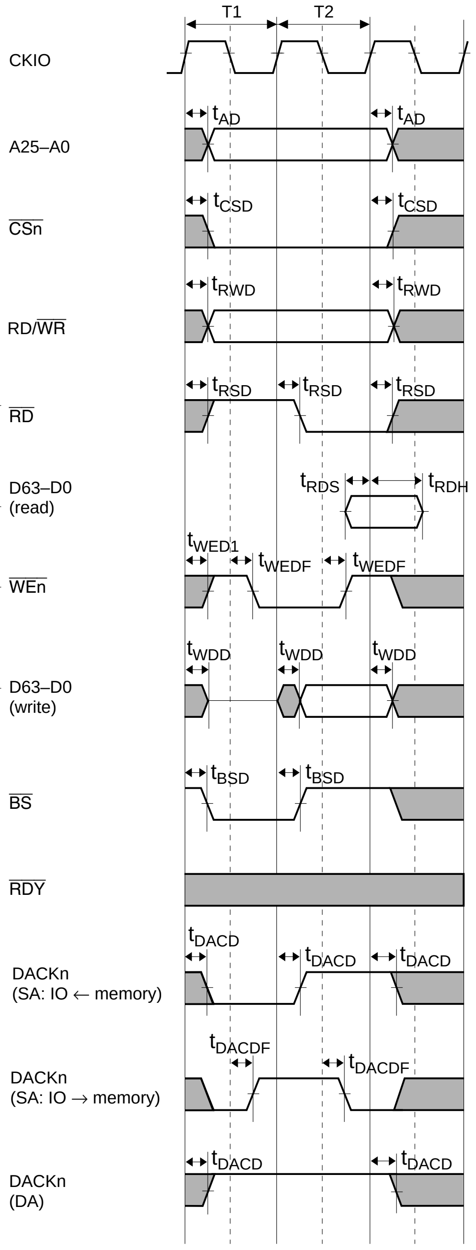
Table 23.7 Bus Timing

(V_{DDQ} = 3.0 to 3.6 V, V_{DD} = typ. 1.8 V, T_a = –20 to +75°C, C_L = 30 pF, PLL2 on)

		66 MHz		83 MHz		100 MHz			
Item	**Symbol**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Unit**	**Notes**
Address delay time	t_{AD}	—	10	—	8	—	6	ns	
$\overline{BS}$ delay time	t_{BSD}	—	10	—	8	—	6	ns	
$\overline{CS}$ delay time	t_{CSD}	—	10	—	8	—	6	ns	
$\overline{RW}$ delay time	t_{RWD}	—	10	—	8	—	6	ns	
$\overline{RD}$ delay time	t_{RSD}	—	10	—	8	—	6	ns	
Read data setup time	t_{RDS}	2	—	2	—	2	—	ns	BGA
		3.5	—	3.5	—	—	—	ns	QFP
Read data hold time	t_{RDH}	1.5	—	1.5	—	1.5	—	ns	
$\overline{WE}$ delay time (falling edge)	t_{WEDF}	—	10	—	8	—	6	ns	Relative to CKIO falling edge
$\overline{WE}$ delay time	t_{WED1}	—	10	—	8	—	6	ns	
Write data delay time	t_{WDD}	—	10	—	8	—	6	ns	
$\overline{RDY}$ setup time	t_{RDYS}	2	—	2	—	2	—	ns	BGA
		3.5	—	3.5	—	—	—	ns	QFP
$\overline{RDY}$ hold time	t_{RDYH}	1.5	—	1.5	—	1.5		ns	
$\overline{RAS}$ delay time	t_{RASD}	—	10	—	8	—	6	ns	
$\overline{CAS}$ delay time 1	t_{CASD1}	—	10	—	8	—	6	ns	DRAM
$\overline{CAS}$ delay time 2	t_{CASD2}	—	10	—	8	—	6	ns	SDRAM
CKE delay time	t_{CKED}	—	10	—	8	—	6	ns	SDRAM
DQM delay time	t_{DQMD}	—	10	—	8	—	6	ns	SDRAM
$\overline{FRAME}$ delay time	t_{FMD}	—	10	—	8	—	6	ns	MPX
$\overline{IOIS16}$ setup time	t_{IO16S}	2	—	2	—	2	—	ns	BGA
		3.5	—	3.5	—	—	—	ns	QFP
$\overline{IOIS16}$ hold time	t_{IO16H}	1.5	—	1.5	—	1.5	—	ns	PCMCIA
$\overline{ICIOWR}$ delay time (falling edge)	t_{ICWSDF}	—	10	—	8	—	6	ns	PCMCIA
$\overline{ICIORD}$ delay time	t_{ICRSD}	—	10	—	8	—	6	ns	PCMCIA
DACK delay time	t_{DACD}	—	10	—	8	—	6	ns	

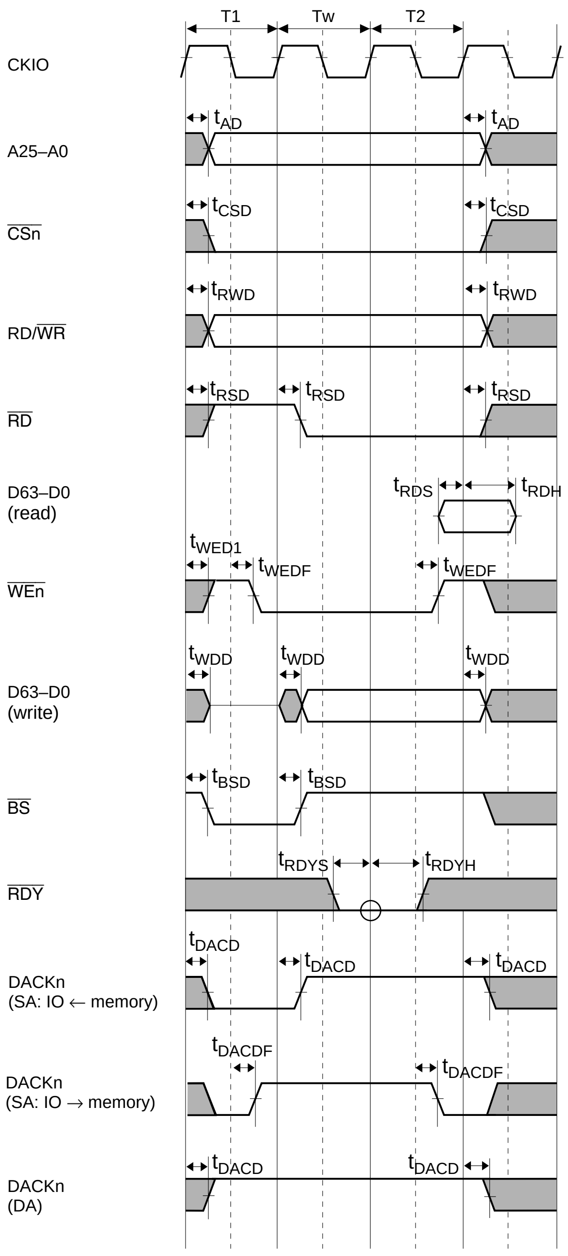
Table 23.7 Bus Timing (cont)

		66 MHz		83 MHz		100 MHz			
Item	**Symbol**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Unit**	**Notes**
DACK delay time (falling edge)	t_{DACDF}	—	10	—	8	—	6	ns	Relative to CKIO falling edge



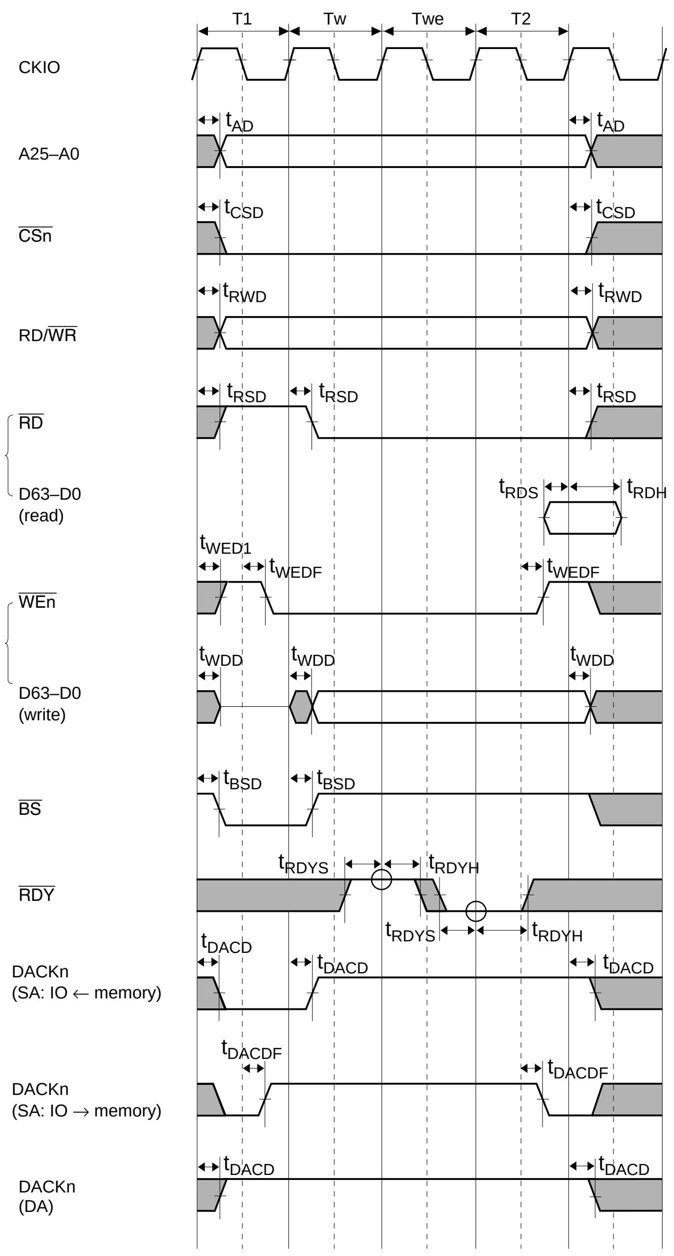
Note: IO: DACK device
SA: Single address DMA transfer
DA: Dual address DMA transfer
DACK set to active-high

Figure 23.14 SRAM Bus Cycle: Basic Bus Cycle (No Wait)



Note: IO: DACK device
SA: Single address DMA transfer
DA: Dual address DMA transfer

Figure 23.15 SRAM Bus Cycle: Basic Bus Cycle (One Internal Wait)



Note: IO: DACK device
SA: Single address DMA transfer
DA: Dual address DMA transfer

Figure 23.16 SRAM Bus Cycle: Basic Bus Cycle (One Internal Wait + One External Wait)

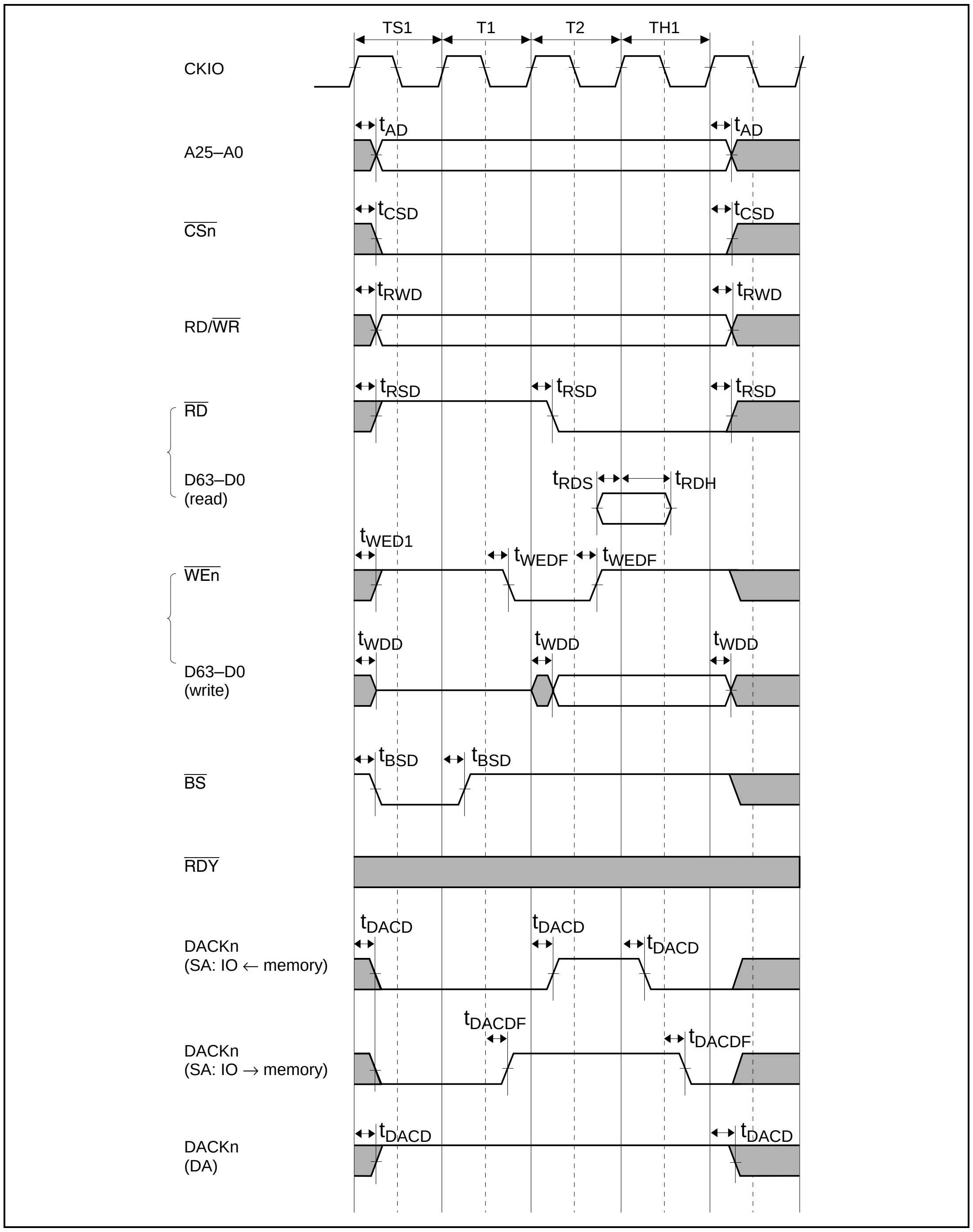
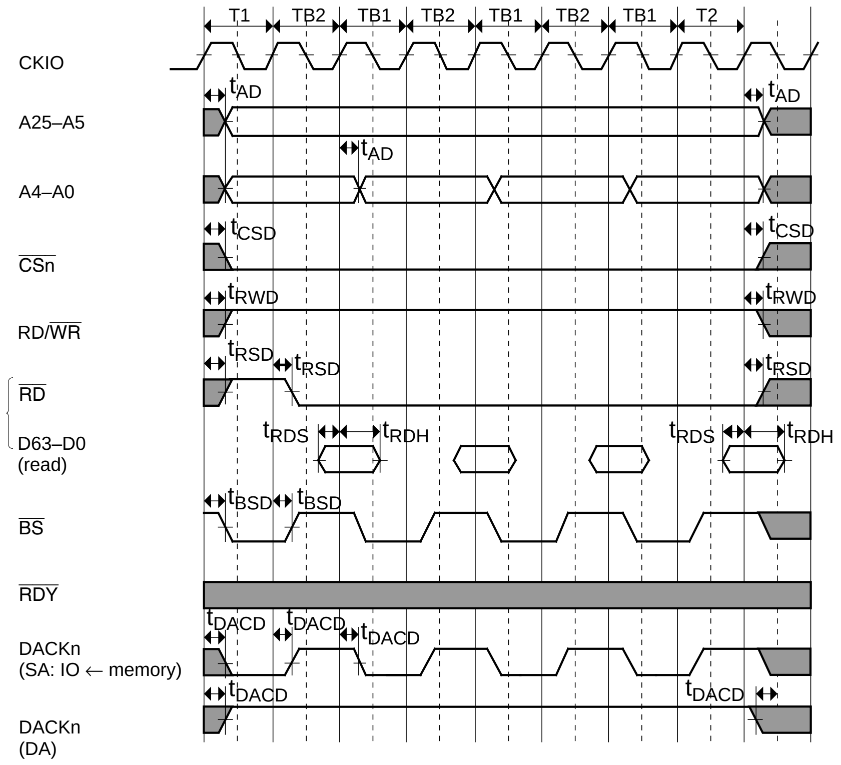


Figure 23.17 SRAM Bus Cycle: Basic Bus Cycle (No Wait, Address Setup/Hold Time Insertion, AnS = 1, AnH = 1)



Note: IO: DACK device
SA: Single address DMA transfer
DA: Dual address DMA transfer
DACK set to active-high

Figure 23.18 Burst ROM Bus Cycle (No Wait)

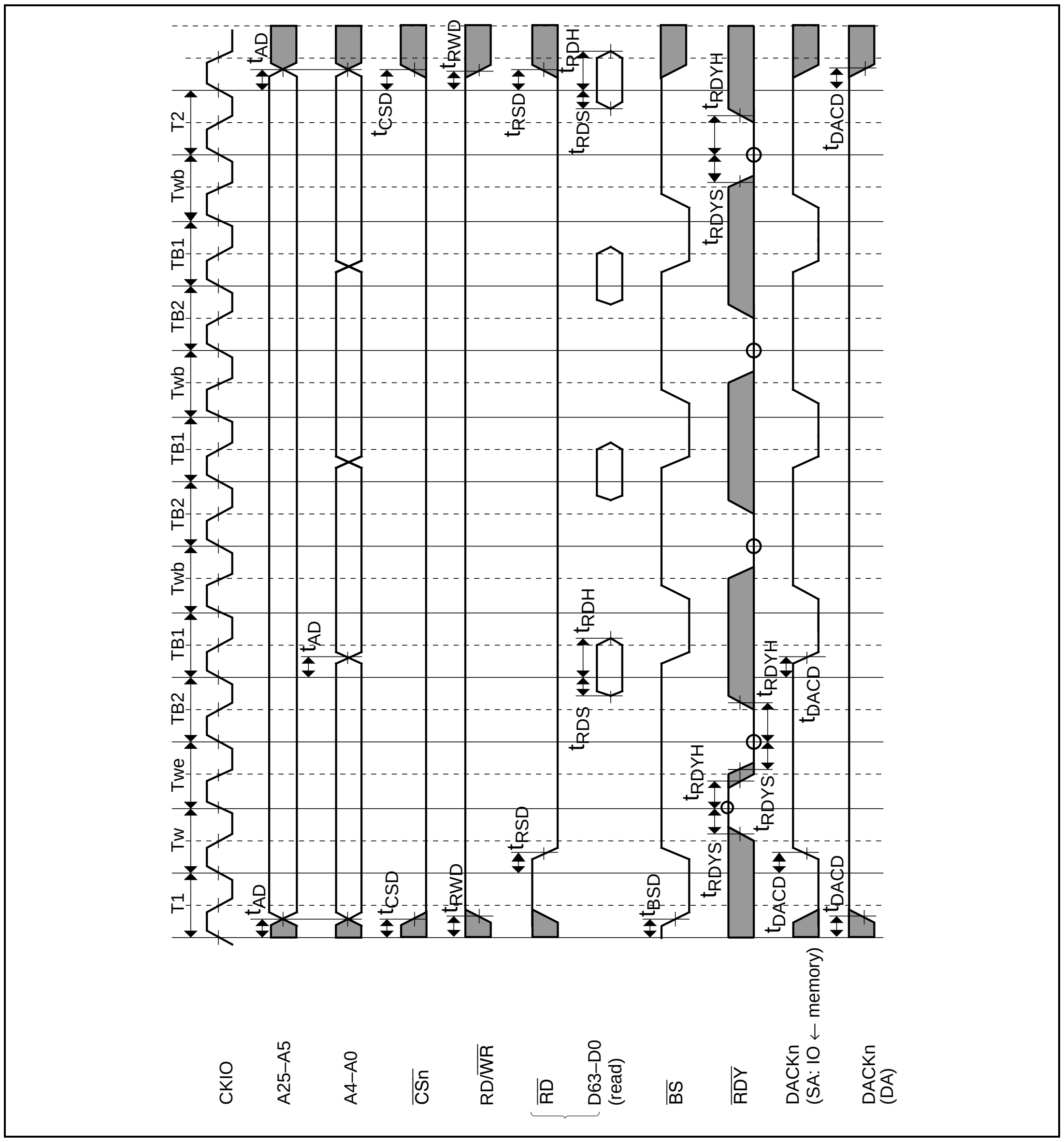


Figure 23.19 Burst ROM Bus Cycle
(1st Data: One Internal Wait + One External Wait; 2nd/3rd/4th Data: One Internal Wait)

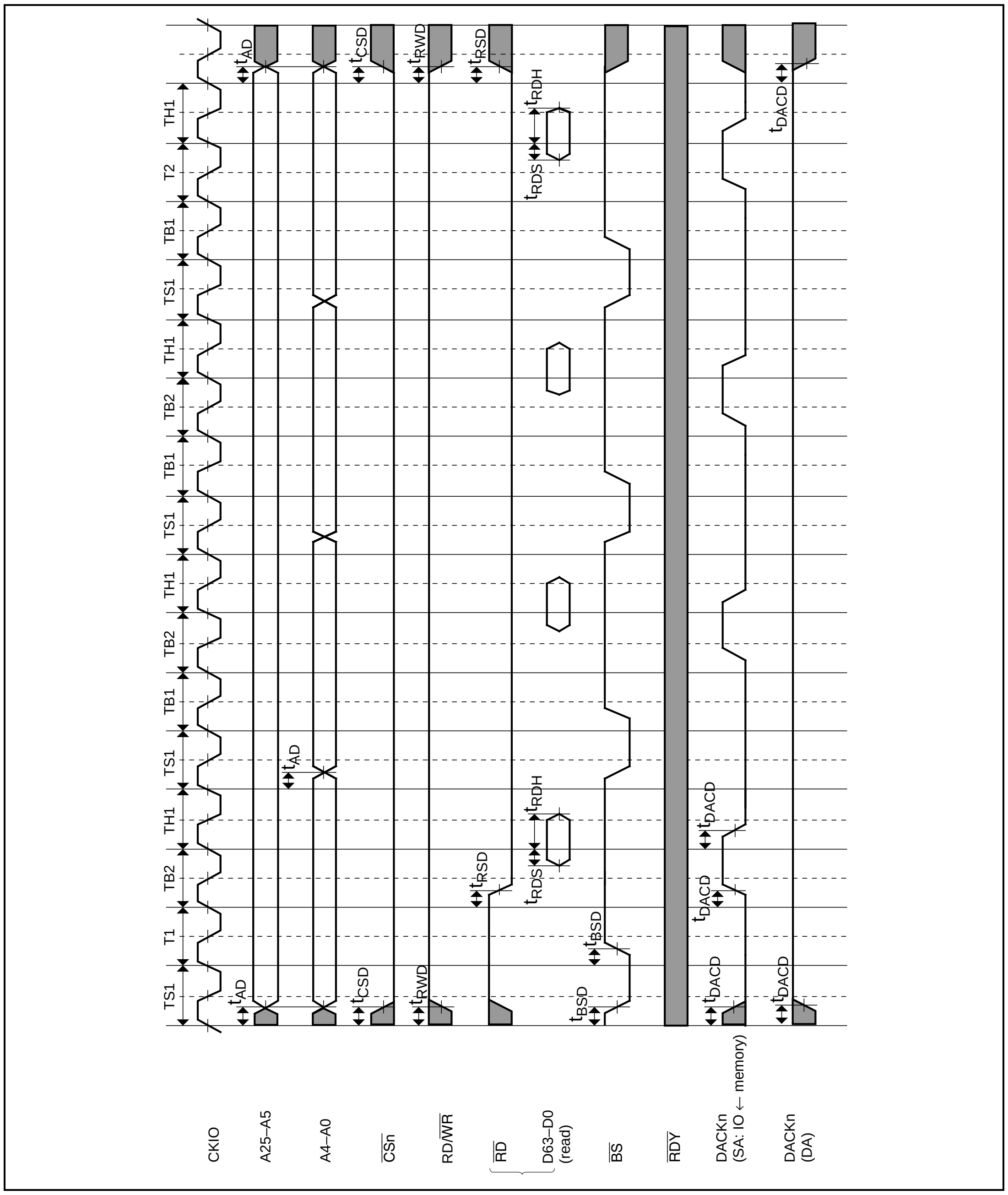


Figure 23.20 Burst ROM Bus Cycle
(No Wait, Address Setup/Hold Time Insertion, AnS = 1, AnH = 1)

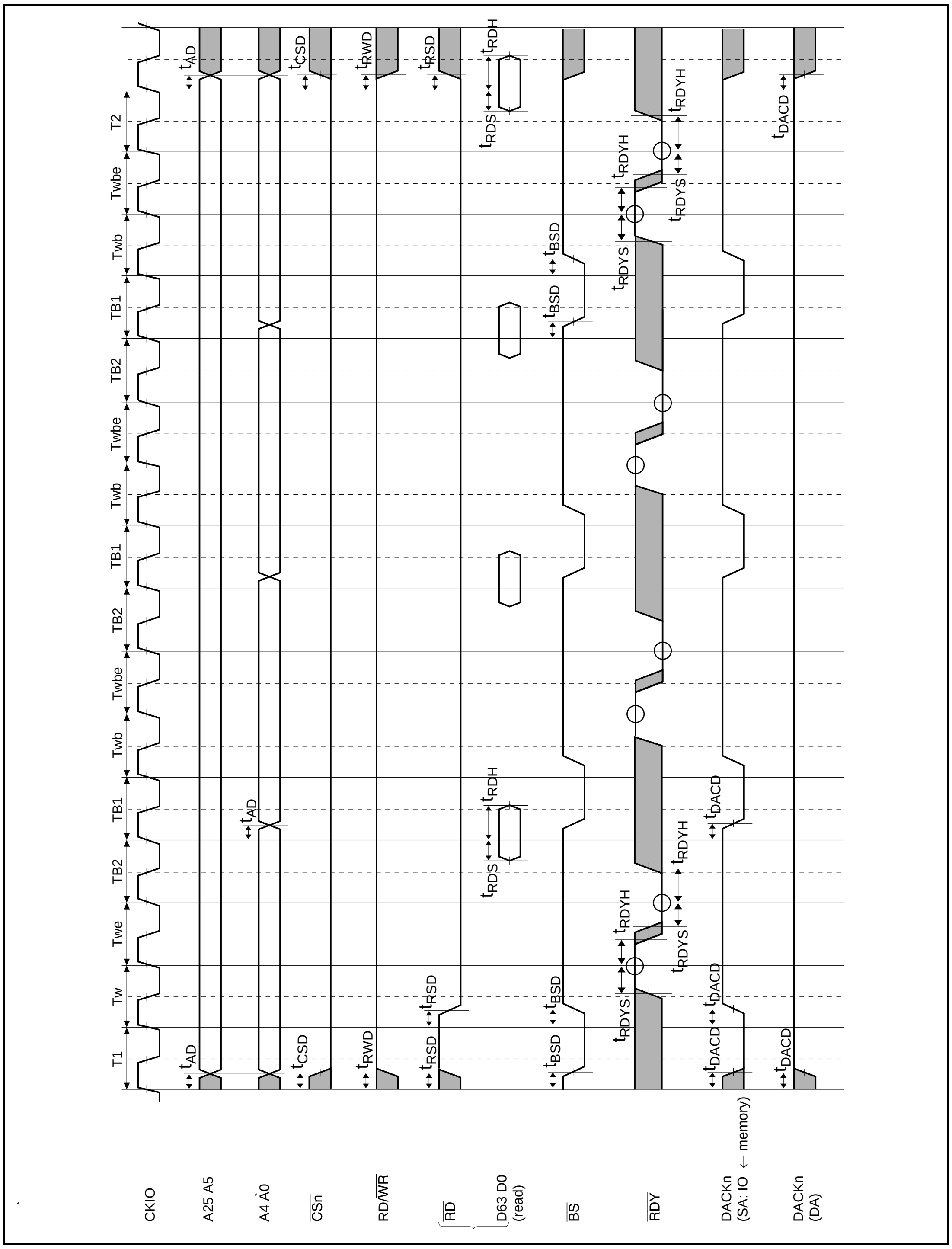


Figure 23.21 Burst ROM Bus Cycle (One Internal Wait + One External Wait)

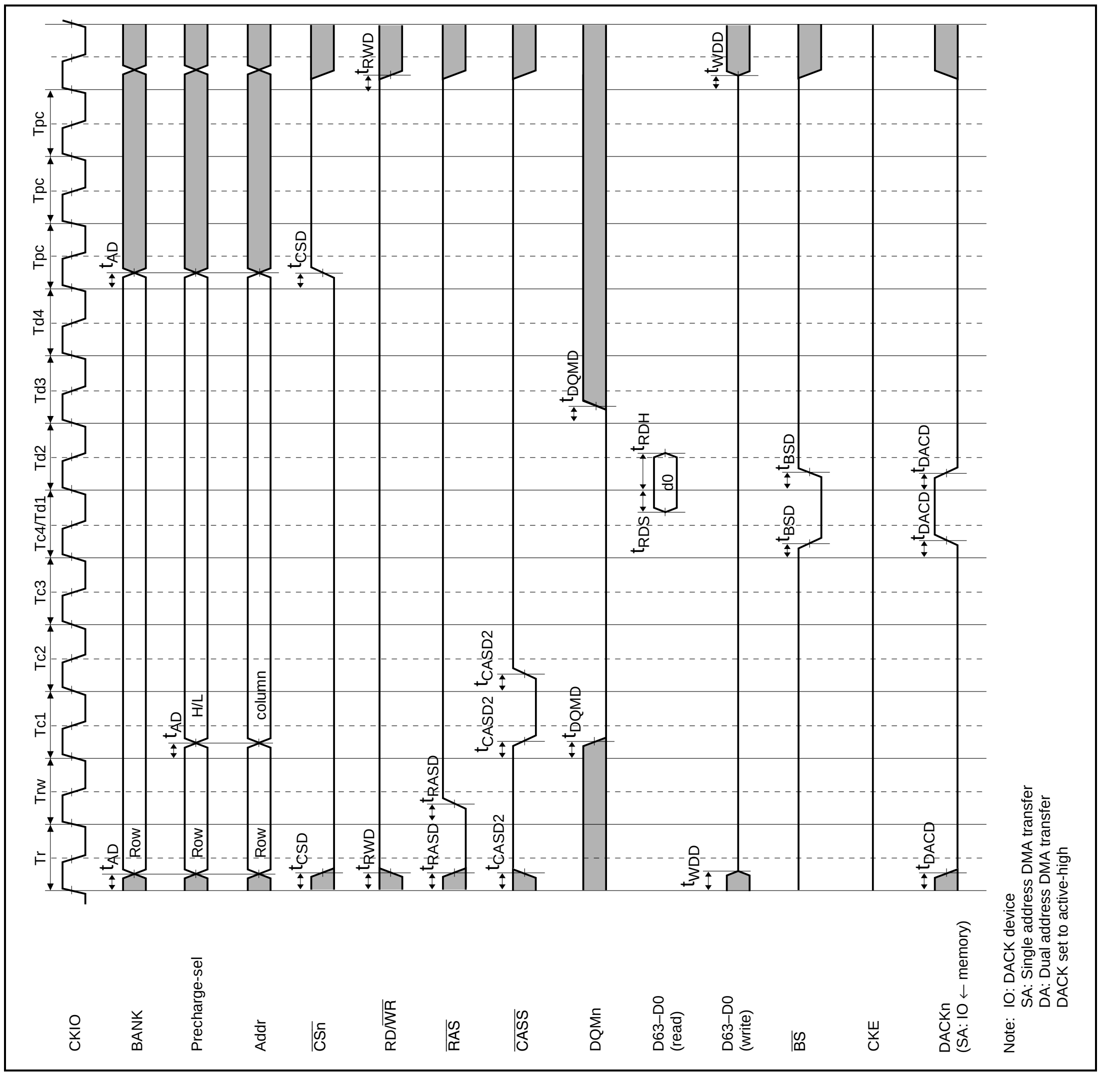


Figure 23.22 Synchronous DRAM Auto-Precharge Bus Cycle: Single (RCD = 1, CAS Latency = 3, TPC = 3)

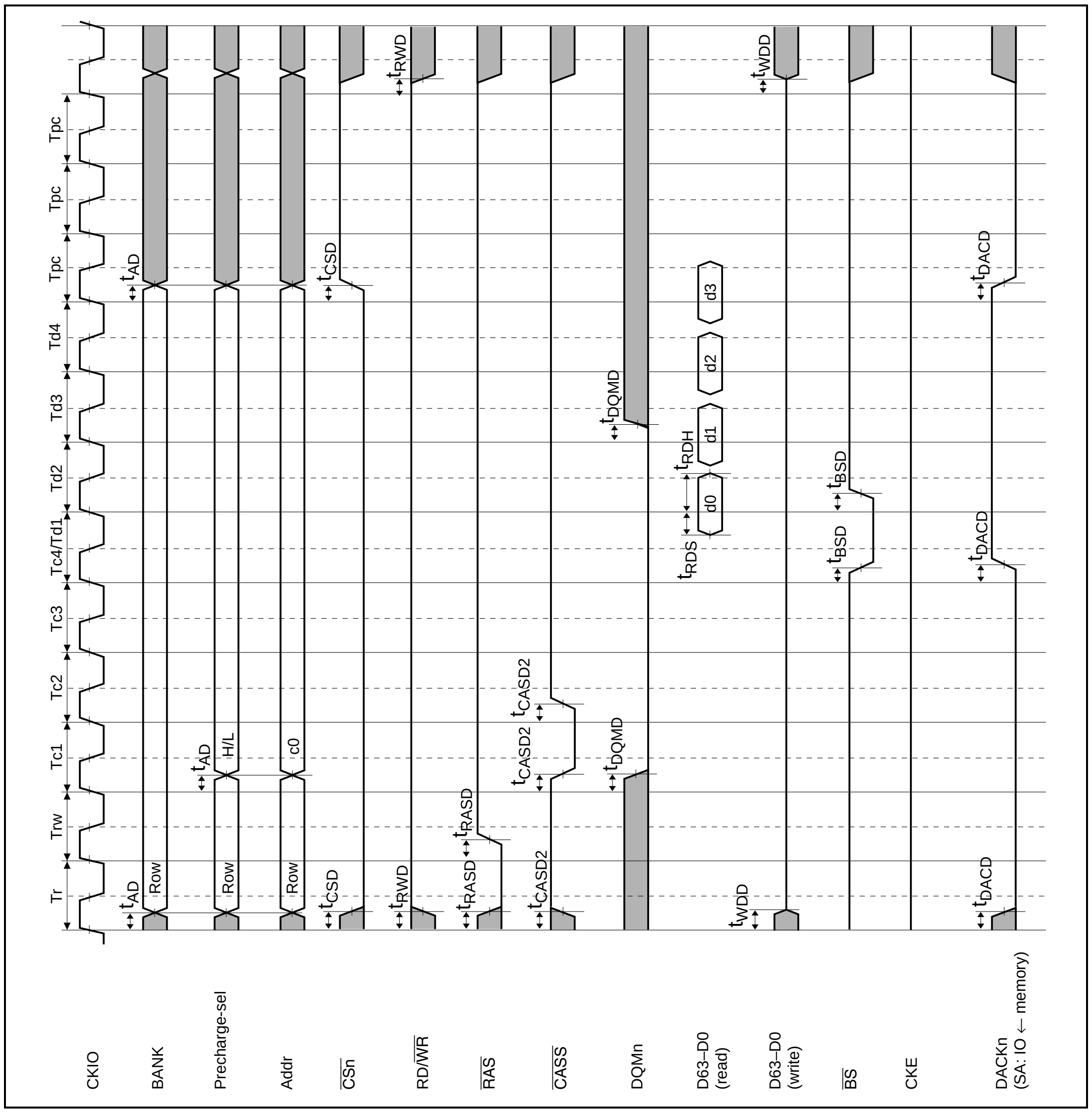


Figure 23.23 Synchronous DRAM Auto-Precharge Read Bus Cycle: Burst (RCD = 1, CAS Latency = 3, TPC = 3)

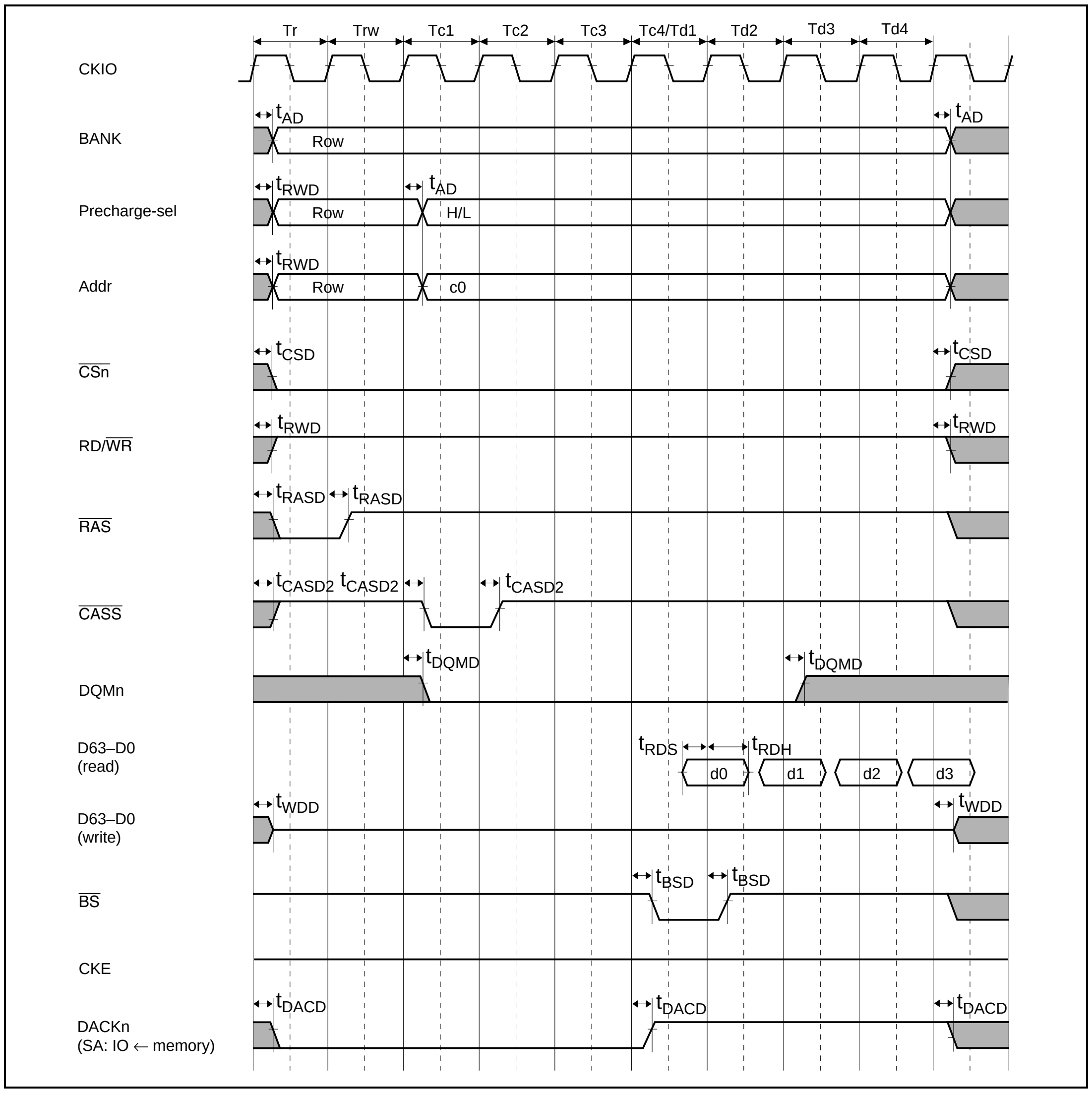


Figure 23.24 Synchronous DRAM Normal Read Bus Cycle: ACT + READ Commands, Burst (RCD = 1, CAS Latency = 3)

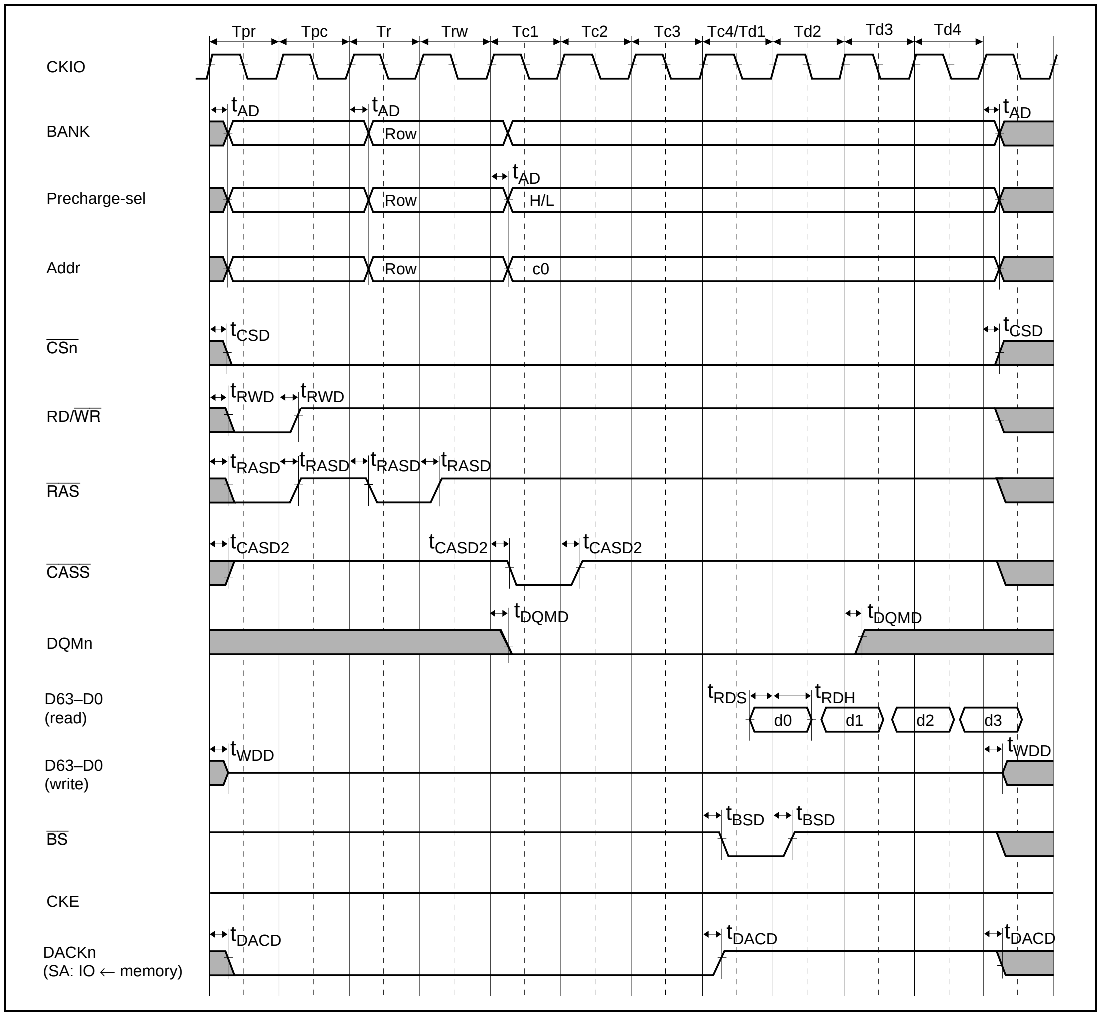


Figure 23.25 Synchronous DRAM Normal Read Bus Cycle: PRE + ACT + READ Commands, Burst (TPC = 1, RCD = 1, CAS Latency = 3)

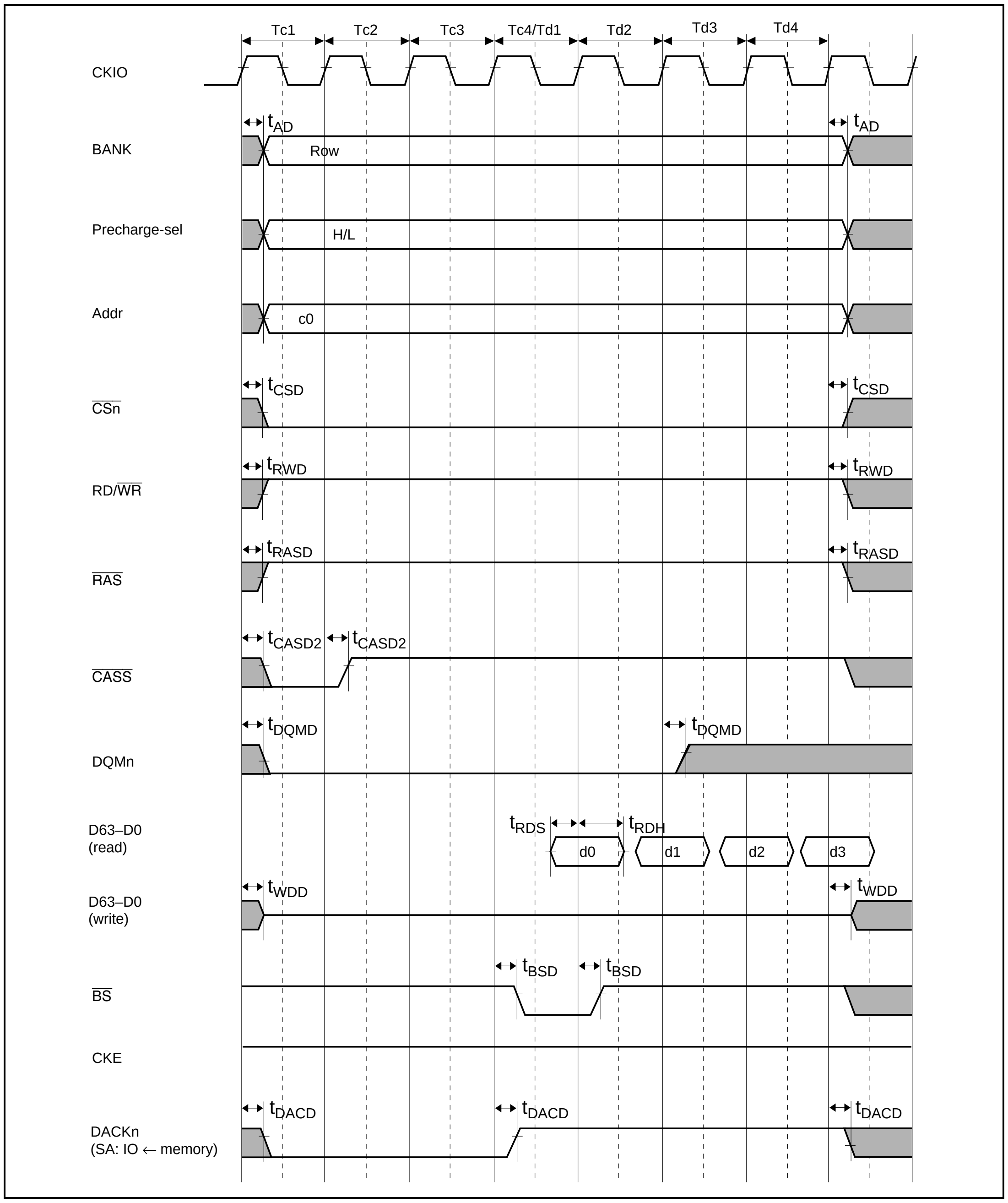


Figure 23.26 Synchronous DRAM Normal Read Bus Cycle: READ Command, Burst (CAS Latency = 3)

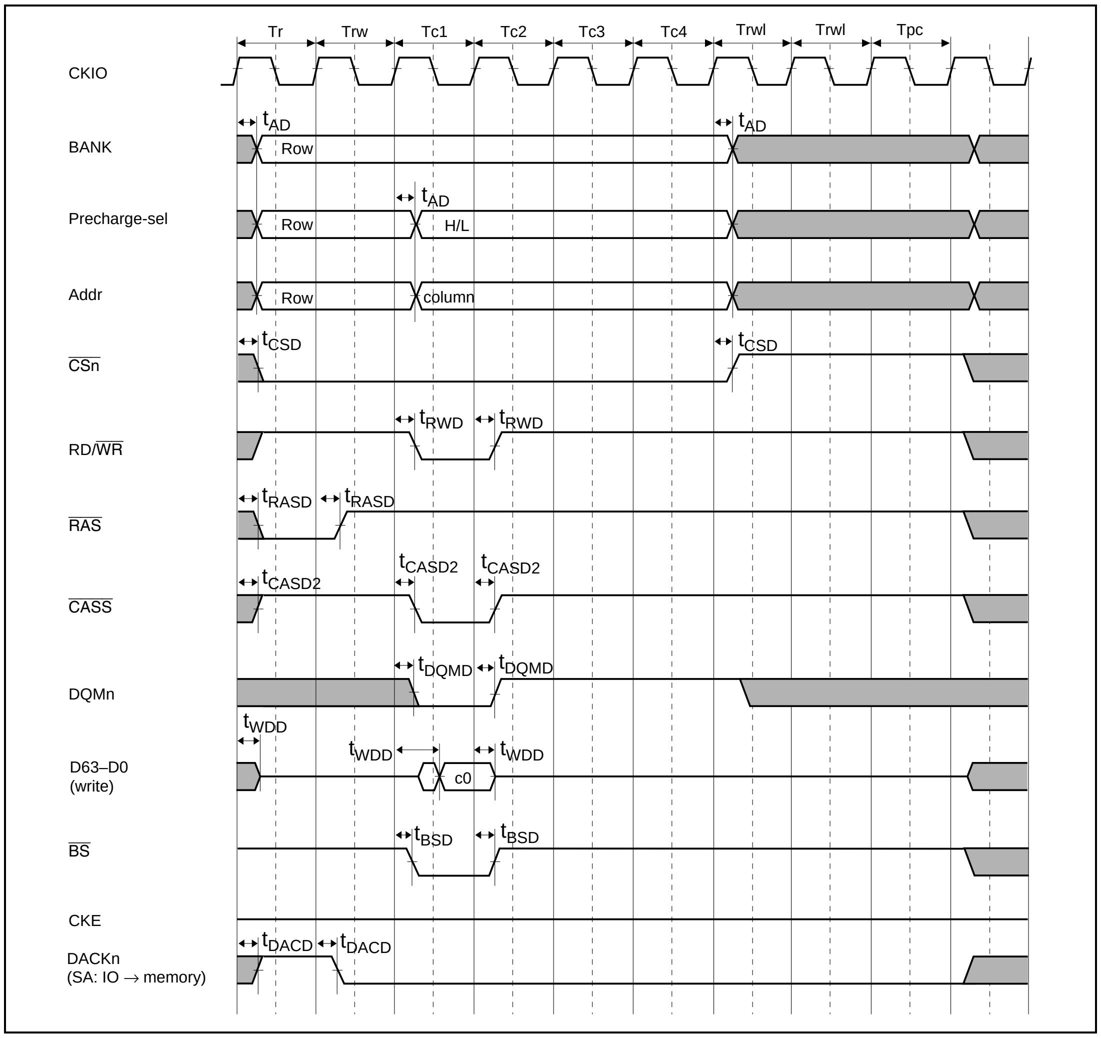


Figure 23.27 Synchronous DRAM Auto-Precharge Write Bus Cycle: Single (RCD = 1, TRWL = 2, TPC = 1)

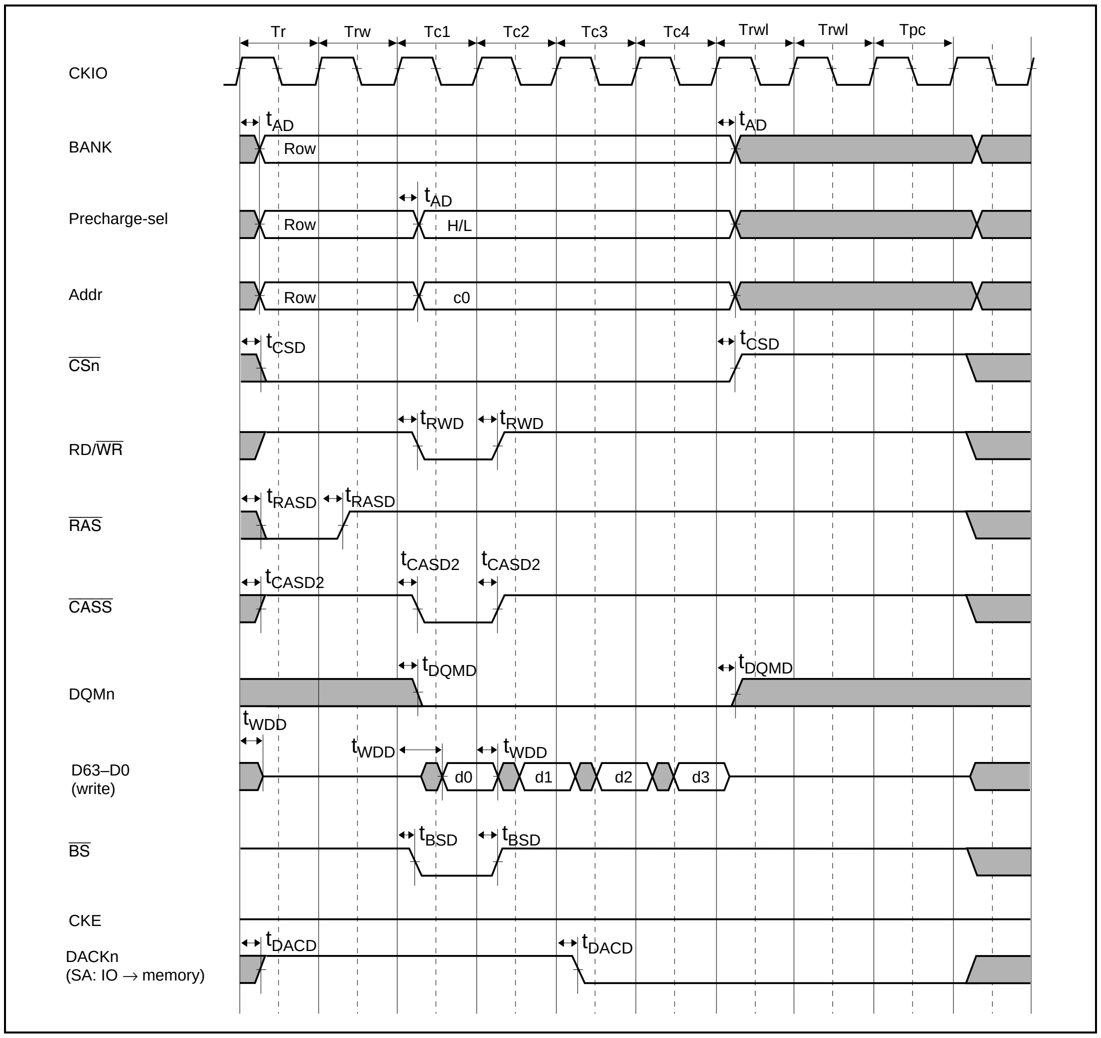


Figure 23.28 Synchronous DRAM Auto-Precharge Write Bus Cycle: Burst (RCD = 1, TRWL = 2, TPC = 1)

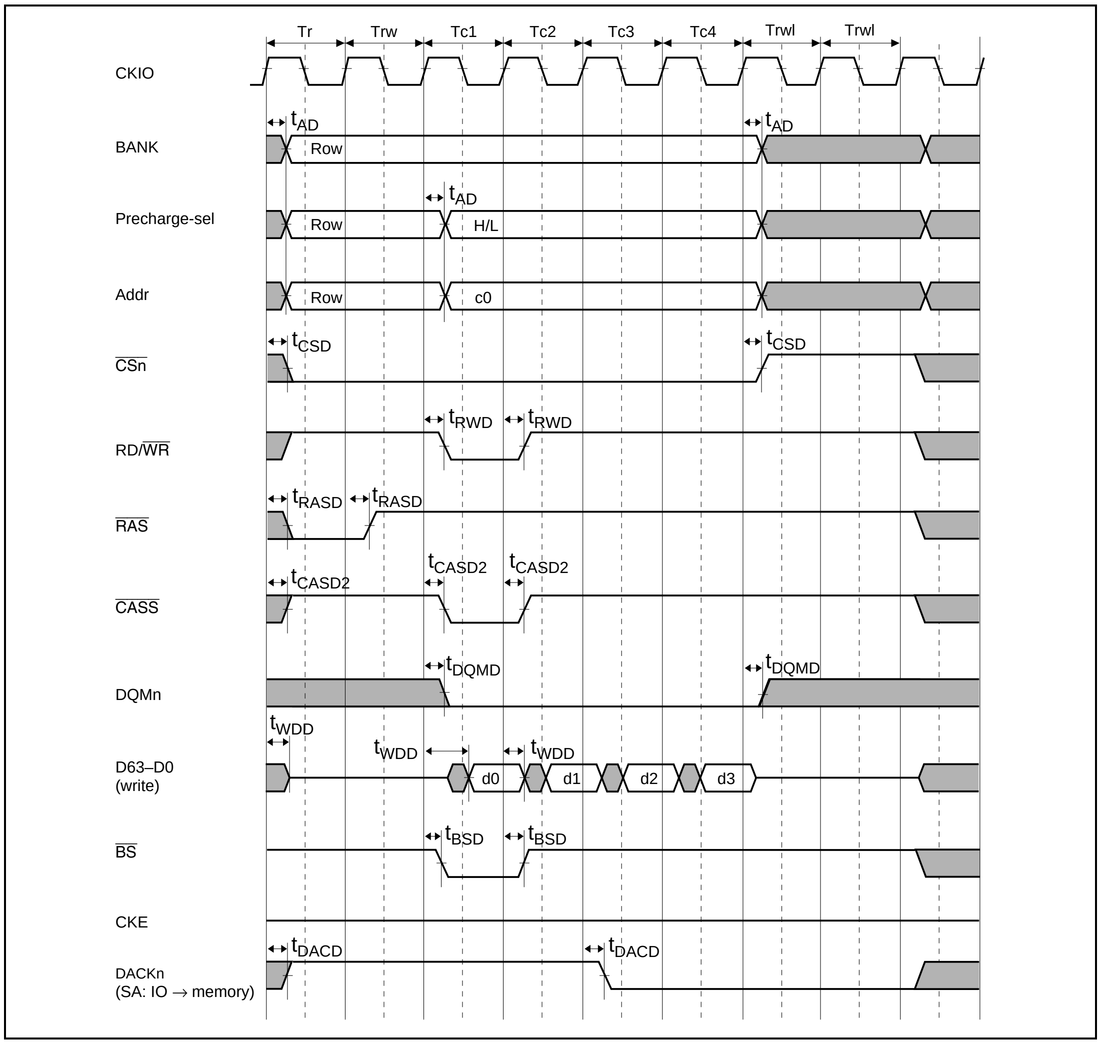


Figure 23.29 Synchronous DRAM Normal Write Bus Cycle: ACT + WRITE Commands, Burst (RCD = 1, TRWL = 2)

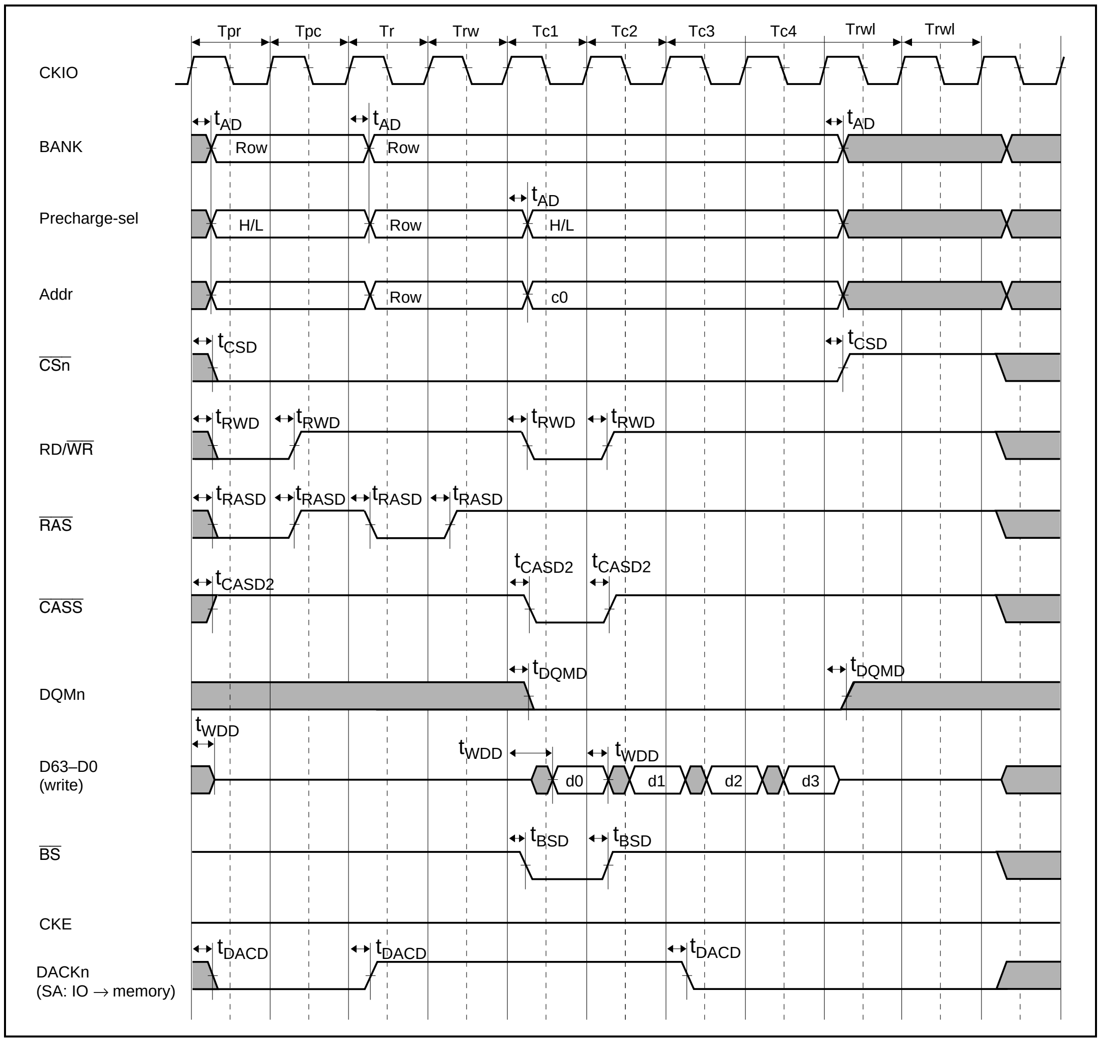
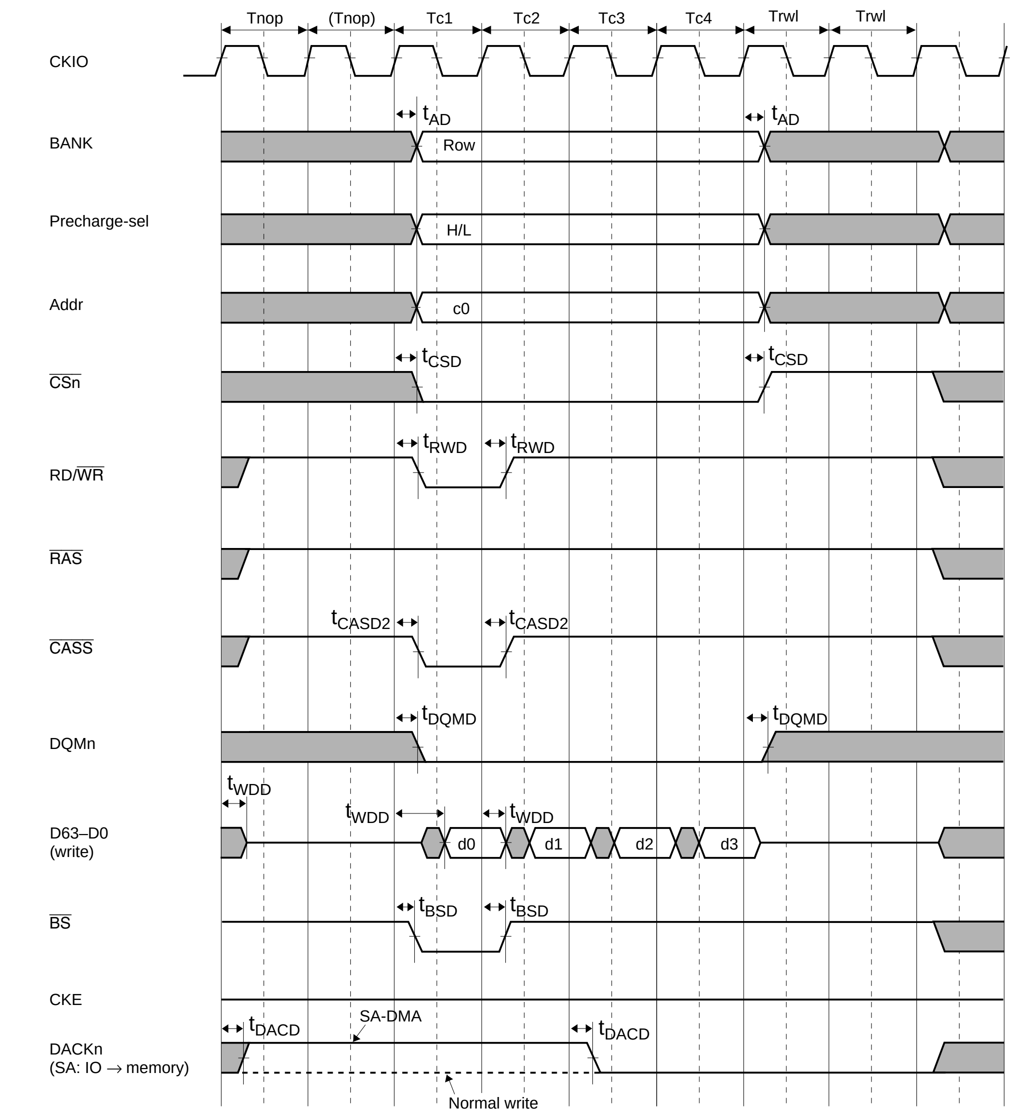


Figure 23.30 Synchronous DRAM Normal Write Bus Cycle: PRE + ACT + WRITE Commands, Burst (TPC = 1, RCD = 1, TRWL = 2)



Note: In the case of SA-DMA only, the (Tnop) cycle is inserted, and the DACKn signal is output as shown by the solid line. In a normal write, the (Tnop) cycle is omitted and the DACKn signal is output as shown by the dotted line.

Figure 23.31 Synchronous DRAM Normal Write Bus Cycle: WRITE Command, Burst (TRWL = 2)

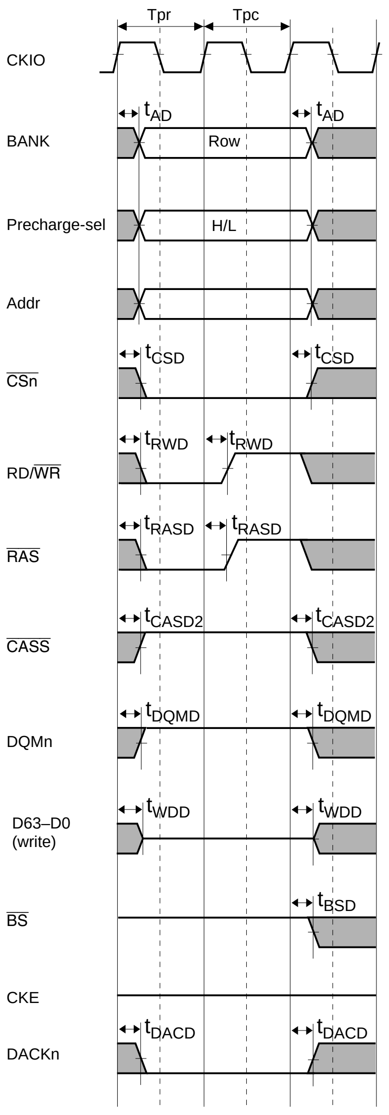


Figure 23.32 Synchronous DRAM Bus Cycle: Synchronous DRAM Precharge Command (TPC = 1)

HITACHI

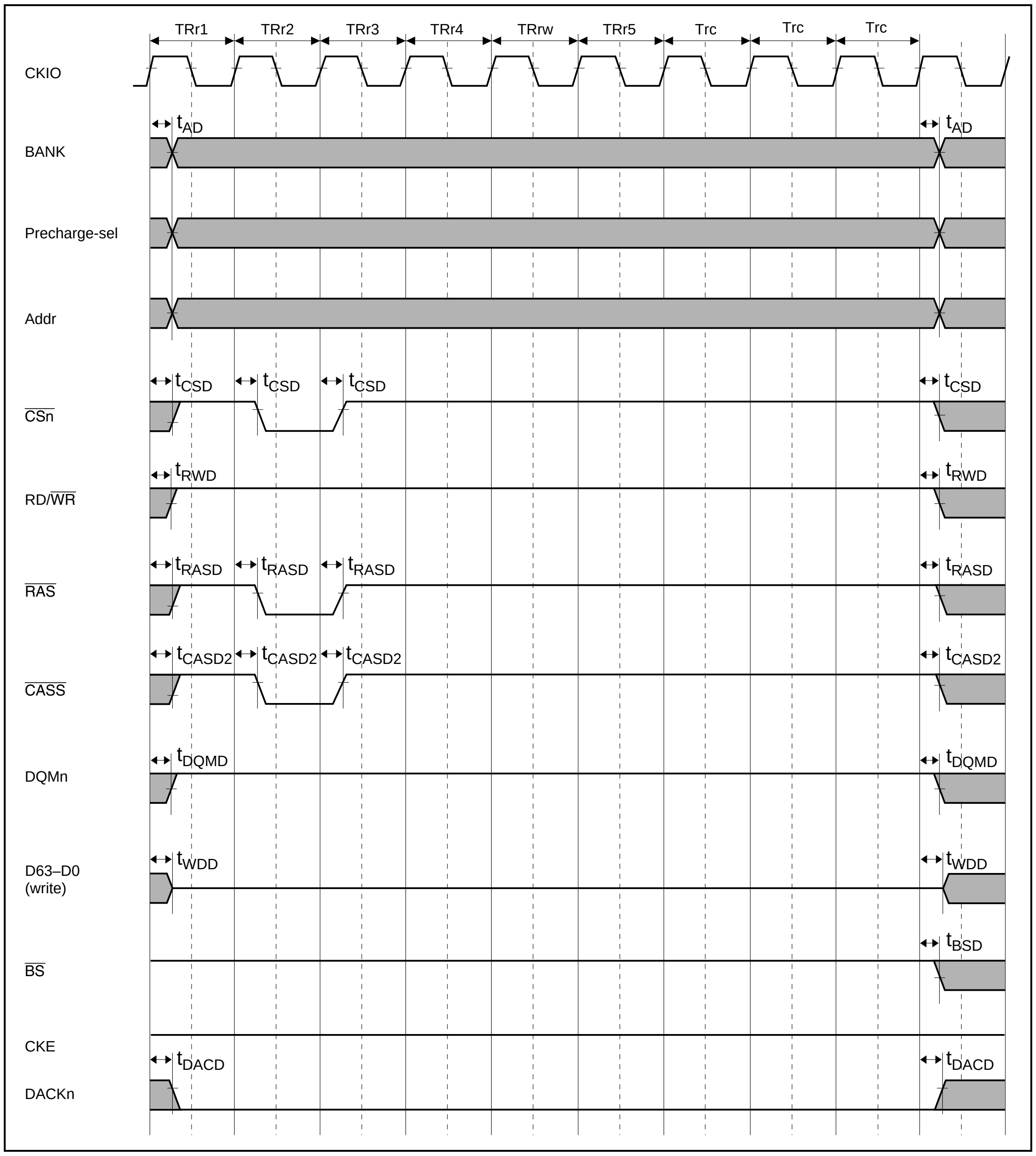


Figure 23.33 Synchronous DRAM Bus Cycle: Synchronous DRAM Auto-Refresh (TRAS = 1, TRC = 1)

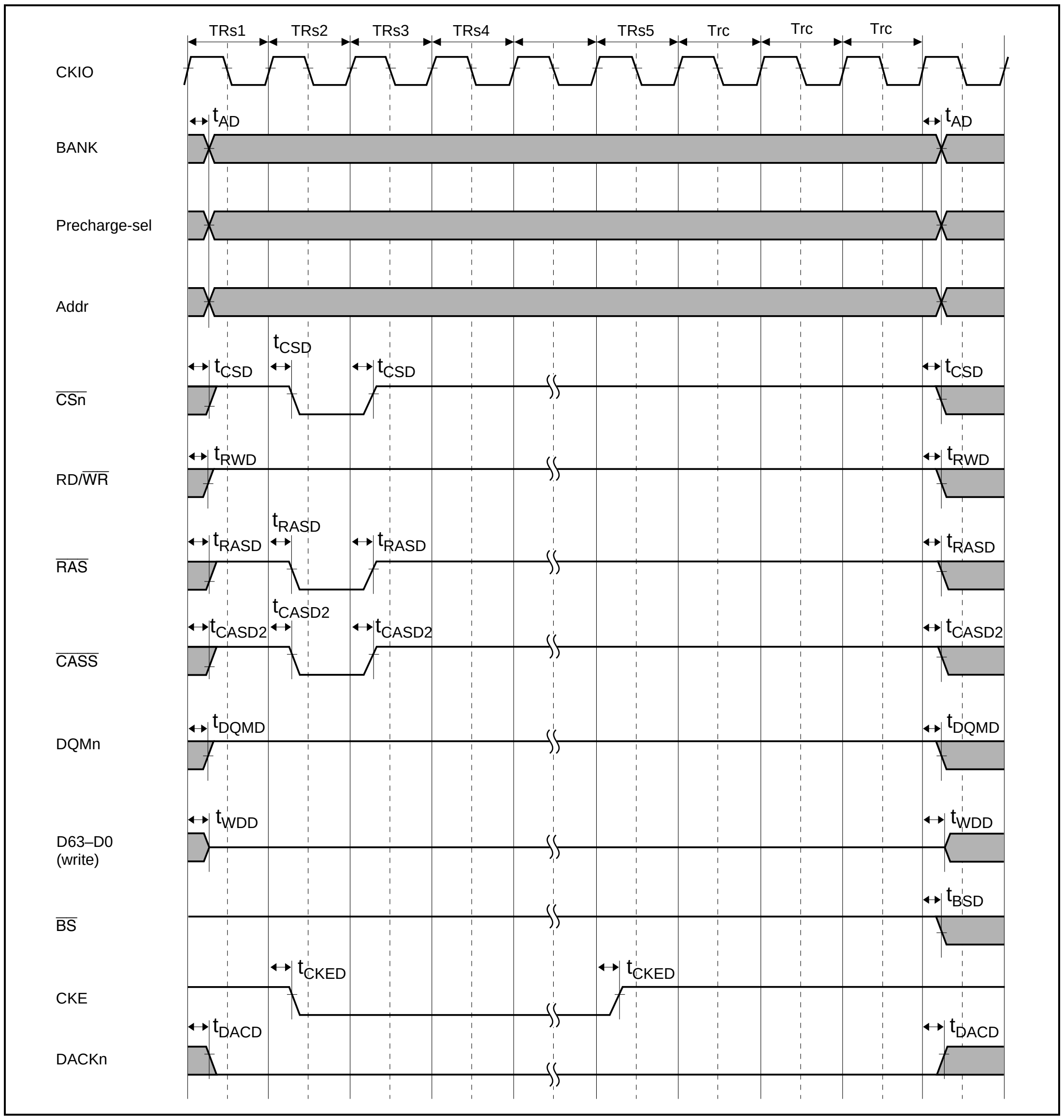


Figure 23.34 Synchronous DRAM Bus Cycle: Synchronous DRAM Self-Refresh (TRC = 1)

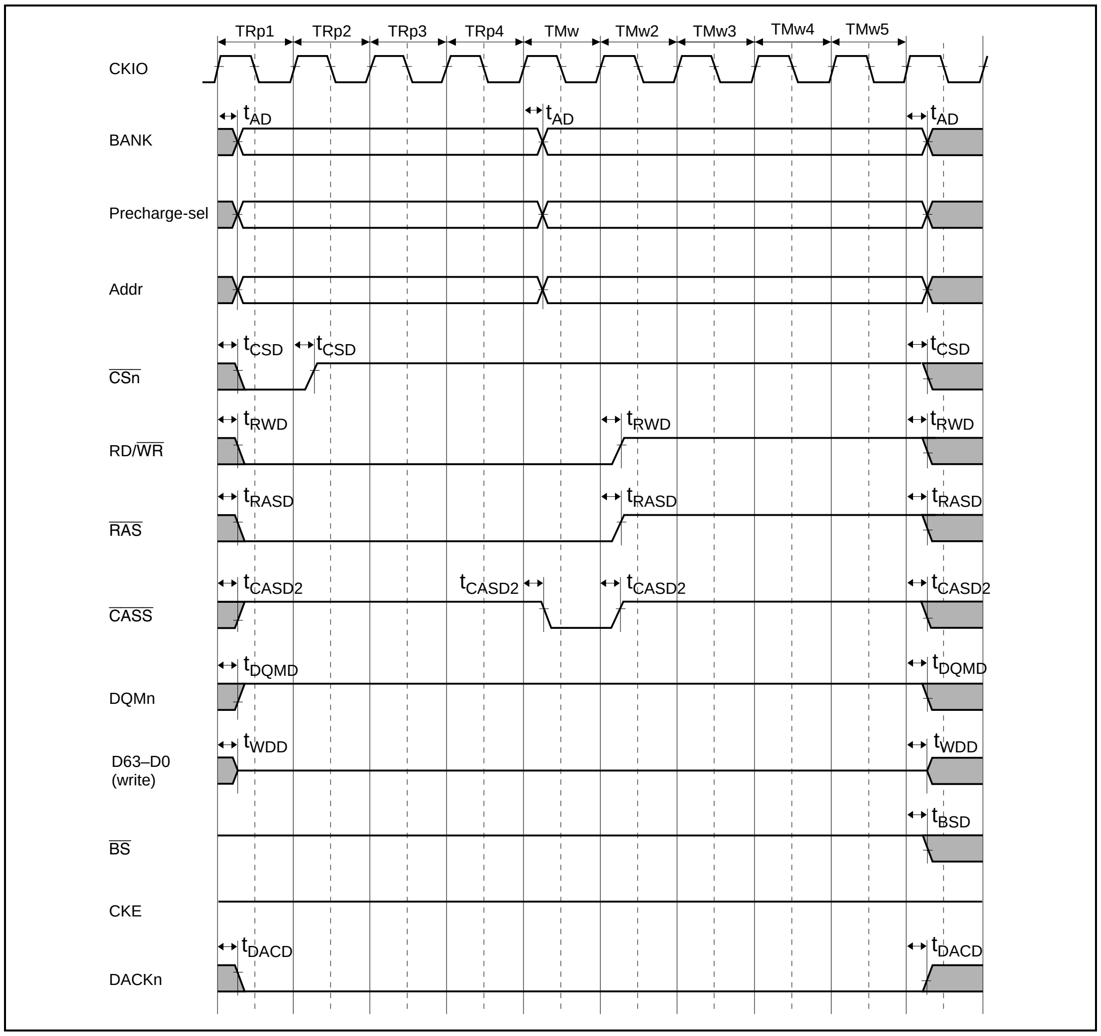


Figure 23.35 (a) Synchronous DRAM Bus Cycle: Synchronous DRAM Mode Register Setting (PALL)

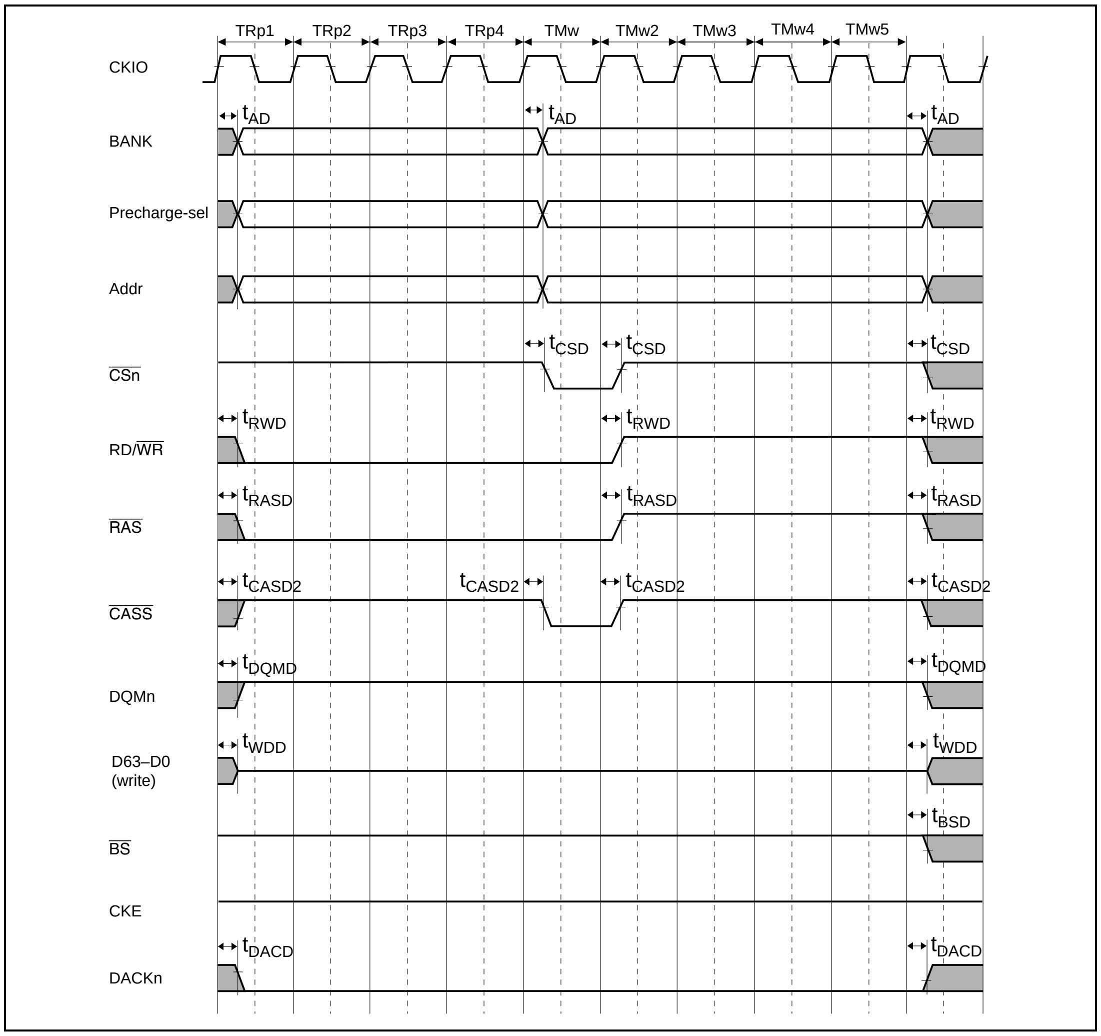


Figure 23.35 (b) Synchronous DRAM Bus Cycle: Synchronous DRAM Mode Register Setting (SET)

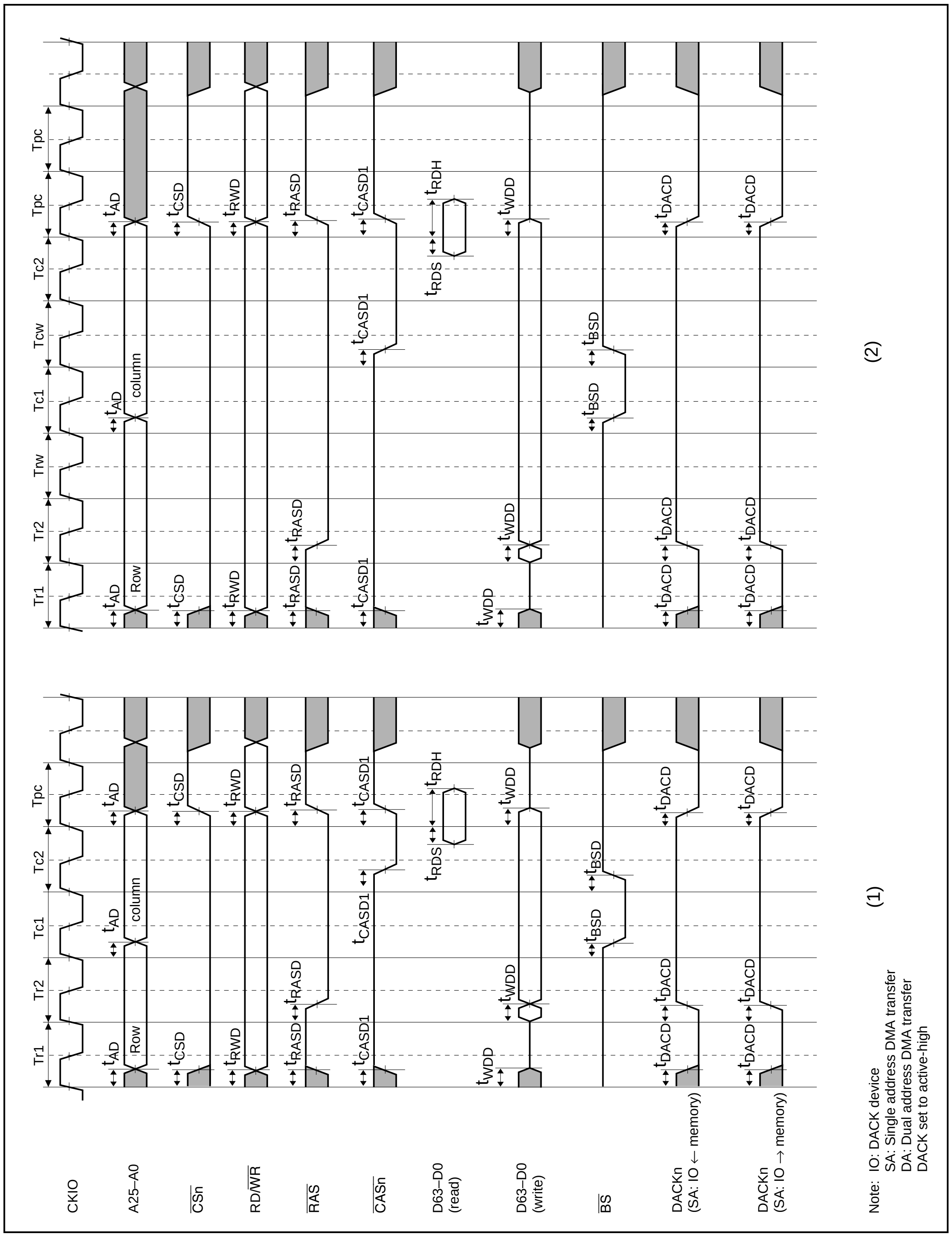


Figure 23.36 DRAM Bus Cycles
((1) RCD = 0, AnW = 0, TPC = 1; (2) RCD = 1, AnW = 1, TPC = 2)

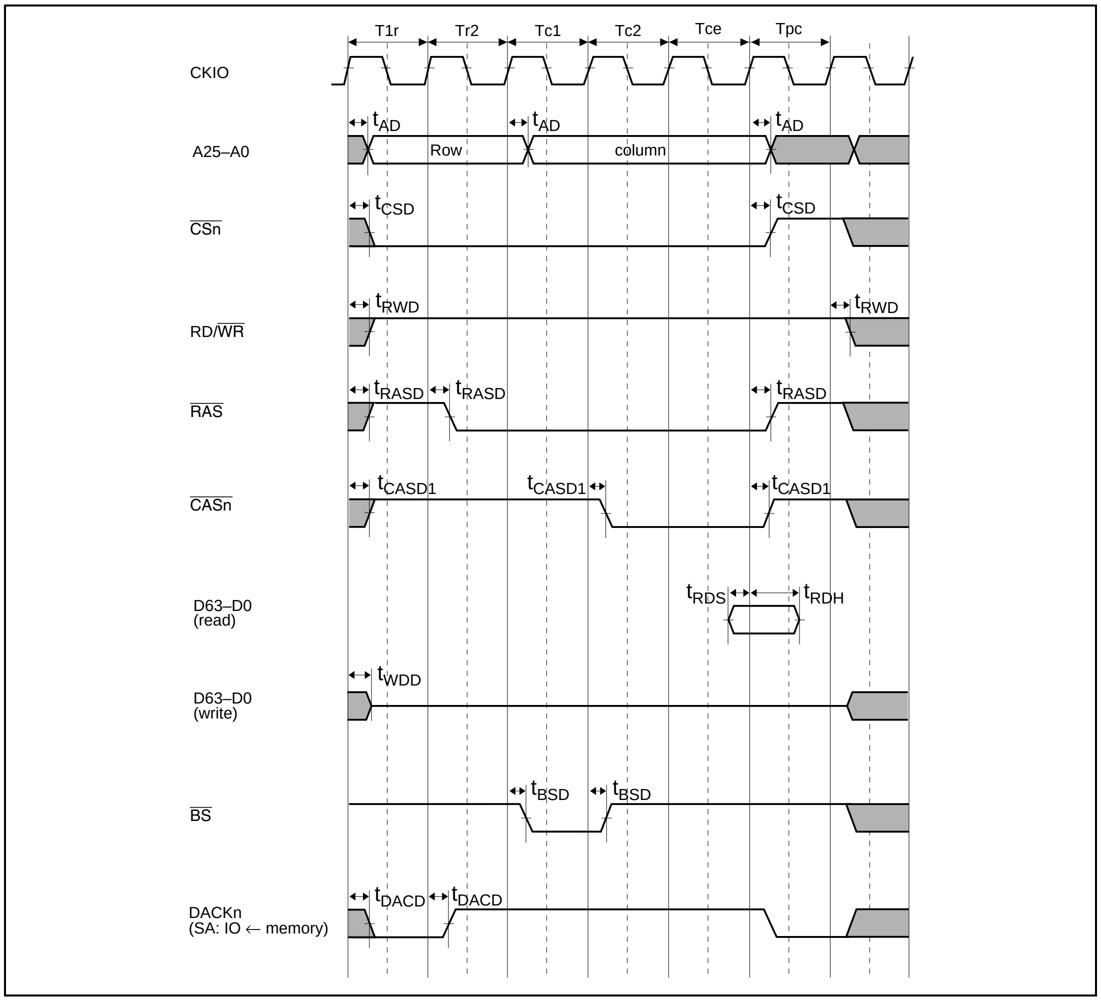


Figure 23.37 DRAM Bus Cycle (EDO Mode, RCD = 0, AnW = 0, TPC = 1)

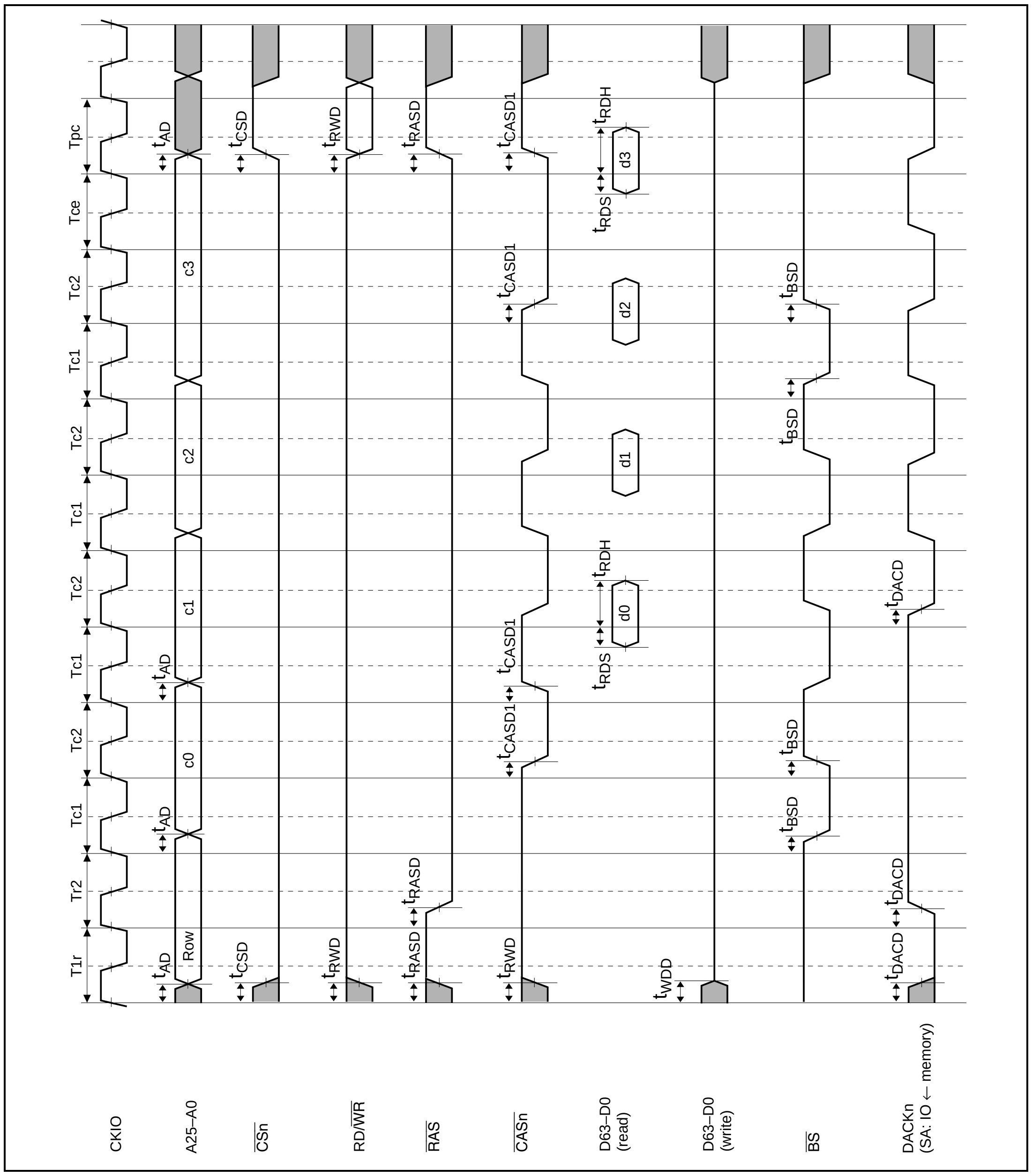


Figure 23.38 DRAM Burst Bus Cycle (EDO Mode, RCD = 0, AnW = 0, TPC = 1)

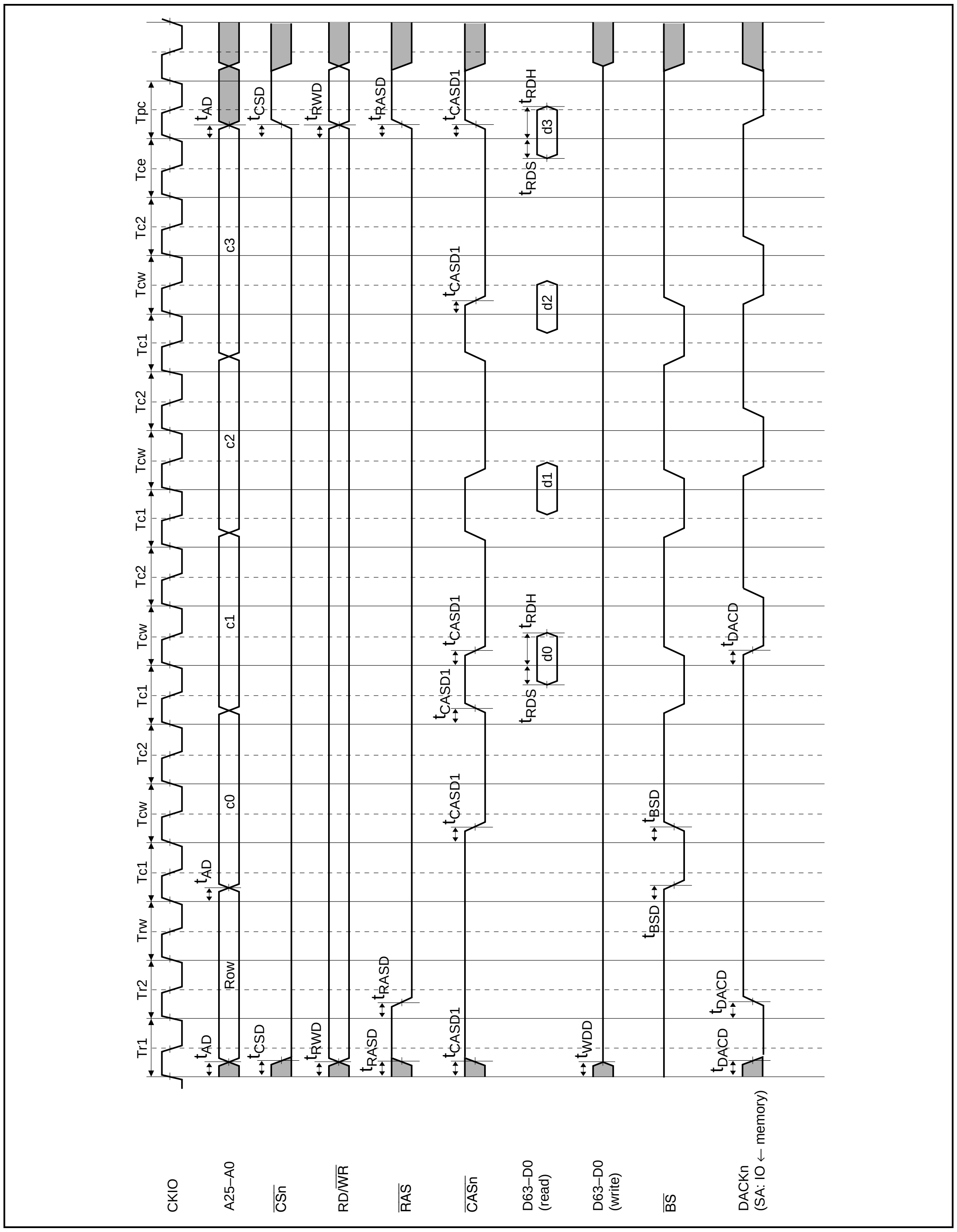


Figure 23.39 DRAM Burst Bus Cycle (EDO Mode, RCD = 1, AnW = 1, TPC = 1)

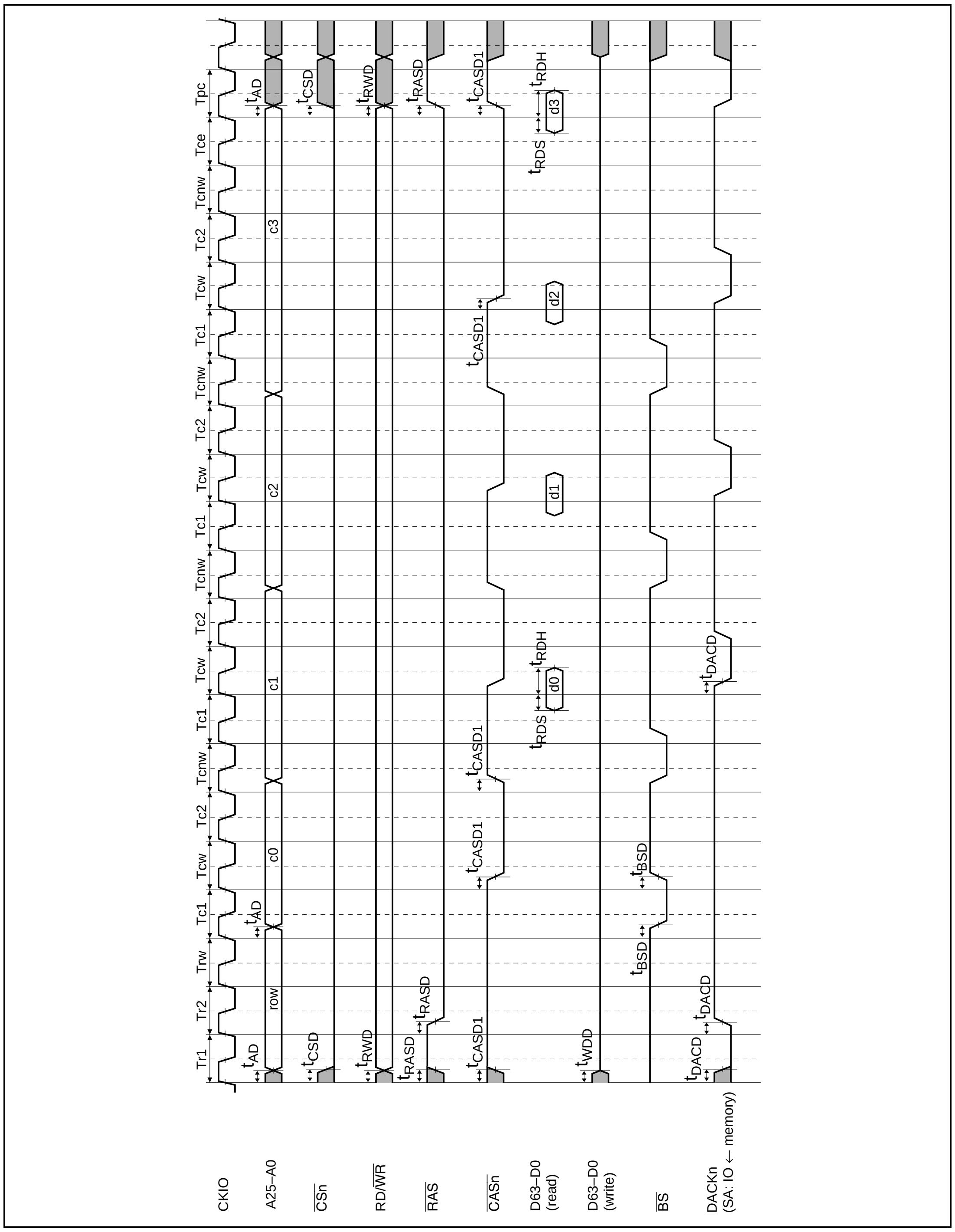


Figure 23.40 DRAM Burst Bus Cycle (EDO Mode, RCD = 1, AnW = 1, TPC = 1, 2-Cycle CAS Negate Pulse Width)

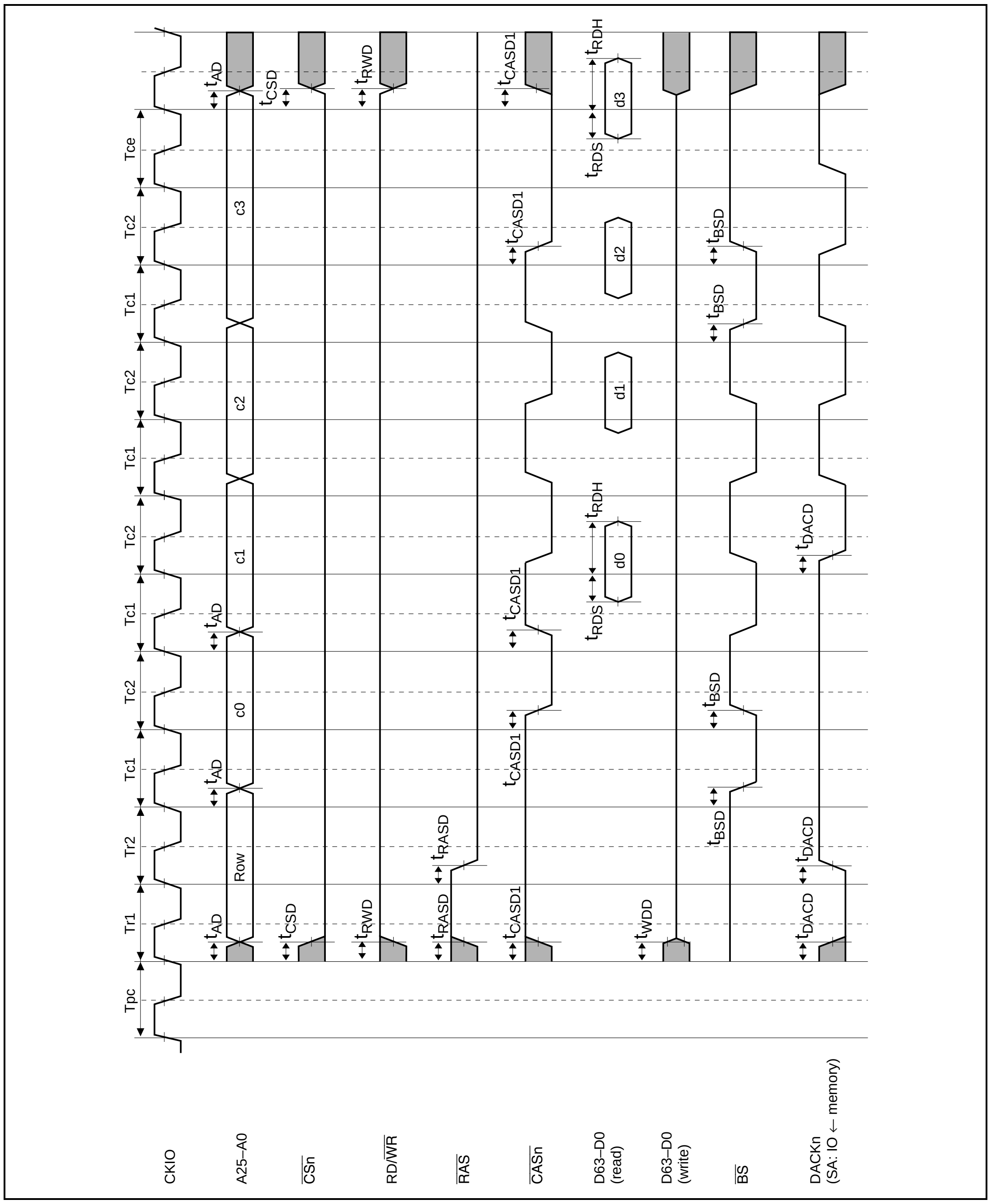


Figure 23.41 DRAM Burst Bus Cycle: RAS Down Mode State (EDO Mode, RCD = 0, AnW = 0)

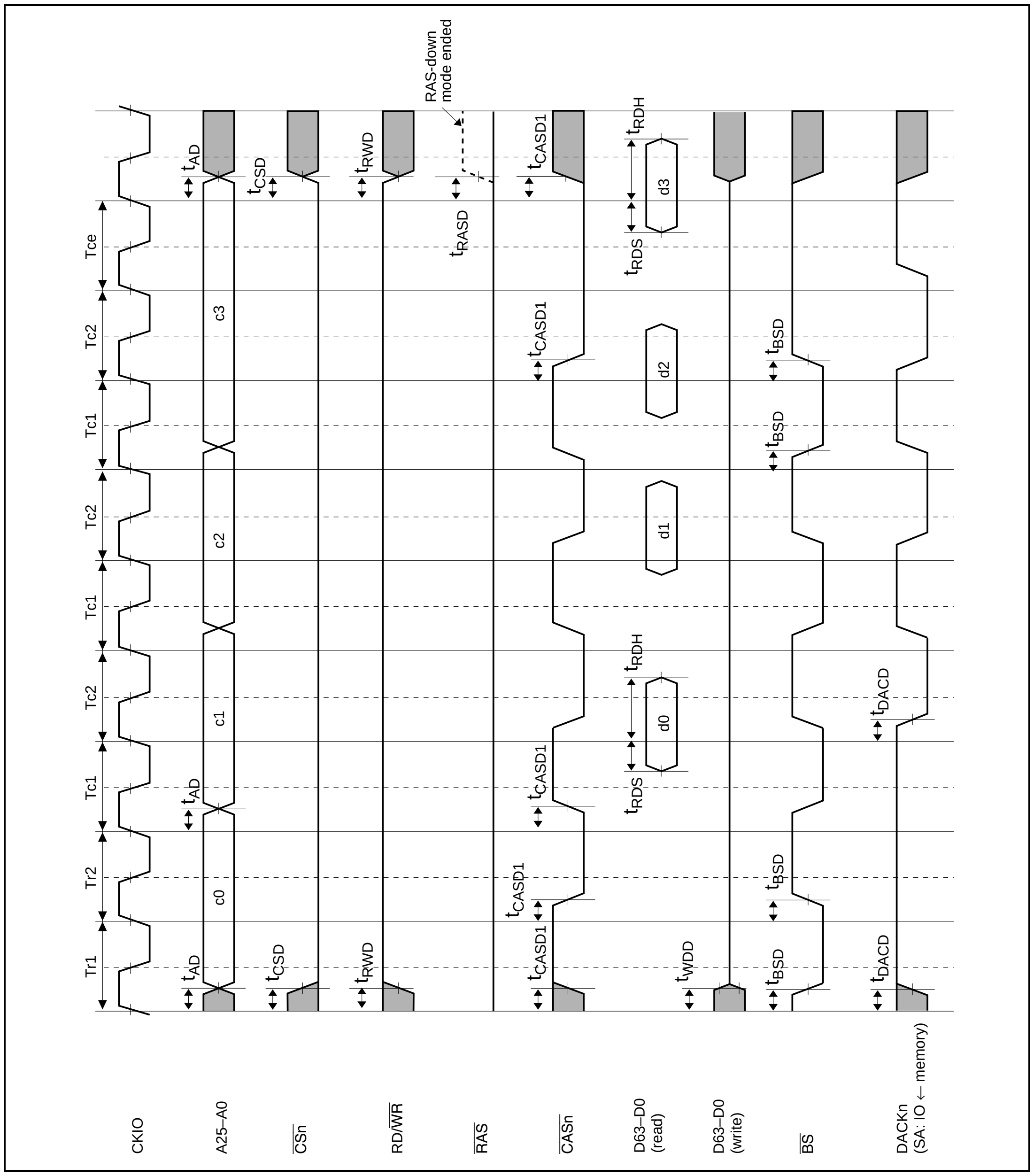


Figure 23.42 DRAM Burst Bus Cycle: RAS Down Mode Continuation (EDO Mode, RCD = 0, AnW = 0)

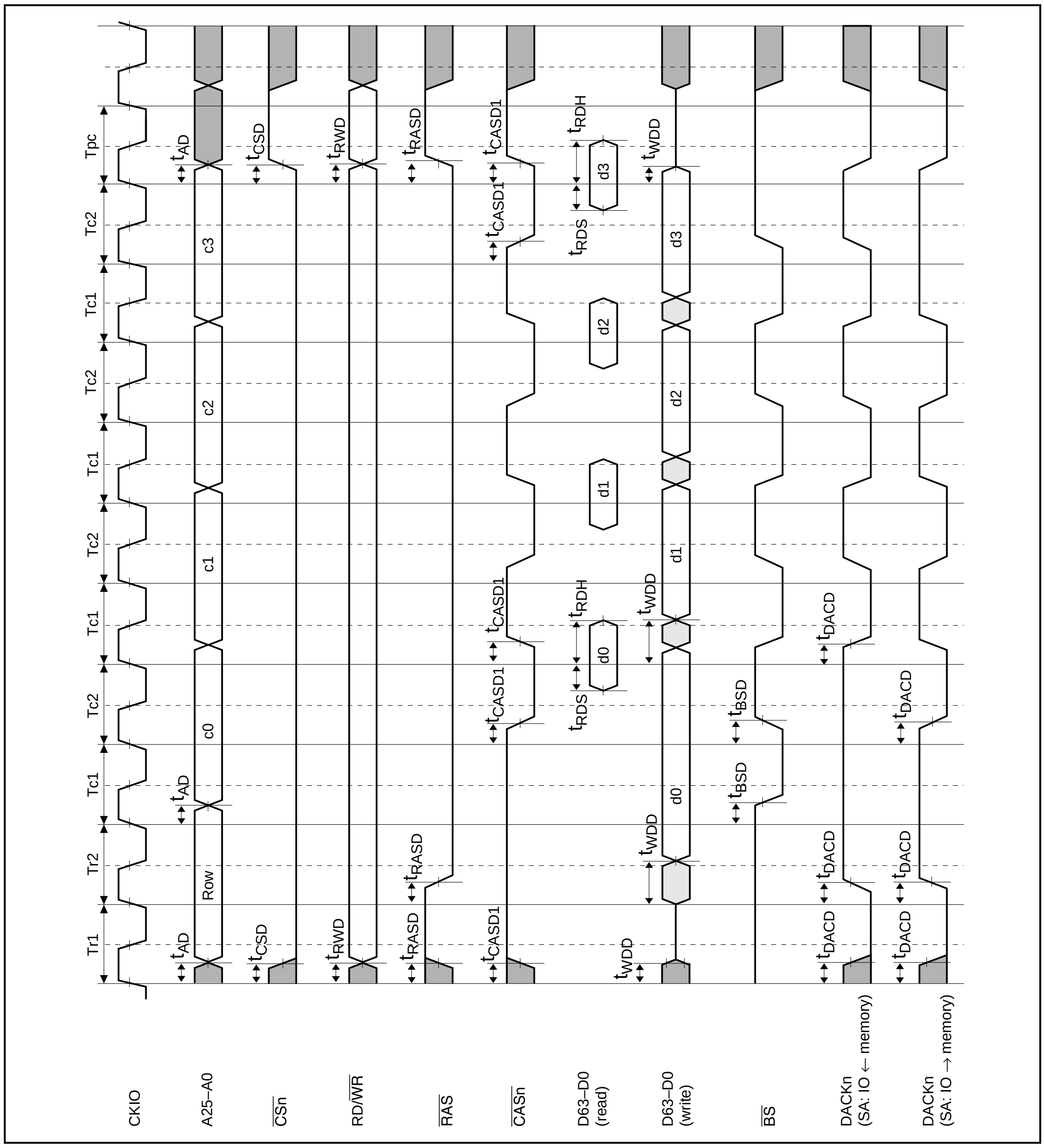


Figure 23.43 DRAM Burst Bus Cycle (Fast Page Mode, RCD = 0, AnW = 0, TPC = 1)

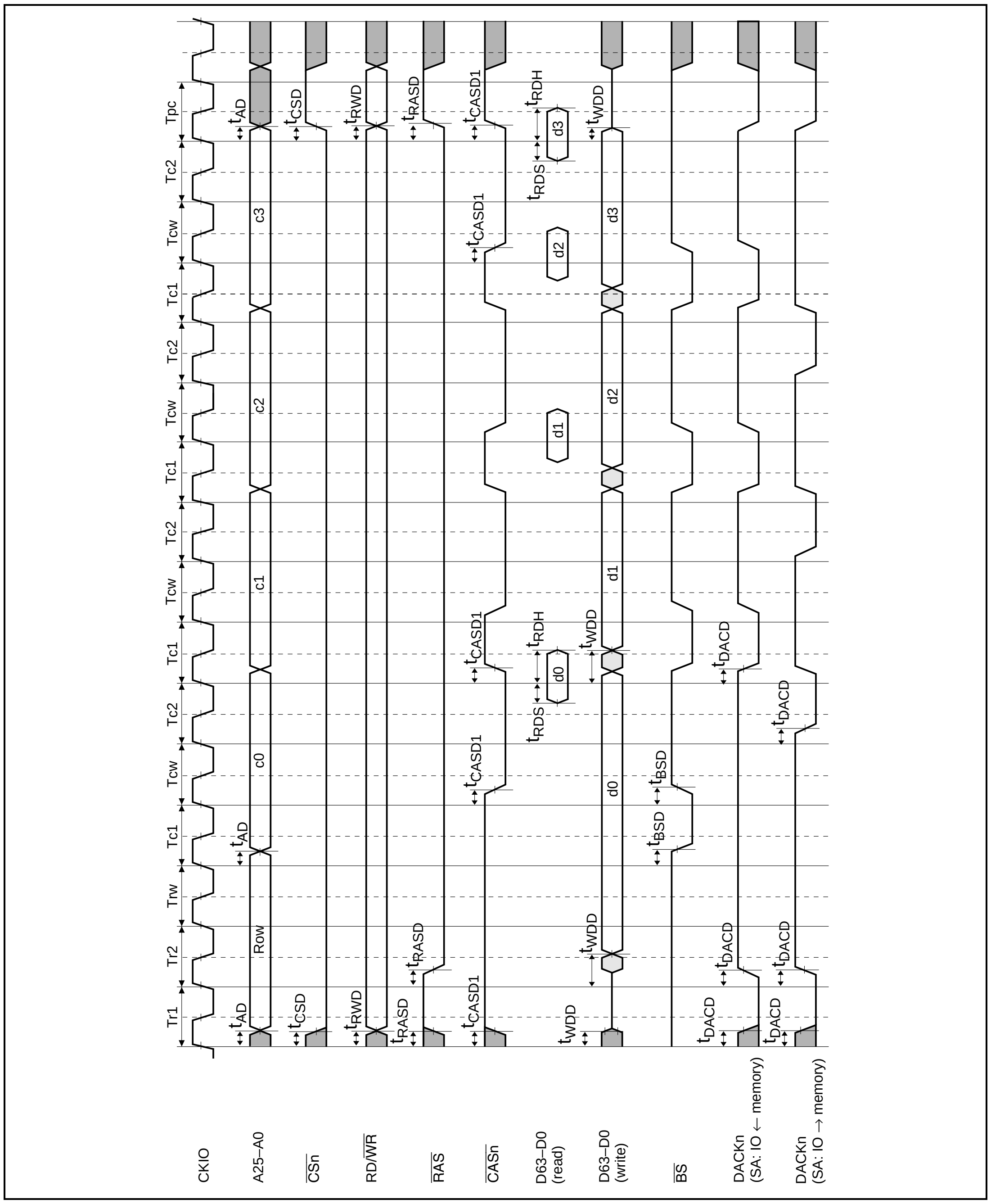


Figure 23.44 DRAM Burst Bus Cycle (Fast Page Mode, RCD = 1, AnW = 1, TPC = 1)

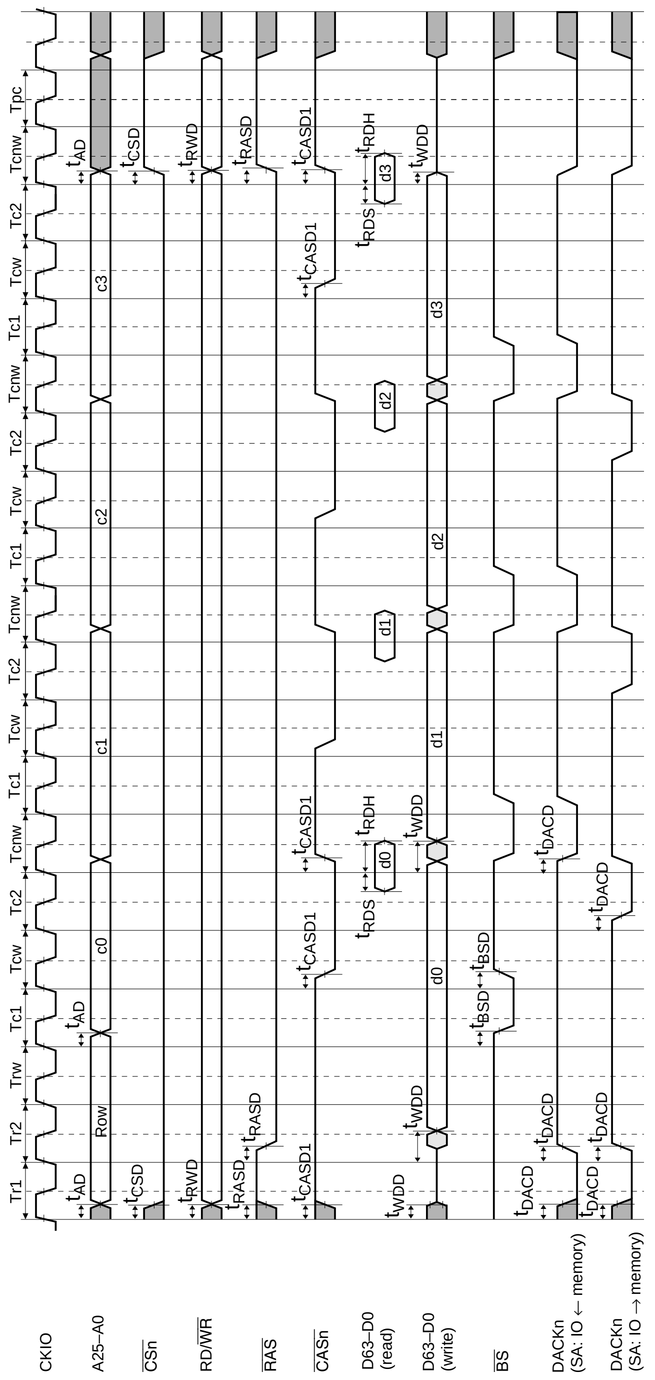


Figure 23.45 DRAM Burst Bus Cycle
(Fast Page Mode, RCD = 1, AnW = 1, TPC = 1, 2-Cycle CAS Negate Pulse Width)

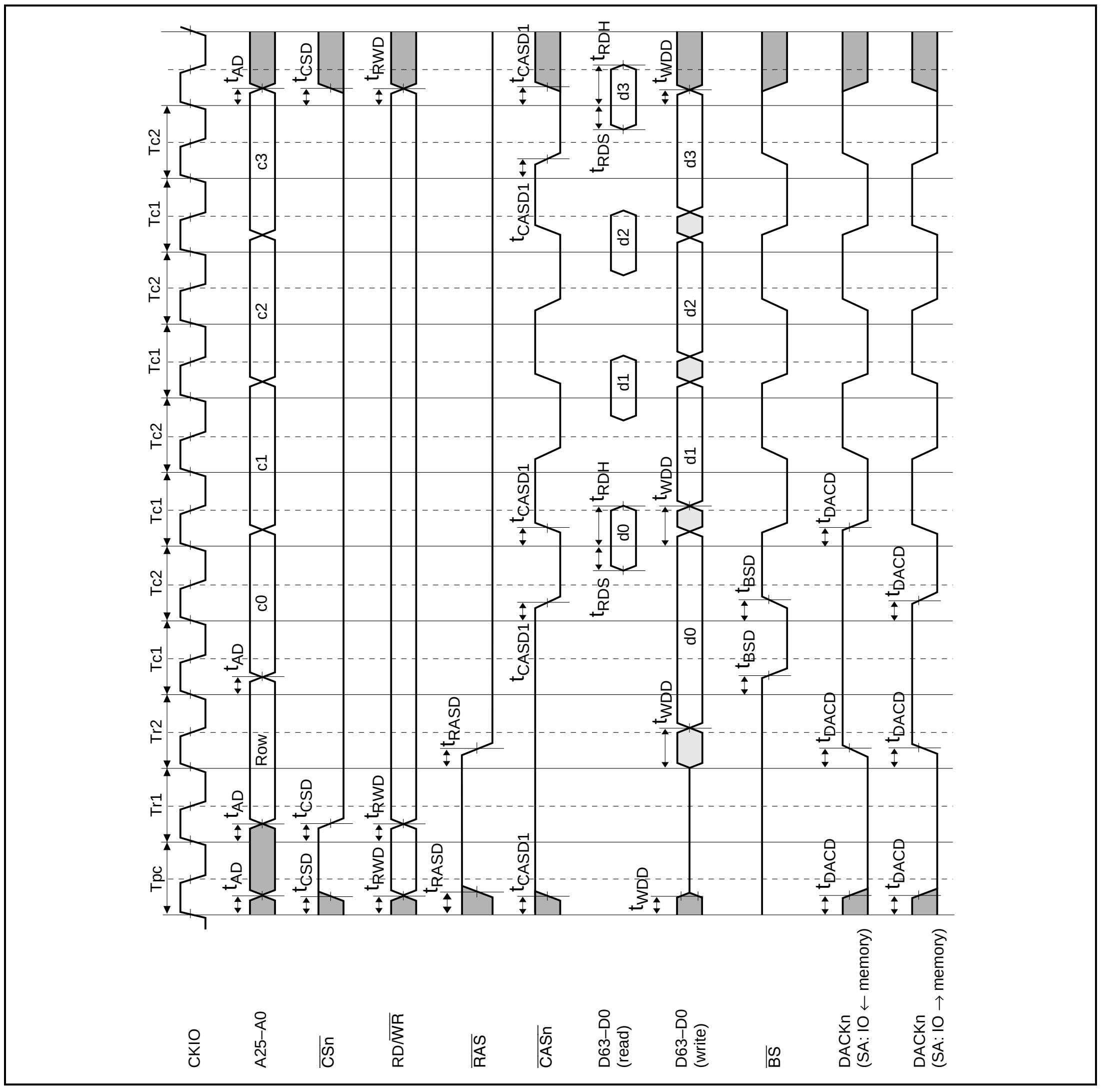


Figure 23.46 DRAM Burst Bus Cycle: RAS Down Mode State (Fast Page Mode, RCD = 0, AnW = 0)

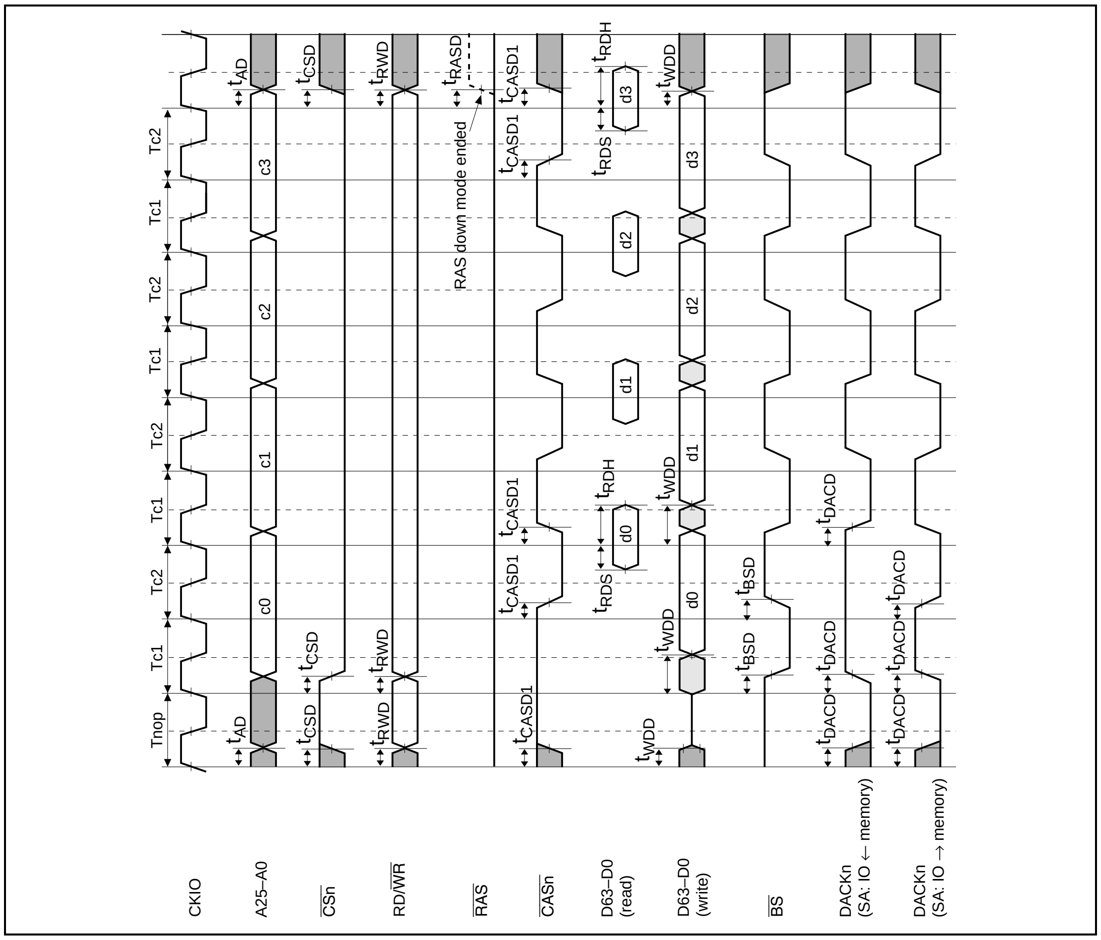


Figure 23.47 DRAM Burst Bus Cycle: RAS Down Mode Continuation (Fast Page Mode, RCD = 0, AnW = 0)

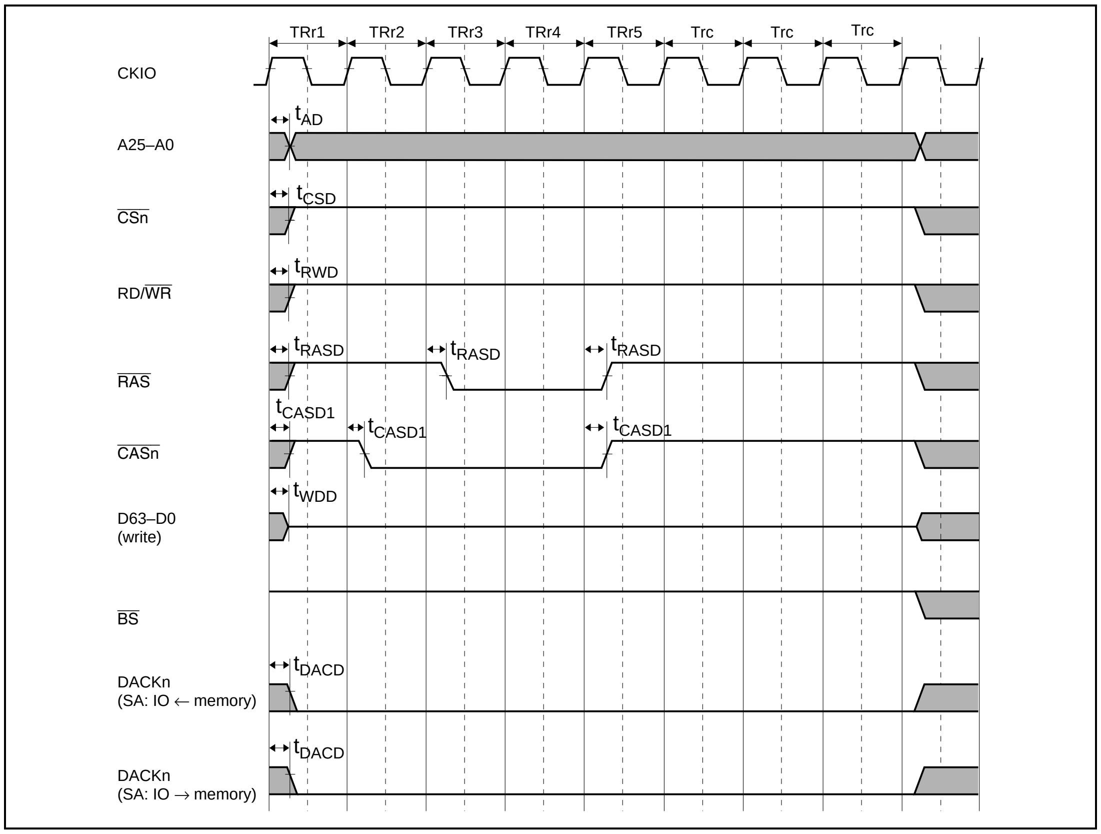


Figure 23.48 DRAM Bus Cycle: DRAM CAS-Before-RAS Refresh (TRAS = 0, TRC = 1)

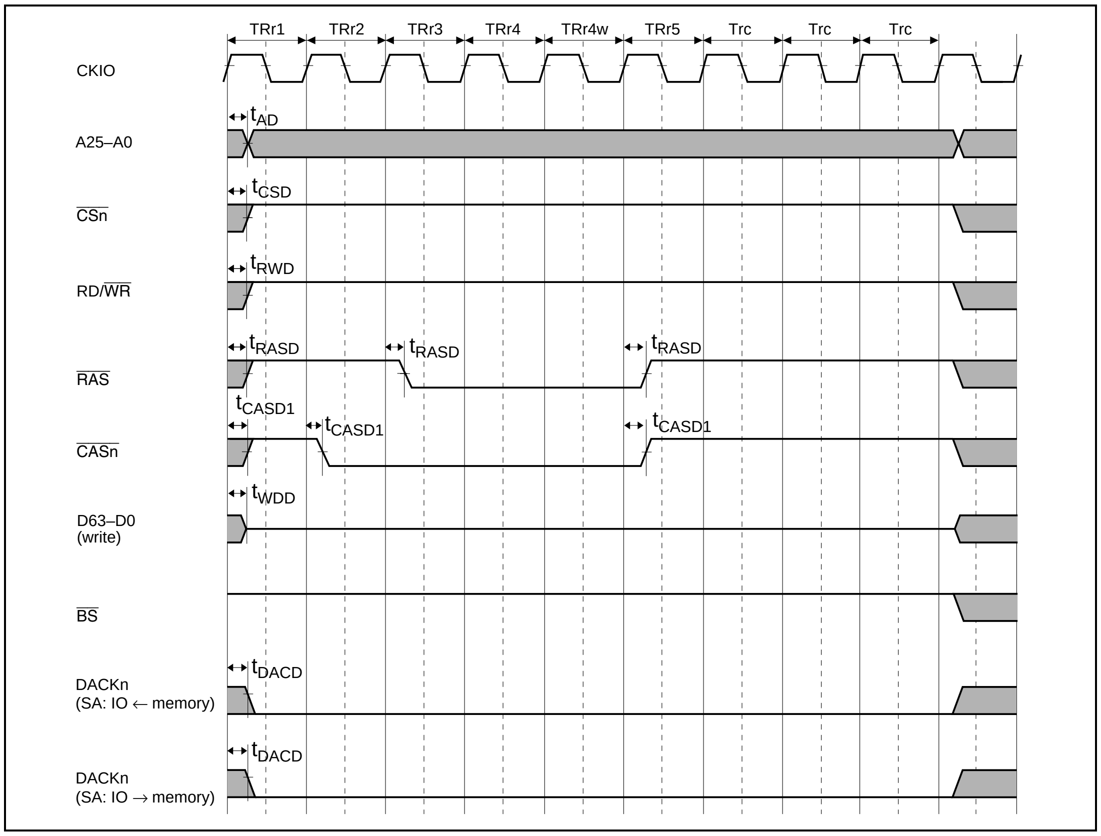


Figure 23.49 DRAM Bus Cycle: DRAM CAS-Before-RAS Refresh (TRAS = 1, TRC = 1)

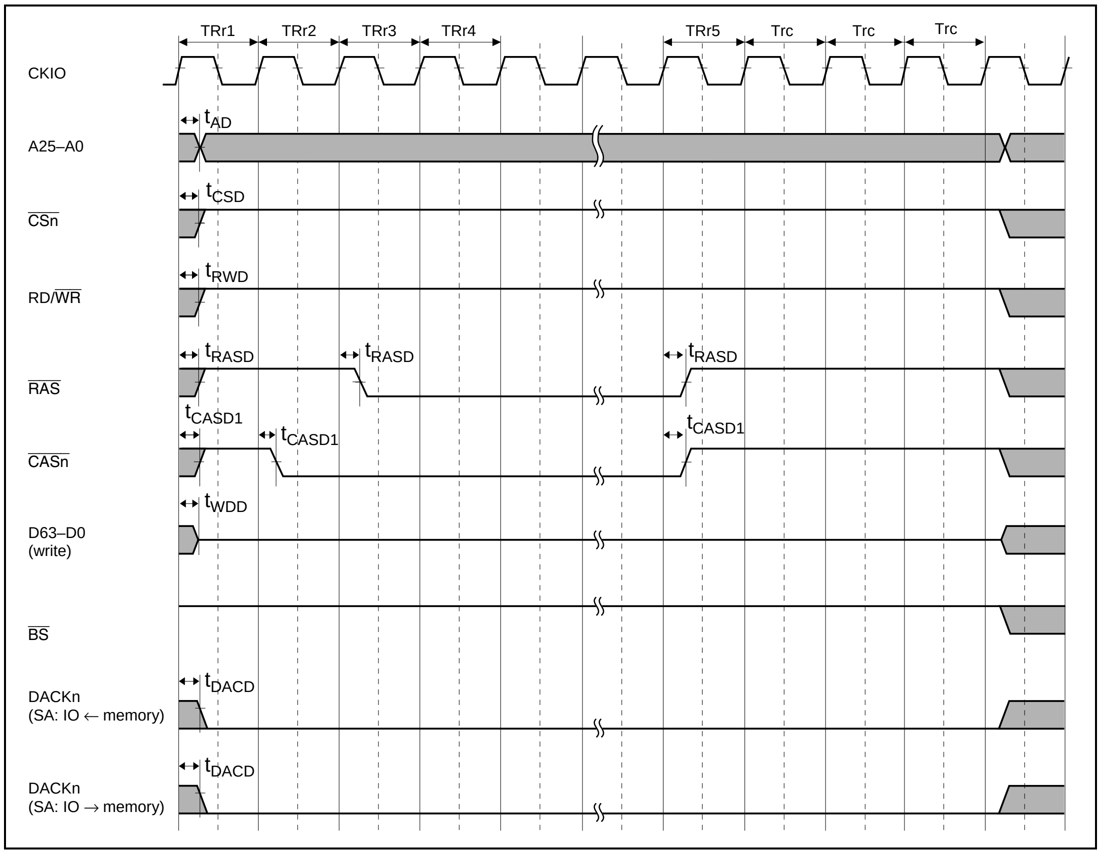


Figure 23.50 DRAM Bus Cycle: DRAM Self-Refresh (TRC = 1)

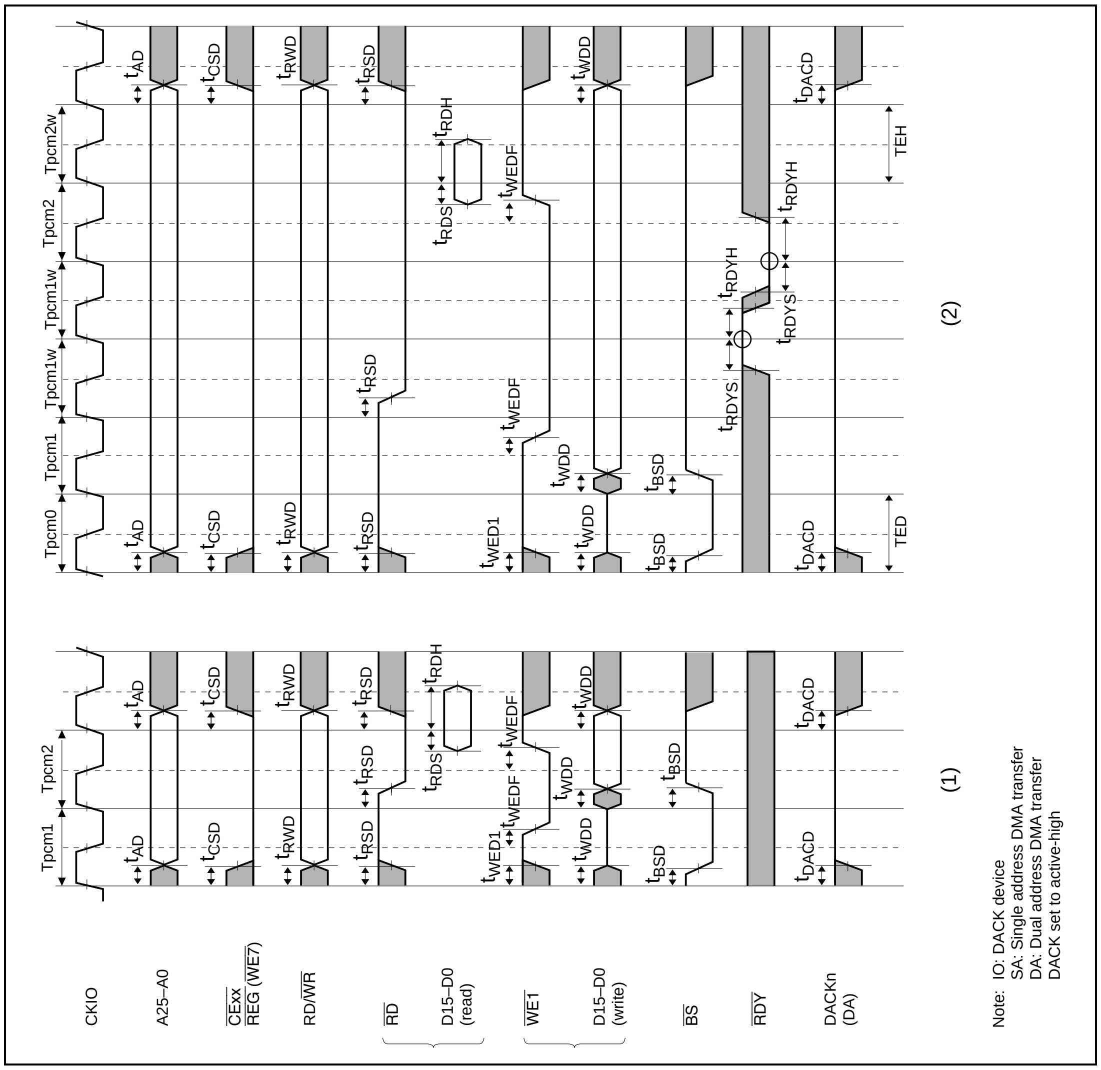


**Figure 23.51 (1) PCMCIA Memory Bus Cycle (TED = 0, TEH = 0, No Wait)
(2) PCMCIA Memory Bus Cycle (TED = 1, TEH = 1,
One Internal Wait + One External Wait)**

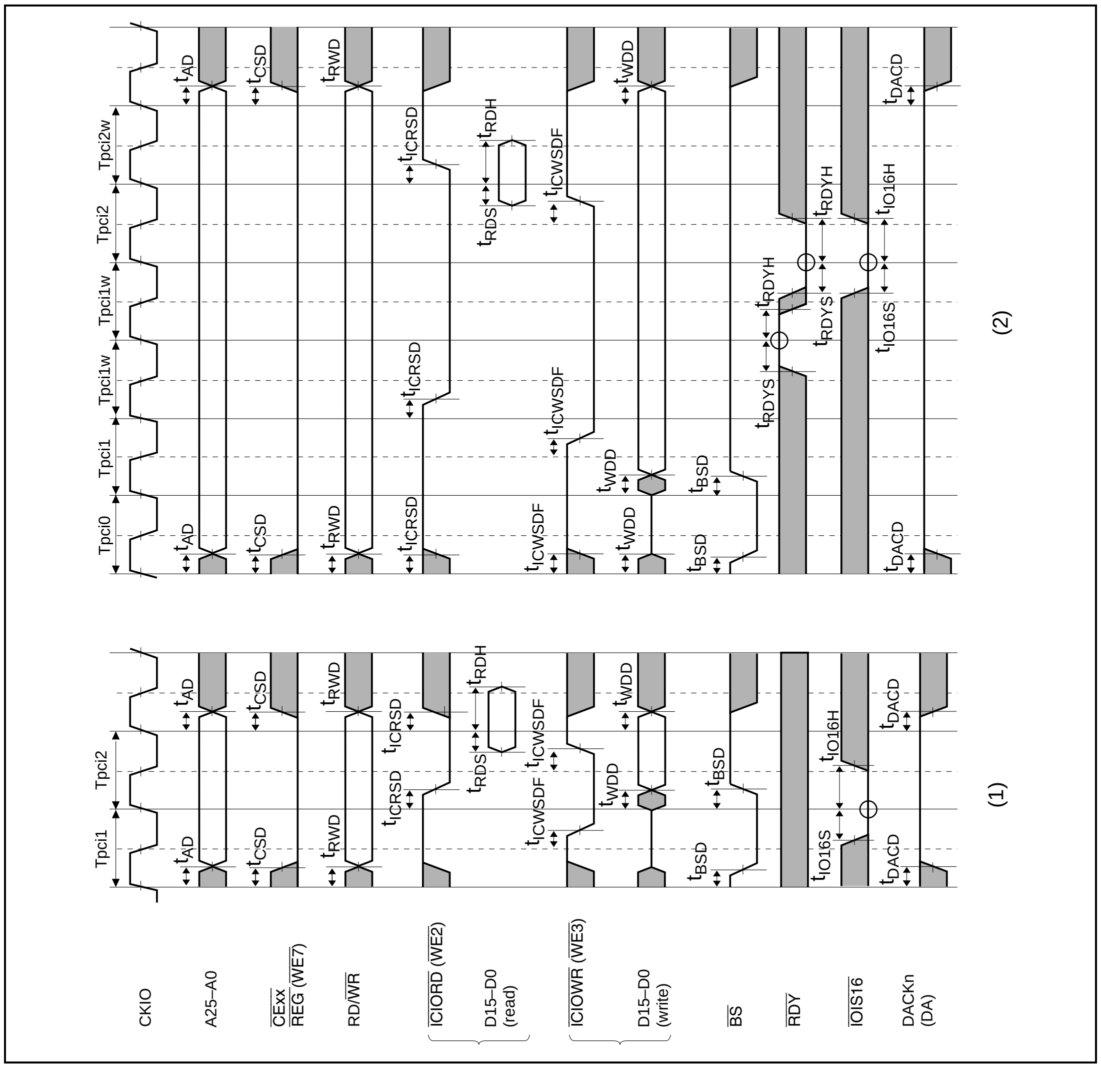


Figure 23.52 (1) PCMCIA I/O Bus Cycle (TED = 0, TEH = 0, No Wait)
(2) PCMCIA I/O Bus Cycle (TED = 1, TEH = 1,
One Internal Wait + One External Wait)

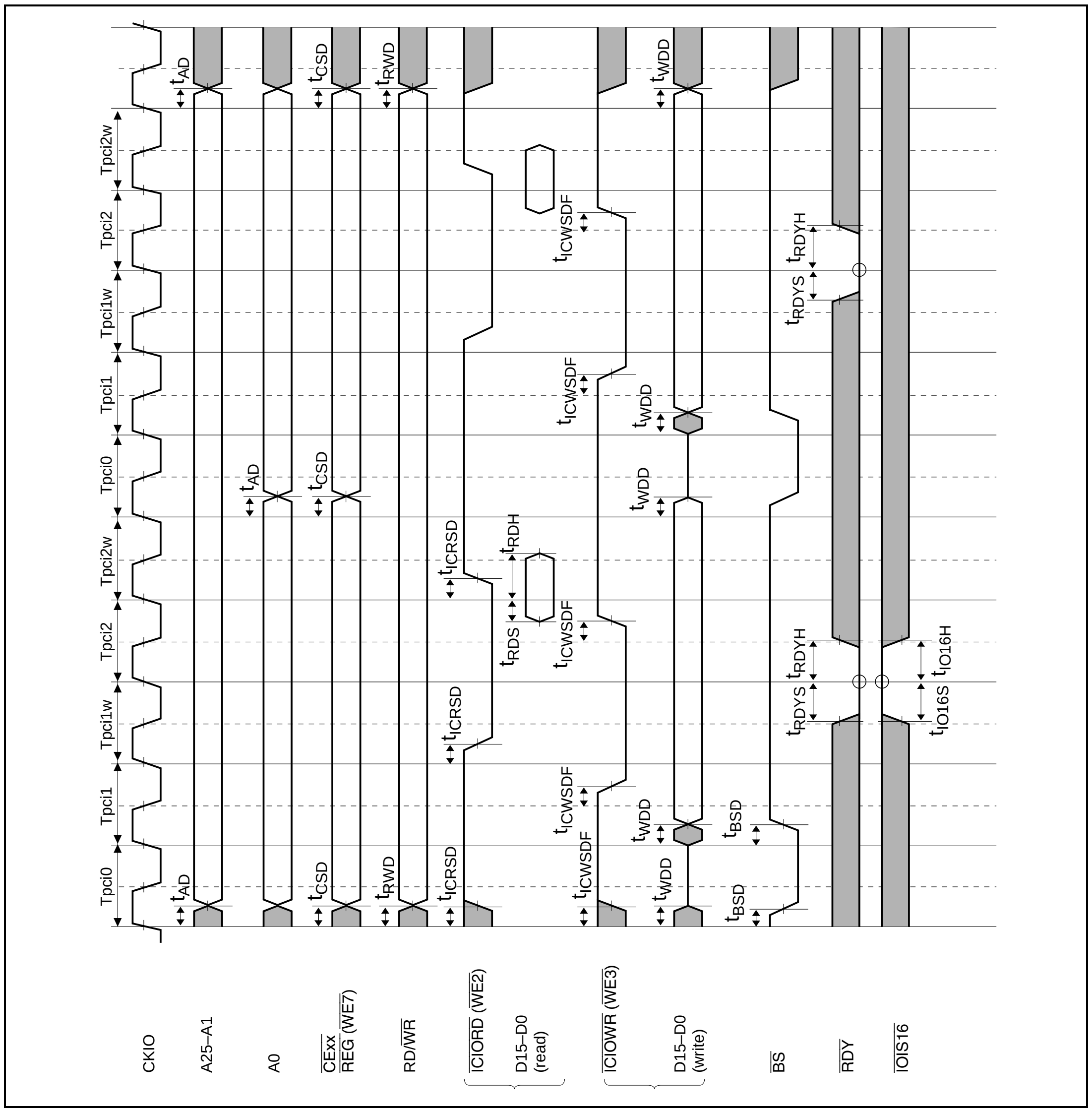


Figure 23.53 PCMCIA I/O Bus Cycle (TED = 1, TEH = 1, One Internal Wait, Bus Sizing)

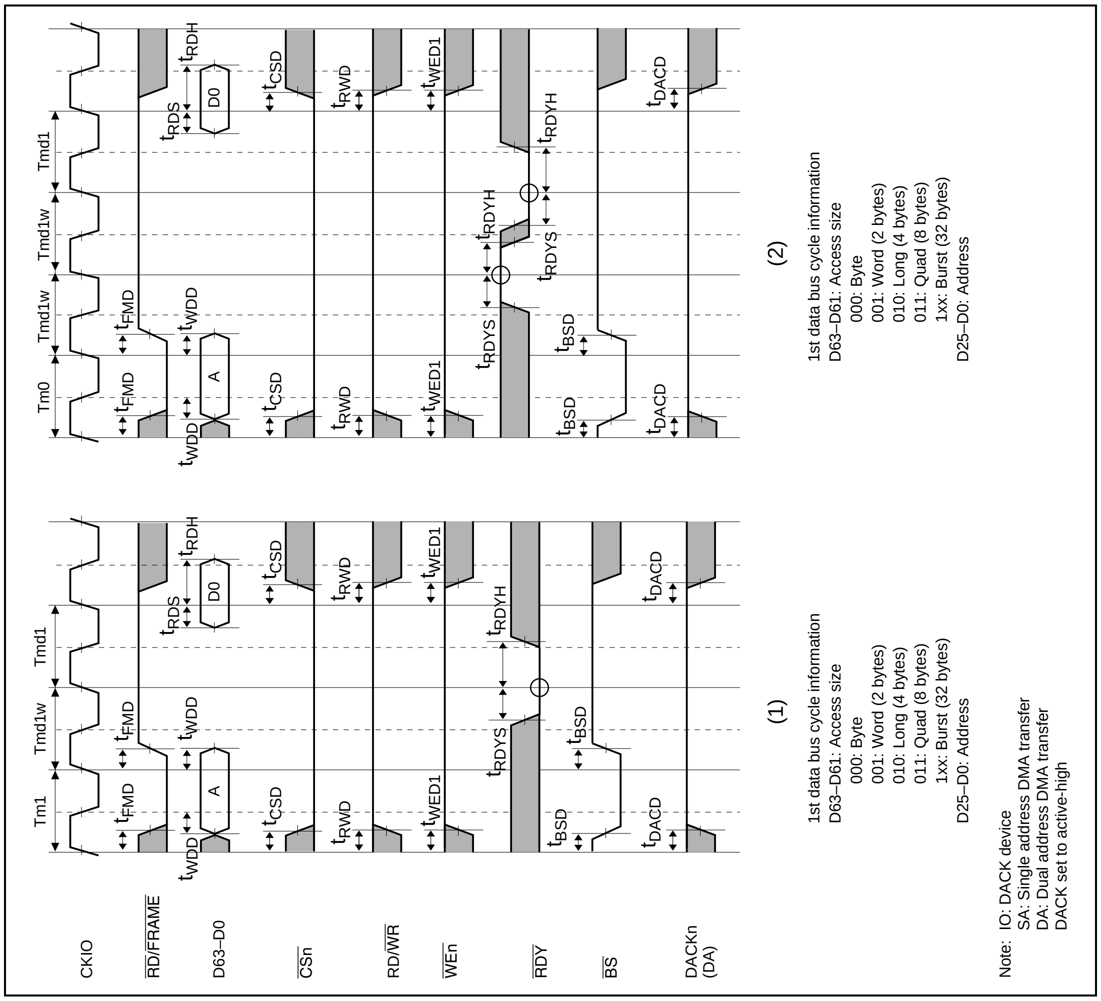


Figure 23.54 (1) MPX Basic Bus Cycle: Read (1st Data: One Internal Wait)
(2) MPX Basic Bus Cycle: Read (1st Data: One Internal Wait + One External Wait)

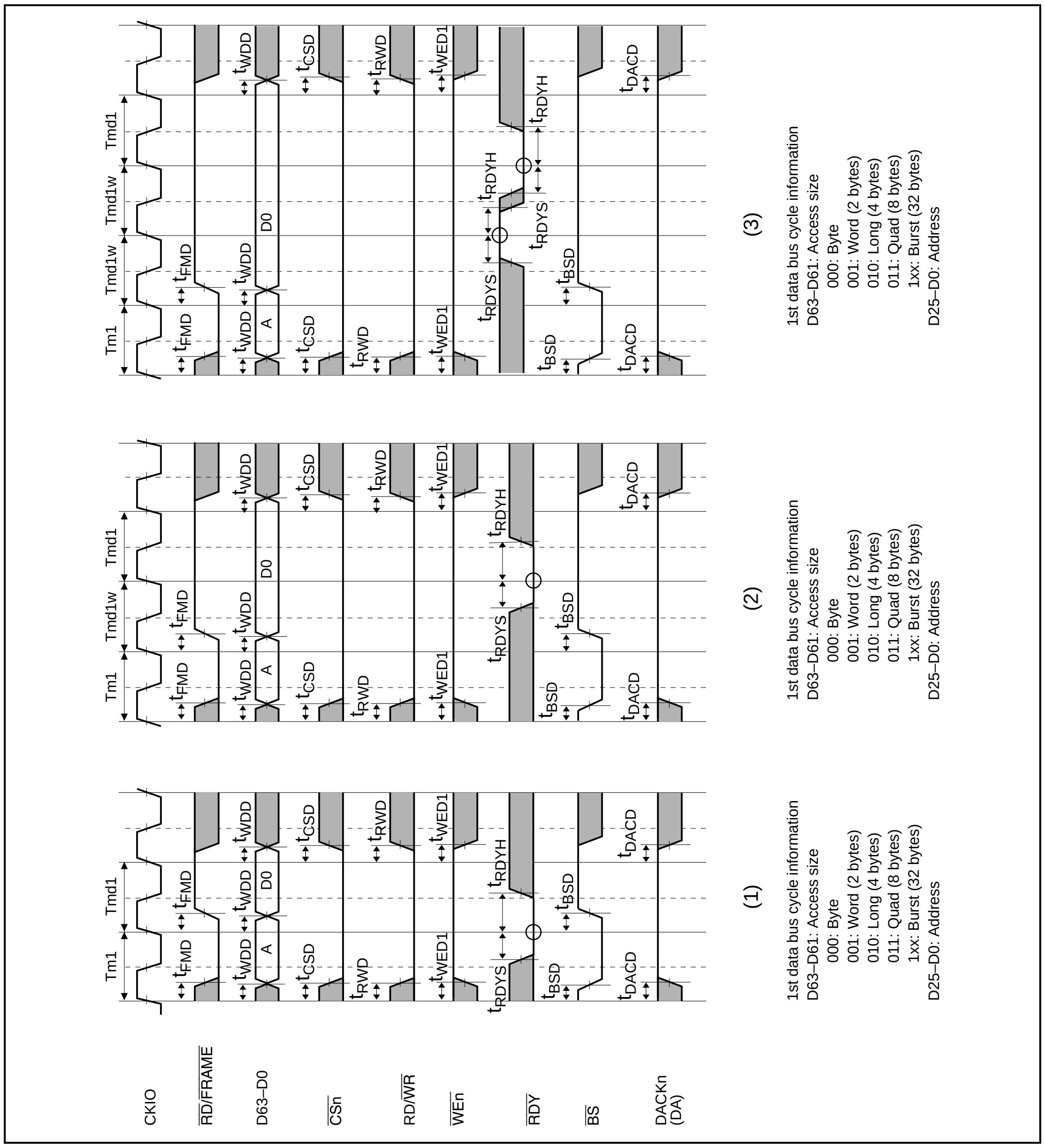


Figure 23.55 (1) MPX Basic Bus Cycle: Write (1st Data: No Wait)
(2) MPX Basic Bus Cycle: Write (1st Data: One Internal Wait)
(3) MPX Basic Bus Cycle: Write (1st Data: One Internal Wait + One External Wait)

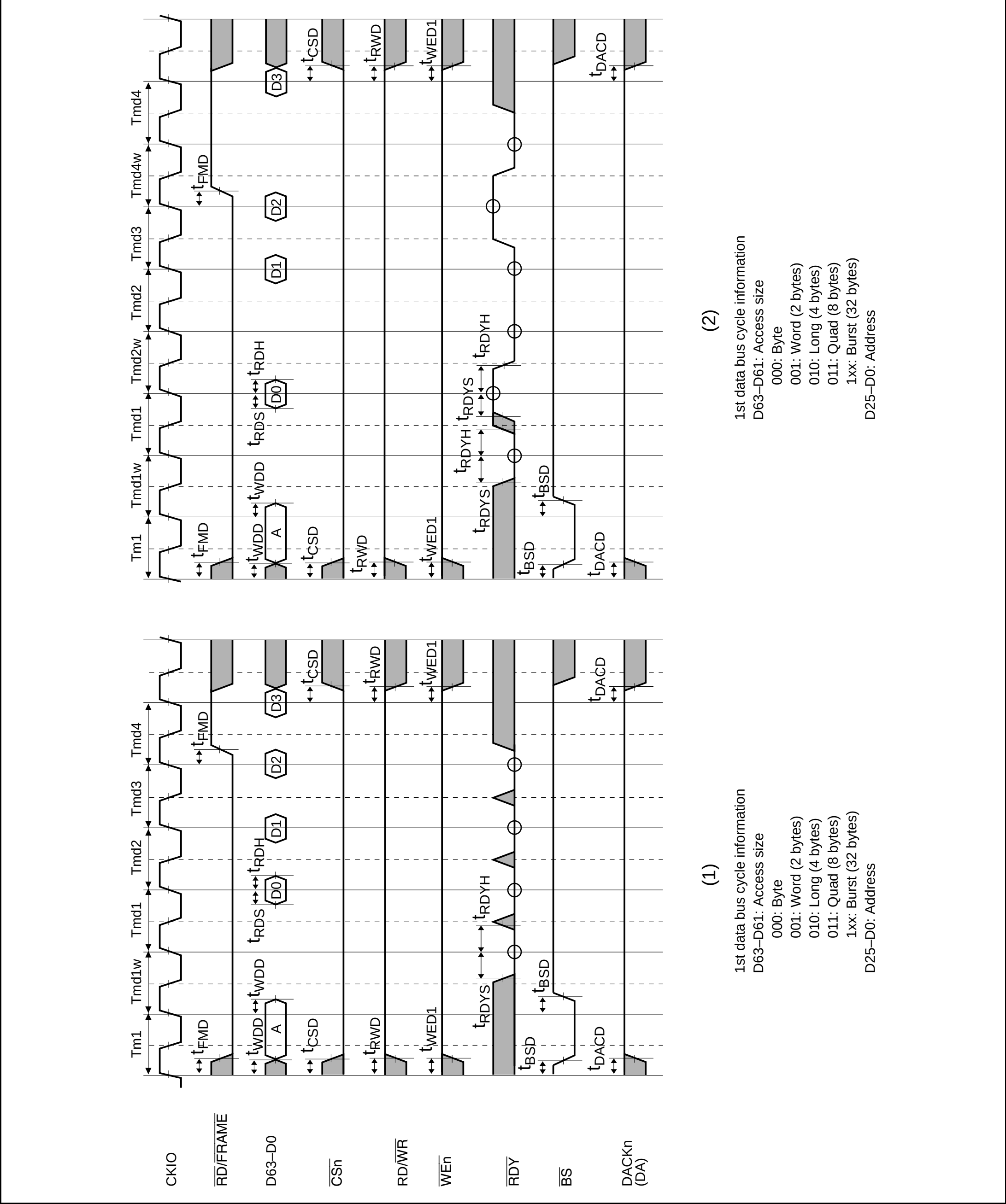


**Figure 23.56 (1) MPX Bus Cycle: Burst Read (1st Data: One Internal Wait; 2nd/3rd/4th Data: No Internal Wait)
(2) MPX Bus Cycle: Burst Read (1st Data: No Internal Wait; 2nd/3rd/4th Data: External Wait Control)**

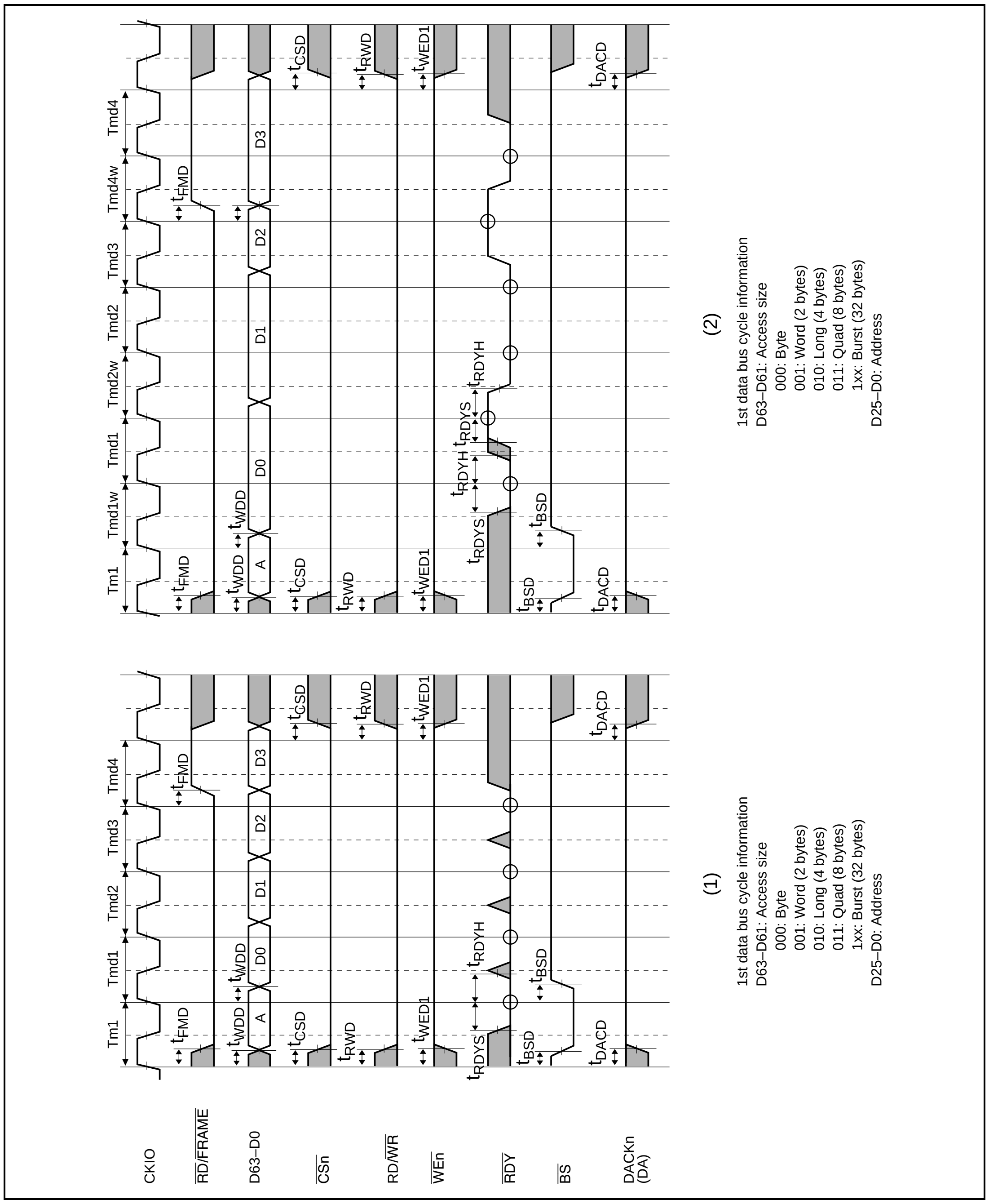
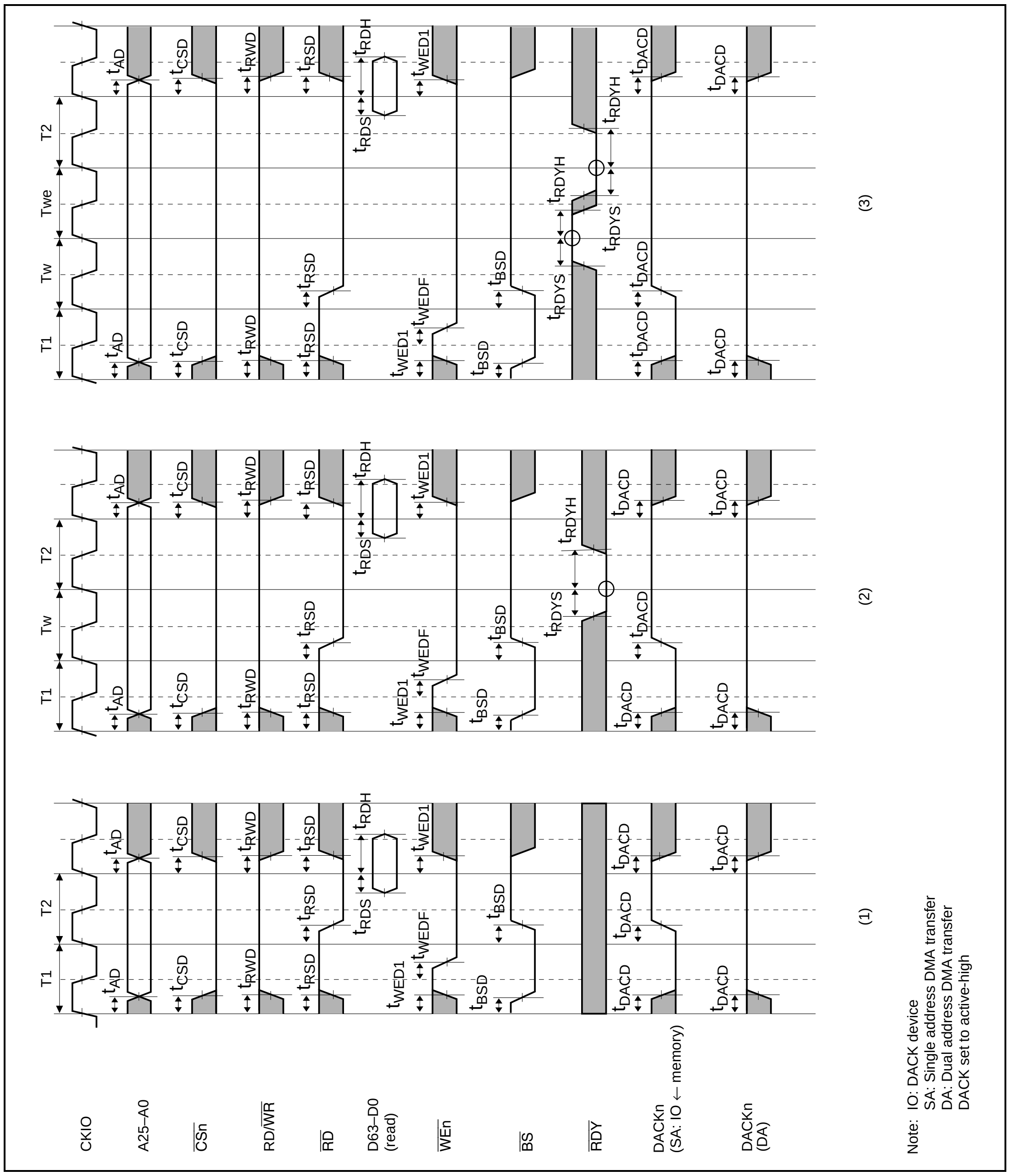


Figure 23.57 (1) MPX Bus Cycle: Burst Write (1st Data: One Internal Wait; 2nd/3rd/4th Data: No Internal Wait)
(2) MPX Bus Cycle: Burst Write (1st Data: One Internal Wait; 2nd/3rd/4th Data: No Internal Wait + External Wait Control)



Note: IO: DACK device
SA: Single address DMA transfer
DA: Dual address DMA transfer
DACK set to active-high

Figure 23.58 Memory Byte Control SRAM Bus Cycles
(1) Basic Read Cycle (No Wait)
(2) Basic Read Cycle (One Internal Wait)
(3) Basic Read Cycle (One Internal Wait + One External Wait)

HITACHI

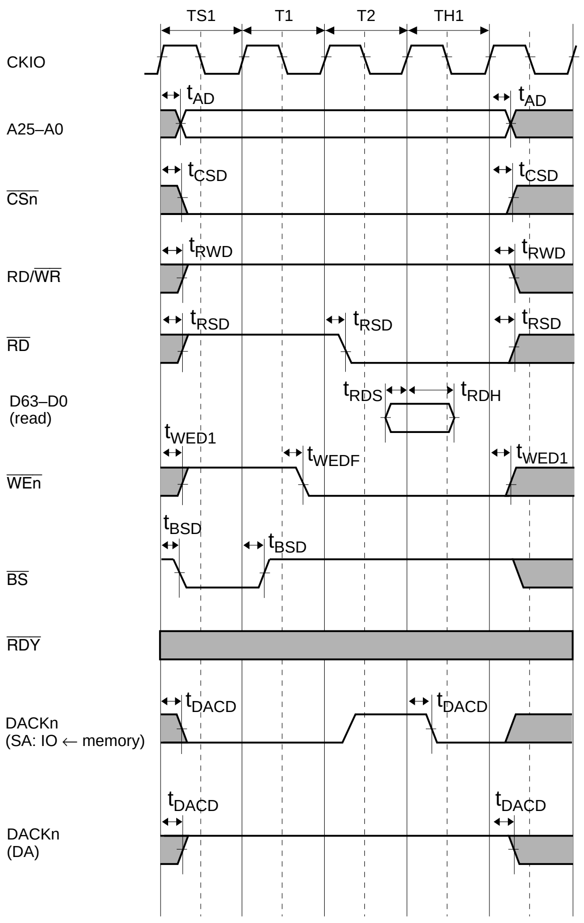


Figure 23.59 Memory Byte Control SRAM Bus Cycle: Basic Read Cycle (No Wait, Address Setup/Hold Time Insertion, AnS = 1, AnH = 1)

23.3.4 Peripheral Module Signal Timing

Table 23.8 Peripheral Module Signal Timing

(V_{DDQ} = 3.0 to 3.6 V, V_{DD} = typ. 1.8 V, T_a = –20 to +75°C, C_L = 30 pF, PLL2 on)

			66 MHz		83 MHz		100 MHz			
Module	**Item**	**Symbol**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Unit**	**Figure**
TMU, RTC	Timer clock pulse width (high)	t_{TCLKWH}	4	—	4	—	4	—	Pcyc*	23.60
	Timer clock pulse width (low)	t_{TCLKWL}	4	—	4	—	4	—	Pcyc*	23.60
	Timer clock rise time	t_{TCLKr}	—	0.8	—	0.8	—	0.8	Pcyc*	23.60
	Timer clock fall time	t_{TCLKf}	—	0.8	—	0.8	—	0.8	Pcyc*	23.60
	Oscillation settling time	t_{ROSC}	—	3	—	3	—	3	s	23.61
SCI	Input clock cycle (asynchronous)	t_{Scyc}	4	—	4	—	4	—	Pcyc*	23.62
	Input clock cycle (synchronous)	t_{Scyc}	6	—	6	—	6	—	Pcyc*	23.62
	Input clock pulse width	t_{SCKW}	0.4	0.6	0.4	0.6	0.4	0.6	t_{Scyc}	23.62
	Input clock rise time	t_{SCKr}	—	0.8	—	0.8	—	0.8	Pcyc*	23.62
	Input clock fall time	t_{SCKf}	—	0.8	—	0.8	—	0.8	Pcyc*	23.62
	Transfer data delay time	t_{TXD}	—	30	—	30	—	30	ns	23.63
	Receive data setup time (synchronous)	t_{RXS}	0.8	—	0.8	—	0.8	—	Pcyc*	23.63
	Receive data hold time (synchronous)	t_{RXH}	0.8	—	0.8	—	0.8	—	Pcyc*	23.63

Table 23.8 Peripheral Module Signal Timing (cont)

(V_{DDQ} = 3.0 to 3.6 V, V_{DD} = typ. 1.8 V, T_a = –20 to +75°C, C_L = 30 pF, PLL2 on)

			66 MHz		83 MHz		100 MHz				
Module	**Item**	**Symbol**	**Min**	**Max**	**Min**	**Max**	**Min**	**Max**	**Unit**	**Figure**	**Note**
I/O ports	Output data delay time	t_{PORTD}	—	10	—	8	—	6	ns	23.64	
	Input data setup time	t_{PORTS}	2	—	2	—	2	—	ns	23.64	**BGA**
			3.5	—	3.5	—	—	—	ns		**QFP**
	Input data hold time	t_{PORTH}	1.5	—	1.5	—	1.5	—	ns	23.64	
DMAC	$\overline{\text{DREQn}}$ setup time	t_{DRQS}	2	—	2	—	2	—	ns	23.65	**BGA**
			3.5	—	3.5	—	—	—	ns		**QFP**
	$\overline{\text{DREQn}}$ hold time	t_{DRQH}	1.5	—	1.5	—	1.5	—	ns	23.65	
	DRAKn delay time	t_{DRAKD}	—	10	—	8	—	6	ns	23.65	
Hitachi-UDI	Input clock cycle	t_{TCKcyc}	50	—	50	—	50	—	ns	23.66	
	Input clock pulse width (high)	t_{TCKH}	15	—	15	—	15	—	ns	23.66	
	Input clock pulse width (low)	t_{TCKL}	15	—	15	—	15	—	ns	23.66	
	Input clock rise time	t_{TCKr}	—	10	—	10	—	10	ns	23.66	
	Input clock fall time	t_{TCKf}	—	10	—	10	—	10	ns	23.66	
Hitachi-UDI	$\overline{\text{ASEBRK}}$ setup time	$t_{ASEBRKS}$	10	—	10	—	10	—	t_{cyc}	23.67	
	$\overline{\text{ASEBRK}}$ hold time	$t_{ASEBRKH}$	10	—	10	—	10	—	t_{cyc}	23.67	
	TDI/TMS setup time	t_{TDIS}	15	—	15	—	15	—	ns	23.68	
	TDI/TMS hold time	t_{TDIH}	15	—	15	—	15	—	ns	23.68	
	TDO delay time	t_{TDO}	0	10	0	10	0	10	ns	23.68	
	ASE-PINBRK pulse width	t_{PINBRK}	2	—	2	—	2	—	Pcyc*	23.69	

Note: * Pcyc: P clock cycles

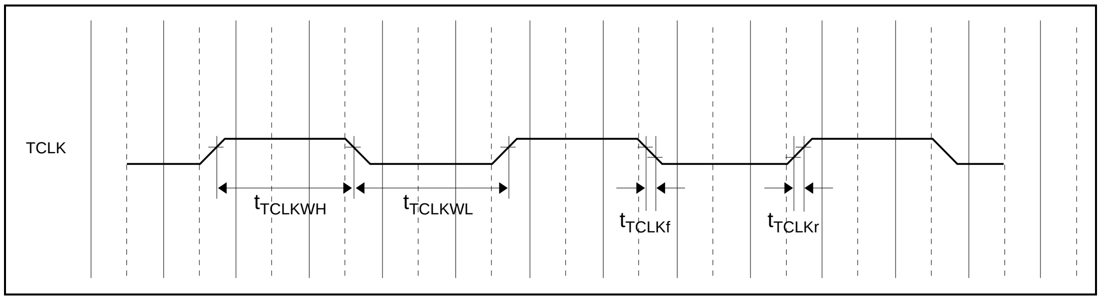


Figure 23.60 TCLK Input Timing

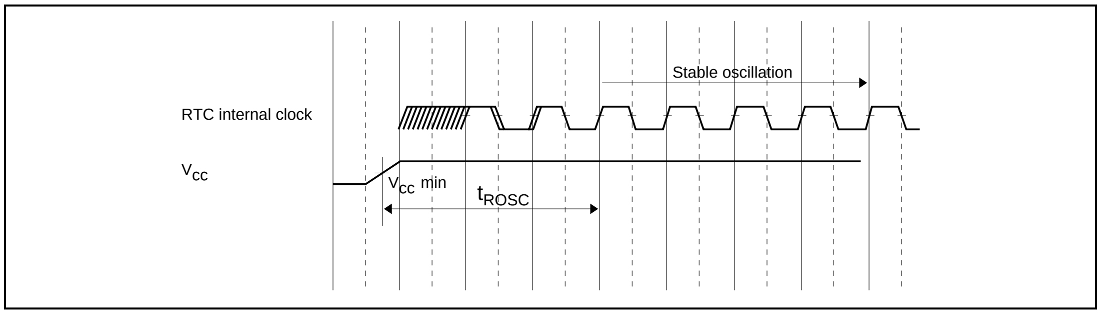


Figure 23.61 RTC Oscillation Settling Time at Power-On

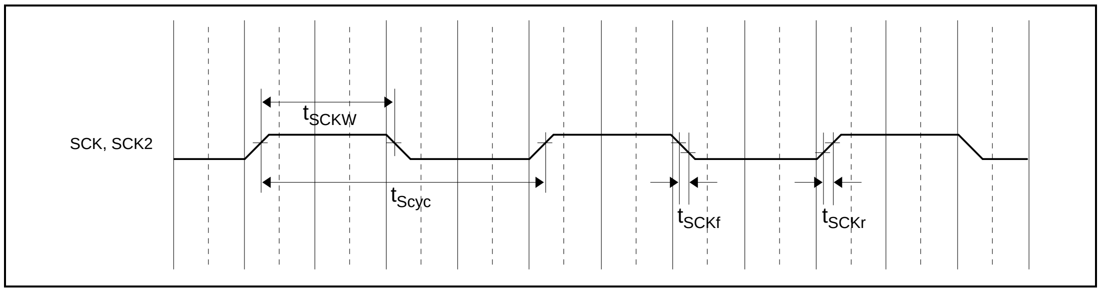


Figure 23.62 SCK Input Clock Timing

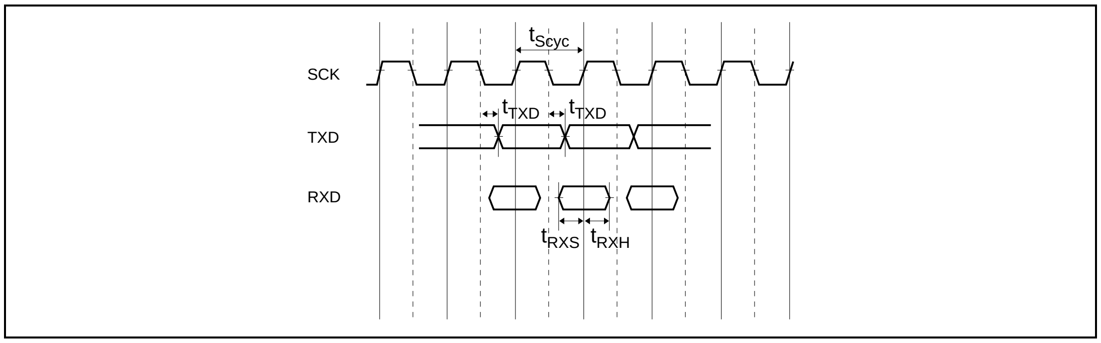


Figure 23.63 SCI I/O Synchronous Mode Clock Timing

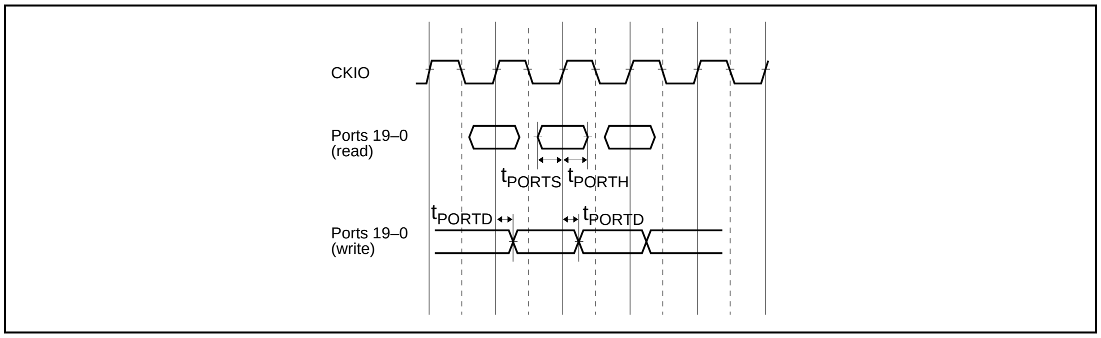


Figure 23.64 I/O Port Input/Output Timing

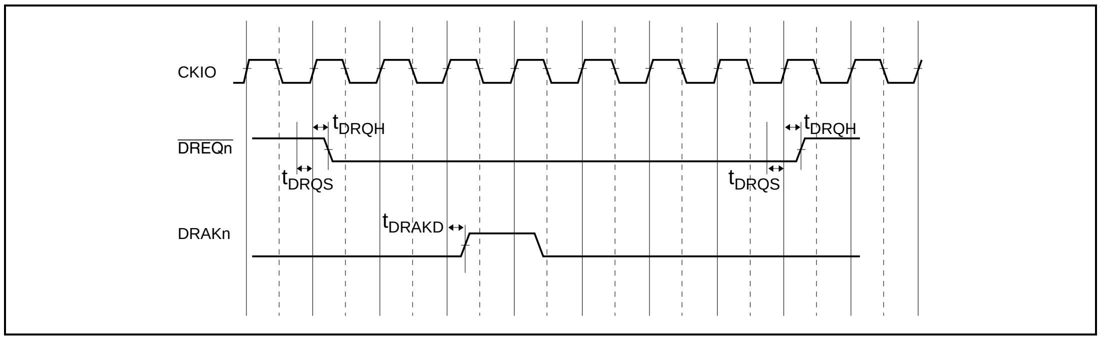


Figure 23.65 $\overline{\text{DREQ}}$/DRAK Timing

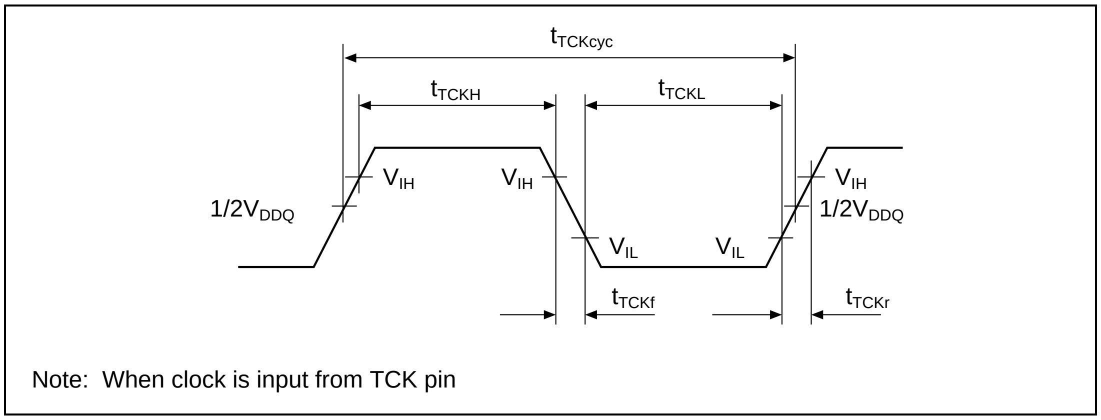


Figure 23.66 TCK Input Timing

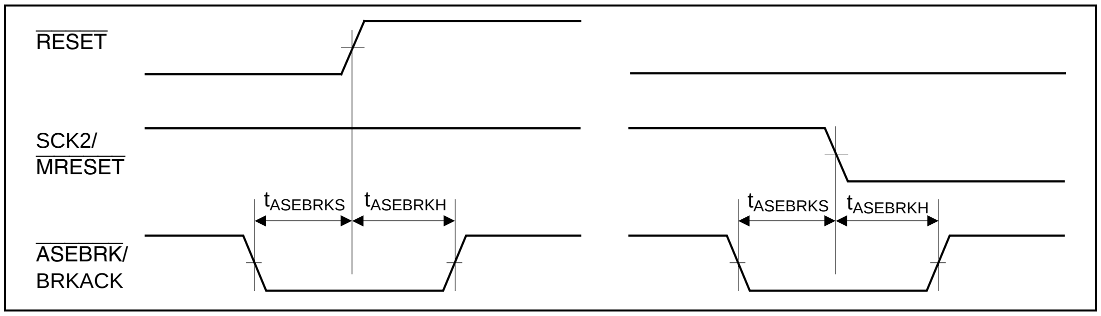

$\overline{\text{RESET}}$
SCK2/
$\overline{\text{MRESET}}$
$\overline{\text{ASEBRK}}$/
BRKACK
$t_{ASEBRKS}$
$t_{ASEBRKH}$
$t_{ASEBRKS}$
$t_{ASEBRKH}$

Figure 23.67 Reset Hold Timing

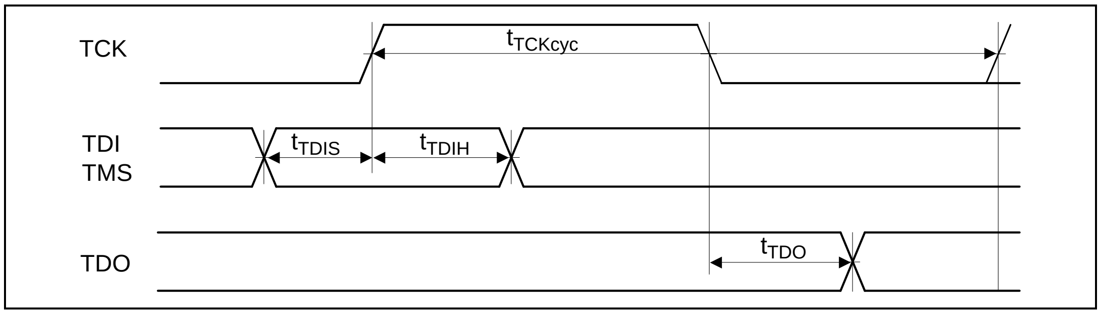


Figure 23.68 Hitachi-UDI Data Transfer Timing

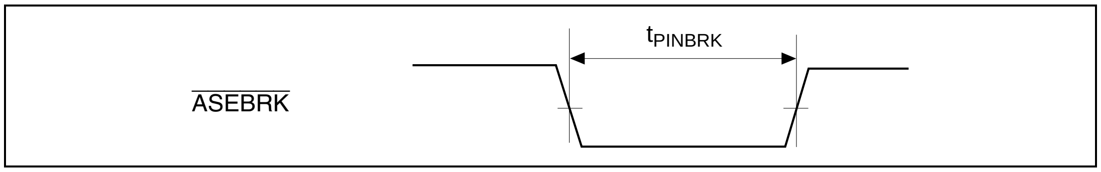


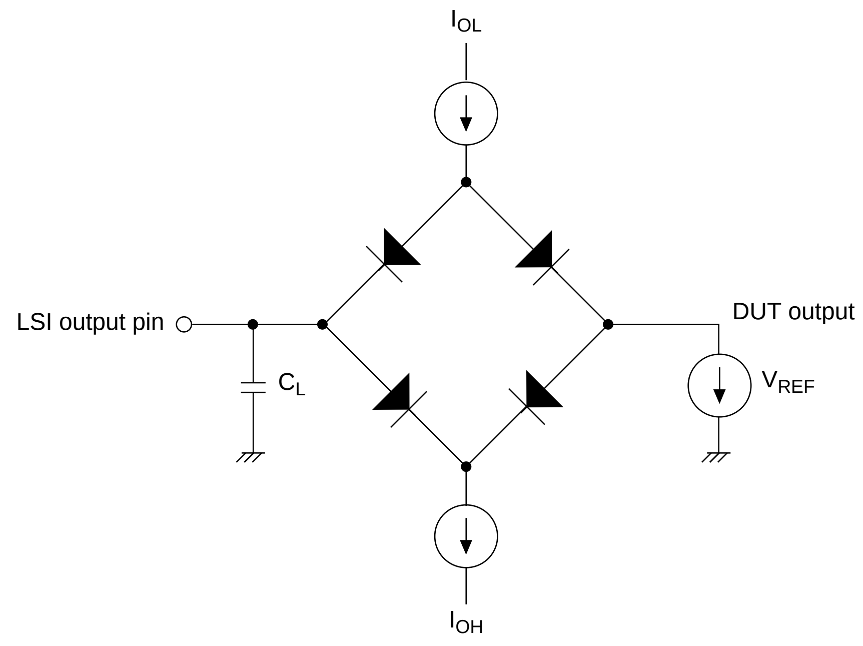
Figure 23.69 Pin Break Timing

23.3.5 AC Characteristic Test Conditions

The AC characteristic test conditions are as follows:

- Input/output signal reference level: 1.5 V (V_{DDQ} = 3.3 ±0.3 V)
- Input pulse level: V_{SSQ}–3.0 V (V_{SSQ}–V_{DDQ} for $\overline{\text{RESET}}$, $\overline{\text{TRST}}$, NMI, and $\overline{\text{ASEBRK}}$/BRKACK)
- Input rise/fall time: 1 ns

The output load circuit is shown in figure 23.70.



Notes: 1. C_L is the total value, including the capacitance of the test jig, etc. The capacitance of each pin is set to 30 pF.
2. I_{OL} and I_{OH} values are as shown in table 23.3, Permissible Output Currents.

Figure 23.70 Output Load Circuit

23.3.6 Delay Time Variation Due to Load Capacitance

A graph (reference data) of the variation in delay time when a load capacitance greater than that stipulated (30 pF) is connected to the SH7750's pins is shown below. The graph shown in figure 23.71 should be taken into consideration if the stipulated capacitance is exceeded when connecting an external device.

The graph will not be linear if the connected load capacitance exceeds the range shown in figure 23.71.

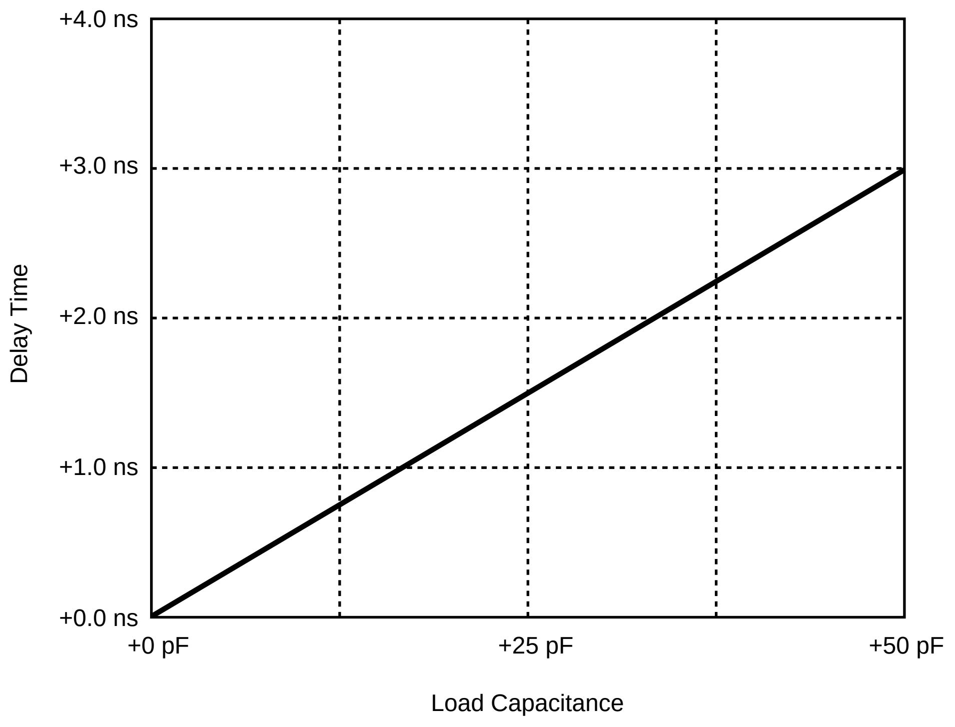


Figure 23.71 Load Capacitance vs. Delay Time

Appendix A   Address List

Table A.1   Address List

Module	Register	P4 Address	Area 7 Address*[1]	Size	Power-On Reset	Manual Reset	Sleep	Standby	Synchro-nization Clock
CCN	PTEH	H'FF00 0000	H'1F00 0000	32	Undefined	Undefined	Held	Held	Iclk
CCN	PTEL	H'FF00 0004	H'1F00 0004	32	Undefined	Undefined	Held	Held	Iclk
CCN	TTB	H'FF00 0008	H'1F00 0008	32	Undefined	Undefined	Held	Held	Iclk
CCN	TEA	H'FF00 000C	H'1F00 000C	32	Undefined	Held	Held	Held	Iclk
CCN	MMUCR	H'FF00 0010	H'1F00 0010	32	H'0000 0000	H'0000 0000	Held	Held	Iclk
CCN	BASRA	H'FF00 0014	H'1F00 0014	8	Undefined	Held	Held	Held	Iclk
CCN	BASRB	H'FF00 0018	H'1F00 0018	8	Undefined	Held	Held	Held	Iclk
CCN	CCR	H'FF00 001C	H'1F00 001C	32	H'0000 0000	H'0000 0000	Held	Held	Iclk
CCN	TRA	H'FF00 0020	H'1F00 0020	32	Undefined	Undefined	Held	Held	Iclk
CCN	EXPEVT	H'FF00 0024	H'1F00 0024	32	H'0000 0000	H'0000 0020	Held	Held	Iclk
CCN	INTEVT	H'FF00 0028	H'1F00 0028	32	Undefined	Undefined	Held	Held	Iclk
CCN	PTEA	H'FF00 0034	H'1F00 0034	32	Undefined	Undefined	Held	Held	Iclk
CCN	QACR0	H'FF00 0038	H'1F00 0038	32	Undefined	Undefined	Held	Held	Iclk
CCN	QACR1	H'FF00 003C	H'1F00 003C	32	Undefined	Undefined	Held	Held	Iclk
UBC	BARA	H'FF20 0000	H'1F20 0000	32	Undefined	Held	Held	Held	Iclk
UBC	BAMRA	H'FF20 0004	H'1F20 0004	8	Undefined	Held	Held	Held	Iclk
UBC	BBRA	H'FF20 0008	H'1F20 0008	16	H'0000	Held	Held	Held	Iclk
UBC	BARB	H'FF20 000C	H'1F20 000C	32	Undefined	Held	Held	Held	Iclk
UBC	BAMRB	H'FF20 0010	H'1F20 0010	8	Undefined	Held	Held	Held	Iclk
UBC	BBRB	H'FF20 0014	H'1F20 0014	16	H'0000	Held	Held	Held	Iclk
UBC	BDRB	H'FF20 0018	H'1F20 0018	32	Undefined	Held	Held	Held	Iclk
UBC	BDMRB	H'FF20 001C	H'1F20 001C	32	Undefined	Held	Held	Held	Iclk
UBC	BRCR	H'FF20 0020	H'1F20 0020	16	H'0000*[2]	Held	Held	Held	Iclk
BSC	BCR1	H'FF80 0000	H'1F80 0000	32	H'0000 0000*[2]	Held	Held	Held	Bclk
BSC	BCR2	H'FF80 0004	H'1F80 0004	16	H'3FFC*[2]	Held	Held	Held	Bclk
BSC	WCR1	H'FF80 0008	H'1F80 0008	32	H'7777 7777	Held	Held	Held	Bclk
BSC	WCR2	H'FF80 000C	H'1F80 000C	32	H'FFFE EFFF	Held	Held	Held	Bclk
BSC	WCR3	H'FF80 0010	H'1F80 0010	32	H'0777 7777	Held	Held	Held	Bclk

Table A.1 Address List (cont)

Module	Register	P4 Address	Area 7 Address*[1]	Size	Power-On Reset	Manual Reset	Sleep	Standby	Synchro-nization Clock
BSC	MCR	H'FF80 0014	H'1F80 0014	32	H'0000 0000	Held	Held	Held	Bclk
BSC	PCR	H'FF80 0018	H'1F80 0018	16	H'0000	Held	Held	Held	Bclk
BSC	RTCSR	H'FF80 001C	H'1F80 001C	16	H'0000	Held	Held	Held	Bclk
BSC	RTCNT	H'FF80 0020	H'1F80 0020	16	H'0000	Held	Held	Held	Bclk
BSC	RTCOR	H'FF80 0024	H'1F80 0024	16	H'0000	Held	Held	Held	Bclk
BSC	RFCR	H'FF80 0028	H'1F80 0028	16	H'0000	Held	Held	Held	Bclk
BSC	PCTRA	H'FF80 002C	H'1F80 002C	32	H'0000 0000	Held	Held	Held	Bclk
BSC	PDTRA	H'FF80 0030	H'1F80 0030	16	Undefined	Held	Held	Held	Bclk
BSC	PCTRB	H'FF80 0040	H'1F80 0040	32	H'0000 0000	Held	Held	Held	Bclk
BSC	PDTRB	H'FF80 0044	H'1F80 0044	16	Undefined	Held	Held	Held	Bclk
BSC	GPIOIC	H'FF80 0048	H'1F80 0048	16	H'0000 0000	Held	Held	Held	Bclk
BSC	SDMR2	H'FF90 xxxx	H'1F90 xxxx	8	Write-only				Bclk
BSC	SDMR3	H'FF94 xxxx	H'1F94 xxxx	8					Bclk
DMAC	SAR0	H'FFA0 0000	H'1FA0 0000	32	Undefined	Undefined	Held	Held	Bclk
DMAC	DAR0	H'FFA0 0004	H'1FA0 0004	32	Undefined	Undefined	Held	Held	Bclk
DMAC	DMATCR0	H'FFA0 0008	H'1FA0 0008	32	Undefined	Undefined	Held	Held	Bclk
DMAC	CHCR0	H'FFA0 000C	H'1FA0 000C	32	H'0000 0000	H'0000 0000	Held	Held	Bclk
DMAC	SAR1	H'FFA0 0010	H'1FA0 0010	32	Undefined	Undefined	Held	Held	Bclk
DMAC	DAR1	H'FFA0 0014	H'1FA0 0014	32	Undefined	Undefined	Held	Held	Bclk
DMAC	DMATCR1	H'FFA0 0018	H'1FA0 0018	32	Undefined	Undefined	Held	Held	Bclk
DMAC	CHCR1	H'FFA0 001C	H'1FA0 001C	32	H'0000 0000	H'0000 0000	Held	Held	Bclk
DMAC	SAR2	H'FFA0 0020	H'1FA0 0020	32	Undefined	Undefined	Held	Held	Bclk
DMAC	DAR2	H'FFA0 0024	H'1FA0 0024	32	Undefined	Undefined	Held	Held	Bclk
DMAC	DMATCR2	H'FFA0 0028	H'1FA0 0028	32	Undefined	Undefined	Held	Held	Bclk
DMAC	CHCR2	H'FFA0 002C	H'1FA0 002C	32	H'0000 0000	H'0000 0000	Held	Held	Bclk
DMAC	SAR3	H'FFA0 0030	H'1FA0 0030	32	Undefined	Undefined	Held	Held	Bclk
DMAC	DAR3	H'FFA0 0034	H'1FA0 0034	32	Undefined	Undefined	Held	Held	Bclk
DMAC	DMATCR3	H'FFA0 0038	H'1FA0 0038	32	Undefined	Undefined	Held	Held	Bclk
DMAC	CHCR3	H'FFA0 003C	H'1FA0 003C	32	H'0000 0000	H'0000 0000	Held	Held	Bclk
DMAC	DMAOR	H'FFA0 0040	H'1FA0 0040	32	H'0000 0000	H'0000 0000	Held	Held	Bclk

Table A.1 Address List (cont)

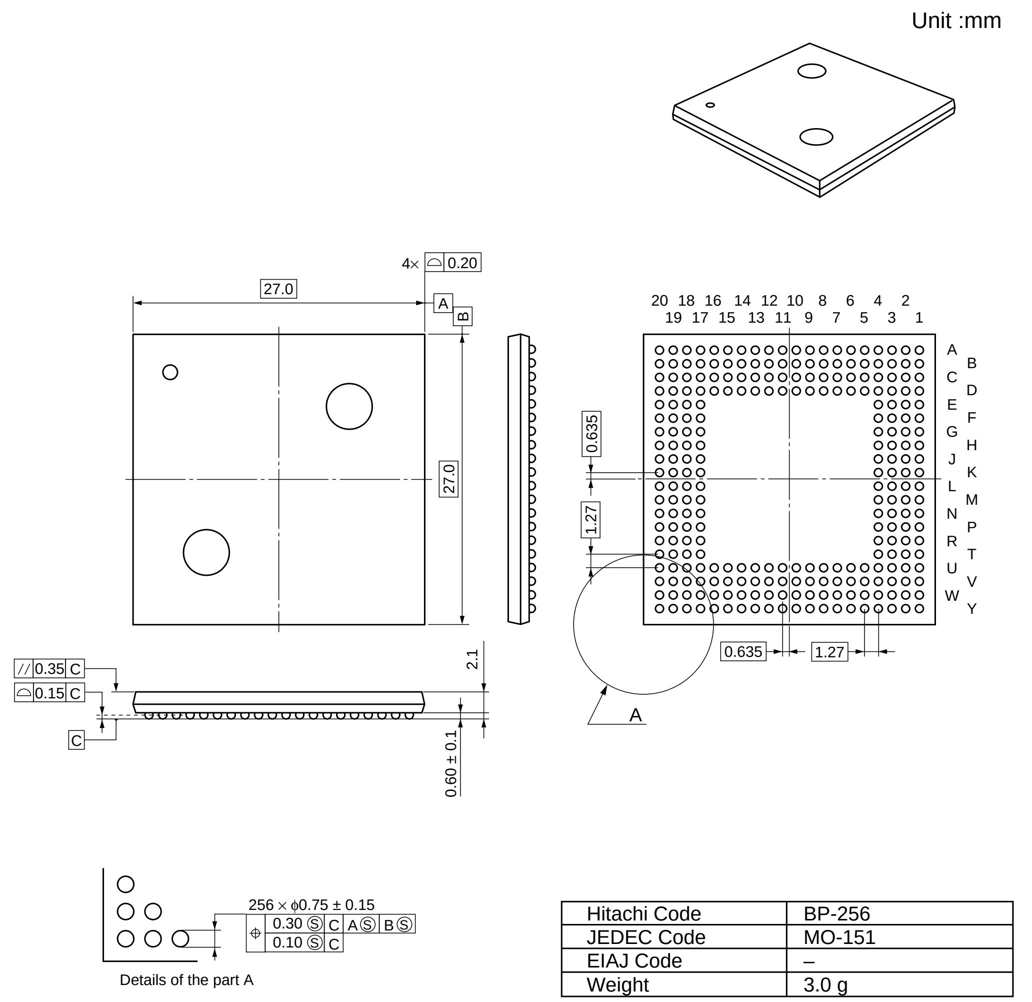
Module	Register	P4 Address	Area 7 Address*[1]	Size	Power-On Reset	Manual Reset	Sleep	Standby	Synchro-nization Clock
CPG	FRQCR	H'FFC0 0000	H'1FC0 0000	16	*[2]	Held	Held	Held	Pclk
CPG	STBCR	H'FFC0 0004	H'1FC0 0004	8	H'00	Held	Held	Held	Pclk
CPG	WTCNT	H'FFC0 0008	H'1FC0 0008	8/16*[3]	H'00	Held	Held	Held	Pclk
CPG	WTCSR	H'FFC0 000C	H'1FC0 000C	8/16*[3]	H'00	Held	Held	Held	Pclk
CPG	STBCR2	H'FFC0 0010	H'1FC0 0010	8	H'00	Held	Held	Held	Pclk
RTC	R64CNT	H'FFC8 0000	H'1FC8 0000	8	Held	Held	Held	Held	Pclk
RTC	RSECCNT	H'FFC8 0004	H'1FC8 0004	8	Held	Held	Held	Held	Pclk
RTC	RMINCNT	H'FFC8 0008	H'1FC8 0008	8	Held	Held	Held	Held	Pclk
RTC	RHRCNT	H'FFC8 000C	H'1FC8 000C	8	Held	Held	Held	Held	Pclk
RTC	RWKCNT	H'FFC8 0010	H'1FC8 0010	8	Held	Held	Held	Held	Pclk
RTC	RDAYCNT	H'FFC8 0014	H'1FC8 0014	8	Held	Held	Held	Held	Pclk
RTC	RMONCNT	H'FFC8 0018	H'1FC8 0018	8	Held	Held	Held	Held	Pclk
RTC	RYRCNT	H'FFC8 001C	H'1FC8 001C	16	Held	Held	Held	Held	Pclk
RTC	RSECAR	H'FFC8 0020	H'1FC8 0020	8	Held *[2]	Held	Held	Held	Pclk
RTC	RMINAR	H'FFC8 0024	H'1FC8 0024	8	Held *[2]	Held	Held	Held	Pclk
RTC	RHRAR	H'FFC8 0028	H'1FC8 0028	8	Held *[2]	Held	Held	Held	Pclk
RTC	RWKAR	H'FFC8 002C	H'1FC8 002C	8	Held *[2]	Held	Held	Held	Pclk
RTC	RDAYAR	H'FFC8 0030	H'1FC8 0030	8	Held *[2]	Held	Held	Held	Pclk
RTC	RMONAR	H'FFC8 0034	H'1FC8 0034	8	Held *[2]	Held	Held	Held	Pclk
RTC	RCR1	H'FFC8 0038	H'1FC8 0038	8	H'00*[2]	H'00*[2]	Held	Held	Pclk
RTC	RCR2	H'FFC8 003C	H'1FC8 003C	8	H'09*[2]	H'00*[2]	Held	Held	Pclk
INTC	ICR	H'FFD0 0000	H'1FD0 0000	16	H'0000*[2]	H'0000*[2]	Held	Held	Pclk
INTC	IPRA	H'FFD0 0004	H'1FD0 0004	16	H'0000	H'0000	Held	Held	Pclk
INTC	IPRB	H'FFD0 0008	H'1FD0 0008	16	H'0000	H'0000	Held	Held	Pclk
INTC	IPRC	H'FFD0 000C	H'1FD0 000C	16	H'0000	H'0000	Held	Held	Pclk
TMU	TOCR	H'FFD8 0000	H'1FD8 0000	8	H'00	H'00	Held	Held	Pclk
TMU	TSTR	H'FFD8 0004	H'1FD8 0004	8	H'00	H'00	Held	H'00*[2]	Pclk
TMU	TCOR0	H'FFD8 0008	H'1FD8 0008	32	H'FFFF FFFF	H'FFFF FFFF	Held	Held	Pclk
TMU	TCNT0	H'FFD8 000C	H'1FD8 000C	32	H'FFFF FFFF	H'FFFF FFFF	Held	Held	Pclk
TMU	TCR0	H'FFD8 0010	H'1FD8 0010	16	H'0000	H'0000	Held	Held	Pclk

Table A.1 Address List (cont)

Module	Register	P4 Address	Area 7 Address*[1]	Size	Power-On Reset	Manual Reset	Sleep	Standby	Synchro-nization Clock
TMU	TCOR1	H'FFD8 0014	H'1FD8 0014	32	H'FFFF FFFF	H'FFFF FFFF	Held	Held	Pclk
TMU	TCNT1	H'FFD8 0018	H'1FD8 0018	32	H'FFFF FFFF	H'FFFF FFFF	Held	Held	Pclk
TMU	TCR1	H'FFD8 001C	H'1FD8 001C	16	H'0000	H'0000	Held	Held	Pclk
TMU	TCOR2	H'FFD8 0020	H'1FD8 0020	32	H'FFFF FFFF	H'FFFF FFFF	Held	Held	Pclk
TMU	TCNT2	H'FFD8 0024	H'1FD8 0024	32	H'FFFF FFFF	H'FFFF FFFF	Held	Held	Pclk
TMU	TCR2	H'FFD8 0028	H'1FD8 0028	16	H'0000	H'0000	Held	Held	Pclk
TMU	TCPR2	H'FFD8 002C	H'1FD8 002C	32	Held	Held	Held	Held	Pclk
SCI	SCSMR1	H'FFE0 0000	H'1FE0 0000	8	H'00	H'00	Held	H'00	Pclk
SCI	SCBRR1	H'FFE0 0004	H'1FE0 0004	8	H'FF	H'FF	Held	H'FF	Pclk
SCI	SCSCR1	H'FFE0 0008	H'1FE0 0008	8	H'00	H'00	Held	H'00	Pclk
SCI	SCTDR1	H'FFE0 000C	H'1FE0 000C	8	H'FF	H'FF	Held	H'FF	Pclk
SCI	SCSSR1	H'FFE0 0010	H'1FE0 0010	8	H'84	H'84	Held	H'84	Pclk
SCI	SCRDR1	H'FFE0 0014	H'1FE0 0014	8	H'00	H'00	Held	H'00	Pclk
SCI	SCSCMR1	H'FFE0 0018	H'1FE0 0018	8	H'00	H'00	Held	H'00	Pclk
SCI	SCSPTR1	H'FFE0 001C	H'1FE0 001C	8	H'00*[2]	H'00*[2]	Held	H'00*[2]	Pclk
SCIF	SCSMR2	H'FFE8 0000	H'1FE8 0000	16	H'0000	H'0000	Held	Held	Pclk
SCIF	SCBRR2	H'FFE8 0004	H'1FE8 0004	8	H'FF	H'FF	Held	Held	Pclk
SCIF	SCSCR2	H'FFE8 0008	H'1FE8 0008	16	H'0000	H'0000	Held	Held	Pclk
SCIF	SCFTDR2	H'FFE8 000C	H'1FE8 000C	8	Undefined	Undefined	Held	Held	Pclk
SCIF	SCFSR2	H'FFE8 0010	H'1FE8 0010	16	H'0060	H'0060	Held	Held	Pclk
SCIF	SCFRDR2	H'FFE8 0014	H'1FE8 0014	8	Undefined	Undefined	Held	Held	Pclk
SCIF	SCFCR2	H'FFE8 0018	H'1FE8 0018	16	H'0000	H'0000	Held	Held	Pclk
SCIF	SCFDR2	H'FFE8 001C	H'1FE8 001C	16	H'0000	H'0000	Held	Held	Pclk
SCIF	SCSPTR2	H'FFE8 0020	H'1FE8 0020	16	H'0000*[2]	H'0000*[2]	Held	Held	Pclk
SCIF	SCLSR2	H'FFE8 0024	H'1FE8 0024	16	H'0000	H'0000	Held	Held	Pclk
Hitachi-UDI	SDIR	H'FFF0 0000	H'1FF0 0000	16	H'FFFF*[2]	Held	Held	Held	Pclk
Hitachi-UDI	SDDR	H'FFF0 0008	H'1FF0 0008	32	Held	Held	Held	Held	Pclk

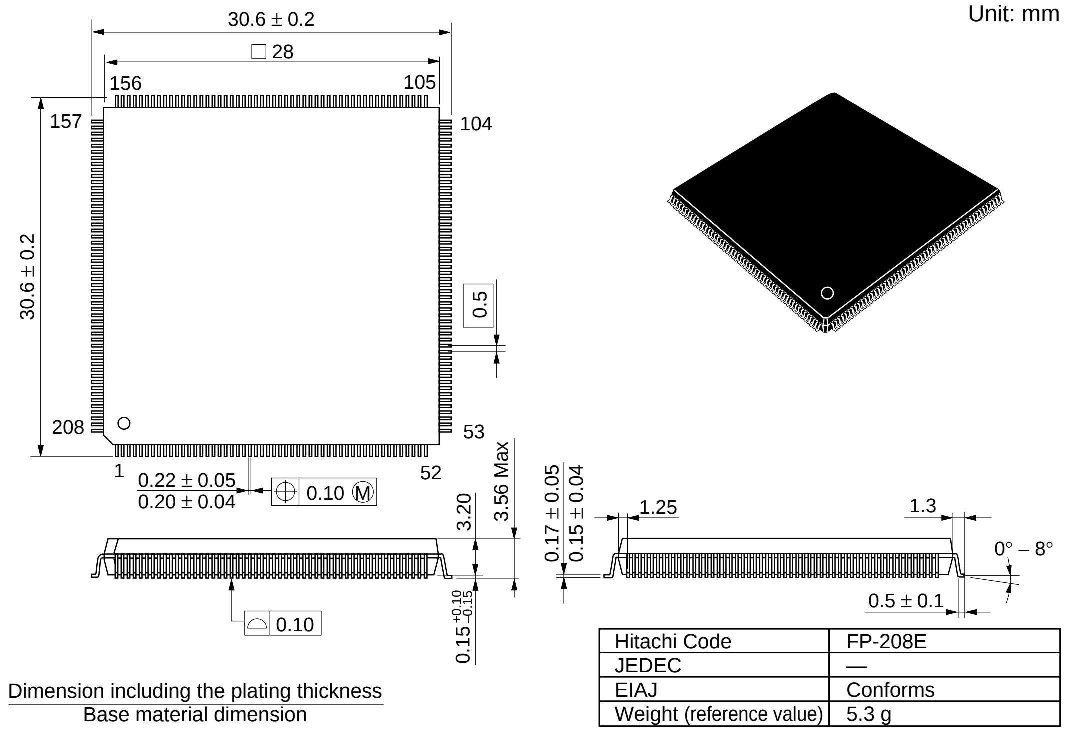
Notes: 1. With control registers, the above addresses in the physical page number field can be accessed by means of a TLB setting. When these addresses are referenced directly without using the TLB, operations are limited.
2. Includes undefined bits. See the descriptions of the individual modules.
3. Use word-size access when writing. Perform the write with the upper byte set to H'5A or H'A5, respectively. Byte- and longword-size writes cannot be used.
Use byte-size access when reading.

Appendix B   Package Dimensions



Hitachi Code	BP-256
JEDEC Code	MO-151
EIAJ Code	–
Weight	3.0 g

Figure B.1   Package Dimensions (256-Pin BGA)



Hitachi Code	FP-208E
JEDEC	—
EIAJ	Conforms
Weight (reference value)	5.3 g

Figure B.2   Package Dimensions (208-Pin QFP)

Appendix C Mode Pin Settings

The MD8–MD0 pin values are input in the event of a power-on reset via the $\overline{\text{RESET}}$ or SCK2/$\overline{\text{MRESET}}$ pin.

Clock Modes

	Pin Values			**Frequency Divider 1**	**PLL1**	**PLL2**	**Initial Clock Frequency Ratio***[2]		
Mode	**MD2**	**MD1**	**MD0**				**CPU Clock**	**Bus Clock**	**Peripheral Module Clock**
0	0	0	0	Off	On	On	6	3/2	3/2
1	0	0	1	Off	On	On	6	1	1
2	0	1	0	On	On	On	3	1	1/2
3	0	1	1	Off	On	On	6	2	1
4	1	0	0	On	On	On	3	3/2	3/4
5	1	0	1	Off	On	On	6	3	3/2

Notes: 1. MD2–MD0 pin value combinations other than those shown above cannot be set.
2. Taking the input clock (EXTAL or crystal resonator frequency) as 1.

Area 0 Bus Width

Pin Value		
MD4	**MD3**	**Bus Width**
0	0	64 bits
	1	8 bits
1	0	16 bits
	1	32 bits

Endian

Pin Value	
MD5	**Endian**
0	Big endian
1	Little endian

Area 0 Memory Type

Pin Value	
MD6	**Memory Type**
0	MPX bus
1	Normal memory

Master/Slave

Pin Value	
MD7	**Master/Slave**
0	Slave
1	Master

Clock Input

Pin Value	
MD8	**Clock Input**
0	External input clock
1	Crystal resonator

Appendix D $\overline{\text{CKIO2ENB}}$ Pin Configuration

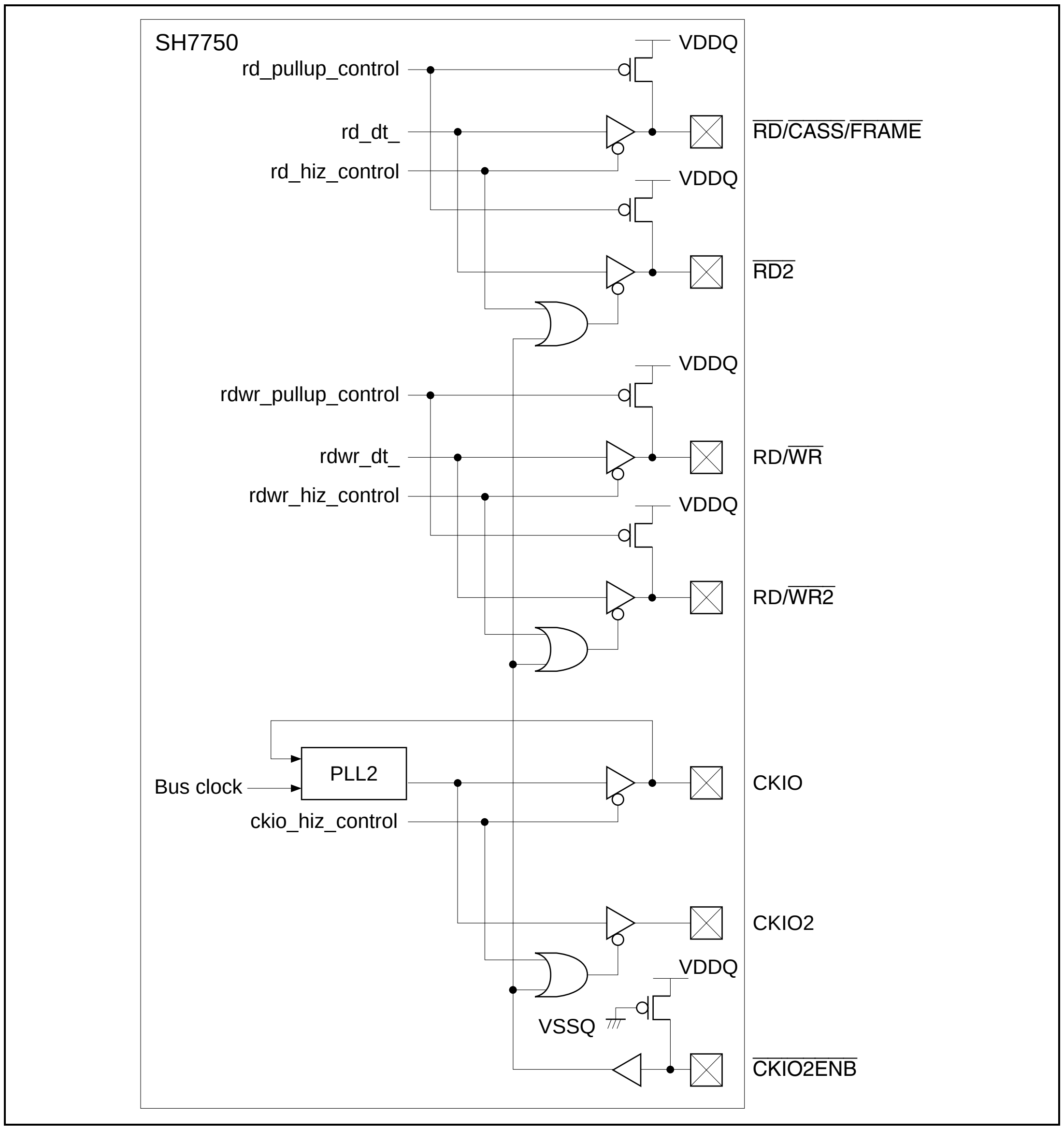


Figure D.1 $\overline{\text{CKIO2ENB}}$ Pin Configuration

$\overline{\text{CKIO2ENB}}$	**Description**
0	$\overline{\text{RD2}}$, RD/$\overline{\text{WR2}}$, and CKIO2 have the same pin states as $\overline{\text{RD}}$, RD/$\overline{\text{WR}}$, and CKIO, respectively
1	$\overline{\text{RD2}}$, RD/$\overline{\text{WR2}}$, and CKIO2 are in the high-impedance state

Note: CKIO is fed back to PLL2 to coordinate the external clock and internal clock phases. However, CKIO2 is not fed back.

Appendix E Pin Functions

E.1 Pin States

Table E.1 Pin States in Reset, Power-Down State, and Bus-Released State

Signal Name	I/O	Reset (Power-On)		Reset (Manual)		Sleep	Standby	Bus Released	Notes
		Master	Slave	Master	Slave				
D0–D7	I/O	Z	Z	Z	Z	Z	Z	Z	
D8–D15	I/O	Z	Z	Z	Z	Z	Z	Z	
D16–D23	I/O	Z	Z	Z	Z	Z	Z	Z	
D24–D31	I/O	Z	Z	Z	Z	Z	Z	Z	
D32–D39	I/O	Z	Z	ZK	ZK	ZK	ZK	ZK	Output state held when used as port
D40–D47	I/O	Z	Z	ZK	ZK	ZK	ZK	ZK	Output state held when used as port
D48–D55	I/O	Z	Z	Z	Z	Z	Z	Z	
D56–D63	I/O	Z	Z	Z	Z	Z	Z	Z	
A0, A1, A18–A25	O	Z	Z	Z	Z	Z	Z	Z	
A2–A17	O	Z	Z	ZO*^{9}	Z	O	ZO*^{7}	Z	
$\overline{\text{RESET}}$	I	I	I	I	I	I	I	I	
$\overline{\text{BACK}}$/$\overline{\text{BSREQ}}$	O	H	H	H	H	O	H	O	
$\overline{\text{BREQ}}$/$\overline{\text{BSACK}}$	I	I	I	I	I	I	I	I	
$\overline{\text{BS}}$	O	H	Z	H	Z	O*^{4}	ZH*^{7}	Z	
CKE	O	H	Z	O*^{6}	Z	O*^{6}	L	O*^{6}	
$\overline{\text{CS6}}$–$\overline{\text{CS0}}$	O	H	Z	H	Z	O*^{4}	ZH*^{7}	Z	
$\overline{\text{RAS}}$	O	H	Z	O*^{6}	Z	O*^{4}	ZO*^{5}	ZO*^{5}	
$\overline{\text{RD}}$/$\overline{\text{CASS}}$	O	H	Z	O*^{6}	Z	O*^{4}	ZO*^{5}	ZO*^{5}	
RD/$\overline{\text{WR}}$	O	H	Z	H	Z	O*^{4}	ZH*^{7}	Z	
$\overline{\text{RDY}}$	I	I	I	I	I	I	I	I	

Table E.1 Pin States in Reset, Power-Down State, and Bus-Released State (cont)

Signal Name	I/O	Reset (Power-On)		Reset (Manual)		Sleep	Standby	Bus Released	Notes
		Master	Slave	Master	Slave				
$\overline{WE7}$/$\overline{CAS7}$/DQM7	O	H	Z	O*6	Z	O*4	ZO*5	ZO*5	
$\overline{WE6}$/$\overline{CAS6}$/DQM6	O	H	Z	O*6	Z	O*4	ZO*5	ZO*5	
$\overline{WE5}$/$\overline{CAS5}$/DQM5	O	H	Z	O*6	Z	O*4	ZO*5	ZO*5	
$\overline{WE4}$/$\overline{CAS4}$/DQM4	O	H	Z	O*6	Z	O*4	ZO*5	ZO*5	
$\overline{WE3}$/$\overline{CAS3}$/DQM3	O	H	Z	O*6	Z	O*4	ZO*5	ZO*5	
$\overline{WE2}$/$\overline{CAS2}$/DQM2	O	H	Z	O*6	Z	O*4	ZO*5	ZO*5	
$\overline{WE1}$/$\overline{CAS1}$/DQM1	O	H	Z	O*6	Z	O*4	ZO*5	ZO*5	
$\overline{WE0}$/$\overline{CAS0}$/DQM0	O	H	Z	O*6	Z	O*4	ZO*5	ZO*5	
DACK1–DACK0	O	L	L	L	L	O*4	ZO*8	O	DMAC
MD7/TXD	I/O	I	I	I	I	IO	ZO*8	IO	SCI
MD6/$\overline{IOIS16}$	I	I	I	I	I	I	I	I	PCMCIA (I/O)
MD5/$\overline{RAS2}$	I/O*1	I	I	IO*6	I	IO*4	IO*5	IO*5	DRAM2
MD4/$\overline{CE2B}$	I/O*2	I	I	IH	I	IO*4	IH*7	I	PCMCIA
MD3/$\overline{CE2A}$	I/O*3	I	I	IH	I	IO*4	IH*7	I	PCMCIA
CKIO	O	O	O	ZO*11	ZO*11	ZO*11	ZO*11	ZO*11	
STATUS1–STATUS0	O	O	O	O	O	O	O	O	
$\overline{IRL3}$–$\overline{IRL0}$	I	I	I	I	I	I	I	I	INTC
NMI	I	I	I	I	I	I	I	I	INTC
$\overline{DREQ1}$–$\overline{DREQ0}$	I	I	I	I	I	I	I	I	DMAC
DRAK1–DRAK0	O	L	L	L	L	O*4	ZO*8	O	DMAC
MD0/SCK	I/O	I	I	I	I	IO	IO*8	IO	SCI
RXD	I	I	I	I	I	I	I	I	SCI
SCK2/$\overline{MRESET}$	I	I	I	I	I	I	I	I	SCIF
MD1/TXD2	I/O	I	I	I	I	IO	IO*8	IO	SCIF
MD2/RXD2	I	I	I	I	I	I	I	I	SCIF
$\overline{CTS2}$	I/O	I	I	I	I	IO	IO*8	IO	SCIF
MD8/$\overline{RTS2}$	I/O	I	I	I	I	IO	IO*8	IO	SCIF
TCLK	I/O	I	I	I	I	IO	IO	IO	TMU

Table E.1 Pin States in Reset, Power-Down State, and Bus-Released State (cont)

Signal Name	I/O	Reset (Power-On)		Reset (Manual)		Sleep	Standby	Bus Released	Notes
		Master	Slave	Master	Slave				
TDO	I/O	O	O	O	O	O	O	O	Hitachi-UDI
TMS	I	I	I	I	I	I	I	I	Hitachi-UDI
TCK	I	I	I	I	I	I	I	I	Hitachi-UDI
TDI	I	I	I	I	I	I	I	I	Hitachi-UDI
$\overline{\text{TRST}}$	I	I	I	I	I	I	I	I	Hitachi-UDI
CKIO2*^{10}	O	O	O	ZO*^{11}	ZO*^{11}	ZO*^{11}	ZO*^{11}	ZO*^{11}	
$\overline{\text{RD2}}$*^{10}	O	H	Z	O*^{6}	Z	O*^{4}	ZO*^{5}	ZO*^{5}	
RD/$\overline{\text{WR2}}$*^{10}	O	H	Z	H	Z	O*^{4}	ZH*^{7}	Z	
$\overline{\text{CKIO2ENB}}$	I	I	I	I	I	I	I	I	

Notes: I: Input

O: Output

H: High-level output

L: Low-level output

Z: High-impedance

K: Output state held

1. Output when area 2 DRAM is used.
2. Output when area 5 PCMCIA is used.
3. Output when area 6 PCMCIA is used.
4. Depends on refresh and DMAC operations.
5. Z (I) or O (refresh), depending on register setting (BCR1.HIZCNT).
6. Depends on refresh operation.
7. Z (I) or H (state held), depending on register setting (BCR1.HIZMEM).
8. Z or O, depending on register setting (STBCR.PHZ).
9. Output when refreshing is set.
10. Operation in respective state when $\overline{\text{CKIO2ENB}}$ = 0; Z when $\overline{\text{CKIO2ENB}}$ = 1.
11. Z or O, depending on register setting (FRQCR.CKOEN).

E.2 Handling of Unused Pins

- When RTC is not used
  - EXTAL2: Pull up to 3.3 V
  - XTAL2: Leave unconnected
  - VDD-RTC: Power supply (3.3 V)
  - VSS-RTC: Power supply (0 V)
- When PLL1 is not used
  - VDD-PLL1: Power supply (3.3 V)
  - VSS-PLL1: Power supply (0 V)
- When PLL2 is not used
  - VDD-PLL2: Power supply (3.3 V)
  - VSS-PLL2: Power supply (0 V)
- When on-chip crystal oscillator is not used
  - XTAL: Leave unconnected
  - VDD-CPG: Power supply (3.3 V)
  - VSS-CPG: Power supply (0 V)

Appendix F Synchronous DRAM Address Multiplexing Tables

(1) BUS 64 (16M: 512k × 16b × 2) × 4
AMX 0 AMXEXT 0 16M, column-addr-8bit 8MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A14	A22	A22	A11	BANK selects bank address
A13	A21	H/L	A10	Address precharge setting
A12	A20	0	A9	Address
A11	A19	0	A8	
A10	A18	A10	A7	
A9	A17	A9	A6	
A8	A16	A8	A5	
A7	A15	A7	A4	
A6	A14	A6	A3	
A5	A13	A5	A2	
A4	A12	A4	A1	
A3	A11	A3	A0	
A2	Not used			
A1	Not used			
A0	Not used			

(2) BUS 32 (16M: 512k × 16b × 2) × 2
AMX 0 AMXEXT 0 16M, column-addr-8bit 4MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A14				
A13	A21	A21	A11	BANK selects bank address
A12	A20	H/L	A10	Address precharge setting
A11	A19	0	A9	Address
A10	A18	0	A8	
A9	A17	A9	A7	
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	Not used			
A0	Not used			

(3) BUS 64 (16M: 512k × 16b × 2) × 4
AMX 0 AMXEXT 1 16M, column-addr-8bit 8MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A14	A21	A21	A11	BANK selects bank address
A13	A22	H/L	A10	Address precharge setting
A12	A20	0	A9	Address
A11	A19	0	A8	
A10	A18	A10	A7	
A9	A17	A9	A6	
A8	A16	A8	A5	
A7	A15	A7	A4	
A6	A14	A6	A3	
A5	A13	A5	A2	
A4	A12	A4	A1	
A3	A11	A3	A0	
A2	Not used			
A1	Not used			
A0	Not used			

(4) BUS 32 (16M: 512k × 16b × 2) × 2
AMX 0 AMXEXT 1 16M, column-addr-8bit 4MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A14				
A13	A20	A20	A11	BANK selects bank address
A12	A21	H/L	A10	Address precharge setting
A11	A19	0	A9	Address
A10	A18	0	A8	
A9	A17	A9	A7	
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	Not used			
A0	Not used			

(5) BUS 64 (16M: 1M × 8b × 2) × 8
AMX 1 AMXEXT 0 16M, column-addr-9bit 16MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	**CAS Cycle**		
A14	A23	A23	A11	BANK selects bank address
A13	A22	H/L	A10	Address precharge setting
A12	A21	0	A9	Address
A11	A20	A11	A8	
A10	A19	A10	A7	
A9	A18	A9	A6	
A8	A17	A8	A5	
A7	A16	A7	A4	
A6	A15	A6	A3	
A5	A14	A5	A2	
A4	A13	A4	A1	
A3	A12	A3	A0	
A2	Not used			
A1	Not used			
A0	Not used			

(6) BUS 32 (16M: 1M × 8b × 2) × 4
AMX 1 AMXEXT 0 16M, column-addr-9bit 8MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A14				
A13	A22	A22	A11	BANK selects bank address
A12	A21	H/L	A10	Address precharge setting
A11	A20	0	A9	Address
A10	A19	A10	A8	
A9	A18	A9	A7	
A8	A17	A8	A6	
A7	A16	A7	A5	
A6	A15	A6	A4	
A5	A14	A5	A3	
A4	A13	A4	A2	
A3	A12	A3	A1	
A2	A11	A2	A0	
A1	Not used			
A0	Not used			

(7) BUS 64 (16M: 1M × 8b × 2) × 8
AMX 1 AMXEXT 1 16M, column-addr-9bit 16MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A14	A22	A22	A11	BANK selects bank address
A13	A23	H/L	A10	Address precharge setting
A12	A21	0	A9	Address
A11	A20	A11	A8	
A10	A19	A10	A7	
A9	A18	A9	A6	
A8	A17	A8	A5	
A7	A16	A7	A4	
A6	A15	A6	A3	
A5	A14	A5	A2	
A4	A13	A4	A1	
A3	A12	A3	A0	
A2	Not used			
A1	Not used			
A0	Not used			

(8) BUS 32 (16M: 1M × 8b × 2) × 4
AMX 1 AMXEXT 1 16M, column-addr-9bit 8MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A14				
A13	A21	A21	A11	BANK selects bank address
A12	A22	H/L	A10	Address precharge setting
A11	A20	0	A9	Address
A10	A19	A10	A8	
A9	A18	A9	A7	
A8	A17	A8	A6	
A7	A16	A7	A5	
A6	A15	A6	A4	
A5	A14	A5	A3	
A4	A13	A4	A2	
A3	A12	A3	A1	
A2	A11	A2	A0	
A1	Not used			
A0	Not used			

(9) BUS 64 (64M: 1M × 16b × 4) × 4
AMX 2 64M, column-addr-8bit 32MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A16	A24	A24	A13	BANK selects bank address
A15	A23	A23	A12	
A14	A22	0	A11	Address precharge setting
A13	A21	H/L	A10	
A12	A20	0	A9	Address
A11	A19	0	A8	
A10	A18	A10	A7	
A9	A17	A9	A6	
A8	A16	A8	A5	
A7	A15	A7	A4	
A6	A14	A6	A3	
A5	A13	A5	A2	
A4	A12	A4	A1	
A3	A11	A3	A0	
A2	Not used			
A1	Not used			
A0	Not used			

(10) BUS 32 (64M: 1M × 16b × 4) × 2
AMX 2 64M, column-addr-8bit 16MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A16				
A15	A23	A23	A13	BANK selects bank address
A14	A22	A22	A12	
A13	A21	0	A11	Address precharge setting
A12	A20	H/L	A10	
A11	A19	0	A9	Address
A10	A18	0	A8	
A9	A17	A9	A7	
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	Not used			
A0	Not used			

(11) BUS 64 (64M: 2M × 8b × 4) × 8
AMX 3 64M, column-addr-9bit 64MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A16	A25	A25	A13	BANK selects bank address
A15	A24	A24	A12	
A14	A23	0	A11	Address precharge setting
A13	A22	H/L	A10	
A12	A21	0	A9	Address
A11	A20	A11	A8	
A10	A19	A10	A7	
A9	A18	A9	A6	
A8	A17	A8	A5	
A7	A16	A7	A4	
A6	A15	A6	A3	
A5	A14	A5	A2	
A4	A13	A4	A1	
A3	A12	A3	A0	
A2	Not used			
A1	Not used			
A0	Not used			

(12) BUS 32 (64M: 2M × 8b × 4) × 4
AMX 3 64M, column-addr-9bit 32MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A16				
A15	A24	A24	A13	BANK selects bank address
A14	A23	A23	A12	
A13	A22	0	A11	Address precharge setting
A12	A21	H/L	A10	
A11	A20	0	A9	Address
A10	A19	A10	A8	
A9	A18	A9	A7	
A8	A17	A8	A6	
A7	A16	A7	A5	
A6	A15	A6	A4	
A5	A14	A5	A3	
A4	A13	A4	A2	
A3	A12	A3	A1	
A2	A11	A2	A0	
A1	Not used			
A0	Not used			

(13) BUS 64 (64M: 512k × 32b × 4) × 2
AMX 4 64M, column-addr-8bit 16MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A15	A23	A23	A12	BANK selects bank address
A14	A22	A22	A11	
A13	A21	H/L	A10	Address precharge setting
A12	A20	0	A9	Address
A11	A19	0	A8	
A10	A18	A10	A7	
A9	A17	A9	A6	
A8	A16	A8	A5	
A7	A15	A7	A4	
A6	A14	A6	A3	
A5	A13	A5	A2	
A4	A12	A4	A1	
A3	A11	A3	A0	
A2	Not used			
A1	Not used			
A0	Not used			

(14) BUS 32 (64M: 512k × 32b × 4) × 1
AMX 4 64M, column-addr-8bit 8MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	**CAS Cycle**		
A15				
A14	A22	A22	A12	BANK selects bank address
A13	A21	A21	A11	
A12	A20	H/L	A10	Address precharge setting
A11	A19	0	A9	Address
A10	A18	0	A8	
A9	A17	A9	A7	
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	Not used			
A0	Not used			

(15) BUS 64 (64M: 1M × 32b × 2) × 2
AMX 5 64M, column-addr-8bit 16MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A15	A23	A23	A12	BANK selects bank address
A14	A22	0	A11	
A13	A21	H/L	A10	Address precharge setting
A12	A20	0	A9	Address
A11	A19	0	A8	
A10	A18	A10	A7	
A9	A17	A9	A6	
A8	A16	A8	A5	
A7	A15	A7	A4	
A6	A14	A6	A3	
A5	A13	A5	A2	
A4	A12	A4	A1	
A3	A11	A3	A0	
A2	Not used			
A1	Not used			
A0	Not used			

(16) BUS 32 (64M: 1M × 32b × 2) × 1
AMX 5 64M, column-addr-8bit 8MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A15				
A14	A22	A22	A12	BANK selects bank address
A13	A21	0	A11	
A12	A20	H/L	A10	Address precharge setting
A11	A19	0	A9	Address
A10	A18	0	A8	
A9	A17	A9	A7	
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	Not used			
A0	Not used			

(17) BUS 64 (16M: 256k × 32b × 2) × 2
AMX 7 16M, column-addr-8bit 4MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A13	A21	A21	A10	BANK selects bank address
A12	A20	H/L	A9	Address precharge setting
A11	A19	0	A8	Address
A10	A18	A10	A7	
A9	A17	A9	A6	
A8	A16	A8	A5	
A7	A15	A7	A4	
A6	A14	A6	A3	
A5	A13	A5	A2	
A4	A12	A4	A1	
A3	A11	A3	A0	
A2	Not used			
A1	Not used			
A0	Not used			

(18) BUS 32 (16M: 256k × 32b × 2) × 1
AMX 7 16M, column-addr-8bit 2MB

SH7750 Address Pins			Synchronous DRAM Address Pins	Function
	RAS Cycle	CAS Cycle		
A13				
A12	A20	A20	A10	BANK selects bank address
A11	A19	H/L	A9	Address precharge setting
A10	A18	0	A8	Address
A9	A17	A9	A7	
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	Not used			
A0	Not used			

Appendix G SH7750 On-Demand Data Transfer Mode

G.1 Pins in DDT Mode

Figure G.1 shows the system configuration in DDT mode.

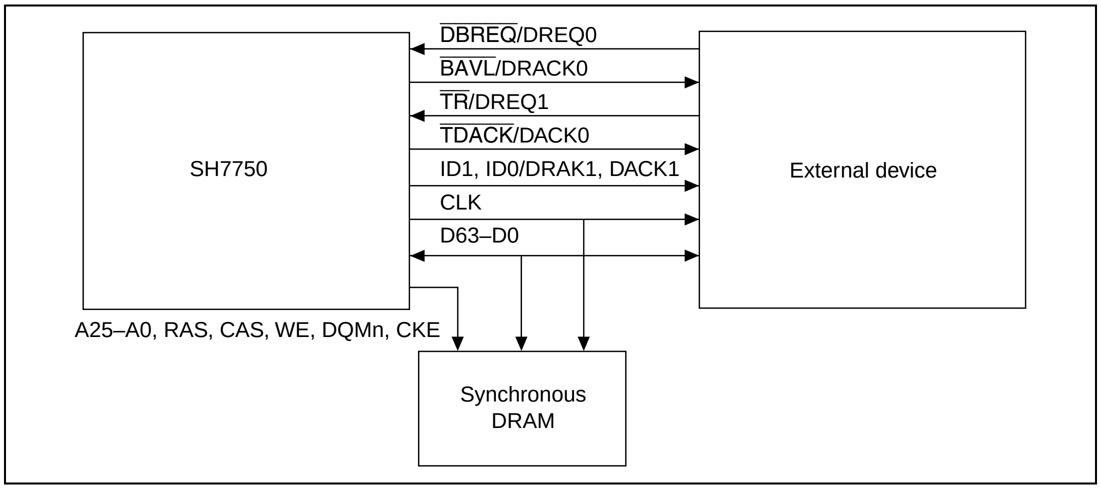

$\overline{\text{DBREQ}}$/DREQ0
$\overline{\text{BAVL}}$/DRACK0
$\overline{\text{TR}}$/DREQ1
$\overline{\text{TDACK}}$/DACK0
ID1, ID0/DRAK1, DACK1
CLK
D63–D0
SH7750
External device
A25–A0, RAS, CAS, WE, DQMn, CKE
Synchronous DRAM

Figure G.1 System Configuration in On-Demand Data Transfer Mode

- **$\overline{\text{DBREQ}}$:** Data bus release request signal for transmitting the data transfer request format (DTR format) or a DMA request from an external device to the DMAC

  If there is a wait for release of the data bus, an external device can have the data bus released by asserting $\overline{\text{DBREQ}}$. When $\overline{\text{DBREQ}}$ is accepted, the BSC asserts $\overline{\text{BAVL}}$.

- **$\overline{\text{BAVL}}$:** Data bus D63–D0 release signal

  Assertion of $\overline{\text{BAVL}}$ means that the data bus will be released two cycles later.

- **$\overline{\text{TR}}$:** Transfer request signal

  Assertion of $\overline{\text{TR}}$ has the following different meanings.

  — In normal data transfer mode (except channel 0), $\overline{\text{TR}}$ is asserted, and at the same time the DTR format is output, two cycles after $\overline{\text{BAVL}}$ is asserted.

  — In the case of the handshake protocol without use of the data bus, asserting $\overline{\text{TR}}$ enables a transfer request to be issued for the channel for which a transfer request was made immediately before. This function can be used only when $\overline{\text{BAVL}}$ is not asserted two cycles earlier.

  — In the case of direct data transfer mode (valid only for channel 2), a direct transfer request can be made to channel 2 by asserting $\overline{\text{DBREQ}}$ and $\overline{\text{TR}}$ simultaneously.

- **$\overline{\text{TDACK}}$:** Reply strobe signal for external device from DMAC
  In the case of a read cycle, the SH7750 asserts $\overline{\text{TDACK}}$ in the same cycle in which valid read data is carried. In the case of a write cycle, the SH7750 asserts $\overline{\text{TDACK}}$ two cycles before the valid write data output cycle.

- **ID1, ID0:** Channel number notification signals
  — 00: Channel 0 (means demand data transfer)
  — 01: Channel 1
  — 10: Channel 2
  — 11: Channel 3

Data Transfer Request Format

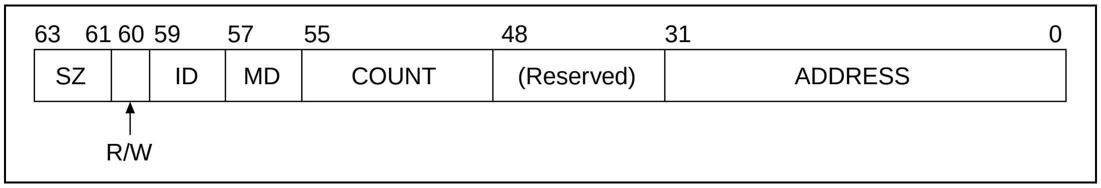


Figure G.2 Data Transfer Request Format

The data transfer request format (DTR format) consists of 64 bits. In the case of normal data transfer mode (channel 0, except channel 0) and the handshake protocol using the data bus, the transfer data size, read/write access, channel number, transfer request mode, number of transfers, and transfer source or transfer destination address are specified. A specification in bits 47–32 is invalid.

In normal data transfer mode (channel 0), only single address mode can be set. With the DTR format, DS = (0: MD = 10, 11, 1: MD = 01), RL = 0, AL = 0, DM[1:0] = 01, SM[1:0] = 01, RS[3:0] = (0010: R/W = 0, 0011: R/W = 1), TM = (0: MD = 11, 1: MD = 01, 10), TS[2:0] = (SZ), and IE = 0 settings are made in DMA channel control register 0, COUNT is set in transfer count register 0, and ADDRESS is set in source/destination address register 0. Therefore, in DDT mode, the above control registers cannot be written to by the CPU, but can be read.

Bits 63 to 61: Transmit Size (SZ2–SZ0)

- 000: Byte size (8-bit) specification
- 001: Word size (16-bit) specification
- 010: Longword size (32-bit) specification
- 011: Quadword size (64-bit) specification
- 100: 32-byte block transfer specification
- 101: Reserved
- 110: Reserved
- 111: Transfer end specification

Bit 60: Read/Write (R/W)

- 0: Memory read specification
- 1: Memory write specification

Bits 59 and 58: Channel Number (ID1, ID0)

- 00: Channel 0 (demand data transfer)
- 01: Channel 1
- 10: Channel 2
- 11: Channel 3

Bits 57 and 56: Transfer Request Mode (MD1, MD0)

- 00: Handshake protocol (data bus used)
- 01: Burst mode (edge detection) specification
- 10: Burst mode (level detection) specification
- 11: Cycle steal mode specification

Bits 55 to 48: Transfer Count (COUNT7–COUNT0)

- 00000000: Maximum number of transfers (16M)

Bits 47 to 32: Reserved

Bits 31 to 0: Address (ADDRESS31–ADDRESS0)

- R/W = 0: Transfer source address specification
- R/W = 1: Transfer destination address specification

Notes: 1. Only the ID field is valid for channels 1 to 3.

2. To start data transfer on channel 0, the initial value of MD in the DTR format must be 01, 10, or 11.
3. The COUNT field is ignored if MD = 00.
4. In edge-sense burst mode, DMA transfer is executed continuously. In level-sense burst mode and cycle steal mode, a handshake protocol is used to transfer each unit of data.

5. The maximum number of transfers can be specified by setting COUNT = 0 as DTR format initialization data. If the amount of data to be transferred is unknown, set COUNT = 0, start DMA transfer, and transfer the DTR format (ID = 00, MD ≠ 00, SZ = 111) when the required amount of data has been transferred. This will terminate DMA transfer on channel 0.

   In this case, the TE bit in DMA channel control register 0 is not set, but transfer cannot be restarted.

G.2 Transfer Request Acceptance on Each Channel

On channel 0, a DMA data transfer request can be made by means of the DTR format. No further transfer requests are accepted between DTR format acceptance and the end of the data transfer.

On channels 1 to 3, output a transfer request from an external device by means of the DTR format (ID = 01, 10, or 11) after making DMAC control register settings in the same way as in normal DMA mode. Each of channels 1 to 3 has a request queue that can accept up to four transfer requests. When a request queue is full, the fifth and subsequent transfer requests will be ignored, and so transfer requests must not be output.

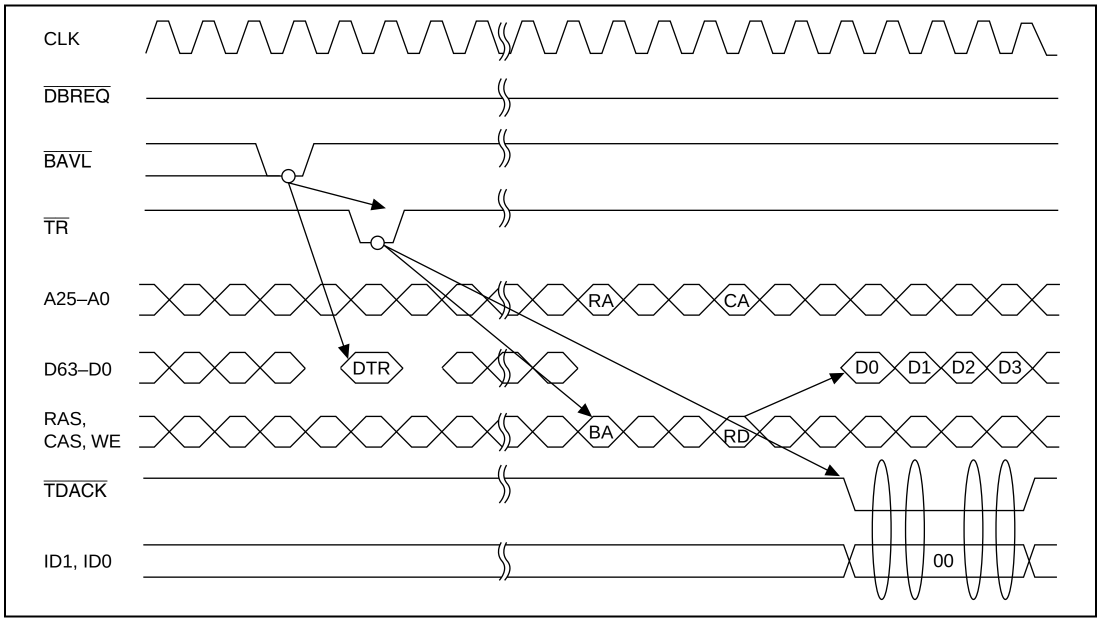


Figure G.3 Single Address Mode/Burst Mode/External Bus → External Device 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer

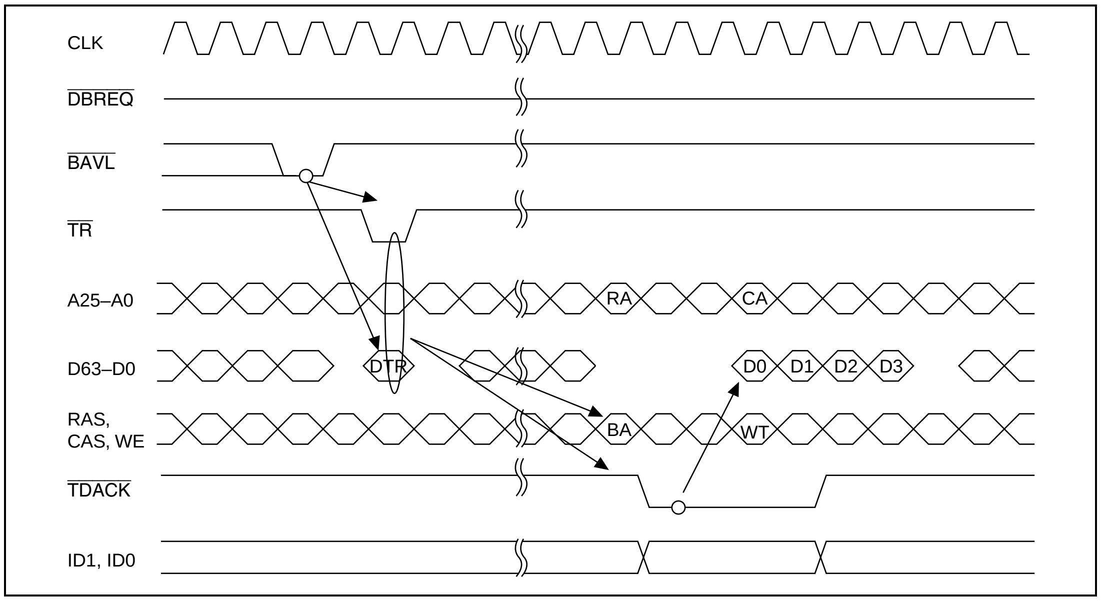


Figure G.4 Single Address Mode/Burst Mode/External Device → External Bus 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer

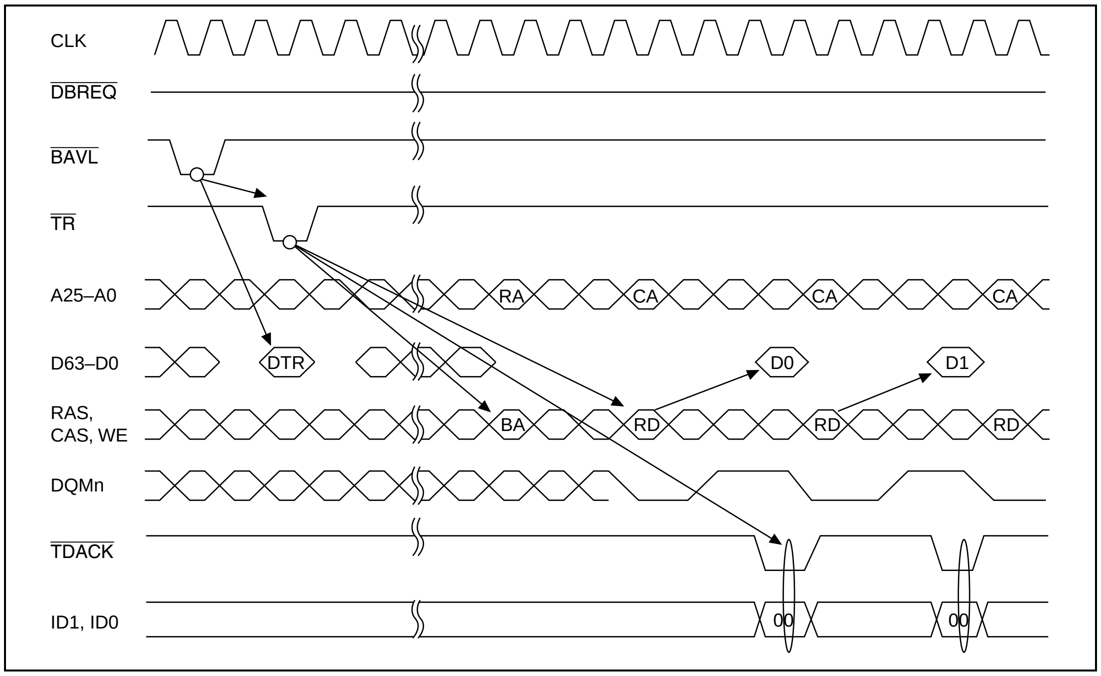


Figure G.5 Single Address Mode/Burst Mode/External Bus → External Device 64-Bit Transfer/Channel 0 On-Demand Data Transfer

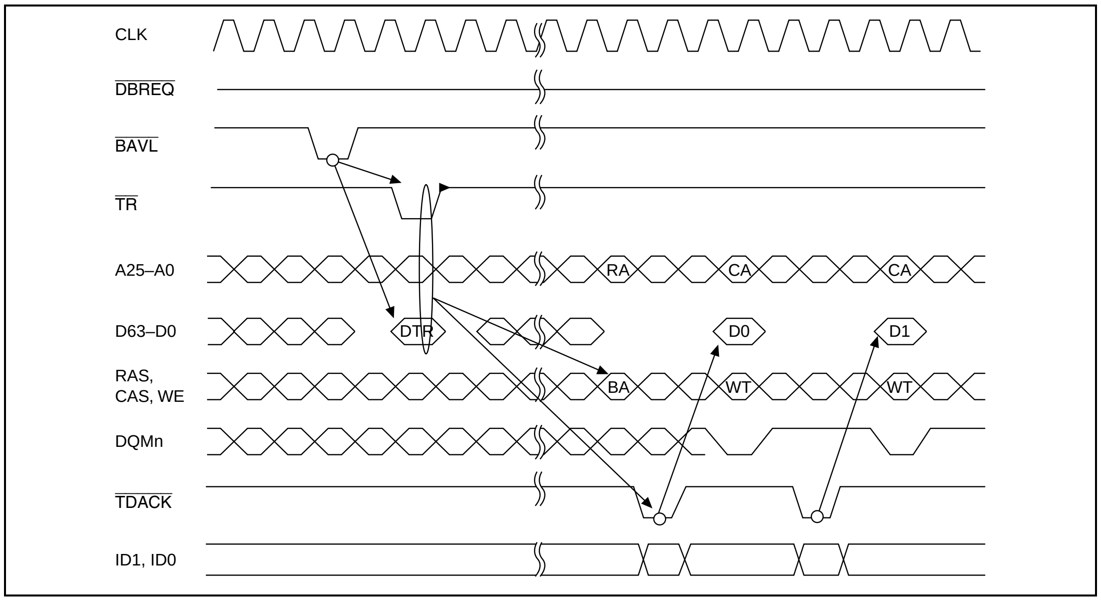


Figure G.6 Single Address Mode/Burst Mode/External Device → External Bus 64-Bit Transfer/Channel 0 On-Demand Data Transfer

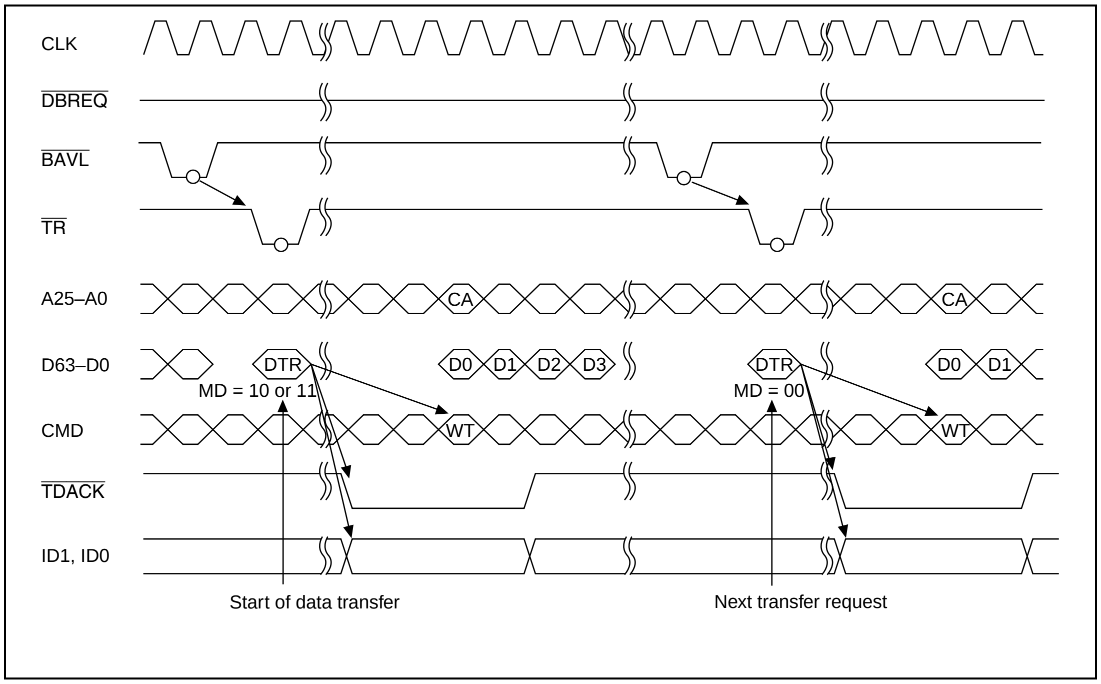


Figure G.7 Handshake Protocol Using Data Bus (Channel 0 On-Demand Data Transfer)

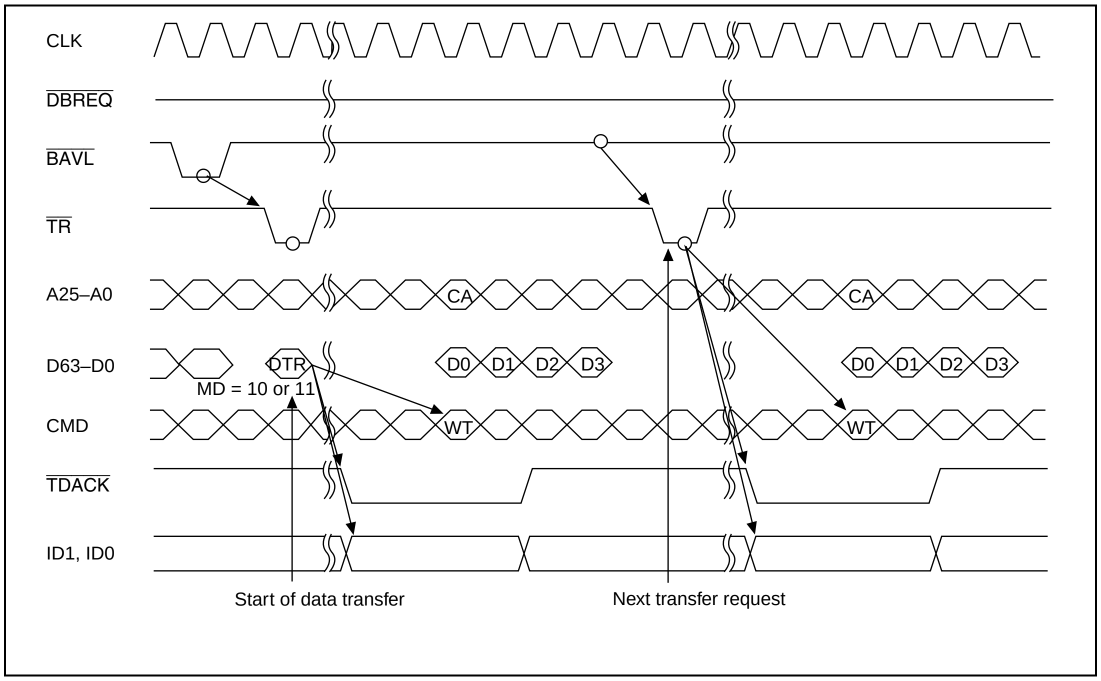


Figure G.8 Handshake Protocol without Use of Data Bus (Channel 0 On-Demand Data Transfer)

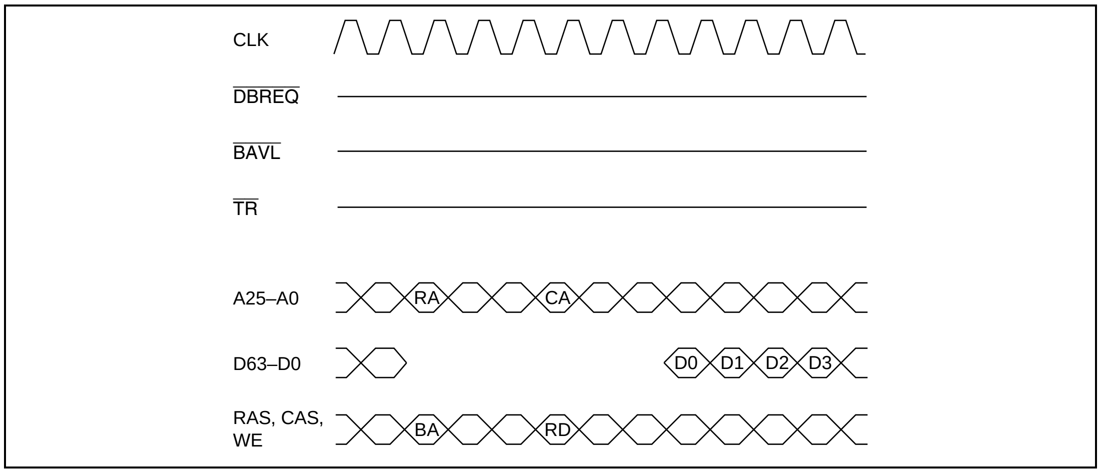


Figure G.9 Read from Synchronous DRAM Precharge Bank

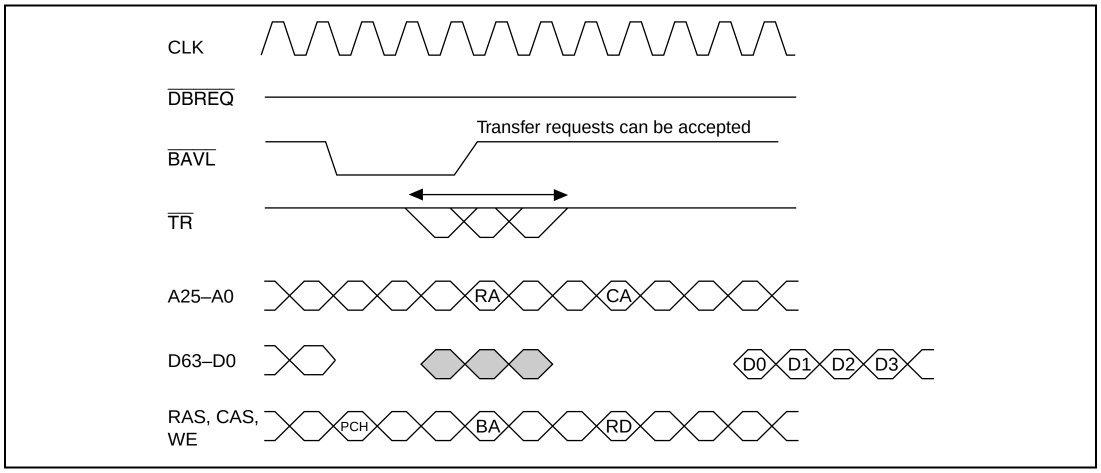


Figure G.10 Read from Synchronous DRAM Non-Precharge Bank (Row Miss)

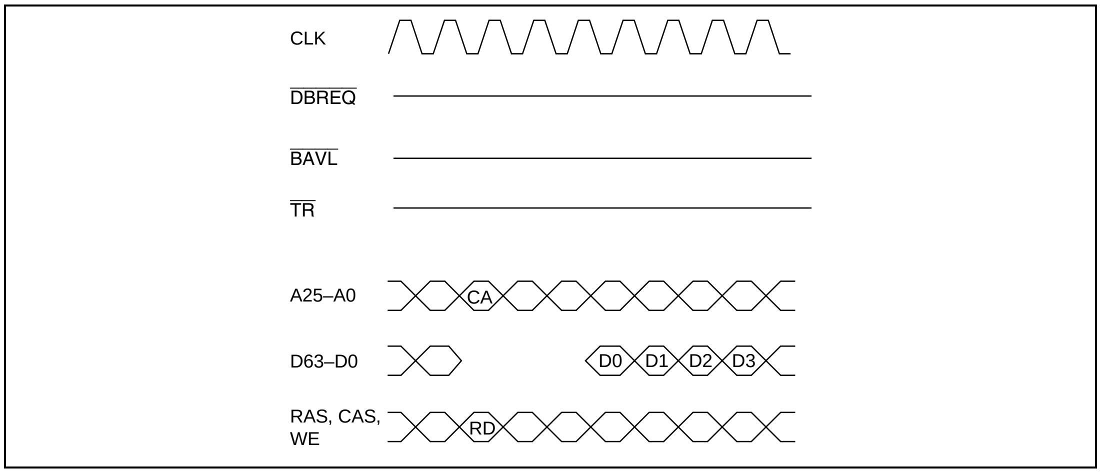


Figure G.11 Read from Synchronous DRAM (Row Hit)

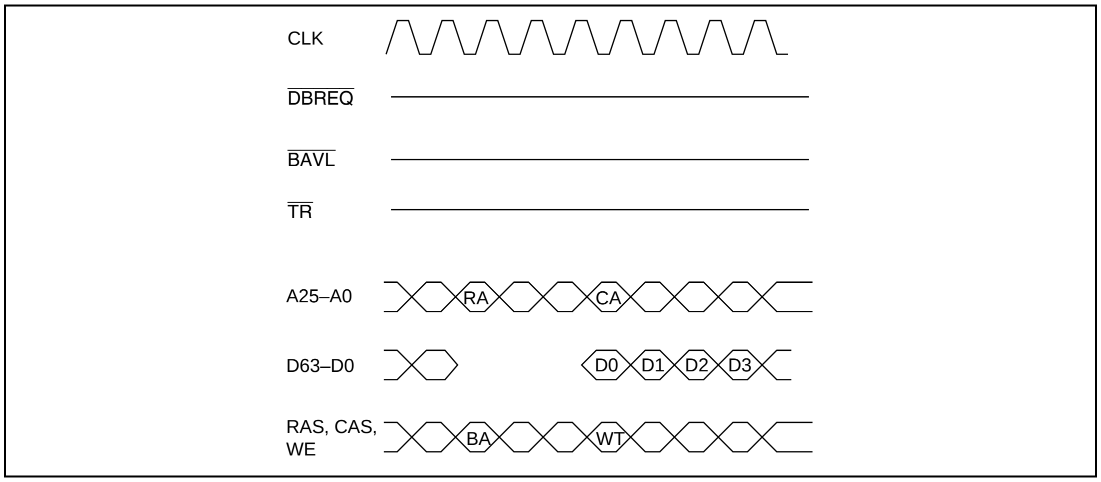


Figure G.12 Write to Synchronous DRAM Precharge Bank

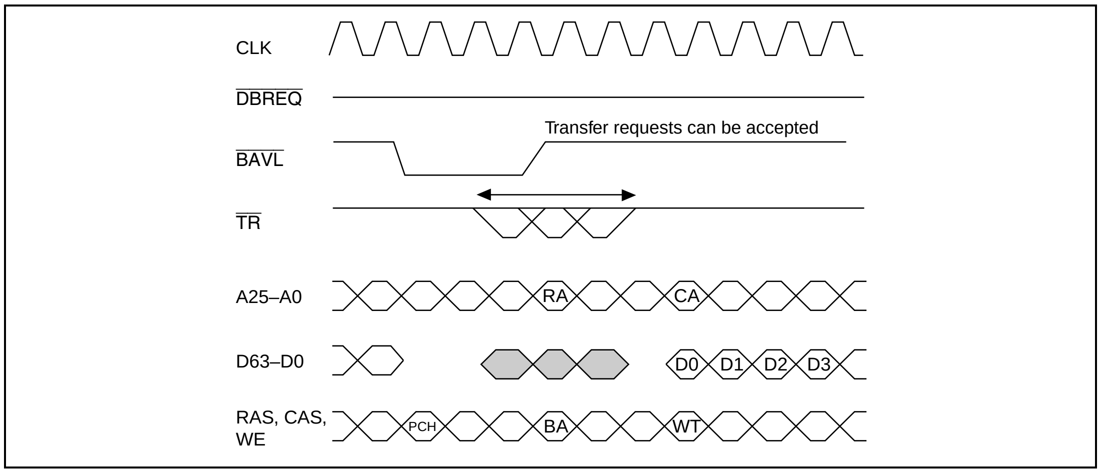


Figure G.13 Write to Synchronous DRAM Non-Precharge Bank (Row Miss)

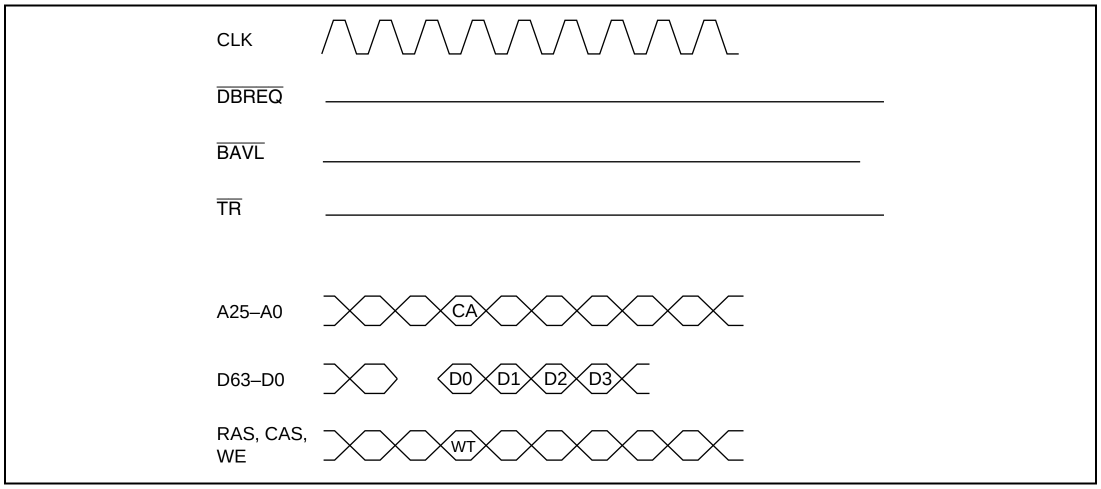


Figure G.14 Write to Synchronous DRAM (Row Hit)

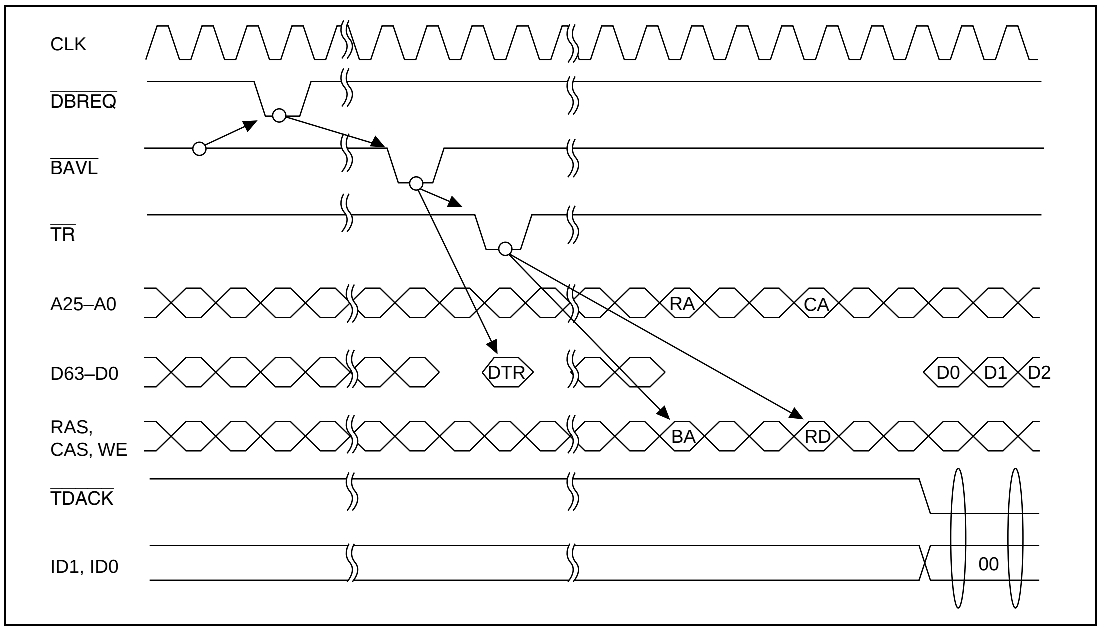


Figure G.15 Single Address Mode/Burst Mode/External Bus → External Device 32-Byte Block Transfer/Channel 0 On-Demand Data Transfer

DMA Operation Register (DMAOR)

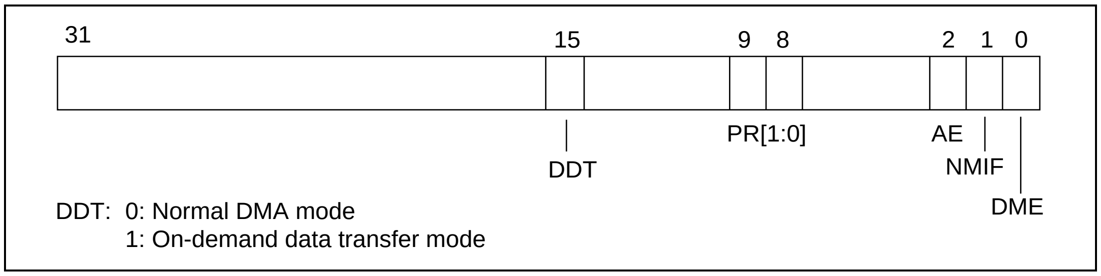


Figure G.16 DDT Mode Setting

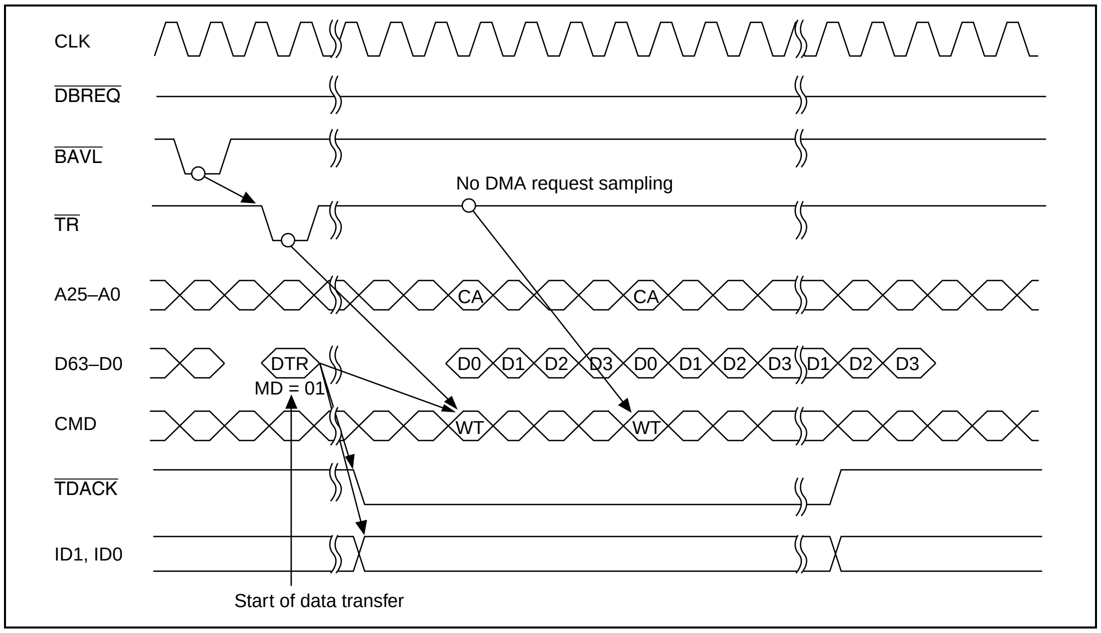


Figure G.17 Single Address Mode/Burst Mode/Edge Detection/ External Device → External Bus Data Transfer

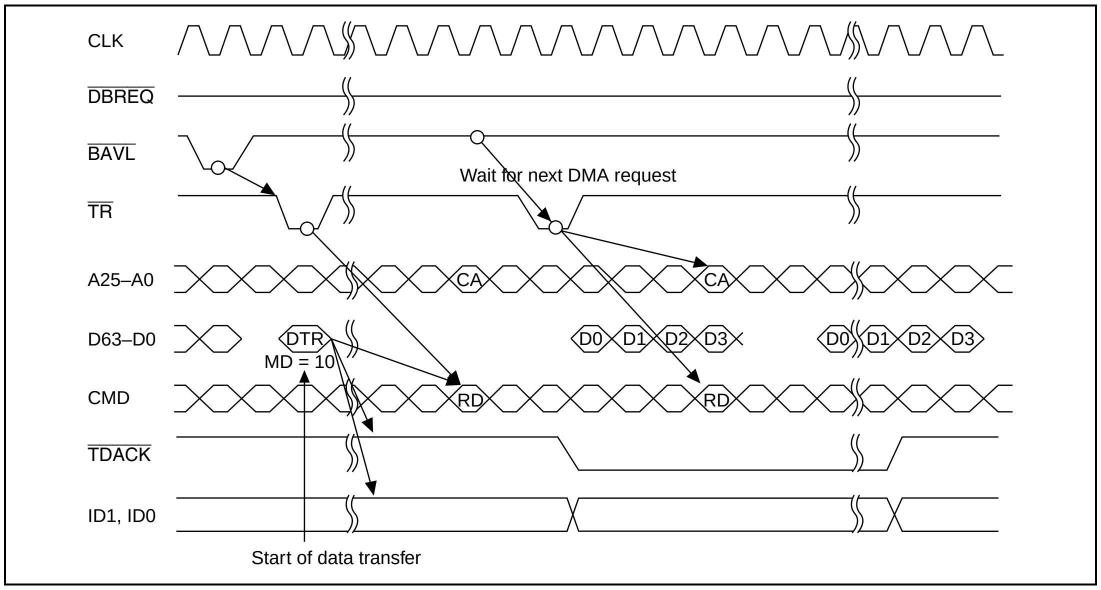


Figure G.18 Single Address Mode/Burst Mode/Level Detection/ External Bus → External Device Data Transfer

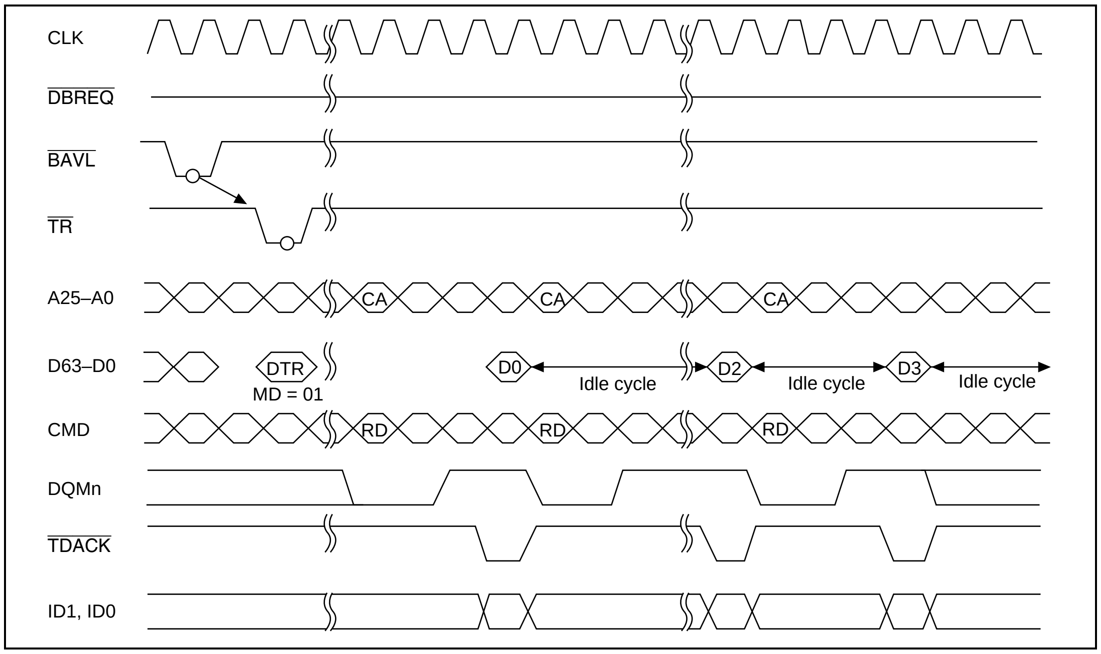


Figure G.19 Single Address Mode/Burst Mode/Edge Detection/Byte, Word, Longword, Quadword/External Bus → External Device Data Transfer

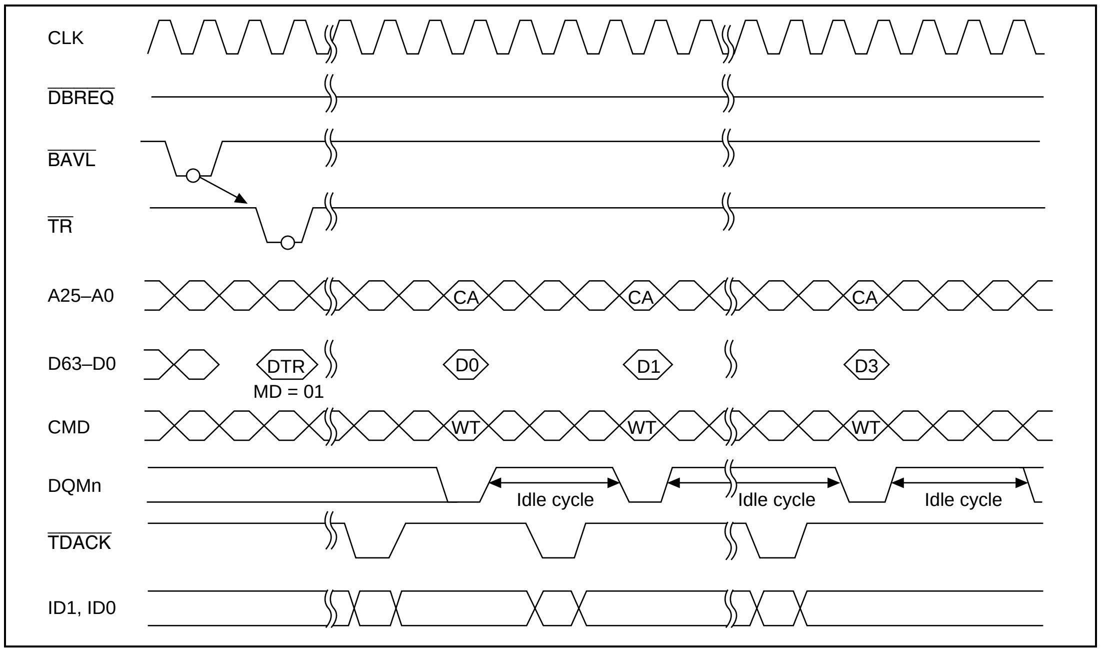


Figure G.20 Single Address Mode/Burst Mode/Edge Detection/Byte, Word, Longword, Quadword/External Device → External Bus Data Transfer

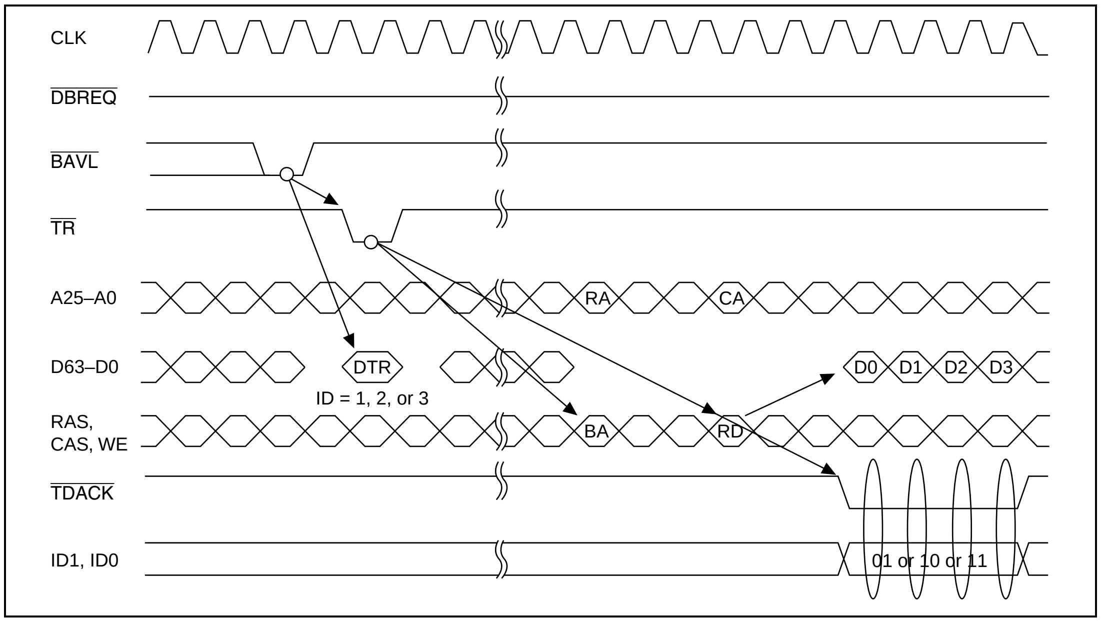


Figure G.21 Single Address Mode/Burst Mode/32-Byte Block Transfer/DMA Transfer Request to Channels 1–3 Using Data Bus

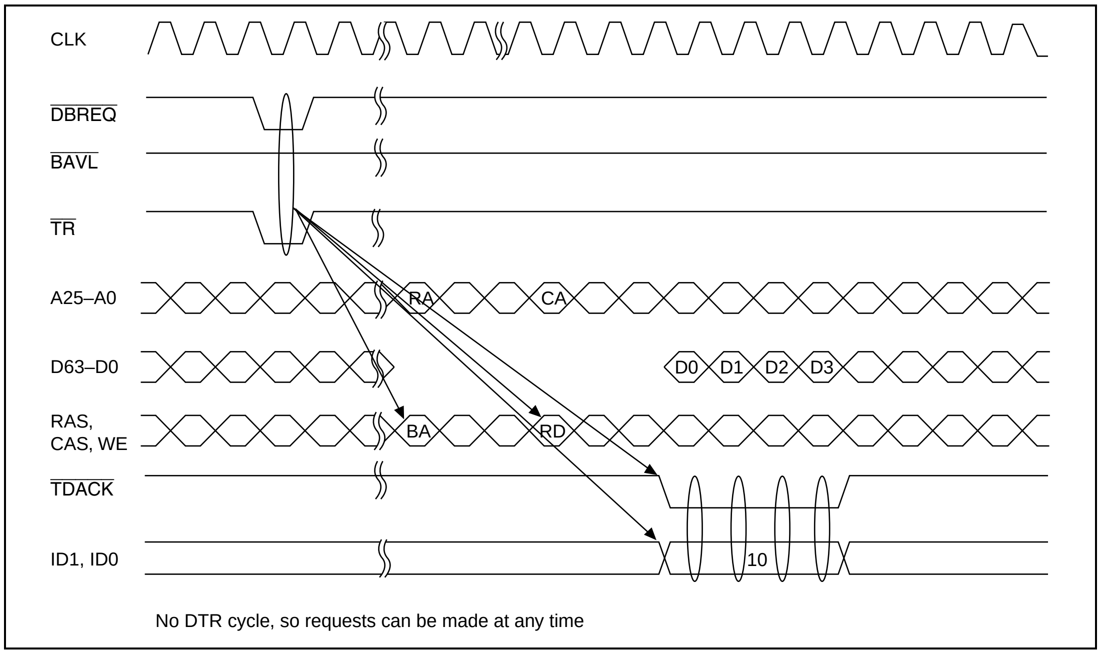


Figure G.22 Single Address Mode/Burst Mode/32-Byte Block Transfer/ External Bus → External Device Data Transfer/ Direct Data Transfer Request to Channel 2 without Using Data Bus

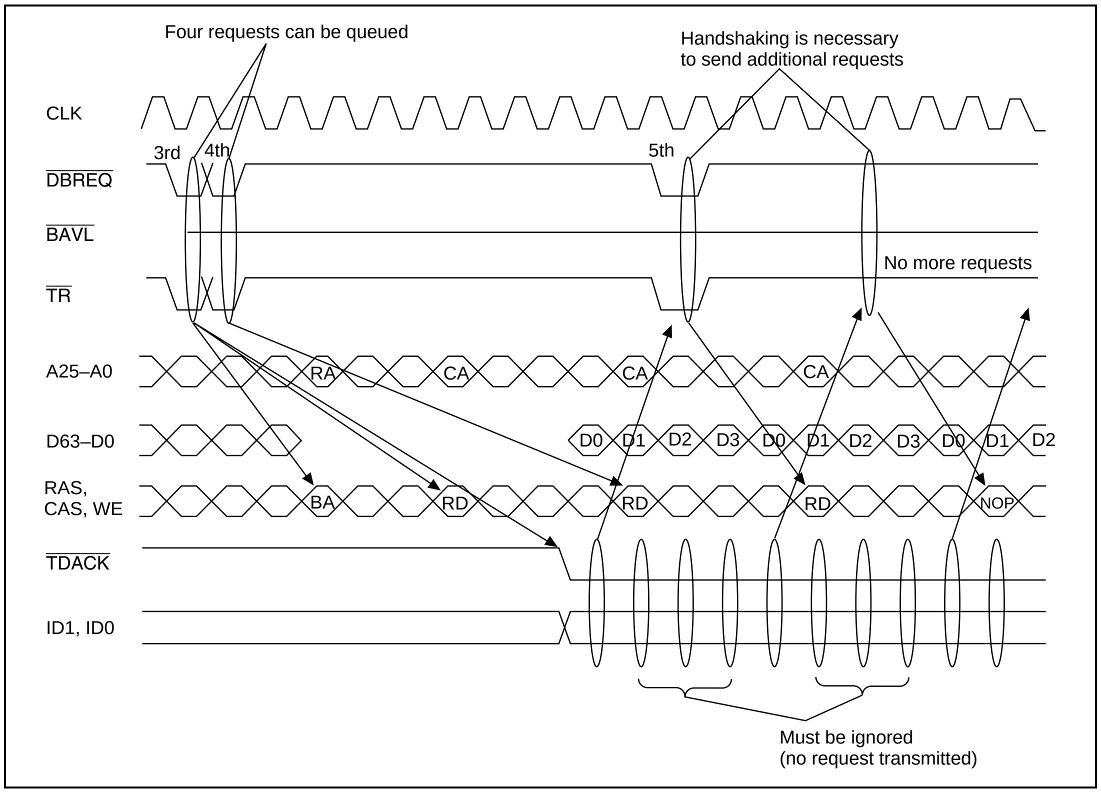


Figure G.23 Single Address Mode/Burst Mode/External Bus → External Device Data Transfer/Direct Data Transfer Request to Channel 2

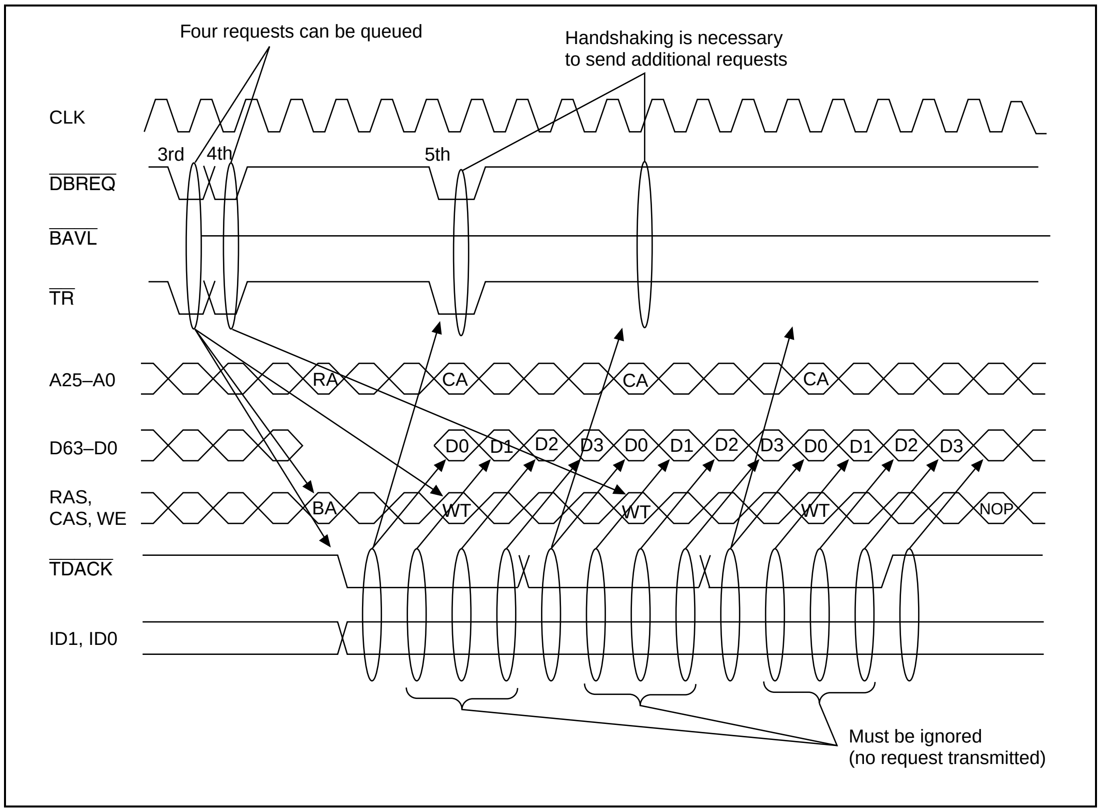


Figure G.24 Single Address Mode/Burst Mode/External Device → External Bus Data Transfer/Direct Data Transfer Request to Channel 2

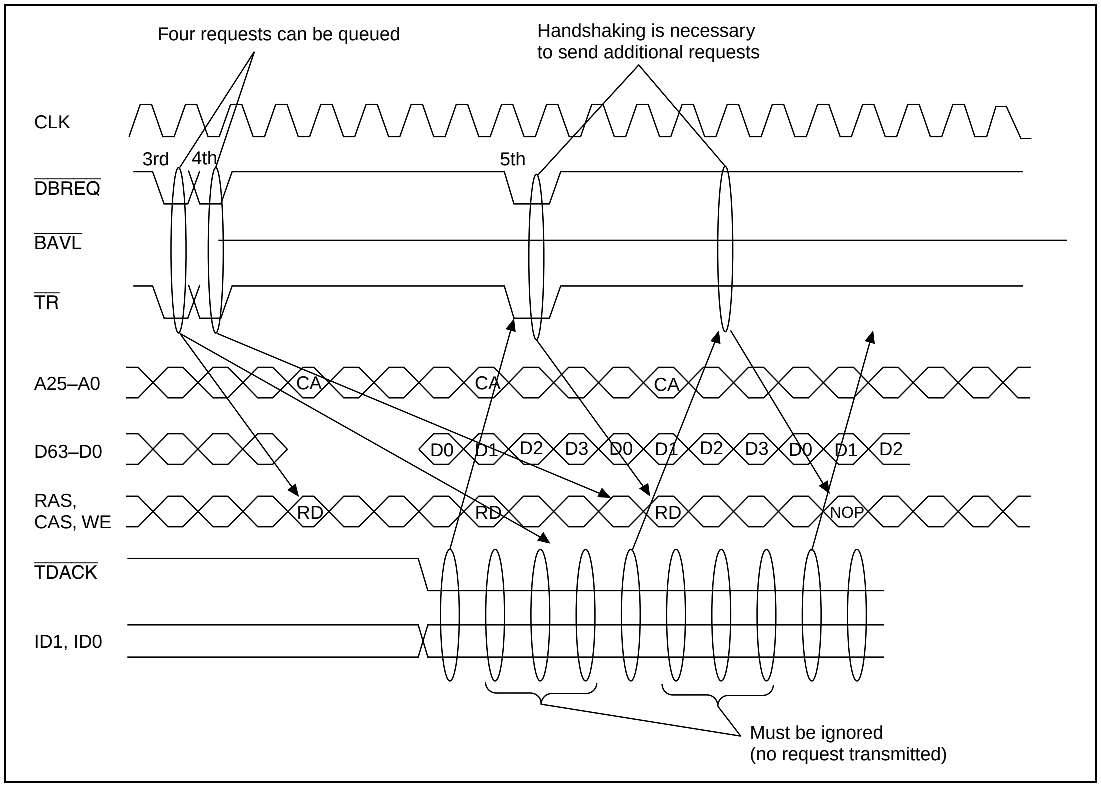


Figure G.25 Single Address Mode/Burst Mode/External Bus → External Device Data Transfer (Active Bank Address)/Direct Data Transfer Request to Channel 2

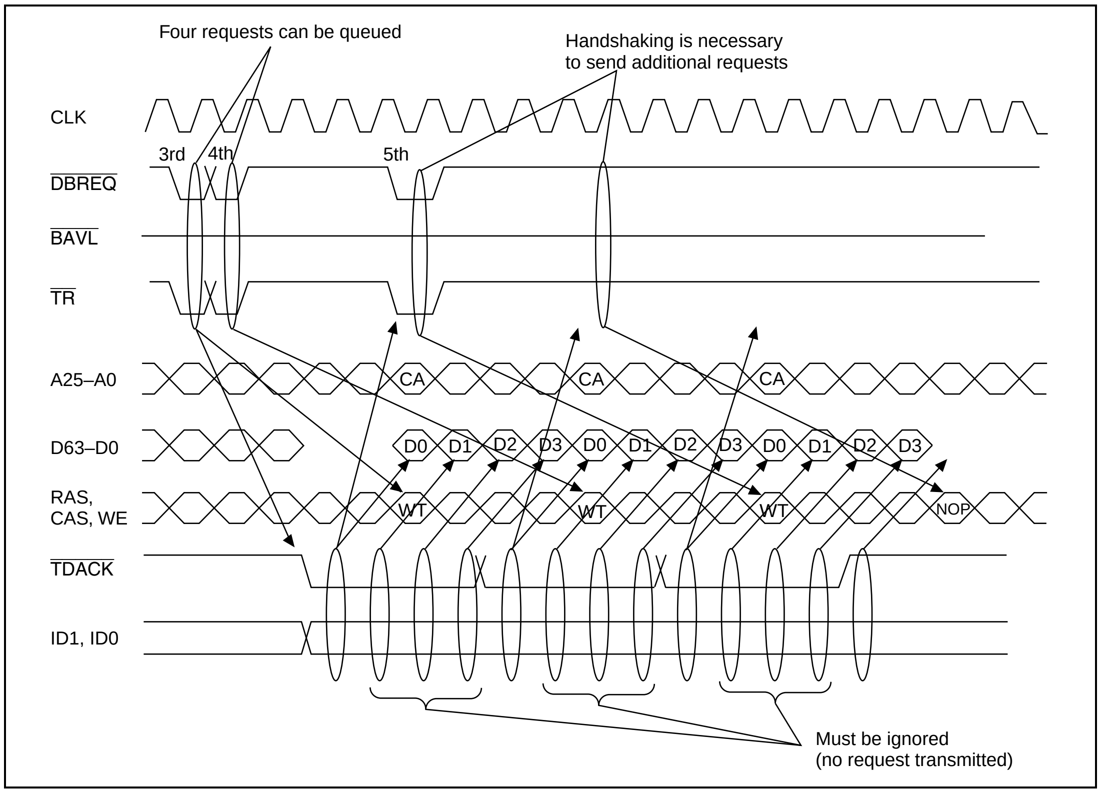


Figure G.26 Single Address Mode/Burst Mode/External Device → External Bus Data Transfer (Active Bank Address)/Direct Data Transfer Request to Channel 2