Hitachi SuperH™ RISC engine

SH7622

Hardware Manual

HITACHI

ADE-602-212A
Rev. 2.0
9/19/01
Hitachi, Ltd.

Hitachi semiconductor

Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi’s or any third party’s patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party’s rights, including intellectual property rights, in connection with use of the information contained in this document.
2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.
3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi’s sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.
4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.
5. This product is not designed to be radiation resistant.
6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.
7. Contact Hitachi’s sales office for any questions regarding this document or Hitachi semiconductor products.

Preface

The SH7622 is a microprocessor that integrates peripheral functions necessary for system configuration with a 32-bit internal architecture SH2-DSP CPU as its core.

The SH7622's on-chip peripheral functions include a DSP, cache memory, internal X/Y memory, an interrupt controller, timers, three serial communication interfaces, a USB function module, a user break controller (UBC), a bus state controller (BSC), and I/O ports, making it ideal for use as a microcomputer in electronic devices that require high speed together with low power consumption.

Intended Readership: This manual is intended for users undertaking the design of an application system using the SH7622. Readers using this manual require a basic knowledge of electrical circuits, logic circuits, and microcomputers.

Purpose: The purpose of this manual is to give users an understanding of the hardware functions and electrical characteristics of the SH7622. Details of execution instructions can be found in the SH-1, SH-2, SH-DSP Programming Manual, which should be read in conjunction with the present manual.

Using this Manual:

- For an overall understanding of the SH7622's functions
  Follow the Table of Contents. This manual is broadly divided into sections on the CPU, system control functions, peripheral functions, and electrical characteristics.
- For a detailed understanding of CPU functions
  Refer to the separate publication SH-1, SH-2, SH-DSP Programming Manual.

  Note on bit notation: Bits are shown in high-to-low order from left to right.

Related Material: The latest information is available at our Web Site. Please make sure that you have the most up-to-date information available.
http://www.hitachisemiconductor.com/

User's Manuals on the SH7622:

Manual Title	ADE No.
SH7622 Hardware Manual	This manual
SH-1, SH-2, SH-DSP Programming Manual	ADE-602-085

Users manuals for development tools:

Manual Title	ADE No.
C/C++ Complier, Assembler, Optimized Linkage Editor User's Manual	ADE-702-304
Simulator Debugger Users Manual	ADE-702-266
Hitachi Embedded Workshop Users Manual	ADE-702-275

Application Note:

Manual Title	ADE No.
C/C++ Complier	ADE-502-046

Contents

Section 9 User Break Controller 155

Section 10 Power-Down Modes 187

Section 18 Serial Communication Interface with FIFO (SCIF2) 441

Section 19 USB Function Module 495

Section 1 Overview and Pin Functions

1.1 SH7622 Features

The SH7622 is a RISC microprocessor with a 32-bit RISC type SuperH architecture CPU plus digital signal processing (DSP) extended functions as its core, and also including cache memory, on-chip X/Y memory, and an interrupt controller necessary for system configuration.

High-speed data transfer is provided by the on-chip DMAC (direct memory access controller), and external memory access support functions allow direct connection to various kinds of memory. The SH7622 also provided with powerful on-chip peripheral functions ideal for system configuration, including a USB function module and serial communication interface with large-capacity built-in FIFOs.

Powerful on-chip power management functions enable power consumption to be reduced even during high-speed operation. The SH7622 is ideally suited to electronic devices and other applications requiring high-speed operation together with low power consumption.

The features of the SH7622 are summarized in table 1.1.

Table 1.1 SH7622 Features

Item	Features
CPU	• Original Hitachi SuperH architecture • Object code level upward compatibility with SH-1, SH-2, SH-DSP • 32-bit internal data bus • General register file — Sixteen 32-bit general registers — Three 32-bit control registers — Four 32-bit system registers • RISC-type instruction set — Fixed 16-bit instruction length for excellent code efficiently — Load-store architecture — Delayed branch instructions — C-based instruction set • Instruction execution time: Basic instructions execute in one cycle • Address space: 4 Gbytes • Five-stage pipeline

Table 1.1 SH7622 Features (cont)

Item	Features
DSP	• Mix of 16-bit and 32-bit instructions • 32-/40-bit internal data bus • Multiplier, ALU, barrel shifter, register file • 16-bit × 16-bit → 32-bit 1-cycle multiplier • Large-capacity DSP data register file — Six 32-bit data registers — Two 40-bit data registers • Extended Harvard architecture for DSP data buses — Two data buses — One instruction bus • Maximum 4 parallel operations: ALU, multiply, two load/store • Two address units for generating addresses for two memory accesses • DSP data addressing modes: Increment/decrement • Zero-overhead repeat loop control • Conditional execution instructions
Clock pulse generator (CPG)	• Clock modes: Choice of external clock (EXTAL or CKIO) or crystal resonator for input clock • Three kinds of clock generated — CPU clock — Bus clock — Peripheral clock • Power-down modes — Standby mode — Module standby mode • Single-channel watchdog timer
Cache memory	• 8-kbyte cache, mixed instructions/data • 128 entries, 4-way set-associative, 16-byte block length • Write-back, write-through, LRU replacement algorithm • Single-stage write-back buffer

Table 1.1 SH7622 Features (cont)

Item	Features
X/Y memory	• Three independent read/write ports — 8-/16-/32-bit access from CPU — Maximum of two 16-bit accesses from DSP • 8-kbyte on-chip RAM for X and Y memory
Interrupt controller (INTC)	• Nine external interrupt pins (NMI, IRQ7 to IRQ0) • On-chip peripheral module interrupts: Priority level can be set for each module • Auto vector mode supported (no external vector mode) • Fixed vector numbers
User break controller (UBC)	• Two break channels • Address, data value, access type, and data size can all be set as break conditions • Supports sequential break function
Bus state controller (BSC)	• External memory space divided into six areas (area 0 and areas 2 to 6), each of up to 64 Mbytes, with the following parameters settable for each area: — Bus size (8, 16, or 32 bits) — Number of wait cycles (hardware wait function also supported) — Direct connection of SRAM, synchronous DRAM, and burst ROM possible by designating memory to be connected to each area — Chip select signals (CS0, CS2 to CS6) output for relevant areas • Synchronous DRAM refresh functions — Programmable refresh interval — Supports auto-refresh and self-refresh modes • Synchronous DRAM burst access function
User debug interface (H-UDI)	• E10A emulator support • Pin arrangement conforming to JTAG specification • Realtime branch trace • 1-kbyte on-chip RAM for high-speed emulation program execution
Timer unit (TMU)	• 3-channel auto-reload 32-bit timer • Input capture function (channel 2 only) • Choice of six counter input clocks

Table 1.1 SH7622 Features (cont)

Item	Features
Compare match timer 1 (CMT1)	• 16-bit counter • Choice of four counter input clocks • CPU interrupt request or DMAC transfer request generated by compare match
Serial communication interface 0 (SCIF0)	• Synchronous mode • Simultaneous transmission/reception (full-duplex) capability, clock pin used for both transmission and reception • DMA transfer capability • 128-byte transmit FIFO, 384-byte receive FIFO
Serial communication interface 1 (SCIF1)	• Synchronous mode • Simultaneous transmission/reception (full-duplex) capability, clock pin used for both transmission and reception • DMAC transfer capability • 128-byte transmit and receive FIFOs
Serial communication interface 2 (SCIF2)	• Choice of synchronous mode or asynchronous mode • 16-byte transmit and receive FIFOs • DMA transfer capability
DMA controller (DMAC)	• Four channels • Burst mode and cycle steal mode • External request capability (channels 0 and 1 only)
I/O ports	• Dual-function input/output ports can be switched between input and output bit by bit

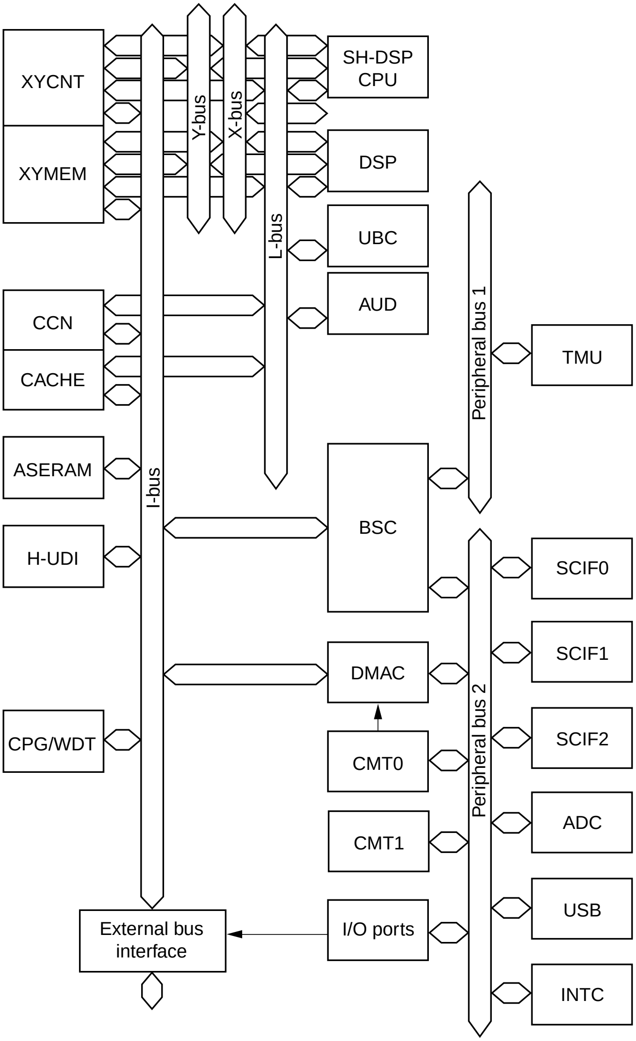
Table 1.1 SH7622 Features (cont)

Item	Features
USB function module	• Conforms to USB 1.0 (Can be connected to a Philips PDIUSBP11 Series transceiver or compatible product (when using a compatible product, carry out evaluation and investigation with the manufacturer supplying the transceiver beforehand), Vcc = 3.0 V to 3.6 V • Supports 12 Mbps full-speed transfer • Supports control (endpoint 0), bulk transfer (endpoints 1 and 2), and interrupt transfer (endpoint 3) • USB standard commands supported; class and vendor commands processed by software • Built-in endpoint FIFO buffers (128 bytes per endpoint) • Supports DMA transfer by on-chip DMAC • Module internal clock: 48 MHz
A/D converter	• 10 bits ±4 LSB, four channels • Input range: 0 to AVcc (max. 3.6 V)
Power supply voltage	• I/O: 3.0 V to 3.6 V, internal: 1.75 to 2.05 V

	Product Name	Voltage	Operating Frequency	Product Code	Package
Product lineup	SH7622	3.3 V	80 MHz	HD6417622FL80	216-pin plastic LQFP (FP-216)
				HD6417622BP80	208-pin TFBGA (TBP-208A)
				HD6417622F80	208-pin plastic QFP (FP-208C)
			100 MHz	HD6417622FL100	216-pin plastic LQFP (FP-216)
				HD6417622BP100	208-pin TFBGA (TBP-208A)
				HD6417622F80	208-pin plastic QFP (FP-208C)

1.2 Block Diagram

Figure 1.1 shows an internal block diagram of the SH7622.

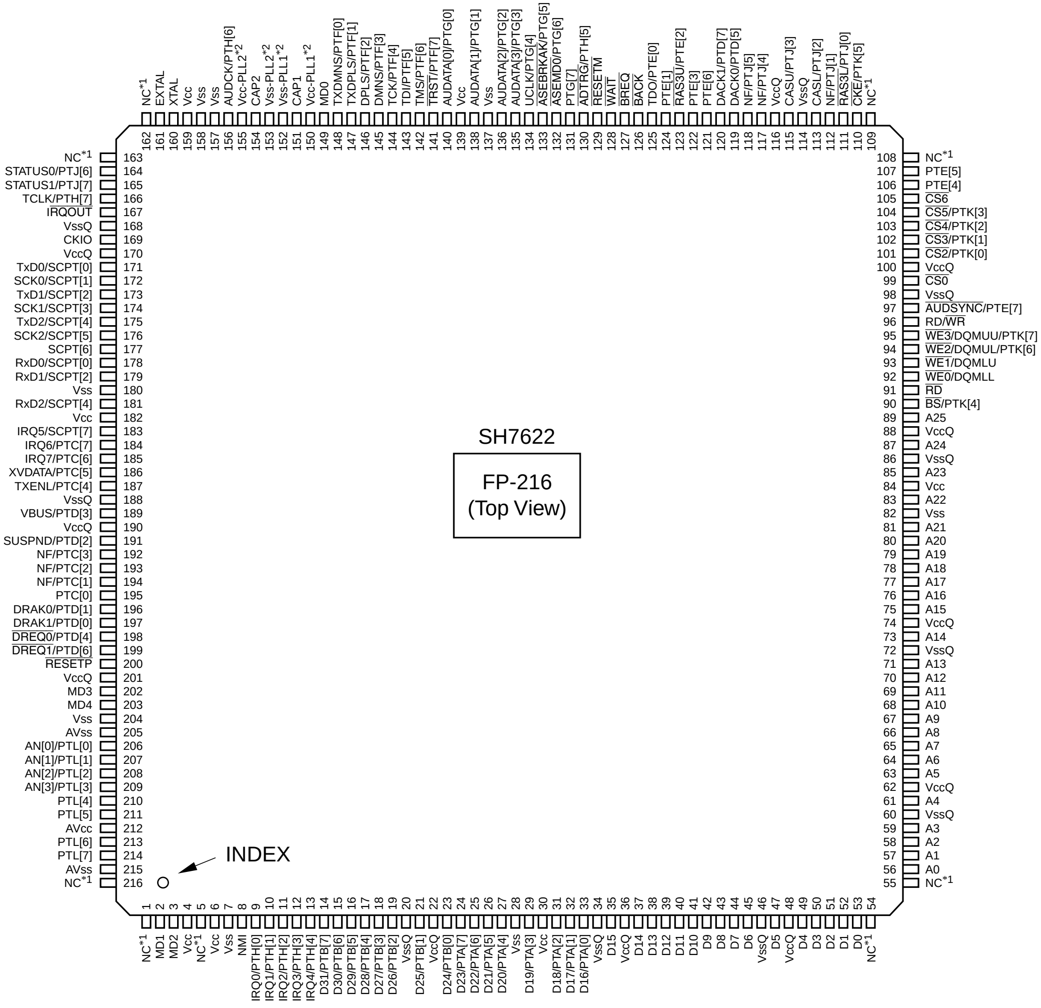


ADC:	A/D converter	CPG/WDT:	Clock pulse generator/watchdog timer
ASERAM:	ASE memory	CPU:	Central processing unit
AUD:	Advanced user debugger	DMAC:	Direct memory access controller
BSC:	Bus state controller	INTC:	Interrupt controller
CACHE:	Cache memory	SCIF:	Serial communication interface (with FIFO)
CCN:	Cache memory controller	TMU:	Timer unit
CMT0:	Compare match timer 0	UBC:	User break controller
CMT1:	Compare match timer 1	H-UDI:	Hitachi user debug interface

Figure 1.1 Block Diagram of SH7622

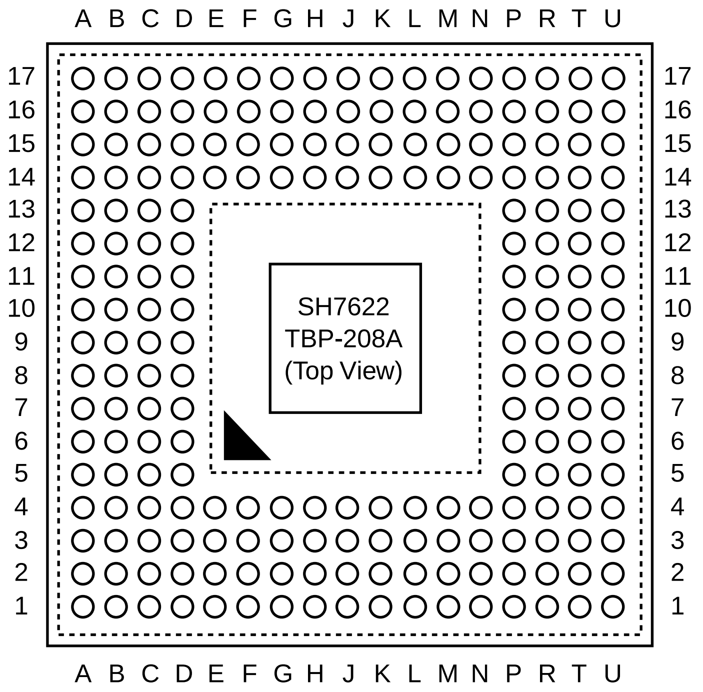
1.3 Pin Description

1.3.1 Pin Arrangement



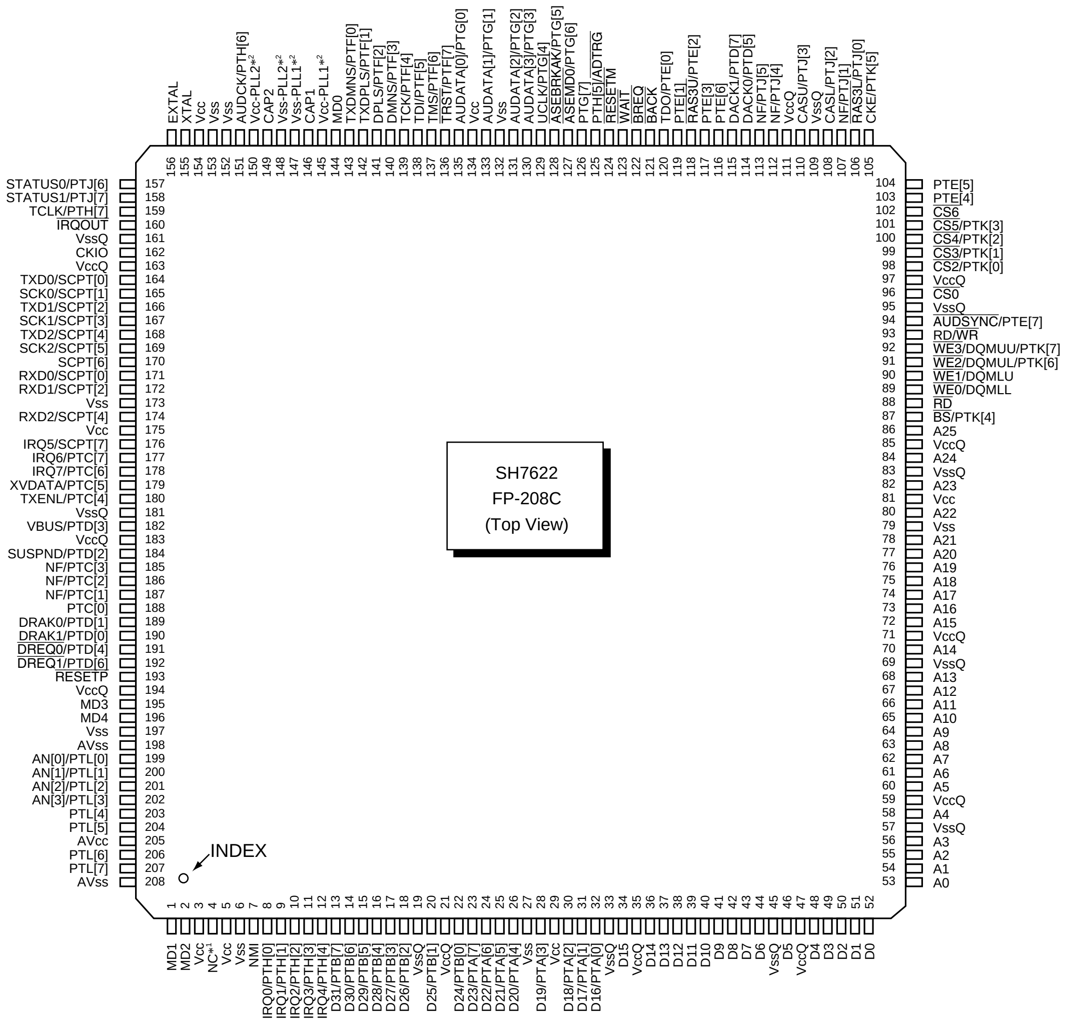
Notes: *1 NC pins must be connected to ground, except for the No.5 NC pin, which should be left open.
*2 Must be connected to the power supply when the on-chip PLL is not used.

Figure 1.2 Pin Arrangement (FP-216)



Note: The area within dotted lines shows a cutaway view of the pins.

Figure 1.3 Pin Arrangement (TBP-208A)



Notes: *1 No. 4 NC pin should be left open.
*2 Must be connected to the power supply when the on-chip PLL is not used.

Figure 1.4 Pin Arrangement (FP-208C)

1.3.2 Pin Functions

Table 1.2 summarizes the pin functions.

Table 1.2 SH7622 Pin Functions

Pin No.					
FP-208C	**FP-216**	**TBP-208A**	**Pin Name**	**I/O**	**Description**
—	1	—	NC*[1], *[4]	—	—
1	2	A1	MD1	I	Clock mode setting
2	3	B1	MD2	I	Clock mode setting
3	4	C3	Vcc*[3]	—	Power supply (1.9 V)
4	5	C2	NC*[1], *[4]	O	—
5	6	C1	Vcc	—	Power supply (1.9 V)
6	7	D3	Vss	—	Power supply (0 V)
7	8	D2	NMI	I	Nonmaskable interrupt request
8	9	D1	IRQ0/PTH[0]	I	External interrupt request/input port H
9	10	E4	IRQ1/PTH[1]	I	External interrupt request/input port H
10	11	E3	IRQ2/PTH[2]	I	External interrupt request/input port H
11	12	E2	IRQ3/PTH[3]	I	External interrupt request/input port H
12	13	E1	IRQ4/PTH[4]	I	External interrupt request/input port H
13	14	F4	D31/PTB[7]	IO	Data bus / input/output port B
14	15	F3	D30/PTB[6]	IO	Data bus / input/output port B
15	16	F2	D29/PTB[5]	IO	Data bus / input/output port B
16	17	F1	D28/PTB[4]	IO	Data bus / input/output port B
17	18	G4	D27/PTB[3]	IO	Data bus / input/output port B
18	19	G3	D26/PTB[2]	IO	Data bus / input/output port B
19	20	G2	VssQ*[3]	—	Input/output power supply (0 V)
20	21	G1	D25/PTB[1]	IO	Data bus / input/output port B
21	22	H4	VccQ*[3]	—	Input/output power supply (3.3 V)
22	23	H3	D24/PTB[0]	IO	Data bus / input/output port B
23	24	H2	D23/PTA[7]	IO	Data bus / input/output port A
24	25	H1	D22/PTA[6]	IO	Data bus / input/output port A
25	26	J4	D21/PTA[5]	IO	Data bus / input/output port A
26	27	J2	D20/PTA[4]	IO	Data bus / input/output port A

Table 1.2 SH7622 Pin Functions (cont)

Pin No.					
FP-208C	FP-216	TBP-208A	Pin Name	I/O	Description
27	28	J1	Vss*3	—	Power supply (0 V)
28	29	J3	D19/PTA[3]	IO	Data bus / input/output port A
29	30	K1	Vcc*3	—	Power supply (1.9 V)
30	31	K2	D18/PTA[2]	IO	Data bus / input/output port A
31	32	K3	D17/PTA[1]	IO	Data bus / input/output port A
32	33	K4	D16/PTA[0]	IO	Data bus / input/output port A
33	34	L1	VssQ*3	—	Input/output power supply (0 V)
34	35	L2	D15	IO	Data bus
35	36	L3	VccQ*3	—	Input/output power supply (3.3 V)
36	37	L4	D14	IO	Data bus
37	38	M1	D13	IO	Data bus
38	39	M2	D12	IO	Data bus
39	40	M3	D11	IO	Data bus
40	41	M4	D10	IO	Data bus
41	42	N1	D9	IO	Data bus
42	43	N2	D8	IO	Data bus
43	44	N3	D7	IO	Data bus
44	45	N4	D6	IO	Data bus
45	46	P1	VssQ*3	—	Input/output power supply (0 V)
46	47	P2	D5	IO	Data bus
47	48	P3	VccQ*3	—	Input/output power supply (3.3 V)
48	49	R1	D4	IO	Data bus
49	50	R2	D3	IO	Data bus
50	51	R4	D2	IO	Data bus
51	52	T1	D1	IO	Data bus
52	53	T2	D0	IO	Data bus
—	54	—	NC*1, *4	—	—
—	55	—	NC*1, *4	—	—
53	56	U1	A0	O	Address bus
54	57	U2	A1	O	Address bus

Table 1.2 SH7622 Pin Functions (cont)

Pin No.					
FP-208C	FP-216	TBP-208A	Pin Name	I/O	Description
55	58	R3	A2	O	Address bus
56	59	T3	A3	O	Address bus
57	60	U3	VssQ*[3]	—	Input/output power supply (0 V)
58	61	R4	A4	O	Address bus
59	62	T4	VccQ*[3]	—	Input/output power supply (3.3 V)
60	63	U4	A5	O	Address bus
61	64	P5	A6	O	Address bus
62	65	R5	A7	O	Address bus
63	66	T5	A8	O	Address bus
64	67	U5	A9	O	Address bus
65	68	P6	A10	O	Address bus
66	69	R6	A11	O	Address bus
67	70	T6	A12	O	Address bus
68	71	U6	A13	O	Address bus
69	72	P7	VssQ*[3]	—	Input/output power supply (0 V)
70	73	R7	A14	O	Address bus
71	74	T7	VccQ*[3]	—	Input/output power supply (3.3 V)
72	75	U7	A15	O	Address bus
73	76	P8	A16	O	Address bus
74	77	R8	A17	O	Address bus
75	78	T8	A18	O	Address bus
76	79	U8	A19	O	Address bus
77	80	P9	A20	O	Address bus
78	81	T9	A21	O	Address bus
79	82	U9	Vss*[3]	—	Power supply (0 V)
80	83	R9	A22	O	Address bus
81	84	U10	Vcc*[3]	—	Power supply (1.9 V)
82	85	T10	A23	O	Address bus
83	86	R10	VssQ*[3]	—	Input/output power supply (0 V)
84	87	P10	A24	O	Address bus

Table 1.2 SH7622 Pin Functions (cont)

Pin No.					
FP-208C	**FP-216**	**TBP-208A**	**Pin Name**	**I/O**	**Description**
85	88	U11	VccQ*^{3}	—	Input/output power supply (3.3 V)
86	89	T11	A25	O	Address bus
87	90	R11	$\overline{\text{BS}}$/PTK[4]	O/IO	Bus cycle start signal / input/output port K
88	91	P11	$\overline{\text{RD}}$	O	Read strobe
89	92	U12	$\overline{\text{WE0}}$/DQMLL	O	D7–D0 select signal / DQM (SDRAM)
90	93	T12	$\overline{\text{WE1}}$/DQMLU	O	D15–D8 select signal / DQM (SDRAM)
91	94	R12	$\overline{\text{WE2}}$/DQMUL/ PTK[6]	O/IO	D23–D16 select signal / DQM (SDRAM) / input/output port K
92	95	P12	$\overline{\text{WE3}}$/DQMUU/ PTK[7]	O/IO	D31–D24 select signal / DQM (SDRAM) / input/output port K
93	96	U13	RD/$\overline{\text{WR}}$	O	Read/write
94	97	T13	$\overline{\text{AUDSYNC}}$/PTE[7]	O/IO	AUD synchronization / input/output port E
95	98	R13	VssQ*^{3}	—	Input/output power supply (0 V)
96	99	P13	$\overline{\text{CS0}}$	O	Chip select 0
97	100	U14	VccQ*^{3}	—	Input/output power supply (3.3 V)
98	101	T14	$\overline{\text{CS2}}$/PTK[0]	O/IO	Chip select 2 / input/output port K
99	102	R14	$\overline{\text{CS3}}$/PTK[1]	O/IO	Chip select 3 / input/output port K
100	103	U15	$\overline{\text{CS4}}$/PTK[2]	O/IO	Chip select 4 / input/output port K
101	104	T15	$\overline{\text{CS5}}$/PTK[3]	O/IO	Chip select 5 / input/output port K
102	105	P14	$\overline{\text{CS6}}$	O	Chip select 6
103	106	U16	PTE[4]	IO	Input/output port E
104	107	T16	PTE[5]	IO	Input/output port E
—	108	—	NC*^{1,} *^{4}	—	—
—	109	—	NC*^{1,} *^{4}	—	—
105	110	U17	CKE/PTK[5]	O/IO	CK enable (SDRAM) / input/output port K
106	111	T17	$\overline{\text{RAS3L}}$/PTJ[0]	O/IO	Lower 32 MB address (SDRAM) RAS / input/output port J
107	112	R15	NF*^{6}/PTJ[1]	O	No function/output port J
108	113	R16	CASL/PTJ[2]	O/IO	Lower 32 MB address (SDRAM) CAS / input/output port J
109	114	R17	VssQ*^{3}	—	Input/output power supply (0 V)

Table 1.2 SH7622 Pin Functions (cont)

Pin No.					
FP-208C	FP-216	TBP-208A	Pin Name	I/O	Description
110	115	P15	CASU/PTJ[3]	O/IO	Upper 32 MB address (SDRAM) CAS / input/output port J
111	116	P16	VccQ*3	—	Input/output power supply (3.3 V)
112	117	P17	NF*6/PTJ[4]	O	No function/output port J
113	118	N14	NF*6/PTJ[5]	O	No function/output port J
114	119	N15	DACK0/PTD[5]	O/IO	DMA acknowledge 0 / input/output port D
115	120	N16	DACK1/PTD[7]	O/IO	DMA acknowledge 1 / input/output port D
116	121	N17	PTE[6]	IO	Input/output port E
117	122	M14	PTE[3]	IO	Input/output port E
118	123	M15	$\overline{\text{RAS3U}}$/PTE[2]	O/IO	Upper 32 MB address (area 3 DRAM, SDRAM) RAS / input/output port E
119	124	M16	PTE[1]	IO	Input/output port E
120	125	M17	TDO/PTE[0]	I/O	Test data output / input/output port E
121	126	L14	$\overline{\text{BACK}}$	O	Bus acknowledge
122	127	L15	$\overline{\text{BREQ}}$	I	Bus request
123	128	L16	$\overline{\text{WAIT}}$	I	Hardware wait request
124	129	L17	$\overline{\text{RESETM}}$	I	Manual reset request
125	130	K14	$\overline{\text{ADTRG}}$/PTH[5]	I	Analog trigger / input port H
126	131	K15	PTG[7]	I	Input port G
127	132	K16	$\overline{\text{ASEMD0}}$/PTG[6]	I	ASE mode / input port G
128	133	K17	$\overline{\text{ASEBRKAK}}$/PTG[5]	O/I	ASE break acknowledge / input port G
129	134	J14	UCLK/PTG[4]	I	USB external input clock / input port G
130	135	J16	AUDATA[3]/PTG[3]	O/I	AUD data / input port G
131	136	J17	AUDATA[2]/PTG[2]	O/I	AUD data / input port G
132	137	J15	Vss*3	—	Power supply (0 V)
133	138	H17	AUDATA[1]/PTG[1]	O/I	AUD data / input port G
134	139	H16	Vcc*3	—	Power supply (1.9 V)
135	140	H15	AUDATA[0]/PTG[0]	O/I	AUD data / input port G
136	141	H14	$\overline{\text{TRST}}$/PTF[7]	I	Test reset / input port F
137	142	G17	TMS/PTF[6]	I	Test mode switch/ input port F

Table 1.2 SH7622 Pin Functions (cont)

Pin No.					
FP-208C	**FP-216**	**TBP-208A**	**Pin Name**	**I/O**	**Description**
138	143	G16	TDI/PTF[5]	I	Test data input/ input port F
139	144	G15	TCK/PTF[4]	I	Test clock/ input port F
140	145	G14	DMNS/ PTF[3]	I	D– input from USB receiver / input port F
141	146	F17	DPLS/ PTF[2]	I	D+ input from USB receiver / input port F
142	147	F16	TXDPLS/ PTF[1]	O/I	USB D+ transmit output / input port F
143	148	F15	TXDMNS/ PTF[0]	O/I	USB D– transmit output / input port F
144	149	F14	MD0	I	Clock mode setting
145	150	E17	Vcc-PLL1*2	—	PLL1 power supply (1.9 V)
146	151	E16	CAP1	—	PLL1 external capacitance pin
147	152	E15	Vss-PLL1*2	—	PLL1 power supply (0 V)
148	153	E14	Vss-PLL2*2	—	PLL2 power supply (0 V)
149	154	D17	CAP2	—	PLL2 external capacitance pin
150	155	D16	Vcc-PLL2*2	—	PLL2 power supply (1.9 V)
151	156	D15	AUDCK/PTH[6]	I	AUD clock / input port H
152	157	C17	Vss*3	—	Power supply (0 V)
153	158	C16	Vss*3	—	Power supply (0 V)
154	159	D14	Vcc*3	—	Power supply (1.9 V)
155	160	B17	XTAL	O	Clock pulse generator pin
156	161	B16	EXTAL	I	External clock / crystal oscillator pin
—	162	—	NC*1, *4	—	—
—	163	—	NC*1, *4	—	—
157	164	A17	STATUS0/PTJ[6]	O/IO	Processor status / input/output port J
158	165	A16	STATUS1/PTJ[7]	O/IO	Processor status / input/output port J
159	166	C15	TCLK/PTH[7]	IO	TMU or RTC clock input/output / input/output port H
160	167	B15	$\overline{\text{IRQOUT}}$	O	Interrupt request notification
161	168	A15	VssQ*3	—	Input/output power supply (0 V)
162	169	C14	CKIO	O	System clock output
163	170	B14	VccQ*3	—	Input/output power supply (3.3 V)
164	171	A14	TxD0/SCPT[0]	O	Transmit data 0 / SCI output port

Table 1.2 SH7622 Pin Functions (cont)

Pin No.					
FP-208C	**FP-216**	**TBP-208A**	**Pin Name**	**I/O**	**Description**
165	172	D13	SCK0/SCPT[1]	IO	Serial clock 0 / SCI input/output port
166	173	C13	TxD1/SCPT[2]	O	Transmit data 1 / SCI output port
167	174	B13	SCK1/SCPT[3]	IO	Serial clock 1 / SCI input/output port
168	175	A13	TxD2/SCPT[4]	O	Transmit data 2 / SCI output port
169	176	D12	SCK2/SCPT[5]	IO	Serial clock 2 / SCI input/output port
170	177	C12	SCPT[6]	IO	SCI input/output port
171	178	B12	RxD0/SCPT[0]	I	Receive data 0 / SCI input port
172	179	A12	RxD1/SCPT[2]	I	Receive data 1 / SCI input port
173	180	D11	Vss*[3]	—	Power supply (0 V)
174	181	C11	RxD2/SCPT[4]	I	Receive data 2 / SCI input port
175	182	B11	Vcc*[3]	—	Power supply (1.9 V)
176	183	A11	IRQ5/SCPT[7]	I	External interrupt request / SCI input port
177	184	D10	IRQ6/PTC[7]	I/IO	External interrupt request / input/output port C
178	185	C10	IRQ7/PTC[6]	I/IO	External interrupt request / input/output port C
179	186	B10	XVDATA/PTC[5]	I/IO	USB differential receive signal input / input/output port C
180	187	A10	TXENL/PTC[4]	O/IO	USB output enable / input/output port C
181	188	D9	VssQ*[3]	—	Input/output power supply (0 V)
182	189	B9	VBUS/PTD[3]	I/IO	USB power supply detection / input/output port D
183	190	A9	VccQ*[3]	—	Input/output power supply (3.3 V)
184	191	C9	SUSPND/PTD[2]	O/IO	USB suspend / input/output port D
185	192	A8	NF*[6]/PTC[3]	O	No function / output port C
186	193	B8	NF*[6]/PTC[2]	O	No function / output port C
187	194	C8	NF*[6]/PTC[1]	I	No function / input port C
188	195	D8	PTC[0]	O	Output port C
189	196	A7	DRAK0/PTD[1]	O/IO	DMA request acknowledge / input/output port D
190	197	B7	DRAK1/PTD[0]	O/IO	DMA request acknowledge / input/output port D

Table 1.2 SH7622 Pin Functions (cont)

Pin No.					
FP-208C	FP-216	TBP-208A	Pin Name	I/O	Description
191	198	C7	$\overline{\text{DREQ0}}$/PTD[4]	I	DMA request / input/output port D
192	199	D7	$\overline{\text{DREQ1}}$/PTD[6]	I	DMA request / input/output port D
193	200	A6	$\overline{\text{RESETP}}$	I	Power-on reset request
194	201	B6	VccQ*3	—	Input/output power supply (3.3 V)
195	202	C6	MD3	I	Area 0 bus width setting
196	203	D6	MD4	I	Area 0 bus width setting
197	204	A5	Vss*3	—	Power supply (0 V)
198	205	B5	AVss*3	—	Analog power supply (0 V)
199	206	C5	AN[0]/PTL[0]	I	A/D converter input / input port L
200	207	D5	AN[1]/PTL[1]	I	A/D converter input / input port L
201	208	A4	AN[2]/PTL[2]	I	A/D converter input / input port L
202	209	B4	AN[3]/PTL[3]	I	A/D converter input / input port L
203	210	C4	PTL[4]	I	Input port L
204	211	A3	PTL[5]	I	Input port L
205	212	B3	AVcc*3	—	Analog power supply (3.3 V)
206	213	D4	PTL[6]	I	Input port L
207	214	A2	PTL[7]	I	Input port L
208	215	B2	AVss*3	—	Analog power supply (0 V)
—	216	—	NC*1,*4	—	—

Notes: *1 NC pins must be connected to ground, except for the No. 5 (FP-216) and No. 4 (FP-208C), which should be left open.

*2 Must be connected to the power supply when the on-chip PLL is not used.

*3 All Vcc/Vss/VccQ/VssQ/AVcc/Avss pins must be connected to the system power supply (Power must be supplied constantly).

*4 Except for pin No.5 on the FP-216, NC pins are not connected internally.

*5 The system design must ensure that noise is not introduced onto the Vss and VssQ pins.

*6 NF pins should be left open when not used as output ports.
An exception is the PTC[1] NF pin, for which a pull-down connection should be made.

Section 2 CPU

2.1 Register Configuration

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

The SH7622 is upwardly compatible with the SH-1, SH-2 on the object code level. For this reason, several registers have been added to the previous SuperH microcontroller registers. The added registers are the three control registers: repeat start register (RS), repeat end register (RE), and modulo register (MOD) and the eight system registers: DSP status register (DSR), and A0, A1, X0, X1, Y0, and Y1 among the DSP data registers.

The general registers are used in the same manner as the SH-1, SH-2 with regard to SuperH microcontroller-type instructions. With regard to DSP type instructions, they are used as address and index registers for accessing memory.

2.1.1 General Registers

There are 16 general registers (Rn) numbered R0–R15, which are 32 bits in length. General registers are used for data processing and address calculation.

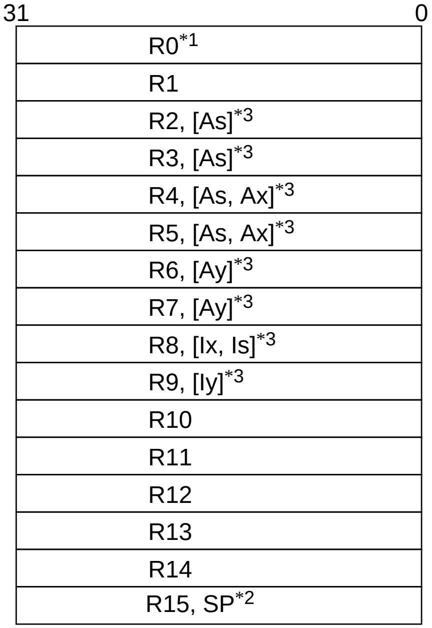
With SuperH microcomputer type instructions, R0 is also used as an index register. Several instructions are limited to use of R0 only. R15 is used as the hardware stack pointer (SP). Saving and recovering the status register (SR) and program counter (PC) in exception processing is accomplished by referencing the stack using R15.

With DSP type instructions, eight of the 16 general registers are used for the addressing of X, Y data memory and data memory (single data) using the L bus.

R4, R5 are used as an X address register (Ax) for X memory accesses, and R8 is used as an X index register (Ix). R6, R7 are used as a Y address register (Ay) for Y memory accesses, and R9 is used as a Y index register (Iy). R2, R3, R4, R5 are used as a single data address register (As) for accessing single data using the L bus, and R8 is used as a single data index register (Is).

DSP type instructions can simultaneously access X and Y data memory. There are two groups of address pointers for designating X and Y data memory addresses.

Figure 2.1 shows the general registers.



Notes: *1 R0 also functions as an index register in the indirect indexed register addressing mode and indirect indexed GBR addressing mode. In some instructions, only the R0 functions as a source register or destination register.

*2 R15 functions as a hardware stack pointer (SP) during exception processing.

*3 Used as memory address registers, memory index registers with DSP type instructions.

Figure 2.1 General Register Configuration

With the assembler, symbol names are used for R2, R3 ... R9. If it is wished to use a name that makes clear the role of a register for DSP type instructions, a different register name (alias) can be used. This is written in the following manner for the assembler.

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

The name Ix is an alias for R8. The other aliases are assigned as follows:

```
Ax0:    .REG (R4)
Ax1:    .REG (R5)
Ix:     .REG (R8)
Ay0:    .REG (R6)
Ay1:    .REG (R7)
Iy:     .REG (R9)
As0:    .REG (R4)  ; defined when an alias is required for single data transfer
As1:    .REG (R5)  ; defined when an alias is required for single data transfer
As2:    .REG (R2)  ; defined when an alias is required for single data transfer
As3:    .REG (R3)  ; defined when an alias is required for single data transfer
Is:     .REG (R8)  ; defined when an alias is required for single data transfer
```

2.1.2 Control Registers

The six 32-bit control registers consist of the status register (SR), repeat start register (RS), repeat end register (RE), global base register (GBR), vector base register (VBR), and modulo register (MOD).

The SR register indicates processing states.

The GBR register functions as a base address for the indirect GBR addressing mode, and is used for such as on-chip peripheral module register data transfers.

The VBR register functions as the base address of the exception processing vector area (including interrupts).

The RS and RE registers are used for program repeat (loop) control. The repeat count is designated in the SR register repeat counter (RC), the repeat start address in the RS register, and the repeat end address in the RE register. However, note that the address values stored in the RS and RE registers are not necessarily always the same as the physical start and end address values of the repeat.

The MOD register is used for modulo addressing to buffer the repeat data. The modulo addressing designation is made by DMX or DMY, the modulo end address (ME) is designated in the upper 16 bits of the MOD register, and the modulo start address (MS) is designated in the lower 16 bits. Note that the DMX and DMY bits cannot simultaneously designate modulo addressing. Modulo addressing is possible with X and Y data transfer instructions (MOVX, MOVY). It is not possible with single data transfer instructions (MOVS).

Figure 2.2 shows the control registers. Table 2.1 indicates the SR register bits.

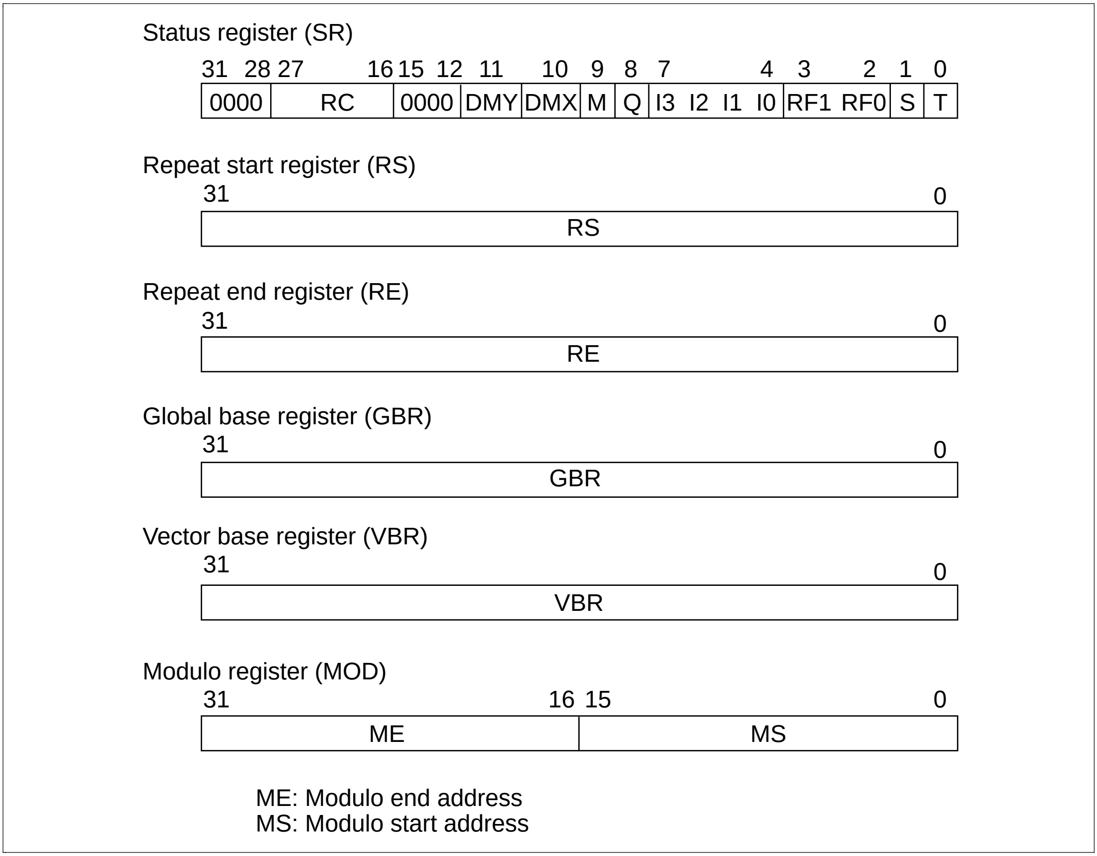


Figure 2.2 Control Register Configuration

Table 2.1 SR Register Bits

Bit	Name (Abbreviation)	Function
27–16	Repeat counter (RC)	Designate the repeat count (2–4095) for repeat (loop) control
11	Y pointer usage modulo addressing designation (DMY)	1: modulo addressing mode becomes valid for Y memory address pointer, Ay (R6, R7)
10	X pointer usage modulo addressing designation (DMX)	1: modulo addressing mode becomes valid for X memory address pointer, Ax (R4, R5)
9	M bit	Used by the DIV0S/U, DIV1 instructions
8	Q bit	Used by the DIV0S/U, DIV1 instructions
7–4	Interrupt request mask (I3–I0)	Indicate the receive level of an interrupt request (0 to 15)
3–2	Repeat flags (RF1, RF0)	Used in zero overhead repeat (loop) control. Set as below for an SETRC instruction For 1 step repeat 00 RE—RS=–4 For 2 step repeat 01 RE—RS=–2 For 3 step repeat 11 RE—RS=0 For 4 steps or more 10 RE—RS>0
1	Saturation arithmetic bit (S)	Used with MAC instructions and DSP instructions 1: Designates saturation arithmetic (prevents overflows)
0	T bit	For MOVT, CMP/cond, TAS, TST, BT, BT/S, BF, BF/S, SETT, CLRT and DT instructions, 0: represents false 1: represents true
		For ADDV/ADDC, SUBV/SUBC, DIV0U/DIV0S, DIV1, NEGC, SHAR/SHAL, SHLR/SHLL, ROTR/ROTL and ROTCR/ROTCL instructions, 1: represents occurrence of carry, borrow, overflow or underflow
31–28 15–12	0 bit	0: 0 is always read out; write a 0

There are dedicated load/store instructions for accessing the RS, RE and MOD registers. For example, the RS register is accessed as follows.

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

The following instructions set addresses in the RS, RE registers for zero overhead repeat control:

```
LDRS    @(disp,PC);    disp×2 + PC→RS
LDRE    @(disp,PC);    disp×2 + PC→RE
```

2.2 Features of CPU Instructions

2.2.1 Fetching and Decoding

The SH7622 supports a series of mixed 16-bit and 32-bit instructions. There are no restrictions on the order of instructions within a series of mixed 16-bit and 32-bit instructions.

2.2.2 Integer Unit

The SH7622's integer unit has extended SH-2 CPU core functions and supports DSP operations. The integer unit can execute all SH-1 and SH-2 object code, but is not upward-compatible with SH-3 object code. The integer unit has the following features in addition to those of the SH-2 CPU core.

- Dual addressing function:

  Two on-chip memories can be accessed simultaneously using the main integer unit ALU for X memory address calculation, and a separate 16-bit ALU called a pointer arithmetic unit (PAU) for Y memory address calculation.

- Indexed addressing with pointer update function:

  The addressing mechanism supports indexed addressing with an automatic address pointer update function. The address pointer can be automatically incremented or decremented by 2 or 4 when consecutive words or longwords are accessed in memory. In addition, the address pointer can be incremented by the specified index amount after each memory access.

- Modulo addressing:

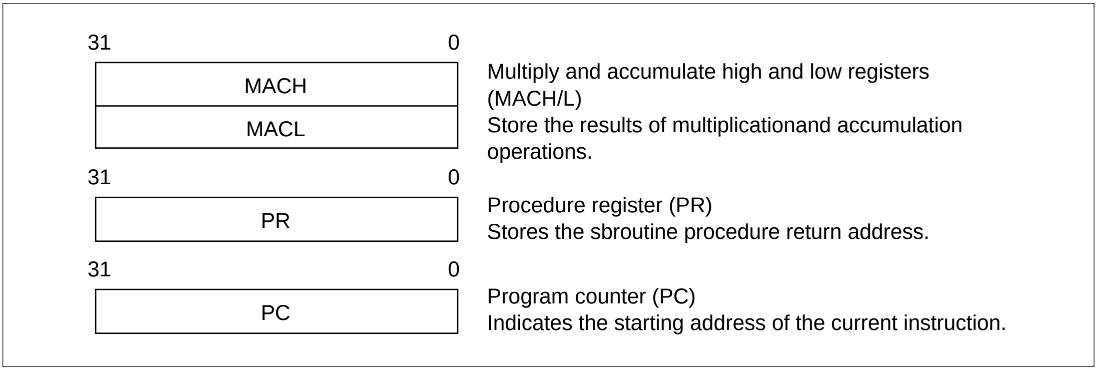
  Modulo addressing is useful for implementing a circular buffer. The start and end modulo addresses are specified by the MOD control register. When the address register is incremented up to the end address value, it is automatically reset to the first address.

- Zero-overhead loop control:

  The integer unit supports zero-overhead program loops, in which loop counter incrementing and judgment of completion of the loop are performed automatically. These loops are important for high-speed DSP applications. To set up such a loop, special registers are used to specify the number of repetitions of the instruction loop, and the loop start address and end address. The processor then automatically executes the loop the specified number of times.

2.2.3 System Registers

SH7622 has four system registers, MACL, MACH, PR and PC (figure 2.3).



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

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

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

Figure 2.3 System Registers

DSR, A0, X0, X1, Y0 and Y1 registers are also treated as system registers. So, data transfer instructions between general registers and system registers are supported for them.

2.2.4 DSP Registers

The SH7622 has eight data registers and one control register (figure 2.4). The data registers are 32-bit width with the exception of registers A0 and A1. Registers A0 and A1 include 8 guard bits (fields A0G and A1G), giving them a total width of 40 bits.

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

Second one is X and Y data transfer operation, “MOVX.W MOVY.W”. This operation accesses the X and Y memories through 16-bit X and Y data buses (figure 2.8). Registers to be loaded or stored by this operation are always upper 16 bits (bits 31–16). X0 and X1 can be destination of the

X memory load and Y0 and Y1 can be destination of Y memory load, but other register cannot be destination register of this operation. When data is read into the upper 16 bits of a register (bits 31–16), the lower 16 bits of the register (bits 15–0) are automatically cleared. A0 and A1 can be stored to the X or Y memory by this operation, but other registers cannot be stored.

Third one is single-data transfer instruction, “MOVS.W” and “MOVS.L”. This instruction accesses any memory location through LDB (figure 2.5). All DSP registers connect to the LDB and be able to be source and destination register of the data transfer. It has word and longword access modes. In the word mode, registers to be loaded or stored by this instruction are upper 16 bits (bits 31–16) for the DSP registers except A0G and A1G. When data is loaded into a register other than A0G and A1G in the word mode, lower half of the register is cleared. When it is A0 or A1, the data is sign-extended to bits 39–32 and lower half of it is cleared. When A0G or A1G is a destination register in the word mode, data is loaded into 8-bit register, but A0 or A1 is not cleared. In the longword mode, when a destination register is A0 or A1, it is sign-extended to bits 39–32.

Tables 2.2 and 2.3 show the data type of registers used in the DSP instructions. Some instructions cannot use some registers shown in the tables because of instruction code limitation. For example, PMULS can use A1 for source registers, but cannot use A0. These tables ignore details of the register selectability.

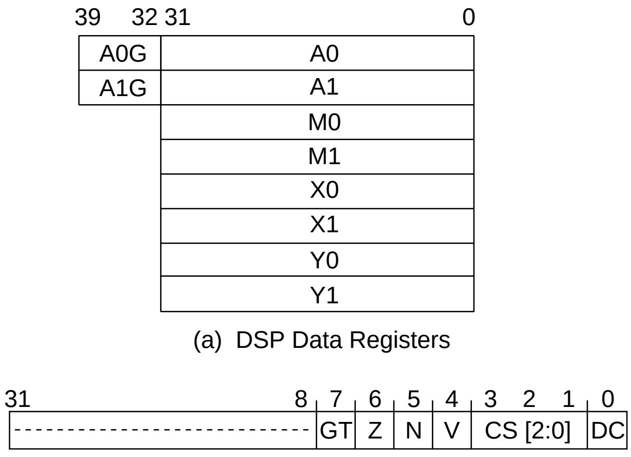
Table 2.2 Destination Register of DSP Instructions

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

Table 2.3 Source Register of DSP Operations

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

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



(a) DSP Data Registers

(b) DSP Status Register (DSR)

Reset status
DSR: All zeros
Others: Undefined

Figure 2.4 DSP Registers

Table 2.4 DSR Register Bits

Bit	Name (Abbreviation)	Function
31–8	Reserved bits	0: Always read out; always use 0 as a write value
7	Signed greater than bit (GT)	Indicates that the operation result is positive (excepting 0), or that operand 1 is greater than operand 2 1: Operation result is positive, or operand 1 is greater
6	Zero bit (Z)	Indicates that the operation result is zero (0), or that operand 1 is equal to operand 2 1: Operation result is zero (0), or equivalence
5	Negative bit (N)	Indicates that the operation result is negative, or that operand 1 is smaller than operand 2 1: Operation result is negative, or operand 1 is smaller
4	Overflow bit (V)	Indicates that the operation result has overflowed 1: Operation result has overflowed
3–1	Status selection bits (CS)	Designate the mode for selecting the operation result status set in the DC bit Do not set either 110 or 111 000: Carry/borrow mode 001: Negative value mode 010: Zero mode 011: Overflow mode 100: Signed greater mode 101: Signed above mode
0	DSP status bit (DC)	Sets the status of the operation result in the mode designated by the CS bits 0: Designated mode status not realized (unrealized) 1: Designated mode status realized

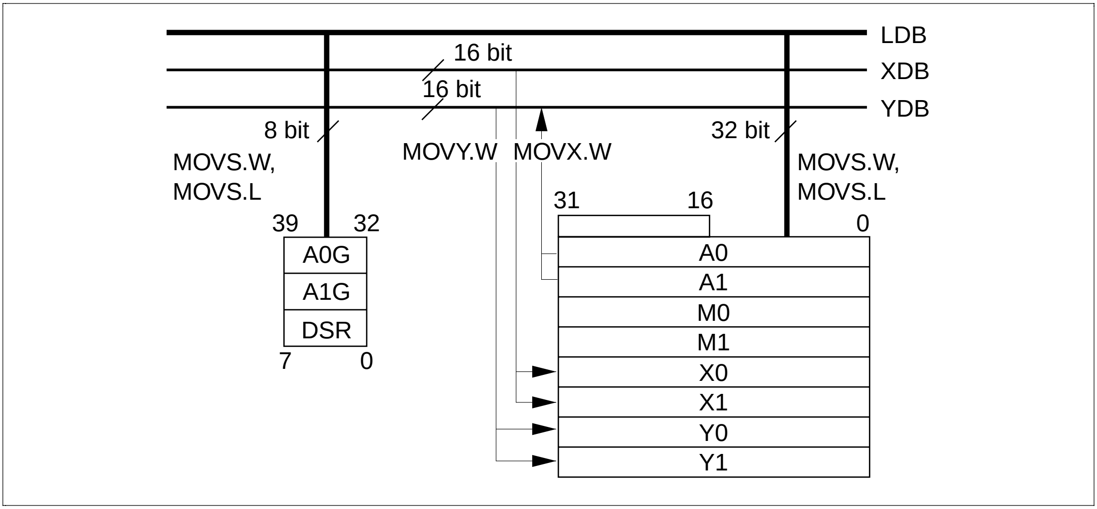


Figure 2.5 Connections of DSP Registers and Buses

The DSP unit has one control register DSP Status Register (DSR). The DSR has conditions of the DSP data operation result (zero, negative, and so on) and a DC bit which is similar to the T bit in the CPU. The DC bit indicates the one of the conditional flags. A DSP data processing instruction controls its execution based on the DC bit. This control affects only the operations in the DSP unit; it controls the update of DSP registers only. It cannot control operations in CPU, such as address register updating and load/store operations. The control bit CS[2:0] specifies the condition to be reflect to the DC bit (table 2.5).

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

Table 2.5 Mode of the DC Bit

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

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

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

When DSR is read by the STS instructions, the upper bits (bit 31 to bit 8) are all 0.

2.3 Data Format

2.3.1 Data Format in Registers (Non-DSP Type)

Register operands are always longwords (32 bits) (figure 2.6). When the memory operand is only a byte (8 bits) or a word (16 bits), it is sign-extended into a longword when loaded into a register.

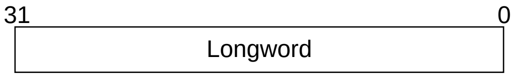


Figure 2.6 Longword Operand

2.3.2 DSP-Type Data Format

The SH7622 has several different data formats that depend on operations. This section explains the data formats for DSP type instructions.

Figure 2.7 shows three DSP-type data formats with different binary point positions. A CPU-type data format with the binary point to the right of bit 0 is also shown for reference.

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

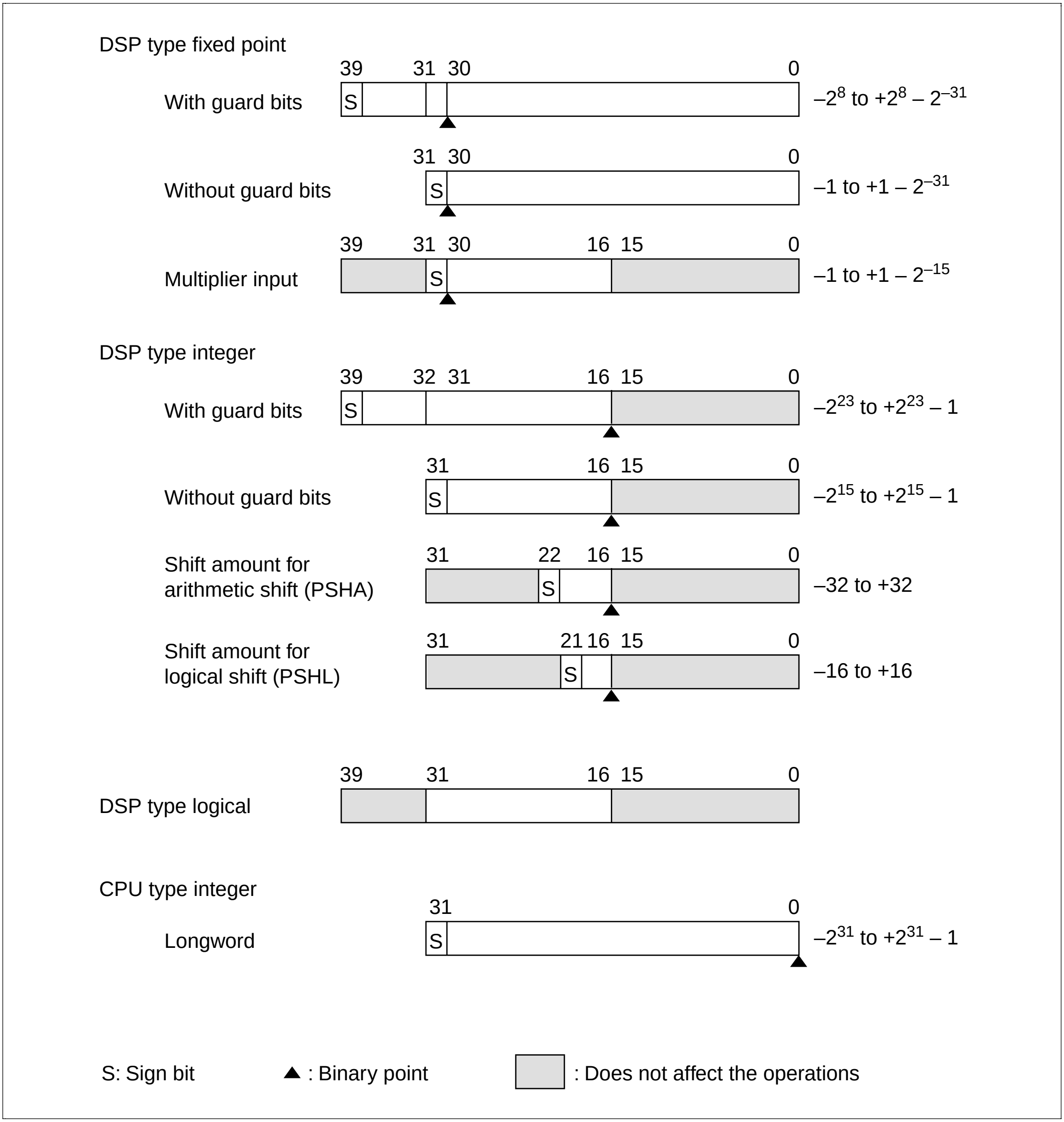


Figure 2.7 Data Format

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

2.3.3 Data Format in Memory

Memory data formats are classified into bytes, words, and longwords. Byte data can be accessed from any address, but an address error will occur if the word data starting from an address other than 2n or longword data starting from an address other than 4n is accessed. In such cases, the data accessed cannot be guaranteed (figure 2.8).

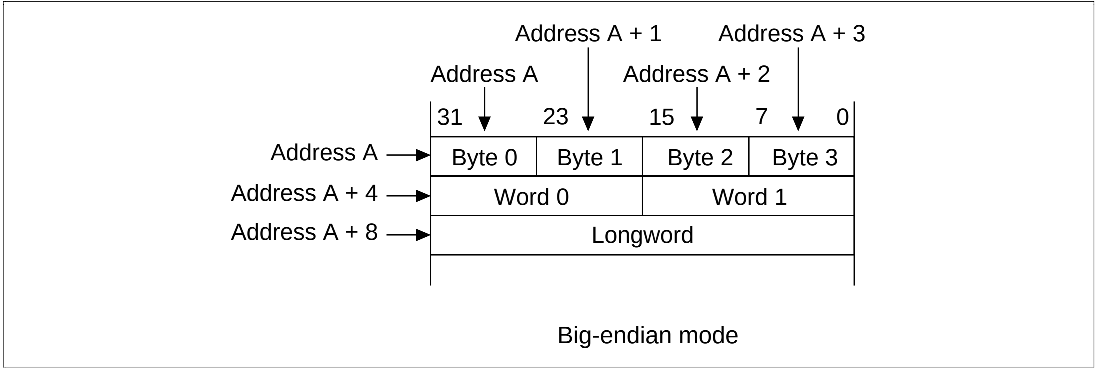


Figure 2.8 Byte, Word, and Longword Alignment

3.1 Data Operations of DSP Unit

3.1.1 ALU Fixed-Point Operations

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

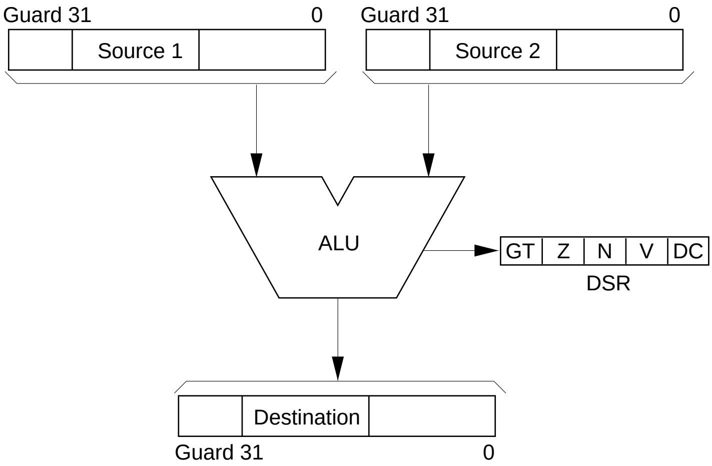


Figure 3.1 ALU Fixed-Point Arithmetic Operation Flow

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

ALU fixed-point operations are executed between registers. Each source and destination operand are selected independently from one of the DSP registers. When a register providing guard bits is specified as an operand, the guard bits are activated for this type of operation. These operations are executed in the final stage of the pipeline sequence, named DSP stage, as shown in figure 3.2.

Table 3.1 Variation of ALU Fixed-Point Operation

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

Table 3.2 Operand Flexibility

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

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

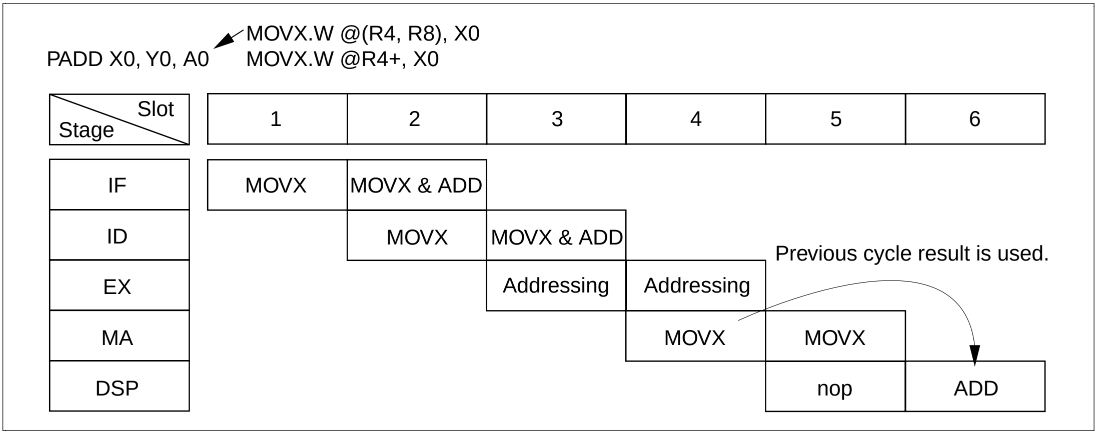


Figure 3.2 Operation Sequence Example

Every time an ALU arithmetic operation is executed, the DC, N, Z, V and GT bits in the DSR register are basically updated in accordance with the operation result. However, in case of a conditional operation, they are not updated even though the specified condition is true and the operation is executed. In case of an unconditional operation, they are always updated in accordance with the operation result. The definition of a DC bit is selected by CS0–2 (condition selection) bits in the DSR register. The DC bit result is as follows:

Carry or Borrow Mode: CS [2:0] = 000: The DC bit indicates that carry or borrow is generated from the most significant bit of the operation result, except the guard-bit parts. Some examples are shown in figure 3.3. This mode is the default condition.

Negative Value Mode: CS [2:0] = 001: The DC flag indicates the same state as the MSB of the operation result. When the result is a negative number, the DC bit shows 1. When it is a positive number, the DC bit shows 0. The ALU always executes 40-bit arithmetic operation, so the sign bit to detect whether positive or negative is always got from the MSB of the operation result regardless of the destination operand. Some examples are shown in figure 3.4.

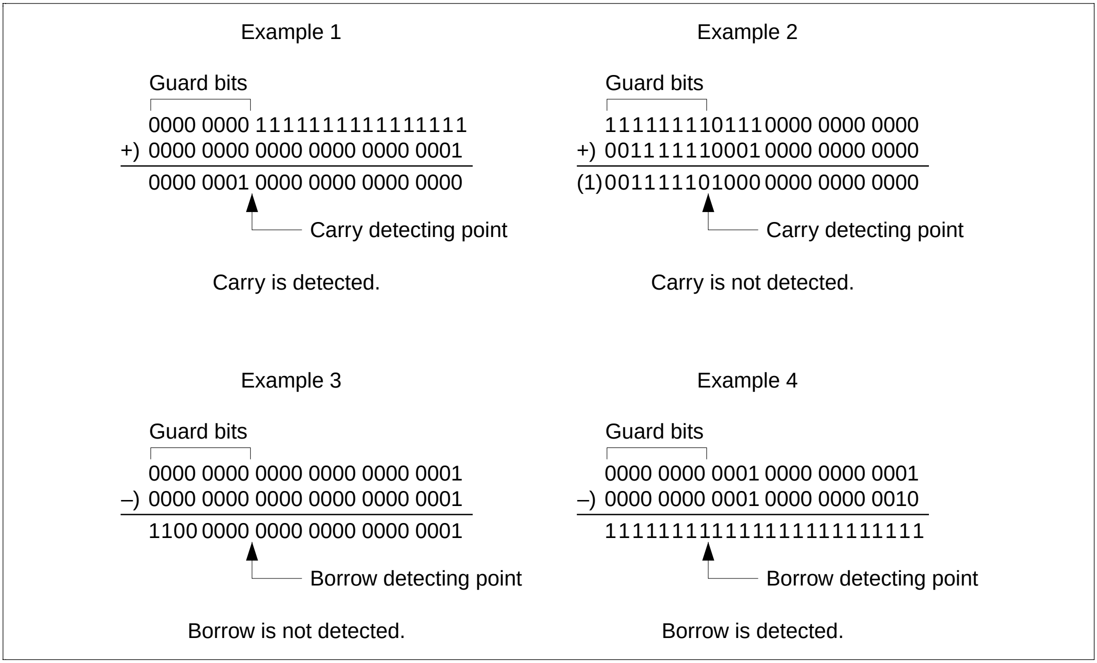


Figure 3.3 DC Bit Generation Examples in Carry or Borrow Mode

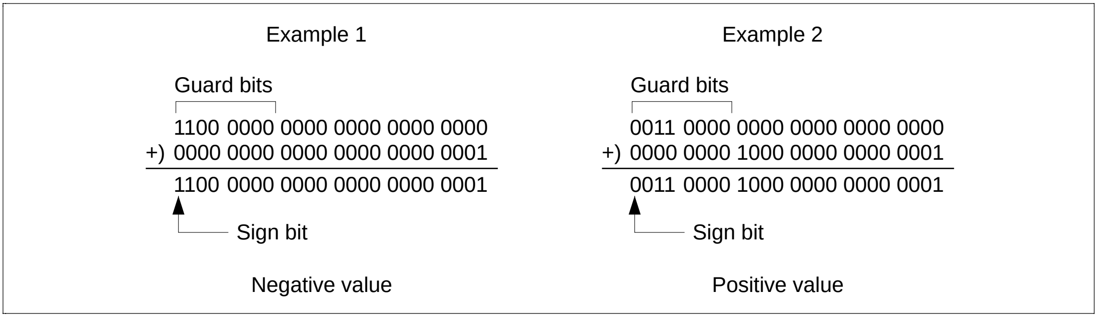


Figure 3.4 DC Bit Generation Examples in Negative Value Mode

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

Overflow Mode: CS [2:0] = 011: The DC bit indicates whether or not overflow occurs in the result. When an operation yields a result beyond the range of the destination register, except the guard-bit parts, the DC bit is set. Even though guard bits are provided, the DC bit always indicates the result of guard bits not provided case. So, the DC bit is always set if the guard-bit parts are used for large number representation. Some flag detection examples are shown in figure 3.5.

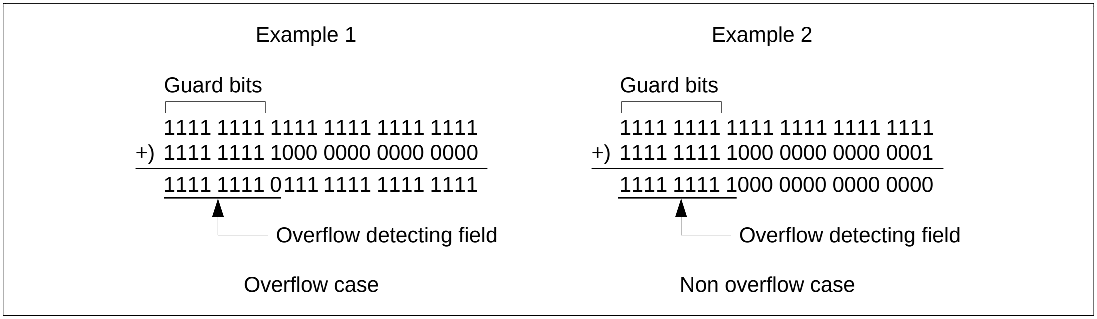


Figure 3.5 DC Bit Generation Examples in Overflow Mode

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

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

When the PCMP operation is executed under this condition mode, the result of the DC bit is the same as the T bit result of the PCMP/GT operation of the SH core instruction.

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

DC = ~ (Negative ^ Over-range)

When the PCMP operation is executed under this condition mode, the result of the DC bit is the same as T bit result of a PCMP/GE operation of the SH core instruction.

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

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

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

3.1.2 ALU Integer Operations

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

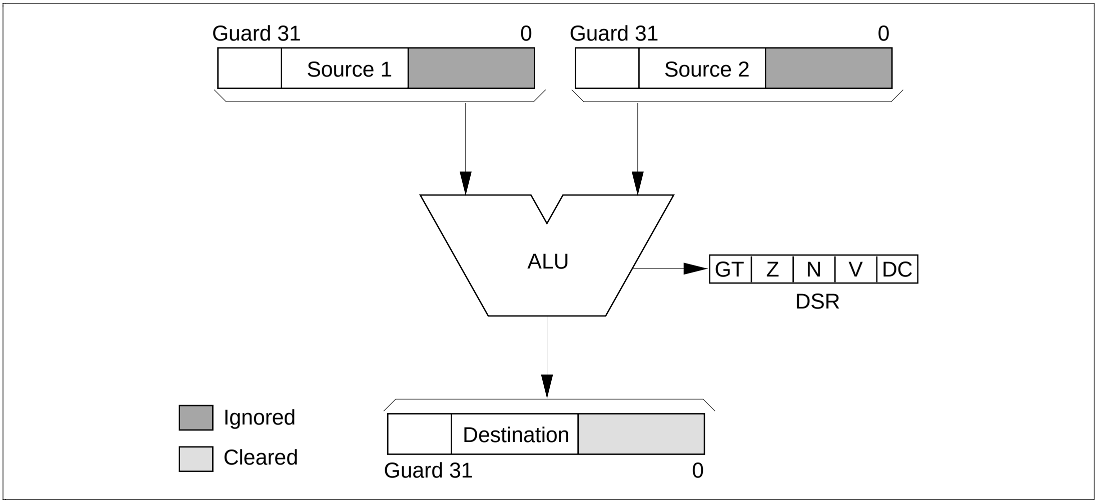


Figure 3.6 ALU Integer Arithmetic Operation Flow

Table 3.3 Variation of ALU Integer Operation

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

Note: The ALU integer operations are basically 24-bit operation, the upper 16 bits of the base precision and 8 bits of the guard-bits parts. So the signed bit is copied to the guard-bit parts when a register not providing the guard-bit parts is specified as the source operand. When a register not providing the guard-bit parts as a destination operand, the lower 32 bits of the operation result are input into the destination register.

In ALU integer arithmetic operations, the lower word of the source operand is ignored and the lower word of the destination operand is automatically cleared. The guard-bit parts are effective in integer arithmetic operations if they are supported. Others are basically the same operation as ALU fixed-point arithmetic operations. As shown in table 3.3, however, this type of operation provides two kinds of instructions only, so that the second operand is actually either +1 or –1. When a word data is loaded into one of the DSP unit's registers, it is input as an upper word data. So it is reasonable for increment and decrement operations to execute using the upper word in the DSP unit. When a register providing guard bits is specified as an operand, the guard bits are also activated. These operations are executed in the final stage of the pipeline sequence, named the DSP stage, as shown in figure 3.2, as well as fixed-point operations.

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

In case of a conditional operation, they are not updated even though the specified condition is true and the operation is executed. In case of an unconditional operation, they are always updated in accordance with the operation result. See section 3.1.1, ALU Fixed-Point Operations, for details.

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

3.1.3 ALU Logical Operations

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

Logical operations are also executed between registers. Each source and destination operand are selected independently from one of the DSP registers. As shown in figure 3.7, this type of operation uses the upper word of each operand only. Lower word and guard-bit parts are ignored for the source operand and those of the destination operand are automatically cleared. These operations are also executed in the final stage of the pipeline sequence, named DSP stage, as shown in figure 3.2.

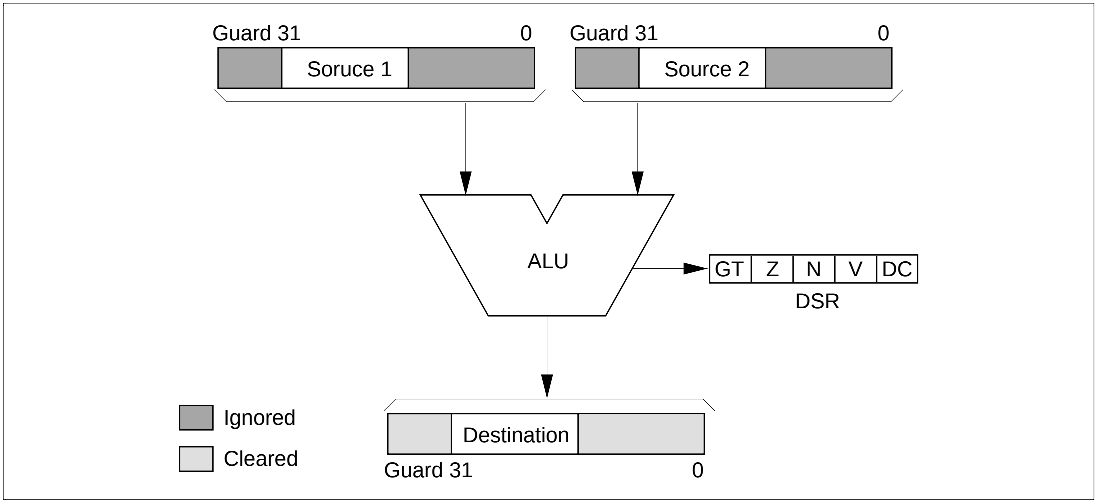


Figure 3.7 ALU Logical Operation Flow

Table 3.4 Variation of ALU Logical Operation

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

Every time an ALU logical operation is executed, the DC, N, Z, V and GT bits in the DSR register are basically updated in accordance with the operation result. In case of a conditional operation, they are not updated even though the specified condition is true and the operation is executed. In case of an unconditional operation, they are always updated in accordance with the operation result. The definition of DC bit is selected by CS0–2 (condition selection) bits in the DSR register. The DC bit result is

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

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

The following restriction applies to the PXOR instruction.

- In PXOR Sx, Sy, Sz, if the MSB of both Sx and Sy is 1 and the upper word of Sx and the upper word of Sy are equal, the zero flag (Z bit in DSR) will not be set. If zero mode is set at this time (DSR.CS[2:0] = 010), the DC bit in DSR will not be set.

3.1.4 Fixed-Point Multiply Operation

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

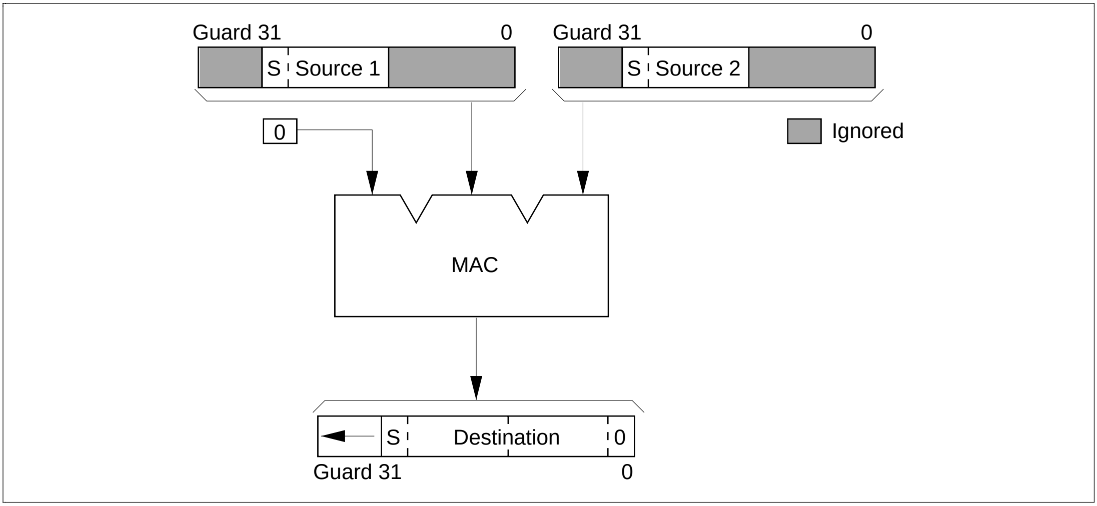


Figure 3.8 Fixed-Point Multiply Operation Flow

Table 3.5 Variation of Fixed-Point Multiply Operation

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

Table 3.6 Operand Flexibility

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

Note: The multiply operations basically generates 32 bits of the operation result. So when a register providing the guard-bit parts are specified as a destination operand, the guard-bit parts will copy the 32nd bit (MSB) of the operation result.

The multiply operation of the DSP unit side is not integer but fixed-point arithmetic. So, the upper words of each multiplier and multiplicand are input into a MAC unit as shown in figure 3.8. In the SH's standard multiply operations, the lower words of both source operands are input into a MAC unit. The operation result is also different from the SH's case. The SH's multiply operation result is aligned to the LSB of the destination, but the fixed-point multiply operation result is aligned to the MSB, so that the LSB of the fixed-point multiply operation result is always 0.

This fixed-point multiply operation is executed in one cycle using 32 bits by 8-bit MAC unit. The other SH's multiply and MAC operations are executed as the SH-2's are. Multiply operation doesn't affect any condition code bits, DC, N, Z, V and GT, in the DSR register.

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

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

3.1.5 Shift Operations

Shift operations can use either register or immediate value as the shift amount operand. Other source and destination operands are specified by the register. There are two kinds of shift operations. Table 3.7 shows the variations of this type of operation. The flexibility of each operand, except for immediate operands, is the same as the ALU fixed-point operations as shown in table 3.2. See section 4.3 for more detailed information about each instruction and operand. See note of section 3.1.1, ALU Fixed-Point Operations.

Table 3.7 Variation of Shift Operations

Mnemonic	Function	Source 1	Source 2	Destination
PSHA Sx, Sy, Dz	Arithmetic shift	Sx	Sy	Dz
PSHL Sx, Sy, Dz	Logical shift	Sx	Sy	Dz
PSHA #Imm, Dz	Arithmetic shift w/imm.	Dz	Imm1	Dz
PSHL #Imm, Dz	Logical shift w/imm.	Dz	Imm2	Dz

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

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

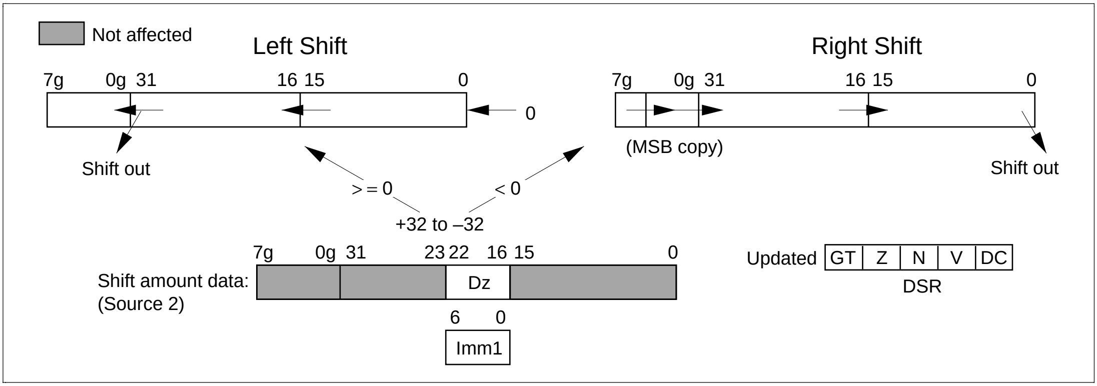


Figure 3.9 Shift Operation Flow

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

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

Every time an arithmetic shift operation is executed, the DC, N, Z, V and GT bits in the DSR register are basically updated in accordance with the operation result. In case of a conditional operation, they are not updated even though the specified condition is true and the operation is executed. In case of an unconditional operation, they are always updated in accordance with the operation result. The definition of DC bit is selected by CS0–2 (condition selection) bits in the DSR register. The DC bit result is

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

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

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

   Same definition as ALU fixed-point arithmetic operations.

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

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

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

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

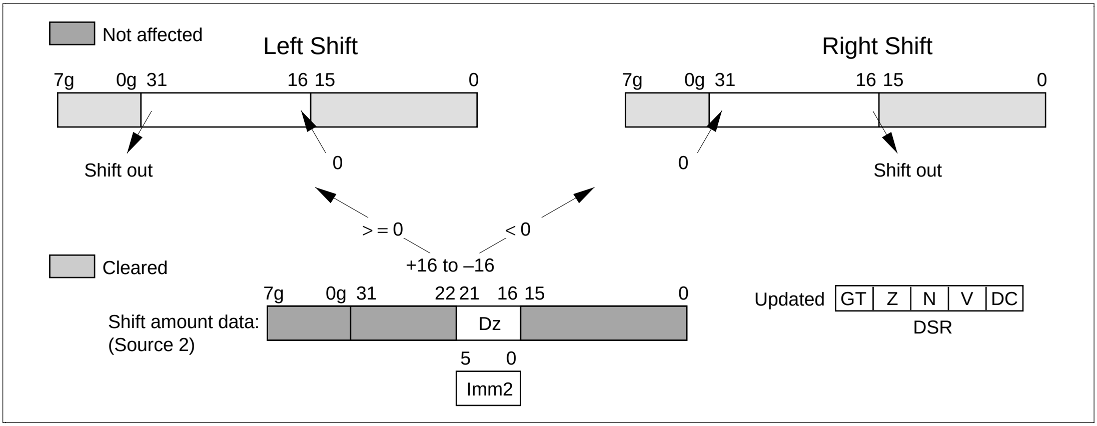


Figure 3.10 Logical Shift Operation Flow

As shown in figure 3.10, the logical shift operation uses the upper word of the source 1 and the destination operands. The lower word and guard-bit parts are ignored for the source operand and those of the destination operand are automatically cleared as in the ALU logical operations. The shift amount is specified by the source 2 operand as an integer data. The source 2 operand can be specified by either register or immediate operand. Available shift range is from –16 to +16. Here, negative value means the right shift, positive value means the left shift. It's possible for any source

2 operand to specify from –32 to +31 but the result is unknown if invalid shift value is specified. In case of the shift with immediate operand instruction, the source 1 operand must be the same register as the destination’s. These operations are executed in the final stage of the pipeline sequence, named DSP stage, as shown in figure 3.2.

Every time a logical shift operation is executed, the DC, N and Z bits in the DSR register are basically updated with the operation result but V and GT bits are always cleared. In case of a conditional operation, they are not updated, even though the specified condition is true and the operation is executed. In case of unconditional operation, they are always updated with the operation result. The definition of the DC bit is selected by CS0–2 (condition selection) bits in the DSR register. The DC bit result is

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

   The DC bit indicates the last shifted out data as the operation result.
2. Negative Value Mode: CS [2:0] = 001

   Same definition as ALU logical operations.
3. Zero Value Mode: CS [2:0] = 010

   Same definition as ALU logical operations.
4. Overflow Mode: CS [2:0] = 011

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

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

   The DC bit is always cleared.

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

The V bit set by the PSHA instruction may not have been set after it should have been, and so should not be used. Also, the DC bit set by the PSHA instruction should not be used in overflow mode (DSR.CS[2:0] = 011).

3.1.6 Most Significant Bit Detection Operation

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

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

As shown in figure 3.11, the ‘PDMSB’ operation uses full-size data as a source operand, but the destination operand is treated as an integer operation result because shift amount data for normalization should be integer data as described in section 3.1.5 (Arithmetic Shift). These operations are executed in the final stage of the pipeline sequence, named DSP stage, as shown in figure 3.11.

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

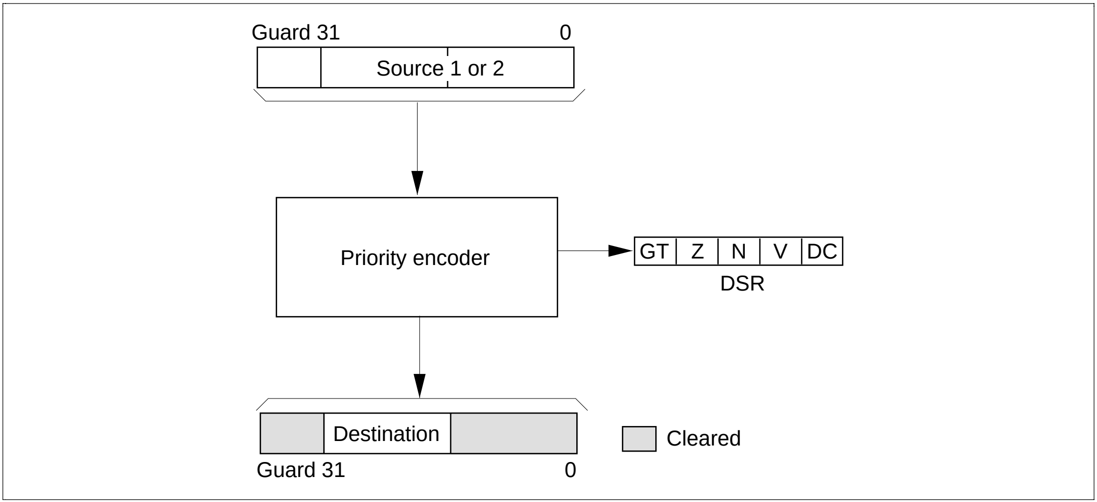


Figure 3.11 ‘PDMSB’ Operation Flow

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

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

Table 3.8 Operation Definition of ‘PDMSB’

Data in SRC														Result for DST								
Guard Bit					Upper Word					Lower Word				Guard Bit	Upper Word							
7g	6g	…	1g	0g	31	30	29	28	…	3	2	1	0	7g–0g	31–22	21	20	19	18	17	16	Decimal
0	0	…	0	0	0	0	0	0	…	0	0	0	0	All 0	All 0	0	1	1	1	1	1	+31
0	0	…	0	0	0	0	0	0	…	0	0	0	1	All 0	All 0	0	1	1	1	1	0	+30
0	0	…	0	0	0	0	0	0	…	0	0	1	*	All 0	All 0	0	1	1	1	0	1	+29
0	0	…	0	0	0	0	0	0	…	0	1	*	*	All 0	All 0	0	1	1	1	0	0	+28
		:							:									:				
0	0	…	0	0	0	0	0	1	…	*	*	*	*	All 0	All 0	0	0	0	0	1	0	+2
0	0	…	0	0	0	0	1	*	…	*	*	*	*	All 0	All 0	0	0	0	0	0	1	+1
0	0	…	0	0	0	1	*	*	…	*	*	*	*	All 0	All 0	0	0	0	0	0	0	0
0	0	…	0	0	1	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	1	1	1	–1
0	0	…	0	1	*	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	1	1	0	–2
		:							:									:				
0	1	…	*	*	*	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	0	0	0	–8
1	0	…	*	*	*	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	0	0	0	–8
																		:				
1	1	…	1	0	*	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	1	1	0	–2
1	1	…	1	1	0	*	*	*	…	*	*	*	*	All 1	All 1	1	1	1	1	1	1	–1
1	1	…	1	1	1	0	*	*	…	*	*	*	*	All 0	All 0	0	0	0	0	0	0	0
1	1	…	1	1	1	1	0	*	…	*	*	*	*	All 0	All 0	0	0	0	0	0	1	+1
1	1	…	1	1	1	1	1	0	…	*	*	*	*	All 0	All 0	0	0	0	0	1	0	+2
		:							:									:				
1	1	…	1	1	1	1	1	1	…	1	0	*	*	All 0	All 0	0	1	1	1	0	0	+28
1	1	…	1	1	1	1	1	1	…	1	1	0	*	All 0	All 0	0	1	1	1	0	1	+29
1	1	…	1	1	1	1	1	1	…	1	1	1	0	All 0	All 0	0	1	1	1	1	0	+30
1	1	…	1	1	1	1	1	1	…	1	1	1	1	All 0	All 0	0	1	1	1	1	1	+31

Note: * Don’t care.

Table 3.9 Variation of 'PDMSB' Operations

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

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

3.1.7 Rounding Operation

The DSP unit provides the rounding function that rounds from 32 bits to 16 bits. In case of providing guard-bit parts, it rounds from 40 bits to 24 bits. When a round instruction is executed, H'00008000 is added to the source operand data and then, the lower word is cleared. Figure 3.12 shows the rounding operation flow and figure 3.13 shows the operation definition. Table 3.10 shows the variation of this type of operation. The flexibility of each operand is the same as ALU fixed-point operations as shown in table 3.2. See note of section 3.1.1, ALU Fixed-Point Operations.

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

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

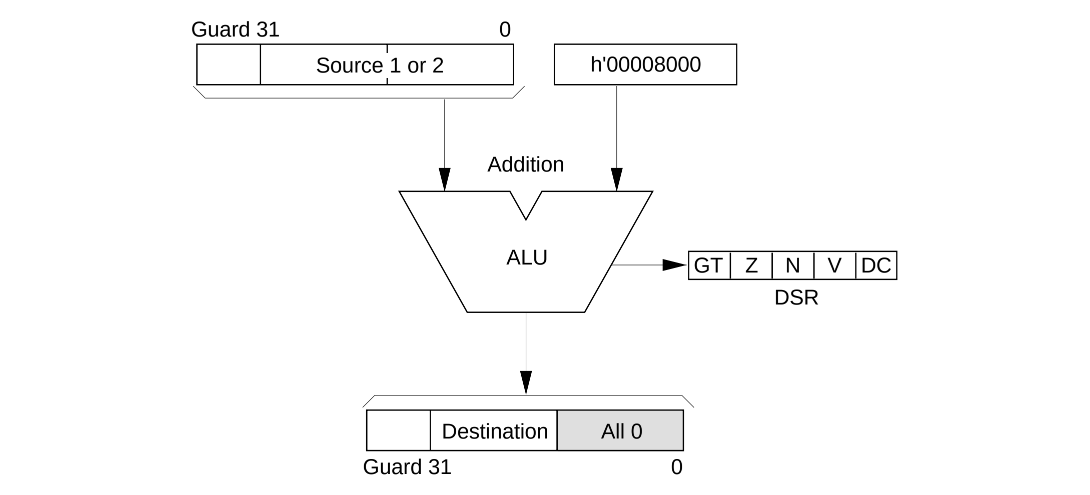


Figure 3.12 Rounding Operation Flow

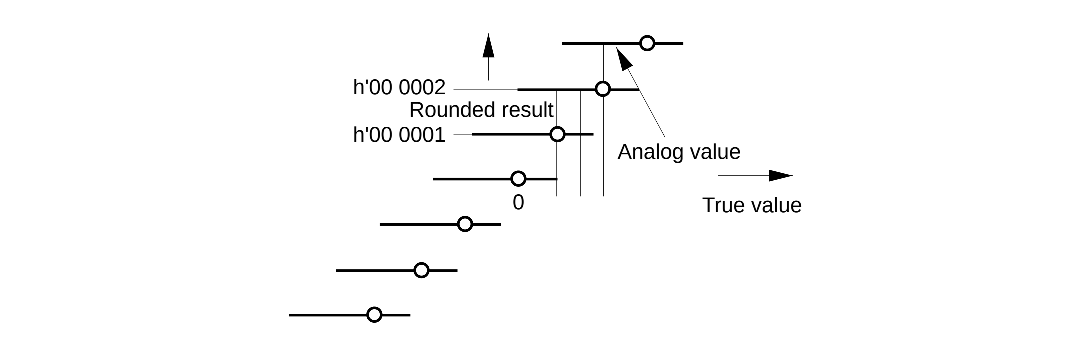


Figure 3.13 Definition of Rounding Operation

Table 3.10 Variation of Rounding Operations

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

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

3.1.8 Overflow Protection

The S bit in the SR register is effective for any arithmetic operations executed in the DSP unit, including the conventional SH's multiply and the MAC operations. The S bit in the SR register, in SH's CPU core, is used as the overflow protection enable bit. The arithmetic operation overflows when the operation result exceeds the range of two's complement representation without guard-bit parts. Table 3.11 shows the definition of overflow protection for fixed-point arithmetic operations, including fixed-point signed by signed multiplication described in section 3.1.4, Fixed-Point Multiply Operation. Table 3.12 shows the definition of overflow protection for integer arithmetic operations. When a SH's conventional multiply or MAC operation is executed, the S bit function is completely the same as the current SH-2's definition.

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

Table 3.11 Definition of Overflow Protection for Fixed-Point Arithmetic

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

Table 3.12 Definition of Overflow Protection for Integer Arithmetic

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

Note: * Don't care.

3.1.9 Data Transfer Operation

The SH7622 can execute a maximum of two data transfer operations between the DSP register and the on-chip data memory in parallel for the DSP unit. This results almost the same performance as other DSPs. The SH7622 provides three types of data transfer instructions for the DSP unit.

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

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

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

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

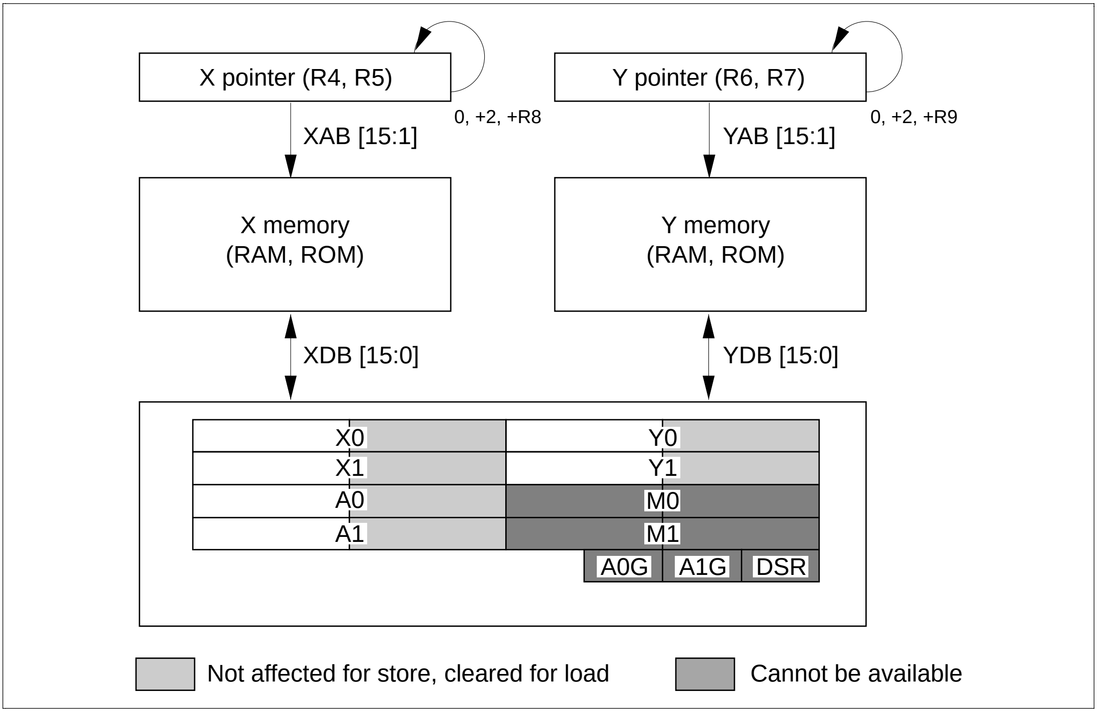


Figure 3.14 Data Transfer Operation Flow

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

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

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

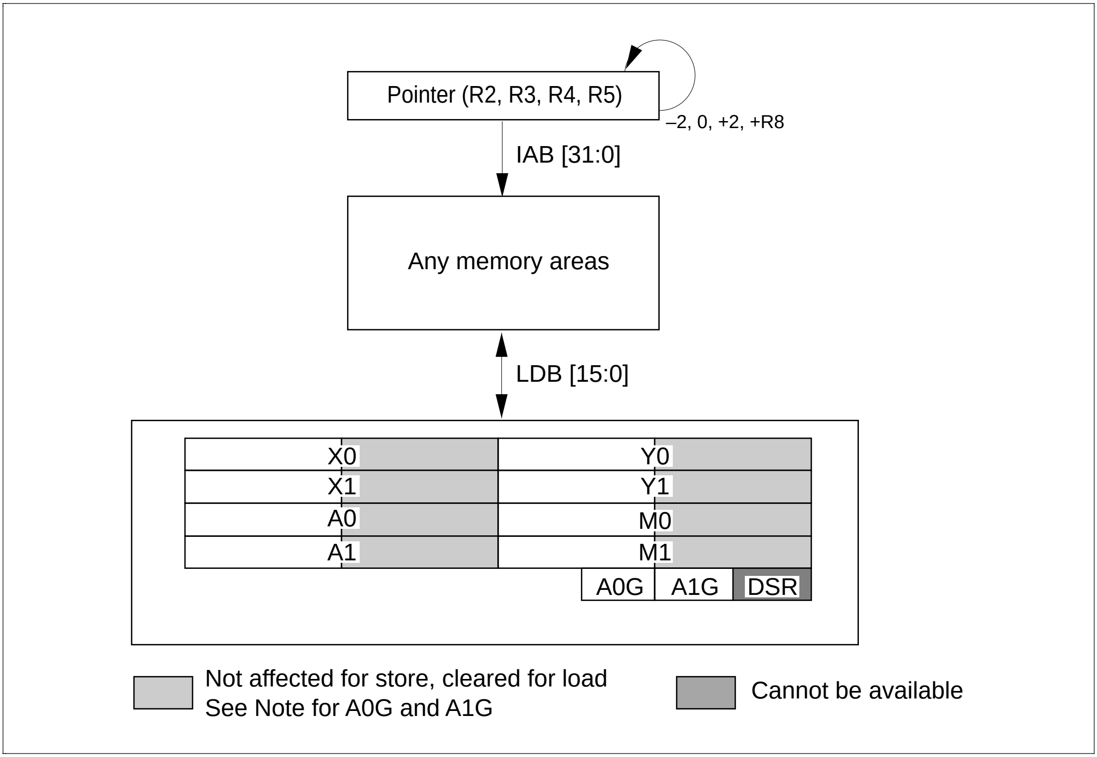


Figure 3.15 Word of Data-Transfer Operation Flow

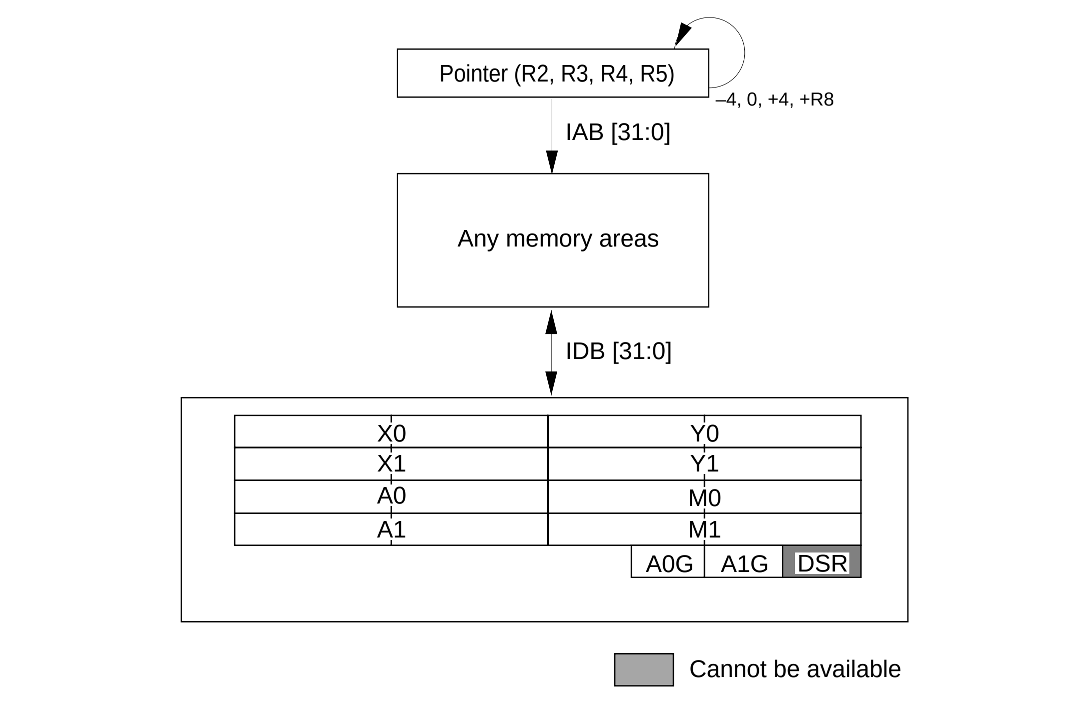
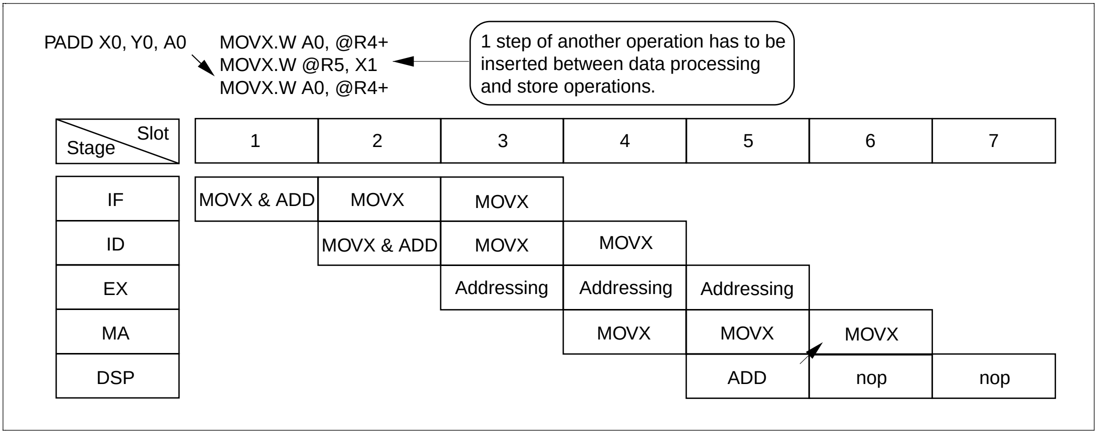


Figure 3.16 Longword of Data-Transfer Operation Flow

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



```
PADD X0, Y0, A0    MOVX.W A0, @R4+
                   MOVX.W @R5, X1
                   MOVX.W A0, @R4+
```

1 step of another operation has to be inserted between data processing and store operations.

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

Figure 3.17 Instruction Sequence Example Between Data Processing and Store

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

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

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

3.1.10 Local Data Move Operation

The DSP unit of the SH7622 provides additional two independent registers, MACL and MACH, in order to support conventional SH-2's multiply/MAC operations. They can be also used as temporary storage registers. Local data move instruction between MACH/L and other DSP registers to make use of this benefit. Figure 3.18 shows the flow of seven local data move instructions. Table 3.13 shows the variation of this type of instruction.

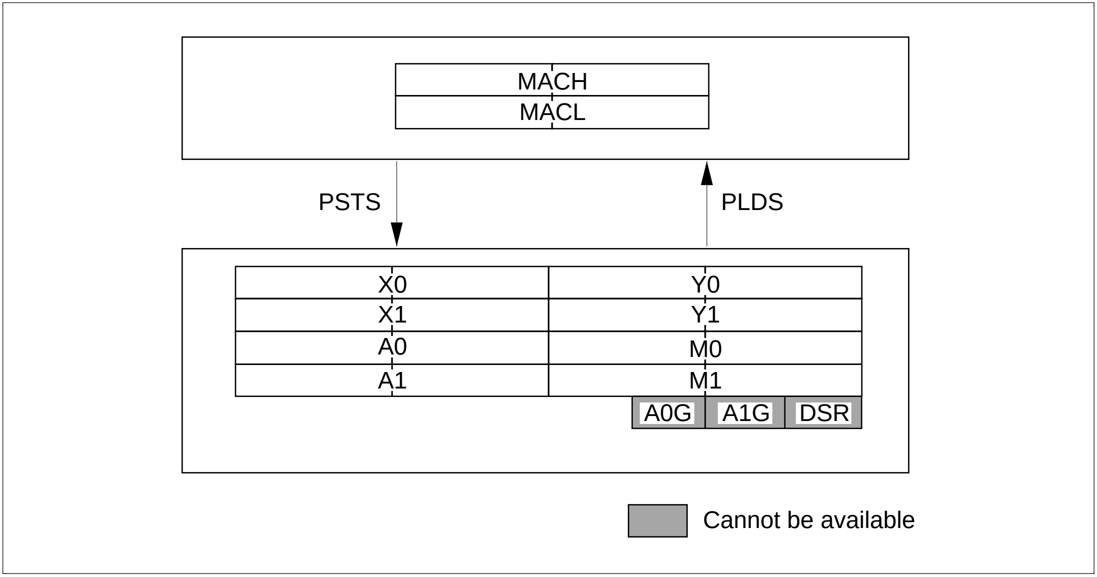


Figure 3.18 Local Data Move Instruction Flow

Table 3.13 Variation of Local Data Move Operation

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

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

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

3.1.11 Operand Conflict

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

Table 3.14 Operand Flexibility

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

Notes: ○ Operand confliction case

* Available regs (for operand)

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

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

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

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

3.2 DSP Addressing

3.2.1 DSP Loop Control

The SH7622 prepares a special control mechanism for efficient loop control. An instruction ‘SETRC’ sets repeat times into the repeat counter RC (12 bits), and an execution mode in which a program loop executes repetitively until RC is equal to ‘1’. After completion of the repeat instructions, the contents of the RC becomes 0.

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

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

To do steps 1 and 2, use new instructions:

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

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

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

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

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

```
RptEnd:     instr5;
            instr6;
```

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

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

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

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

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

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

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

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

    a. When instructions are located in external memory or cache memory, use at least four instructions in a loop.

    b. When instructions are located in X/Y memory, place an instruction that does not access X/Y memory at address 4n in the loop.

    If there is no problem with having bus release or the start of DMA delayed until the end of a repeat loop, this restriction is irrelevant.

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

RC>=1 cases,

1. One step repeat

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

2. Two step repeat

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

3. Three step repeat

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

4. Four or more steps repeat

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

RC=0 case,

Acceptable for any interrupts

Figure 3.19 Restriction of Interrupt Acceptance in Repeat Loop

Note 1: Actual Implementation

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

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

Table 3.17 RS and RE Setting Rule

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

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

CASE 1: 1 Repeated Instruction

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

CASE 2: 2 Repeated Instructions

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

CASE 3: 3 Repeated Instructions

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

CASE 4: 4 or more Repeated Instructions

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

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

Note 2: Extended Instruction REPEAT

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

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

This instruction can be used in the following way:

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

CASE 1: 1 Repeated Instruction

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

CASE 2: 2 Repeated Instruction

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

CASE 3: 3 Repeated Instruction

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

CASE 4: 4 or more Repeated Instructions

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

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

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

3.2.2 DSP Data Addressing

The SH7622 has two types of memory access instructions: one is “X and Y data transfer instruction” (MOVX.W and MOVY.W), and the other one is “single data transfer instruction” (MOVS.W and MOVS.L). Data addressing of these two types of instruction are different. Table 3.18 shows a summary of DSP data transfer instructions.

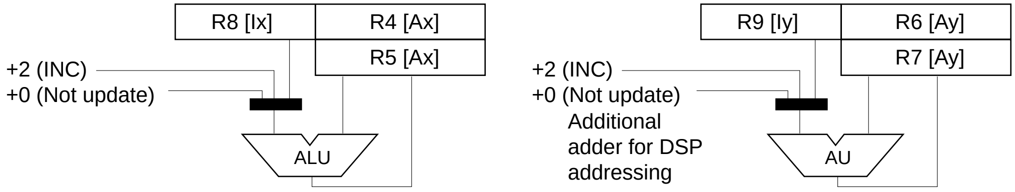
Table 3.18 Summary of DSP Data Transfer Instructions

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

Addressing Instructions for MOVX.W and MOV.W: The SH7622 can access X and Y data memories simultaneously (MOVX.W and MOVY.W). The DSP instructions have two address pointers that *simultaneously* access X and Y data memories. The DSP instruction has only pointer-addressing (it does not have immediate-addressing). Address registers are divided into two sets, R4,5 (Ax: Address register for X memory) and R6,7 (Ay: Address register for Y memory). There are three data addressing operations for X and Y data transfer instruction.

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

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



Three address operation types,
1. Increment
2. Add-index-register (Ix/Iy)
3. Not update
All operations are post-update type.
To decrement address pointer, set –2 to an index register.

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

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

Addressing Instructions for MOVS: The SH7622 has single-data transfer instruction (MOVS.W and MOVS.L) to load/store DSP data registers. In this instruction, R2–5 (As: Address register for single-data transfer) are used for address pointer.

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

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

4. Decrement address register (pre-update)

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

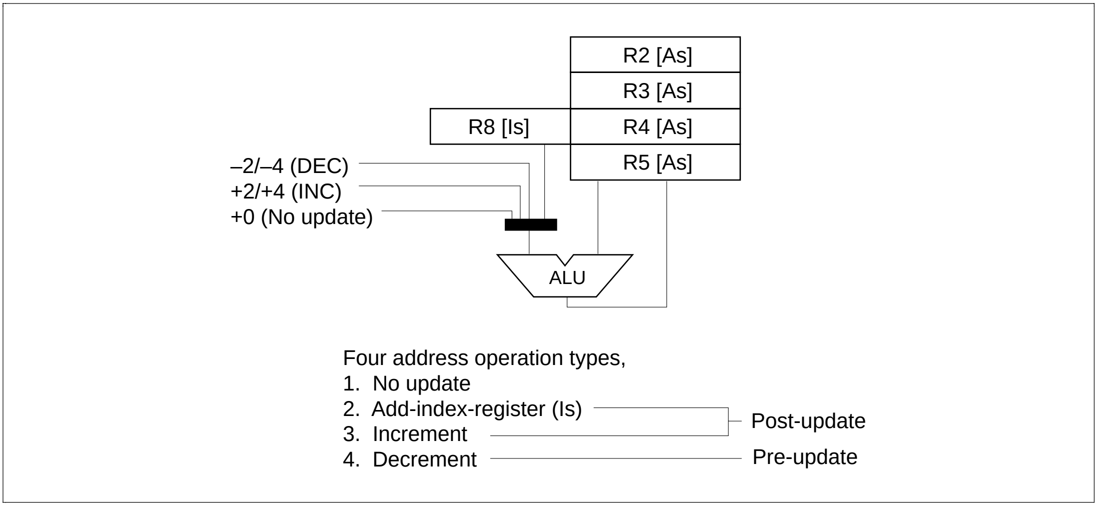


Figure 3.21 DSP Addressing Instructions for MOVS

Modulo Addressing: The SH7622 provides modulo addressing mode, which is common in DSPs. In modulo addressing mode, the address register is updated as explained above. When the address pointer reaches the pre-defined address (the modulo-end address), it goes to the modulo start address.

Modulo addressing is available for X and Y data transfer instruction (MOVX and MOVY), but not single-data transfer instruction (MOVS). The DMX and DMY in the SR register are used for the modulo addressing control. If the DMX is ‘1’ then the modulo addressing mode is effective for the X memory address pointer Ax (R4 or R5). If the DMY is ‘1’ then it is effective for the Y memory address pointer Ay (R6 or R7). Modulo addressing is available for one of X and Y address registers at one time. Do not set DMX and DMY simultaneously. If this is done, only the DMY setting will be valid.

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

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

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

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

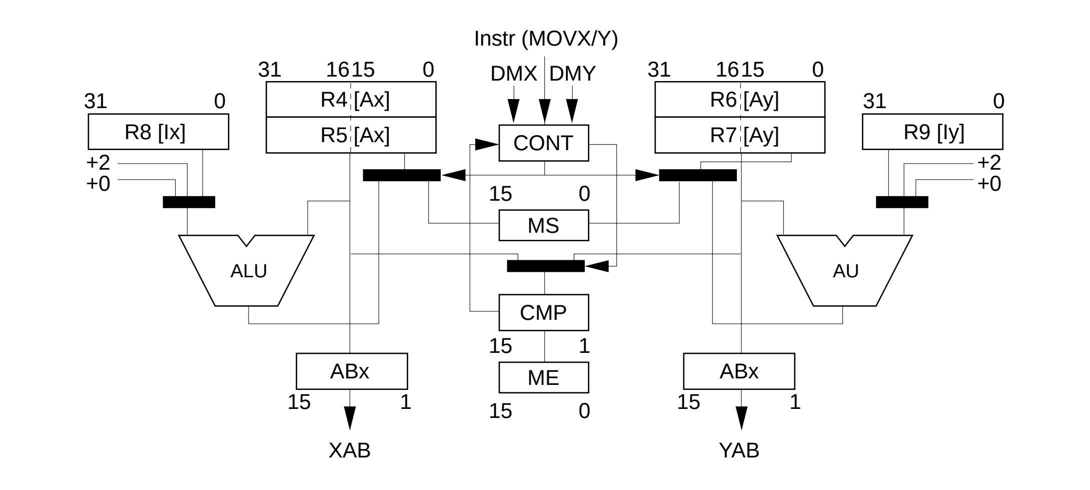


Figure 3.22   Modulo Addressing

An example is shown bellow.

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

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

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

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

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

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

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

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

```
For example, (+2 or R8[Ix] or +0) means that
+2:     if instruction is increment
R8[Ix]: if instruction is add-index-register
+0:     if instruction is not-update
*/

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

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

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

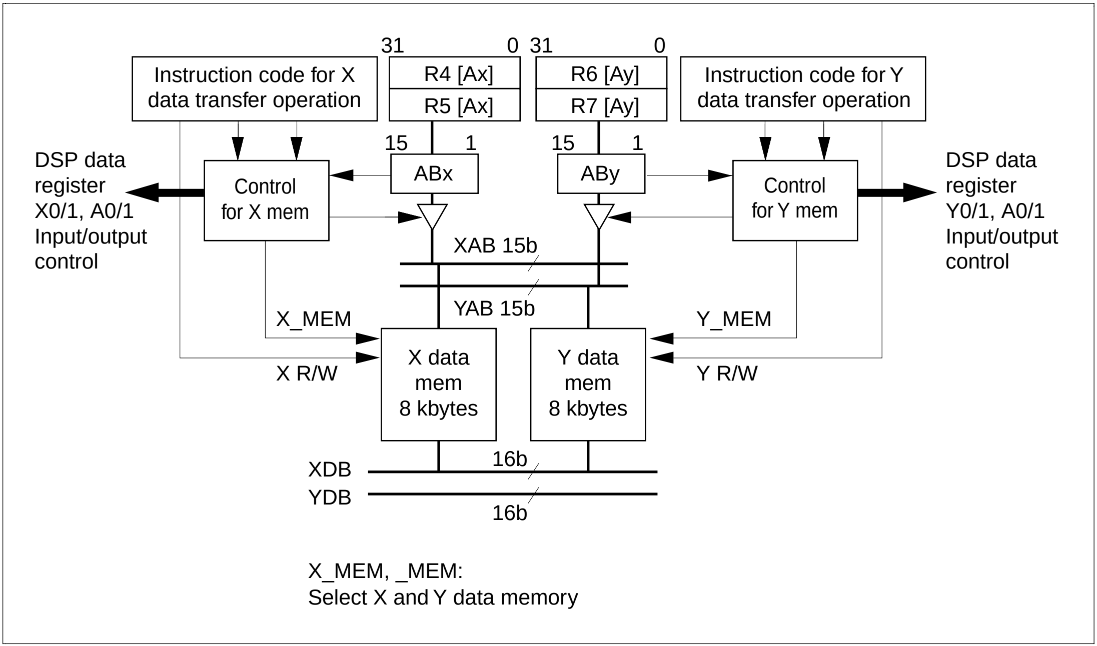

31
0
31
0
Instruction code for X data transfer operation
R4 [Ax]
R5 [Ax]
R6 [Ay]
R7 [Ay]
Instruction code for Y data transfer operation
15
1
15
1
DSP data register X0/1, A0/1 Input/output control
Control for X mem
ABx
ABy
Control for Y mem
DSP data register Y0/1, A0/1 Input/output control
XAB 15b
YAB 15b
X_MEM
Y_MEM
X R/W
Y R/W
X data mem 8 kbytes
Y data mem 8 kbytes
XDB
YDB
16b
16b
X_MEM, _MEM:
Select X and Y data memory

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

Control for X mem

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

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

Single-Data Transfer Instruction (MOVS.W and MOVS.L): The SH7622 has single load/store instruction for the DSP registers. It is similar to a load/store instruction for a system register. It transfers data between memory and DSP data registers using IAB and LDB. There may be access conflict between the data access and the instruction fetch.

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

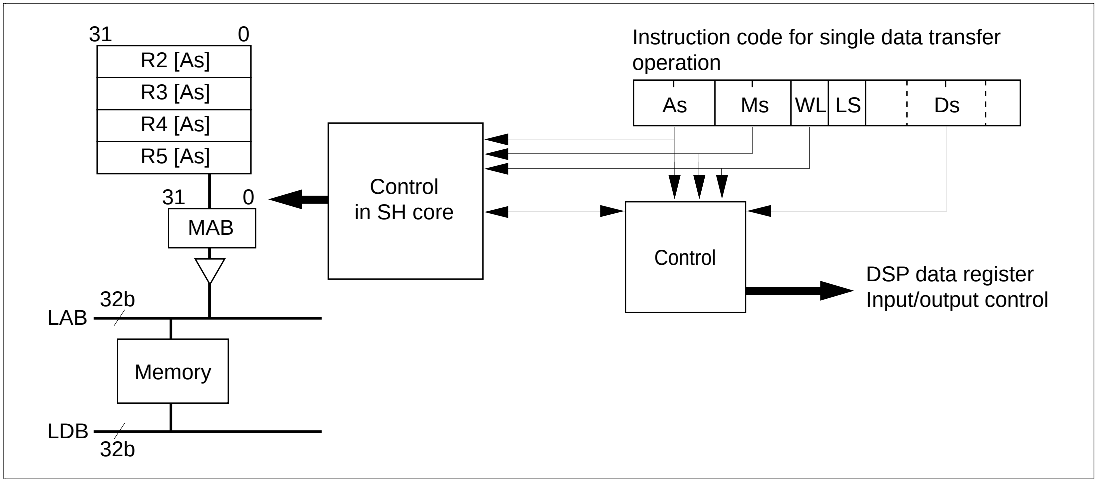


Figure 3.24 Load/Store Control for Single-Data Transfer Instruction

Control

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

Section 4 Instruction Set

4.1 Basic Concept of SH7622 Instruction Set

In order to improve digital signal processing performance, DSP type of instructions are added to form SH7622's ISA. Its relationship with the rest of the SuperH family is

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

This section is organized in two parts: section 4.2, SH-1, SH-2 Compatible Instruction Set and section 4.3, Instructions for DSP Extension. Instructions described in section 4.2 are all 16-bit in length and are compatible to SH-1 and SH-2. DSP Extension instructions are divided into 3 groups:

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

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

4.2 SH-1, SH-2 Compatible Instruction Set

4.2.1 Instruction Set by Classification

SH-1 and SH-2 instruction set include 62 basic instruction types, divided into seven functional classifications, as listed in table 4.1. Tables 4.3 to 4.8 summarize instruction notation, machine mode, execution time, and function.

Table 4.1 Classification of Instructions

Classification	Types	Operation Code	Function	Number of Instructions
Data transfer	5	MOV	Data transfer Immediate data transfer Peripheral module data transfer Structure data transfer	39
		MOVA	Effective address transfer	
		MOVT	T-bit transfer	
		SWAP	Swap of upper and lower bytes	
		XTRCT	Extraction of the middle of registers connected	
Arithmetic operations	21	ADD	Binary addition	33
		ADDC	Binary addition with carry	
		ADDV	Binary addition with overflow check	
		CMP/cond	Comparison	
		DIV1	Division	
		DIV0S	Initialization of signed division	
		DIV0U	Initialization of unsigned division	
		DMULS	Signed double-length multiplication	
		DMULU	Unsigned double-length multiplication	
		DT	Decrement and test	
		EXTS	Sign extension	
		EXTU	Zero extension	
		MAC	Multiply/accumulate, double-length multiply/accumulate operation	
		MUL	Double-length multiplication (32 × 32 bits)	
		MULS	Signed multiplication (16 × 16 bits)	
		MULU	Unsigned multiplication (16 × 16 bits)	
		NEG	Negation	
		NEGC	Negation with borrow	
		SUB	Binary subtraction	
		SUBC	Binary subtraction with carry	
		SUBV	Binary subtraction with underflow check	

Table 4.1 Classification of Instructions (cont)

Classification	Types	Operation Code	Function	Number of Instructions
Logic operations	6	AND	Logical AND	14
		NOT	Bit inversion	
		OR	Logical OR	
		TAS	Memory test and bit set	
		TST	Logical AND and T-bit set	
		XOR	Exclusive OR	
Shift	10	ROTL	One-bit left rotation	14
		ROTR	One-bit right rotation	
		ROTCL	One-bit left rotation with T bit	
		ROTCR	One-bit right rotation with T bit	
		SHAL	One-bit arithmetic left shift	
		SHAR	One-bit arithmetic right shift	
		SHLL	One-bit logical left shift	
		SHLLn	n-bit logical left shift	
		SHLR	One-bit logical right shift	
		SHLRn	n-bit logical right shift	
Branch	9	BF	Conditional branch, conditional branch with delay (T = 0)	11
		BT	Conditional branch, conditional branch with delay (T = 1)	
		BRA	Unconditional branch	
		BRAF	Unconditional branch	
		BSR	Branch to subroutine procedure	
		BSRF	Branch to subroutine procedure	
		JMP	Unconditional branch	
		JSR	Branch to subroutine procedure	
		RTS	Return from subroutine procedure	

Table 4.1 Classification of Instructions (cont)

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

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

Table 4.2 Instruction Code Format

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

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

Table 4.3 Data Transfer Instructions

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

Table 4.3 Data Transfer Instructions (cont)

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

Table 4.4 Arithmetic Instructions

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

Table 4.4 Arithmetic Instructions (cont)

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

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

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

Table 4.5 Logic Operation Instructions

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

Table 4.6 Shift Instructions

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

Table 4.7 Branch Instructions

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

Note: * One state when it does not branch

Table 4.8 System Control Instructions

Instruction	Operation	Code	Cycles	T Bit
CLRMAC	0 → MACH, MACL	0000000000101000	1	—
CLRT	0 → T	0000000000001000	1	0
LDC Rm,SR	Rm → SR	0100mmmm00001110	5	LSB
LDC Rm,GBR	Rm → GBR	0100mmmm00011110	3	—
LDC Rm,VBR	Rm → VBR	0100mmmm00101110	3	—
LDC.L @Rm+,SR	(Rm) → SR, Rm + 4 → Rm	0100mmmm00000111	7	LSB
LDC.L @Rm+,GBR	(Rm) → GBR, Rm + 4 → Rm	0100mmmm00010111	5	—
LDC.L @Rm+,VBR	(Rm) → VBR, Rm + 4 → Rm	0100mmmm00100111	5	—
LDS Rm,MACH	Rm → MACH	0100mmmm00001010	1	—
LDS Rm,MACL	Rm → MACL	0100mmmm00011010	1	—
LDS Rm,PR	Rm → PR	0100mmmm00101010	1	—
LDS.L @Rm+,MACH	(Rm) → MACH, Rm + 4 → Rm	0100mmmm00000110	1	—
LDS.L @Rm+,MACL	(Rm) → MACL, Rm + 4 → Rm	0100mmmm00010110	1	—
LDS.L @Rm+,PR	(Rm) → PR, Rm + 4 → Rm	0100mmmm00100110	1	—
NOP	No operation	0000000000001001	1	—
RTE	Delayed branch, SSR/SPC → SR/PC	0000000000101011	4	—
SETT	1 → T	0000000000011000	1	1
SLEEP	Sleep	0000000000011011	4*	—
STC SR,Rn	SR → Rn	0000nnnn00000010	1	—
STC GBR,Rn	GBR → Rn	0000nnnn00010010	1	—
STC VBR,Rn	VBR → Rn	0000nnnn00100010	1	—
STC.L SR,@–Rn	Rn–4 → Rn, SR → (Rn)	0100nnnn00000011	2	—
STC.L GBR,@–Rn	Rn–4 → Rn, GBR → (Rn)	0100nnnn00010011	2	—
STC.L VBR,@–Rn	Rn–4 → Rn, VBR → (Rn)	0100nnnn00100011	2	—
STS MACH,Rn	MACH → Rn	0000nnnn00001010	1	—
STS MACL,Rn	MACL → Rn	0000nnnn00011010	1	—
STS PR,Rn	PR → Rn	0000nnnn00101010	1	—

Table 4.8 System Control Instructions (cont)

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

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

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

4.3 Instructions for DSP Extension

4.3.1 Introduction

New instructions provided are classified in the following three groups:

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

1 is provided to support loop control and data transfer between CPU core registers, or memory, and new control registers added to the CPU core. Digital signal processing operations have some levels of nested loop structure. In the case of a one-level loop, it is sufficient simply to use the decrement and test, DT Rn, and conditional delayed branch BF/S instructions supported by the SH-1 and SH-2. For the nested loop, however, zero overhead loop control capability improves DSP performance.

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

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

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

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

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

Ax: 0 = R4, 1 = R5

Ay: 0 = R6, 1 = R7

Dx: 0 = X0, 1 = X1

Dy: 0 = Y0, 1 = Y1

Da: 0 = A0, 1 = A1

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

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

Note: (*) System reserved area

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

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

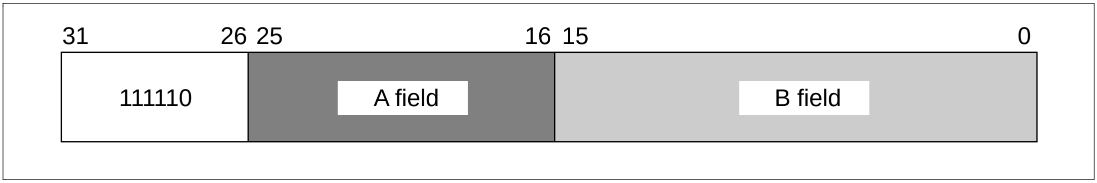


Figure 4.1 Separation of Operation Field

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

Class	Mnemonic	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S side of data transfer	NOPX	1	1	1	1	1	0	0		0		0		0	0			B field															
	MOVX.W @Ax, Dx							Ax		Dx		0		0	1																		
	MOVX.W @Ax+, Dx													1	0																		
	MOVX.W @Ax+Ix, Dx													1	1																		
	MOVX.W Da, @Ax									Da		1		0	1																		
	MOVX.W Da, @Ax+													1	0																		
	MOVX.W Da, @Ax+Ix													1	1																		
Y side of data transfer	NOPY								0		0		0			0	0	B field															
	MOVY.W @Ay, Dy								Ay		Dy		0			0	1																
	MOVY.W @Ay+, Dy															1	0																
	MOVY.W @Ay+Iy, Dy															1	1																
	MOVY.W Da, @Ay										Da		1			0	1																
	MOVY.W Da, @Ay+															1	0																
	MOVY.W Da, @Ay+Iy															1	1																

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

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

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

Note: (*1) System reserved area

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

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

Notes: (*1) System reserved area
(*2) [if cc] field can be selected from DCT (DC bit true), DCF (DC bit false) and omit (unconditional).

4.3.2 Additional System Control Instruction for CPU

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

The DSR, A0, X0, X1, Y0 and Y1 registers in the DSP engine are treated as a system register like MACH and MACL. So, STS and LDS instructions are supported for them. Digital signal processing operations have some levels of nested loop structure. So, zero overhead loop control capability improves DSP performance. The SETRC type instructions are provided to set the repeat count to RC, which is located in SR[27:16]. When the immediate operand type of SETRC is executed, 8 bits of the immediate operand data is set to SR[23:16] and zeros are set to the rest of bits, SR[27:24]. When the register operand type of SETRC instruction is executed, Rn[11:0] is set to SR[27:16]. The start address and end address of repeat loop are set to the RS and the RE registers. There are two methods for the address setting. One is to use LDC type instructions and the other is to use LDRS and LDRE instructions.

Table 4.13 Additional System Control Instructions for CPU

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

Table 4.13 Additional System Control Instructions for CPU (cont)

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

4.3.3 Single- and Double-Data Transfer for DSP Instructions

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

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

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

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

Table 4.14 Table of Double-Data Transfer Instructions

Instruction		Code	Operation	Cycle	DC
X memory no access group	NOPX	1111000*0*0*00**	X memory no access	1	—
	MOVX.W @Ax,Dx	111100A*D*0*01**	(Ax) → MSW of Dx, 0 → LSW of Dx	1	—
	MOVX.W @Ax+,Dx	111100A*D*0*10**	(Ax) → MSW of Dx, 0 → LSW of Dx, Ax + 2 → Ax	1	—
	MOVX.W @Ax+Ix,Dx	111100A*D*0*11**	(Ax) → MSW of Dx, 0 → LSW of Dx, Ax + Ix → Ax	1	—
	MOVX.W Da,@Ax	111100A*D*1*01**	MSW of Da → (Ax)	1	—
	MOVX.W Da,@Ax+	111100A*D*1*10**	MSW of Da → (Ax), Ax + 2 → Ax	1	—
	MOVX.W Da,@Ax+Ix	111100A*D*1*11**	MSW of Da → (Ax), Ax + Ix → Ax	1	—
Y memory no access group	NOPY	111100*0*0*0**00	Y memory no acess by Aa pointers	1	—
	MOVY.W @Ay,Dy	111100*A*D*0**01	(Ay) → MSW of Dy, 0 → LSW of Dy	1	—
	MOVY.W @Ay+,Dy	111100*A*D*0**10	(Ay) → MSW of Dy, 0 → LSW of Dy, Ay + 2 → Ay	1	—
	MOVY.W @Ay+Iy,Dy	111100*A*D*0**11	(Ay) → MSW of Dy, 0 → LSW of Dy, Ay + Iy → Ay	1	—
	MOVY.W Da,@Ay	111100*A*D*1**01	MSW of Da → (Ay)	1	—
	MOVY.W Da,@Ay+	111100*A*D*1**10	MSW of Da → (Ay), Ay + 2 → Ay	1	—
	MOVY.W Da,@Ay+Iy	111100*A*D*1**11	MSW of Da → (Ay), Ay + Iy → Ay	1	—

Table 4.15 Table of Single-Data Transfer Instructions

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

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

Table 4.16 Operand Flexibility of Data Transfer Instructions

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

4.3.4 Parallel Operation for the DSP Unit

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

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

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

Table 4.17 Operand Symbol of Each Group

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

Table 4.18 Operand Flexibility of Parallel Instructions

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

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

Figure 4.2 Examples of Parallel Instruction Program

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

No operation instructions, NOPX and NOPY can be omitted. Table 4.19 shows the abstract of B field of parallel operation instructions. See sections 4.3 and 4.3.3 for A field of operations.

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

Condition code bit (DC) in the DSR register is always updated based upon the result of the ALU or shift operation, if it is unconditional. If it is conditional instruction, then it does not update the DC bit. Multiply operation doesn’t affect the DC bit. Update of the DC bit depends on the states of CS0–2 bits in the DSR register. Table 4.20 shows the definition of the DC-bit update rule.

Table 4.19 Table of B Field of Parallel Operation Instructions

Instruction	Code	Operation	Cycle	DC
PMULS Se,Sf,Dg	111110********** 0100eeff0000gg00	Se * Sf → Dg (Signed)	1	—
PADD Sx,Sy,Du PMULS Se,Sf,Dg	111110********** 0111eeffxxyygguu	Sx + Sy → Du Se * Sf → Dg (Signed)	1	*
PSUB Sx,Sy,Du PMULS Se,Sf,Dg	111110********** 0110eeffxxyygguu	Sy – Sy → Du Se * Sf → Dg (Signed)	1	*
PADD Sx,Sy,Dz	111110********** 10110001xxyyzzzz	Sx + Sy → Dz	1	*
DCT PADD Sx,Sy,Dz	111110********** 10110010xxyyzzzz	If DC = 1, Sx + Sy → Dz If DC = 0, nop	1	*
DCF PADD Sx,Sy,Dz	111110********** 10110011xxyyzzzz	If DC = 0, Sx + Sy → Dz If DC = 1, nop	1	*
PSUB Sx,Sy,Dz	111110********** 10100001xxyyzzzz	Sx – Sy →Dz	1	*
DCT PSUB Sx,Sy,Dz	111110********** 10100010xxyyzzzz	If DC = 1, Sx – Sy → Dz If DC = 0, nop	1	*
DCF PSUB Sx,Sy,Dz	111110********** 10100011xxyyzzzz	If DC = 0, Sx – Sy → Dz If DC = 1, nop	1	*
PSHA Sx,Sy,Dz	111110********** 1010001xxyyzzzz	If Sy >= 0, Sx << Sy → Dz (arith. shift) If Sy < 0, Sx >> Sy → Dz	1	*
DCT PSHA Sx,Sy,Dz	111110********** 10010010xxyyzzzz	If DC = 1 & Sy >= 0, Sx << Sy → Dz (arith. shift) If DC = 1 & Sy < 0, Sx >> Sy → Dz If DC = 0, nop	1	*
DCF PSHA Sx,Sy,Dz	111110********** 10010011xxyyzzzz	If DC = 0 & Sy >= 0, Sx << Sy → Dz (arith. shift) If DC = 0 & Sy < 0, Sx >> Sy → Dz If DC = 1, nop	1	*
PSHL Sx,Sy,Dz	111110********** 10000001xxyyzzzz	If Sy >= 0, Sx << Sy → Dz (log. shift) If Sy < 0, Sx >> Sy → Dz	1	*

Note: * See table 4.20.

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

Instruction		Code	Operation	Cycle	DC
DCT	PSHL Sx,Sy,Dz	111110********** 10000010xxyyzzzz	If DC = 1 & Sy >= 0, Sx << Sy → Dz (log. shift) If DC = 1 & Sy < 0, Sx >> Sy →Dz If DC = 0, nop	1	*
DCF	PSHL Sx,Sy,Dz	111110********** 10000011xxyyzzzz	If DC = 0 & Sy >= 0, Sx << Sy → Dz (log. shift) If DC=0 & Sy<0, Sx>>Sy → Dz If DC=1, nop	1	*
	PCOPY Sx,Dz	111110********** 11011001xx00zzzz	Sx → Dz	1	*
	PCOPY Sy,Dz	111110********** 1111100100yyzzzz	Sy → Dz	1	*
DCT	PCOPY Sx,Dz	111110********** 11011010xx00zzzz	If DC = 1, Sx → Dz If DC = 0, nop	1	*
DCT	PCOPY Sy,Dz	111110********** 1111101000yyzzzz	If DC = 1, Sy → Dz If DC = 0, nop	1	*
DCF	PCOPY Sx,Dz	111110********** 11011011xx00zzzz	If DC = 0, Sx → Dz If DC = 1, nop	1	*
DCF	PCOPY Sx,Dz	111110********** 1111101100yyzzzz	If DC = 0, Sy → Dz If DC = 1, nop	1	*

Note: * See table 4.20.

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

Instruction	Code	Operation	Cycle	DC
PDMSB Sx,Dz	111110********** 10011101xx00zzzz	Count shift value for normalizing Sx → Dz	1	*
PDMSB Sy,Dz	111110********** 1011110100yyzzzz	Count shift value for normalizing Sy → Dz	1	*
DCT PDMSB Sx,Dz	111110********** 10011110xx00zzzz	If DC = 1, Count shift value for normalizing Sx → Dz If DC = 0, nop	1	*
DCT PDMSB Sy,Dz	111110********** 1011111000yyzzzz	If DC = 1, Count shift value for normalizing Sy → Dz If DC = 0, nop	1	*
DCF PDMSB Sx,Dz	111110********** 10011111xx00zzzz	If DC = 0, Count shift value for normalizing Sx → Dz If DC = 1, nop	1	*
DCF PDMSB Sy,Dz	111110********** 1011111100yyzzzz	If DC = 0, Count shift value for normalizing Sy → Dz If DC = 1, nop	1	*
PINC Sx,Dz	111110********** 10011001xx00zzzz	MSW of Sx + 1 → Dz	1	*
PINC Sy,Dz	111110********** 1011100100yyzzzz	MSW of Sy + 1 → Dz	1	*
DCT PINC Sx,Dz	111110********** 10011010xx00zzzz	If DC = 1, MSW of Sx + 1 → Dz If DC = 0, nop	1	*
DCT PINC Sy,Dz	111110********** 1011101000yyzzzz	If DC = 1, MSW of Sy + 1 → Dz If DC = 0, nop	1	*
DCF PINC Sx,Dz	111110********** 10011011xx00zzzz	If DC = 0, MSW of Sx + 1 → Dz If DC = 1, nop	1	*
DCF PINC Sy,Dz	111110********** 1011101100yyzzzz	If DC = 0, MSW of Sy + 1 → Dz If DC = 1, nop	1	*
PNEG Sx,Dz	111110********** 11001001xx00zzzz	0 – Sx → Dz	1	*
PNEG Sy,Dz	111110********** 1110100100yyzzzz	0 – Sy → Dz	1	*

Note: * See table 4.20.

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

Instruction	Code	Operation	Cycle	DC
DCT PNEG Sx,Dz	111110**********	If DC = 1, 0 – Sx → Dz	1	*
	11001010xx00zzzz	If DC = 0, nop		
DCT PNEG Sy,Dz	111110**********	If DC = 1, 0 – Sy → Dz	1	*
	1110101000yyzzzz	If DC = 0, nop		
DCF PNEG Sx,Dz	111110**********	If DC = 0, 0 – Sx → Dz	1	*
	11001011xx00zzzz	If DC = 1, nop		
DCF PNEG Sy,Dz	111110**********	If DC = 0, 0 – Sy → Dz	1	*
	1110101100yyzzzz	If DC=1, nop		
POR Sx,Sy,Dz	111110**********	Sx | Sy → Dz	1	*
	10110101xxyyzzzz			
DCT POR Sx,Sy,Dz	111110**********	If DC = 1, Sx | Sy → Dz	1	*
	10110110xxyyzzzz	If DC = 0, nop		
DCF POR Sx,Sy,Dz	111110**********	If DC = 0, Sx | Sy → Dz	1	*
	10110111xxyyzzzz	If DC = 1, nop		
PAND Sx,Sy,Dz	111110**********	Sx & Sy → Dz	1	*
	10010101xxyyzzzz			
DCT PAND Sx,Sy,Dz	111110**********	If DC = 1, Sx & Sy → Dz	1	*
	10010110xxyyzzzz	If DC = 0, nop		
DCF PAND Sx,Sy,Dz	111110**********	If DC = 0, Sx & Sy → Dz	1	*
	10010111xxyyzzzz	If DC = 1, nop		
PXOR Sx,Sy,Dz	111110**********	Sx ^ Sy → Dz	1	*
	10100101xxyyzzzz			
DCT PXOR Sx,Sy,Dz	111110**********	If DC = 1, Sx ^ Sy → Dz	1	*
	10100110xxyyzzzz	If DC = 0, nop		
DCF PXOR Sx,Sy,Dz	111110**********	If DC = 1, Sx ^ Sy → Dz	1	*
	10100111xxyyzzzz	If DC = 0, nop		

Note: * See table 4.20.

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

Instruction		Code	Operation	Cycle	DC
	PDEC Sx,Dz	111110********** 10001001xx00zzzz	Sx [39:16] – 1 → Dz	1	*
	PDEC Sy,Dz	111110********** 1010100100yyzzzz	Sy [31:16] – 1 → Dz	1	*
DCT	PDEC Sx,Dz	111110********** 10001010xx00zzzz	If DC = 1, Sx [39:16] – 1 → Dz If DC=0, nop	1	*
DCT	PDEC Sy,Dz	111110********** 1010101000yyzzzz	If DC = 1, Sy [31:16] – 1 → Dz If DC = 0, nop	1	*
DCF	PDEC Sx,Dz	111110********** 10001011xx00zzzz	If DC = 0, Sx [39:16] – 1 → Dz If DC = 1, nop	1	*
DCF	PDEC Sy,Dz	111110********** 1010101100yyzzzz	If DC = 0, Sy [31:16] – 1 → Dz If DC = 1, nop	1	*
	PCLR Dz	111110********** 100011010000zzzz	H'00000000 → Dz	1	*
DCT	PCLR Dz	111110********** 100011100000zzzz	If DC = 1, H'00000000 → Dz If DC = 0, nop	1	*
DCF	PCLR Dz	111110********** 100011110000zzzz	If DC = 0, H'00000000 → Dz If DC = 1, nop	1	*
	PSHA #Imm,Dz	111110********** 00010iiiiiiizzzz	If Imm >= 0, Dz << Imm → (arith. shift) If Imm < 0, Dz >> Imm → Dz	1	*
	PSHL #Imm,Dz	111110********** 00000iiiiiiizzzz	If Imm >= 0, Dz << Imm → Dz (logical shift) If Imm < 0, Dz >> Imm → Dz	1	*

Note: * See table 4.20.

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

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

Note: * See table 4.20.

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

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

Note: * See table 4.20.

Table 4.20 Definition of DC-Bit Update Rule

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

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

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

1	1	0	Reserved	
1	1	1	Reserved	

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

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

Condition True Case

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

Condition False Case

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

Figure 4.3   Example of Data Transfer Instruction with Condition

Instruction Code Assignment of NOPX and NOPY: There are several cases for instruction code assignment when a certain operation is unnecessary for this parallel operation. Table 4.21 shows the instruction code definition of some special cases.

Table 4.21 Instruction Code Definition of Special Cases

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

Section 5 Exception Handling

5.1 Overview

5.1.1 Types of Exception Handling and Priority Order

Exception handling is initiated by four sources: resets, address errors, interrupts, and instructions (table 5.1). When several exception sources occur simultaneously, they are accepted and processed according to the priority order shown in table 5.1.

Table 5.1 Types of Exception Handling and Priority Order

Exception	Source		Priority
Reset	Power-on reset		High
	Manual reset		
Address error	CPU address error		
	DMA address error		
Interrupt	NMI		
	User break		
	Hitachi user debug interface (H-UDI)		
	External interrupts (IRQ0–IRQ7)		
	On-chip peripheral modules	DMAC	
		SCIF0	
		SCIF1	
		SCIF2	
		ADC	
		USB	
		CMT1	
		TMU	
		WDT	
		REF	
Instructions	Trap instruction (TRAPA)		
	General illegal instructions (undefined code)		
	Illegal slot instructions (undefined code placed directly following a delayed branch instruction*[1] or instructions that rewrite the PC*[2])		Low

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

*2 Instructions that rewrite the PC: JMP, JSR, BRA, BSR, RTS, RTE, BT, BF, TRAPA, BF/S, BT/S, BSRF, BRAF

5.2 Exception Handling Operations

Exception handling sources are detected, and exception handling started, according to the timing shown in table 5.2.

Table 5.2 Timing of Exception Source Detection and Start of Exception Handling

Exception Source		Timing of Source Detection and Start of Handling
Reset	Power-on reset	The $\overline{\text{RESETP}}$ pin changes from low to high
	Manual reset	The $\overline{\text{RESETM}}$ pin changes from low to high
Address error		Detected when instruction is decoded and starts when the previous executing instruction finishes executing
Interrupts		Detected when instruction is decoded and starts when the previous executing instruction finishes executing
Instructions	Trap instruction	Starts from the execution of a TRAPA instruction
	General illegal instructions	Starts from the decoding of undefined code anytime except after a delayed branch instruction (delay slot)
	Illegal slot instructions	Starts from the decoding of undefined code placed directly following a delayed branch instruction (delay slot) or of an instruction that rewrites the PC

When exception handling starts, the CPU operates as follows:

1. Exception handling triggered by reset

   The initial values of the program counter (PC) and stack pointer (SP) are fetched from the following exception vector table.

   Power-on reset: SP = H'00000004
   PC = H'00000000

   Manual reset: SP = H'0000000C
   PC = H'00000008

   H-UDI reset: SP = H'00000004
   PC = H'00000000

   See section 5.2.1, Exception Vector Table, for more information. 0 is then written to the vector base register (VBR) and 1111 is written to the interrupt mask bits (I3–I0) of the status register (SR). The program begins running from the PC address fetched from the exception vector table.

2. Exception handling triggered by address errors, interrupts, and instructions

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

5.2.1 Exception Vector Table

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

All exception sources are given different vector numbers and vector table address offsets, from which the vector table addresses are calculated. In exception handling, the start address of the exception service routine is fetched from the exception vector table as indicated by the vector table address.

Table 5.3 lists the vector numbers and vector table address offsets. Table 5.4 shows vector table address calculations.

Table 5.3 Exception Vector Table

Exception Source		Vector Number	Vector Table Address Offset	Vector Address
Power-on reset	PC	0	H'00000000–H'00000003	Vector number × 4
	SP	1	H'00000004–H'00000007	
Manual reset	PC	2	H'00000008–H'0000000B	
	SP	3	H'0000000C–H'0000000F	
General illegal instruction		4	H'00000010–H'00000013	VBR + (vector number × 4)
(Reserved by system)		5	H'00000014–H'00000017	
Slot illegal instruction		6	H'00000018–H'0000001B	
(Reserved by system)		7	H'0000001C–H'0000001F	
		8	H'00000020–H'00000023	
CPU address error		9	H'00000024–H'00000027	
DMA address error		10	H'00000028–H'0000002B	
Interrupt	NMI	11	H'0000002C–H'0000002F	
	User break	12	H'00000030–H'00000033	
	H-UDI	13	H'00000034–H'00000037	
(Reserved by system)		14 : 31	H'00000038–H'0000003B : H'0000007C–H'0000007F	
Trap instruction (user vector)		32 : 63	H'00000080–H'00000083 : H'000000FC–H'000000FF	

Table 5.3 Exception Processing Vector Table (cont)

Exception Source		Vector Number	Vector Table Address Offset	Vector Addresses
Interrupt	IRQ0	64	H'00000100–H'00000103	
	IRQ1	65	H'00000104–H'00000107	
	IRQ2	66	H'00000108–H'0000010B	
	IRQ3	67	H'0000010C–H'0000010F	
	IRQ4	68	H'00000110–H'00000113	
	IRQ5	69	H'00000114–H'00000117	
	IRQ6	70	H'00000118–H'0000011B	
	IRQ7	71	H'0000011C–H'0000011F	
(Reserved for system use)		72 : 127	H'00000120–H'00000123 : H'000001FC–H'000001FF	
Interrupt	On-chip peripheral module	128 : 255	H'00000200–H'00000203 : H'000003FC–H'000003FF	

The vector numbers and vector table address offsets of the on-chip peripheral module interrupts are listed in table 8.3 in section 8, Interrupt Controller (INTC).

Table 5.4 Calculating Exception Vector Table Addresses

Exception Source	Vector Table Address Calculation	
Power-on reset Manual reset	(Vector table address)	= (vector table address offset) = (vector number) × 4
Other exception handling	(Vector table address)	= VBR + (vector table address offset) = VBR + (vector number) × 4

Note: VBR: Vector base register

Vector table address offset: See table 5.3.

Vector number: See table 5.3.

5.3 Resets

5.3.1 Types of Resets

Resets have the highest priority of any exception source. There are three types of resets: manual, power-on, and H-UDI. The reset sequence is used to turn on the power supply to the SH7622 in its initial state, or to perform a restart. The reset signals are sampled every clock cycle. In the case of a power-on reset, all processing being executed is stopped, all unfinished events are canceled, and reset processing is executed immediately. In a manual reset, however, processing to retain the contents of external memory continues. As shown in table 5.5, the internal state of the CPU is initialized in both a power-on reset and a manual reset. For details of H-UDI resets, see section 24.4.3.

Table 5.5 Types of Resets

	Conditions for	Internal Status	
Type	**Transition to Reset Status**	**CPU**	**On-Chip Peripheral Modules**
Power-on reset	$\overline{\text{RESETP}}$ = L	Initialized	Initialized
Manual reset	$\overline{\text{RESETM}}$ = L	Initialized	See the register configurations in the relevant sections
H-UDI reset	H-UDI reset command input	Initialized	See the register configurations in the relevant sections

5.3.2 Power-On Reset

When the $\overline{\text{RESETP}}$ pin is high, the device performs a power-on reset. During a power-on reset, the CPU’s internal state and all on-chip peripheral module registers are initialized. See appendix B, Pin States, for the state of individual pins in the power-on reset state.

The CPU will then operate as follows:

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

5.3.3 Manual Reset

When the $\overline{\text{RESETM}}$ pin is low, the device executes a manual reset. For a reliable reset, the $\overline{\text{RES}}$ pin should be kept low for at least 20 clock cycles. During a manual reset, the CPU’s internal state and all on-chip peripheral module registers are initialized. See the relevant sections for further details. When the chip enters the manual reset state in the middle of a bus cycle, manual reset exception handling does not start until the bus cycle has ended. Thus, manual resets do not abort bus cycles. See appendix B, Pin States, for the state of individual pins in the manual reset state.

The CPU will then operate in the same way as for a power-on reset.

5.4 Address Errors

5.4.1 Sources of Address Errors

Address errors are of the following two kinds.

1. CPU address error: An address error when the CPU is the bus master. This kind of error occurs in the following cases in instruction fetch or data read/write operations:
   a. Instruction fetch from an odd address (4n+1, 4n+3)
   b. Word data access from other than a word boundary (4n+1, 4n+3)
   c. Longword data access from other than a longword boundary (4n+1, 4n+2, 4n+3)
2. DMA address error: An address error when the DMAC is the bus master. This kind of error occurs in the following cases in data read/write operations:
   a. Word data access from other than a word boundary (4n+1, 4n+3)
   b. Longword data access from other than a longword boundary (4n+1, 4n+2, 4n+3)

5.4.2 Address Error Exception Handling

When an address error occurs, address error exception handling begins after the end of the bus cycle in which the error occurred and completion of the executing instruction. The CPU operates as follows:

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

5.5 Interrupts

5.5.1 Interrupt Sources

Table 5.6 shows the sources that initiate interrupt exception handling. These are divided into NMI, user breaks, H-UDI, IRQ, and on-chip peripheral modules.

Table 5.6 Types of Interrupt Sources

Type	Request Source	Number of Sources
NMI	NMI pin (external input)	1
User break	User break controller (UBC)	1
H-UDI	Hitachi user debug interface (H-UDI)	1
IRQ	IRQ0–IRQ7 (external input)	8
On-chip peripheral module	DMAC	4
	SCIF0	2
	SCIF1	2
	SCIF2	4
	ADC	1
	USB	2
	CMT1	1
	TMU	4
	WDT	1
	REF	2

Each interrupt source is allocated a different vector number and vector table address offset. See section 8, Interrupt Controller (INTC), for more information.

5.5.2 Interrupt Priority Levels

The interrupt priority order is predetermined. When multiple interrupts occur simultaneously, the interrupt controller (INTC) determines their relative priorities and begins exception handling accordingly.

The priority order of interrupts is expressed as priority levels 0–16, with priority 0 the lowest and priority 16 the highest. The NMI interrupt has priority 16 and cannot be masked, so it is always accepted. The user break interrupt priority level is 15 and IRL interrupts have priorities of 1–15. On-chip peripheral module interrupt priority levels can be set freely using the INTC's interrupt priority level setting registers A–H (IPRA–IPRH). The priority levels that can be set are 0–15. Level 16 cannot be set.

5.5.3 Interrupt Exception Handling

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

When an interrupt is accepted, exception handling begins. In interrupt exception handling, the CPU saves SR and the program counter (PC) to the stack. The priority level value of the accepted interrupt is written to SR bits I3–I0. For NMI, however, the priority level is 16, but the value set in I3–I0 is 1111. Next, the start address of the exception service routine is fetched from the exception vector table for the accepted interrupt, that address is jumped to and execution begins.

5.6 Exceptions Triggered by Instructions

5.6.1 Instruction-Triggered Exception Types

Exception handling can be triggered by a trap instruction, general illegal instruction or illegal slot instruction, as shown in table 5.7.

Table 5.7 Types of Exceptions Triggered by Instructions

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

5.6.2 Trap Instructions

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

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

5.6.3 Illegal Slot Instructions

An Illegal slot Instruction exception occurs in the following cases.

a. Decoding of an undefined instruction in a delay slot
Delayed branch instructions: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT/S, BF/S

b. Decoding of an instruction that rewrites the PC in a delay slot
Instructions that rewrite the PC: JMP, JSR, BRA, BRAF, BSR, BSRF, RTS, RTE, BT, BF, BT/S, BF/S, TRAPA

The CPU handles an illegal slot instruction as follows:

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

5.6.4 General Illegal Instructions

A general Illegal Instruction exception occurs in the following cases.

a. Decoding of an undefined instruction not in a delay slot
Undefined instruction: H'Fxxx, excluding DSP instructions

The operation of undefined instructions with a code other than this cannot be guaranteed.

b. Decoding of an instruction that rewrites the PC/SR/RS/RE in the last three instructions of a repeat loop

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

In the case of a general illegal exception, the CPU operates in the same way as for an Illegal slot Instruction exception. However, unlike the case of an Illegal slot Instruction exception, the PC value saved is the start address of the undefined instruction code.

5.7 When Exception Sources are Not Accepted

When an address error or interrupt is generated after a delayed branch instruction or interrupt-disabled instruction, it is sometimes not immediately accepted but is stored instead, as described in table 5.8. When this happens, it will be accepted when an instruction for which exception acceptance is possible is decoded.

Table 5.8 Exception Source Generation Immediately after a Delayed Branch Instruction or Interrupt-Disabled Instruction

	Exception Source	
Point of Occurrence	**Address Error**	**Interrupt**
Immediately after a delayed branch instruction*[1]	Not accepted	Not accepted
Immediately after an interrupt-disabled instruction*[2]	Accepted	Not accepted
A repeat loop comprising up to three instructions (instruction fetch cycle not generated) First instruction or last three instructions in a repeat loop containing four or more instructions	Not accepted	Not accepted
Fourth from last instruction in a repeat loop containing four or more instructions	Accepted	Not accepted

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

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

5.7.1 Immediately after a Delayed Branch Instruction

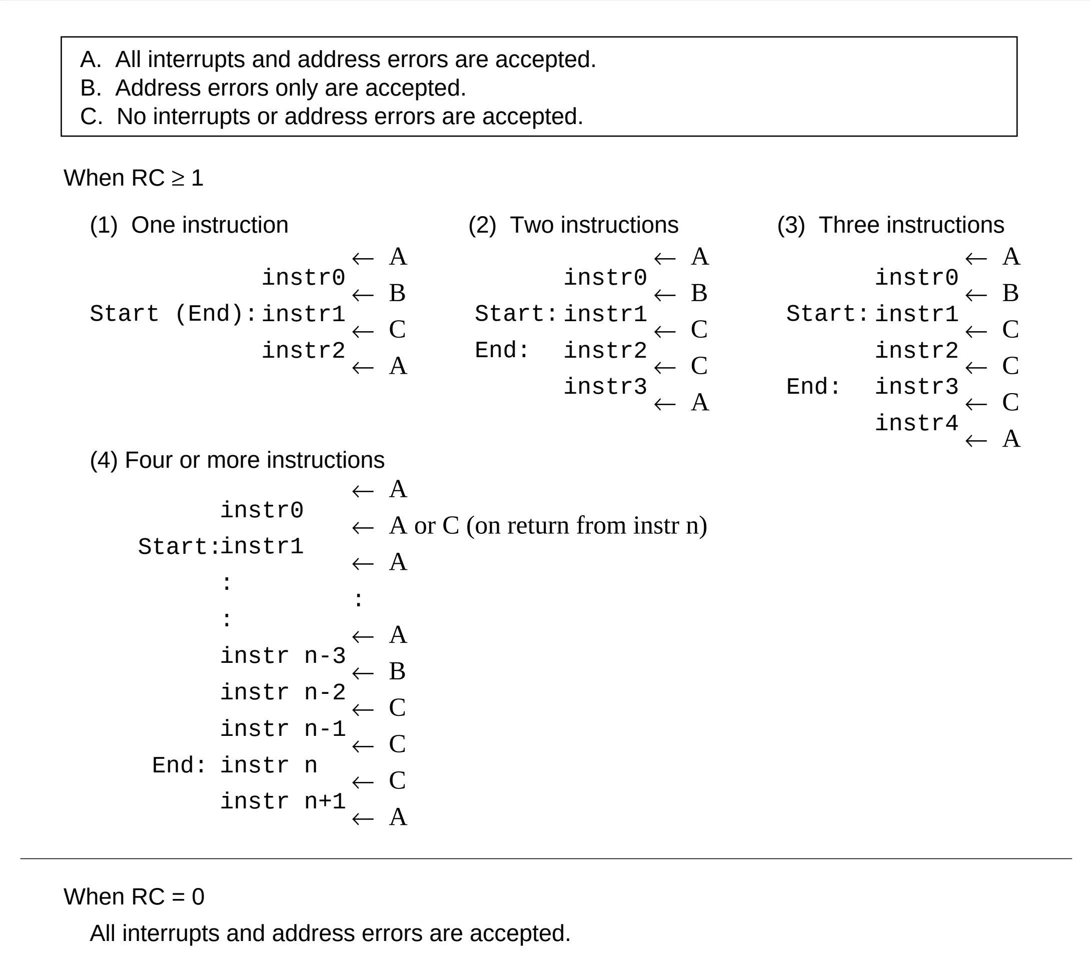
When an instruction placed immediately after a delayed branch instruction (delay slot) is decoded, neither address errors nor interrupts are accepted. The delayed branch instruction and the instruction located immediately after it (delay slot) are always executed consecutively, so no exception handling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

5.7.2 Immediately after an Interrupt-Disabled Instruction

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

5.7.3 Instructions in Repeat Loops

If a repeat loop comprises up to three instructions, neither exceptions nor interrupts are accepted. If a repeat loop contains four or more instructions, neither exceptions nor interrupts are accepted during the execution cycle of the first instruction or the last three instructions. If a repeat loop contains four or more instructions, address errors only are accepted during the execution cycle of the fourth from last instruction. For more information, see the *SH-1, SH-2, SH-DSP Programming Manual*.



A. All interrupts and address errors are accepted.
B. Address errors only are accepted.
C. No interrupts or address errors are accepted.

When RC ≥ 1

(1) One instruction

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

(2) Two instructions

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

(3) Three instructions

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

(4) Four or more instructions

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

When RC = 0

All interrupts and address errors are accepted.

Figure 5.1 Interrupt Acceptance Restrictions in Repeat Mode

5.8 Stack Status after Exception Handling

The status of the stack after exception handling ends is as shown in table 5.9.

Table 5.9 Stack Status After Exception Handling

Type	Stack Status		
Address error	SP →	Address of instruction after executed instruction	32 bits
		SR	32 bits
Trap instruction	SP →	Address of instruction after TRAPA instruction	32 bits
		SR	32 bits
General illegal instruction	SP →	Start address of illegal instruction	32 bits
		SR	32 bits
Interrupt	SP →	Address of instruction after executed instruction	32 bits
		SR	32 bits
Illegal slot instruction	SP →	Jump destination address of delayed branch instruction	32 bits
		SR	32 bits

5.9 Usage Notes

5.9.1 Value of Stack Pointer (SP)

The value of the stack pointer must always be a multiple of four, otherwise operation cannot be guaranteed.

5.9.2 Value of Vector Base Register (VBR)

The value of the vector base register must always be a multiple of four, otherwise operation cannot be guaranteed.

5.9.3 Manual Reset During Register Access

Do not assert the manual reset signal during access to a register whose value is retained in a manual reset, as this may result in a write error.

Section 6 Cache

6.1 Overview

6.1.1 Features

The cache specifications are listed in table 6.1.

Table 6.1 Cache Specifications

Parameter	Specification
Capacity	8 kbytes
Structure	Instruction/data mixed, 4-way set associative
Line size	16 bytes
Number of entries	128 entries/way
Write system	P0, P1, P3: Write-back/write-through selectable
Replacement method	Least-recently-used (LRU) algorithm

In the SH7622, the address space is partitioned into five subdivisions, and the cache access method is determined by the address. Table 6.2 shows the kind of cache access available in each address space subdivision.

Table 6.2 Address Space Subdivisions and Cache Operation

Address Bits A31 to 29	Address Space Subdivision	Cache Operation
0xx	P0	Write-back/write-through selectable
100	P1	Write-back/write-through selectable
101	P2	Non-cacheable
110	P3	Write-back/write-through selectable
111	P4	I/O area, non-cacheable

Note that area P4 is an I/O area, to which the addresses of on-chip registers, etc., are assigned.

To ensure data consistency, the cache stores 32-bit addresses with the upper 3 bits masked to 0.

6.1.2 Cache Structure

The cache mixes data and instructions and uses a 4-way set associative system. It is composed of four ways (banks), each of which is divided into an address section and a data section. Each of the address and data sections is divided into 128 entries. The data section of the entry is called a line. Each line consists of 16 bytes (4 bytes × 4). The data capacity per way is 2 kbytes (16 bytes × 128 entries), with a total of 8 kbytes in the cache as a whole (4 ways). Figure 6.1 shows the cache structure.

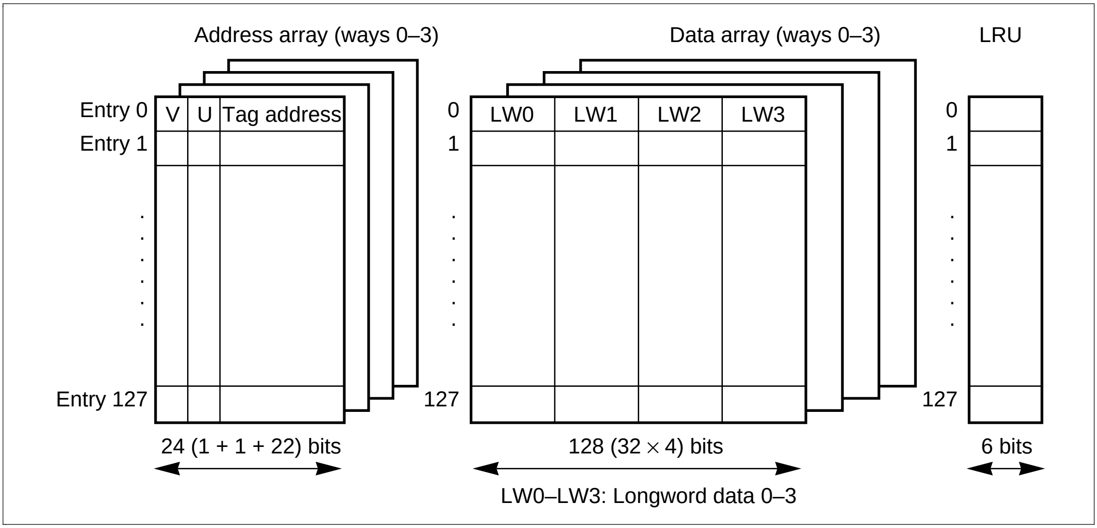


Figure 6.1 Cache Structure

Address Array: The V bit indicates whether the entry data is valid. When the V bit is 1, data is valid; when 0, data is not valid. The U bit indicates whether the entry has been written to in write-back mode. When the U bit is 1, the entry has been written to; when 0, it has not. The address tag holds the physical address used in the external memory access. It is composed of 22 bits (address bits 31–10) used for comparison during cache searches.

In the SH7622, the top three of 32 physical address bits are used as shadow bits (see section 13, Bus State Controller (BSC)), and therefore in a normal replace operation the top three bits of the tag address are cleared to 0.

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

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

LRU: With the 4-way set associative system, up to four instructions or data with the same entry address (address bits 10–4) can be registered in the cache. When an entry is registered, the LRU shows which of the four ways it is recorded in. There are six LRU bits, controlled by hardware. A least-recently-used (LRU) algorithm is used to select the way.

In normal operation, four ways are used as cache and six LRU bits indicate the way to be replaced (table 6.2). If a bit pattern other than those listed in table 6.2 is set in the LRU bits by software, the cache will not function correctly. When modifying the LRU bits by software, set one of the patterns listed in table 6.2.

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

Table 6.3 LRU and Way Replacement

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

6.1.3 Register Configuration

Table 6.4 shows details of the cache control register.

Table 6.4 Register Configuration

Register	Abbr.	R/W	Initial Value	Address	Access Size
Cache control register	CCR	R/W	H'00000000	H'FFFFFFEC	32

6.2 Register Description

6.2.1 Cache Control Register (CCR)

The cache is enabled or disabled using the CE bit of the cache control register (CCR). CCR also has a CF bit (which invalidates all cache entries), and a WT and CB bits (which select either write-through mode or write-back mode). Programs that change the contents of the CCR register should be placed in address space that is not cached. When updating the contents of the CCR register, always set bits 4 and 5 to 0. Figure 6.2 shows the configuration of the CCR register. Set the CF bit before setting other bits in the CCR register.

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

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

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

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

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

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

Bits 31–6: Reserved (always set to 0.)

Figure 6.2 CCR Register Configuration

6.3 Cache Operation

6.3.1 Searching the Cache

If the cache is enabled, whenever instructions or data in memory are accessed the cache will be searched to see if the desired instruction or data is in the cache. Figure 6.3 illustrates the method by which the cache is searched. The cache is a physical cache and holds physical addresses in its address section.

Entries are selected using bits 10–4 of the address (virtual) of the access to memory and the address tag of that entry is read. In parallel to reading of the address tag, the virtual address is translated to a physical address in the MMU. The physical address after translation and the physical address read from the address section are compared. The address comparison uses all four ways. When the comparison shows a match and the selected entry is valid (V = 1), a cache hit occurs. When the comparison does not show a match or the selected entry is not valid (V = 0), a cache miss occurs. Figure 6.3 shows a hit on way 1.

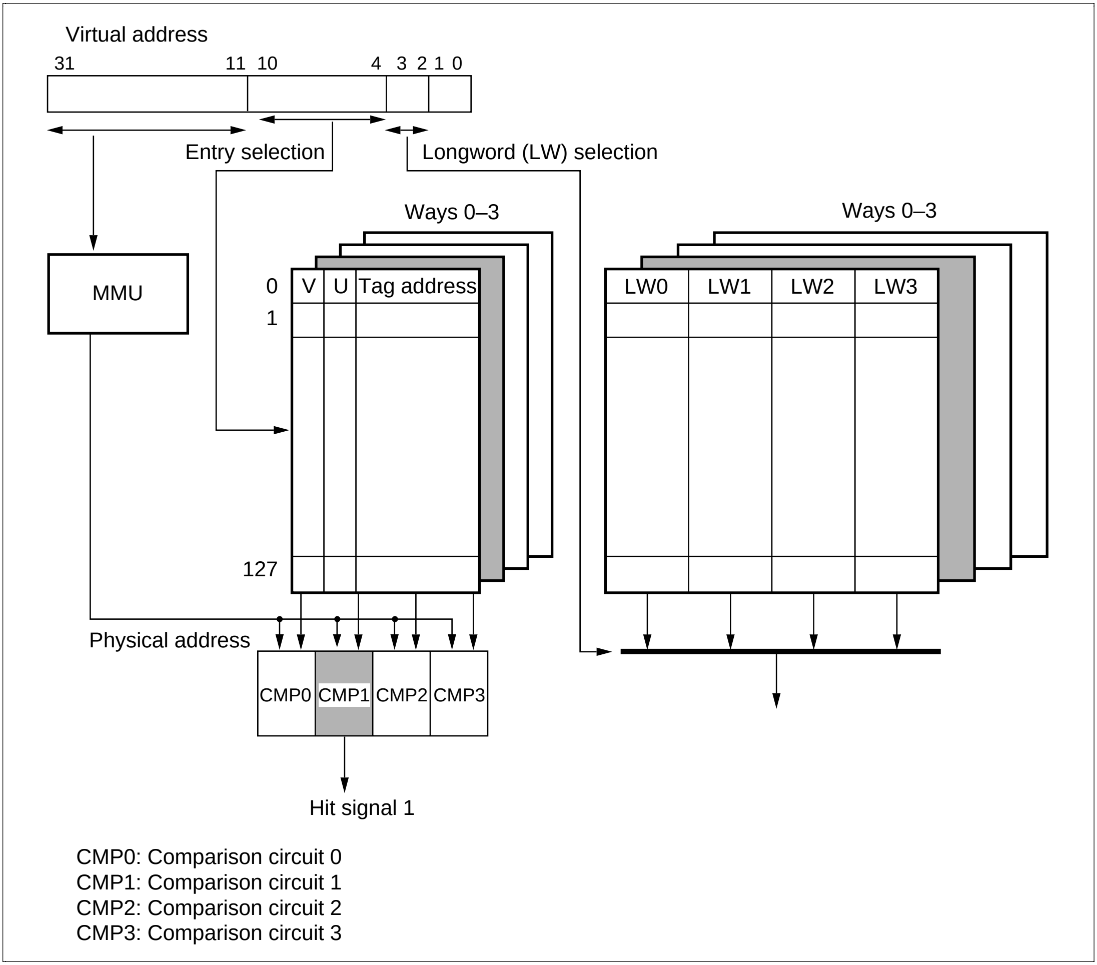


Figure 6.3 Cache Search Scheme

6.3.2 Read Access

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

Read Miss: An external bus cycle starts and the entry is updated. The way replaced is the one least recently used. Entries are updated in 16-byte units. When the desired instruction or data that caused the miss is loaded from external memory to the cache, the instruction or data is transferred to the CPU in parallel with being loaded to the cache. When it is loaded in the cache, the U bit is cleared to 0 and the V bit is set to 1. When the U bit of a replaced entry in write-back mode is 1, the cache fill cycle starts after the entry is transferred to the write-back buffer. After the cache completes its fill cycle, the write-back buffer writes back the entry to the memory. The write-back unit is 16 bytes.

6.3.3 Write Access

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

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

6.3.4 Write-Back Buffer

When the U bit of the entry to be replaced in the write-back mode is 1, it must be written back to the external memory. To increase performance, the entry to be replaced is first transferred to the write-back buffer and fetching of new entries to the cache takes priority over writing back to the external memory. During the write back cycles, the cache can be accessed. The write-back buffer can hold one line of the cache data (16 bytes) and its physical address. Figure 6.4 shows the configuration of the write-back buffer.

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

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

Figure 6.4 Write-Back Buffer Configuration

6.3.5 Coherency of Cache and External Memory

Use software to ensure coherency between the cache and the external memory. When memory shared by the SH7622 and another device is accessed, the latest data may be in a write-back mode cache, so invalidate the entry that includes the latest data in the cache, generate a write back, and update the data in memory before using it. When the caching area is updated by a device other than the SH7622, invalidate the entry that includes the updated data in the cache.

6.4 Memory-Mapped Cache

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

6.4.1 Address Array

The address array is mapped onto H'F0000000 to H'F0FFFFFF. To access an address array, the 32-bit address field (for read/write accesses) and 32-bit data field (for write accesses) must be specified. The address field specifies information for selecting the entry to be accessed; the data field specifies the address, V bit, U bit, and LRU bits to be written to the address array (figure 6.5. (1)).

In the address field, specify the entry address selecting the entry (bits 11–4), W for selecting the way (bits 12–11: in normal mode (8-kbyte cache), 00 is way 0, 01 is way 1, 10 is way 2, and 11 is way 3), and H'F0 to indicate address array access (bits 31–24).

When writing, specify bit 3 as the A bit. The A bit indicates whether addresses are compared during writing. When the A bit is 1, the addresses of four entries selected by the entry addresses are compared to the addresses to be written into the address array specified in the data field. Writing takes place to the way that has a hit. When a miss occurs, nothing is written to the address array and no operation occurs. The way number (W) specified in bits 12–11 is not used. When the A bit is 0, it is written to the entry selected with the entry address and way number without comparing addresses. The address specified by bits 31–10 in the data specification in figure 6.5. (1), address array access, is a virtual address. When the MMU is enabled, the address is translated into a physical address, then the physical address is used in comparing addresses when the A bit is 1. The physical address is written into the address array.

When reading, the address tag, V bit, U bit, and LRU bits of the entry specified by the entry address and way number (W) are read using the data format shown in figure 6.5 without comparing addresses. To invalidate a specific entry, specify the entry by its entry address and way number, and write 0 to its V bit. To invalidate only an entry for an address to be invalidated, specify 1 for the A bit.

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

6.4.2 Data Array

The data array is mapped onto H'F1000000 to H'F1FFFFFF. To access a data array, the 32-bit address field (for read/write accesses) and 32-bit data field (for write accesses) must be specified. The address field specifies information for selecting the entry to be accessed; the data field specifies the longword data to be written to the data array (figure 6.5. (2)).

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

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

(1) Address array access

Address specification

Read access

31 24	23 14	13 12	11	10 4	3	2 0
1111 0000	*············*	W	X	Entry	0	* * *

Write access

31 24	23 14	13 12	11	10 4	3	2 0
1111 0000	*············*	W	X	Entry	A	* * *

Data specification

31	30	29	10	9 4	3 2	1	0
0	0	0	Address tag (28–10)	LRU	X X	U	V

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

Address specification

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

Data specification

31 0
Longword

X: 0 for read and write
*: Don’t care bit

Figure 6.5 Specifying Address and Data for Memory-Mapped Cache Access

6.5 Usage Examples

6.5.1 Invalidating Specific Entries

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

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

6.5.2 Reading the Data of a Specific Entry

This example reads the data section of a specific cache entry. The longword indicated in the data field of the data array in figure 6.5 is read to the register. R0 specifies the address and R1 is read.

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

6.5.3 Usage Notes

1. Caching may not be possible when all the following conditions are met.

(1) The cache is used in write-back mode.

(2) Sequence that satisfies all the following conditions:

- (a) Instruction at which a cache access is missed (read miss or write miss) and a write-back to the write-back buffer occurs as a result
- (b) Instruction that accesses X/Y memory (read or write) several instructions after (a) (excluding a DSP instruction other than a MOVS instruction)
- (c) Instruction for which the store destination is a cache write hit access immediately after the instruction in (b)
- (d) (b) is located at address 4n, and (c) is located at address 4n+2.

This problem can be avoided in either of the following ways.

(1) When X/Y memory is used with the cache on, use write-through mode.

(2) When the cache is used in write-back mode and X/Y memory is used, provide a software countermeasure as follows.

   (a) Use a MOVX, MOVY, or similar instruction for X/Y memory access rather than a MOV or MOVS instruction.

   (b) If a MOV or MOVS instruction is used for X/Y memory access, insert a NOP instruction immediately afterward.

Address	Instruction
4n	Instruction that accesses X/Y memory address
4n + 2	NOP <- addition
4n + 4	Instruction for which store destination is cache write hit access

There is no problem with using write-back mode as long as X/Y memory addresses are not accessed.

2. Caching may not be possible when all the following conditions are met.

(1) The cache is used in write-back mode.

(2) Data transfer is executed using DMA.

(3) A forced purge or associative purge of the cache* is executed during DMA transfer.

This problem can be avoided in any of the following ways.

(1) Use the cache in write-through mode, or do not use DMA.

(2) Do not use forced purges or associative purges of the cache.

(3) When using DMA in write-back mode and performing a forced purge or associative purge of the cache, first disable DMA and also check that the transfer has ended before performing the purge. The execution procedure is shown below.

   <Sample countermeasure>

   1. Write 0 to DMAOR.DME
   2. Read DMAOR.DME
   3. Wait until the value read above is 0
   4. Execute forced purge or associative purge

Note: * Clearing a V bit in the cache address array to invalidate the relevant entry is called a purge. In a purge, the method whereby the A bit is cleared to 0 and the entry address and way to be invalidated are specified directly is called a forced purge, while the method whereby the A bit is set to 1 and the entry address and tag address are specified, and the way matched by the tag address is searched for and a purge performed, is called an associative purge. When an entry for which the U bit is 1 is purged, a write-back is performed.

Section 7 X/Y Memory

7.1 Overview

The SH7622 has on-chip X-RAM and Y-RAM. It can be used by CPU, DSP and DMAC to store instructions or data.

7.1.1 Features

The X-/Y-Memory features are listed in table 7.1.

Table 7.1 X-/Y-Memory Specifications

Parameter	Features
Addressing method	Mapping is possible in area P0 or P2 Do not perform mapping in area P1, P3, or P4, as operation is not guaranteed in this case
Ports	3 independent read/write ports • 8-/16-/32-bit access from the CPU (via L bus or I bus) • Maximum of two simultaneous 16-bit accesses (via X and Y buses), or 16/32-bit accesses, from the DSP (via L bus) • 8-/16-/32-bit access from the DMAC (via I bus)
Size	8-kbyte RAM for X and Y memory each

7.2 X-/Y-Memory Access from the CPU

The X/Y memory can be fixed-mapped in the range from H'0500 0000 to H'05FF FFFF in space P0 or from H'A500 0000 to H'A5FF FFFF in space P2. Reading can be performed by one-cycle access, and writing by two-cycle access.

The X/Y memory resides in addresses H'0500 0000 to H'0501 FFFF in area 1 (64 Mbytes), including reserved space. The areas available in the SH7622 are fixed at H'0500 7000 to H'0500 8FFF (8 kbytes) for X-RAM and H'0501 7000 to H'0501 8FFF (8 kbytes) for Y-RAM. The X- and Y-RAM are divided into page 0 and page 1 according to the addresses. The remainder is reserved space. Only spaces P0 and P2 can be used. Operation cannot be guaranteed if reserved space is accessed. In this way, the X/Y memory can be accessed from the L bus, I bus, X bus, and Y bus.

In the event of simultaneous accesses to the same address from different buses, the priority order is as follows: I bus > L bus > X and Y buses. Since this kind of contention tends to lower X/Y memory accessibility, it is advisable to provide software measures to prevent such contention as far as possible. For example, contention will not arise if different memory or different pages are accessed by each bus. Access from the CPU is via the I bus when X/Y memory is space P0, and via the L bus when space P2. Figure 7.1 shows X-/Y-memory mapping.

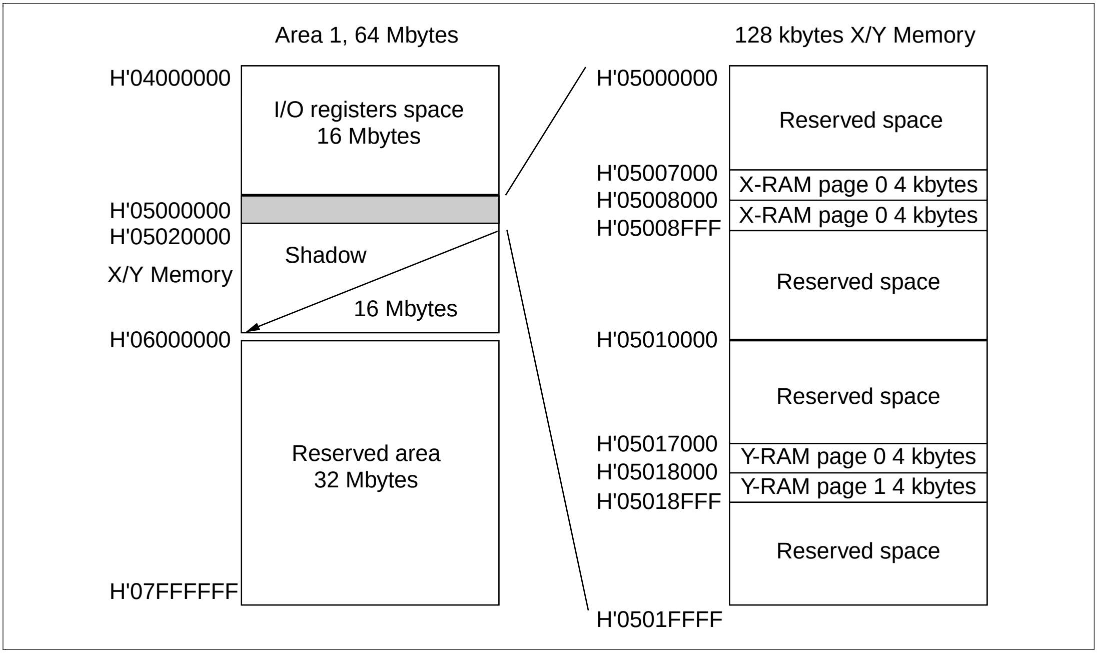


Figure 7.1 X-/Y-Memory Address Mapping

7.3 X-/Y-Memory Access from the DSP

The X/Y memory can be accessed by the DSP through the X bus and Y bus. Accesses via the X bus/Y bus are always 16-bit, while accesses via the L bus are either 16-bit or 32-bit. With X data transfer instructions and Y-data-transfer instructions, the X/Y memory is accessed via the X bus or Y bus. These accesses are always 16-bit.

In the case of a single data transfer instruction, the X/Y memory is accessed via the L bus. In this case the access is either 16-bit or 32-bit. Accesses via the X bus and Y bus cannot be specified simultaneously.

7.4 X-/Y-Memory Access from the DMAC

The X/Y memory also exists on the I bus and can be accessed by the DMAC. However, when the X/Y memory is designated as the transfer source or transfer destination in DMA transfer, ensure that the X/Y memory is not accessed using the CPU or DSP during DMA transfer.

7.5 Usage Note

When accessing the X/Y memory, if the cache is on, access must be performed from space P2 (non-cacheable space). When the cache is off, spaces P0 and P2 can both be used.

Section 8 Interrupt Controller (INTC)

8.1 Overview

The interrupt controller (INTC) ascertains the priority order of interrupt sources and controls interrupt requests to the CPU. The INTC has registers for setting the priority of each interrupt which allows the user to set the order of priority in which interrupt requests are handled.

8.1.1 Features

The INTC has the following features:

- IRQ mode

  Eight external signals function as independent interrupt sources (IRQ0 to IRQ7). Each interrupt source has an interrupt vector, and can be assigned a priority level.
- 16 interrupt priority levels can be set.

  By setting the eight interrupt priority registers, the priorities of IRQ0 to IRQ7 and on-chip peripheral module interrupts can be selected from 16 levels for different request sources.
- Vector assignment method

  An auto-vector method is used, in which vector numbers are decided internally.
- IRQ interrupt settings can be made (low-level, rising-edge, or falling-edge detection).

8.1.2 Pin Configuration

Table 8.1 shows the INTC pin configuration.

Table 8.1 Pin Configuration

Name	I/O	Function
NIM	I	Input of nonmaskable interrupt request signal
IRQ7 to IRQ0	I	Input of maskable interrupt request signals
$\overline{\text{IRQOUT}}$	O	Notifies an external device of the occurrence of an interrupt source or memory refresh request

8.1.3 Register Configuration

The INTC has the 10 registers shown in table 8.2. These registers perform various INTC functions including setting interrupt priority, and controlling external interrupt input signal detection.

Table 8.2 Register Configuration

Name	Abbr.	R/W	Initial Value	Address	Access Size
Interrupt priority register setting register A	IPRA	R/W	H'0000	H'A4002080	16
Interrupt priority register setting register B	IPRB	R/W	H'0000	H'A4002082	16
Interrupt priority register setting register C	IPRC	R/W	H'0000	H'A4002084	16
Interrupt priority register setting register D	IPRD	R/W	H'0000	H'A4002086	16
Interrupt priority register setting register E	IPRE	R/W	H'0000	H'A4002088	16
Interrupt priority register setting register G	IPRG	R/W	H'0000	H'A400208C	16
Interrupt priority register setting register H	IPRH	R/W	H'0000	H'A400208E	16
Interrupt control register 0	ICR0	R/W	H'0000/ H'8000*	H'A4002090	16
Interrupt control register 1	ICR1	R/W	H'0000	H'A4002092	16
Interrupt request register	IRR	R/W	H'0000	H'A4002094	16

Note: * H'8000 when the NMI pin is high, H'0000 when the NMI pin is low.

8.2 Interrupt Sources

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

8.2.1 NMI Interrupt

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

8.2.2 User Break Interrupt

A user break interrupt has priority level 15 and occurs when the break condition set in the user break controller (UBC) is satisfied. User break interrupt exception handling sets the interrupt mask level bits (I3–I0) in the status register (SR) to level 15. For more information about the user break interrupt, see section 9, User Break Controller.

8.2.3 H-UDI Interrupt

The H-UDI interrupt has a priority level of 15, and is generated when an H-UDI interrupt instruction is serially input. H-UDI interrupt exception processing sets the interrupt mask bits (I3–I0) in the status register (SR) to level 15. See section 24, Hitachi User Debug Interface, for details of the H-UDI interrupt.

8.2.4 IRQ Interrupts

Each IRQ interrupt corresponds to input at one of pins IRQ0 to IRQ7. Pins IRQ0 to IRQ7 can be used when IRQE is set to 1 in ICR0. Low-level, rising-edge, or falling-edge detection can be selected individually for each pin by means of IRQ sense select bits 7–0 in ICR1. Using IPRA and IPRB, priority levels 0–15 can be selected individually for each pin.

In IRQ interrupt exception handling, the interrupt mask bits (I3–I0) in SR are set to the priority level of the accepted IRQ interrupt.

To clear IRQ interrupt input with an edge, read 1 from the corresponding bit in IRR, then write 0 to the bit.

When a rewrite is performed on the ICR1 register, an IRQ interrupt may be erroneously detected, depending on the pin states. To prevent this, perform register rewriting when interrupts are masked, clear the illegal interrupt by writing 0 to IRR, and then release the mask.

To detect an edge input interrupt, a pulse with a width of more than two P-clock cycles must be input.

With level detection, the level must be maintained until the interrupt is accepted and the CPU starts interrupt handling.

An IRQ interrupt cannot be used to recover from the standby state.

These bits have an initial value of 0, and must be set by software as follows.

1. Reset start
2. Set the IRQE bit to 1.
3. Select "other function" (IRQ) with the PFC.

The system design must provide for unused IRQ pins to be handled (pulled up, etc.) externally, or set as general output ports by the PFC, and for these pins not to be affected by noise, etc.

IRQ interrupts are detected even if the general port function is set for pins IRQ0 to IRQ7. Therefore, when using pins IRQ0 to IRQ7 solely as general ports, the mask state should be set for the interrupt priority level.

8.2.5 On-Chip Peripheral Module Interrupts

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

- USB function module (USB)
- Direct memory access controller (DMAC)
- Serial communication interface with FIFO (SCIF0, SCIF1, SCIF2)
- Refresh controller (REF)
- Timer unit (TMU)
- Watchdog timer (WDT)
- Memory stick interface (MSIF)
- Hitachi user debug interface (H-UDI)
- A/D converter (ADC)
- Compare match timer 1 (CMT1)

A different interrupt vector is assigned to each interrupt source, so the exception service routine does not have to decide which interrupt has occurred. Priority levels between 0 and 15 can be assigned to individual on-chip peripheral modules in interrupt priority registers A–H (IPRA–IPRH). On-chip peripheral module interrupt exception handling sets the interrupt mask level bits (I3–I0) in the status register (SR) to the priority level value of the on-chip peripheral module interrupt that was accepted.

8.2.6 Interrupt Exception Vectors and Priority Order

Table 8.3 lists interrupt sources and their vector numbers, vector table address offsets and interrupt priorities.

Each interrupt source is allocated a different vector number and vector table address offset. Vector table addresses are calculated from vector numbers and vector table address offsets. In interrupt exception handling, the exception service routine start address is fetched from the vector table entry indicated by the vector table address. See table 5.4, Calculating Exception Vector Table Addresses, in section 5, Exception Handling, for more information on this calculation.

IRQ interrupt and on-chip peripheral module interrupt priorities can be set freely between 0 and 15 for each module by setting interrupt priority registers A–H (IPRA–IPRH). The ranking of interrupt sources for IPRA–IPRH, however, must be the order listed under Priority within IPR Setting Unit in table 8.3 and cannot be changed. A reset assigns priority level 0 to on-chip peripheral module interrupts. If the same priority level is assigned to two or more interrupt sources and interrupts from those sources occur simultaneously, their priority order is the default priority order indicated at the right in table 8.3.

Table 8.3 Interrupt Exception Vectors and Priority Order

Interrupt Source	Interrupt Type	Interrupt Priority Order (Initial Value)	IPR	Vector No.	Vector Table Address	Priority within IPR Setting Unit	Default Priority
NMI	16		—	11	VBR + (vecter No. × 4)	—	High
User break	15		—	12		—	
H-UDI	15		—	13		—	
IRQ0	0 to 15 (0)		IPRA (15 to 12)	64			
IRQ1	0 to 15 (0)		IPRA (11 to 8)	65			
IRQ2	0 to 15 (0)		IPRA (7 to 4)	66			
IRQ3	0 to 15 (0)		IPRA (3 to 0)	67			
IRQ4	0 to 15 (0)		IPRB (15 to 12)	68			
IRQ5	0 to 15 (0)		IPRB (11 to 8)	69			
IRQ6	0 to 15 (0)		IPRB (7 to 4)	70			
IRQ7	0 to 15 (0)		IPRB (3 to 0)	71			Low

Table 8.3 Interrupt Exception Vectors and Priority Order (cont)

Interrupt Source	Interrupt Type	Interrupt Priority Order (Initial Value)	IPR	Vector No.	Vector Table Address	Priority within IPR Setting Unit	Default Priority
DMAC0	DEI0	0 to 15 (0)	IPRC (15 to 12)	128	VBR + (vecter No. × 4)		High
DMAC1	DEI1	0 to 15 (0)	IPRC (11 to 8)	129			↑
DMAC2	DEI2	0 to 15 (0)	IPRC (7 to 4)	132			
DMAC3	DEI3	0 to 15 (0)	IPRC (3 to 0)	133			
SCIF0	RXI0	0 to 15 (0)	IPRD (15 to 12)	136		High	
	TXI0	0 to 15 (0)	IPRD (15 to 12)	137		Low	
SCIF1	RXI1	0 to 15 (0)	IPRD (11 to 8)	140		High	
	TXI1	0 to 15 (0)	IPRD (11 to 8)	141		Low	
SCIF2	ERI2	0 to 15 (0)	IPRD (7 to 4)	144		High	
	RXI2	0 to 15 (0)	IPRD (7 to 4)	145		↑	
	BRI2	0 to 15 (0)	IPRD (7 to 4)	146		↓	
	TXI2	0 to 15 (0)	IPRD (7 to 4)	147		Low	
ADC	ADI	0 to 15 (0)	IPRD (3 to 0)	148			
USB	USI0	0 to 15 (0)	IPRE (15 to 12)	160		High	
	USI1	0 to 15 (0)	IPRE (15 to 12)	161		Low	
CMT1	CMI	0 to 15 (0)	IPRE (11 to 8)	165			
TMU0	TUNI0	0 to 15 (0)	IPRG (15 to 12)	180			
TMU1	TUNI1	0 to 15 (0)	IPRG (11 to 8)	181			
TMU2	TUNI2	0 to 15 (0)	IPRG (7 to 4)	182		High	
	TICPI2	0 to 15 (0)	IPRG (7 to 4)	183		Low	
WDT	ITI	0 to 15 (0)	IPRH (15 to 12)	187			
REF	RCMI	0 to 15 (0)	IPRH (11 to 8)	188		High	↓
	ROVI	0 to 15 (0)	IPRH (11 to 8)	189		Low	Low

8.3 INTC Registers

8.3.1 Interrupt Priority Registers A to H (IPRA–IPRH)

Interrupt priority registers A to H (IPRA to IPRH) are 16-bit readable/writable registers in which priority levels 0–15 are set for on-chip peripheral modules, IRQ, and PINT interrupts. These registers are initialized to H'0000 by a power-on reset or manual reset, but are not initialized in standby mode.

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

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

Table 8.4 lists the relationship between the interrupt sources and the IPRA–IPRH bits.

Table 8.4 Interrupt Request Sources and IPRA–IPRH

Register (address)	Bits 15 to 12	Bits 11 to 8	Bits 7 to 4	Bits 3 to 0
IPRA (H'A4002080)	IRQ0	IRQ1	IRQ2	IRQ3
IPRB (H'A4002082)	IRQ4	IRQ5	IRQ6	IRQ7
IPRC (H'A4002084)	DMAC0	DMAC1	DMAC2	DMAC3
IPRD (H'A4002086)	SCIF0	SCIF1	SCIF2	ADC
IPRE (H'A4002088)	USB	CMT1	Reserved*	Reserved*
IPRG (H'A400208C)	TMU0	TMU1	TMU2	Reserved*
IPRH (H'A400208E)	WDT	REF	Reserved*	Reserved*

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

As listed in table 8.4, on-chip peripheral modules or IRQ or PINT interrupts are assigned to four 4-bit groups in each register. These 4-bit groups (bits 15 to 12, bits 11 to 8, bits 7 to 4, and bits 3 to 0) are set with values from H'0 (0000) to H'F (1111). Setting H'0 means priority level 0 (masking is requested); H'F is priority level 15 (the highest level). A reset initializes IPRA–IPRE to H'0000.

8.3.2 Interrupt Control Register 0 (ICR0)

The ICR0 is a register that sets the input signal detection mode of external interrupt input pin NMI, and indicates the input signal level at the NMI pin. This register is initialized to H'0000 or H'8000 by a power-on reset or manual reset, but is not initialized in standby mode.

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

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

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

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

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

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

Bit 1—Interrupt Request Enable (IRQE): Enables or disables the use of pins IRQ7 to IRQ0 as eight independent interrupt pins.

Bit 1: IRQE	Description	
0	Use prohibited	(Initial value)
1	Use as eight independent interrupt request pins IRQ7 to IRQ0 enabled*	

Note: * Once use has been enabled (by setting IRQE to 1), do not change the IRQE setting until a power-on reset is executed.

Bits 14 to 9, 7 to 2, and 0—Reserved: These bits are always read as 0. The write value should always be 0.

8.3.3 Interrupt Control Register 1 (ICR1)

The ICR1 is a 16-bit register that specifies the detection mode for external interrupt input pins IRQ0 to IRQ7 individually: rising edge, falling edge, or low level. This register is initialized to H'4000 by a power-on reset or manual reset, but is not initialized in standby mode.

Bit:	15	14	13	12	11	10	9	8
	IRQ71S	IRQ70S	IRQ61S	IRQ60S	IRQ51S	IRQ50S	IRQ41S	IRQ40S
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	RW	R/W	R/W	R/W	R/W

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

Bits 2n+1 and 2n—IRQ Sense Select (IRQn1S, IRQn0S) (n = 7 to 0): These bits select whether interrupt request signals corresponding to the eight IRQn pins are detected by a rising edge, falling edge, or low level.

Bit 2n + 1: **IRQn1S**	**Bit 2n:** **IRQn0S**	**Description**
0	0	Interrupt request is detected on low level of IRQn input (Initial value)
	1	Interrupt request is detected on falling edge of IRQn input
1	0	Interrupt request is detected on rising edge of IRQn input
	1	Reserved

8.3.4 Interrupt Request Register (IRR)

The IRR is a 16-bit register that indicates interrupt requests from external input pins IRQ0 to IRQ7. This register is initialized to H'00 by a power-on reset or manual reset, but is not initialized in standby mode.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—
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
	IRQ7R	IRQ6R	IRQ5R	IRQ4R	IRQ3R	IRQ2R	IRQ1R	IRQ0R
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

When clearing an IRQ7R–IRQ0R bit to 0, 0 should be written to the bit after the bit is set to 1 and the contents of 1 are read. Only 0 can be written to IRQ7R–IRQ0R.

Bits 7 to 0—IRQ7 to IRQ0 Flags (IRQ7R to IRQ0R): These flags indicate the status of IRQ7 to IRQ0 interrupt requests.

Bit 7-0: IRQ7R to IRQ0R	Detection Setting	Description	
0	Level detection	There is no IRQn interrupt request [Clearing condition] When IRQn input is high	(Initial value)
	Edge detection	An IRQn interrupt request has not been detected [Clearing conditions] 1. When 0 is written to IRQnR after reading IRQnR while set to 1 2. When an IRQn interrupt is accepted	(Initial value)
1	Level detection	There is an IRQn interrupt request [Setting condition] When IRQn input is low	
	Edge detection	An IRQn interrupt request has been detected [Setting condition] When an IRQn input edge is detected	

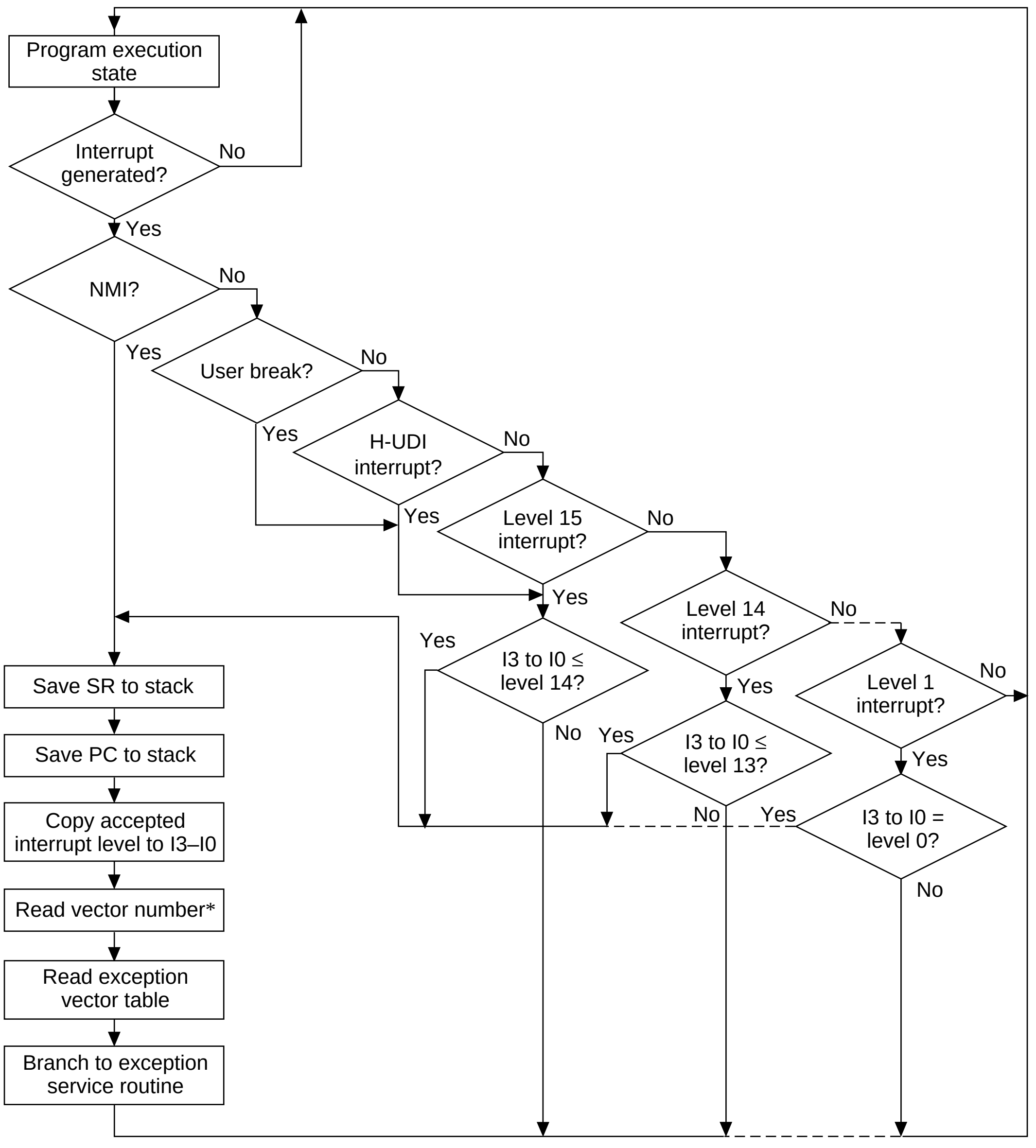
(n = 0 to 7)

8.4 Interrupt Operation

8.4.1 Interrupt Sequence

The sequence of operations in interrupt generation is described below and illustrated in figure 8.1.

1. The interrupt request sources send interrupt request signals to the interrupt controller.
2. The interrupt controller selects the highest-priority among the interrupt requests sent, according to the interrupt priority levels in setting registers A to E (IPRA–IPRE). Lower-priority interrupts are held pending. If two or more of these interrupts have the same priority level or if multiple interrupts occur within a single module, the interrupt with the highest default priority or the highest priority within its IPR setting unit (as indicated in table 8.4) is selected.
3. The interrupt controller compares the priority level of the selected interrupt request with the interrupt mask bits (I3–I0) in the CPU's status register (SR). If the request priority level is equal to or less than the level set in I3–I0, the request is held pending. If the request priority level is higher than the level in bits I3–I0, the interrupt controller accepts the interrupt and sends an interrupt request signal to the CPU.
4. The CPU detects the interrupt request sent from the interrupt controller when it decodes the next instruction to be executed. Instead of executing the decoded instruction, the CPU starts interrupt exception handling.
5. Status register (SR) and program counter (PC) are saved onto the stack.
6. The priority level of the accepted interrupt is copied to the interrupt mask level bits (I3–I0) in the status register (SR).
7. When external vector mode is specified for the IRL/IRQ interrupt, the vector number is read from the external vector number input pins (D7–D0).
8. The CPU reads the start address of the exception service routine from the exception vector table entry for the accepted interrupt, jumps to that address, and starts executing the program there. This jump is not a delayed branch.



I3–I0: Status register interrupt mask bits.

Note: * The vector number is only read from an external source when an external vector number is specified for the IRL/IRQ interrupt vector number.

Figure 8.1 Interrupt Sequence Flowchart

Section 9 User Break Controller

9.1 Overview

The user break controller (UBC) provides functions that simplify program debugging. These functions make it easy to design an effective self-monitoring debugger, enabling the chip to debug programs without using an in-circuit emulator. Break conditions that can be set in the UBC are instruction fetch or data read/write access, data size, data contents, address value, and timing in the case of instruction fetch.

9.1.1 Features

The UBC has the following features:

- The following break comparison conditions can be set.

  Number of break channels: two channels (channels A and B)
  User break can be requested as either the independent or sequential condition on channels A and B (sequential break setting: channel A and, then channel B match with logical AND, but not in the same bus cycle).

  — Address (32 bits, 32-bit maskable)

    One of four address buses (CPU address bus (LAB), cache address bus (IAB),

    X-memory address bus (XAB) and Y-memory address bus (YAB)) can be selected.

  — Data (only on channel B, 32-bit maskable)

    One of the four data buses (CPU data bus (LDB), cache data bus (IDB), X-memory data bus (XDB) and Y-memory data bus (YDB)) can be selected.

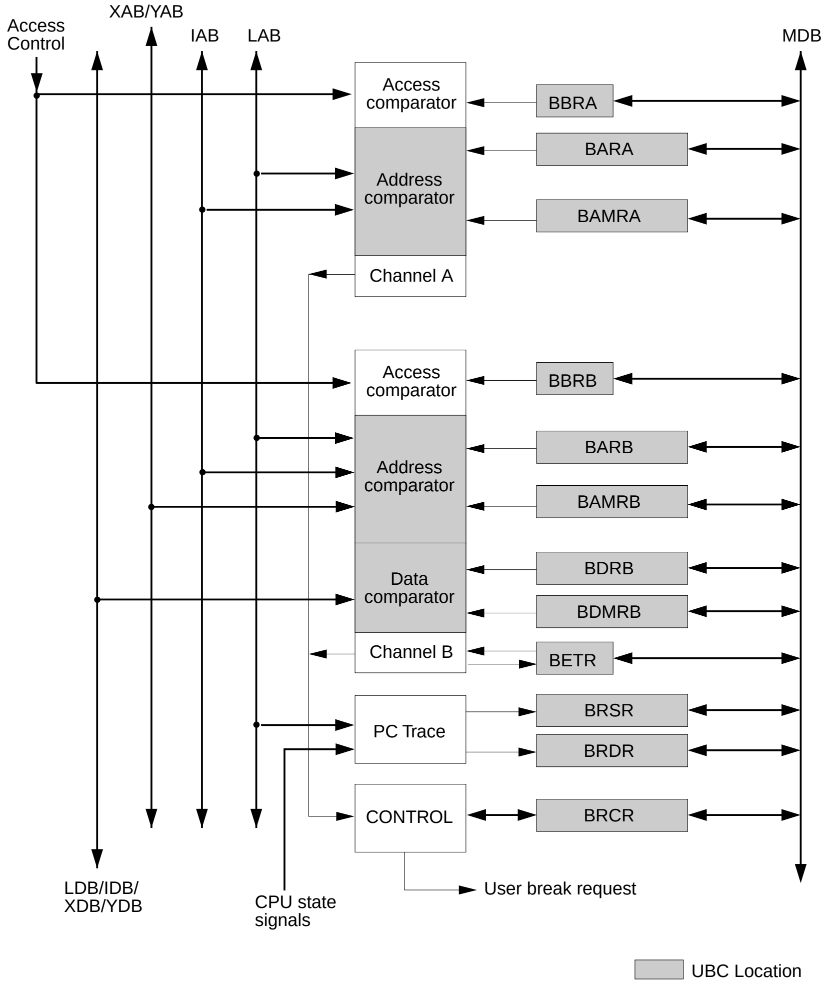
  — Bus master: CPU cycle or DMAC cycle

  — Bus cycle: instruction fetch or data access

  — Read/write

  — Operand size: byte, word, or longword
- User break is generated upon satisfying break conditions. A user-designed user-break condition exception processing routine can be run.
- In an instruction fetch cycle, it can be selected that a break is set before or after an instruction is executed.
- The number of repeat times can be specified as a break condition (It is only for channel B).
- Maximum repeat times for the break condition: 2^{12} – 1 times
- Eight pairs of branch source/destination buffers

9.1.2 Block Diagram



Legend

BBRA:	Break bus cycle register A	BDRB:	Break data register B
BARA:	Break address register A	BDMRB:	Break data mask register B
BAMRA:	Break address mask register A	BETR:	Break execution times register
BBRB:	Break bus cycle register B	BRSR:	Branch source register
BARB:	Break address register B	BRDR:	Branch destination register
BAMRB:	Break address mask register B	BRCR:	Break control register

Figure 9.1 Block Diagram of User Break Controller

9.1.3 Register Configuration

Table 9.1 Register Configuration

Name	Abbr.	R/W	Initial Value*1	Address	Access Size	Location
Break address register A	BARA	R/W	H'00000000	H'FFFFFFB0	16, 32	UBC
Break address mask register A	BAMRA	R/W	H'00000000	H'FFFFFFB4	16, 32	UBC
Break bus cycle register A	BBRA	R/W	H'0000	H'FFFFFFB8	16	UBC
Break address register B	BARB	R/W	H'00000000	H'FFFFFFA0	16, 32	UBC
Break address mask register B	BAMRB	R/W	H'00000000	H'FFFFFFA4	16, 32	UBC
Break bus cycle register B	BBRB	R/W	H'0000	H'FFFFFFA8	16	UBC
Break data register B	BDRB	R/W	H'00000000	H'FFFFFF90	16, 32	UBC
Break data mask register B	BDMRB	R/W	H'00000000	H'FFFFFF94	16, 32	UBC
Break control register	BRCR	R/W	H'00000000	H'FFFFFF98	16, 32	UBC
Execution count break register	BETR	R/W	H'0000	H'FFFFFF9C	16	UBC
Branch source register	BRSR	R	Undefined*2	H'FFFFFFAC	16, 32	UBC
Branch destination register	BRDR	R	Undefined*2	H'FFFFFFBC	16, 32	UBC

Notes: *1 Initialized by power-on reset. Values held in standby state and undefined by manual resets.

*2 Bit 31 of BRSR and BRDR (valid flag) is initialized by power-on resets. But other bits are not initialized.

9.2 Register Descriptions

9.2.1 Break Address Register A (BARA)

BARA is a 32-bit read/write register. BARA specifies the address used as a break condition in channel A. A power-on reset initializes BARA to H'00000000.

Bit:	31	30	29	28	27	26	25	24
	BAA31	BAA30	BAA29	BAA28	BAA27	BAA26	BAA25	BAA24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
	BAA23	BAA22	BAA21	BAA20	BAA19	BAA18	BAA17	BAA16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
	BAA15	BAA14	BAA13	BAA12	BAA11	BAA10	BAA9	BAA8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	BAA7	BAA6	BAA5	BAA4	BAA3	BAA2	BAA1	BAA0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 31 to 0—Break Address A31 to A0 (BAA31 to BAA0): Stores the address on the LAB or IAB specifying break conditions of channel A.

9.2.2 Break Address Mask Register A (BAMRA)

BAMRA is a 32-bit read/write register. BAMRA specifies bits masked in the break address specified by BARA. A power-on reset initializes BAMRA to H'00000000.

Bit:	31	30	29	28	27	26	25	24
	BAMA31	BAMA30	BAMA29	BAMA28	BAMA27	BAMA26	BAMA25	BAMA24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
	BAMA23	BAMA22	BAMA21	BAMA20	BAMA19	BAMA18	BAMA17	BAMA16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
	BAMA15	BAMA14	BAMA13	BAMA12	BAMA11	BAMA10	BAMA9	BAMA8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	BAMA7	BAMA6	BAMA5	BAMA4	BAMA3	BAMA2	BAMA1	BAMA0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 31 to 0—Break Address Mask Register A31 to A0 (BAMA31 to BAMA0): Specifies bits masked in the channel A break address bits specified by BARA (BAA31–BAA0).

Bit 31 to 0: BAMAn	Description
0	Break address bit BAAn of channel A is included in the break condition (Initial value)
1	Break address bit BAAn of channel A is masked and is not included in the break condition

(n = 31 to 0)

9.2.3 Break Bus Cycle Register A (BBRA)

Break bus cycle register A (BBRA) is a 16-bit read/write register, which specifies (1) CPU cycle or DMAC cycle, (2) instruction fetch or data access, (3) read or write, and (4) operand size in the break conditions of channel A. A power-on reset initializes BBRA to H'0000.

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
	CDA1	CDA0	IDA1	IDA0	RWA1	RWA0	SZA1	SZA0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 to 8—Reserved: These bits are always read as 0. The write value should always be 0.

Bits 7 and 6—CPU Cycle/DMAC Cycle Select A (CDA1, CDA0): Selects the CPU cycle or DMAC cycle as the bus cycle of the channel A break condition.

Bit 7: CDA1	Bit 6: CDA0	Description
0	0	Condition comparison is not performed (Initial value)
*	1	The break condition is the CPU cycle
1	0	The break condition is the DMAC cycle

Note: * Don’t care

Bits 5 and 4—Instruction Fetch/Data Access Select A (IDA1, IDA0): Selects the instruction fetch cycle or data access cycle as the bus cycle of the channel A break condition.

Bit 5: IDA1	Bit 4: IDA0	Description
0	0	Condition comparison is not performed (Initial value)
	1	The break condition is the instruction fetch cycle
1	0	The break condition is the data access cycle
	1	The break condition is the instruction fetch cycle or data access cycle

Bits 3 and 2—Read/Write Select A (RWA1, RWA0): Selects the read cycle or write cycle as the bus cycle of the channel A break condition.

Bit 3: RWA1	Bit 2: RWA0	Description
0	0	Condition comparison is not performed (Initial value)
	1	The break condition is the read cycle
1	0	The break condition is the write cycle
	1	The break condition is the read cycle or write cycle

Bits 1 and 0—Operand Size Select A (SZA1, SZA0): Selects the operand size of the bus cycle for the channel A break condition.

Bit 1: SZA1	Bit 0: SZA0	Description
0	0	The break condition does not include operand size (Initial value)
	1	The break condition is byte access
1	0	The break condition is word access
	1	The break condition is longword access

9.2.4 Break Address Register B (BARB)

BARB is a 32-bit read/write register. BARB specifies the address used as a break condition in channel B. Control bits XYE and XYS in the BBRB selects an address bus for break condition B. If the XYE is 0, then BARB specifies the break address on logic or internal bus, LAB or IAB. If the XYE is 1, then the BAB 31–16 specifies the break address on XAB (bits 15–1) and the BAB 15–0 specifies the break address on YAB (bits 15–1). However, you have to choose one of two address buses for the break. A power-on reset initializes BARB to H'00000000.

Bit:	31	30	29	28	27	26	25	24
	BAB31	BAB30	BAB29	BAB28	BAB27	BAB26	BAB25	BAB24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
	BAB23	BAB22	BAB21	BAB20	BAB19	BAB18	BAB17	BAB16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
	BAB15	BAB14	BAB13	BAB12	BAB11	BAB10	BAB9	BAB8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	BAB7	BAB6	BAB5	BAB4	BAB3	BAB2	BAB1	BAB0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

	BAB31–16	**BAB15–0**
XYE = 0	L(I) AB31–16	L(I) AB15–0
XYE = 1	XAB15–1 (XYS = 0)	YAB15–1 (XYS = 1)

9.2.5 Break Address Mask Register B (BAMRB)

BAMRB is a 32-bit read/write register. BAMRB specifies bits masked in the break address specified by BARB. A power-on reset initializes BAMRB to H'00000000.

Bit:	31	30	29	28	27	26	25	24
	BAMB31	BAMB30	BAMB29	BAMB28	BAMB27	BAMB26	BAMB25	BAMB24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
	BAMB23	BAMB22	BAMB21	BAMB20	BAMB19	BAMB18	BAMB17	BAMB16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
	BAMB15	BAMB14	BAMB13	BAMB12	BAMB11	BAMB10	BAMB9	BAMB8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	BAMB7	BAMB6	BAMB5	BAMB4	BAMB3	BAMB2	BAMB1	BAMB0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

	BAMB31–16	**BAMB15–0**
XYE = 0	Mask L(I) AB31–16	Mask L(I) AB15–0
XYE = 1	Mask XAB15–1 (XYS = 0)	Mask YAB15–1 (XYS = 1)

Bit 31–0: BAMBn	**Description**
0	Break address BABn of channel B is included in the break condition (Initial value)
1	Break address BABn of channel B is masked and is not included in the break condition

(n = 31 to 0)

9.2.6 Break Data Register B (BDRB)

BDRB is a 32-bit read/write register. The control bits XYE and XYS in BBRB select a data bus for break condition B. If the XYE is 0, then BDRB specifies the break data on LDB or IDB. If the XYE is 1, then BDB 31–16 specifies the break data on XDB (bits 15–0) and BDB 15–0 specifies the break data on YDB (bits 15–0). However, you have to choose one of two data buses for the break. A power-on reset initializes BDRB to H'00000000.

Bit:	31	30	29	28	27	26	25	24
	BDB31	BDB30	BDB29	BDB28	BDB27	BDB26	BDB25	BDB24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
	BDB23	BDB22	BDB21	BDB20	BDB19	BDB18	BDB17	BDB16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
	BDB15	BDB14	BDB13	BDB12	BDB11	BDB10	BDB9	BDB8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	BDB7	BDB6	BDB5	BDB4	BDB3	BDB2	BDB1	BDB0
Initial value:	0	0	0	0	0	0	0	0

	BDB31–16	**BDB15–0**
XYE = 0	L(I) DB31–16	L(I) DB15–0
XYE = 1	XDB15–1 (XYS = 0)	YDB15–1 (XYS = 1)

9.2.7 Break Data Mask Register B (BDMRB)

BDMRB is a 32-bit read/write register. BDMRB specifies bits masked in the break data specified by BDRB. A power-on reset initializes BDMRB to H'00000000.

Bit:	31	30	29	28	27	26	25	24
	BDMB31	BDMB30	BDMB29	BDMB28	BDMB27	BDMB26	BDMB25	BDMB24
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	23	22	21	20	19	18	17	16
	BDMB23	BDMB22	BDMB21	BDMB20	BDMB19	BDMB18	BDMB17	BDMB16
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
	BDMB15	BDMB14	BDMB13	BDMB12	BDMB11	BDMB10	BDMB9	BDMB8
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	BDMB7	BDMB6	BDMB5	BDMB4	BDMB3	BDMB2	BDMB1	BDMB0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

	BDMB31–16	**BDMB15–0**
XYE = 0	Mask L(I) DB31–16	Mask L(I) DB15–0
XYE = 1	Mask XDB15–0 (XYS = 0)	Mask YDB15–0 (XYS = 1)

Bit 31–0: **BDMBn**	**Description**
0	Break data BDBn of channel B is included in the break condition (Initial value)
1	Break data BDBn of channel B is masked and is not included in the break condition

(n = 31 to 0)

Notes: 1. Specify an operand size when including the value of the data bus in the break condition.

2. When a byte size is selected as a break condition, the break data must be set in bits 15–8 in BDRB for an even break address and bits 7–0 for an odd break address. Another 8 bits have no influence on a break condition.

9.2.8 Break Bus Cycle Register B (BBRB)

Break bus cycle register B (BBRB) is a 16-bit read/write register, which specifies (1) logic or internal bus (L or I bus), X bus, or Y bus, (2) CPU cycle or DMAC cycle, (3) instruction fetch or data access, (4) read/write, and (5) operand size in the break conditions of channel B. A power-on reset initializes BBRB to H'0000.

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

Bit:	7	6	5	4	3	2	1	0
	CDB1	CDB0	IDB1	IDB0	RWB1	RWB0	SZB1	SZB0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 to 10—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 9—X/Y Memory Bus Enable (XYE): Selects the logic or internal bus (L or I bus) or X/Y memory bus as the bus of the channel B break condition.

Bit 9: XYE	Description
0	Select internal bus (I bus) for the channel B break condition
1	Select X/Y memory bus (X/Y bus) for the channel B break condition

Bit 8—X or Y Memory Bus Select (XYS): Selects the X bus or the Y bus as the bus of the channel B break condition.

Bit 8: XYS	Description
0	Select the X bus for the channel B break condition
1	Select the Y bus for the channel B break condition

Bits 7 and 6—CPU Cycle/DMAC Cycle Select B (CDB1, CDB0): Selects the CPU cycle or DMAC cycle as the bus cycle of the channel B break condition.

Bit 7: CDB1	Bit 6: CDB0	Description
0	0	Condition comparison is not performed (Initial value)
*	1	The break condition is the CPU cycle
1	0	The break condition is the DMAC cycle

Note: * Don't care.

Bits 5 and 4—Instruction Fetch/Data Access Select B (IDB1, IDB0): Selects the instruction fetch cycle or data access cycle as the bus cycle of the channel B break condition.

Bit 5: IDB1	Bit 4: IDB0	Description
0	0	Condition comparison is not performed (Initial value)
	1	The break condition is the instruction fetch cycle
1	0	The break condition is the data access cycle
	1	The break condition is the instruction fetch cycle or data access cycle

Bits 3 and 2—Read/Write Select B (RWB1, RWB0): Selects the read cycle or write cycle as the bus cycle of the channel B break condition.

Bit 3: RWB1	Bit 2: RWB0	Description
0	0	Condition comparison is not performed (Initial value)
	1	The break condition is the read cycle
1	0	The break condition is the write cycle
	1	The break condition is the read cycle or write cycle

Bits 1 and 0—Operand Size Select B (SZB1, SZB0): Selects the operand size of the bus cycle for the channel B break condition.

Bit 1: SZB1	Bit 0: SZB0	Description
0	0	The break condition does not include operand size (Initial value)
	1	The break condition is byte access
1	0	The break condition is word access
	1	The break condition is longword access

9.2.9 Break Control Register (BRCR)

BRCR sets the following conditions:

1. Channels A and B are used in two independent channels condition or under the sequential condition.
2. A break is set before or after instruction execution.
3. A break is set by the number of execution times.
4. Determine whether to include data bus on channel B in comparison conditions.
5. Enable PC trace.

The break control register (BRCR) is a 32-bit read/write register that has break conditions match flags and bits for setting a variety of break conditions.

A power-on reset initializes BRCR to H'00000000.

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	1	1	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bit:	15	14	13	12	11	10	9	8
	SCMFCA	SCMFCB	SCMFDA	SCMFDB	PCTE	PCBA	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R	R

Bit:	7	6	5	4	3	2	1	0
	DBEB	PCBB	—	—	SEQ	—	—	ETBE
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R/W	R	R	R/W

Bits 31 to 22, and 19 to 16—Reserved: These bits are always read as 0. The write value should always be 0.

Bits 21 and 20—Reserved: These bits are always read as 1. The write value should always be 1.

Bit 15—CPU Condition Match Flag A (SCMFCA): When the CPU bus cycle condition in the break conditions set for channel A is satisfied, this flag is set to 1 (not cleared to 0). In order to clear this flag, write 0 into this bit.

Bit 15: SCMFCA	Description	
0	The CPU cycle condition for channel A does not match	(Initial value)
1	The CPU cycle condition for channel A matches	

Bit 14—CPU Condition Match Flag B (SCMFCB): When the CPU bus cycle condition in the break conditions set for channel B is satisfied, this flag is set to 1 (not cleared to 0). In order to clear this flag, write 0 into this bit.

Bit 14: SCMFCB	Description	
0	The CPU cycle condition for channel B does not match	(Initial value)
1	The CPU cycle condition for channel B matches	

Bit 13—DMAC Condition Match Flag A (SCMFDA): When the on-chip DMAC bus cycle condition in the break conditions set for channel A is satisfied, this flag is set to 1 (not cleared to 0). In order to clear this flag, write 0 into this bit.

Bit 13: SCMFDA	Description	
0	The DMAC cycle condition for channel A does not match	(Initial value)
1	The DMAC cycle condition for channel A matches	

Bit 12—DMAC Condition Match Flag B (SCMFDB): When the on-chip DMAC bus cycle condition in the break conditions set for channel B is satisfied, this flag is set to 1 (not cleared to 0). In order to clear this flag, write 0 into this bit.

Bit 12: SCMFDB	Description	
0	The DMAC cycle condition for channel B does not match	(Initial value)
1	The DMAC cycle condition for channel B matches	

Bit 11—PC Trace Enable (PCTE): Enables PC trace.

Bit 11: PCTE	Description	
0	Disables PC trace	(Initial value)
1	Enables PC trace	

Bit 10—PC Break Select A (PCBA): Selects the break timing of the instruction fetch cycle for channel A as before or after instruction execution.

Bit 10: PCBA	Description	
0	PC break of channel A is set before instruction execution	(Initial value)
1	PC break of channel A is set after instruction execution	

Bits 9 and 8—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 7—Data Break Enable B (DBEB): Selects whether or not the data bus condition is included in the break condition of channel B.

Bit 7: DBEB	Description	
0	No data bus condition is included in the condition of channel B	(Initial value)
1	The data bus condition is included in the condition of channel B	

Bit 6—PC Break Select B (PCBB): Selects the break timing of the instruction fetch cycle for channel B as before or after instruction execution.

Bit 6: PCBB	Description	
0	PC break of channel B is set before instruction execution	(Initial value)
1	PC break of channel B is set after instruction execution	

Bits 5 and 4—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 3—Sequence Condition Select (SEQ): Selects two conditions of channels A and B as independent or sequential.

Bit 3: SEQ	Description	
0	Channels A and B are compared under the independent condition	(Initial value)
1	Channels A and B are compared under the sequential condition	

Bits 2 and 1—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 0—The Number of Execution Times Break Enable (ETBE): Enables the execution-times break condition only on channel B. If this bit is 1 (break enable), a user break is issued when the number of break conditions matches with the number of execution times that is specified by the BETR register.

Bit 0: ETBE	Description	
0	The execution-times break condition is masked on channel B	(Initial value)
1	The execution-times break condition is enabled on channel B	

9.2.10 Execution Times Break Register (BETR)

When the execution-times break condition of channel B is enabled, this register specifies the number of execution times to make the break. The maximum number is 2^{12} – 1 times. A power-on reset initializes BETR to H'0000. When a break condition is satisfied, it decreases the BETR. A break is issued when the break condition is satisfied after the BETR becomes H'0001. Bits 15–12 are always read as 0 and 0 should always be written in these bits.

Instructions in a repeat loop comprising no more than three instructions do not accept external interrupts. Therefore, BETR is not decremented in the case of a break condition match that occurs for an instruction in a repeated repeat loop comprising no more than three instructions.

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/W	R/W	R/W	R/W

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

9.2.11 Branch Source Register (BRSR)

BRSR is a 32-bit read register. BRSR stores the last fetched address before branch and the pointer (3 bits) which indicates the number of cycles from fetch to execution for the last executed instruction. BRSR has the flag bit that is set to 1 when branch occurs. This flag bit is cleared to 0, when BRSR is read and also initialized by power-on resets or manual resets. Other bits are not initialized by reset. Four BRSR registers have queue structure and a stored register is shifted at every branch.

Bit:	31	30	29	28	27	26	25	24
	SVF	PID2	PID1	PID0	BSA27	BSA26	BSA25	BSA24
Initial value:	0	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	23	22	21	20	19	18	17	16
	BSA23	BSA22	BSA21	BSA20	BSA19	BSA18	BSA17	BSA16
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	15	14	13	12	11	10	9	8
	BSA15	BSA14	BSA13	BSA12	BSA11	BSA10	BSA9	BSA8
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	BSA7	BSA6	BSA5	BSA4	BSA3	BSA2	BSA1	BSA0
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Note: * Undefined

Bit 31—BRSR Valid Flag (SVF): Indicates whether the address and the pointer by which the branch source address can be calculated. When a branch source address is fetched, this flag is set to 1. This flag is cleared to 0 in reading BRSR.

Bit 31: SVF	Description
0	The value of BRSR register is invalid
1	The value of BRSR register is valid

Bits 30 to 28—Instruction Decode Pointer (PID2, PID1, PID0): These bits are 3-bit binary pointers. These bits indicate the instruction buffer number which stores the last executed instruction before branch.

Bits 30 to 28:

PID	**Description**
Even	PID indicates the instruction buffer number
Odd	PiD+2 indicates the instruction buffer number

Bits 27 to 0—Branch Source Address (BSA27 to BSA0): These bits store the last fetched address before branch.

9.2.12 Branch Destination Register (BRDR)

BRDR is a 32-bit read register. BRDR stores the branch destination fetch address. BRDR has the flag bit that is set to 1 when branch occurs. This flag bit is cleared to 0, when BRDR is read and also initialized by power-on resets or manual resets. Other bits are not initialized by resets. Four BRDR registers have queue structure and a stored register is shifted at every branch.

Bit:	31	30	29	28	27	26	25	24
	DVF	—	—	—	BDA27	BDA26	BDA25	BDA24
Initial value:	0	0	0	0	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	23	22	21	20	19	18	17	16
	BDA23	BDA22	BDA21	BDA20	BDA19	BDA18	BDA17	BDA16
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	15	14	13	12	11	10	9	8
	BDA15	BDA14	BDA13	BDA12	BDA11	BDA10	BDA9	BDA8
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	BDA7	BDA6	BDA5	BDA4	BDA3	BDA2	BDA1	BDA0
Initial value:	*	*	*	*	*	*	*	*
R/W:	R	R	R	R	R	R	R	R

Note: * Undefined

Bit 31—BRDR Valid Flag (DVF): Indicates whether a branch destination address is stored. When a branch destination address is fetched, this flag is set to 1. This flag is set to 0 in reading BRDR.

Bit 31: DVF	Description
0	The value of BRDR register is invalid
1	The value of BRDR register is valid

Bits 30 to 28—Reserved: These bits are always read as 0. The write value should always be 0.

Bits 27 to 0—Branch Destination Address (BDA27 to BDA0): These bits store the first fetched address after branch.

9.3 Operation Description

9.3.1 Flow of the User Break Operation

The flow from setting of break conditions to user break exception processing is described below:

1. The break addresses are loaded in the break address registers (BARA and BARB). The masked addresses are set in the break address mask registers (BAMRA and BAMRB). The break data is set in the break data register (BDRB). The masked data is set in the break data mask register (BDMRB). The breaking bus conditions are set in the break bus cycle registers (BBRA and BBRB). Three groups of the BBRA and BBRB (CPU cycle/DMAC cycle select, instruction fetch/data access select, and read/write select) are each set. No user break will be generated if even one of these groups is set with 00. The respective conditions are set in the bits of the BRCR.
2. When the break conditions are satisfied, the UBC sends a user break request to the interrupt controller. The break type will be sent to CPU indicating the instruction fetch, pre-/post-instruction break, data access break.
3. The interrupt controller performs priority determination for user break interrupts. As the priority level of a user break interrupt is 15, it is accepted when the setting of the interrupt mask bits (I3–I0) in the status register (SR) is 14 or below. If the setting of bits I3–I0 is level 15, a user break interrupt is not accepted, but is held pending until it can be. For details of interrupt priority determination, see section 8, Interrupt Controller.
4. If a user break interrupt is accepted as the result of interrupt priority determination, the CPU initiates the user break interrupt. The break type is sent to the CPU, indicating instruction fetch, pre-/post-instruction break, or data access break.
5. The appropriate condition match flags (SCMFCA, SCMFDA, SCMFCB, and SCMFDB) can be used to check if the set conditions match or not. The matching of the conditions sets flags, but they are not reset. 0 must first be written to them before they can be used again.
6. There is a chance that the data access break and its following instruction fetch break occur around the same time, there will be only one break request to the CPU, but these two break channel match flags could be both set.

9.3.2 Break on Instruction Fetch Cycle

1. When CPU/instruction fetch/read/word or longword is set in the break bus cycle registers (BBRA/BBRB), the break condition becomes the CPU instruction fetch cycle. Whether it then breaks before or after the execution of the instruction can then be selected with the PCBA/PCBB bits of the break control register (BRCR) for the appropriate channel.
2. An instruction set for a break before execution breaks when it is confirmed that the instruction has been fetched and will be executed. This means this feature cannot be used on instructions fetched by overrun (instructions fetched at a branch or during an interrupt transition, but not to be executed). When this kind of break is set for the delay slot of a delay branch instruction, the

break is generated prior to execution of the instruction that then first accepts the break. Meanwhile, the break set for pre-instruction-break on delay slot instruction and post-instruction-break on SLEEP instruction are also prohibited.

3. When the condition is specified to be occurred after execution, the instruction set with the break condition is executed and then the break is generated prior to the execution of the next instruction. As with pre-execution breaks, this cannot be used with overrun fetch instructions. When this kind of break is set for a delay branch instruction or an instruction for which interrupts are disabled, such as LDC, an interrupt is not generated until the first instruction at which interrupts are accepted.
4. When an instruction fetch cycle is set for channel B, break data register B (BDRB) is ignored. There is thus no need to set break data for the break of the instruction fetch cycle.

9.3.3 Break by Data Access Cycle

1. The memory cycles in which CPU data access breaks occur are from instructions.
2. The relationship between the data access cycle address and the comparison condition for operand size are listed in table 9.2:

Table 9.2 Data Access Cycle Addresses and Operand Size Comparison Conditions

Access Size	Address Compared
Longword	Compares break address register bits 31–2 to address bus bits 31–2
Word	Compares break address register bits 31–1 to address bus bits 31–1
Byte	Compares break address register bits 31–0 to address bus bits 31–0

This means that when address H'00001003 is set without specifying the size condition, for example, the bus cycle in which the break condition is satisfied is as follows (where other conditions are met).

Longword access at H'00001000

Word access at H'00001002

Byte access at H'00001003

3. When the data value is included in the break conditions on B channel:

When the data value is included in the break conditions, either longword, word, or byte is specified as the operand size of the break bus cycle registers (BBRA and BBRB). When data values are included in break conditions, a break is generated when the address conditions and data conditions both match. To specify byte data for this case, set the same data in two bytes at bits 15–8 and bits 7–0 of the break data register B (BDRB) and break data mask register B (BDMRB). When word or byte is set, bits 31–16 of BDRB and BDMRB are ignored.

4. When the DMAC data access is included in the break condition:

   When the address is included in the break condition on DMAC data access, the operand size of the break bus cycle registers (BBRA and BBRB) should be byte, word or no specified operand size. When the data value is included, select either byte or word.

9.3.4 Break on X-/Y-Memory Bus Cycle

1. The break condition on X-/Y-memory bus cycle is specified only in channel B. If XYE in BBRB is set to 1, break address and break data on X-/Y-memory bus are selected. At this time, select X-memory bus or Y-memory bus by specifying XYS in BBRB. The Break condition cannot include both X-memory and Y-memory at the same time. The break condition is applied to X-/Y-memory bus cycle by specifying CPU/data access/read or write/word or no specified operand size in the break bus cycle register B (BBRB).
2. When X-memory address is selected as the break condition, specify X-memory address in upper 16 bits in BARB and BAMRB. When Y-memory address is selected, specify Y-memory address in lower 16 bits. Specification of X-/Y-memory data is the same for BDRB and BDMRB.

9.3.5 Sequential Break

1. By specifying SEQ in BRCR is set to 1, the sequential break is issued when channel B break condition matches after channel A break condition matches. A user break is ignored even if channel B break condition matches before channel A break condition matches. When channels A and B condition match at the same time, the sequential break is not issued.
2. In sequential break specification, internal/X/Y bus can be selected and the execution times break condition can be also specified. For example, when the execution times break condition is specified, the break condition is satisfied at channel B condition match with BETR = H'0001 after channel A condition match.

9.3.6 Value of Saved Program Counter

1. When instruction fetch (before instruction execution) is specified as a break condition:

   The program counter (PC) value saved to the stack in user break interrupt handling is the address of the instruction that matches the break condition. A user break interrupt is generated and the fetched instruction is not executed. However, if a setting is made for an instruction following an instruction for which interrupts are disabled, a break occurs before execution of the next instruction at which interrupts are accepted, and so the PC value saved is the address at which the break occurs.
2. When instruction fetch (after instruction execution) is specified as a break condition:

   The PC value saved to the stack in user break interrupt handling is the address of the instruction following the instruction that matches the break condition. The fetched instruction is executed, and a break interrupt is generated before execution of the next instruction. However, if a setting is made for an instruction for which interrupts are disabled, a break occurs before execution of the next instruction at which interrupts are accepted, and so the PC value saved is the address at which the break occurs.

3. When data access (CPU/DMAC) is specified as a break condition:

   The PC value saved is the start address of the instruction following the instruction for which execution has been completed when user break exception handling is started. When data access (CPU/DMAC) is designated as a break condition, it is not possible to specify where the break will occur. The break will occur at an instruction that was about to be fetched in the vicinity of the data access for which the break should be made. The PC value is the start address of the instruction following the instruction already executed at the point at which user break processing is started. When a data value is added to the break condition, the location at which the break will occur cannot be specified precisely. The break will occur before execution of an instruction fetched in the vicinity of the data access at which the break was generated.

9.3.7 PC Trace

1. Setting PCTE in BRCR to 1 enables PC traces. When branch (branch instruction, repeat, and interrupt) is generated, the address from which the branch source address can be calculated and the branch destination address are stored in BRSR and BRDR, respectively. The branch address and the pointer, which corresponds to the branch, are included in BRSR.

2. The branch address before branch occurs can be calculated from the address and the pointer stored in BRSR. The expression from BSA (the address in BRSR), PID (the pointer in BRSR), and IA (the instruction address before branch occurs) is as follows: IA = BSA – 2 ∗ PID.

   Notes are needed when an interrupt (a branch) is issued before the branch destination instruction is executed. In case of the next figure, the instruction “Exec” executed immediately before branch is calculated by IA = BSA – 2 ∗ PID. However, when branch “branch” has delay slot and the destination address is 4n + 2 address, the address “Dest” which is specified by branch instruction is stored in BRSR (Dest = BSA). Therefore, as IA = BSA – 2 ∗ PID is not applied to this case, this PID is invalid. The case where BSA is 4n + 2 boundary is applied only to this case and then some cases are classified as follows:

```
Exec: branch  Dest
Dest: instr          (not executed)
      interrupt
Int:  interrupt routine
```

   If the PID value is odd, instruction buffer indicates PID+2 buffer. However, these expressions in this table are accounted for it. Therefore, the true branch source address is calculated with BSA and PID values stored in BRSR.

3. The branch address before branch occurrence, IA, has different values due to some kinds of branch.

   a. Branch instruction

      The branch instruction address

   b. Repeat

      The instruction before the last instruction of a repeat loop

      ```
      Repeat_Start:inst (1);  ---> BRDR
                   inst (2);
                     :
                   inst (n-1); --> the address calculated from BRSR
      Repeat_End: inst (n);
      ```

   c. Interrupt

      The last instruction executed before interrupt

      The top address of interrupt routine is stored in BRDR.

   In a repeat loop with instructions less than three, no instruction fetch cycle appears and branch source address is unknown. Therefore, PC trace is disabled.

4. BRSR and BRDR have eight pairs of queue structures. The top of queues is read first when the address stored in the PC trace register is read. BRSR and BRDR share the read pointer. Read BRSR and BRDR in order, the queue only shifts after BRDR is read. When reading BRDR, longword access should be used. Also, the PC trace has a trace pointer, which initially points to the bottom of the queues. The first pair of branch addresses will be stored at the bottom of the queues, then push up when next pairs come into the queues. The trace pointer will points to the next branch address to be executed, unless it got push out of the queues. When the branch address has been executed, the trace pointer will shift down to next pair of addresses, until it reaches the bottom of the queues. After switching the PCTE bit (in BRCR) off and on, the values in the queues are invalid. The read pointer stays at the position before PCTE is switched, but the trace pointer restart at the bottom of the queues.

9.3.8 Usage Examples

Break Condition Specified to a CPU Instruction Fetch Cycle

1. Register specifications

   BARA = H'00000404, BAMRA = H'00000000, BBRA = H'0054, BARB = H'00008010, BAMRB = H'00000006, BBRB = H'0054, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00300400

   Specified conditions: Channel A/channel B independent mode

   - Channel A

     Address: H'00000404, Address mask: H'00000000

     Bus cycle: CPU/instruction fetch (after instruction execution)/read (operand size is not included in the condition)

   - Channel B

     Address: H'00008010, Address mask: H'00000006

     Data: H'00000000, Data mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/read (operand size is not included in the condition)

   A user break occurs after an instruction of address H'00000404 is executed or before instructions of adresses H'00008010 to H'00008016 are executed.

2. Register specifications

   BARA = H'00037226, BAMRA = H'00000000, BBRA = H'0056, BARB = H'0003722E, BAMRB = H'00000000, BBRB = H'0056, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00300008

   Specified conditions: Channel A/channel B sequence mode

   - Channel A

     Address: H'00037226, Address mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/read/word

   - Channel B

     Address: H'0003722E, Address mask: H'00000000

     Data: H'00000000, Data mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/read/word

   An instruction with address H'00037226 is executed, and a user break occurs before an instruction with address H'0003722E is executed.

3. Register specifications

   BARA = H'00027128, BAMRA = H'00000000, BBRA = H'005A, BARB = H'00031415, BAMRB = H'00000000, BBRB = H'0054, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00300000

   Specified conditions: Channel A/channel B independent mode

   - Channel A

     Address: H'00027128, Address mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/write/word

   - Channel B

     Address: H'00031415, Address mask: H'00000000

     Data: H'00000000, Data mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/read (operand size is not included in the condition)

   On channel A, no user break occurs since instruction fetch is not a write cycle. On channel B, no user break occurs since instruction fetch is performed for an even address.

4. Register specifications

   BARA = H'00037226, BAMRA = H'00000000, BBRA = H'005A, BARB = H'0003722E, BAMRB = H'00000000, BBRB = H'0056, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00300008

   Specified conditions: Channel A/channel B sequence mode

   - Channel A

     Address: H'00037226, Address mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/write/word

   - Channel B

     Address: H'0003722E, Address mask: H'00000000

     Data: H'00000000, Data mask: H'00000000

     Bus cycle: CPU/instruction fetch (before instruction execution)/read/word

   Since instruction fetch is not a write cycle on channel A, a sequence condition does not match. Therefore, no user break occurs.

5. Register specifications

BARA = H'00000500, BAMRA = H'00000000, BBRA = H'0057, BARB = H'00001000, BAMRB = H'00000000, BBRB = H'0057, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00300001, BETR = H'0005

Specified conditions: Channel A/channel B independent mode

- Channel A

  Address: H'00000500, Address mask: H'00000000

  Bus cycle: CPU/instruction fetch (before instruction execution)/read/longword

- Channel B

  Address: H'00001000, Address mask: H'00000000

  Data: H'00000000, Data mask: H'00000000

  Bus cycle: CPU/instruction fetch (before instruction execution)/read/longword

  The number of execution-times break enable (5 times)

On channel A, a user break occurs before an instruction of address H'00000500 is executed. On channel B, a user break occurs before the fifth instruction execution after instructions of address H'00001000 are executed four times.

6. Register specifications

BARA = H'00008404, BAMRA = H'00000FFF, BBRA = H'0054, BARB = H'00008010, BAMRB = H'00000006, BBRB = H'0054, BDRB = H'00000000, BDMRB = H'00000000, BRCR = H'00300400

Specified conditions: Channel A/channel B independent mode

- Channel A

  Address: H'00008404, Address mask: H'00000FFF

  Bus cycle: CPU/instruction fetch (after instruction execution)/read (operand size is not included in the condition)

- Channel B

  Address: H'00008010, Address mask: H'00000006

  Data: H'00000000, Data mask: H'00000000

  Bus cycle: CPU/instruction fetch (before instruction execution)/read (operand size is not included in the condition)

A user break occurs after an instruction with address H'00008000 to H'00008FFE is executed or before instructions with addresses H'00008010 to H'00008016 are executed.

Break Condition Specified to a CPU Data Access Cycle

1. Register specifications

   BARA = H'00123456, BAMRA = H'00000000, BBRA = H'0064, BARB = H'000ABCDE, BAMRB = H'000000FF, BBRB = H'006A, BDRB = H'0000A512, BDMRB = H'00000000, BRCR = H'00300080

   Specified conditions: Channel A/channel B independent mode

   - Channel A

     Address: H'00123456, Address mask: H'00000000

     Bus cycle: CPU/data access/read (operand size is not included in the condition)

   - Channel B

     Address: H'000ABCDE, Address mask: H'000000FF

     Data: H'0000A512, Data mask: H'00000000

     Bus cycle: CPU/data access/write/word

   On channel A, a user break occurs with longword read to address H'00123454, word read to address H'00123456, or byte read to address H'00123456. On channel B, a user break occurs when word H'A512 is written in addresses H'000ABC00 to H'000ABCFE.

2. Register specifications:

   BARA = H'01000000, BAMRA = H'00000000, BBRA = H'0066, BARB = H'0000F000, BAMRB = H'FFFF0000, BBRB = H'036A, BDRB = H'00004567, BDMRB = H'00000000, BRCR = H'00300080

   Specified conditions: Channel A/channel B independent mode

   - Channel A

     Address: H'01000000, Address mask: H'00000000

     Bus cycle: CPU/data access/read/word

   - Channel B

     Y Address: H'0001F000, Address mask: H'FFFF0000

     Data: H'00004567, Data mask: H'00000000

     Bus cycle: CPU/data access/write/word

   On channel A, a user break occurs during word read to address H'01000000 on the memory space. On channel B, a user break occurs when word H'4567 is written in address H'0001F000 on Y-memory space. The X-/Y-memory space is changed by a mode specification. For details of the memory space, see figure 7.1.

Break Condition Specified to a DMAC Data Access Cycle

1. Register specifications:

   BARA = H'00314156, BAMRA = H'00000000, BBRA = H'0094, BARB = H'00055555, BAMRB = H'00000000, BBRB = H'00A9, BDRB = H'00000078, BDMRB = H'0000000F, BRCR = H'00300080

   Specified conditions: Channel A/channel B independent mode

   - Channel A

     Address: H'00314156, Address mask: H'00000000

     Bus cycle: DMAC/instruction fetch/read (operand size is not included in the condition)

   - Channel B

     Address: H'00055555, Address mask: H'00000000

     Data: H'00000078, Data mask: H'0000000F

     Bus cycle: DMAC/data access/write/byte

   On channel A, no user break occurs since instruction fetch is not performed in DMAC cycles. On channel B, a user break occurs when the DMAC writes byte H'7* in address H'00055555.

9.3.9 Notes

1. Only CPU can read/write UBC registers.
2. UBC cannot monitor CPU and DMAC access in the same channel.
3. Notes in specification of sequential break are described below:
    a. A condition match occurs when B-channel match occurs in a bus cycle after an A-channel match occurs in another bus cycle in sequential break setting. Therefore, no condition match occurs even if a bus cycle, in which an A-channel match and a B-channel match occur simultaneously, is set.
    b. Since the CPU has a pipeline configuration, the pipeline determines the order of an instruction fetch cycle and a memory cycle. Therefore, when a channel condition matches in the order of bus cycles, a sequential condition is satisfied.
    c. When the bus cycle condition for channel A is specified as a break before execution (PCBA = 0 in BRCR) and an instruction fetch cycle (in BBRA), the attention is as follows. A break is issued and condition match flags in BRCR are set to 1, when the bus cycle conditions both for channels A and B match simultaneously.
4. The change of a UBC register value is executed in MA (memory access) stage. Therefore, even if the break condition matches in the instruction fetch address following the instruction in which the pre-execution break is specified as the break condition, no break occurs. In order to know the timing UBC register is changed, read the last written register. Instructions after then are valid for the newly written register value.
5. Notes in specifying the instruction during repeat execution with repeat instruction as the break condition are as follows: When the instruction during repeat execution is specified as the break condition,
    a. The break is not issued during repeat execution, which has fewer than three instructions.
    b. When the execution times break is set, no instruction fetch from memory occurs during repeat execution under three instructions. Therefore, the execution times register BETR is not decreased.
6. The branch instruction should not be executed as soon as PC trace register BRSR and BRDR are read.

Section 10 Power-Down Modes

10.1 Overview

In the power-down modes, all CPU and some on-chip supporting module functions are halted. This lowers power consumption.

10.1.1 Power-Down Modes

The SH7622 has two power-down modes:

1. Standby mode
2. Module standby function

Table 10.1 shows the transition conditions for entering the modes from the program execution state, as well as the CPU and supporting module states in each mode and the procedures for canceling each mode.

Table 10.1 Power-Down Modes

		State								
Mode	**Transition Conditions**	**CPG**	**CPU**	**CPU Register**	**On-Chip Memory**	**On-Chip Peripheral Modules**	**USB**	**Pins**	**External Memory**	**Canceling Procedure**
Standby mode	Execute SLEEP instruction with STBY bit set to 1 in STBCR	Halted	Halted	Held	Held	Halted	Halted	Held	Self-refresh	1. NMI interrupt 2. Reset
Module standby function	Set MSTP bit of STBCR to 1	Running	Running or halted	Held	Held	Specified module halted	Specified module halted	*	Refresh	1. Clear MSTP bit to 0 2. Reset

Note: * Depends on the on-chip supporting module.
TMU external pin: Held
SCI external pin: Reset

10.1.2 Pin Configuration

Table 10.2 lists the pins used for the power-down modes.

Table 10.2 Pin Configuration

Pin Name	Symbol	I/O	Description
Processing state 1	STATUS1	O	Operating state of the processor HH: Reset, LH: Standby mode, LL: Normal operation
Processing state 0	STATUS0		

Note: H means high level, and L means low level.

10.1.3 Register Configuration

Table 10.3 shows the configuration of the control register for the power-down modes.

Table 10.3 Register Configuration

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Standby control register	STBCR	R/W	H'00*	H'FFFFFF82	8
Standby control register 2	STBCR2	R/W	H'00*	H'FFFFFF88	8
Standby control register 3	STBCR3	R/W	H'00	H'A4000A10	8

Note: * Initialized by power-on resets. This value is not initialized by manual resets but the contents are held.

10.2 Register Description

10.2.1 Standby Control Register (STBCR)

The standby control register (STBCR) is an 8-bit read/write register that sets the power-down mode. The STBCR is initialized to H'00 by a power-on reset.

Bit:	7	6	5	4	3	2	1	0
	STBY	—	—	—	—	MSTP2	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R	R	R	R/W	R	R

Bit 7—Standby (STBY): Specifies transition to standby mode.

Bit 7: STBY	Description	
0	Standby mode is disabled by execution of a SLEEP instruction	(Initial value)
1	Standby mode is enabled by execution of a SLEEP instruction	

Note: When putting the chip into standby mode, set the STBY bit to 1 before executing the SLEEP instruction. Operation is not guaranteed if STBY is not set to 1.

Bits 6 to 3, 1, 0—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 2—Module Standby 2 (MSTP2): Specifies halting the clock supply to the timer unit TMU (an on-chip supporting module). When the MSTP2 bit is set to 1, the supply of the clock to the TMU is halted.

Bit 2: MSTP2	Description	
0	TMU runs	(Initial value)
1	Clock supply to TMU is halted	

10.2.2 Standby Control Register 2 (STBCR2)

The standby control register 2 (STBCR2) is a read/write 8-bit register that specifies the power-down mode state.The STBCR2 is initialized to H'00 during a power-on reset.

Bit:	7	6	5	4	3	2	1	0
	—	—	MSTP8	MSTP7	—	MSTP5	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R	R/W	R	R

Bits 7, 6, 3 and 1—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 5— Module Stop 8 (MSTP8): Specifies halting the clock supply to the user break controller UBC (an on-chip supporting module). When the MSTP8 bit is set to 1, the supply of the clock to the UBC is halted.

Bit 5: MSTP8	Description	
0	UBC runs	(Initial value)
1	Clock supply to UBC is halted	

Bit 4—Module Stop 7 (MSTP7): Specifies halting of clock supply to the direct memory access controller DMAC (an on-chip peripheral module). When the MSTP7 bit is set to 1, the supply of the clock to the DMAC is halted.

Bit 4: MSTP7	Description	
0	DMAC runs	(Initial value)
1	Clock supply to DMAC is halted	

Bit 2—Module Stop 5 (MSTP5): Specifies halting of clock supply to the A/D converter ADC (an on-chip peripheral module). When the MSTP5 bit is set to 1, the supply of the clock to the ADC is halted.

Bit 2: MSTP5	Description	
0	ADC runs	(Initial value)
1	Clock supply to ADC is halted	

10.2.3 Standby Control Register 3 (STBCR3)

The standby control registr 3 (STBCR3) is a read/write 8-bit register that sets the power-down mode. The STBCR3 is initialized to H'00 during a power on reset.

Bit:	7	6	5	4	3	2	1	0
	—	—	MSTPE	MSTPD	—	MSTPB	MSTPA	MSTP9
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R	R/W	R/W	R/W

Bits 7, 6 and 3—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 5—Module Stop E (MSTPE): Specifies halting the clock supply to the compair match timer 1 CMT1 (an on-chip peripheral module). When the MSTPE bit is set to 1, the supply of the clock to the CMT1 is halted.

Bit 5: MSTPE	Description	
0	CMT1 runs	(Initial value)
1	Clock supply to CMT1 is halted	

Bit 4—Module Stop D (MSTPD): Specifies halting the clock supply to the USB function module USB (an on-chip peripheral module). When the MSTPD bit is set to 1, the supply of the clock to the USB is halted.

Bit 4: MSTPD	Description	
0	USB runs	(Initial value)
1	Clock supply to USB is halted	

Bit 2—Module Stop B (MSTPB): Specifies halting the clock supply to the serial communication interface 2 SCIF2 (an on-chip peripheral module). When the MSTPB bit is set to 1, the supply of the clock to the SCIF2 is halted.

Bit 2: MSTPB	Description	
0	SCIF2 runs	(Initial value)
1	Clock supply to SCIF2 is halted	

Bit 1—Module Stop A (MSTPA): Specifies halting the clock supply to the serial communication interface 1 SCIF1 (an on-chip peripheral module). When the MSTPA bit is set to 1, the supply of the clock to the SCIF1 is halted.

Bit 1: MSTPA	Description	
0	SCIF1 runs	(Initial value)
1	Clock supply to SCIF1 is halted	

Bit 0—Module Stop 9 (MSTP9): Specifies halting the clock supply to the serial communication interface 0 SCIF0 (an on-chip peripheral module). When the MSTP9 bit is set to 1, the supply of the clock to the SCIF0 is halted.

Bit 0: MSTP9	Description	
0	SCIF0 runs	(Initial value)
1	Clock supply to SCIF0 is halted	

10.3 Standby Mode

10.3.1 Transition to Standby Mode

To enter standby mode, set the STBY bit to 1 in STBCR, then execute the SLEEP instruction. The chip moves from the program execution state to standby mode. In standby mode, power consumption is greatly reduced by halting not only the CPU, but the clock and on-chip supporting modules as well. The clock output from the CKIO pin is also halted. CPU and cache register contents are held, but some on-chip supporting modules are initialized. Table 10.4 lists the states of registers in standby mode.

Table 10.4 Register States in Standby Mode

Module	Registers Initialized	Registers Retaining Data
Interrupt controller (INTC)	—	All registers
Clock pulse generator (CPG)	—	All registers
User break controller (UBC)	—	All registers
Bus state controller (BSC)	—	All registers
Timer unit (TMU)	TSTR register	Registers other than TSTR
A/D converter (ADC)	All registers	—
SCIF, CMT1, USB	—	All registers

The procedure for moving to standby mode is as follows:

1. Clear the TME bit in the WDT’s timer control register (WTCSR) to 0 to stop the WDT. Set the WDT’s timer counter (WTCNT) and the CKS2–CKS0 bits of the WTCSR register to appropriate values to secure the specified oscillation settling time.
2. After the STBY bit in the STBCR register is set to 1, a SLEEP instruction is executed.
3. Standby mode is entered and the clocks within the chip are halted. The STATUS1 pin output goes low and the STATUS0 pin output goes high.

10.3.2 Canceling Standby Mode

Standby mode is canceled by an NMI interrupt or a reset.

Canceling with an NMI Interrupt: The on-chip WDT can be used for hot starts. When the chip detects an NMI interrupt, the clock will be supplied to the entire chip and standby mode canceled after the time set in the WDT's timer control/status register has elapsed. The STATUS1 and STATUS0 pins both go low. Following this, the interrupt exception processing is executed, and NMI interrupt processing is performed. After branching to the interrupt processing routine occurs, clear the STBY bit in the STBCR register. The WDT stops automatically. If the STBY bit is not cleared, WDT continues operation and transits to the standby mode* when it reaches H'80. At this time, a manual reset is not accepted while the WDT is running. Immediately after an NMI interrupt is detected, the phase of the clock output of the CKIO pin may be unstable, until the processor starts interrupt processing.

Note: * Standby mode can be canceled only by power-on resets.

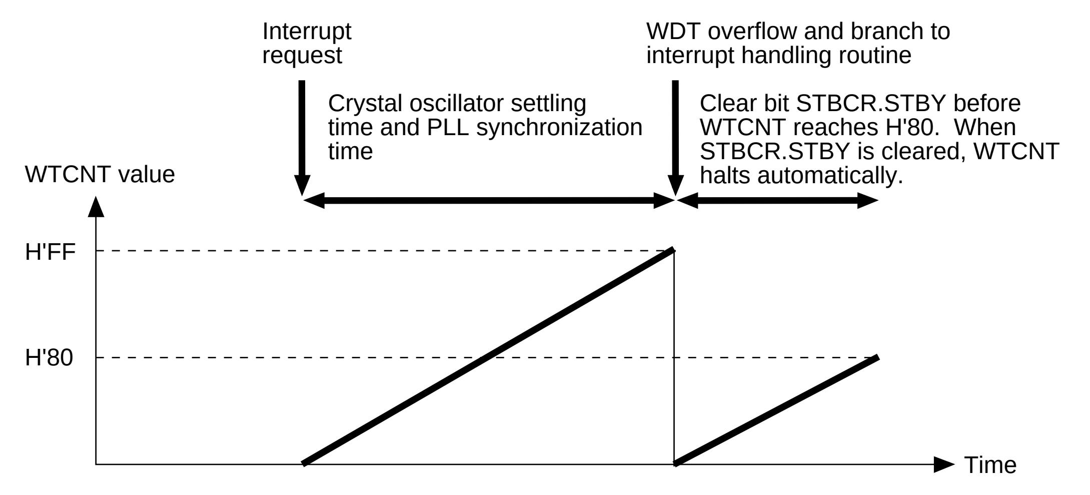


Figure 10.1 Canceling Standby Mode with STBCR.STBY

Canceling with a Reset: Standby mode can be canceled with a reset (power-on or manual). Keep the $\overline{\text{RESET}}$ pin low until the clock oscillation settles. The internal clock will continue to be output to the CKIO pin.

10.3.3 Usage Note

When standby mode is used in the SH7622, make the circuit connections shown in figure 10.2, and make a transition to standby mode in accordance with the sample software settings shown in figure 10.3.

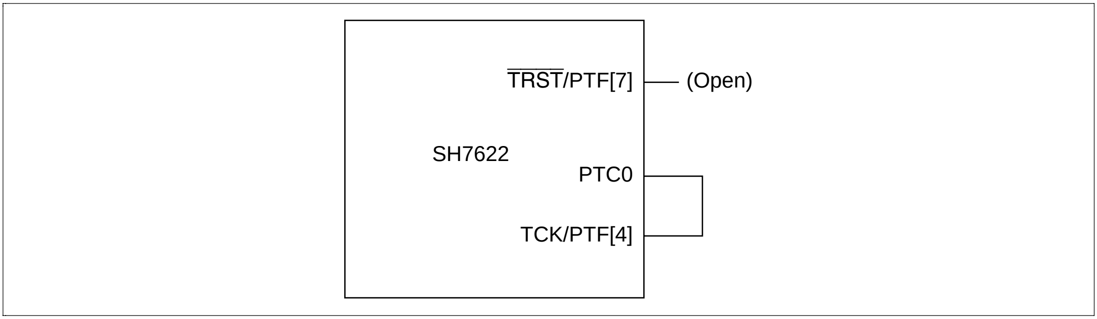


Figure 10.2 Example of Circuit Connections

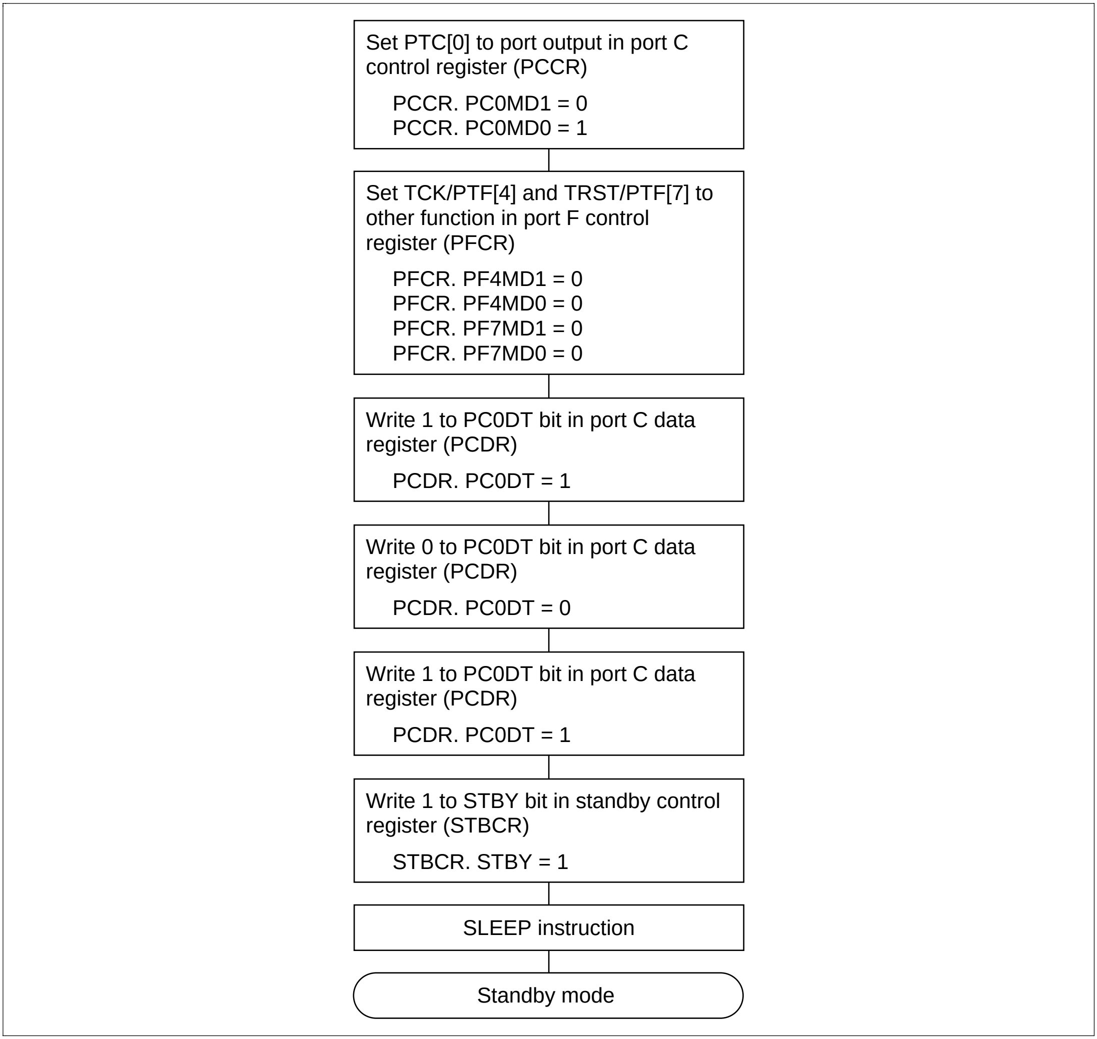

Set PTC[0] to port output in port C control register (PCCR)
PCCR. PC0MD1 = 0
PCCR. PC0MD0 = 1
Set TCK/PTF[4] and TRST/PTF[7] to other function in port F control register (PFCR)
PFCR. PF4MD1 = 0
PFCR. PF4MD0 = 0
PFCR. PF7MD1 = 0
PFCR. PF7MD0 = 0
Write 1 to PC0DT bit in port C data register (PCDR)
PCDR. PC0DT = 1
Write 0 to PC0DT bit in port C data register (PCDR)
PCDR. PC0DT = 0
Write 1 to PC0DT bit in port C data register (PCDR)
PCDR. PC0DT = 1
Write 1 to STBY bit in standby control register (STBCR)
STBCR. STBY = 1
SLEEP instruction
Standby mode

Figure 10.3 Example of Software Settings

10.4 Module Standby Function

10.4.1 Transition to Module Standby Function

Setting the standby control register MSTPE, MSTPD, MSTPB, MSTPA, MSTP9–MSTP7, MSTP5, and MSTP2 bits to 1 halts the supply of clocks to the corresponding on-chip supporting modules. This function can be used to reduce the power consumption in sleep mode. The module standby function holds the state prior to halt of the external pins of the on-chip supporting modules. TMU external pins hold their state prior to the halt. SCI external pins go to the reset state. With a few exceptions, all registers hold their values.

Bit	Register	Value	Description	
MSTP2	STBCR	0	TMU runs	(Initial value)
		1	Supply of clock to TMU is halted*	
MSTP5	STBCR2	0	ADC runs	(Initial value)
		1	Supply of clock to ADC is halted	
MSTP7	STBCR2	0	DMAC runs	(Initial value)
		1	Supply of clock to DMAC is halted	
MSTP8	STBCR2	0	UBC runs	(Initial value)
		1	Supply of clock to UBC is halted	
MSTP9	STBCR3	0	SCIF0 runs	(Initial value)
		1	Supply of clock to SCIF0 is halted	
MSTPA	STBCR3	0	SCIF1 runs	(Initial value)
		1	Supply of clock to SCIF1 is halted	
MSTPB	STBCR3	0	SCIF2 runs	(Initial value)
		1	Supply of clock to SCIF2 is halted	
MSTPD	STBCR3	0	USB runs	(Initial value)
		1	Supply of clock to USB is halted	
MSTPE	STBCR3	0	CMT1 runs	(Initial value)
		1	Supply of clock to CMT1 is halted	

Note: * The registers initialized are the same as in the standby mode.

10.4.2 Clearing the Module Standby Function

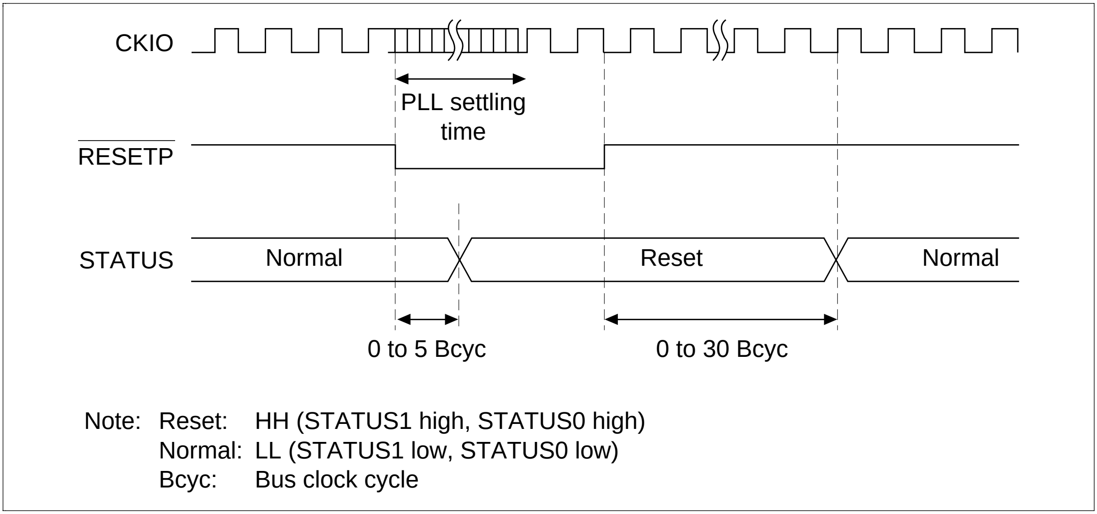
The module standby function can be cleared by clearing the MSTPE, MSTPD, MSTPB, MSTPA, MSTP9–MSTP7, MSTP5, and MSTP2 bits to 0, or by a power-on reset or manual reset.

10.5 Timing of STATUS Pin Changes

The timing of STATUS1 and STATUS0 pin changes is shown in figures 10.1 through 10.9.

10.5.1 Timing for Resets

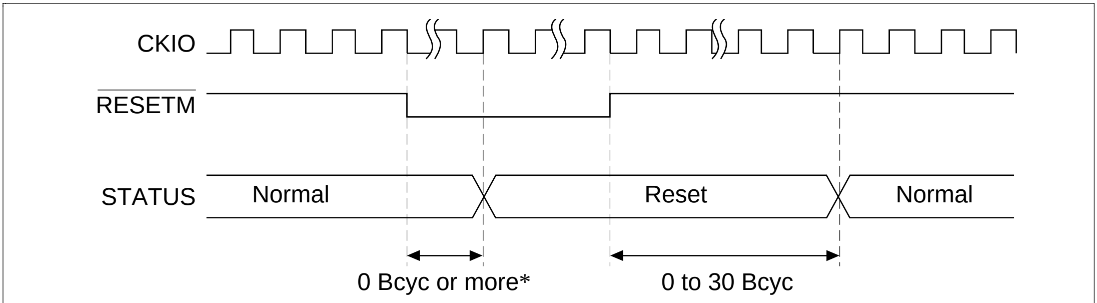
Power-On Reset



Note: Reset: HH (STATUS1 high, STATUS0 high)
Normal: LL (STATUS1 low, STATUS0 low)
Bcyc: Bus clock cycle

Figure 10.4 Power-On Reset STATUS Output

Manual Reset



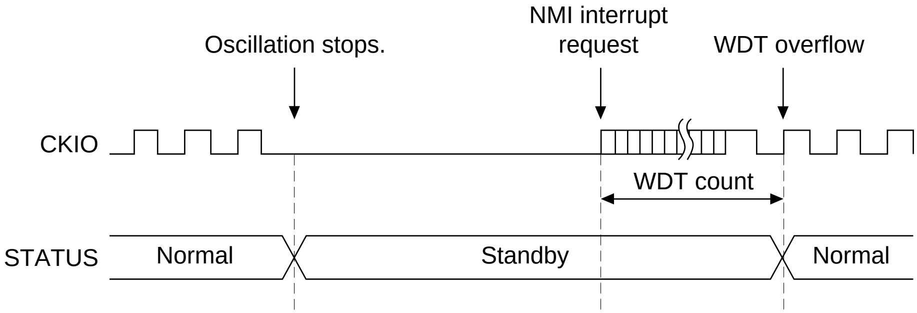
Note: * In a manual reset, STATUS becomes HH (reset) and the internal reset begins after waiting for the executing bus cycle to end.

Reset: HH (STATUS1 high, STATUS0 high)
Normal: LL (STATUS1 low, STATUS0 low)
Bcyc: Bus clock cycle

Figure 10.5 Manual Reset STATUS Output

10.5.2 Timing for Canceling Standbys

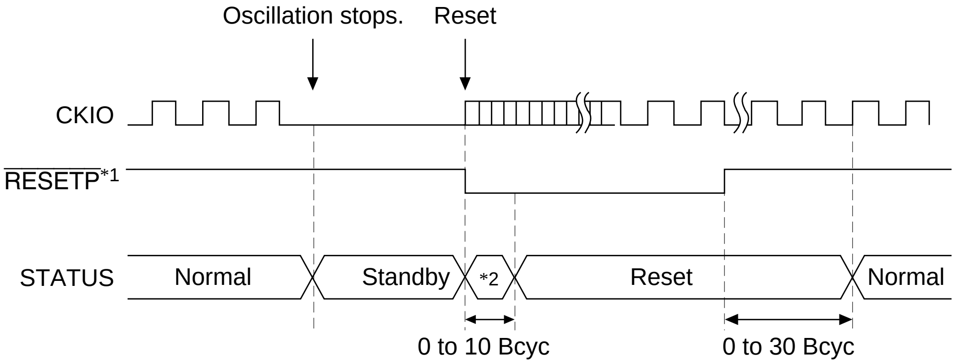
Standby to NMI Interrupt



Note: Standby: LH (STATUS1 low, STATUS0 high)
Normal: LL (STATUS1 low, STATUS0 low)

Figure 10.6 Standby to NMI Interrupt STATUS Output

Standby to Power-On Reset

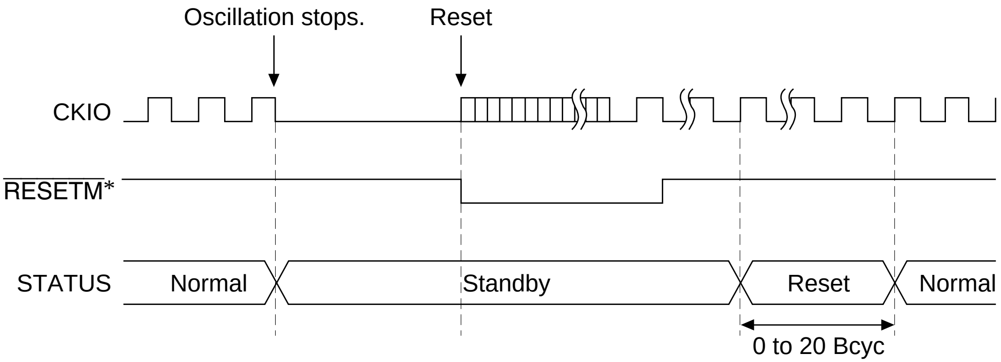


Notes: *1 When standby mode is cleared with a power-on reset, the WDT does not count. Keep $\overline{\text{RESETP}}$ low during the PLL's oscillation settling time.
*2 Undefined

Reset: HH (STATUS1 high, STATUS0 high)
Standby: LH (STATUS1 low, STATUS0 high)
Normal: LL (STATUS1 low, STATUS0 low)
Bcyc: Bus clock cycle

Figure 10.7 Standby to Power-On Reset STATUS Output

Standby to Manual Reset



Note: * When standby mode is cleared with a manual reset, the WDT does not count. Keep $\overline{\text{RESETM}}$ low during the PLL's oscillation settling time.

Reset:	HH (STATUS1 high, STATUS0 high)
Standby:	LH (STATUS1 low, STATUS0 high)
Normal:	LL (STATUS1 low, STATUS0 low)
Bcyc:	Bus clock cycle

Figure 10.8 Standby to Manual Reset STATUS Output

Section 11 On-Chip Oscillator Circuit

11.1 Overview

The on-chip oscillator circuit consists of a clock pulse generator (CPG) block and a watchdog timer (WDT) block.

The clock pulse generator (CPG) supplies all clocks to the processor and controls the power-down modes. The watchdog timer (WDT) is a single-channel timer that counts the clock settling time and is used when clearing standby mode and temporary standbys, such as frequency changes. It can also be used as an ordinary watchdog timer or interval timer.

11.1.1 Features

The CPG has the following features:

- Three clock modes: Selection of three clock modes for direct crystal input, and external clock input
- Three clocks generated independently: An internal clock for the CPU, cache, and TLB (Iϕ); a peripheral clock (Pϕ) for the on-chip supporting modules; and a bus clock (CKIO) for the external bus interface.
- Frequency change function: Internal and peripheral clock frequencies can be changed independently using the PLL circuit and divider circuit within the CPG. Frequencies are changed by software using frequency control register (FRQCR) settings.
- Power-down mode control: The clock can be stopped for sleep mode and standby mode and specific modules can be stopped using the module standby function.

The WDT has the following features:

- Can be used to ensure the clock settling time: Use the WDT to cancel standby mode and the temporary standbys which occur when the clock frequency is changed.
- Can switch between watchdog timer mode and interval timer mode.
- Generates internal resets in watchdog timer mode: Internal resets occur after counter overflow. Selection of power-on reset or manual reset.
- Generates interrupts in interval timer mode: Internal timer interrupts occur after counter overflow.
- Selection of eight counter input clocks. Eight clocks (×1 to ×1/4096) can be obtained by dividing the peripheral clock.

11.2 Overview of the CPG

11.2.1 CPG Block Diagram

A block diagram of the on-chip clock pulse generator is shown in figure 11.1.

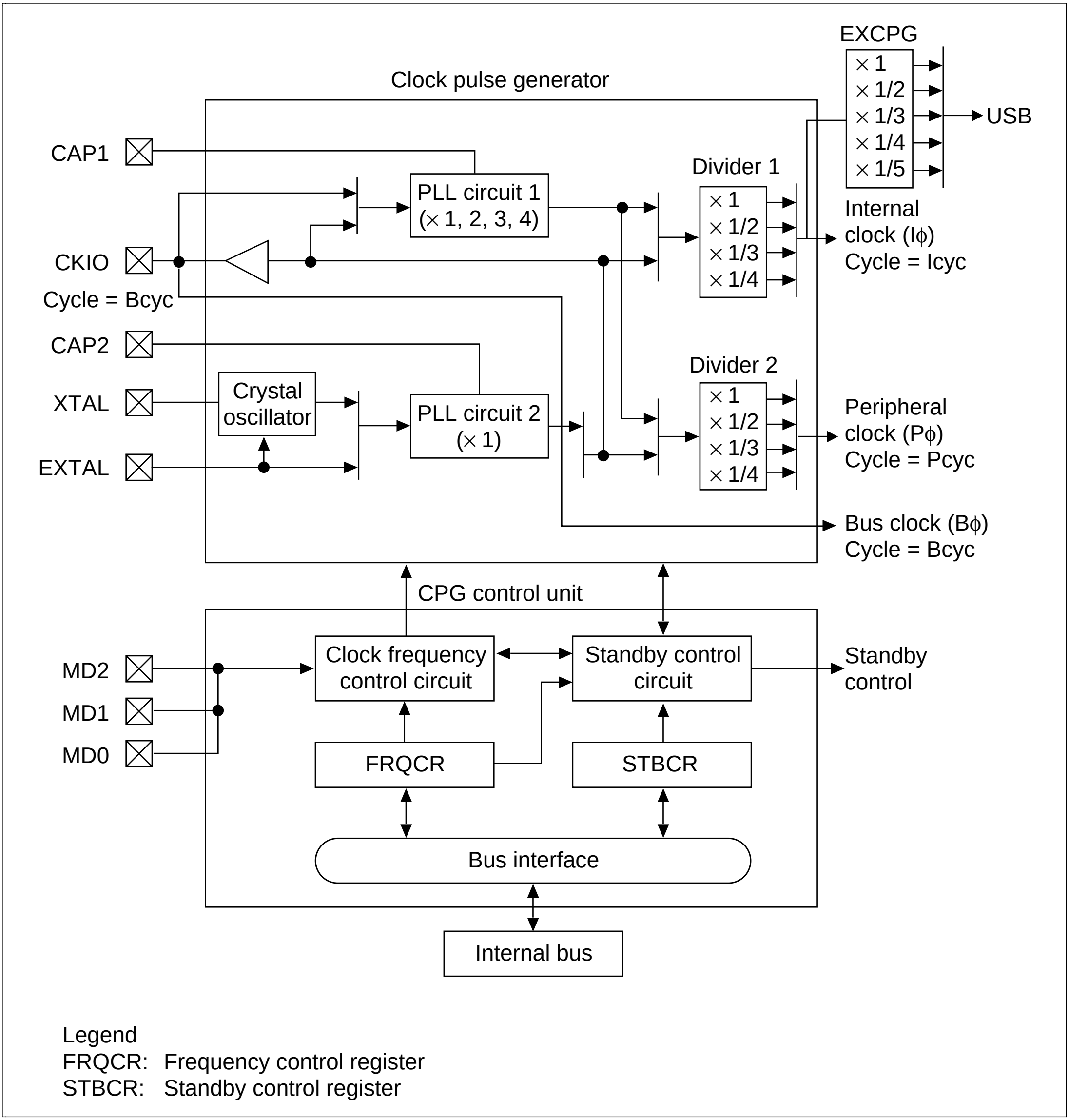


Figure 11.1 Block Diagram of Clock Pulse Generator

The clock pulse generator blocks function as follows:

1. PLL Circuit 1: PLL circuit 1 doubles, triples, quadruples, or leaves unchanged the input clock frequency from the CKIO terminal. The multiplication rate is set by the frequency control register. When this is done, the phase of the leading edge of the internal clock is controlled so that it will agree with the phase of the leading edge of the CKIO pin.
2. PLL Circuit 2: PLL circuit 2 leaves unchanged the frequency of the crystal oscillator or the input clock frequency coming from the EXTAL pin. The clock operation mode is set by pins MD0, MD1, and MD2. See table 11.3 for more information on clock operation modes.
3. Crystal Oscillator: This oscillator is used when a crystal oscillator element is connected to the XTAL and EXTAL pins. It operates according to the clock operating mode setting.
4. Divider 1: Divider 1 generates a clock at the operating frequency used by the internal clock. The operating frequency can be 1, 1/2, 1/3, or 1/4 times the output frequency of PLL circuit 1, as long as it stays at or above the clock frequency of the CKIO pin. The division ratio is set in the frequency control register.
5. Divider 2: Divider 2 generates a clock at the operating frequency used by the peripheral clock. The operating frequencies can be 1, 1/2, 1/3, or 1/4 times the output frequency of PLL circuit 1 or the clock frequency of the CKIO pin, as long as it stays at or below the clock frequency of the CKIO pin. The division ratio is set in the frequency control register.
6. Clock Frequency Control Circuit: The clock frequency control circuit controls the clock frequency using the MD pin and the frequency control register.
7. Standby Control Circuit: The standby control circuit controls the state of the clock pulse generator and other modules during clock switching and sleep/standby modes.
8. Frequency Control Register: The frequency control register has control bits assigned for the following functions: clock output/non-output from the CKIO pin, on/off control of PLL circuit 1, PLL standby, the frequency multiplication ratio of PLL circuit 1, and the frequency division ratio of the internal clock and the peripheral clock.
9. Standby Control Register: The standby control register has bits for controlling the power-down modes. See section 10, Power-Down Modes, for more information.

11.2.2 CPG Pin Configuration

Table 11.1 lists the CPG pins and their functions.

Table 11.1 Clock Pulse Generator Pins and Functions

Pin Name	Symbol	I/O	Description
Mode control pins	MD0 MD1 MD2	I	Set the clock operating mode
Crystal I/O pins (clock input pins)	XTAL	O	Connects a crystal oscillator
	EXTAL	I	Connects a crystal oscillator. Also used to input an external clock
Clock I/O pin	CKIO	I/O	Inputs or outputs an external clock
Capacitor connection pins for PLL	CAP1	I	Connects capacitor for PLL circuit 1 operation (recommended value 470 pF)
	CAP2	I	Connects capacitor for PLL circuit 2 operation (recommended value 470 pF)

11.2.3 CPG Register Configuration

Table 11.2 shows the CPG register configuration.

Table 11.2 Register Configuration

Register Name	Abbreviation	R/W	Initial Value	Address	Access Size
Frequency control register	FRQCR	R/W	H'0102	H'FFFFFF80	16

11.3 Clock Operating Modes

Table 11.3 shows the relationship between the mode control pin (MD2 to MD0) combinations and the clock operating modes. Table 11.4 shows the usable frequency ranges in the clock operating modes. Only the three clock modes shown below can be set. Operation cannot be guaranteed if any other clock mode is set.

Table 11.3 Clock Operating Modes

	Pin Values			Clock I/O		PLL2	PLL1	Divider 1	Divider 2	CKIO
Mode	MD2	MD1	MD0	Source	Output	On/Off	On/Off	Input	Input	Frequency
0	0	0	0	EXTAL	CKIO	On, multiplication ratio: 1	On	PLL1 output	PLL1	(EXTAL)
2	0	1	0	Crystal oscillator	CKIO	On, multiplication ratio: 1	On	PLL1 output	PLL1	(Crystal)
7	1	1	1	CKIO	—	Off	On	PLL1 output	PLL1	(CKIO)

MODE 0: An external clock is input from the EXTAL pin and undergoes waveform shaping by PLL circuit 2 before being supplied inside the chip. PLL circuit 1 is constantly on. An input clock frequency of 20 MHz to 33 MHz can be used, and the CKIO frequency range is 20 MHz to 33 MHz.

MODE 2: The on-chip crystal oscillator operates, and oscillation frequency wave shaping is performed by PLL circuit 2 before the signal is supplied inside the chip. A crystal with an oscillation frequency of 5 MHz to 25 MHz can be used, and the CKIO frequency range is 5 MHz to 25 MHz.

MODE 7: In this mode, the CKIO pin is an input, an external clock is input to this pin, and undergoes waveform shaping , and also frequency multiplication according to the setting, by PLL circuit 1 before being supplied to the chip. In modes 0 and 2, the system clock is generated from the output of the chip's CKIO pin. Consequently, if a large number of chips are operating on the clock cycle, the CKIO pin load will be large. This mode, however, assumes a comparatively large-scale system. If a large number of chips are operating on the clock cycle, a clock generator with a number of low-skew clock outputs can be provided, so that the chips can operate cyclically by distributing the clocks to each one. An input clock frequency of 20 MHz to 33 MHz can be used.

As PLL circuit 1 compensates for fluctuations in the CKIO pin load, this mode is suitable for connection of synchronous DRAM.

Table 11.4 Available Combination of Clock Mode and FRQCR Values

Clock Mode	FRQCR	PLL1	PLL2	Clock Rate* (I:B:P)
0	H'0100	ON (×1)	ON (×1)	1:1:1
	H'0101	ON (×1)	ON (×1)	1:1:1/2
	H'0102	ON (×1)	ON (×1)	1:1:1/4
	H'0111	ON (×2)	ON (×1)	2:1:1
	H'0112	ON (×2)	ON (×1)	2:1:1/2
	H'0115	ON (×2)	ON (×1)	1:1:1
	H'0116	ON (×2)	ON (×1)	1:1:1/2
	H'0122	ON (×4)	ON (×1)	4:1:1
	H'0126	ON (×4)	ON (×1)	2:1:1
	H'012A	ON (×4)	ON (×1)	1:1:1
	H'A100	ON (×3)	ON (×1)	3:1:1
	H'A101	ON (×3)	ON (×1)	3:1:1/2
	H'E100	ON (×3)	ON (×1)	1:1:1
	H'E101	ON (×3)	ON (×1)	1:1:1/2

Table 11.4 Available Combination of Clock Mode and FRQCR Values (cont)

Clock Mode	FRQCR	PLL1	PLL2	Clock Rate* (I:B:P)
2	H'0100	ON (×1)	ON (×1)	1:1:1
	H'0101	ON (×1)	ON (×1)	1:1:1/2
	H'0102	ON (×1)	ON (×1)	1:1:1/4
	H'0111	ON (×2)	ON (×1)	2:1:1
	H'0112	ON (×2)	ON (×1)	2:1:1/2
	H'0115	ON (×2)	ON (×1)	1:1:1
	H'0116	ON (×2)	ON (×1)	1:1:1/2
	H'0122	ON (×4)	ON (×1)	4:1:1
	H'0126	ON (×4)	ON (×1)	2:1:1
	H'012A	ON (×4)	ON (×1)	1:1:1
	H'A100	ON (×3)	ON (×1)	3:1:1
	H'A101	ON (×3)	ON (×1)	3:1:1/2
	H'E100	ON (×3)	ON (×1)	1:1:1
	H'E101	ON (×3)	ON (×1)	1:1:1/2
7	H'0100	ON (×1)	OFF	1:1:1
	H'0101	ON (×1)	OFF	1:1:1/2
	H'0102	ON (×1)	OFF	1:1:1/4
	H'0111	ON (×2)	OFF	2:1:1
	H'0112	ON (×2)	OFF	2:1:1/2
	H'0115	ON (×2)	OFF	1:1:1
	H'0116	ON (×2)	OFF	1:1:1/2
	H'0122	ON (×4)	OFF	4:1:1
	H'0126	ON (×4)	OFF	2:1:1
	H'012A	ON (×4)	OFF	1:1:1
	H'A100	ON (×3)	OFF	3:1:1
	H'E100	ON (×3)	OFF	1:1:1
	H'E101	ON (×3)	OFF	1:1:1/2

Note: * With input clock as 1

Maximum frequencies: Iϕ = 100 MHz, Bϕ (CKIO) = 33 MHz, Pϕ = 33 MHz, input clock = 33 MHz (where Bϕ ≥ Pϕ)

Cautions:

1. The input to divider 1 becomes the output of PLL circuit 1
2. The input of divider 2 becomes the output of PLL circuit 1
3. The frequency of the internal clock (Iϕ) becomes as follows:
   - The product of the frequency of the CKIO pin, the frequency multiplication ratio of PLL circuit 1, and the division ratio of divider 1 when PLL circuit 1 is on.
   - Do not set the internal clock frequency lower than the CKIO pin frequency.
4. The frequency of the peripheral clock (Pϕ) becomes as follows:
   - The product of the frequency of the CKIO pin, the frequency multiplication ratio of PLL circuit 1, and the division ratio of divider 2.
   - The peripheral clock frequency should not be set higher than the frequency of the CKIO pin, or lower than 1/6 the internal clock (Iϕ).
5. The output frequency of PLL circuit 1 is the product of the CKIO frequency and the multiplication ratio of PLL circuit 1.
6. × 1, × 2, × 3, or × 4 can be used as the multiplication ratio of PLL circuit 1. × 1, × 1/2, × 1/3, and × 1/4 can be selected as the division ratios of dividers 1 and 2. Set the rate in the frequency control register.
7. The maximum frequency setting for the crystal resonator must be made after consulting the manufacturer of the crystal resonator to be used.

11.4 Register Descriptions

11.4.1 Frequency Control Register (FRQCR)

The frequency control register (FRQCR) is a 16-bit read/write register used to specify whether a clock is output from the CKIO pin, the on/off state of PLL circuit 1, PLL standby, the frequency multiplication ratio of PLL circuit 1, and the frequency division ratio of the internal clock and the peripheral clock.

Only word access can be used on the FRQCR register. FRQCR is initialized to H'0102 by a power-on reset, but retains its value in a manual reset and in standby mode.

Bit:	15	14	13	12	11	10	9	8
	STC2	IFC2	PFC2	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	1
R/W:	R/W	R/W	R/W	R	R	R	R	R

Bit:	7	6	5	4	3	2	1	0
	—	—	STC1	STC0	IFC1	IFC0	PFC1	PFC0
Initial value:	0	0	0	0	0	0	1	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15, 5, and 4—Frequency Multiplication Ratio (STC2, STC1, STC0): These bits specify the frequency multiplication ratio of PLL circuit 1.

Bit 15: STC2	Bit 5: STC1	Bit 4: STC0	Description	
0	0	0	× 1	(Initial value)
0	0	1	× 2	
1	0	0	× 3	
0	1	0	× 4	

Bits 14, 3, and 2—Internal Clock Frequency Division Ratio (IFC2, IFC1, IFC0): These bits specify the frequency division ratio of the internal clock with respect to the output frequency of PLL circuit 1.

Bit 14: IFC2	Bit 3: IFC1	Bit 2: IFC0	Description	
0	0	0	× 1	(Initial value)
0	0	1	× 1/2	
1	0	0	× 1/3	
0	1	0	× 1/4	

Note: Do not set the internal clock frequency lower than the CKIO frequency.

Bits 13, 1, and 0—Peripheral Clock Frequency Division Ratio (PFC2, PFC1, PFC0): These bits specify the division ratio of the peripheral clock frequency with respect to the frequency of the output frequency of PLL circuit 1 or the frequency of the CKIO pin.

Bit 13: PFC2	Bit 1: PFC1	Bit 0: PFC0	Description	
0	0	0	× 1	
0	0	1	× 1/2	
1	0	0	× 1/3	
0	1	0	× 1/4	(Initial value)

Note: Do not set the peripheral clock frequency higher than the frequency of the CKIO pin.

Bits 12 to 9, 7, and 6—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 8—Reserved: This bit is always read as 1. The write value should always be 1.

11.5 Changing the Frequency

The frequency of the internal clock and peripheral clock can be changed either by changing the multiplication rate of PLL circuit 1 or by changing the division rates of dividers 1 and 2. All of these are controlled by software through the frequency control register. The methods are described below.

11.5.1 Changing the Multiplication Rate

A PLL settling time is required when the multiplication rate of PLL circuit 1 is changed. The on-chip WDT counts the settling time.

1. In the initial state, the multiplication rate of PLL circuit 1 is 1.
2. Set a value that will become the specified oscillation settling time in the WDT and stop the WDT. The following must be set:
   WTCSR register TME bit = 0: WDT stops
   WTCSR register CKS2–CKS0 bits: Division ratio of WDT count clock
   WTCNT counter: Initial counter value
3. Set the desired value in the STC2, STC1 and STC0 bits. The division ratio can also be set in the IFC2–IFC0 bits and PFC2–PFC0 bits.
4. The processor pauses internally and the WDT starts incrementing. The internal and peripheral clocks both stop. The clock will continue to be output at the CKIO pin.
5. Supply of the clock that has been set begins at WDT count overflow, and the processor begins operating again. The WDT stops after it overflows.

11.5.2 Changing the Division Ratio

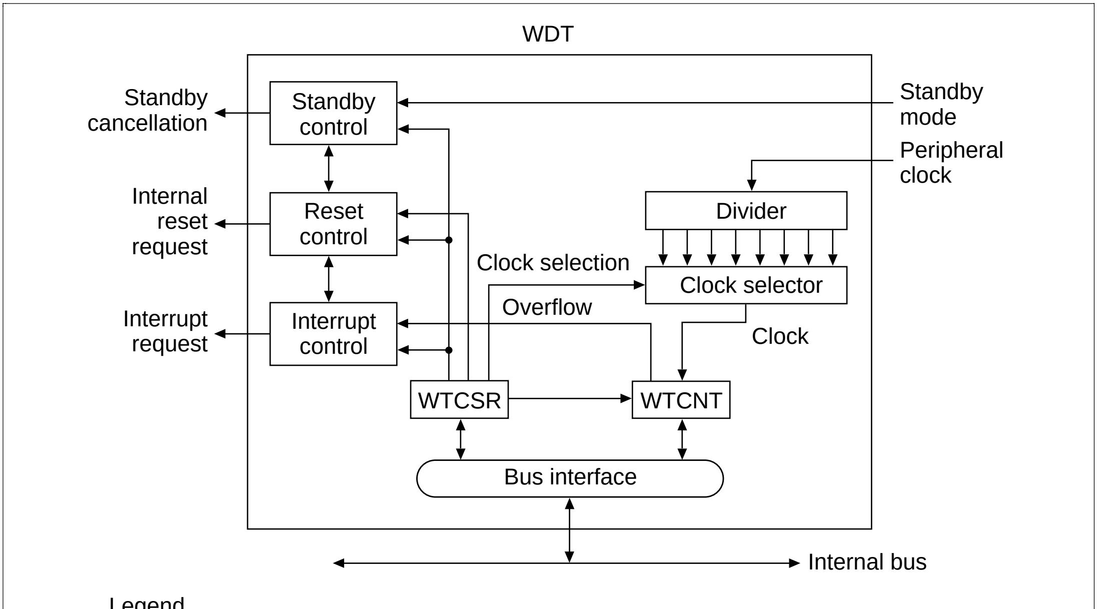
The WDT will not count unless the multiplication rate is changed simultaneously.

1. In the initial state, IFC2–IFC0 = 000 and PFC2–PFC0 = 010.
2. Set the IFC2, IFC1, IFC0, PFC2, PFC1, and PFC0 bits to the new division ratio. The values that can be set are limited by the clock mode and the multiplication rate of PLL circuit 1. Note that if the wrong value is set, the processor will malfunction.
3. The clock is immediately supplied at the new division ratio.

11.6 Overview of the WDT

11.6.1 Block Diagram of the WDT

Figure 11.2 shows a block diagram of the WDT.



Legend
WTCSR: Watchdog timer control/status register
WTCNT: Watchdog timer counter

Figure 11.2 Block Diagram of the WDT

11.6.2 Register Configurations

The WDT has two registers that select the clock, switch the timer mode, and perform other functions. Table 11.5 shows the WDT register.

Table 11.5 Register Configuration

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Watchdog timer counter	WTCNT	R/W*	H'00	H'FFFFFF84	R: 8 W: 16*
Watchdog timer control/status register	WTCSR	R/W*	H'00	H'FFFFFF86	R: 8 W: 16*

Note: * Write with a word access. Write H'5A and H'A5, respectively, in the upper bytes. Byte or longword writes are not possible. Read with a byte access.

11.7 WDT Registers

11.7.1 Watchdog Timer Counter (WTCNT)

The watchdog timer counter (WTCNT) is an 8-bit read/write register that increments on the selected clock. When an overflow occurs, it generates a reset in watchdog timer mode and an interrupt in interval time mode. Its address is H'FFFFFF84. The WTCNT counter is initialized to H'00 only by a power-on reset through the $\overline{\text{RESETP}}$ pin. Use a word access to write to the WTCNT counter, with H'5A in the upper byte. Use a byte access to read WTCNT.

Note: The WTCNT differs from other registers in that it is more difficult to write to. See section 11.7.3, Notes on Register Access, for details.

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

11.7.2 Watchdog Timer Control/Status Register (WTCSR)

The watchdog timer control/status register (WTCSR) is an 8-bit read/write register composed of bits to select the clock used for the count, bits to select the timer mode, and overflow flags. Its address is H'FFFFFF86. The WTCSR register is initialized to H'00 only by a power-on reset through the $\overline{\text{RESETP}}$ pin. When a WDT overflow causes an internal reset, the WTCSR retains its value. When used to count the clock settling time for canceling a standby, it retains its value after counter overflow. Use a word access to write to the WTCSR counter, with H'A5 in the upper byte. Use a byte access to read WTCSR.

Note: The WTCSR differs from other registers in that it is more difficult to write to. See section 11.7.3, Notes on Register Access, for details.

Bit:	7	6	5	4	3	2	1	0
	TME	WT/$\overline{\text{IT}}$	RSTS	WOVF	IOVF	CKS2	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7—Timer Enable (TME): Starts and stops timer operation. Clear this bit to 0 when using the WDT in standby mode or when changing the clock frequency.

Bit 7: TME	Description
0	Timer disabled: Count-up stops and WTCNT value is retained (Initial value)
1	Timer enabled

Bit 6—Timer Mode Select (WT/$\overline{\text{IT}}$): Selects whether to use the WDT as a watchdog timer or an interval timer.

Bit 6: WT/$\overline{\text{IT}}$	Description
0	Use as interval timer (Initial value)
1	Use as watchdog timer

Note: If WT/$\overline{\text{IT}}$ is modified when the WDT is running, the up-count may not be performed correctly.

Bit 5—Reset Select (RSTS): Selects the type of reset when the WTCNT overflows in watchdog timer mode. In interval timer mode, this setting is ignored.

Bit 5: RSTS	Description
0	Power-on reset (Initial value)
1	Manual reset

Bit 4—Watchdog Timer Overflow (WOVF): Indicates that the WTCNT has overflowed in watchdog timer mode. This bit is not set in interval timer mode.

Bit 4: WOVF	Description
0	No overflow (Initial value)
1	WTCNT has overflowed in watchdog timer mode

Bit 3—Interval Timer Overflow (IOVF): Indicates that the WTCNT has overflowed in interval timer mode. This bit is not set in watchdog timer mode.

Bit 3: IOVF	Description
0	No overflow (Initial value)
1	WTCNT has overflowed in interval timer mode

Bits 2 to 0—Clock Select 2 to 0 (CKS2, CKS1, CKS0): These bits select the clock to be used for the WTCNT count from the eight types obtainable by dividing the peripheral clock. The overflow period in the table is the value when the peripheral clock (Pϕ) is 15 MHz.

Bit 2: CKS2	Bit 1: CKS1	Bit 0: CKS0	Clock Division Ratio	Overflow Period (when Pϕ = 15 MHz)
0	0	0	1 (Initial value)	17 µs
		1	1/4	68 µs
	1	0	1/16	273 µs
		1	1/32	546 µs
1	0	0	1/64	1.09 ms
		1	1/256	4.36 ms
	1	0	1/1024	17.46 ms
		1	1/4096	69.84 ms

Note: If bits CKS2–CKS0 are modified when the WDT is running, the up-count may not be performed correctly. Ensure that these bits are modified only when the WDT is not running.

11.7.3 Notes on Register Access

The watchdog timer counter (WTCNT) and watchdog timer control/status register (WTCSR) are more difficult to write to than other registers. The procedure for writing to these registers are given below.

Writing to WTCNT and WTCSR: These registers must be written by a word transfer instruction. They cannot be written by a byte or longword transfer instruction. When writing to WTCNT, set the upper byte to H'5A and transfer the lower byte as the write data, as shown in figure 11.3. When writing to WTCSR, set the upper byte to H'A5 and transfer the lower byte as the write data. This transfer procedure writes the lower byte data to WTCNT or WTCSR.

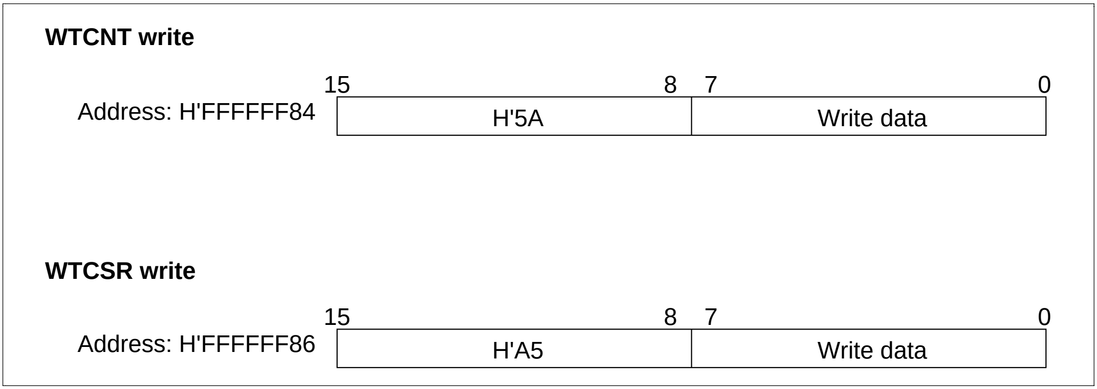


Figure 11.3 Writing to WTCNT and WTCSR

11.8 Using the WDT

11.8.1 Canceling Standbys

The WDT can be used to cancel standby mode with an NMI or other interrupts. The procedure is described below. (The WDT does not run when resets are used for canceling, so keep the RESET pin low until the clock stabilizes.)

1. Before transitioning to standby mode, always clear the TME bit in WTCSR to 0. When the TME bit is 1, an erroneous reset or interval timer interrupt may be generated when the count overflows.
2. Set the type of count clock used in the CKS2–CKS0 bits in WTCSR and the initial values for the counter in the WTCNT counter. These values should ensure that the time till count overflow is longer than the clock oscillation settling time.
3. Move to standby mode by executing a SLEEP instruction to stop the clock.
4. The WDT starts counting by detecting the edge change of the NMI signal or detecting interrupts.
5. When the WDT count overflows, the CPG starts supplying the clock and the processor resumes operation. The WOVF flag in WTCSR is not set when this happens.
6. Since the WDT continues counting from H’00, set the STBY bit in the STBCR register to 0 in the interrupt processing program and this will stop the WDT. When the STBY bit remains 1, the SH7622 again enters the standby mode when the WDT has counted up to H’80. This standby mode can be canceled by power-on resets.

11.8.2 Changing the Frequency

To change the frequency used by the PLL, use the WDT. When changing the frequency only by switching the divider, do not use the WDT.

1. Before changing the frequency, always clear the TME bit in WTCSR to 0. When the TME bit is 1, an erroneous reset or interval timer interrupt may be generated when the count overflows.
2. Set the type of count clock used in the CKS2–CKS0 bits of WTCSR and the initial values for the counter in the WTCNT counter. These values should ensure that the time till count overflow is longer than the clock oscillation settling time.
3. When the frequency control register (FRQCR) is written, internal processor operation stops temporarily and the WDT starts counting.
4. When the WDT count overflows, the CPG resumes supplying the clock and the processor resumes operation. The WOVF flag in WTCSR is not set when this happens.
5. The counter stops at the values H'00–H'01. The stop value depends on the clock ratio.

6. When writing to WTCNT after the frequency is changed, read WTCNT and confirm that its value is H'00 before performing the write.

11.8.3 Using Watchdog Timer Mode

1. Set the WT/$\overline{\text{IT}}$ bit in the WTCSR register to 1, set the reset type in the RSTS bit, set the type of count clock in the CKS2–CKS0 bits, and set the initial value of the counter in the WTCNT counter.
2. Set the TME bit in WTCSR to 1 to start the count in watchdog timer mode.
3. While operating in watchdog timer mode, rewrite the counter periodically to H'00 to prevent the counter from overflowing.
4. When the counter overflows, the WDT sets the WOVF flag in WTCSR to 1 and generates the type of reset specified by the RSTS bit. The counter then resumes counting.

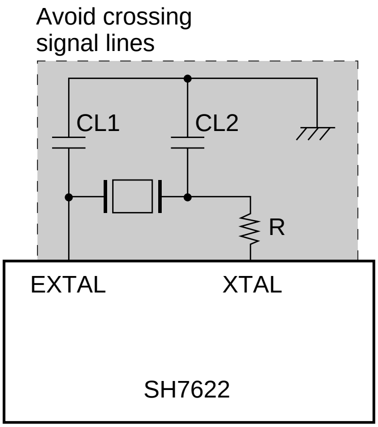
11.8.4 Using Interval Timer Mode

When operating in interval timer mode, interval timer interrupts are generated at every overflow of the counter. This enables interrupts to be generated at set periods.

1. Clear the WT/$\overline{\text{IT}}$ bit in the WTCSR register to 0, set the type of count clock in the CKS2–CKS0 bits, and set the initial value of the counter in the WTCNT counter.
2. Set the TME bit in WTCSR to 1 to start the count in interval timer mode.
3. When the counter overflows, the WDT sets the IOVF flag in WTCSR to 1 and an interval timer interrupt request is sent to INTC. The counter then resumes counting.

11.9 Notes on Board Design

When Using an External Crystal Resonator: Place the crystal resonator, capacitors CL1 and CL2, and damping resistor R close to the EXTAL and XTAL pins. To prevent induction from interfering with correct oscillation, use a common grounding point for the capacitors connected to the resonator, and do not locate a wiring pattern near these components.



Note: The values for CL1, CL2, and the damping resistance should be determined after consultation with the crystal manufacturer.

Figure 11.4 Points for Attention when Using Crystal Resonator

Decoupling Capacitors: Insert a laminated ceramic capacitor of 0.1 to 1 µF as a passive capacitor for each V_{SS}/V_{CC} pair. Mount the passive capacitors to the SH-3 power supply pins, and use components with a frequency characteristic suitable for the SH-3 operating frequency, as well as a suitable capacitance value.

V_{SS}/V_{CC} pairs (example of 208-pin LQFP version): 19-21, 27-29, 33-35, 45-47, 57-59, 69-71, 79-81, 83-85, 95-97, 109-111, 132-134, 153-154, 161-163, 173-175, 181-183, 205-208, 3-6, 145-147, 148-150

When Using a PLL Oscillator Circuit: Keep the wiring from the PLL V_{CC} and V_{SS} connection pattern to the power supply pins short, and make the pattern width large, to minimize the inductance component. Ground the oscillation stabilization capacitors C1 and C2 to V_{SS} (PLL1) and V_{SS} (PLL2), respectively. Place C1 and C2 close to the CAP1 and CAP2 pins and do not locate a wiring pattern in the vicinity.

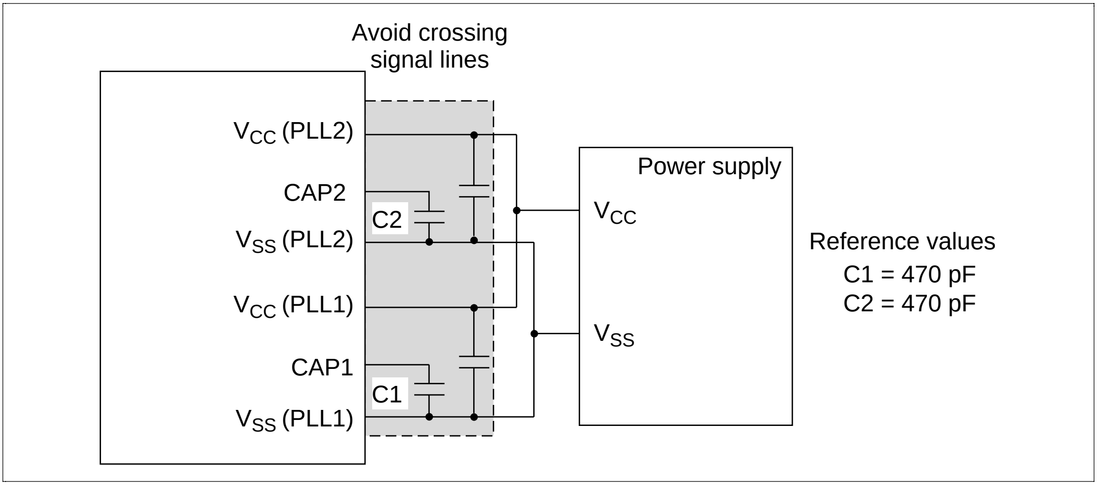


Figure 11.5 Points for Attention when Using PLL Oscillator Circuit

11.10 Usage Notes

1. When changing the multiplication factor, either turn the cache off or execute the change in a non-cacheable area.
2. When using the cache after changing the multiplication factor, leave an interval of at least 30 external bus clock cycles after the CPU is restarted when the WDT overflows and clocks are supplied within the chip.

Section 12 Extend Clock Pulse Generator for USB (EXCPG)

12.1 Overview of EXCPG

12.1.1 EXCPG Features

The SH7622 has an on-chip USB function module (USB). Use of USB requires a fixed 48 MHz clock for the USB.

The extend clock pulse generator (EXCPG) is a module that uses the CPU clock (Iϕ) or an external input clock as the source clock, and uses frequency dividers to generate the fixed clocks required by the USB.

12.1.2 EXCPG Configuration

Figure 12.1 shows a block diagram of the EXCPG.

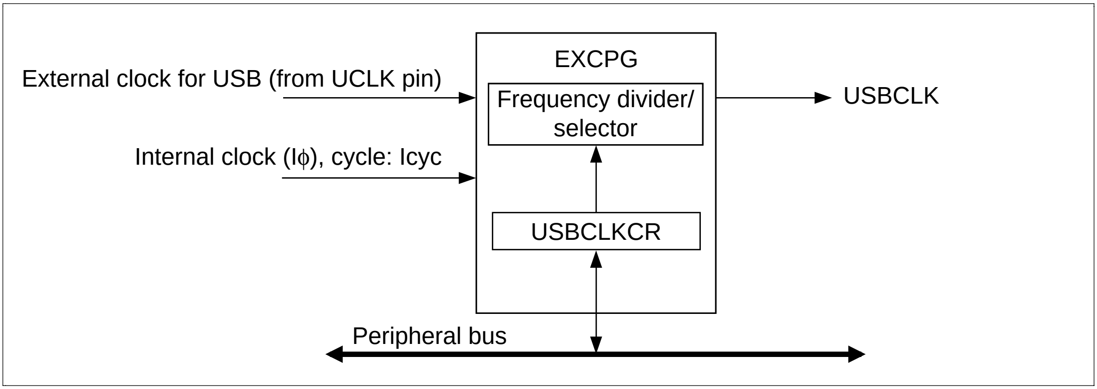

External clock for USB (from UCLK pin)
Internal clock (Iϕ), cycle: Icyc
EXCPG
Frequency divider/ selector
USBCLKCR
USBCLK
Peripheral bus

Figure 12.1 Block Diagram of EXCPG

The EXCPG performs frequency divider and source clock selection according to the settings in the USBCLKCR registers, and outputs the USB clock (USBCLK).

12.1.3 Register Configuration

The EXCPG has the internal registers shown in the following table.

Name	Abbreviation	R/W	Initial Value	Address	Access Size
USB clock control register	USBCLKCR	R/W	H'C0	H'A4000A20	R:8, W:16

Use word-size (16-bit) writes and byte-size (8-bit) reads on these registers.

12.2 Register Descriptions

12.2.1 USB Clock Control Register (USBCLKCR)

The USB clock control register (USBCLKCR) is an 8-bit readable register that selects the source clock and division ratio for generation of the USB clock.

USBCLKCR is initialized to H'00 by a power-on reset.

Bit:	7	6	5	4	3	2	1	0
	USSCS1	USSCS0	—	—	—	—	—	USDIVS0
Initial value:	1	1	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R	R/W

Word-size access is used to write to USBCLKCR. To prevent inadvertent overwriting, writes are performed with H'A5 in the upper byte and the write data in the lower byte.

To write to USBCLKCR:

15 8	7 0
H'A5	Write data

Bit Descriptions

Bits 7 and 6—Source Clock Select (USSCS1, USSCS0): These bits select the source clock.

Bit 7: USSCS1	Bit 6: USSCS0	Function (Source Clock Selection)
0	0	Clock stopped
	1	Iφ
1	0	Reserved (Do not set)
	1	External input clock (Initial value)

Bits 5 to 1—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 0—Divider Select (USDIVS0): Selects the source clock frequency division ratio. Use the setting that gives a 48 MHz output clock. This bit is valid only when the source clock is Iϕ. With external clock input, the division ratio is ×1 regardless of the setting of this bit.

Bit 0: USDIVS0	Function (Divider Selection)	
0	×1	(Initial value)
1	×1/2	

12.3 Usage Notes

Note the following when using the EXCPG. If the EXCPG is used incorrectly, the correct clocks may not be generated, causing faulty operation of the USB.

- The EXCPG is used only for generation of the USB clocks. When the USB is not used, it is recommended that the USBCLKCR register be cleared to H'00 to halt the clock.

- The USBCLKCR registers are initialized only by a power-on reset. In a manual reset, they retain their current set values.

- Word-size access must be used to write to the USBCLKCR registers. Byte-size or longword-size access cannot used. H'A5 must be set in the upper byte when writing to USBCLKCR. This is necessary to prevent inadvertent overwriting of these registers.

- If the source clock frequency is changed when the PLL circuit multiplication factor is changed by means of the FRQCR register (CPG: frequency control register), USBCLKCR register settings must be made again. If the USB source clock is an internal input clock, the USB module must be halted. This is done by selecting the “Clock stopped” setting with the source clock bits in the EXCPG’s USBCLKCR registers.

- Do not make settings in the USBCLKCR register during the oscillation stabilization period associated with a change of the PLL circuit multiplication factor. Wait until the oscillation stabilization time has elapsed before making any settings.

- If Iϕ is used as the source clock, standby mode cannot be used.

Section 13 Bus State Controller (BSC)

13.1 Overview

The bus state controller (BSC) divides physical address space and output control signals for various types of memory and bus interface specifications. BSC functions enable this LSI to link directly with synchronous DRAM, SRAM, ROM, and other memory storage devices without an external circuit.

13.1.1 Features

The BSC has the following features:

- Physical address space is divided into six areas.
  - A maximum 64 Mbytes for each of the six areas, 0, 2–6
  - Area bus width can be selected by register (area 0 is set by external pin.)
  - Wait states can be inserted using the $\overline{\text{WAIT}}$ pin.
  - Wait state insertion can be controlled through software. Register settings can be used to specify the insertion of 1–10 cycles independently for each area.
  - The type of memory connected can be specified for each area, and control signals are output for direct memory connection.
  - Wait cycles are automatically inserted to avoid data bus conflict for continuous memory accesses to different areas or writes directly following reads of the same area.

- Direct interface to synchronous DRAM
  - Multiplexes row/column addresses according to synchronous DRAM capacity
  - Supports burst operation
  - Supports bank active mode (only 32-bit access)
  - Has both auto-refresh and self-refresh functions.
  - Controls timing of synchronous DRAM direct-connection control signals according to register setting.

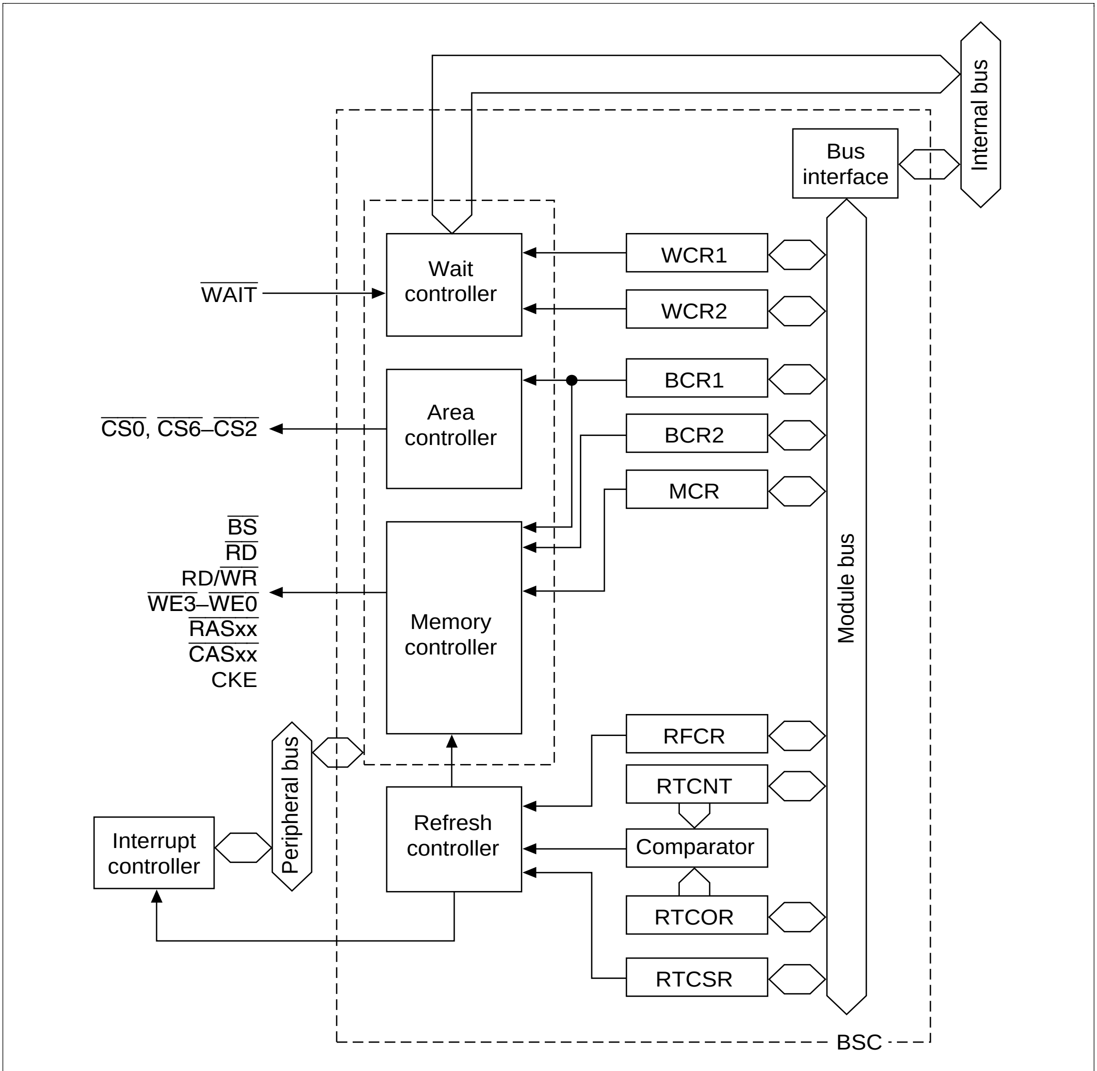
- Burst ROM interface
  - Insertion of wait states controllable through software
  - Register setting control of burst transfers

- Short refresh cycle control
  - The overflow interrupt function of the refresh counter enables the refresh function immediately after the self-refresh operation using the low power-consumption DRAM.

- The refresh counter can be used as an interval timer.
  - Outputs an interrupt request signal using the compare-matching function
  - Outputs an interrupt request signal when the refresh counter overflows
- Automatically disables the output of clock signals to anywhere but the refresh counter, except during execution of external bus cycles.

13.1.2 Block Diagram

Figure 13.1 shows the functional block diagram of the bus state controller.



Legend

WCR: Wait state control register

BCR: Bus control register

MCR: Memory control register

RFCR: Refresh count register

RTCNT: Refresh timer count register

RTCOR: Refresh time constant register

RTCSR: Refresh timer control/status register

Figure 13.1 BSC Functional Block Diagram

13.1.3 Pin Configuration

Table 13.1 shows the BSC pin configuration.

Table 13.1 BSC Pins

Pin Name	Signal	I/O	Description
Address bus	A25–A0	O	Address output
Data bus	D15–D0	I/O	Data I/O
	D31–D16	I/O	Data I/O when using 32-bit bus width
Bus cycle start	$\overline{\text{BS}}$	O	Shows start of bus cycle. During burst transfers, asserted every data cycle
Chip select 0, 2–6	$\overline{\text{CS0}}$, $\overline{\text{CS2}}$–$\overline{\text{CS6}}$	O	Chip select signals to indicate area being accessed
Read/write	RD/$\overline{\text{WR}}$	O	Data bus direction indication signal. synchronous DRAM write indication signal
Row address strobe 3L	$\overline{\text{RAS3L}}$	O	When synchronous DRAM is used, RAS3L for lower 32-Mbyte address
Row address strobe 3U	$\overline{\text{RAS3U}}$	O	When synchronous DRAM is used, RAS3U for upper 32-Mbyte address
Column address strobe	$\overline{\text{CASL}}$	O	When synchronous DRAM is used, CASL signal for lower 32-Mbyte address
Column address strobe LH	$\overline{\text{CASU}}$	O	When synchronous DRAM is used, CASU signal for upper 32-Mbyte address
Data enable 0	$\overline{\text{WE0}}$/DQMLL	O	When memory other than synchronous DRAM is used, D7–D0 write strobe signal. When synchronous DRAM is used, selects D7–D0
Data enable 1	$\overline{\text{WE1}}$/DQMLU	O	When memory other than synchronous DRAM is used, D15–D8 write strobe signal. When synchronous DRAM is used, selects D15–D8
Data enable 2	$\overline{\text{WE2}}$/DQMUL	O	When memory other than synchronous DRAM is used, D23–D16 write strobe signal. When synchronous DRAM is used, selects D23–D16
Data enable 3	$\overline{\text{WE3}}$/DQMUU	O	When memory other than synchronous DRAM is used, D31–D24 write strobe signal. When synchronous DRAM is used, selects D31–D24
Read	$\overline{\text{RD}}$	O	Strobe signal indicating read cycle
Wait	$\overline{\text{WAIT}}$	I	Wait state request signal
Clock enable	CKE	O	Clock enable control signal for synchronous DRAM
Bus release request	$\overline{\text{BREQ}}$	I	Bus release request signal
Bus release acknowledgment	$\overline{\text{BACK}}$	O	Bus release acknowledge signal

13.1.4 Register Configuration

The BSC has 10 registers (table 13.2). The synchronous DRAM also has a built-in synchronous DRAM mode register. These registers control direct connection interfaces to memory, wait states, and refreshes.

Table 13.2 Register Configuration

Name	Abbr.	R/W	Initial Value*	Address	Bus Width
Bus control register 1	BCR1	R/W	H'0000	H'FFFFFF60	16
Bus control register 2	BCR2	R/W	H'3FF0	H'FFFFFF62	16
Wait state control register 1	WCR1	R/W	H'3FF3	H'FFFFFF64	16
Wait state control register 2	WCR2	R/W	H'FFFF	H'FFFFFF66	16
Individual memory control register	MCR	R/W	H'0000	H'FFFFFF68	16
Refresh timer control/status register	RTCSR	R/W	H'0000	H'FFFFFF6E	16
Refresh timer counter	RTCNT	R/W	H'0000	H'FFFFFF70	16
Refresh time constant register	RTCOR	R/W	H'0000	H'FFFFFF72	16
Refresh count register	RFCR	R/W	H'0000	H'FFFFFF74	16
Synchronous DRAM mode register, area 2	SDMR	W	—	H'FFFFD000– H'FFFFDFFF	8
Synchronous DRAM mode register, area 3				H'FFFFE000– H'FFFFEFFF	

Note: * Initialized by power-on resets.

13.1.5 Area Overview

Space Allocation: In the architecture of this LSI, both logical spaces and physical spaces have 32-bit address spaces. The cache access method is shown by the upper 3 bits. For details see section 6, Cache. The remaining 29 bits are used for division of the space into eight areas. The BSC performs control for this 29-bit space.

As listed in table 13.3, this LSI can be connected directly to six areas of memory, and it outputs chip select signals ($\overline{\text{CS0}}$, $\overline{\text{CS2}}$–$\overline{\text{CS6}}$) for each of them. $\overline{\text{CS0}}$ is asserted during area 0 access; $\overline{\text{CS6}}$ is asserted during area 6 access. When PCMCIA interface is selected in area 2 or 3, in addition to $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and DQM are asserted for the corresponding bytes accessed.

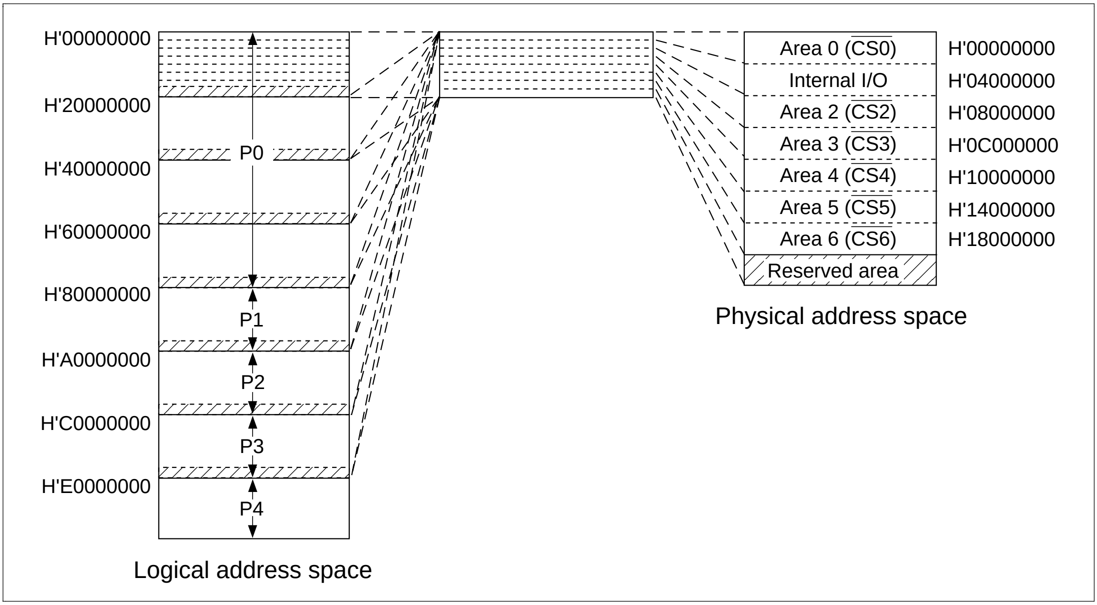


Figure 13.2 Address Space

Table 13.3 Physical Address Space Map

Area	Connectable Memory	Physical Address	Capacity	Access Size
0	Ordinary memory*1, burst ROM	H'00000000 to H'03FFFFFF	64 Mbytes	8, 16, 32*2
		H'00000000 + H'20000000 × n to H'03FFFFFF + H'20000000 × n	Shadow	n: 1–6
1	Internal I/O registers*6	H'04000000 to H'07FFFFFF	64 Mbytes	8, 16, 32*3
		H'04000000 + H'20000000 × n to H'07FFFFFF + H'20000000 × n	Shadow	n: 1–6
2	Ordinary memory*1 Synchronous DRAM	H'08000000 to H'0BFFFFFF	64 Mbytes	8, 16, 32*3, *4
		H'08000000 + H'20000000 × n to H'0BFFFFFF + H'20000000 × n	Shadow	n: 1–6
3	Ordinary memory Synchronous DRAM	H'0C000000 to H'0FFFFFFF	64 Mbytes	8, 16, 32*3, *5
		H'0C000000 + H'20000000 × n to H'0FFFFFFF + H'20000000 × n	Shadow	n: 1–6
4	Ordinary memory	H'10000000 to H'13FFFFFF	64 Mbytes	8, 16, 32*3
		H'10000000 + H'20000000 × n to H'13FFFFFF + H'20000000 × n	Shadow	n: 1–6
5	Ordinary memory Burst ROM	H'14000000 to H'15FFFFFF	32 Mbytes	8, 16, 32*3
		H'16000000 to H'17FFFFFF	32 Mbytes	
		H'14000000 + H'20000000 × n to H'17FFFFFF + H'20000000 × n	Shadow	n: 1–6
6	Ordinary memory Burst ROM	H'18000000 to H'19FFFFFF	32 Mbytes	8, 16, 32*3
		H'1A000000 to H'1BFFFFFF	32 Mbytes	
		H'18000000 + H'20000000 × n to H'1BFFFFFF + H'20000000 × n	Shadow	n: 1–6
7	Reserved area*7	H'1C000000 + H'20000000 × n to H'1FFFFFFF + H'20000000 × n		n: 0–7

Notes: *1 Memory with interface such as SRAM or ROM

*2 Use external pin to specify memory bus width.

*3 Use register to specify memory bus width.

*4 With synchronous DRAM interfaces, bus width must be 16 or 32 bits.

*5 With synchronous DRAM interfaces, bus width must be 16 or 32 bits.

*6 When the control register in area 1 sets the top 3 bits of the logical address to 101 to allocate in the P2 space.

*7 Do not access the reserved area. If the reserved area is accessed, the correct operation cannot be guaranteed.

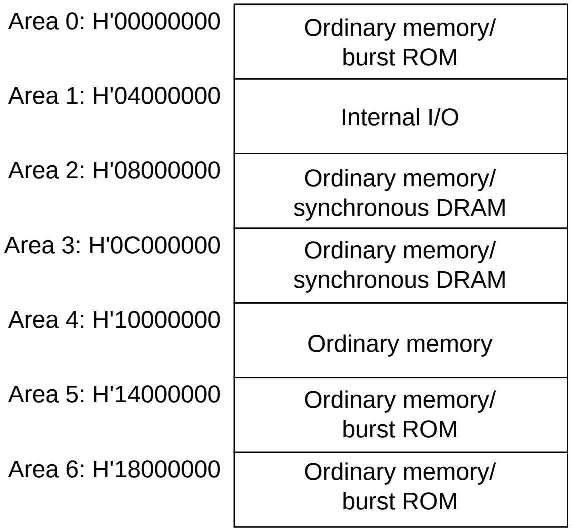


Figure 13.3 Physical Space Allocation

Memory Bus Width: The memory bus width in this LSI can be set for each area. In area 0, an external pin can be used to select byte (8 bits), word (16 bits), or longword (32 bits) on power-on reset. The correspondence between the external pins (MD4 and MD3) and memory size is listed in table below.

Table 13.4 Correspondence between External Pins (MD4 and MD3) and Memory Size

MD4	MD3	Memory Size
0	0	Reserved (Do not set)
0	1	8 bits
1	0	16 bits
1	1	32 bits

For areas 2–6, byte, word, and longword may be chosen for the bus width using bus control register 2 (BCR2) whenever ordinary memory, ROM, or burst ROM are used. When the DRAM or synchronous DRAM interface is used, word or longword can be chosen as the bus width.

When port A or B is used, set the bus width of all areas to 8-bit or 16-bit.

For details see section 13.2.2, Bus Control Register 2 (BCR2), and section 13.2.5, Individual Memory Control Register (MCR).

Shadow Space: Areas 0, 2–6 are decoded by physical addresses A28–A26, which correspond to areas 000 to 110. Address bits 31–29 are ignored. This means that the range of area 0 addresses, for example, is H'00000000 to H'03FFFFFF, and its corresponding shadow space is the address space obtained by adding to it H'20000000 × n (n = 1 to 6). The address range for area 7, which is on-chip I/O space, is H'1C000000 to H'1FFFFFFF. The address space H'1C000000 + H'20000000 × n–H'1FFFFFFF + H'20000000 × n (n = 0 to 7) corresponding to the area 7 shadow space is reserved, so do not use it.

13.2 BSC Registers

13.2.1 Bus Control Register 1 (BCR1)

Bus control register 1 (BCR1) is a 16-bit read/write register that sets the functions and bus cycle state for each area. It is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or by standby mode. Do not access external memory outside area 0 until BCR1 register initialization is complete.

Bit:	15	14	13	12	11	10	9	8
	—	—	HIZMEM	HIZCNT	—	A0BST1	A0BST0	A5BST1
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	A5BST0	A6BST1	A6BST0	—	DRAM TP1	DRAM TP0	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R	R	R/W	R	R

Bits 15, 14, 11, 4, 1, and 0—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 13—Hi-Z Memory Control (HIZMEM): Specifies the state of A25–0, $\overline{\text{BS}}$, $\overline{\text{CS}}$, RD/$\overline{\text{WR}}$, $\overline{\text{WE}}$/DQM, $\overline{\text{RD}}$, and DRAK0/1 in standby mode.

Bit 13: HIZMEM	Description	
0	Hi-Z in standby mode	(Initial value)
1	Driven in standby mode	

Bit 12—High-Z Control (HIZCNT): Specifies the state of the CKIO, CKE, $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ signals at standby and bus right release.

Bit 12: HIZCNT	Description
0	The signals are high-impedance state (High-Z) at standby and bus right release (Initial value)
1	The signals are driven at standby and bus right release

Bits 10 and 9—Area 0 Burst ROM Control (A0BST1, A0BST0): Specify whether to use burst ROM in physical space area 0. When burst ROM is used, set the number of burst transfers.

Bit 10: A0BST1	Bit 9: A0BST0	Description
0	0	Access area 0 as ordinary memory (Initial value)
	1	Access area 0 as burst ROM (4 consecutive accesses). Can be used when bus width is 8, 16, or 32
1	0	Access area 0 as burst ROM (8 consecutive accesses). Can be used when bus width is 8 or 16
	1	Access area 0 as burst ROM (16 consecutive accesses). Can be used only when bus width is 8

Bits 8 and 7—Area 5 Burst Enable (A5BST1, A5BST0): Specify whether to use burst ROM burst mode in physical space area 5. When burst ROM burst mode is used, these bits set the number of burst transfers.

Bit 8: A5BST1	Bit 7: A5BST0	Description
0	0	Access area 5 as ordinary memory (Initial value)
	1	Burst access of area 5 (4 consecutive accesses). Can be used when bus width is 8, 16, or 32
1	0	Burst access of area 5 (8 consecutive accesses). Can be used when bus width is 8 or 16
	1	Burst access of area 5 (16 consecutive accesses). Can be used only when bus width is 8

Bits 6 and 5—Area 6 Burst Enable (A6BST1, A6BST0): Specify whether to use burst ROM burst mode in physical space area 6. When burst ROM burst mode is used, these bits set the number of burst transfers.

Bit 6: A6BST1	Bit 5: A6BST0	Description
0	0	Access area 6 as ordinary memory (Initial value)
	1	Burst access of area 6 (4 consecutive accesses). Can be used when bus width is 8, 16, or 32
1	0	Burst access of area 6 (8 consecutive accesses). Can be used when bus width is 8 or 16
	1	Burst access of area 6 (16 consecutive accesses). Can be used only when bus width is 8

Bits 3 and 2—Area 2, Area 3 Memory Type (DRAMTP1, DRAMTP0): Designate the types of memory connected to physical space areas 2 and 3. Ordinary memory, such as ROM, SRAM, or flash ROM, can be directly connected. DRAM, and synchronous DRAM can also be directly connected.

Bit 3: DRAMTP1	Bit 2: DRAMTP0	Description
0	0	Areas 2 and 3 are ordinary memory (Initial value)
	1	Reserved (Setting disabled)
1	0	Area 2: ordinary memory; area 3: synchronous DRAM
	1	Areas 2 and 3 are synchronous DRAM*

Note: * When selecting this mode, set the same bus width for area 2 and area 3.

13.2.2 Bus Control Register 2 (BCR2)

The bus control register 2 (BCR2) is a 16-bit read/write register that selects the bus-size width of each area. It is initialized to H'3FF0 by a power-on reset, but is not initialized by a manual reset or by standby mode. Do not access external memory outside area 0 until BCR2 register initialization is complete.

Bit:	15	14	13	12	11	10	9	8
	—	—	A6SZ1	A6SZ0	A5SZ1	A5SZ0	A4SZ1	A4SZ0
Initial value:	0	0	1	1	1	1	1	1
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	A3SZ1	A3SZ0	A2SZ1	A2SZ0	—	—	—	—
Initial value:	1	1	1	1	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R	R	R	R

Bits 15, 14, 3, 2, 1, and 0—Reserved: These bits are always read as 0. The write value should always be 0.

Bits 2n + 1, 2n—Area n (2–6) Bus Size Specification (AnSZ1, AnSZ0): Specify the bus sizes of physical space area n (n = 2 to 6).

Bit 2n + 1: AnSZ1	Bit 2n: AnSZ0	Port A / B	Description
0	0	Unused	Reserved (Setting disabled)
	1		Byte (8-bit) size
1	0		Word (16-bit) size
	1		Longword (32-bit) size
0	0	Used	Reserved (Setting disabled)
	1		Byte (8-bit) size
1	0		Word (16-bit) size
	1		Reserved (Setting disabled)

13.2.3 Wait Control Register 1 (WCR1)

Wait control register 1 (WCR1) is a 16-bit read/write register that specifies the number of idle (wait) state cycles inserted for each area. For some memories, the drive of the data bus may not be turned off quickly even when the read signal from the external device is turned off. This can result in conflicts between data buses when consecutive memory accesses are to different memories or when a write immediately follows a memory read. This LSI automatically inserts idle states equal to the number set in WCR1 in those cases.

WCR1 is initialized to H'3FF3 by a power-on reset. It is not initialized by a manual reset or by standby mode.

Bit:	15	14	13	12	11	10	9	8
	WAITSEL	—	A6IW1	A6IW0	A5IW1	A5IW0	A4IW1	A4IW0
Initial value:	0	0	1	1	1	1	1	1
R/W:	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	A3IW1	A3IW0	A2IW1	A2IW0	—	—	A0IW1	A0IW0
Initial value:	1	1	1	1	0	0	1	1
R/W:	R/W	R/W	R/W	R/W	R	R	R/W	R/W

Bit 15—WAIT Sampling Timing Select (WAITSEL): Specifies the WAIT signal sampling timing.

Bit 15: WAITSEL	Description
0	Sampled at rise of CKIO. In this case, the WAIT signal must be input synchronously (Initial value)
1	Sampled at fall of CKIO. In this case, the WAIT signal can be input asynchronously

Bits 14, 3, and 2 —Reserved: These bits are always read as 0. The write value should always be 0.

Bits 2n + 1, 2n—Area n (6–2, 0) Intercycle Idle Specification (AnIW1, AnIW0): Specify the number of idles inserted between bus cycles when switching between physical space area n (6–2, 0) to another space or between a read access to a write access in the same physical space.

Bit 2n + 1: AnIW1	Bit 2n: AnIW0	Description
0	0	1 idle cycle inserted
	1	1 idle cycle inserted
1	0	2 idle cycles inserted
	1	3 idle cycles inserted (Initial value)

13.2.4 Wait Control Register 2 (WCR2)

Wait control register 2 (WCR2) is a 16-bit read/write register that specifies the number of wait state cycles inserted for each area. It also specifies the pitch of data access for burst memory accesses. This allows direct connection of even low-speed memories without an external circuit. WCR2 is initialized to H'FFFF by a power-on reset. It is not initialized by a manual reset or by standby mode.

Bit:	15	14	13	12	11	10	9	8
	A6 W2	A6 W1	A6 W0	A5 W2	A5 W1	A5 W0	A4 W2	A4 W1
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	A4 W0	A3 W1	A3 W0	A2 W1	A2 W0	A0 W2	A0 W1	A0 W0
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 to 13—Area 6 Wait Control (A6W2, A6W1, A6W0): Specify the number of wait states inserted into physical space area 6. Also specify the burst pitch for burst transfer.

			Description			
			First Cycle		Burst Cycle (Excluding First Cycle)	
Bit 15: A6W2	Bit 14: A6W1	Bit 13: A6W0	Inserted Wait States	$\overline{\text{WAIT}}$ Pin	Number of States Per Data Transfer	$\overline{\text{WAIT}}$ Pin
0	0	0	0	Disable	2	Enable
		1	1	Enable	2	Enable
	1	0	2	Enable	3	Enable
		1	3	Enable	4	Enable
1	0	0	4	Enable	4	Enable
		1	6	Enable	6	Enable
	1	0	8	Enable	8	Enable
		1	10 (Initial value)	Enable	10	Enable

Bits 12 to 10—Area 5 Wait Control (A5W2, A5W1, A5W0): Specify the number of wait states inserted into physical space area 5. Also specify the burst pitch for burst transfer.

			Description			
			First Cycle		Burst Cycle (Excluding First Cycle)	
Bit 12: A5W2	Bit 11: A5W1	Bit 10: A5W0	Inserted Wait States	$\overline{\text{WAIT}}$ Pin	Number of States Per Data Transfer	$\overline{\text{WAIT}}$ Pin
0	0	0	0	Disable	2	Enable
		1	1	Enable	2	Enable
	1	0	2	Enable	3	Enable
		1	3	Enable	4	Enable
1	0	0	4	Enable	4	Enable
		1	6	Enable	6	Enable
	1	0	8	Enable	8	Enable
		1	10 (Initial value)	Enable	10	Enable

Bits 9 to 7—Area 4 Wait Control (A4W2, A4W1, A4W0): Specify the number of wait states inserted into physical space area 4.

			Description	
Bit 9: A4W2	**Bit 8: A4W1**	**Bit 7: A4W0**	**Inserted Wait State**	**$\overline{\text{WAIT}}$ Pin**
0	0	0	0	Ignored
		1	1	Enable
	1	0	2	Enable
		1	3	Enable
1	0	0	4	Enable
		1	6	Enable
	1	0	8	Enable
		1	10	Enable (Initial value)

Bits 6 and 5—Area 3 Wait Control (A3W1, A3W0): Specify the number of wait states inserted into physical space area 3.

- For Ordinary memory

		Description	
Bit 6: A3W1	**Bit 5: A3W0**	**Inserted Wait States**	**$\overline{\text{WAIT}}$ Pin**
0	0	0	Ignored
	1	1	Enable
1	0	2	Enable
	1	3	Enable (Initial value)

- For Synchronous DRAM

		Description
Bit 6: A3W1	**Bit 5: A3W0**	**CAS Latency**
0	0	1
	1	1
1	0	2
	1	3 (Initial value)

Bits 4 and 3—Area 2 Wait Control (A2W1, A2W0): Specify the number of wait states inserted into physical space area 2.

- For Ordinary memory

		Description		
Bit 4: A2W0	**Bit 3: A2W0**	**Inserted Wait States**	**$\overline{\text{WAIT}}$ Pin**	
0	0	0	Ignored	
	1	1	Enable	
1	0	2	Enable	
	1	3	Enable	(Initial value)

- For Synchronous DRAM

		Description	
Bit 4: A2W1	**Bit 3: A2W0**	**CAS Latency**	
0	0	1	
	1	1	
1	0	2	
	1	3	(Initial value)

Bits 2 to 0—Area 0 Wait Control (A0W2, A0W1, A0W0): Specify the number of wait states inserted into physical space area 0. Also specify the burst pitch for burst transfer.

			Description			
			First Cycle		**Burst Cycle (Excluding First Cycle)**	
Bit 2: A0W2	**Bit 1: A0W1**	**Bit 0: A0W0**	**Inserted Wait States**	**$\overline{\text{WAIT}}$ Pin**	**Number of States Per Data Transfer**	**$\overline{\text{WAIT}}$ Pin**
0	0	0	0	Ignored	2	Enable
		1	1	Enable	2	Enable
	1	0	2	Enable	3	Enable
		1	3	Enable	4	Enable
1	0	0	4	Enable	4	Enable
		1	6	Enable	6	Enable
	1	0	8	Enable	8	Enable
		1	10 (Initial value)	Enable	10	Enable

13.2.5 Individual Memory Control Register (MCR)

The individual memory control register (MCR) is a 16-bit read/write register that specifies $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ timing and burst control for DRAM (area 3 only), synchronous DRAM (areas 2 and 3), specifies address multiplexing, and controls refresh. This enables direct connection of DRAM, and synchronous DRAM without external circuits.

The MCR is initialized to H'0000 by power-on resets, but is not initialized by manual resets or standby mode. The bits TPC1–TPC0, RCD1–RCD0, TRWL1–TRWL0, TRAS1–TRAS0, RASD, and AMX2–AMX0 are written to at the initialization after a power-on reset and are not then modified again. When RFSH and RMODE are written to, write the same values to the other bits. When using synchronous DRAM, do not access areas 2 and 3 until this register is initialized.

Bit:	15	14	13	12	11	10	9	8
	TPC1	TPC0	RCD1	RCD0	TRWL1	TRWL0	TRAS1	TRAS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	RASD	—	AMX2	AMX1	AMX0	RFSH	RMODE	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R/W	R/W	R/W	R/W	R/W	R

Bits 15 and 14—RAS Precharge Time (TPC1, TPC0): When synchronous DRAM interface is selected, the TPC bits set the minimum number of cycles until output of the next bank-active command after precharge.

The number of cycles inserted immediately after issuing the Precharge All Banks command (PALL) in auto-refreshing or Precharge command (PRE) in bank active mode is one less than the normal value. In bank active mode, do not set TPC1 = 0 and TPC0 = 0.

		Description		
Bit 15: TPC1	**Bit 14: TPC0**	**Normal Operation**	**Immediately after Precharge Command***	**Immediately after Self-Refresh**
0	0	1 cycle (Initial value)	0 cycle (Initial value)	2 cycles
	1	2 cycles	1 cycle	5 cycles
1	0	3 cycles	2 cycles	8 cycles
	1	4 cycles	3 cycles	11 cycles

Note: * Immediately after Precharge All Banks (PALL) command in auto-refreshing or Precharge (PRE) command in bank active mode

Bits 13 and 12—RAS–CAS Delay (RCD1, RCD0): The RCD bits set the bank active read/write command delay time.

Bit 13: RCD1	Bit 12: RCD0	Description	
0	0	1 cycle	(Initial value)
	1	2 cycles	
1	0	3 cycles	
	1	4 cycles	

Bits 11 and 10—Write-Precharge Delay (TRWL1, TRWL0): The TRWL bits set the synchronous DRAM write-precharge delay time. This designates the time between the end of a write cycle and the next bank-active command. After the write cycle, the next bank-active command is not issued for the period TPC + TRWL.

Bit 11: TRWL1	Bit 10: TRWL0	Description	
0	0	1 cycle	(Initial value)
	1	2 cycles	
1	0	3 cycles	
	1	Reserved (Setting disabled)	

Bits 9 and 8—$\overline{\text{CAS}}$-Before-$\overline{\text{RAS}}$ Refresh $\overline{\text{RAS}}$ Assert Time (TRAS1, TRAS0): No bank-active command is issued during the period TPC + TRAS after an auto-refresh command is issued.

Bit 9: TRAS1	Bit 8: TRAS0	Description
0	0	2 cycles (Initial value)
	1	3 cycles
1	0	4 cycles
	1	5 cycles

Bit 7—Synchronous DRAM Bank Active (RASD): Specifies whether synchronous DRAM is used in bank active mode or auto-precharge mode. Set auto-precharge mode when areas 2 and 3 are both designated as synchronous DRAM space. However, do not set bank-active mode when the synchronous DRAM is used with a 16-bit width, as operation cannot be guaranteed in this case.

Bit 7: RASD	Description
0	Auto-precharge mode (Initial value)
1	Bank active mode

Bits 6 and 0—Reserved: These bits are always read as 0. The write value should always be 0.

Bits 5 to 3—Address Multiplex (AMX2, AMX1, AMX0): The AMX bits specify address multiplexing for synchronous DRAM.

- For Synchronous DRAM Interface

Bit5: AMX2	Bit 4: AMX1	Bit 3: AMX0	Description
1	0	0	The row address begins with A9 (The A9 value is output at A1 when the row address is output. 4 M × 16-bit products) (Initial value)
		1	The row address begins with A10 (The A10 value is output at A1 when the row address is output. 8 M × 8-bit products)
	1	0	Reserved
		1	The row address begins with A9 (The A9 value is output at A1 when the row address is output. 2 M × 32-bit products)
0	0	0	The row address begins with A9 (The A9 value is output at A1 when the row address is output. 1 M × 16-bit products) (Initial value)
		1	The row address begins with A10 (The A10 value is output at A1 when the row address is output. 2 M × 8-bit products)
	1	0	The row address begins with A11 (The A11 value is output at A1 when the row address is output. 4 M × 4-bit products)
		1	The row address begins with A9 (The A9 value is output at A1 when the row address is output. 256 K × 16-bit products)

Bit 2—Refresh Control (RFSH): The RFSH bit determines whether or not the refresh operation of the synchronous DRAM is performed. The timer for generation of the refresh request frequency can also be used as an interval timer.

Bit 2: RFSH	Description
0	No refresh (Initial value)
1	Refresh

Bit 1—Refresh Mode (RMODE): The RMODE bit selects whether to perform an ordinary refresh or a self-refresh when the RFSH bit is 1. When the RFSH bit is 1 and this bit is 0, a CAS-before-RAS refresh or an auto-refresh is performed on synchronous DRAM at the period set by the refresh-related registers RTCNT, RTCOR and RTCSR. When a refresh request occurs during an external bus cycle, the bus cycle will be ended and the refresh cycle performed. When the RFSH bit is 1 and this bit is also 1, the synchronous DRAM will wait for the end of any executing external bus cycle before going into a self-refresh. All refresh requests to memory that is in the self-refresh state are ignored.

When changing the refresh mode, the refresh control bit (RFSH) must first be cleared to 0 so that refreshing is not performed. Then set the refresh bit to 1 and change the refresh mode.

Bit 1: RMODE	Description	
0	CAS-before-RAS refresh (RFSH must be 1)	(Initial value)
1	Self-refresh (RFSH must be 1)	

13.2.6 Synchronous DRAM Mode Register (SDMR)

The synchronous DRAM mode register (SDMR) is written to via the synchronous DRAM address bus and is an 8-bit write-only register. It sets synchronous DRAM mode for areas 2 and 3. SDMR is undefined after a power-on reset. The register contents are not initialized by a manual reset or standby mode; values remain unchanged.

Writes to the synchronous DRAM mode register use the address bus rather than the data bus. If the value to be set is X and the SDMR address is Y, the value X is written in the synchronous DRAM mode register by writing in address X + Y. Since, with a 32-bit bus width, A0 of the synchronous DRAM is connected to A2 of the chip and A1 of the synchronous DRAM is connected to A3 of the chip, the value actually written to the synchronous DRAM is the X value shifted 2 bits right. With a 16-bit bus width, the value written is the X value shifted 1 bit right. For example, with a 32-bit bus width, when H'0230 is written to the SDMR register of area 2, random data is written to the address H'FFFFD000 (address Y) + H'08C0 (value X), or H'FFFFD8C0. As a result, H'0230 is written to the SDMR register. The range for value X is H’0000 to H’0FFC. When H'0230 is written to the SDMR register of area 3, random data is written to the address H'FFFFE000 (address Y) + H'08C0 (value X), or H'FFFFE8C0. As a result, H'0230 is written to the SDMR register. The range for value X is H'0000 to H'0FFC.

Bit:	31		12	11	10	9	8
	SDMR address						
Initial value:	—	....	—	—	—	—	—
R/W:	—	....	—	W*	W*	W	W

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

Note: * Depends on the type of synchronous DRAM.

13.2.7 Refresh Timer Control/Status Register (RTCSR)

The refresh timer control/status register (RTSCR) is a 16-bit read/write register that specifies the refresh cycle, whether to generate an interrupt, and that interrupt's cycle. It is initialized to H'0000 by a power-on reset, but is not initialized by a manual reset or standby mode. RTCOR settings must be made before setting CSK2 to CSK0 in RTCSR.

Note: Writing to the RTCSR differs from that to general registers to ensure the RTCSR is not rewritten incorrectly. Use the word-transfer instruction to set the upper byte as B'10100101 and the lower byte as the write data. For details, see section 13.2.11, Cautions on Accessing Refresh Control Related Registers.

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

Bit:	7	6	5	4	3	2	1	0
	CMF	CMIE	CKS2	CKS1	CKS0	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.

Bit 7—Compare Match Flag (CMF): The CMF status flag indicates that the values of RTCNT and RTCOR match.

Bit 7: CMF	Description
0	The values of RTCNT and RTCOR do not match Clear condition: When a refresh is performed After 0 has been written in CMF and RFSH = 1 and RMODE = 0 (to perform a CBR refresh) (Initial value)
1	The values of RTCNT and RTCOR match Set condition: RTCNT = RTCOR *

Note: * Contents do not change when 1 is written to CMF.

Bit 6—Compare Match Interrupt Enable (CMIE): Enables or disables an interrupt request caused when the CMF of RTCSR is set to 1. Do not set this bit to 1 when using CAS-before-RAS refresh or auto-refresh.

Bit 6: CMIE	Description	
0	Disables an interrupt request caused by CMF	(Initial value)
1	Enables an interrupt request caused by CMF	

Bits 5 to 3—Clock Select Bits (CKS2–CKS0): Select the clock input to RTCNT. The source clock is the external bus clock (BCLK). The RTCNT count clock is CKIO divided by the specified ratio.

Bit 5: CKS2	Bit 4: CKS1	Bit 3: CKS0	Description
0	0	0	Disables clock input
		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): The OVF status flag indicates when the number of refresh requests indicated in the refresh count register (RFCR) exceeds the limit set in the LMTS bit of RTCSR.

Bit 2: OVF	Description	
0	RFCR has not exceeded the count limit value set in LMTS Clear Conditions: When 0 is written to OVF	(Initial value)
1	RFCR has exceeded the count limit value set in LMTS Set Conditions: When the RFCR value has exceeded the count limit value set in LMTS*	

Note: * Contents don’t change when 1 is written to OVF.

Bit 1—Refresh Count Overflow Interrupt Enable (OVIE): OVIE selects whether to suppress generation of interrupt requests by OVF when the OVF bit of RTCSR is set to 1.

Bit 1: OVIE	Description	
0	Disables interrupt requests from the OVF	(Initial value)
1	Enables interrupt requests from the OVF	

Bit 0—Refresh Count Overflow Limit Select (LMTS): Indicates the count limit value to be compared to the number of refreshes indicated in the refresh count register (RFCR). When the value RFCR overflows the value specified by LMTS, the OVF flag is set.

Bit 0: LMTS	Description	
0	Count limit value is 1024	(Initial value)
1	Count limit value is 512	

13.2.8 Refresh Timer Counter (RTCNT)

RTCNT is a 16-bit read/write register. RTCNT is an 8-bit counter that counts up with input clocks. The clock select bits (CKS2–CKS0) of RTCSR select the input clock. When RTCNT matches RTCOR, the CMF bit of RTCSR is set and RTCNT is cleared. RTCNT is initialized to H'00 by a power-on reset; it continues incrementing after a manual reset; it is not initialized by standby mode and holds its values unchanged.

Note: Writing to the RTCNT differs from that to general registers to ensure the RTCNT is not rewritten incorrectly. Use the word-transfer instruction to set the upper byte as B'10100101 and the lower byte as the write data. For details, see section 13.2.11, Cautions on Accessing Refresh Control Related Registers.

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

Bit:	7	6	5	4	3	2	1	0
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

13.2.9 Refresh Time Constant Register (RTCOR)

The refresh time constant register (RTCOR) is a 16-bit read/write register. The values of RTCOR and RTCNT (bottom 8 bits) are constantly compared. When the values match, the compare match flag (CMF) of RTCSR is set and RTCNT is cleared to 0. When the refresh bit (RFSH) of the individual memory control register (MCR) is set to 1 and the refresh mode is set to CAS-before-RAS refresh, a memory refresh cycle occurs when the CMF bit is set. RTCOR is initialized to H'00 by a power-on reset. It is not initialized by a manual reset or standby mode, but holds its contents. Make the RTCOR setting before setting bits CKS2 to CKS0 in RTCSR.

Note: Writing to the RTCOR differs from that to general registers to ensure the RTCOR is not rewritten incorrectly. Use the word-transfer instruction to set the upper byte as B'10100101 and the lower byte as the write data. For details, see section 13.2.11, Cautions on Accessing Refresh Control Related Registers.

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

Bit:	7	6	5	4	3	2	1	0
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

13.2.10 Refresh Count Register (RFCR)

The refresh count register (RFCR) is a 16-bit read/write register. It is a 10-bit counter that increments every time RTCOR and RTCNT match. When RFCR exceeds the count limit value set in the LMTS of RTCSR, RTCSR’s OVF bit is set and RFCR clears. RFCR is initialized to H'0000 when a power-on reset is performed. It is not initialized by a manual reset or standby mode, but holds its contents.

Note: Writing to the RFCR differs from that to general registers to ensure the RFCR is not rewritten incorrectly. Use the word-transfer instruction to set the MSB and followed 6 bits of upper bytes as B'101001 and remaining bits as the write data. For details, see section 13.2.11, Cautions on Accessing Refresh Control Related Registers.

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

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

13.2.11 Cautions on Accessing Refresh Control Related Registers

RFCR, RTCSR, RTCNT, and RTCOR require that a specific code be appended to the data when it is written to prevent data from being mistakenly overwritten by program overruns or other write operations (figure 13.4). Perform reads and writes using the following methods:

1. When writing to RFCR, RTCSR, RTCNT, and RTCOR, use only word transfer instructions. Not to write with byte transfer instructions.

   When writing to RTCNT, RTCSR, or RTCOR, place B'10100101 in the upper byte and the write data in the lower byte. When writing to RFCR, place B'101001 in the top 6 bits and the write data in the remaining bits, as shown in figure 13.4.

	15							8	7 0
RTCSR, RTCNT, RTCOR	1	0	1	0	0	1	0	1	Write data

	15					10	9 0
RFCR	1	0	1	0	0	1	Write data

Figure 13.4 Writing to RFCR, RTCSR, RTCNT, and RTCOR

2. When reading from RFCR, RTCSR, RTCNT, and RTCOR, carry out reads with 16-bit width. 0 is read out from undefined bit sections.

13.3 BSC Operation

13.3.1 Access Size and Data Alignment

This LSI supports big endian, in which the 0 address is the most significant byte in the byte data.

Three data bus widths are available for ordinary memory (byte, word, or longword). Word and longword are available for synchronous DRAM. Data alignment is performed in accordance with the data bus width of the device. This also means that when longword data is read from a byte-width device, the read operation must happen 4 times. In this LSI, data alignment and conversion of data length is performed automatically between the respective interfaces.

Tables 13.5 through 13.7 show the relationship between endian, device data width, and access unit.

Table 13.5 32-Bit External Device/Big Endian Access and Data Alignment

	Data Bus				Strobe Signals			
Operation	**D31–D24**	**D23–D16**	**D15–D8**	**D7–D0**	**$\overline{\text{WE3}}$, DQMUU**	**$\overline{\text{WE2}}$, DQMUL**	**$\overline{\text{WE1}}$, DQMLU**	**$\overline{\text{WE0}}$, DQMLL**
Byte access at 0	Data 7–0	—	—	—	Assert			
Byte access at 1	—	Data 7–0	—	—		Assert		
Byte access at 2	—	—	Data 7–0	—			Assert	
Byte access at 3	—	—	—	Data 7–0				Assert
Word access at 0	Data 15–8	Data 7–0	—	—	Assert	Assert		
Word access at 2	—	—	Data 15–8	Data 7–0			Assert	Assert
Longword access at 0	Data 31–24	Data 23–16	Data 15–8	Data 7–0	Assert	Assert	Assert	Assert

Table 13.6 16-Bit External Device/Big Endian Access and Data Alignment

		Data Bus				Strobe Signals			
Operation		**D31–D24**	**D23–D16**	**D15–D8**	**D7–D0**	**$\overline{\text{WE3}}$, DQMUU**	**$\overline{\text{WE2}}$, DQMUL**	**$\overline{\text{WE1}}$, DQMLU**	**$\overline{\text{WE0}}$, DQMLL**
Byte access at 0		—	—	Data 7–0	—			Assert	—
Byte access at 1		—	—	—	Data 7–0				Assert
Byte access at 2		—	—	Data 7–0	—			Assert	—
Byte access at 3		—	—	—	Data 7–0				Assert
Word access at 0		—	—	Data 15–8	Data 7–0			Assert	Assert
Word access at 2		—	—	Data 15–8	Data 7–0			Assert	Assert
Longword access at 0	1st time at 0	—	—	Data 31–24	Data 23–16			Assert	Assert
	2nd time at 2	—	—	Data 15–8	Data 7–0			Assert	Assert

Table 13.7 8-Bit External Device/Big Endian Access and Data Alignment

		Data Bus				Strobe Signals			
Operation		D31–D24	D23–D16	D15–D8	D7–D0	$\overline{\text{WE3}}$, DQMUU	$\overline{\text{WE2}}$, DQMUL	$\overline{\text{WE1}}$, DQMLU	$\overline{\text{WE0}}$, DQMLL
Byte access at 0		—	—	—	Data 7–0				Assert
Byte access at 1		—	—	—	Data 7–0				Assert
Byte access at 2		—	—	—	Data 7–0				Assert
Byte access at 3		—	—	—	Data 7–0				Assert
Word access at 0	1st time at 0	—	—	—	Data 15–8				Assert
	2nd time at 1	—	—	—	Data 7–0				Assert
Word access at 2	1st time at 2	—	—	—	Data 15–8				Assert
	2nd time at 3	—	—	—	Data 7–0				Assert
Longword access at 0	1st time at 0	—	—	—	Data 31–24				Assert
	2nd time at 1	—	—	—	Data 23–16				Assert
	3rd time at 2	—	—	—	Data 15–8				Assert
	4th time at 3	—	—	—	Data 7–0				Assert

13.3.2 Description of Areas

Area 0: Area 0 physical addresses A28–A26 are 000. Addresses A31–A29 are ignored and the address range is H'00000000 + H'20000000 × n – H'03FFFFFF + H'20000000 × n (n = 0 to 6 and n = 1 to 6 are the shadow spaces).

Ordinary memories such as SRAM, ROM, and burst ROM with a burst function can be connected to this space. Byte, word, or longword can be selected as the bus width using external pins MD3 and MD4. When the area 0 space is accessed, a $\overline{\text{CS0}}$ signal is asserted. An $\overline{\text{RD}}$ signal that can be used as $\overline{\text{OE}}$ and the $\overline{\text{WE0}}$–$\overline{\text{WE3}}$ signals for write control are also asserted. The number of bus cycles is selected between 0 and 10 wait cycles using the A0W2–A0W0 bits of WCR2. When the burst function is used, the bus cycle pitch of the burst cycle is determined within a range of 2–10 according to the number of waits.

Area 1: Area 1 physical addresses A28–A26 are 001. Addresses A31–A29 are ignored and the address range is H'04000000 + H'20000000 × n – H'07FFFFFF + H'20000000 × n (n = 0 to 6 and n = 1 to 6 are the shadow spaces).

Area 1 is the area specifically for the internal peripheral modules. The external memories cannot be connected.

Control registers of peripheral modules shown below are mapped to this area 1. Their addresses are physical address, to which logical addresses can be mapped with the MMU enabled:

DMAC, PORT, SCIF0/1/2, ADC, INTC, USB

Those registers must not be cached.

Area 2: Area 2 physical addresses A28–A26 are 010. Addresses A31–A29 are ignored and the address range is H'08000000 + H'20000000 × n – H'0BFFFFFF + H'20000000 × n (n = 0 to 6 and n = 1 to 6 are the shadow spaces).

Ordinary memories such as SRAM and ROM, and synchronous DRAM, can be connected to this space. Byte, word, or longword can be selected as the bus width using the A2SZ1–A2SZ0 bits of BCR2 for ordinary memory. When synchronous DRAM is connected to Area 2, word or longword can be selected as the bus width.

When the area 2 space is accessed, a $\overline{\text{CS2}}$ signal is asserted. When ordinary memories are connected, an $\overline{\text{RD}}$ signal that can be used as $\overline{\text{OE}}$ and the $\overline{\text{WE0}}$–$\overline{\text{WE3}}$ signals for write control are also asserted and the number of bus cycles is selected between 0 and 3 wait cycles using the A2W1 to A2W0 bits of WCR2.

When synchronous DRAM is connected, the $\overline{\text{RAS3U}}$, $\overline{\text{RAS3L}}$ signal, $\overline{\text{CASU}}$, $\overline{\text{CASL}}$ signal, RD/$\overline{\text{WR}}$ signal, and byte controls DQMHH, DQMHL, DQMLH, and DQMLL are all asserted and addresses multiplexed. Control of $\overline{\text{RAS3U}}$, $\overline{\text{RAS3L}}$, $\overline{\text{CASU}}$, $\overline{\text{CASL}}$, data timing, and address multiplexing is set with MCR.

Area 3: Area 3 physical addresses A28–A26 are 011. Addresses A31–A29 are ignored and the address range is H'0C000000 + H'20000000 × n – H'0FFFFFFF + H'20000000 × n (n = 0 to 6 and n = 1 to 6 are the shadow spaces).

Ordinary memories such as SRAM and ROM, and synchronous DRAM, can be connected to this space. Byte, word or longword can be selected as the bus width using the A3SZ1–A3SZ0 bits of BCR2 for ordinary memory. When synchronous DRAM is connected to Area 3, word or longword can be selected as the bus width.

When area 3 space is accessed, $\overline{\text{CS3}}$ is asserted.

When ordinary memories are connected, an $\overline{\text{RD}}$ signal that can be used as $\overline{\text{OE}}$ and the $\overline{\text{WE0}}$–$\overline{\text{WE3}}$ signals for write control are asserted and the number of bus cycles is selected between 0 and 3 wait cycles using the A3W1 to A3W0 bits of WCR2.

When synchronous DRAM is connected, the $\overline{\text{RAS3U}}$, $\overline{\text{RAS3L}}$ signal, $\overline{\text{CASU}}$, $\overline{\text{CASL}}$ signal, RD/$\overline{\text{WR}}$ signal, and byte controls DQMHH, DQMHL, DQMLH, and DQMLL are all asserted and addresses multiplexed. Control of $\overline{\text{RAS}}$, $\overline{\text{CAS}}$, and data timing and of address multiplexing is set with MCR.

Area 4: Area 4 physical addresses A28–A26 are 100. Addresses A31–A29 are ignored and the address range is H'10000000 + H'20000000 × n – H'13FFFFFF + H'20000000 × n (n = 0 to 6 and n = 1 to 6 are the shadow spaces).

Only ordinary memories such as SRAM and ROM can be connected to this space. Byte, word, or longword can be selected as the bus width using the A4SZ1–A4SZ0 bits of BCR2. When the area 4 space is accessed, a $\overline{\text{CS4}}$ signal is asserted. An $\overline{\text{RD}}$ signal that can be used as $\overline{\text{OE}}$ and the $\overline{\text{WE0}}$–$\overline{\text{WE3}}$ signals for write control are also asserted. The number of bus cycles is selected between 0 and 10 wait cycles using the A4W2–A4W0 bits of WCR2. An arbitrary wait can also be inserted in each bus cycle by means of the external wait pin ($\overline{\text{WAIT}}$).

Area 5: Area 5 physical addresses A28–A26 are 101. Addresses A31–A29 are ignored and the address range is the 64 Mbytes at H'14000000 + H'20000000 × n – H'17FFFFFF + H'20000000 × n (n = 0 to 6 and n = 1 to 6 are the shadow spaces).

Ordinary memories such as SRAM, ROM, and burst ROM with a burst function can be connected to this space.

For ordinary memory and burst ROM, byte, word, or longword can be selected as the bus width using the A5SZ1–A5SZ0 bits of BCR2.

When the area 5 space is accessed and ordinary memory is connected, a $\overline{\text{CS5}}$ signal is asserted. An RD signal that can be used as $\overline{\text{OE}}$ and the $\overline{\text{WE0}}$–$\overline{\text{WE3}}$ signals for write control are also asserted.

The number of bus cycles is selected between 0 and 10 wait cycles using the A5W2–A5W0 bits of WCR2. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{WAIT}}$). When a burst function is used, the bus cycle pitch of the burst cycle is determined within a range of 2–10 according to the number of waits.

Area 6: Area 6 physical addresses A28–A26 are 110. Addresses A31–A29 are ignored and the address range is the 64 Mbytes at H'18000000 + H'20000000 × n – H'1BFFFFFF + H'20000000 × n (n = 0 to 6 and n = 1 to 6 are the shadow spaces).

Ordinary memories such as SRAM and ROM, and burst ROM with a burst function can be connected to this space.

For ordinary memory and burst ROM, byte, word, or longword can be selected as the bus width using the A6SZ1–A6SZ0 bits of BCR2.

When the area 6 space is accessed and ordinary memory is connected, a $\overline{\text{CS6}}$ signal is asserted. An $\overline{\text{RD}}$ signal that can be used as $\overline{\text{OE}}$ and the $\overline{\text{WE0}}$–$\overline{\text{WE3}}$ signals for write control are also asserted.

The number of bus cycles is selected between 0 and 10 wait cycles using the A6W2–A6W0 bits of WCR2. In addition, any number of waits can be inserted in each bus cycle by means of the external wait pin ($\overline{\text{WAIT}}$). The bus cycle pitch of the burst cycle is determined within a range of 2–10 according to the number of waits.

13.3.3 Basic Interface

Basic Timing: The basic interface of the SH7622 uses strobe signal output in consideration of the fact that mainly static RAM will be directly connected. Figure 13.5 shows the basic timing of normal space accesses. A no-wait normal access is completed in 2 cycles. The $\overline{\text{BS}}$ signal is asserted for 1 cycle to indicate the start of a bus cycle. The $\overline{\text{CSn}}$ signal is negated on the T2 clock falling edge to secure the negation period. Therefore, in case of access at minimum pitch, there is a half-cycle negation period.

There is no access size specification when reading. The correct access start address is output in the least significant bit of the address, but since there is no access size specification, 32 bits are always read in case of a 32-bit device, and 16 bits in case of a 16-bit device. When writing, only the $\overline{\text{WE}}$ signal for the byte to be written is asserted. For details, see section 13.3.1, Access Size and Data Alignment.

Read/write for cache fill or write-back follows the set bus width and transfers a total of 16 bytes continuously. The bus is not released during this transfer. For cache misses that occur during byte or word operand accesses or branching to odd word boundaries, the fill is always performed by longword accesses on the chip-external interface. Write-through-area write access and non-cacheable read/write access are based on the actual address size.

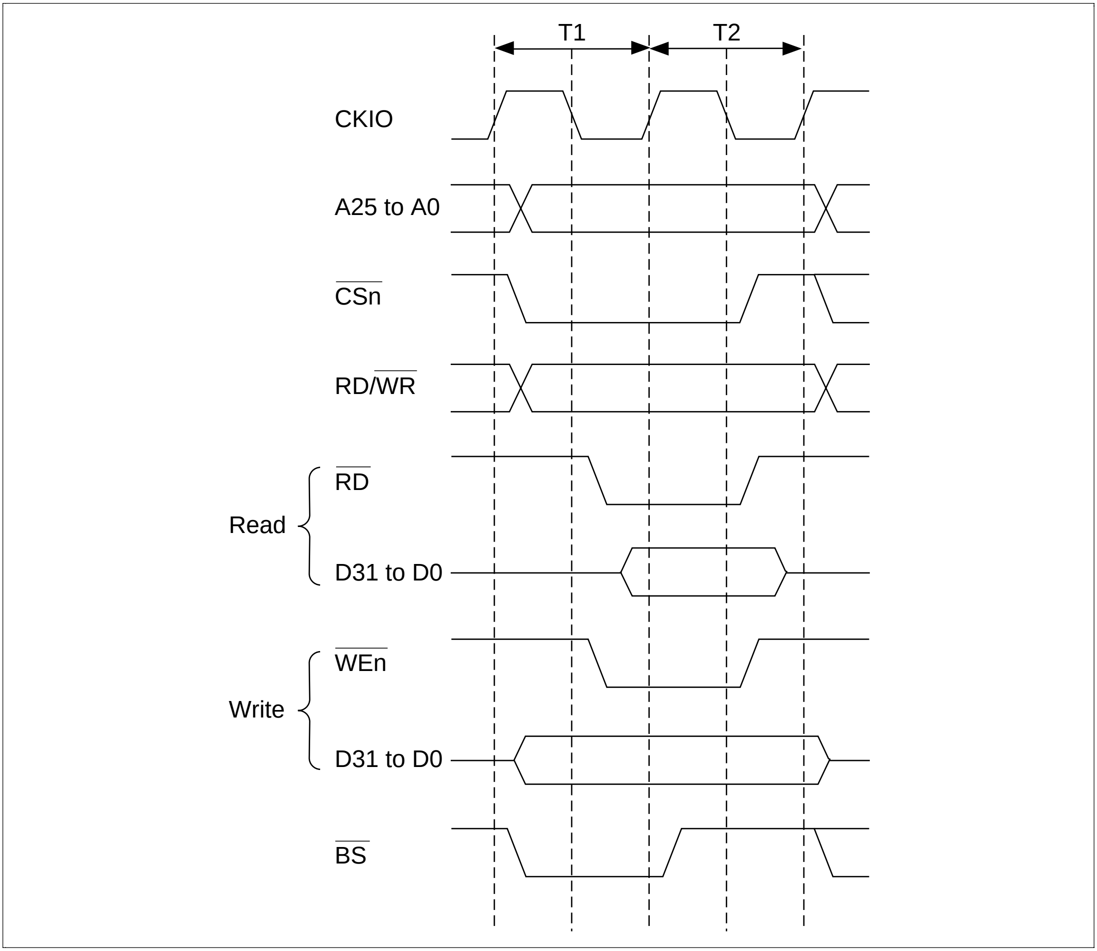


Figure 13.5 Basic Timing of Basic Interface

Figures 13.6, 13.7, and 13.8 show examples of connection to 32, 16, and 8-bit data-width static RAM, respectively.

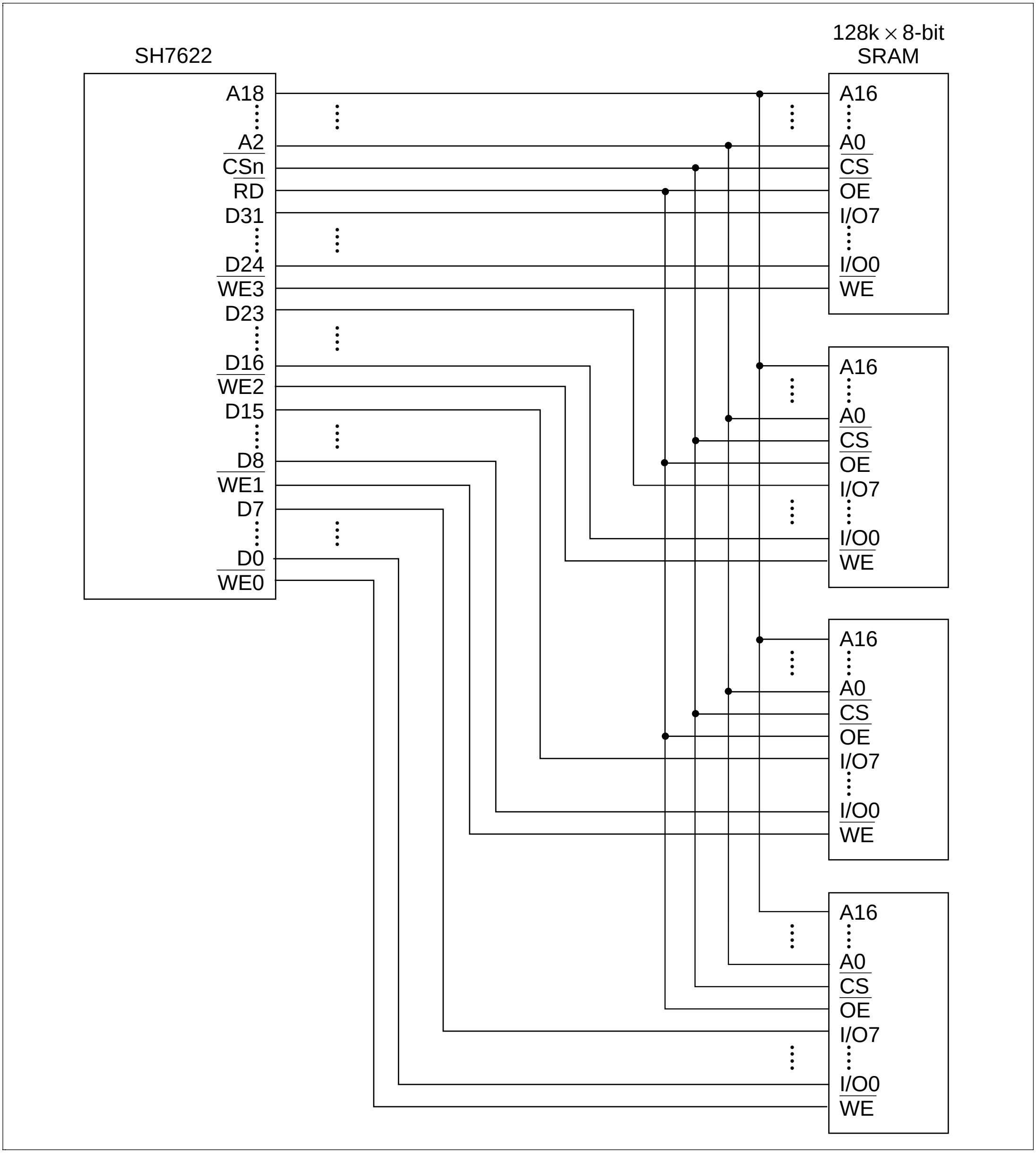


Figure 13.6 Example of 32-Bit Data-Width Static RAM Connection

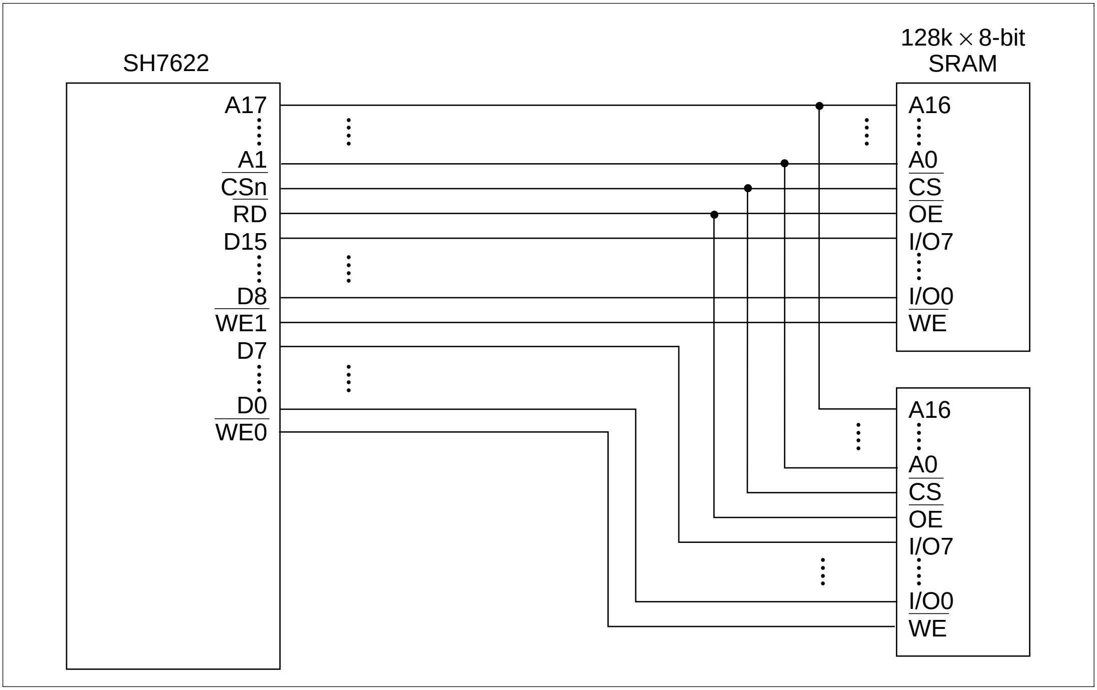


Figure 13.7 Example of 16-Bit Data-Width Static RAM Connection

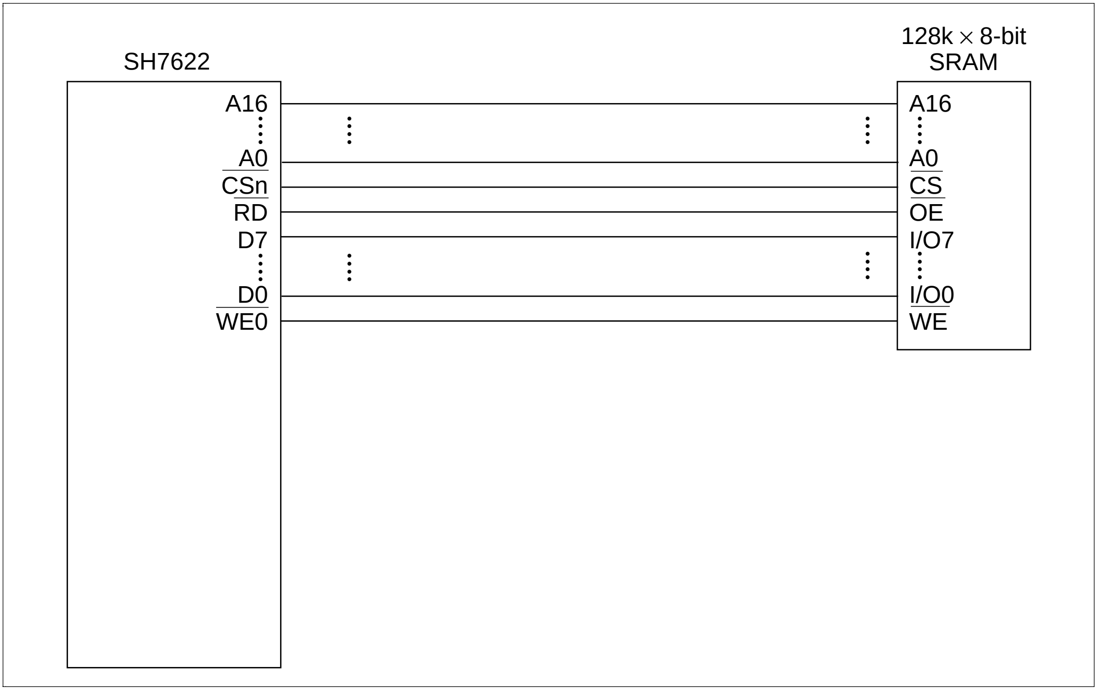


Figure 13.8 Example of 8-Bit Data-Width Static RAM Connection

Wait State Control: Wait state insertion on the basic interface can be controlled by the WCR2 settings. If the WCR2 wait specification bits corresponding to a particular area are not zero, a software wait is inserted in accordance with that specification. For details, see section 13.2.4, 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 13.9.

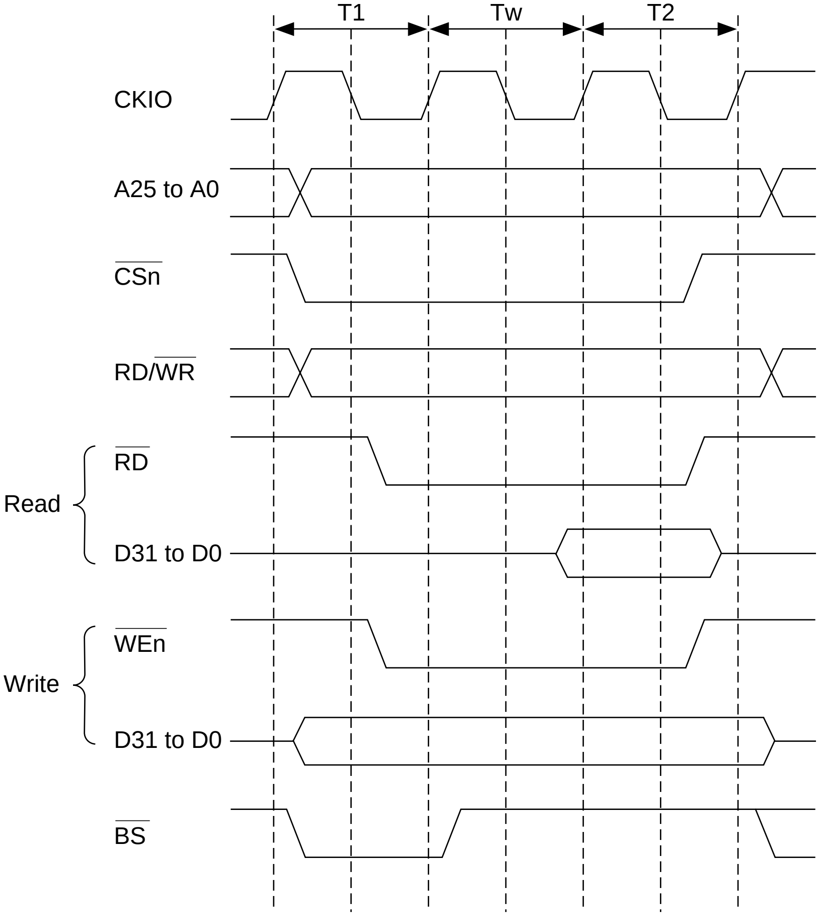


Figure 13.9 Basic Interface Wait Timing (Software Wait Only)

When software wait insertion is specified by WCR2, the external wait input $\overline{\text{WAIT}}$ signal is also sampled. $\overline{\text{WAIT}}$ pin sampling is shown in figure 13.10. A 2-cycle wait is specified as a software wait. Sampling is performed at the transition from the Tw state to the T2 state; therefore, if the $\overline{\text{WAIT}}$ signal has no effect if asserted in the T1 cycle or the first Tw cycle. When the WAITSEL bit in WCR1 is cleared to 0, the $\overline{\text{WAIT}}$ signal is sampled on the rising edge of the clock. In this case $\overline{\text{WAIT}}$ is a synchronous input signal, so setup and hold times must be observed.

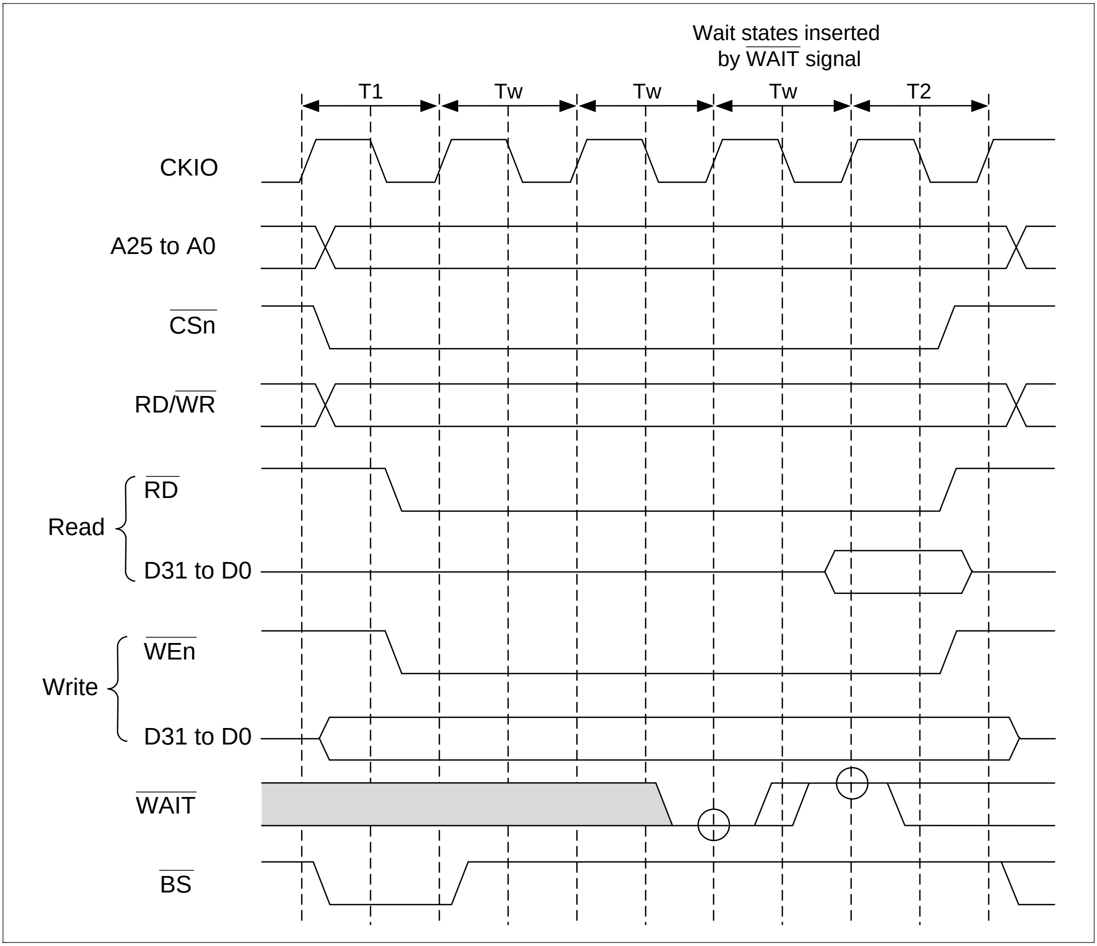


Figure 13.10 Basic Interface Wait State Timing (Wait State Insertion by $\overline{\text{WAIT}}$ Signal WAITSEL = 0)

When the WAITSEL bit in the WCR1 register is set to 1, the $\overline{\text{WAIT}}$ signal is sampled at the falling edge of the clock. If the setup time and hold times with respect to the falling edge of the clock are not satisfied, the value sampled at the next falling edge is used.

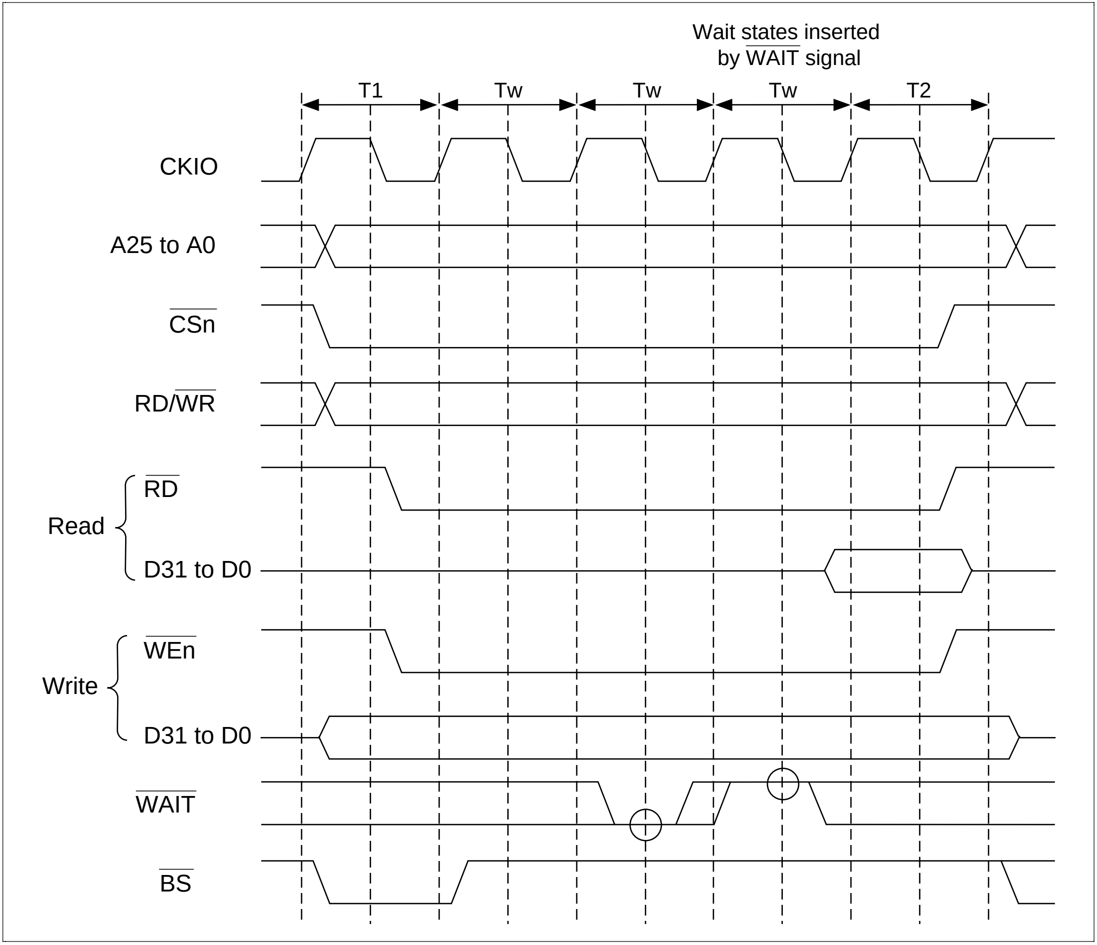


Figure 13.11 Basic Interface Wait State Timing
(Wait State Insertion by $\overline{\text{WAIT}}$ Signal WAITSEL = 1)

13.3.4 Synchronous DRAM Interface

Synchronous DRAM Direct Connection: Since synchronous DRAM can be selected by the $\overline{\text{CS}}$ signal, physical space areas 2 and 3 can be connected 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 ordinary memory space and area 3 is synchronous DRAM space; if set to 011, areas 2 and 3 are both synchronous DRAM space.

With the SH7622, burst length 1 burst read/single write mode is supported as the synchronous DRAM operating mode. A data bus width of 16 or 32 bits can be selected. The burst enable bit (BE) in MCR is ignored, a 16-bit burst transfer is performed in a cache fill/write-back cycle, and only one access is performed in a write-through area write or a non-cacheable area read/write.

The control signals for direct connection of synchronous DRAM are $\overline{\text{RAS3L}}$, $\overline{\text{RAS3U}}$, $\overline{\text{CASL}}$, $\overline{\text{CASU}}$, RD/$\overline{\text{WR}}$, $\overline{\text{CS2}}$ or $\overline{\text{CS3}}$, DQMUU, DQMUL, DQMLU, DQMLL, 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 fetched to the synchronous DRAM only when CS2 or CS3 is asserted. Synchronous DRAM can therefore be connected in parallel to a number of areas. CKE is negated (low) only when self-refreshing is performed, and is always asserted (high) at other times.

However, which of the $\overline{\text{RAS3L}}$, $\overline{\text{RAS3U}}$, $\overline{\text{CASL}}$, and $\overline{\text{CASU}}$ signals are output is determined by whether the address is in the upper or lower 32 Mbytes in each area. When the address is in upper 32 Mbytes, $\overline{\text{RAS3U}}$ and $\overline{\text{CASU}}$ are output; when in lower 32 Mbytes, $\overline{\text{RAS3L}}$ and $\overline{\text{CASL}}$ are output. In the refresh cycle and mode-register write cycle, $\overline{\text{RAS3U}}$ and $\overline{\text{RAS3L}}$ or $\overline{\text{CASU}}$ and $\overline{\text{CASL}}$ are output.

Commands for synchronous DRAM are specified by $\overline{\text{RAS3L}}$, $\overline{\text{RAS3U}}$, $\overline{\text{CASL}}$, $\overline{\text{CASU}}$, RD/$\overline{\text{WR}}$, and special address signals. The commands are NOP, auto-refresh (REF), self-refresh (SELF), precharge all banks (PALL), row address strobe bank active (ACTV), read (READ), read with precharge (READA), write (WRIT), write with precharge (WRITA), and mode register write (MRS).

Byte specification is performed by DQMUU, DQMUL, DQMLU, and DQMLL. A read/write is performed for the byte for which the corresponding DQM is low. In big-endian mode, DQMUU specifies an access to address 4n, and DQMLL specifies an access to address 4n + 3. In little-endian mode, DQMUU specifies an access to address 4n + 3, and DQMLL specifies an access to address 4n.

Figure 13.12 shows an example of the connection of two 4M × 16-bit synchronous DRAMs, and figures 13.13 (1) and 13.13 (2) show examples of the connection of one 2M × 32-bit synchronous DRAM, and one 4M × 16-bit synchronous DRAM, respectively.

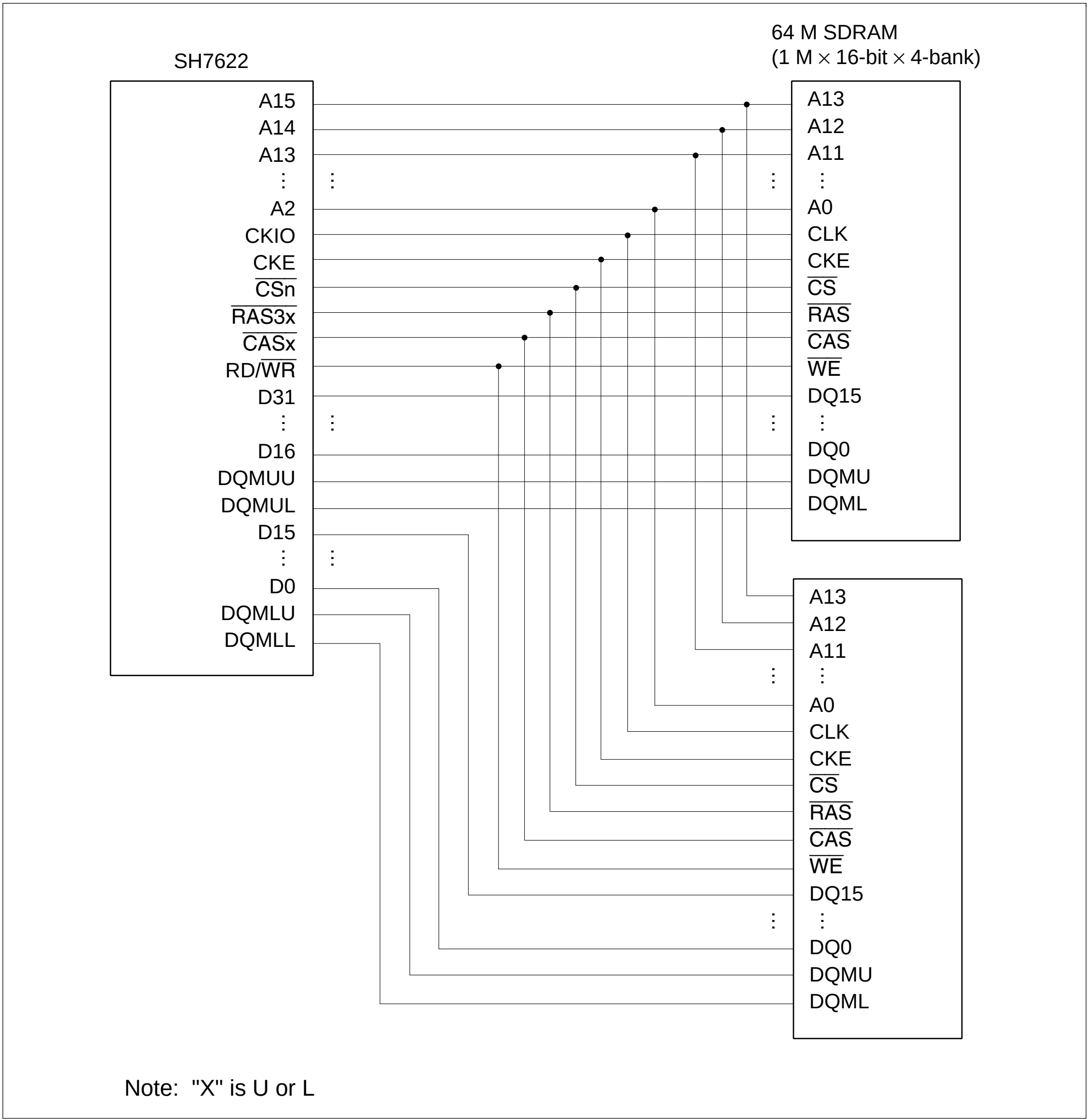


Figure 13.12 Example of 64-Mbit Synchronous DRAM Connection (32-Bit Bus Width)

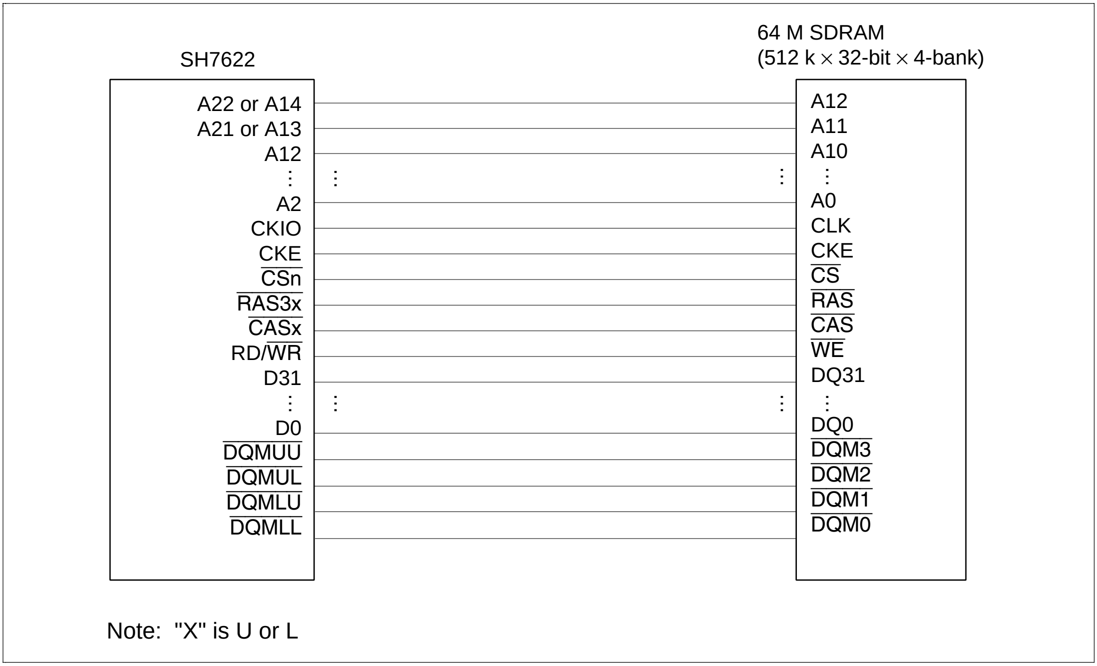


Figure 13.13 (1) Example of 64-Mbit Synchronous DRAM (32-Bit Bus Width)

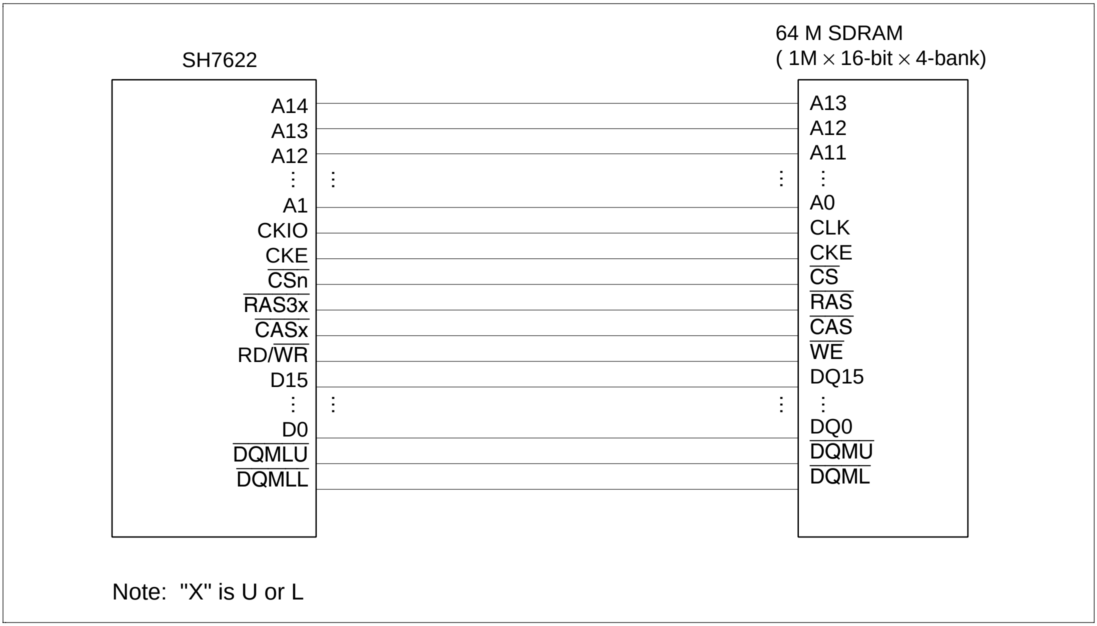


Figure 13.13 (2) Example of 64-Mbit Synchronous DRAM (16-Bit Bus Width)

Address Multiplexing: Synchronous DRAM can be connected without external multiplexing circuitry in accordance with the address multiplex specification bits AMX2-AMX0 in MCR. Table 13.8 shows the relationship between the address multiplex specification bits and the bits output at the address pins.

A25–A16 and A0 are not multiplexed; the original values are always output at these pins.

When A0, the LSB of the synchronous DRAM address, is connected to the SH7622, it performs longword address specification. Connection should therefore be made in the following order: with a 32-bit bus width, connect pin A0 of the synchronous DRAM to pin A2 of the SH7622, then connect pin A1 to pin A3; with a 16-bit bus width, connect pin A0 of the synchronous DRAM to pin A1 of the SH7622, then connect pin A1 to pin A2.

Table 13.8 Relationship between Bus Width, AMX, and Address Multiplex Output

	Setting				External Address Pins							
Bus Width	**AMX2**	**AMX1**	**AMX0**	**Output Timing**	**A1 to A8**	**A9**	**A10**	**A11**	**A12**	**A13**	**A14**	**A15**
32 bits	1	0	0	Column address	A1 to A8	A9	A10	A11	L/H*[1]	A13	A22*[2]	A23*[2]
				Row address	A9 to A16	A17	A18	A19	A20	A21	A22*[2]	A23*[2]
			1	Column address	A1 to A8	A9	A10	A11	L/H*[1]	A13	A23*[2]	A24*[2]
				Row address	A10 to A17	A18	A19	A20	A21	A22	A23*[2]	A24*[2]
		1	1	Column address	A1 to A8	A9	A20	A11	L/H*[1]	A21*[2]	A22*[2]	A15
				Row address	A9 to A16	A17	A18	A19	A20	A21*[2]	A22*[2]	A23
	0	0	0	Column address	A1 to A8	A9	A10	A11	L/H*[1]	A21*[2]	A14	A15
				Row address	A9 to A16	A17	A18	A19	A20	A21*[2]	A22	A23
			1	Column address	A1 to A8	A9	A10	A11	L/H*[1]	A22*[2]	A14	A15
				Row address	A10 to A17	A18	A19	A20	A21	A22*[2]	A23	A24
		1	0	Column address	A1 to A8	A9	A20	A11	L/H*[1]	A23*[2]	A14	A15
				Row address	A11 to A18	A19	A20	A21	A22	A23*[2]	A24	A25
			1	Column address	A1 to A8	A9	L/H*[1]	A19*[2]	A12	A13	A14	A15
				Row address	A9 to A16	A17	A18	A19*[2]	A20	A21	A22	A23

Notes: *1 L/H is a bit used in the command specification; it is fixed at L or H according to the access mode.

*2 Bank address specification

Table 13.8 Relationship between Bus Width, AMX, and Address Multiplex Output (cont)

	Setting				External Address Pins							
Bus Width	**AMX2**	**AMX1**	**AMX0**	**Output Timing**	**A1 to A8**	**A9**	**A10**	**A11**	**A12**	**A13**	**A14**	**A15**
16 bits	1	0	0	Column address	A1 to A8	A9	A10	L/H*1	A12	A21*2	A22*2	A15
				Row address	A9 to A16	A17	A18	A19	A20	A21*2	A22*2	A23
			1	Column address	A1 to A8	A9	A10	L/H*1	A12	A22*2	A23*2	A15
				Row address	A10 to A17	A18	A19	A20	A21	A22*2	A23*2	A24
		1	0	Column address	A1 to A8	A9	A10	L/H*1	A12	A23*2	A24*2	A15
				Row address	A11 to A18	A19	A20	A21	A22	A23*2	A24*2	A25
	0	0	0	Column address	A1 to A8	A9	A10	L/H*1	A20*2	A13	A14	A15
				Row address	A9 to A16	A17	A18	A19	A20*2	A21	A22	A23
			1	Column address	A1 to A8	A9	A10	L/H*1	A21*2	A13	A14	A15
				Row address	A10 to A17	A18	A19	A20	A21*2	A22	A23	A24
		1	0	Column address	A1 to A8	A9	A10	L/H*1	A22*2	A13	A14	A15
				Row address	A11 to A18	A19	A20	A21	A22*2	A23	A24	A25
			1	Column address	A1 to A8	L/H*1	A18*2	A11	A12	A13	A14	A15
				Row address	A9 to A16	A17	A18*2	A19	A20	A21	A22	A23

Notes: *1 L/H is a bit used in the command specification; it is fixed at L or H according to the access mode.

*2 Bank address specification

Table 13.9 Example of Correspondence between SH7622 and Synchronous DRAM Address Pins (AMX (2–0) = 011 (32-Bit Bus Width))

SH7622 Address Pin			Synchronous DRAM Address Pin	
	RAS Cycle	CAS Cycle		Function
A11	A19	A19	A9	BANK select bank address
A10	A18	L/H	A8	Address precharge setting
A9	A17	A9	A7	Address
A8	A16	A8	A6	
A7	A15	A7	A5	
A6	A14	A6	A4	
A5	A13	A5	A3	
A4	A12	A4	A2	
A3	A11	A3	A1	
A2	A10	A2	A0	
A1	A9	A1	Not used	
A0	A0	A0	Not used	

Burst Read: In the example in figure 13.14 it is assumed that four 2M × 8-bit synchronous DRAMs are connected and a 32-bit data width is used, and the burst length is 1. Following the Tr cycle in which ACTV command output is performed, a READ command is issued in the Tc1, Tc2, and Tc3 cycles, and a READA command in the Tc4 cycle, and the read data is accepted on the rising edge of the external command clock (CKIO) from cycle Td1 to cycle Td4. The Tpc cycle is used to wait for completion of auto-precharge based on the READA command inside the synchronous DRAM; no new access command can be issued to the same bank during this cycle, but access to synchronous DRAM for another area is possible. In the SH7622, the number of Tpc cycles is determined by the TPC bit specification in MCR, and commands cannot be issued for the same synchronous DRAM during this interval.

The example in figure 13.14 shows the basic timing. To connect low-speed 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 READ command output cycle, Tc1, can be specified by the RCD bit in MCR, with a values of 0–3 specifying 1–4 cycles, respectively. In 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 READ and READA command output cycles Tc1–Tc4 to the first read data latch cycle, Td1, can be specified as 1–3 cycles independently for areas 2 and 3 by means of A2W1 and A2W0 or A3W1 and A3W0 in WCR2. This number of cycles corresponds to the number of synchronous DRAM CAS latency cycles.

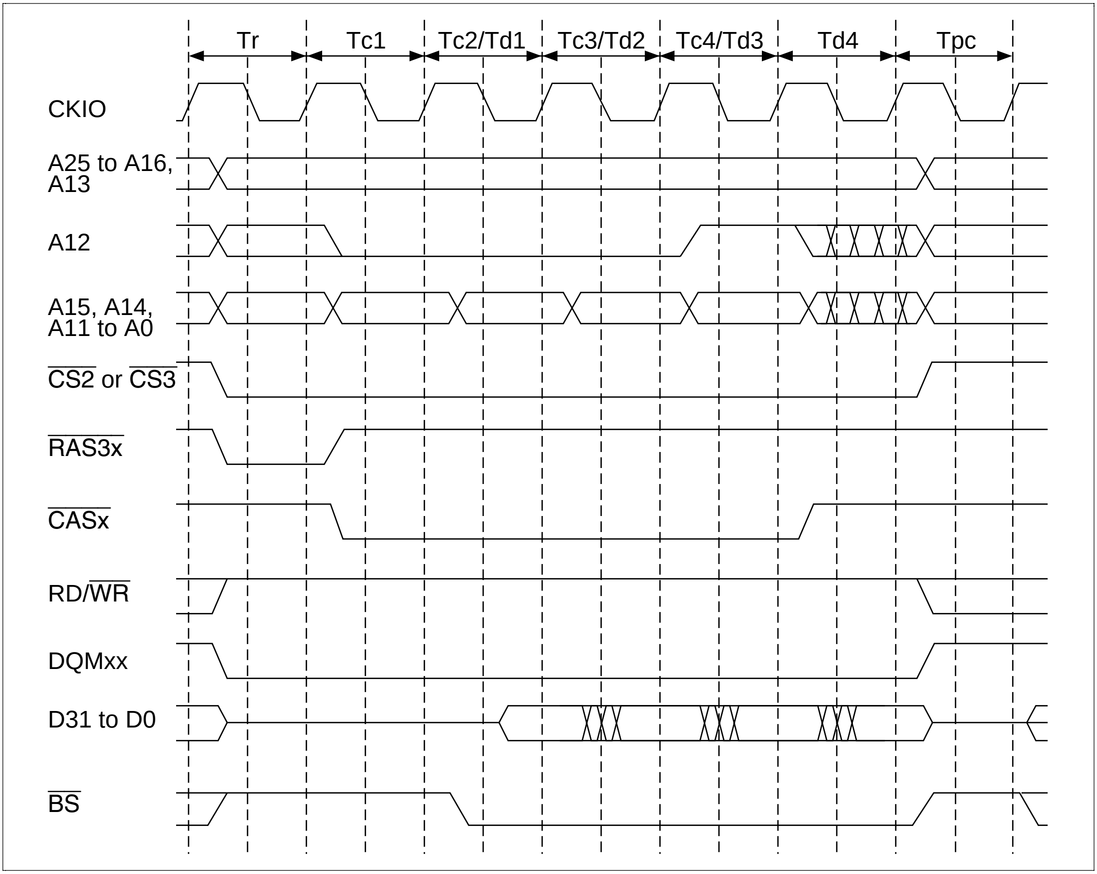


Figure 13.14 Basic Timing for Synchronous DRAM Burst Read

Figure 13.15 shows the burst read timing when RCD is set to 1, A3W1 and A3W0 are set to 10, and TPC is set to 1.

The $\overline{\text{BS}}$ signal, which is asserted for 1 cycle at the start of a bus cycle for normal access space, is asserted in each of cycles Td1–Td4 in a synchronous DRAM cycle. When a burst read is performed, the address is updated each time $\overline{\text{CAS}}$ is asserted. As the unit of burst transfer is 16 bytes, address updating is performed for A3 and A2 only (or for A3, A2, and A1 when using a 16-bit bus width). The order of access is as follows: in a fill operation in the event of a cache miss, the missed data is read first, then 16-byte boundary data including the missed data is read in wraparound mode.

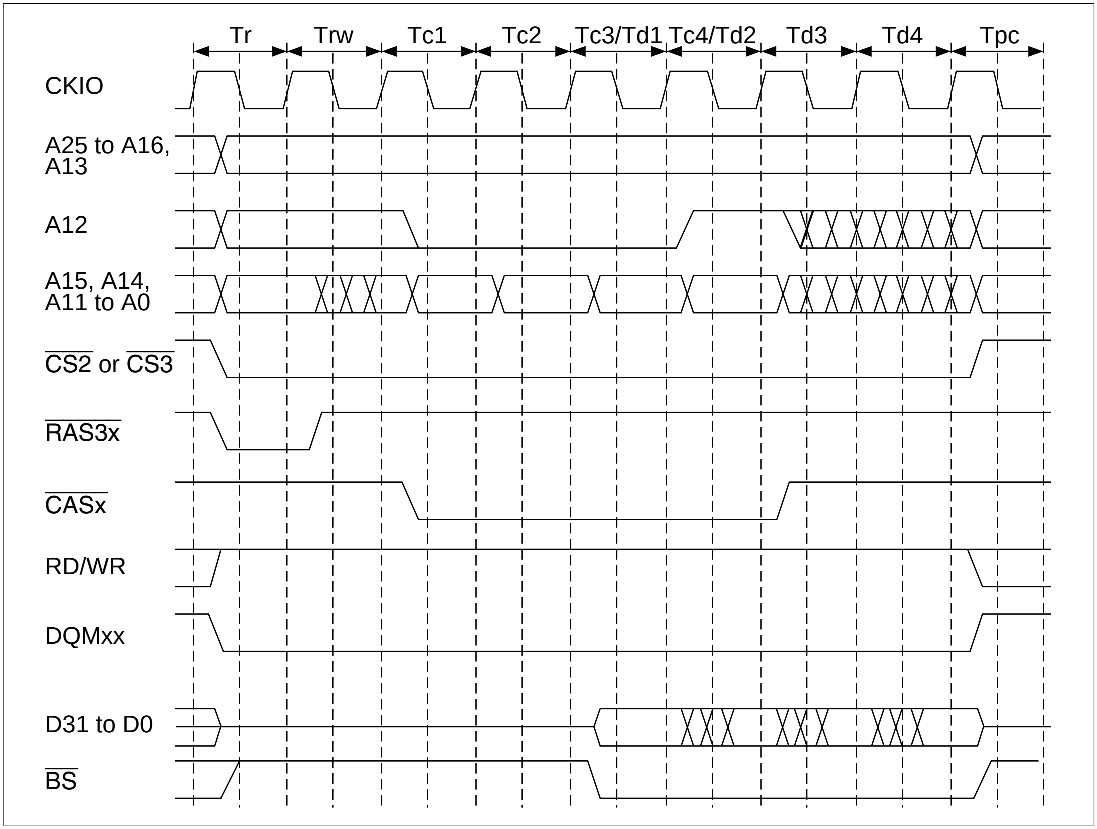


Figure 13.15 Synchronous DRAM Burst Read Wait Specification Timing

Single Read: Figure 13.16 shows the timing when a single address read is performed. As the burst length is set to 1 in synchronous DRAM burst read/single write mode, only the required data is output. Consequently, no unnecessary bus cycles are generated even when a cache-through area is accessed.

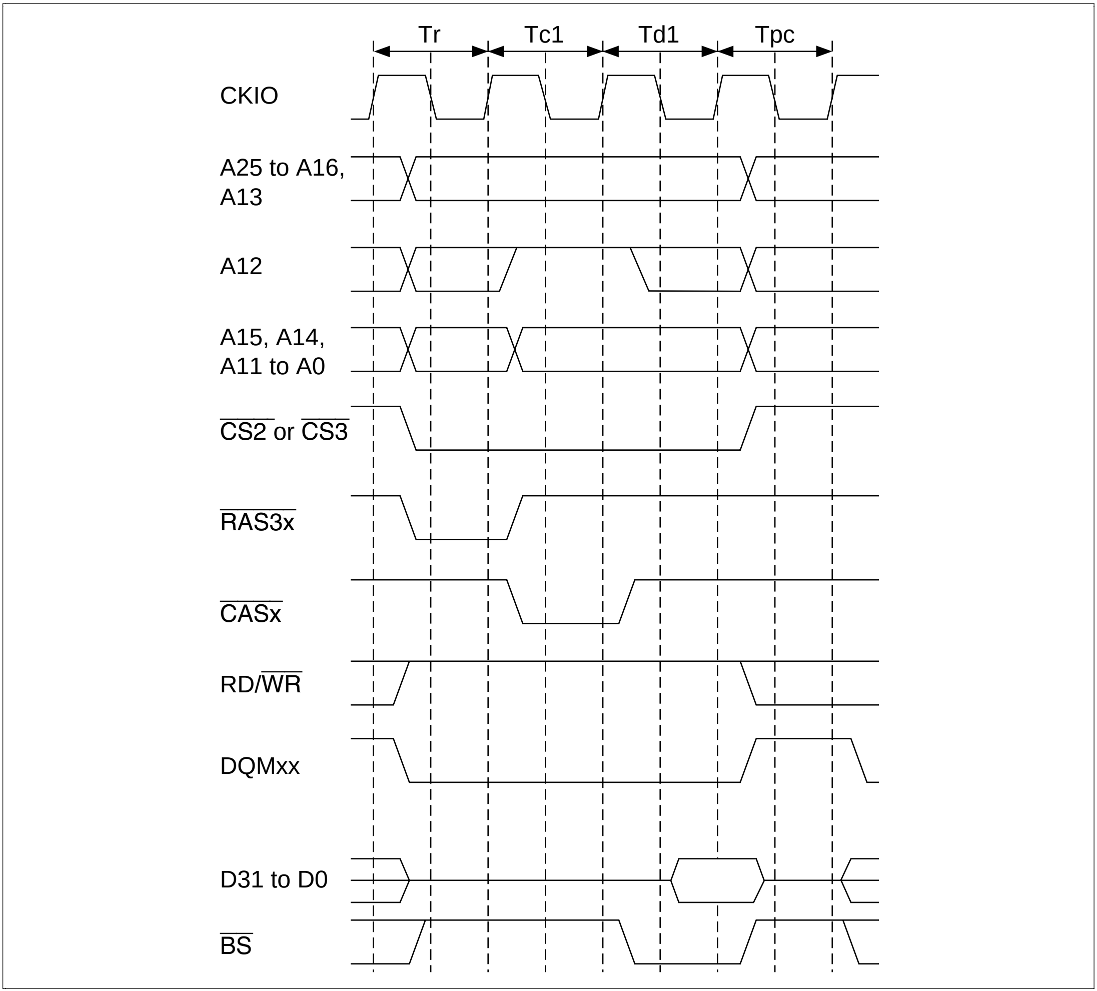


Figure 13.16 Basic Timing for Synchronous DRAM Single Read

Burst Write: The timing chart for a burst write is shown in figure 13.17. In the SH7622, a burst write occurs only in the event of copy-back or DMAC 16-byte transfer. In a burst write operation, following the Tr cycle in which ACTV command output is performed, a WRIT command is issued in the Tc1, Tc2, and Tc3 cycles, and a WRITA command that performs auto-precharge is issued in the Tc4 cycle. In the write cycle, the write data is output at the same time as the write command. In case of the write with auto-precharge command, precharging of the relevant bank is performed in the 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 the TRWL bit in MCR.

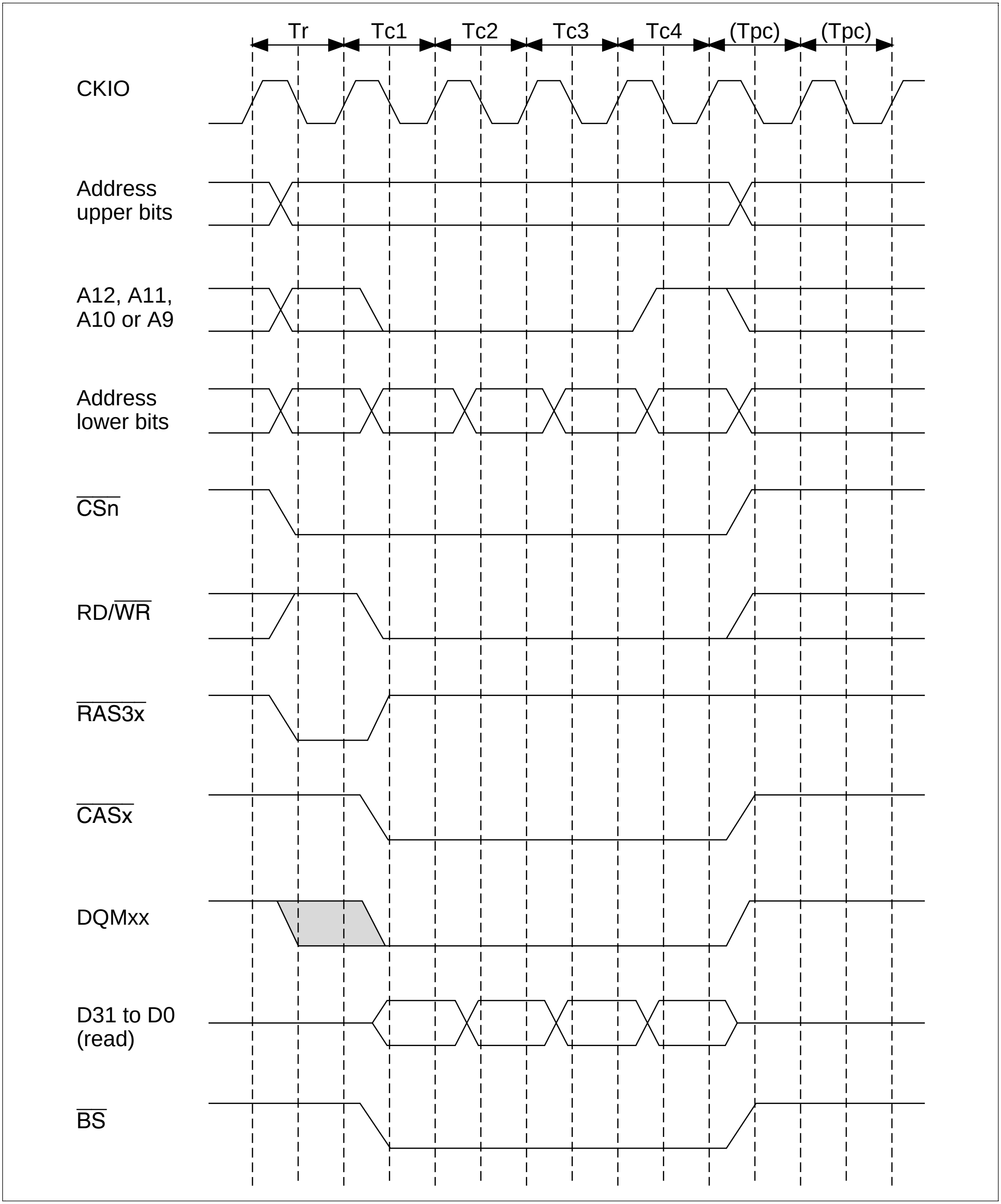


Figure 13.17 Basic Timing for Synchronous DRAM Burst Write

Single Write: The basic timing chart for write access is shown in figure 13.18. In a single write operation, following the Tr cycle in which ACTV command output is performed, a WRITA command that performs auto-precharge is issued in the Tc1 cycle. In the write cycle, the write data is output at the same time as the write command. In case of the write with auto-precharge command, precharging of the relevant bank is performed in the 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 the TRWL bit in MCR.

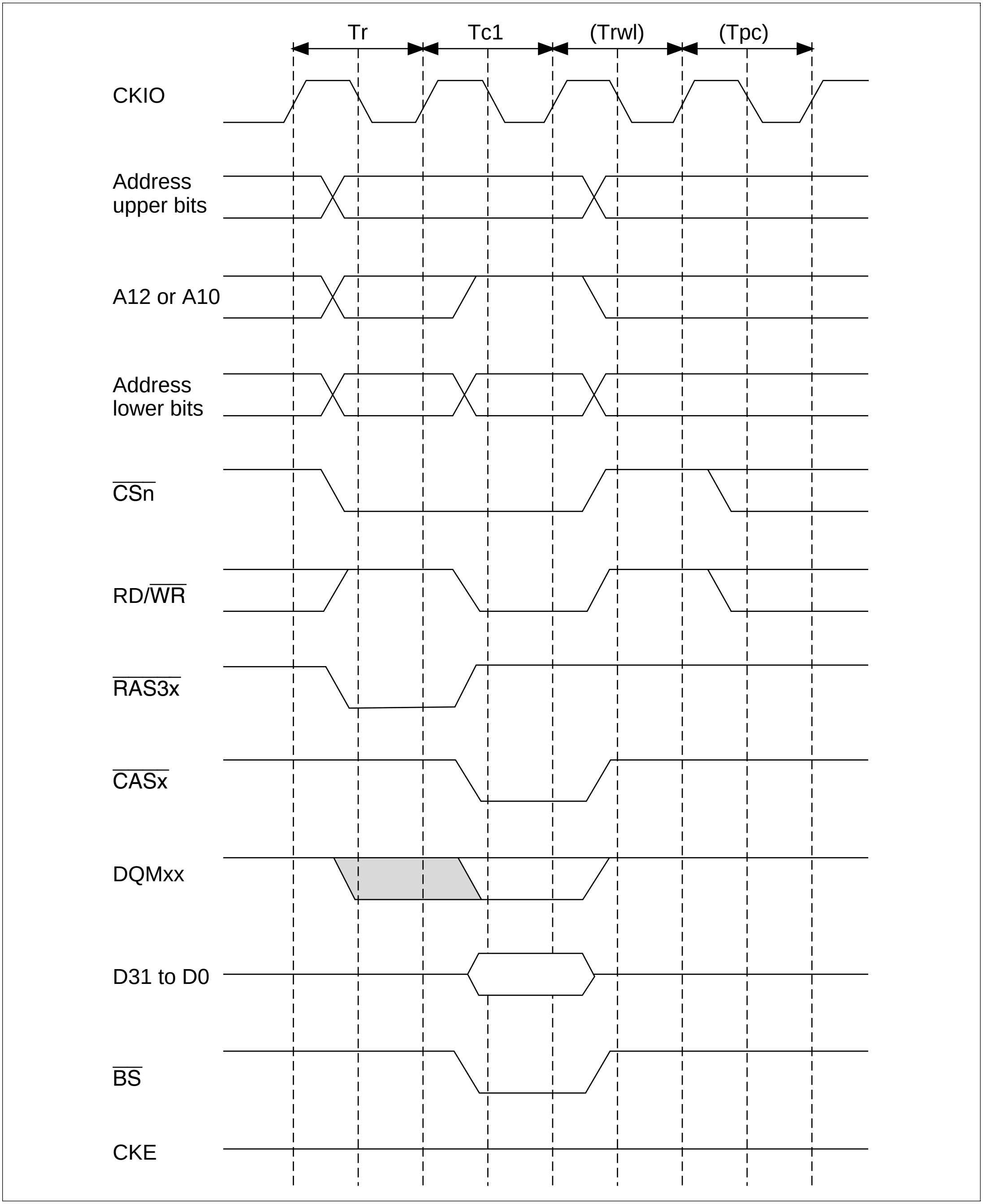


Figure 13.18 Basic Timing for Synchronous DRAM Single Write

Bank Active: 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 RAS down state in DRAM fast page mode. 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 WRITA command. When bank active mode is used, READ or WRIT commands can be issued successively if the row address is the same. The number of cycles can thus be reduced by Trwl + Tpc cycles for each write. The number of cycles between issuance of the precharge command and the row address strobe command is determined by the TPC bit in MCR.

Whether faster execution speed is achieved by use of bank active mode or by use of basic access is determined by the probability of accessing the same row address (P1), and the average number of cycles from completion of one access to the next access (Ta). If Ta is greater than Tpc, the delay due to the precharge wait when writing is imperceptible. In this case, the access speed for bank active mode and basic access is determined by the number of cycles from the start of access to issuance of the read/write command: (Tpc + Trcd) × (1 – P1) and Trcd, respectively.

There is a limit on Tras, the time for placing each bank in the active state. If there is no guarantee that there will not be a cache hit and another row address will be accessed within the period in which this value is maintained by program execution, it is necessary to set auto-refresh and set the refresh cycle to no more than the maximum value of Tras. In this way, it is possible to observe the restrictions on the maximum active state time for each bank. If auto-refresh is not used, measures must be taken in the program to ensure that the banks do not remain active for longer than the prescribed time.

A burst read cycle without auto-precharge is shown in figure 13.19, a burst read cycle for the same row address in figure 13.20, and a burst read cycle for different row addresses in figure 13.21. Similarly, a burst write cycle without auto-precharge is shown in figure 13.22, a burst write cycle for the same row address in figure 13.23, and a burst write cycle for different row addresses in figure 13.24.

A Tnop cycle, in which no operation is performed, is inserted before the Tc cycle in which the READ command is issued in figure 13.20, but when synchronous DRAM is read, there is a 2-cycle latency for the DQMxx signal that performs the byte specification. The reason for inserting the Tnop cycle is that, if the Tc cycle were performed immediately, without an intervening Tnop cycle, it would not be possible to make the DQMxx signal specification for Td1 cycle data output. If the CAS latency is 2 cycles or longer, Tnop cycle insertion is not performed, since the timing requirements will be met even if the DQMxx signal is set after the Tc cycle.

When bank active mode is set, if only accesses to the respective banks in the area 3 space are considered, as long as accesses to the same row address continue, the operation starts with the cycle in figure 13.19 or 13.22, followed by repetition of the cycle in figure 13.20 or 13.23. An access to a different area 3 space during this time has no effect. If there is an access to a different row address in the bank active state, after this is detected the bus cycle in figure 13.21 or 13.24 is executed instead of that in figure 13.20 or 13.23. In bank active mode, too, all banks become inactive after a refresh cycle or after the bus is released as the result of bus arbitration.

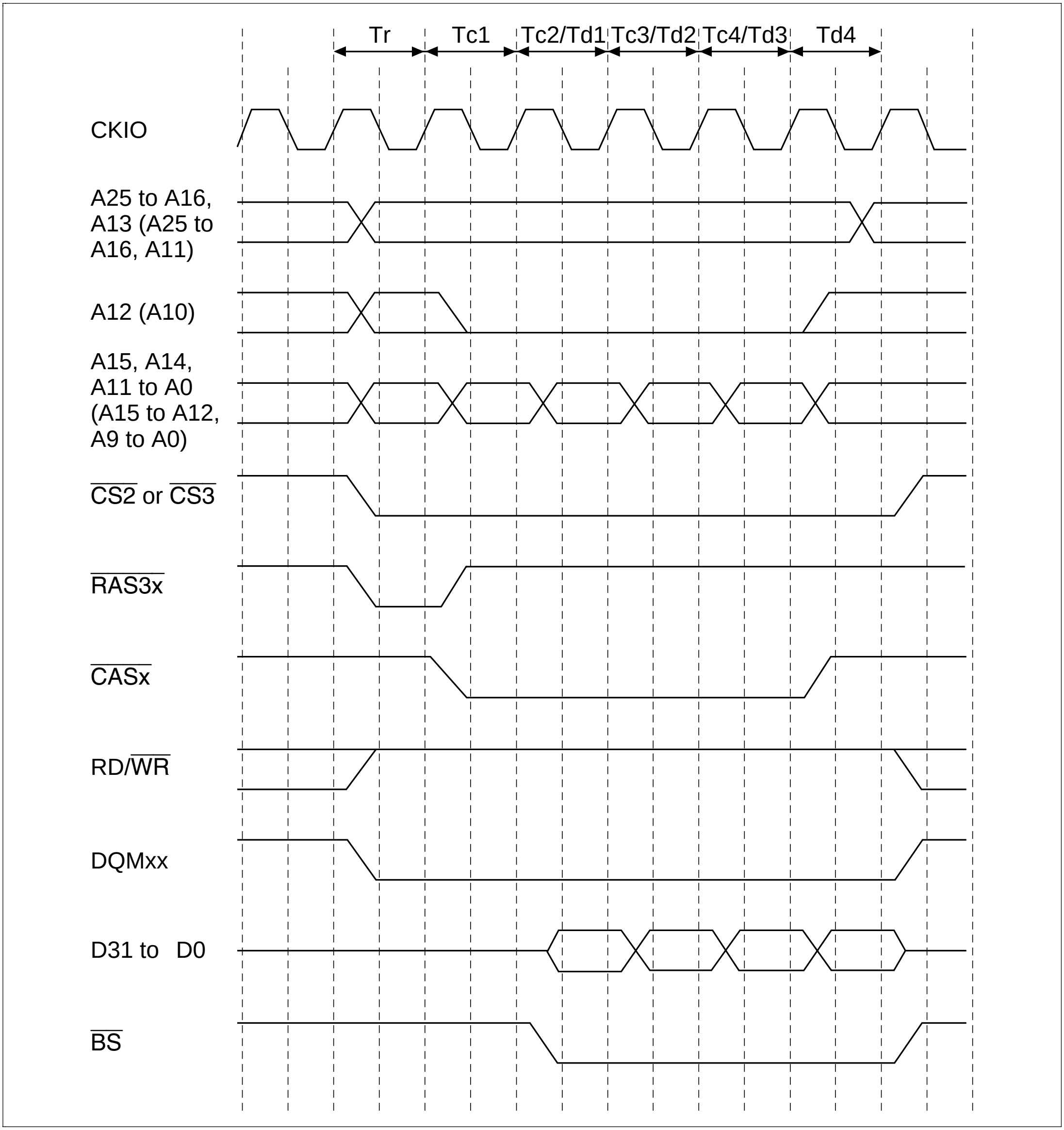

Tr
Tc1
Tc2/Td1
Tc3/Td2
Tc4/Td3
Td4
CKIO
A25 to A16, A13 (A25 to A16, A11)
A12 (A10)
A15, A14, A11 to A0 (A15 to A12, A9 to A0)
$\overline{\text{CS2}}$ or $\overline{\text{CS3}}$
$\overline{\text{RAS3x}}$
$\overline{\text{CASx}}$
RD/$\overline{\text{WR}}$
DQMxx
D31 to D0
$\overline{\text{BS}}$

Figure 13.19 Burst Read Timing (No Precharge)

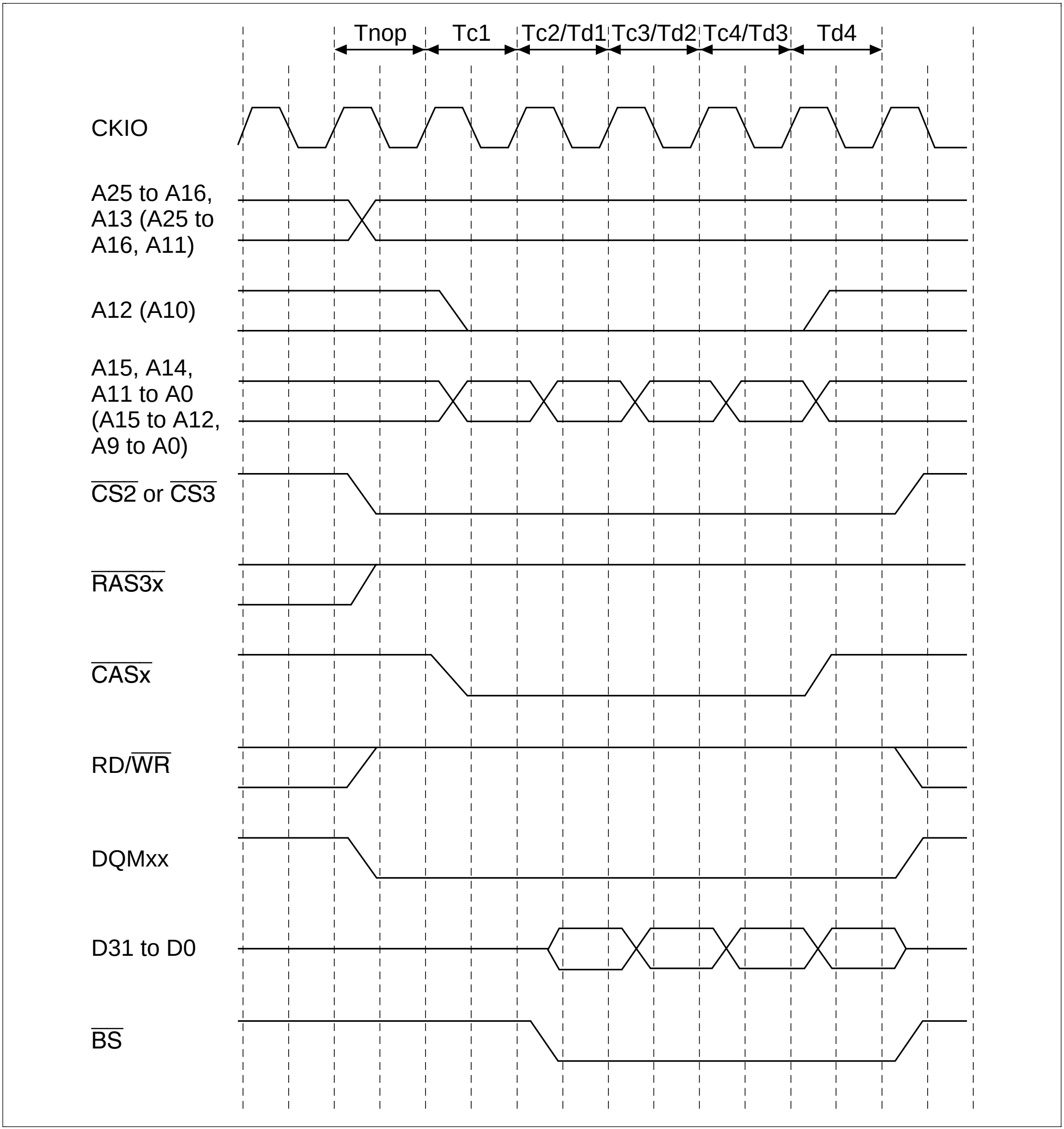


Figure 13.20 Burst Read Timing (Same Row Address)

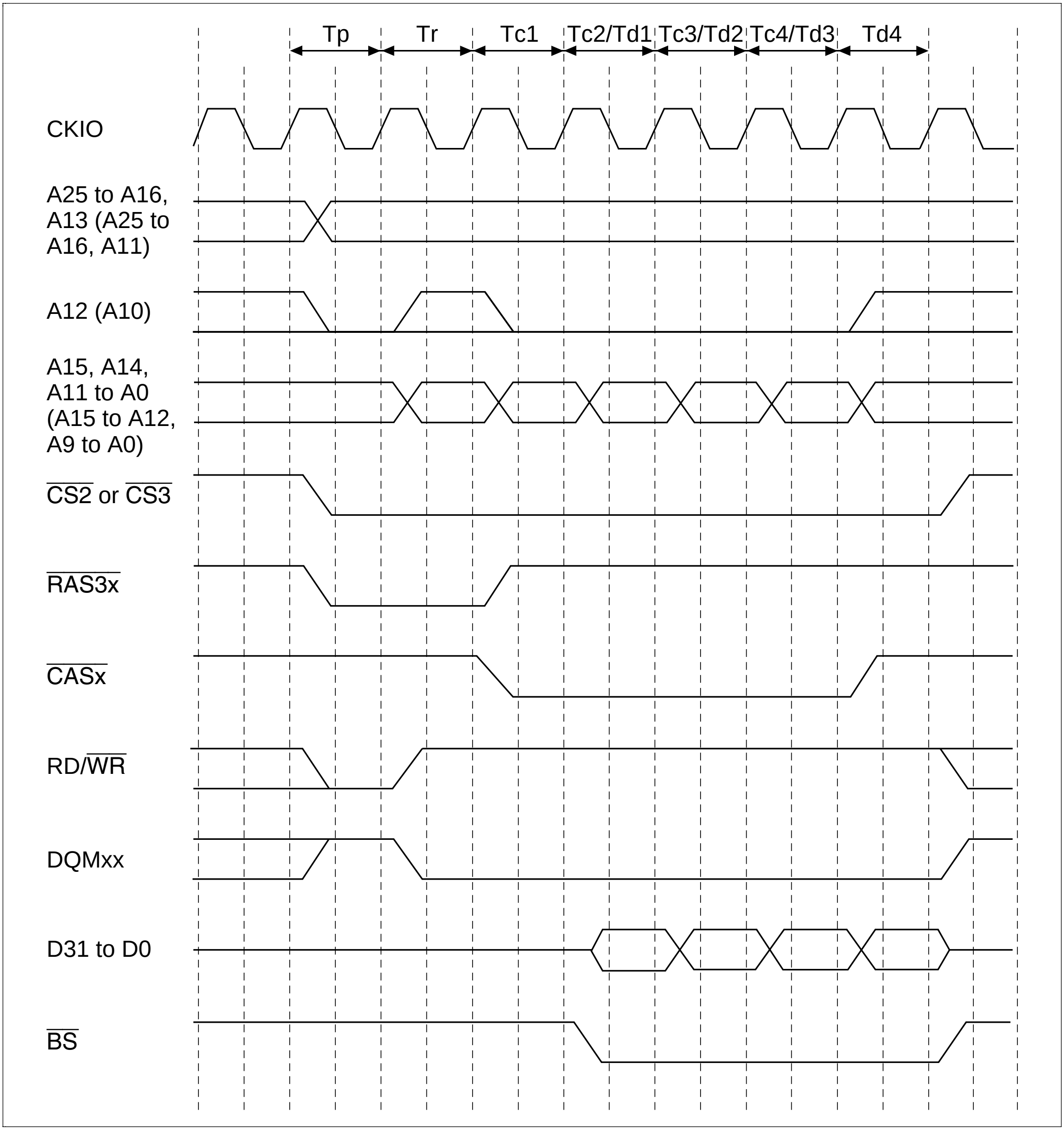


Figure 13.21 Burst Read Timing (Different Row Addresses)

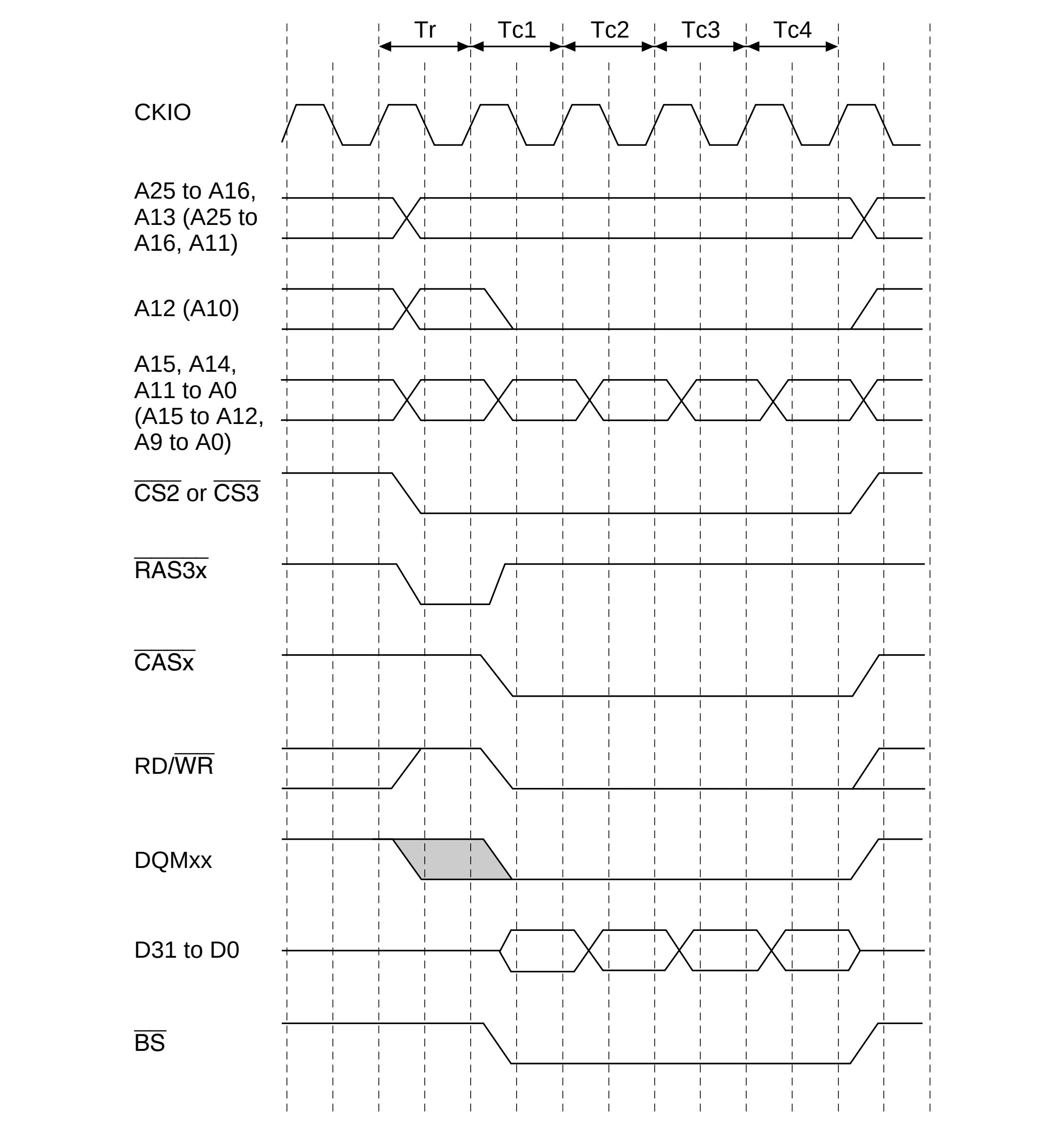

Tr
Tc1
Tc2
Tc3
Tc4
CKIO
A25 to A16, A13 (A25 to A16, A11)
A12 (A10)
A15, A14, A11 to A0 (A15 to A12, A9 to A0)
$\overline{\text{CS2}}$ or $\overline{\text{CS3}}$
$\overline{\text{RAS3x}}$
$\overline{\text{CASx}}$
RD/$\overline{\text{WR}}$
DQMxx
D31 to D0
$\overline{\text{BS}}$

Figure 13.22 Burst Write Timing (No Precharge)

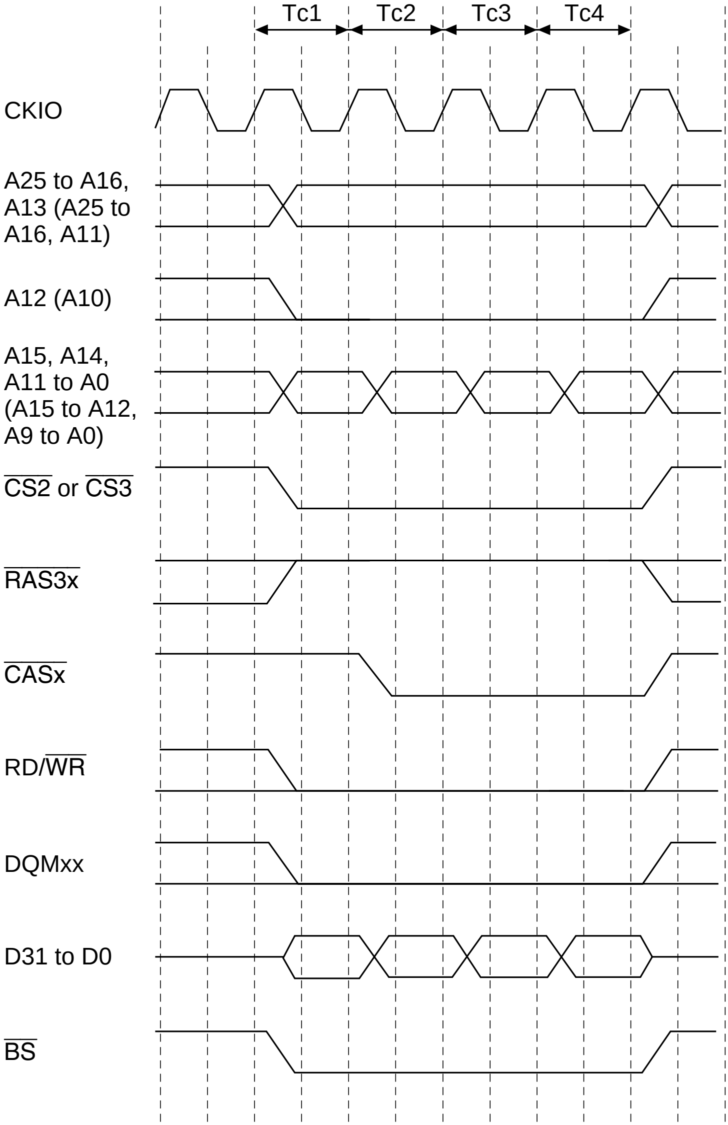

Tc1
Tc2
Tc3
Tc4
CKIO
A25 to A16, A13 (A25 to A16, A11)
A12 (A10)
A15, A14, A11 to A0 (A15 to A12, A9 to A0)
$\overline{\text{CS2}}$ or $\overline{\text{CS3}}$
$\overline{\text{RAS3x}}$
$\overline{\text{CASx}}$
RD/$\overline{\text{WR}}$
DQMxx
D31 to D0
$\overline{\text{BS}}$

Figure 13.23 Burst Write Timing (Same Row Address)

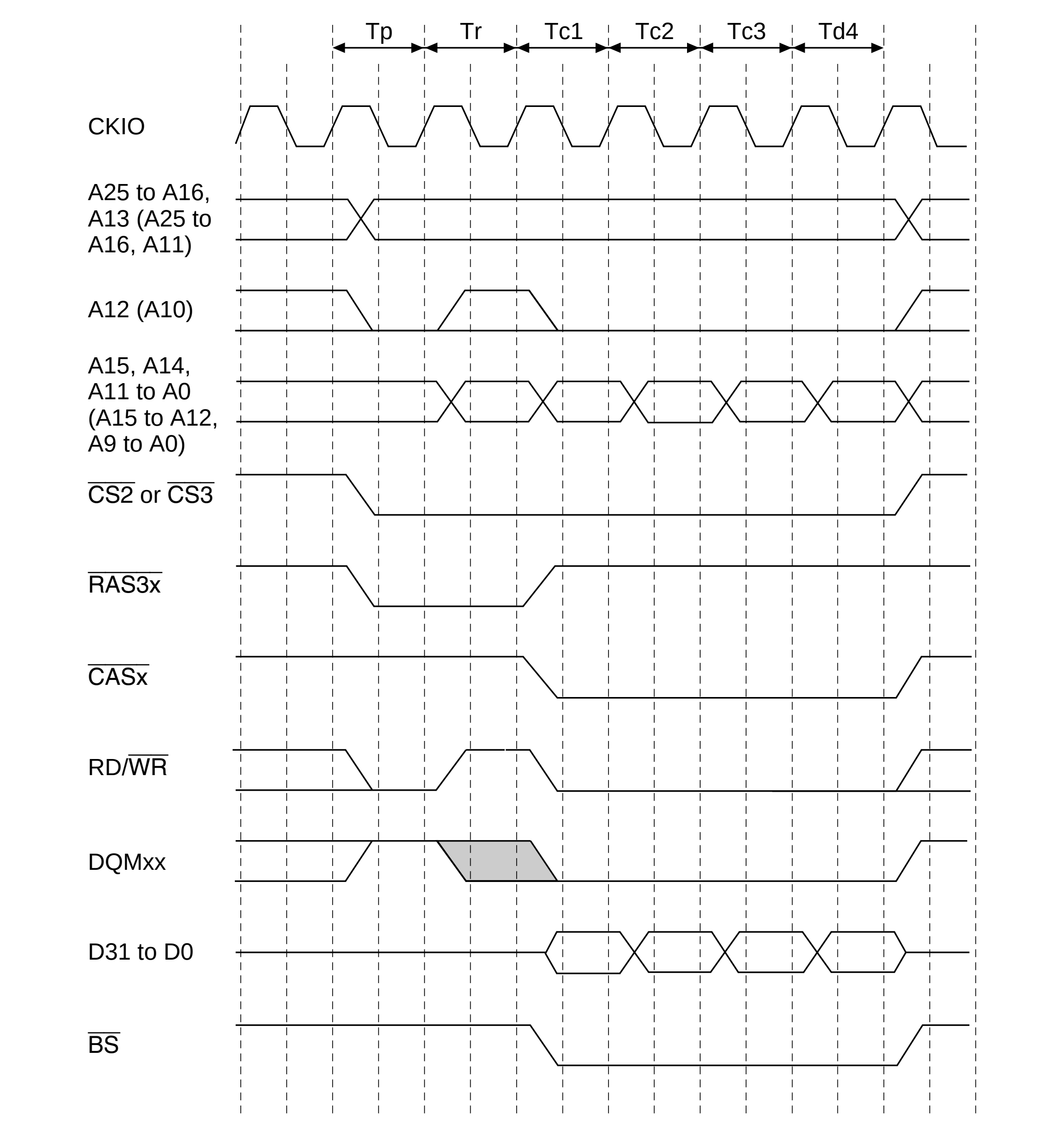

Tp
Tr
Tc1
Tc2
Tc3
Td4
CKIO
A25 to A16, A13 (A25 to A16, A11)
A12 (A10)
A15, A14, A11 to A0 (A15 to A12, A9 to A0)
CS2 or CS3
RAS3x
CASx
RD/WR
DQMxx
D31 to D0
BS

Figure 13.24 Burst Write Timing (Different Row Addresses)

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.

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 stipulation 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. 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 13.26 shows the auto-refresh cycle timing.

   All-bank precharging is performed in the Tp cycle, then an REF command is issued in the TRr cycle following the interval specified by the TPC bits in MCR. After the TRr cycle, new command output cannot be performed for the duration of the number of cycles specified by the TRAS bits in MCR plus the number of cycles specified by the TPC bits in MCR. The TRAS and TPC bits must be set so as to satisfy the synchronous DRAM refresh cycle time stipulation (active/active command delay time).

   Auto-refreshing is performed in normal operation, in sleep mode, and in case of a manual reset.

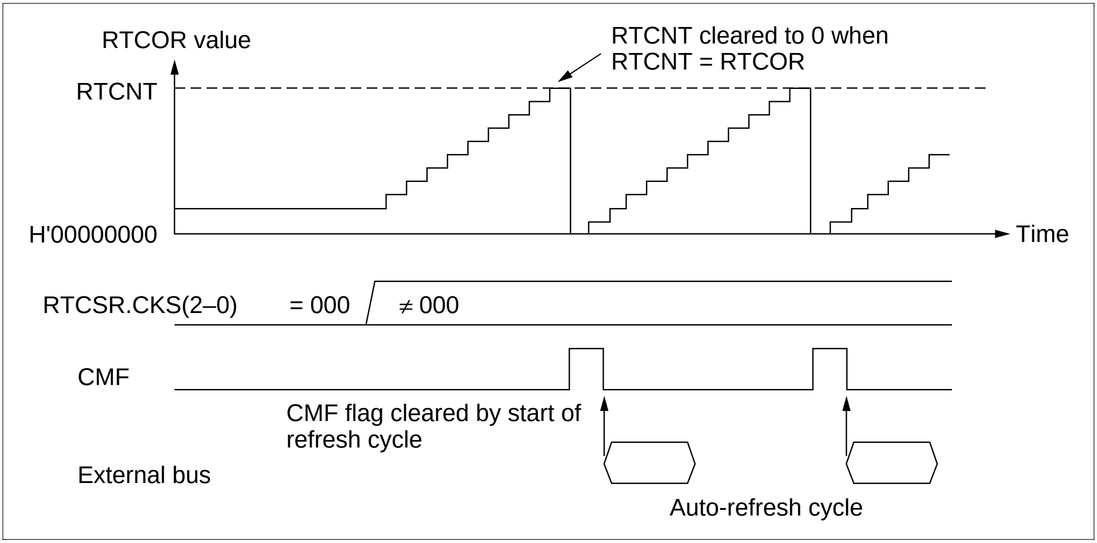


Figure 13.25 Auto-Refresh Operation

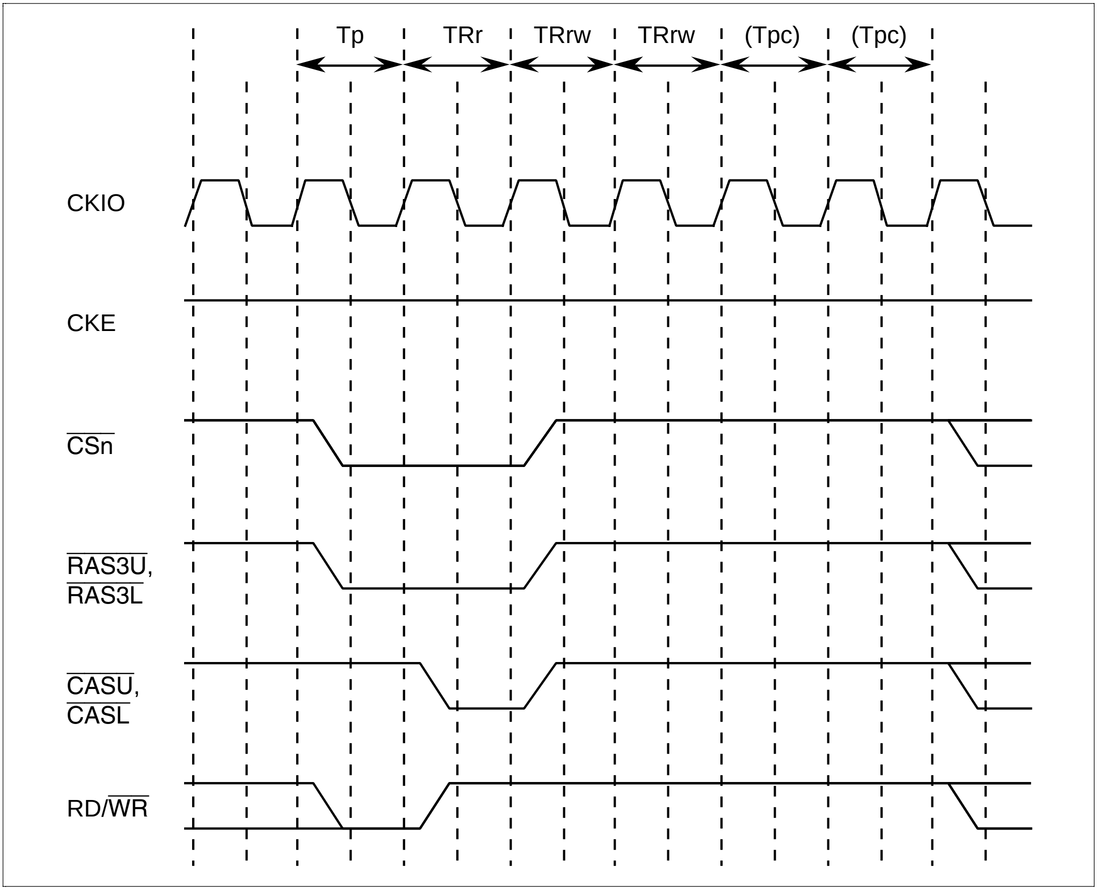

Tp
TRr
TRrw
TRrw
(Tpc)
(Tpc)
CKIO
CKE
$\overline{\text{CSn}}$
$\overline{\text{RAS3U}}$,
$\overline{\text{RAS3L}}$
$\overline{\text{CASU}}$,
$\overline{\text{CASL}}$
RD/$\overline{\text{WR}}$

Figure 13.26 Synchronous DRAM Auto-Refresh Timing

2. Self-Refreshing

Self-refresh mode is a kind of standby mode in which the refresh timing and refresh addresses are generated within the 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 the TPC bits in MCR. Self-refresh timing is shown in figure 13.27. 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 SH7622’s standby function, and is maintained even after recovery from standby mode other than through a power-on reset. In case of a power-on reset, the bus state controller’s registers are initialized, and therefore the self-refresh state is cleared.

Self-refreshing is performed in normal operation, in sleep mode, in standby mode, and in case of a manual reset. When using SDRAM, the following procedure should be used to start self-refreshing.

(1) Clear the refresh control bit to 0.

(2) Write H'00 to the RTCNT register.

(3) Set the refresh control bit and refresh mode bit to 1.

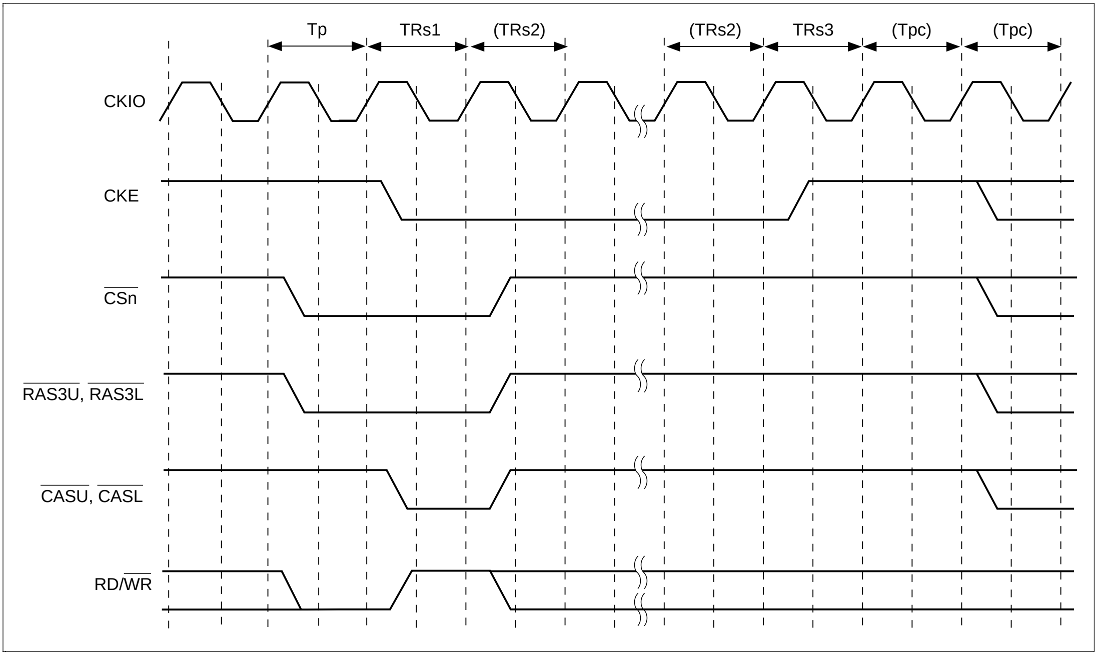


Figure 13.27 Synchronous DRAM Self-Refresh Timing

3. Relationship between Refresh Requests and Bus Cycle Requests

   If a refresh request is generated during execution of a bus cycle, execution of the refresh is deferred until the bus cycle is completed. If a refresh request occurs when the bus has been released by the bus arbiter, refresh execution is deferred until the bus is acquired. If a match between RTCNT and RTCOR occurs while a refresh is waiting to be executed, so that a new refresh request is generated, the previous refresh request is eliminated. In order for refreshing to be performed normally, care must be taken to ensure that no bus cycle or bus mastership occurs that is longer than the refresh interval. When a refresh request is generated, the IRQOUT pin is asserted (driven low). Therefore, normal refreshing can be performed by having the $\overline{\text{IRQOUT}}$ pin monitored by a bus master other than the SH7622 requesting the bus, or the bus arbiter, and returning the bus to the SH7622. When refreshing is started, and if no other interrupt request has been generated, the $\overline{\text{IRQOUT}}$ pin is negated (driven high).

Power-On Sequence: In order to use 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'FFFFD000 + X for area 2 synchronous DRAM, and to address H'FFFFE000 + 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/single write, CAS latency 1 to 3, wrap type = sequential, and burst length 1 supported by the SH7622, arbitrary data is written in a byte-size access to the following addresses.

		Area 2	**Area 3**
32-bit bus width	CAS latency 1	FFFFD840	FFFFE840
32-bit bus width	CAS latency 2	FFFFD880	FFFFE880
32-bit bus width	CAS latency 3	FFFFD8C0	FFFFE8C0
16-bit bus width	CAS latency 1	FFFFD420	FFFFE420
16-bit bus width	CAS latency 2	FFFFD440	FFFFE440
16-bit bus width	CAS latency 3	FFFFD460	FFFFE460

Mode register setting timing is shown in figure 13.28.

As a result of the write to address H'FFFFD000 + X or H'FFFFE000 + X, a precharge all banks (PALL) command is first issued in the TRp1 cycle, then a mode register write command is issued in the TMw1 cycle.

Address signals, when the mode-register write command is issued, are as follows:

1. 32-bit width connection	A15–A9	0000100 (burst read and single write)
	A8–A6	CAS latency
	A5	0 (burst type = sequential)
	A4–A2	000 (burst length 1)
2. 16-bit width connection	A14–A8	0000100 (burst read and single write)
	A7–A5	CAS latency
	A4	0 (burst type = sequential)
	A3–A1	000 (burst length 1)

Before mode register setting, a 100 µs idle time (depending on the memory manufacturer) must be guaranteed after powering on requested by the synchronous DRAM. If the reset signal pulse width is greater than this idle time, there is no problem in performing mode register setting immediately. The number of dummy auto-refresh cycles specified by the manufacturer (usually 8) or more must be executed. This is usually achieved automatically while various kinds of initialization are being performed after auto-refresh setting, but a way of carrying this out more dependably is to set a short refresh request generation interval just while these dummy cycles are being executed. With simple read or write access, the address counter in the synchronous DRAM used for auto-refreshing is not initialized, and so the cycle must always be an auto-refresh cycle.

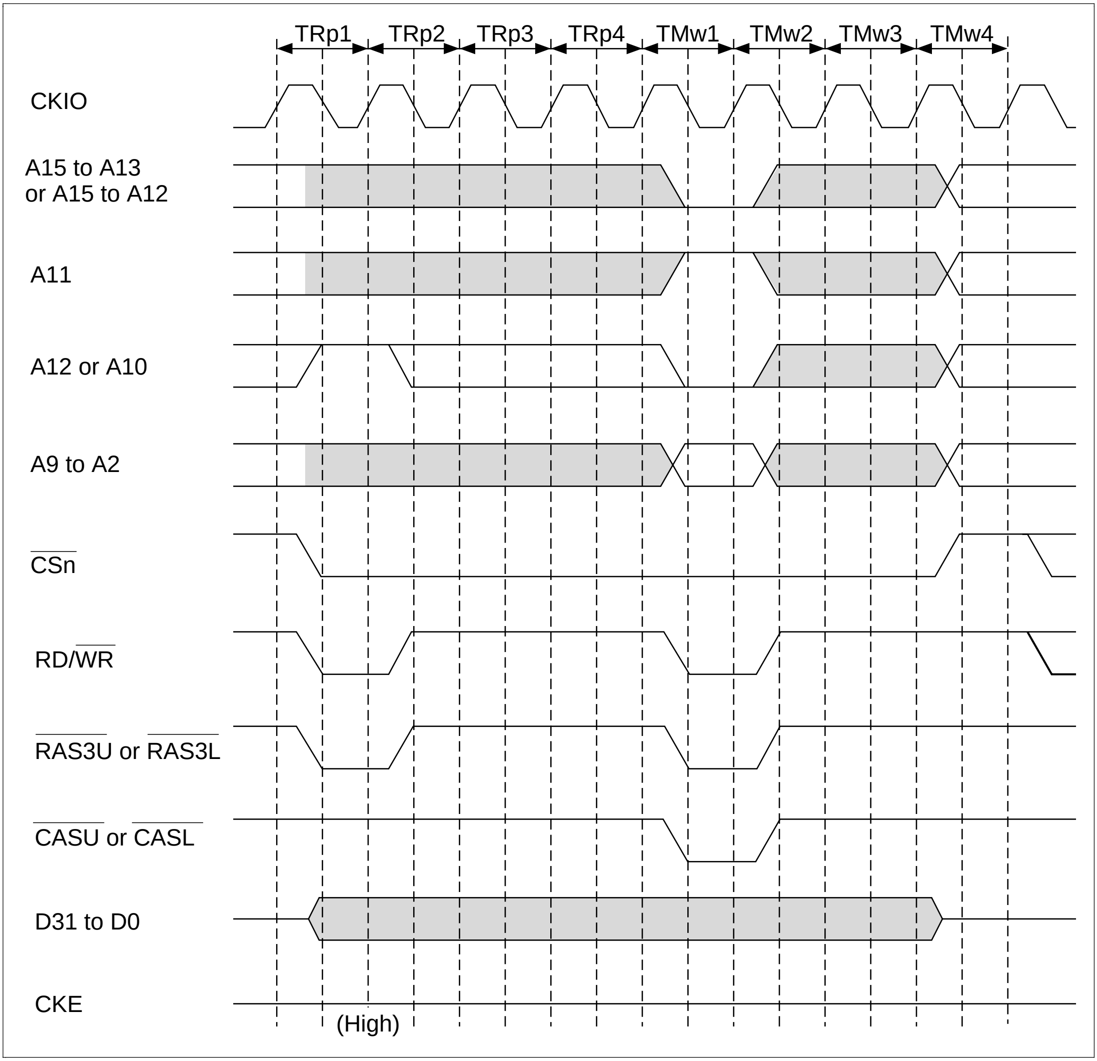


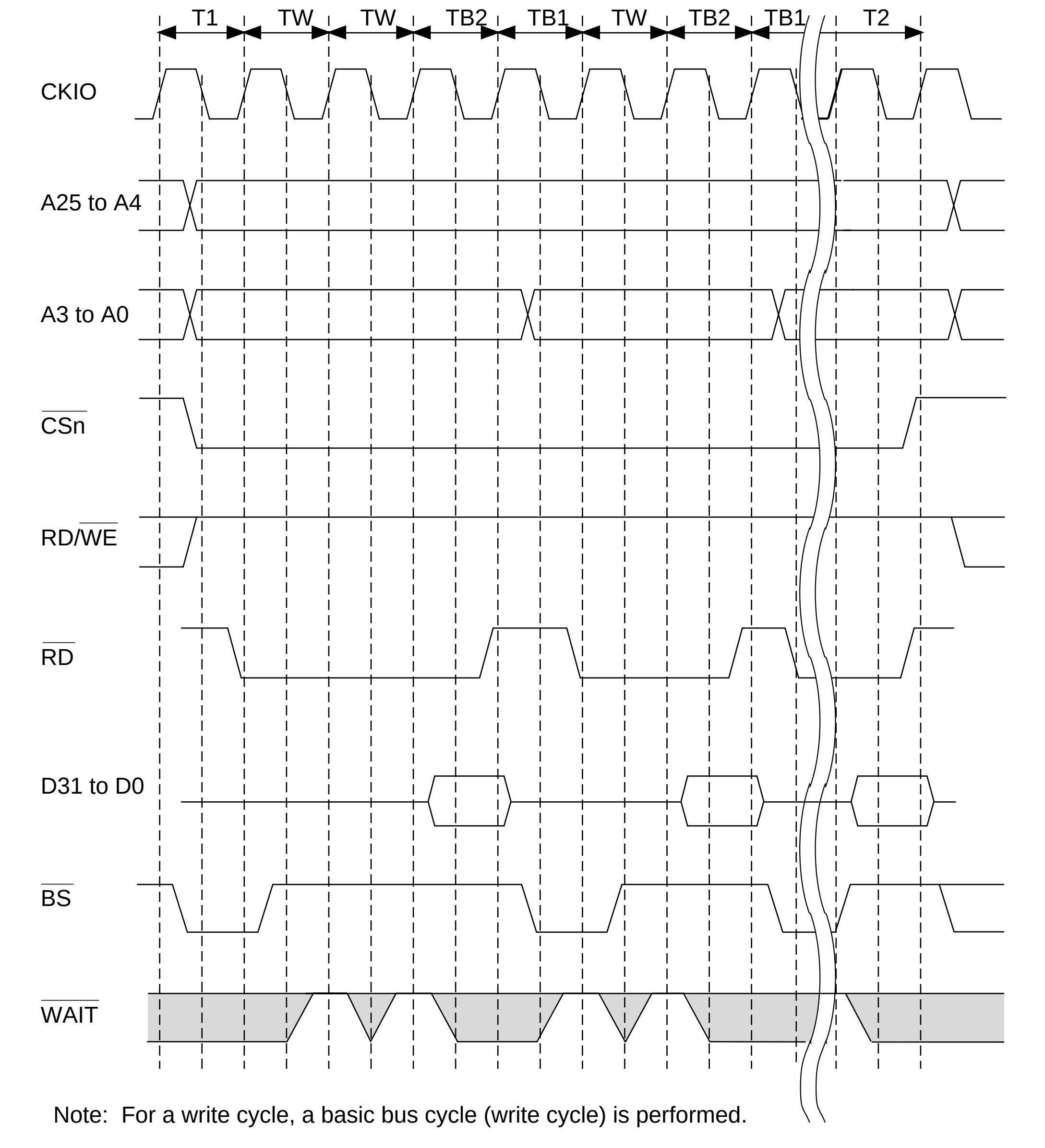
Figure 13.28   Synchronous DRAM Mode Write Timing

13.3.5 Burst ROM Interface

Setting bits A0BST 1–0, A5BST 1–0, and A6BST 1–0 in BCR1 to a non-zero value allows burst ROM to be connected to areas 0, 5, and 6. The burst ROM interface provides high-speed access to ROM that has a nibble access function. The timing for nibble access to burst ROM is shown in figure 13.29. Two wait cycles are set. Basically, access is performed in the same way as for normal space, but when the first cycle ends the $\overline{\text{CS0}}$ signal is not negated, and only the address is changed before the next access is executed. When 8-bit ROM is connected, the number of consecutive accesses can be set as 4, 8, or 16 by bits A0BST 1–0, A5BST 1–0, or A6BST 1–0. When 16-bit ROM is connected, 4 or 8 can be set in the same way. When 32-bit ROM is connected, only 4 can be set.

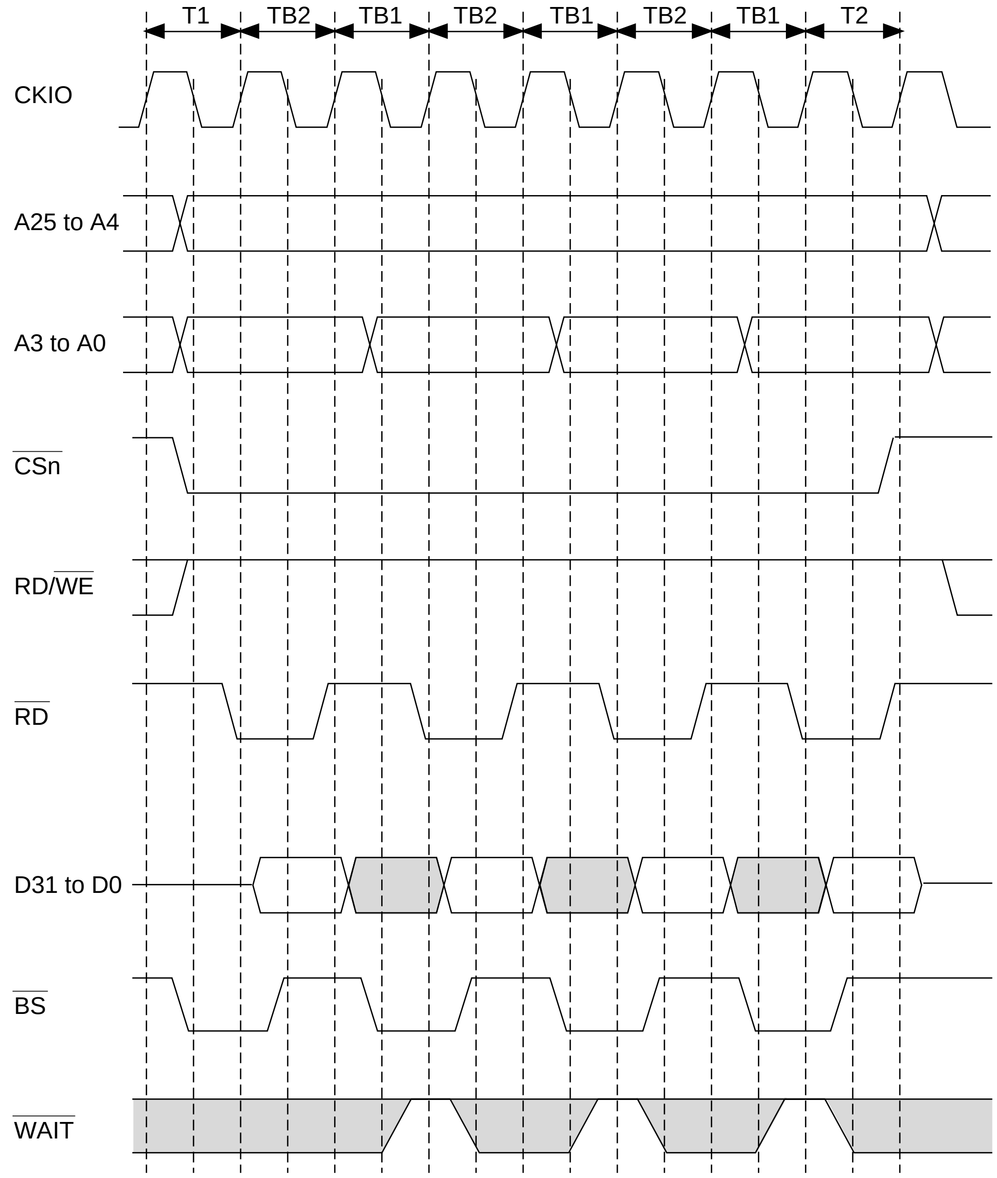
$\overline{\text{WAIT}}$ pin sampling is performed in the first access if one or more wait states are set, and is always performed in the second and subsequent accesses.

The second and subsequent access cycles also comprise two cycles when a burst ROM setting is made and the wait specification is 0. The timing in this case is shown in figure 13.30.



Note: For a write cycle, a basic bus cycle (write cycle) is performed.

Figure 13.29 Burst ROM Wait Access Timing



Note: For a write cycle, a basic bus cycle (write cycle) is performed.

Figure 13.30 Burst ROM Basic Access Timing

13.3.6 Waits between Access Cycles

A problem associated with higher external memory bus operating frequencies is that data buffer turn-off on completion of a read from a low-speed device may be too slow, causing a collision with data in the next access. This results in lower reliability or incorrect operation. To avoid this problem, a data collision prevention feature has been provided. This memorizes the preceding access area and the kind of read/write. If there is a possibility of a bus collision when the next access is started, a wait cycle is inserted before the access cycle thus preventing a data collision. There are two cases in which a wait cycle is inserted: when an access is followed by an access to a different area, and when a read access is followed by a write access from the SH7622. When the SH7622 performs consecutive write cycles, the data transfer direction is fixed (from the SH7622 to other memory) and there is no problem. With read accesses to the same area, in principle, data is output from the same data buffer, and wait cycle insertion is not performed. Bits AnIW1 and AnIW0 (n = 0, 2 to 6) in WCR1 specify the number of idle cycles to be inserted between access cycles when a physical space area access is followed by an access to another area, or when the SH7622 performs a write access after a read access to physical space area n. If there is originally space between accesses, the number of idle cycles inserted is the specified number of idle cycles minus the number of empty cycles.

Waits are not inserted between accesses when bus arbitration is performed, since empty cycles are inserted for arbitration purposes.

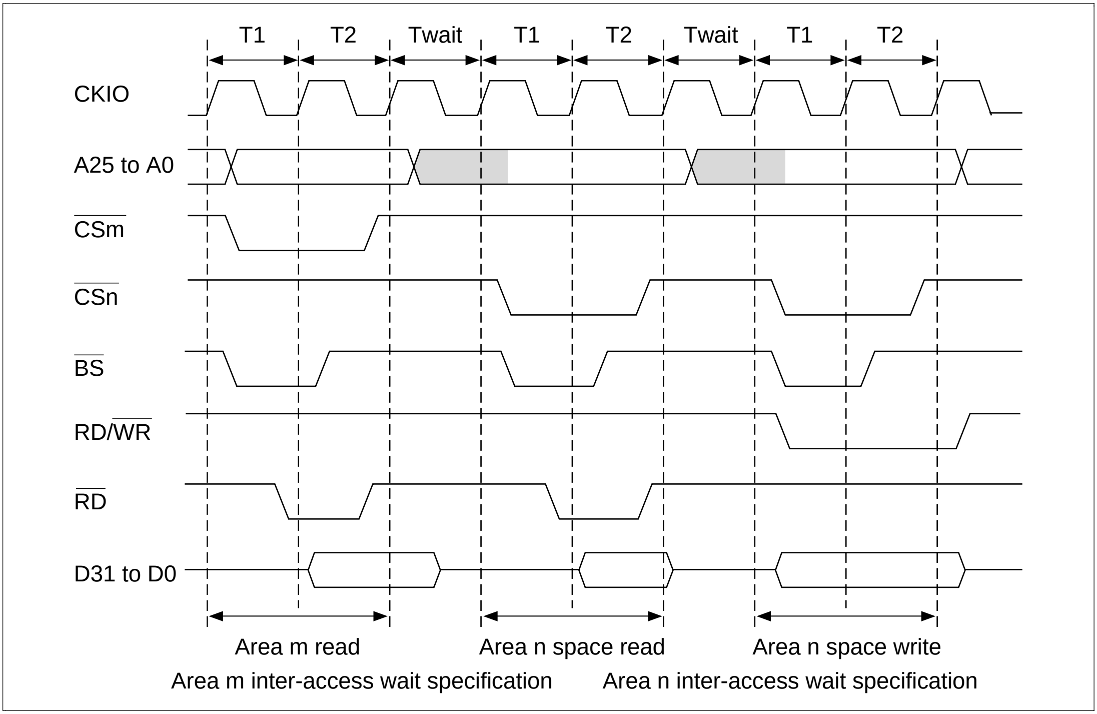


Figure 13.31 Waits between Access Cycles

13.3.7 Bus Arbitration

When a bus release request ($\overline{\text{BREQ}}$) is received from an external device, buses are released after the bus cycle being executed is completed and a bus grant signal ($\overline{\text{BACK}}$) is output. The bus is not released during burst transfers for cache fills or TAS instruction execution between the read cycle and write cycle. Bus arbitration is not executed in multiple bus cycles that are generated when the data bus width is shorter than the access size; i.e. in the bus cycles when longword access is executed for the 8-bit memory. $\overline{\text{BREQ}}$ must be kept asserted until the start of $\overline{\text{BACK}}$ assertion. At the negation of $\overline{\text{BREQ}}$, $\overline{\text{BACK}}$ is negated and bus use is restarted. See Appendix B.1, Pin States, for the pin state when the bus is released.

The SH7622 sometimes needs to retrieve a bus it has released. For example, when memory generates a refresh request or an interrupt request internally, the SH7622 must perform the appropriate processing. The SH7622 has a bus request signal ($\overline{\text{IRQOUT}}$) for this purpose. When it must retrieve the bus, it asserts the $\overline{\text{IRQOUT}}$ signal. Devices asserting an external bus release request receive the assertion of the $\overline{\text{IRQOUT}}$ signal and negate the $\overline{\text{BREQ}}$ signal to release the bus. The SH7622 retrieves the bus and carries out the processing.

- $\overline{\text{IRQOUT}}$ pin assertion conditions
  - When a memory refresh request is generated and refresh cycle has not yet started
  - When an interrupt source is generated and the interrupt request level is higher than the setting of the interrupt mask bits (I3–I0) in the status register (SR)

Section 14 Direct Memory Access Controller (DMAC)

14.1 Overview

This chip includes a four-channel direct memory access controller (DMAC). The DMAC can be used in place of the CPU to perform high-speed transfers between external devices that have DACK (transfer request acknowledge signal), external memory, memory-mapped external devices, X/Y memory, and on-chip peripheral modules (SCIF0/1/2, USB, CMT1, and A/D converter). Using the DMAC reduces the burden on the CPU and increases overall operating efficiency.

14.1.1 Features

The DMAC has the following features.

- Four channels
- 4-GB physical address space
- 8-bit, 16-bit, 32-bit, or 16-byte transfer (In 16-byte transfer, four 32-bit reads are executed, followed by four 32-bit writes.)
- 16 Mbytes (16777216 transfers)
- Address mode: Dual address mode and single address mode are supported. In addition, direct address transfer mode or indirect address transfer mode can be selected.
  - Dual address mode transfer: Both the transfer source and transfer destination are accessed by address. Dual address mode has direct address transfer mode and indirect address transfer mode.

    Direct address transfer mode: The values specified in the DMAC registers indicates the transfer source and transfer destination. 2 bus cycles are required for one data transfer.

    Indirect address transfer mode: Data is transferred with the address stored prior to the address specified in the transfer source address in the DMAC. Other operations are the same as those of direct address transfer mode. This function is only valid in channel 3. 4 bus cycles are requested for one data transfer.
  - Single address mode transfer: Either the transfer source or transfer destination peripheral device is accessed (selected) by means of the DACK signal, and the other device is accessed by address. One transfer unit of data is transferred in 1 bus cycle.
- Channel functions: Transfer mode that can be specified is different in each channel.
  - Channel 0: External request can be accepted.
  - Channel 1: External request can be accepted.
  - Channel 2: This channel has a source address reload function, which reloads a source address for each four transfers.

  — Channel 3: In this channel, direct address mode or indirect address transfer mode can be specified.

- Reload function: The value that was specified in the source address register can be automatically reloaded every four DMA transfers. This function is only valid in channel 2.

- Transfer requests
  — External request (From two $\overline{\text{DREQ}}$ pins (channels 0 and 1 only). $\overline{\text{DREQ}}$ can be detected either by edge or by level)
  — On-chip module request (This request can be accepted in all the channels)
  — Auto request (The transfer request is generated automatically within the DMAC)
- Selectable bus modes: Cycle-steal mode or burst mode
- Selectable channel priority levels:

  Fixed mode: The channel priority is fixed.

  Round-robin mode: The priority of the channel in which the execution request was accepted is made the lowest.
- Interrupt request: An interrupt request can be generated to the CPU after transfers end by the specified counts.

14.1.2 Block Diagram

Figure 14.1 is a block diagram of the DMAC.

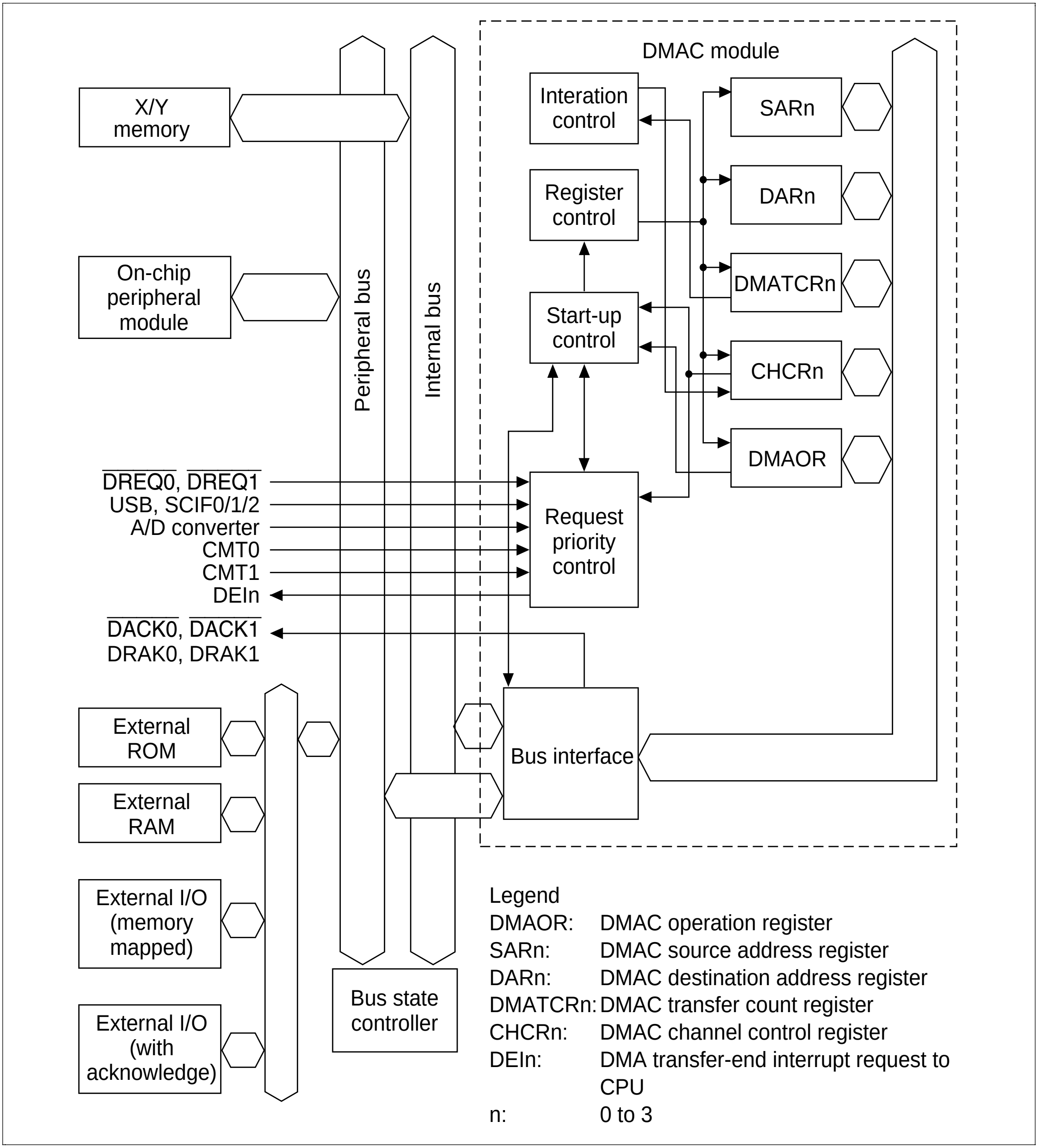

DMAC module
X/Y memory
On-chip peripheral module
Peripheral bus
Internal bus
Interation control
Register control
Start-up control
Request priority control
Bus interface
SARn
DARn
DMATCRn
CHCRn
DMAOR
$\overline{\text{DREQ0}}$, $\overline{\text{DREQ1}}$
USB, SCIF0/1/2
A/D converter
CMT0
CMT1
DEIn
$\overline{\text{DACK0}}$, $\overline{\text{DACK1}}$
DRAK0, DRAK1
External ROM
External RAM
External I/O (memory mapped)
External I/O (with acknowledge)
Bus state controller
Legend
DMAOR: DMAC operation register
SARn: DMAC source address register
DARn: DMAC destination address register
DMATCRn: DMAC transfer count register
CHCRn: DMAC channel control register
DEIn: DMA transfer-end interrupt request to CPU
n: 0 to 3

Figure 14.1 DMAC Block Diagram

14.1.3 Pin Configuration

Table 14.1 shows the DMAC pins.

Table 14.1 Pin Configuration

Channel	Name	Symbol	I/O	Function
0	DMA transfer request	$\overline{\text{DREQ0}}$	I	DMA transfer request input from external device to channel 0
	DREQ acknowledge	DACK0	O	Strobe output to an external I/O at DMA transfer request from external device to channel 0
	DMA request acknowledge	DRAK0	O	Output showing that DREQ0 has been accepted
1	DMA transfer request	$\overline{\text{DREQ1}}$	I	DMA transfer request input from external device to channel 1
	DREQ acknowledge	DACK1	O	Strobe output to an external I/O at DMA transfer request from external device to channel 1
	DMA request acknowledge	DRAK1	O	Output showing that DREQ1 has been accepted

14.1.4 Register Configuration

Table 14.2 summarizes the DMAC registers. The DMAC has four registers assigned to each of the channels, a single register for overall DMAC control, and two expansion registers for the modules added to the SH7622.

Table 14.2 DMAC Registers

Channel	Name	Abbreviation	R/W	Initial Value	Address	Register Size	Access Size
0	DMA source address register 0	SAR0	R/W	Undefined	H'A4000020	32 bits	16, 32*[2]
	DMA destination address register 0	DAR0	R/W	Undefined	H'A4000024	32 bits	16, 32*[2]
	DMA transfer count register 0	DMATCR0	R/W	Undefined	H'A4000028	24 bits	16, 32*[3]
	DMA channel control register 0	CHCR0	R/W*[1]	H'00000000	H'A400002C	32 bits	8, 16, 32*[2]
1	DMA source address register 1	SAR1	R/W	Undefined	H'A4000030	32 bits	16, 32*[2]
	DMA destination address register 1	DAR1	R/W	Undefined	H'A4000034	32 bits	16, 32*[2]
	DMA transfer count register 1	DMATCR1	R/W	Undefined	H'A4000038	24 bits	16, 32*[3]
	DMA channel control register 1	CHCR1	R/W*[1]	H'00000000	H'A400003C	32 bits	8, 16, 32*[2]
2	DMA source address register 2	SAR2	R/W	Undefined	H'A4000040	32 bits	16, 32*[2]
	DMA destination address register 2	DAR2	R/W	Undefined	H'A4000044	32 bits	16, 32*[2]
	DMA transfer count register 2	DMATCR2	R/W	Undefined	H'A4000048	24 bits	16, 32*[3]
	DMA channel control register 2	CHCR2	R/W*[1]	H'00000000	H'A400004C	32 bits	8, 16, 32*[2]
3	DMA source address register 3	SAR3	R/W	Undefined	H'A4000050	32 bits	16, 32*[2]
	DMA destination address register 3	DAR3	R/W	Undefined	H'A4000054	32 bits	16, 32*[2]
	DMA transfer count register 3	DMATCR3	R/W	Undefined	H'A4000058	24 bits	16, 32*[3]
	DMA channel control register 3	CHCR3	R/W*[1]	H'00000000	H'A400005C	32 bits	8, 16, 32*[2]
Shared	DMA operation register	DMAOR	R/W*[1]	H'0000	H'A4000060	16 bits	8, 16*[2]
0/1	DMA channel expansion request register 0	CHCRA0	R/W	H'0000	H'A4000900	16 bits	16
2/3	DMA channel expansion request register 1	CHCRA1	R/W	H'0000	H'A4000902	16 bits	16

Notes: *1 Only 0s can be written to bits 1 of CHCR0 to CHCR3, and bits 1 and 2 of DMAOR to clear flag after 1 is read.

*2 If SAR0 to SAR3, DAR0 to DAR3, and CHCR0 to CHCR3 are accessed in 16 bits, the value in 16 bits that were not accessed are held.

*3 DMATCR comprises the 24 bits from bit 0 to bit 23. The upper 8 bits, bits 24 to 31, cannot be written with 1 and are always read as 0.

14.2 Register Descriptions

14.2.1 DMA Source Address Registers 0–3 (SAR0–SAR3)

DMA source address registers 0–3 (SAR0–SAR3) are 32-bit read/write registers that specify the source address of a DMA transfer. During a DMA transfer, these registers indicate the next source address.

To transfer data in 16 bits or in 32 bits, specify the address with 16-bit or 32-bit address boundary. When transferring data in 16-byte units, a 16-byte boundary (address 16n) must be set for the source address value. Specifying other addresses does not guarantee operation.

The initial value is undefined by resets. The previous value is held in standby mode.

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

Bit:	23	22	21	20	…	0
					…	
Initial value:	—	—	—	—	…	—
R/W:	R/W	R/W	R/W	R/W	…	R/W

14.2.2 DMA Destination Address Registers 0–3 (DAR0–DAR3)

DMA destination address registers 0–3 (DAR0–DAR3) are 32-bit read/write registers that specify the destination address of a DMA transfer. These registers include count functions, and during a DMA transfer, these registers indicate the next destination address.

To transfer data in 16 bits or in 32 bits, specify the address with 16-bit or 32-bit address boundary. Specifying other addresses does not guarantee operation.

The initial value is undefined by resets. The previous value is held in standby mode.

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

Bit:	23	22	21	20	…	0
					…	
Initial value:	—	—	—	—	…	—
R/W:	R/W	R/W	R/W	R/W	…	R/W

14.2.3 DMA Transfer Count Registers 0–3 (DMATCR0–DMATCR3)

DMA transfer count registers 0–3 (DMATCR0–DMATCR3) are 24-bit read/write registers that specify the DMA transfer count (bytes, words, or longwords). The number of transfers is 1 when the setting is H'000001, and 16777216 (the maximum) when H'000000 is set. During a DMA transfer, these registers indicate the remaining transfer count.

To transfer data in 16 bytes, one 16-byte transfer (128 bits) counts one. The upper 8 bits of DMATCR will return 0 if read, and should only be written with 0.

The initial value is undefined by resets. The previous value is held in standby mode.

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

Bit:	23	22	21	20	...	0
					...	
Initial value:	—	—	—	—	...	—
R/W:	R/W	R/W	R/W	R/W	...	R/W

14.2.4 DMA Channel Control Registers 0–3 (CHCR0–CHCR3)

DMA channel control registers 0–3 (CHCR0–CHCR3) are 32-bit read/write registers that specifies operation mode, transfer method, or others in each channel. Writing to bits 31–21 and 7 in this register is invalid; 0 s are read if these bits are read.

Bit 20 is only used in CHCR3. It is not used in CHCR0–CHCR2. Consequently, writing to this bit is invalid in CHCR0–CHCR2; 0 is read if this bit is read. Bit 19 is only used in CHCR2; it is not used in CHCR0, CHCR1, and CHCR3. Consequently, writing to this bit is invalid in CHCR0, CHCR1, and CHCR3; 0 is read if this bit is read. Bits 6 and 16–18 are only used in CHCR0 and CHCR1; they are not used in CHCR2 and CHCR3. Consequently, writing to these bits is invalid in CHCR2 and CHCR3; 0s are read if these bits are read.

These register values are initialized to 0s after power-on resets. The previous value is held in standby mode.

Bit:	31	...	21	20	19	18	17	16
	—	...	—	DI	RO	RL	AM	AL
Initial value:	0	...	0	0	0	0	0	0
R/W:	R	...	R	(R/W)*2	(R/W)*2	(R/W)*2	(R/W)*2	(R/W)*2

Bit:	15	14	13	12	11	10	9	8
	DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	—	DS	TM	TS1	TS0	IE	TE	DE
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	(R/W)*2	R/W	R/W	R/W	R/W	R/(W)*1	R/W

Notes: *1 Only 0 can be written to the TE bit after 1 is read.
*2 DI, RO, RL, AM, AL, and DS bits are not included in some channels.

Bits 31 to 21 and 7—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 20—Direct/Indirect Selection (DI): DI selects direct address mode or indirect address mode in channel 3.

This bit is only valid in CHCR3. Writing to this bit is invalid in CHCR0–CHCR2; 0 is read if this bit is read. When using 16-byte transfer, direct address mode must be specified. Operation is not guaranteed if indirect address mode is specified.

Bit 20: DI	Description	
0	Direct address mode	(Initial value)
1	Indirect address mode	

Bit 19—Source Address Reload Bit (RO): RO selects whether the source address initial value is reloaded in channel 2.

This bit is only valid in CHCR2. Writing to this bit is invalid in CHCR0, CHCR1, and CHCR3; 0 is read if this bit is read. When using 16-byte transfer, this bit must be cleared to 0, specifying non-reloading. Operation is not guaranteed if reloading is specified.

Bit 19: RO	Description	
0	A source address is not reloaded	(Initial value)
1	A source address is reloaded	

Bit 18—Request Check Level Bit (RL): RL specifies the DRAK (acknowledge of $\overline{\text{DREQ}}$) signal output is high active or low active.

This bit is only valid in CHCR0 and CHCR1. Writing to this bit is invalid in CHCR2 and CHCR3; 0 is read if this bit is read.

Bit 18: RL	Description	
0	Low-active output of DRAK	(Initial value)
1	High-active output of DRAK	

Bit 17—Acknowledge Mode Bit (AM): AM specifies whether DACK is output in data read cycle or in data write cycle in dual address mode.

This bit is only valid in CHCR0 and CHCR1. Writing to this bit is invalid in CHCR2 and CHCR3; 0 is read if this bit is read.

Bit 17: AM	Description	
0	DACK output in read cycle	(Initial value)
1	DACK output in write cycle	

Bit 16—Acknowledge Level (AL): AL specifies the DACK (acknowledge) signal output is high active or low active.

This bit is only valid in CHCR0 and CHCR1. Writing to this bit is invalid in CHCR2 and CHCR3; 0 is read if this bit is read.

Bit 16: AL	Description	
0	Low-active output of DACK	(Initial value)
1	High-active output of DACK	

Bits 15 and 14—Destination Address Mode Bits 1 and 0 (DM1, DM0): DM1 and DM0 select whether the DMA destination address is incremented, decremented, or left fixed.

Bit 15: DM1	Bit 14: DM0	Description	
0	0	Fixed destination address*	(Initial value)
0	1	Destination address is incremented (+1 in 8-bit transfer, +2 in 16-bit transfer, +4 in 32-bit transfer, +16 in 16-byte transfer)	
1	0	Destination address is decremented (–1 in 8-bit transfer, –2 in 16-bit transfer, –4 in 32-bit transfer; illegal setting in 16-byte transfer)	
1	1	Illegal setting	

Note: * This setting cannot be used when the transfer destination is X/Y memory and the transfer size is 16 bytes.

Bits 13 and 12—Source Address Mode Bits 1 and 0 (SM1, SM0): SM1 and SM0 select whether the DMA source address is incremented, decremented, or left fixed.

Bit 13: SM1	Bit 12: SM0	Description	
0	0	Fixed source address*	(Initial value)
0	1	Source address is incremented (+1 in 8-bit transfer, +2 in 16-bit transfer, +4 in 32-bit transfer, +16 in 16-byte transfer)	
1	0	Source address is decremented (–1 in 8-bit transfer, –2 in 16-bit transfer, –4 in 32-bit transfer; illegal setting in 16-byte transfer)	
1	1	Illegal setting	

Note: * This setting cannot be used when the transfer destination is X/Y memory and the transfer size is 16 bytes.

If the transfer source is specified in indirect address, specify the address, in which the data to be transferred is stored and which is stored as data (indirect address), in source address register 3 (SAR3).

Specification of SAR3 increment or decrement in indirect address mode depends on SM1 and SM0 settings. In this case, however, the SAR3 increment or decrement value is +4, –4, or fixed to 0 regardless of the transfer data size specified in TS1 and TS0.

Bits 11 to 8—Resource Select Bits 3–0 (RS3–RS0): RS3–RS0 specify which transfer requests will be sent to the DMAC.

Bit 11: RS3	Bit 10: RS2	Bit 9: RS1	Bit 8: RS0	Description
0	0	0	0	External request, dual address mode (Initial value)
0	0	0	1	Illegal setting
0	0	1	0	External request / Single address mode External address space → external device with DACK
0	0	1	1	External request / Single address mode External device with DACK → external address space
0	1	0	0	Auto request
0	1	0	1	Illegal setting
0	1	1	0	Illegal setting
0	1	1	1	Illegal setting
1	0	0	0	DMA expansion request module selection specification
1	0	0	1	Illegal setting
1	0	1	0	SCIF0 transmission
1	0	1	1	SCIF0 reception
1	1	0	0	SCIF1 transmission
1	1	0	1	SCIF1 reception
1	1	1	0	A/D converter
1	1	1	1	CMT0

Notes: 1. External request specification is valid only in channels 0 and 1. None of the request sources can be selected in channels 2 and 3.

2. When using 16-byte transfer, the following settings must not be made:

1000	SCIF2 transmission/reception among DMA expansion request modules
1110	A/D converter

Operation is not guaranteed if these settings are made.

Bit 6—$\overline{\text{DREQ}}$ Select Bit (DS): DS selects the sampling method of the $\overline{\text{DREQ}}$ pin that is used in external request mode is detection in low level or at the falling edge.

This bit is only valid in CHCR0 and CHCR1. Writing to this bit is invalid in CHCR2 and CHCR3; 0 is read if this bit is read.

In channels 0 and 1, also, if the transfer request source is specified as an on-chip peripheral module set in CHCR0 to CHCR3, or if an auto-request is specified, the specification by this bit is ignored and, except in the case of an auto-request, falling edge detection is fixed. For on-chip peripheral modules set with CHCRA0/1 (USB, SCIF2, CMT1), the specification by this bit is ignored and low-level detection is used.

Bit 6: DS	Description	
0	$\overline{\text{DREQ}}$ detected in low level	(Initial value)
1	$\overline{\text{DREQ}}$ detected at falling edge	

Bit 5—Transmit Mode (TM): TM specifies the bus mode when transferring data.

Bit 5: TM	Description	
0	Cycle steal mode	(Initial value)
1	Burst mode	

Bits 4 and 3—Transmit Size Bits 1 and 0 (TS1, TS0): TS1 and TS0 specify the size of data to be transferred.

Bit 4: TS1	Bit 3: TS0	Description	
0	0	Byte size (8 bits)	(Initial value)
0	1	Word size (16 bits)	
1	0	Longword size (32 bits)	
1	1	16-byte unit (4 longword transfers)	

Bit 2—Interrupt Enable Bit (IE): Setting this bit to 1 generates an interrupt request when data transfer end (TE = 1) by the count specified in DMATCR.

Bit 2: IE	Description	
0	Interrupt request is not generated even if data transfer ends by the specified count	(Initial value)
1	Interrupt request is generated if data transfer ends by the specified count	

Bit 1—Transfer End Bit (TE): TE is set to 1 when data transfer ends by the count specified in DMATCR. At this time, if the IE bit is set to 1, an interrupt request is generated.

Before this bit is set to 1, if data transfer ends due to an NMI interrupt, a DMAC address error, or clearing the DE bit or the DME bit in DMAOR, this bit is not set to 1. Even if the DE bit is set to 1 while this bit is set to 1, transfer is not enabled.

Bit 1: TE	Description
0	Data transfer does not end by the count specified in DMATCR (Initial value) [Clearing condition] • Writing 0 after TE = 1 read at power-on reset or manual reset
1	Data transfer ends by the specified count

Bit 0—DMAC Enable Bit (DE): DE enables channel operation.

Bit 0: DE	Description
0	Disables channel operation (Initial value)
1	Enables channel operation

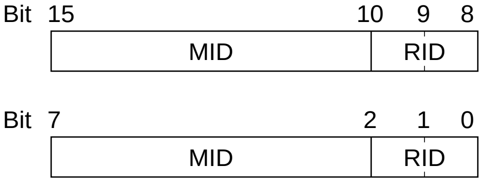
If an auto request is specifies (specified in RS3 to RS0), transfer starts when this bit is set to 1. In an external request or an internal module request, transfer starts if transfer request is generated after this bit is set to 1. Clearing this bit during transfer can terminate transfer.

Even if the DE bit is set, transfer is not enabled if the TE bit is 1, the DME bit in DMAOR is 0, or the NMIF bit in DMAOR is 1.

14.2.5 DMA Channel Expansion Request Registers 0 and 1 (CHCRA0, CHCRA1)

CHCRA0 and CHCRA1 are 16-bit readable/writable registers that specify DMA transfer request sources from peripherals expanded by the SH7622 for each channel. These registers are initialized to 0000 by a reset. In standby mode they retain their previous values. These registers can be used to set SCIF2, USB, and CMT1 transfer requests. CHCRA0 is used for channel 0 and 1 settings, and CHCRA1 for channel 2 and 3 settings.

The register configuration is shown below. The configuration is the same for CHCRA0 and CHCRA1.



Bits 15 to 10 and bits 7 to 2—These bits indicate the transfer request source.

Bits 15 to 10: Channel 1 (register 0) / channel 3 (register 1)
Bits 7 to 2: Channel 0 (register 0) / channel 2 (register 1)

Bits 9 and 8 and bits 1 and 0—These bits indicate the type of transfer of the transfer request source.

Bits 9 and 8: Channel 1 (register 0) / channel 3 (register 1)
Bits 1 and 0: Channel 0 (register 0) / channel 2 (register 1)

The table below shows the contents of MID and RID.

	CHCRA0, CHCRA1	
Peripheral Module	**MID [7:2]**	**RID [1:0]**
SCIF2	B'100000	B'00 (transmission) B'01 (reception)
USB	B'100001	Don’t care
CMT1	B'100011	Don’t care

Operation cannot be guaranteed if initial values or MID/RID settings other than initial value or those shown in the table are used.

Transfer requests from the CHCRA0 and CHCRA1 registers are valid only when the setting of RS (Resource Select) 3:0 in registers CHCR0 to CHCR3 is 1000. When this setting is other than 1000, transfer request sources set in MID and RID will not be accepted.

14.2.6 DMA Operation Register (DMAOR)

The DMA operation register (DMAOR) is a 16-bit read/write register that controls the DMAC transfer mode. Writing to bits 15–10 and bits 7–3 is invalid in this register; 0 is always read if these bits are read.

It is initialized to 0 at power-on reset, or in hardware standby mode or software standby mode.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	PR1	PR0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	AE	NMIF	DME
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/(W)*	R/(W)*	R/W

Note: * Only 0 can be written to the AE and NMIF bits after 1 is read.

Bits 15 to 10—Reserved: These bits are always read as 0. The write value should always be 0.

Bits 9 and 8—Priority Mode Bits 1 and 0 (PR1, PR0): PR1 and PR0 select the priority level between channels when there are transfer requests for multiple channels simultaneously.

Bit 9: PR1	Bit 8: PR0	Description	
0	0	CH0 > CH1 > CH2 > CH3	(Initial value)
0	1	CH0 > CH2 > CH3 > CH1	
1	0	CH2 > CH0 > CH1 > CH3	
1	1	Round-robin	

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

Bit 2—Address Error Flag Bit (AE): AE indicates that an address error occurred during DMA transfer. If this bit is set during data transfer, transfers on all channels are suspended. The CPU cannot write 1 to this bit. This bit can only be cleared by writing 0 after reading 1.

Bit 2: AE	Description
0	No DMAC address error. DMA transfer is enabled (Initial value) [Clearing conditions] • Writing AE = 0 after AE = 1 read • Power-on reset, manual reset
1	DMAC address error. DMA transfer is disabled. This bit is set by occurrence of a DMAC address error.

Bit 1—NMI Flag Bit (NMIF): NMIF indicates that an NMI interrupt occurred. This bit is set regardless of whether DMAC is in operating or halt state. The CPU cannot write 1 to this bit. Only 0 can be written to clear this bit after 1 is read.

Bit 1: NMIF	Description
0	No NMI input. DMA transfer is enabled (Initial value) [Clearing conditions] • Writing NMIF = 0 after NMIF = 1 read • Power-on reset, manual reset
1	NMI input. DMA transfer is disabled This bit is set by occurrence of an NMI interrupt

Bit 0—DMA Master Enable Bit (DME): DME enables or disables DMA transfers on all channels. If the DME bit and the DE bit corresponding to each channel in CHCR are set to 1s, transfer is enabled in the corresponding channel. If this bit is cleared during transfer, transfers in all the channels can be terminated.

Even if the DME bit is set, transfer is not enabled if the TE bit is 1 or the DE bit is 0 in CHCR, or the NMIF bit is 1 in DMAOR.

Bit 0: DME	Description
0	Disable DMA transfers on all channels (Initial value)
1	Enable DMA transfers on all channels

14.3 Operation

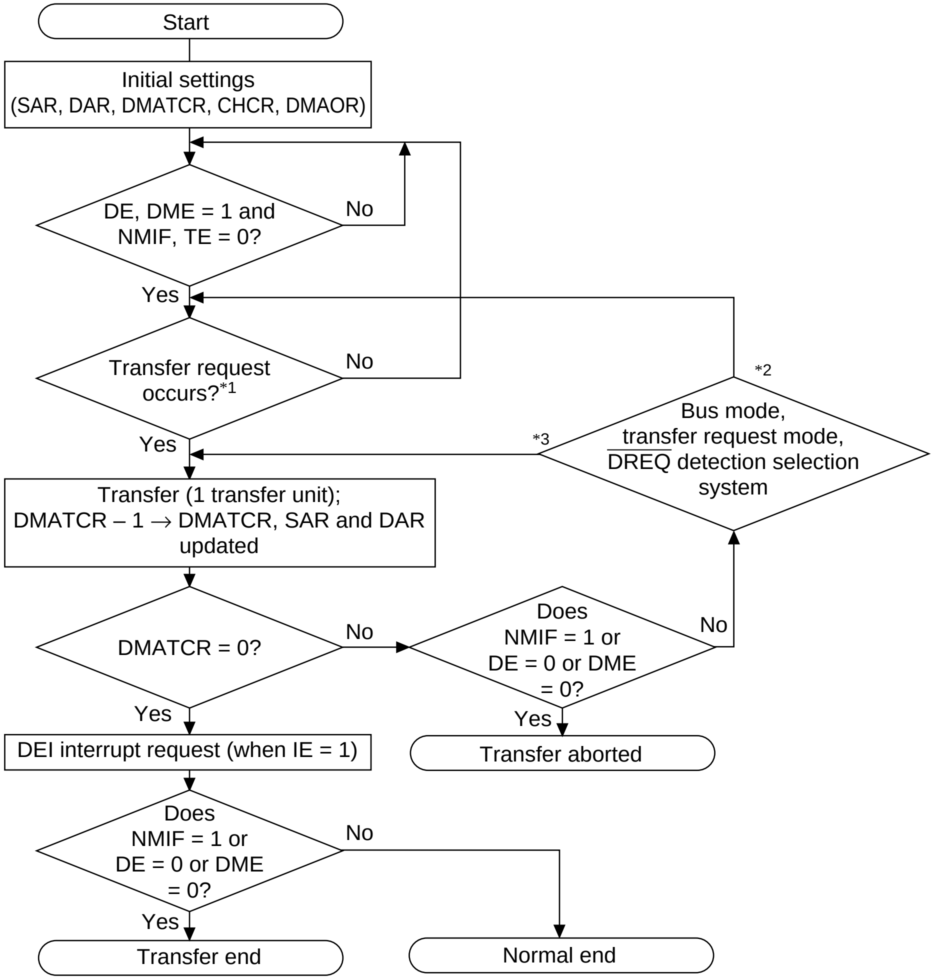
When there is a DMA transfer request, the DMAC starts the transfer according to the predetermined channel priority order; when the transfer end conditions are satisfied, it ends the transfer. Transfers can be requested in three modes: auto request, external request, and on-chip module request. The dual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

14.3.1 DMA Transfer Flow

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

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

Figure 14.2 is a flowchart of this procedure.



Notes: *1 In auto-request mode, transfer begins when NMIF and TE are all 0 and the DE and DME bits are set to 1.

*2 DREQ = level detection in burst mode (external request) or cycle-steal mode

*3 DREQ = edge detection in burst mode (external request), or auto-request mode in burst mode

Figure 14.2 DMAC Transfer Flowchart

14.3.2 DMA Transfer Requests

DMA transfer requests are basically generated in either the data transfer source or destination, but they can also be generated by devices and on-chip peripheral modules that are neither the source nor the destination. Transfers can be requested in three modes: auto request, external request, and on-chip module request. The request mode is selected in the RS3–RS0 bits of the DMA channel control registers 0–3 (CHCR0–CHCR3).

Auto-Request Mode: When there is no transfer request signal from an external source, as in a memory-to-memory transfer or a transfer between memory and an on-chip peripheral module unable to request a transfer, the auto-request mode allows the DMAC to automatically generate a transfer request signal internally. When the DE bits of CHCR0–CHCR3 and the DME bit of the DMAOR are set to 1, the transfer begins so long as the TE bits of CHCR0–CHCR3 and the NMIF bit of DMAOR are all 0.

External Request Mode: In this mode a transfer is performed at the request signal ($\overline{\text{DREQ}}$) of an external device. Choose one of the modes shown in table 14.3 according to the application system. When this mode is selected, if the DMA transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0), a transfer is performed upon a request at the $\overline{\text{DREQ}}$ input. Choose to detect $\overline{\text{DREQ}}$ by either the falling edge or low level of the signal input with the DS bit of CHCR0 and CHCR1 (DS = 0 is level detection, DS = 1 is edge detection). The source of the transfer request does not have to be the data transfer source or destination.

Table 14.3 Selecting External Request Modes with the RS Bits

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

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

On-Chip Module Request: In this mode a transfer is performed at the transfer request signal (interrupt request signal) of an on-chip module. Transfer request signals comprise the receive FIFO data full interrupt (RXI) and transmit FIFO data interrupt (TXI) from SCIF0 and SCIF1, the A/D conversion end interrupt (ADI) from the A/D converter, the compare-match timer transfer request (CMI) from CMT0, and transfer requests from the USB, SCIF2, and CMT1 that can be set in CHCRA0/1. When this mode is selected, if the DMA transfer is enabled (DE = 1, DME = 1, TE = 0, NMIF = 0), a transfer is performed upon the input of a transfer request signal. The source of the transfer request does not have to be the data transfer source or destination. When RXI is set as the transfer request, however, the transfer source must be the SCI's receive data register. Likewise, when TXI is set as the transfer request, the transfer source must be the SCI's transmit data register. These conditions also apply to the USB. And if the transfer requester is the A/D converter, the data transfer source must be the A/D data register.

Table 14.4 Selecting On-Chip Peripheral Module Request Modes with the RS Bit

RS3	RS2	RS1	RS0	DMA Transfer Request Source	DMA Transfer Request Signal	Source	Desti-nation	Bus Mode
1	0	1	0	SCIF0 transmitter*2	SCIF0 transmit	Any*1	SCFTDR0	Burst/ cycle steal
1	0	1	1	SCIF0 receiver*2	SCIF0 receive	SCFRDR0	Any*1	Burst/ cycle steal
1	1	0	0	SCIF1 transmitter*2	SCIF1 transmit	Any*1	SCFTDR1	Burst/ cycle steal
1	1	0	1	SCIF1 receiver*2	SCIF1 receive	SCFRDR1	Any*1	Burst/ cycle steal
1	1	1	0	A/D converter	ADI (A/D conversion end interrupt)	ADDR	Any*1	Cycle steal
1	1	1	1	CMT	CMI (Compare match timer interrupt)	Any*1	Any*1	Burst/ cycle steal

ADDR: A/D data register of A/D converter

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

*2 When SCIF0/1/2 is designated as the transfer request source, PMA transfer data exceeding the FIFO trigger set number is ignored.

Table 14.4 Selecting On-Chip Peripheral Module Request Modes with the RS Bit (cont)

CHCR	CHCRA						
RS [3:0]	**MID [15:10]/ [7:2]**	**RID [9:8]/ [1:0]**	**DMA Transfer Request Source**	**DMA Transfer Request Signal**	**Source**	**Destination**	**Bus Mode**
1000	100000	00	SCIF2 transmitter*2	TXI2 (transmit data FIFO empty interrupt)	Any*1	SCFTDR2	Cycle steal
		01	SCIF2 receiver*2	RXI2 (receive data FIFO full interrupt)	SCFRDR2	Any*1	Cycle steal
	100001	Don't care	USB	EP1 FIFO full interrupt setting	USBEPDR1	Any*1	Burst/ cycle steal
				EP2 FIFO empty interrupt setting	Any*1	USBEPDR2	Burst/ cycle steal
	100011	Don't care	CMT1	CMT1 compare-match interrupt	Any*1	Any*1	Cycle steal

USBEPDR1: USB function module EP1 data register

USBEPDR2: USB function module EP2 data register

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

*2 When SCIF0/1/2 is designated as the transfer request source, DMA transfer data exceeding the FIFO trigger set number is ignored.

When outputting transfer requests from on-chip peripheral modules, the appropriate interrupt enable bits must be set to output the interrupt signals.

If the interrupt request signal of the on-chip peripheral module is used as a DMA transfer request signal, an interrupt is not generated to the CPU.

The DMA transfer request signals of table 14.4 are automatically withdrawn when the corresponding DMA transfer is performed. If the cycle steal mode is being employed, they are withdrawn at the first transfer; if the burst mode is being used, they are withdrawn at the last transfer.

14.3.3 Channel Priority

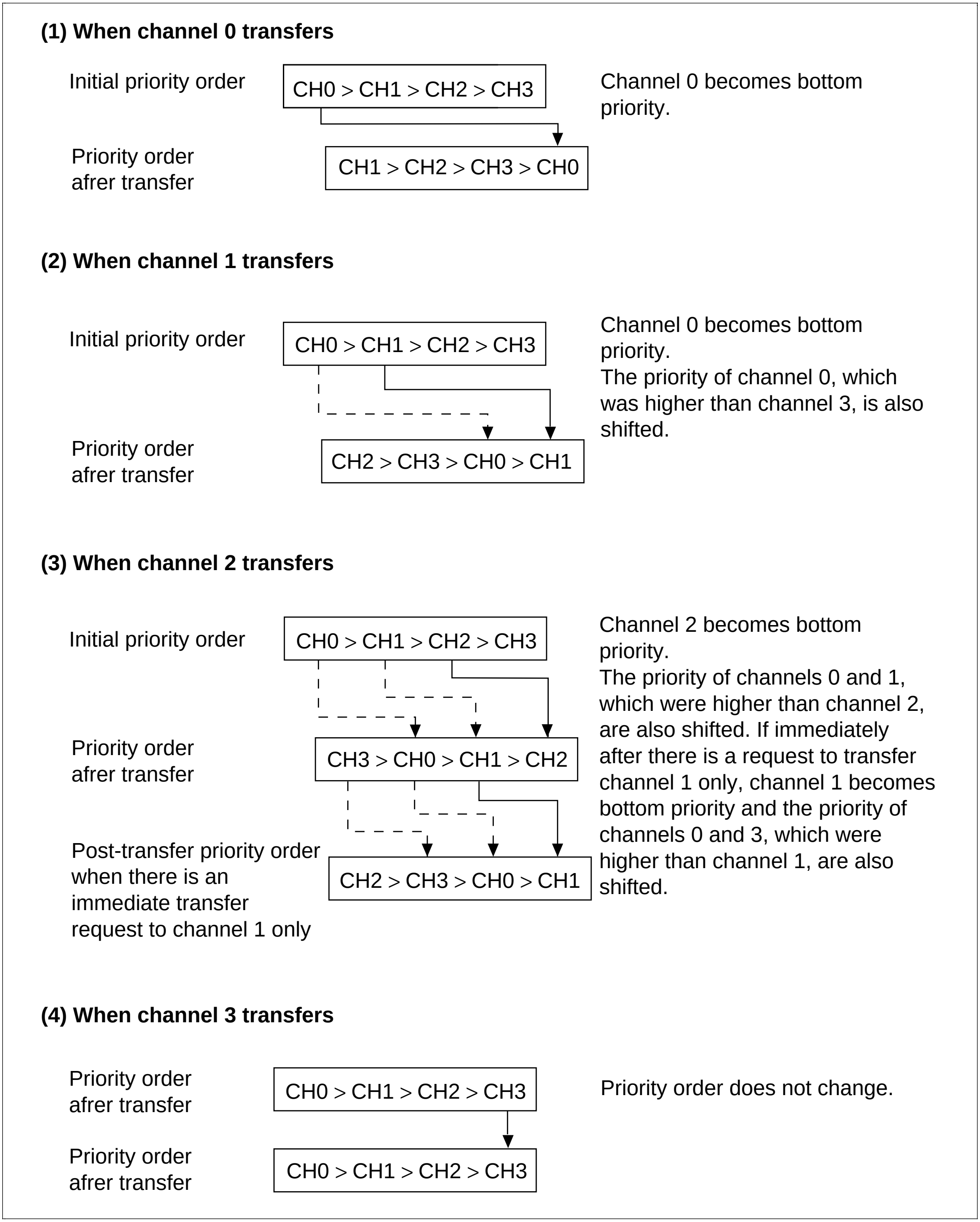
When the DMAC receives simultaneous transfer requests on two or more channels, it selects a channel according to a predetermined priority order. Two modes (fixed mode and round-robin mode) are selected by the priority bits PR1 and PR0 in the DMA operation register.

Fixed Mode: In these modes, the priority levels among the channels remain fixed. There are three kinds of fixed modes as follows:

CH0 > CH1 > CH2 > CH3
CH0 > CH2 > CH3 > CH1
CH2 > CH0 > CH1 > CH3

These are selected by the PR1 and the PR0 bits in the DMA operation register (DMAOR).

Round-Robin Mode: Each time one word, byte, or longword is transferred on one channel, the priority order is rotated. The channel on which the transfer was just finished rotates to the bottom of the priority order. The round-robin mode operation is shown in figure 14.3. The priority of the round-robin mode is CH0 > CH1 > CH2 > CH3 immediately after reset.


(1) When channel 0 transfers
Initial priority order
CH0 > CH1 > CH2 > CH3
Priority order afrer transfer
CH1 > CH2 > CH3 > CH0
(2) When channel 1 transfers
Initial priority order
CH0 > CH1 > CH2 > CH3
Priority order afrer transfer
CH2 > CH3 > CH0 > CH1
(3) When channel 2 transfers
Initial priority order
CH0 > CH1 > CH2 > CH3
Priority order afrer transfer
CH3 > CH0 > CH1 > CH2
Post-transfer priority order when there is an immediate transfer request to channel 1 only
CH2 > CH3 > CH0 > CH1
(4) When channel 3 transfers
Priority order afrer transfer
CH0 > CH1 > CH2 > CH3
Priority order afrer transfer
CH0 > CH1 > CH2 > CH3

(1) Channel 0 becomes bottom priority.

(2) Channel 0 becomes bottom priority. The priority of channel 0, which was higher than channel 3, is also shifted.

(3) Channel 2 becomes bottom priority. The priority of channels 0 and 1, which were higher than channel 2, are also shifted. If immediately after there is a request to transfer channel 1 only, channel 1 becomes bottom priority and the priority of channels 0 and 3, which were higher than channel 1, are also shifted.

(4) Priority order does not change.

Figure 14.3 Round-Robin Mode

Figure 14.4 shows how the priority order changes when channel 0 and channel 3 transfers are requested simultaneously and a channel 1 transfer is requested during the channel 0 transfer. The DMAC operates as follows:

1. Transfer requests are generated simultaneously to channels 0 and 3.
2. Channel 0 has a higher priority, so the channel 0 transfer begins first (channel 3 waits for transfer).
3. A channel 1 transfer request occurs during the channel 0 transfer (channels 1 and 3 are both waiting).
4. When the channel 0 transfer ends, channel 0 becomes lowest priority.
5. At this point, channel 1 has a higher priority than channel 3, so the channel 1 transfer begins (channel 3 waits for transfer).
6. When the channel 1 transfer ends, channel 1 becomes lowest priority.
7. The channel 3 transfer begins.
8. When the channel 3 transfer ends, channels 3 and 2 shift downward in priority so that channel 3 becomes the lowest priority.

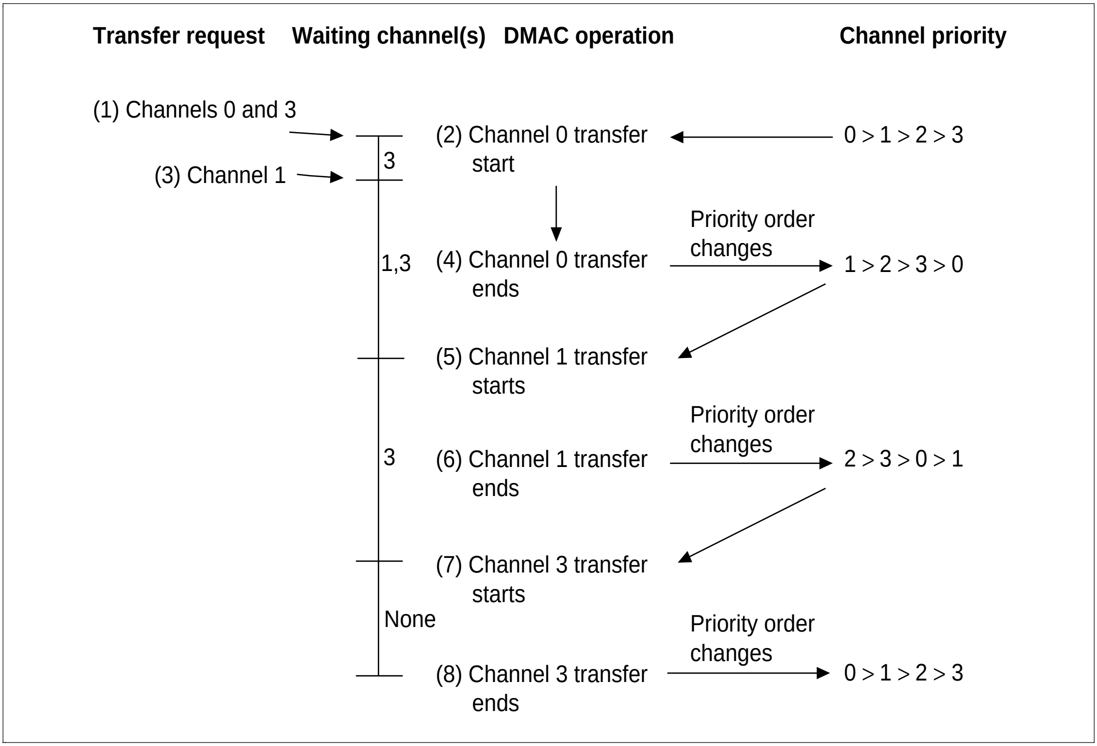


Figure 14.4 Changes in Channel Priority in Round-Robin Mode

14.3.4 DMA Transfer Types

The DMAC supports the transfers shown in table 14.5. In the dual address mode, both the transfer source address and the transfer destination address are output. The dual address mode has the direct address mode and the indirect address mode. In the direct address mode, an output address value is the data transfer target address; in the indirect address mode, the value stored in the output address, not the output address value itself, is the data transfer target address. A data transfer timing depends on the bus mode, which has cycle steal mode and burst mode.

Table 14.5 Supported DMA Transfers

	Destination			
Source	**External Device with DACK**	**External Memory**	**Memory-Mapped External Device**	**On-Chip Peripheral Module or X/Y Memory**
External device with DACK	Not available	Dual, single	Dual, single	Not available
External memory	Dual, single	Dual	Dual	Dual
Memory-mapped external device	Dual, single	Dual	Dual	Dual
On-chip peripheral module or X/Y memory	Not available	Dual	Dual	Dual

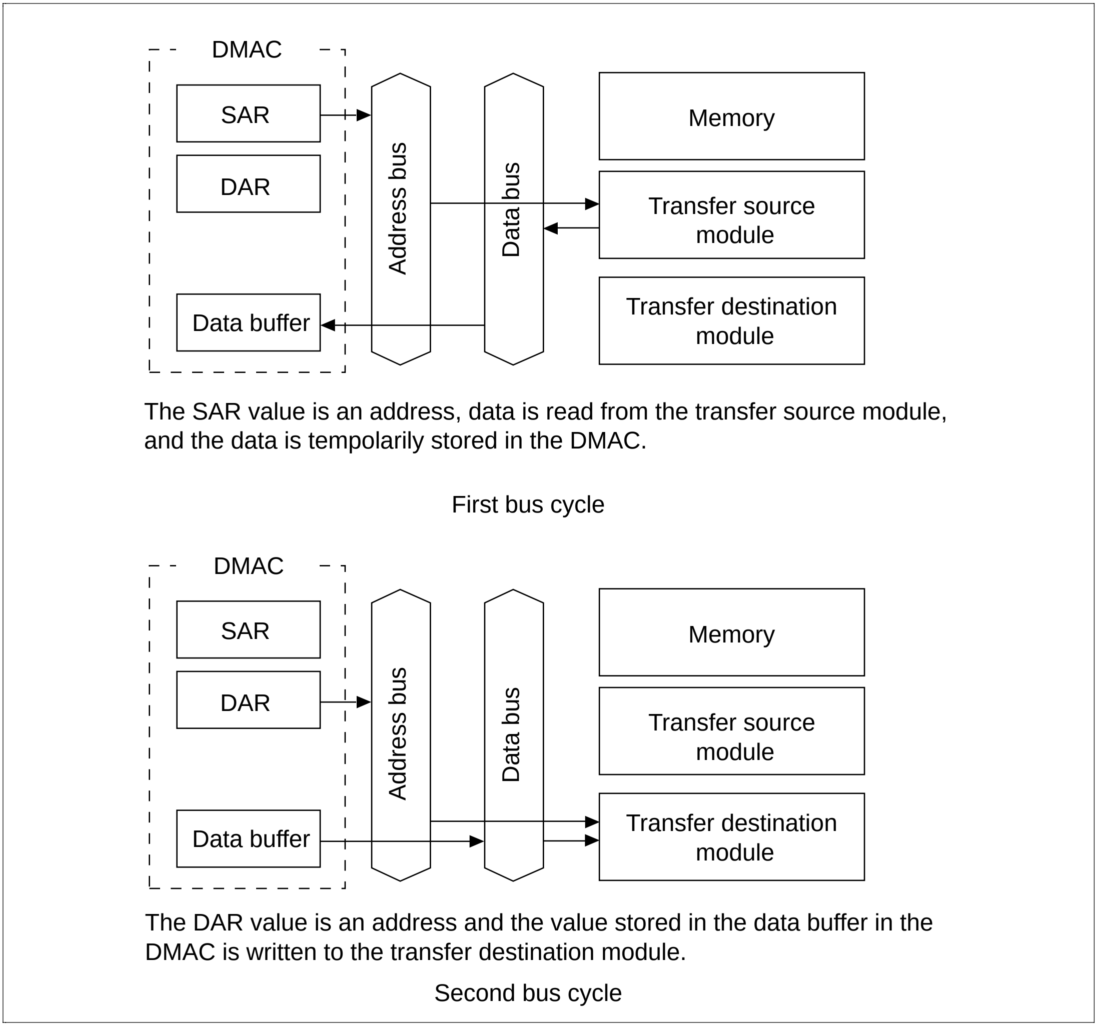
Notes: 1. Dual: Dual address mode
2. Single: Single address mode
3. The dual address mode includes the direct address mode and the indirect address mode.
4. 16-byte transfer is not available for on-chip peripheral modules.

Address Modes

- Dual Address Mode

  In the dual address mode, both the transfer source and destination are accessed (selectable) by an address. The source and destination can be located externally or internally. The dual address mode has (1) direct address transfer mode and (2) indirect address transfer mode.

  (1) In the direct address transfer mode, DMA transfer requires two bus cycles because data is read from the transfer source in a data read cycle and written to the transfer destination in a data write cycle. At this time, transfer data is temporarily stored in the DMAC. In the transfer between external memories as shown in figure 14.5, data is read to the DMAC from one external memory in a data read cycle, and then that data is written to the other external memory in a write cycle. Figures 14.6 to 14.8 show examples of the timing at this time.



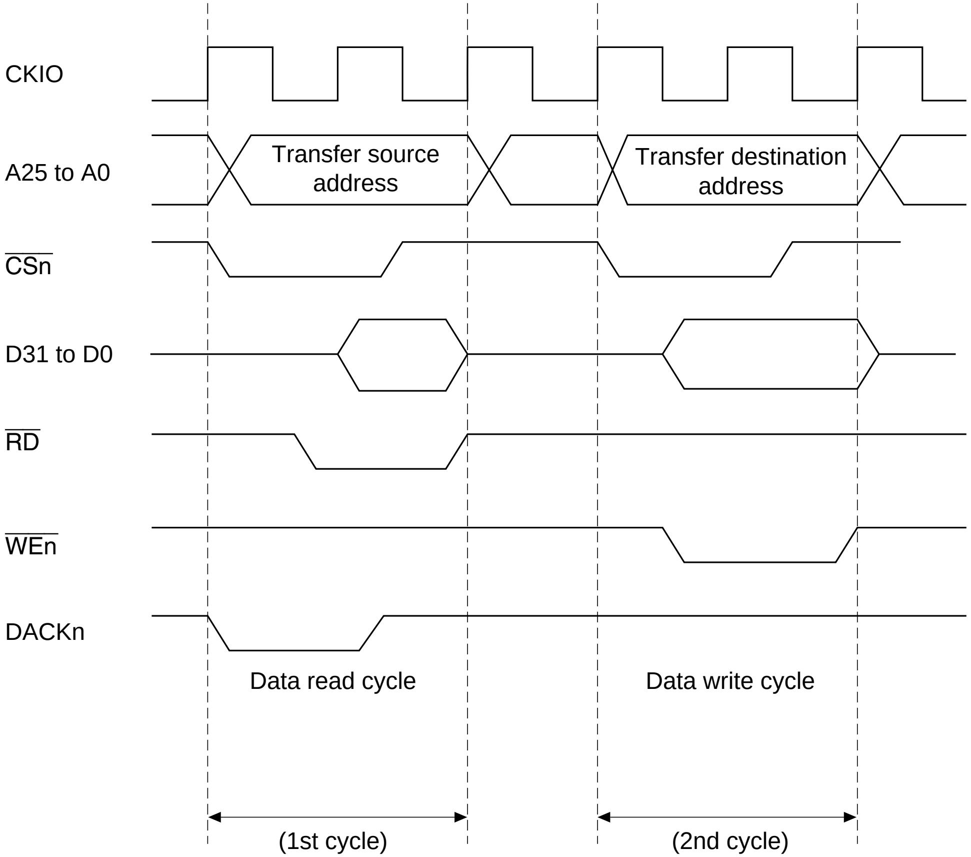
The SAR value is an address, data is read from the transfer source module, and the data is tempolarily stored in the DMAC.

First bus cycle

The DAR value is an address and the value stored in the data buffer in the DMAC is written to the transfer destination module.

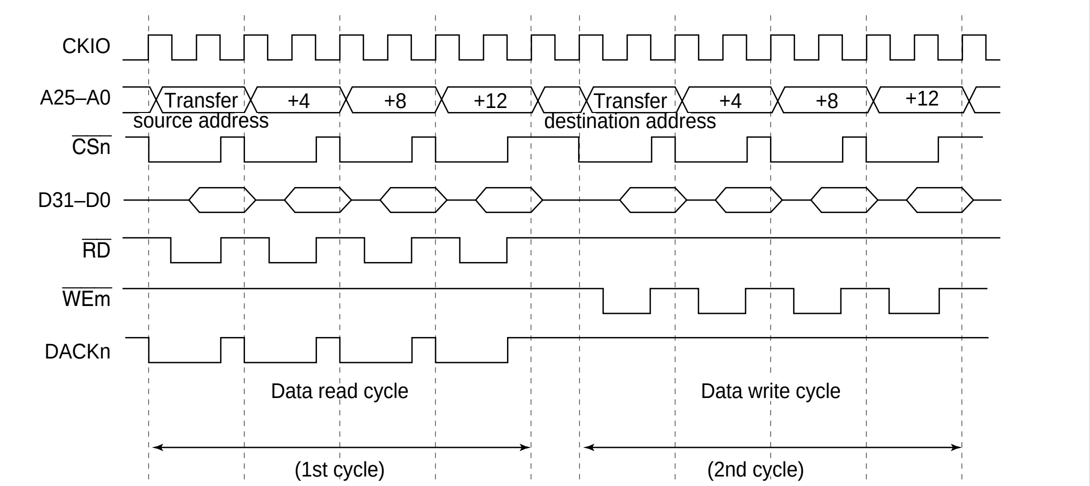
Second bus cycle

Figure 14.5 Operation of Direct Address Mode in Dual Address Mode



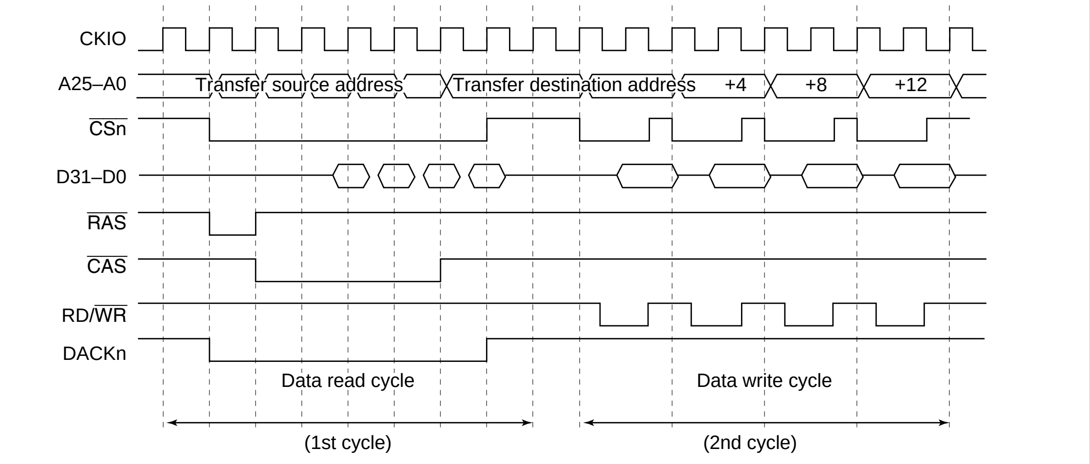
Note: Transfer between external memories, DACK output in a read cycle DACK output timing is the same as that of CSn.

Figure 14.6 Example of Direct Address Mode DMA Transfer Timing in Dual Mode (Transfer Source: Ordinary Memory, Transfer Destination: Ordinary Memory)



Note: Transfer between external memories, DACK output in a read cycle DACK output timing is the same as that of $\overline{\text{CSn}}$.

Figure 14.7 Example of Direct Address Mode DMA Transfer Timing in Dual Mode (16-byte Transfer, Transfer Source: Ordinary Memory, Transfer Destination: Ordinary Memory)



Note: Transfer between external memories, DACK output in a read cycle DACK output timing is the same as that of $\overline{\text{CSn}}$.

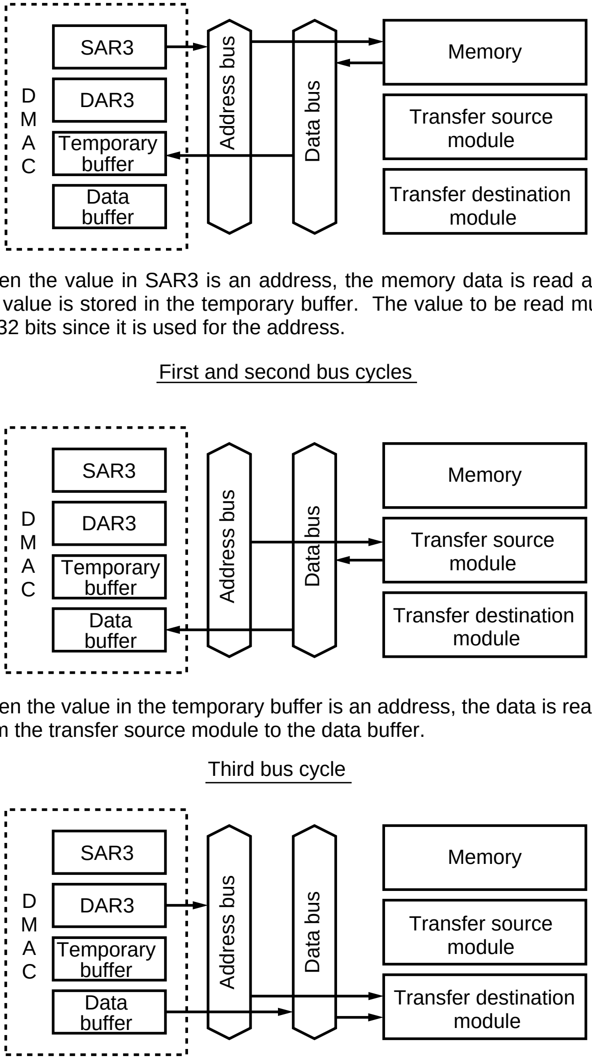
Figure 14.8 Example of Direct Address Mode DMA Transfer Timing in Dual Mode (16-byte Transfer, Transfer Source: Synchronous DRAM, Transfer Destination: Ordinary Memory)

(2) In the indirect address transfer mode, the address of memory in which data to be transferred is stored is specified in the transfer source address register (SAR3) in the DMAC. 16-byte transfer is not possible. Consequently, in this mode, the address value specified in the transfer source address register in the DMAC is read first. This value is temporarily stored in the DMAC. Next, the read value is output as an address, and the value stored in that address is stored in the DMAC again. Then, the value read afterwards is written to the address specified in the transfer destination address; this completes one DMA transfer.

Figure 14.9 shows one example. In this example, the transfer destination, the transfer source, and the storage destination of the indirect address are external memories, and transfer data is 16 or 8 bits. Figure 14.10 shows an example of the transfer timing.

In this mode, one NOP cycle (CK1 cycle shown in figure 14.10) is required to output data read as an indirect address to an address bus.

If transfer data is 32 bits, third and fourth bus cycles shown in figure 14.10 is required twice for each; a total of six bus cycles and one NOP cycle are required.



When the value in SAR3 is an address, the memory data is read and the value is stored in the temporary buffer. The value to be read must be 32 bits since it is used for the address.

First and second bus cycles

When the value in the temporary buffer is an address, the data is read from the transfer source module to the data buffer.

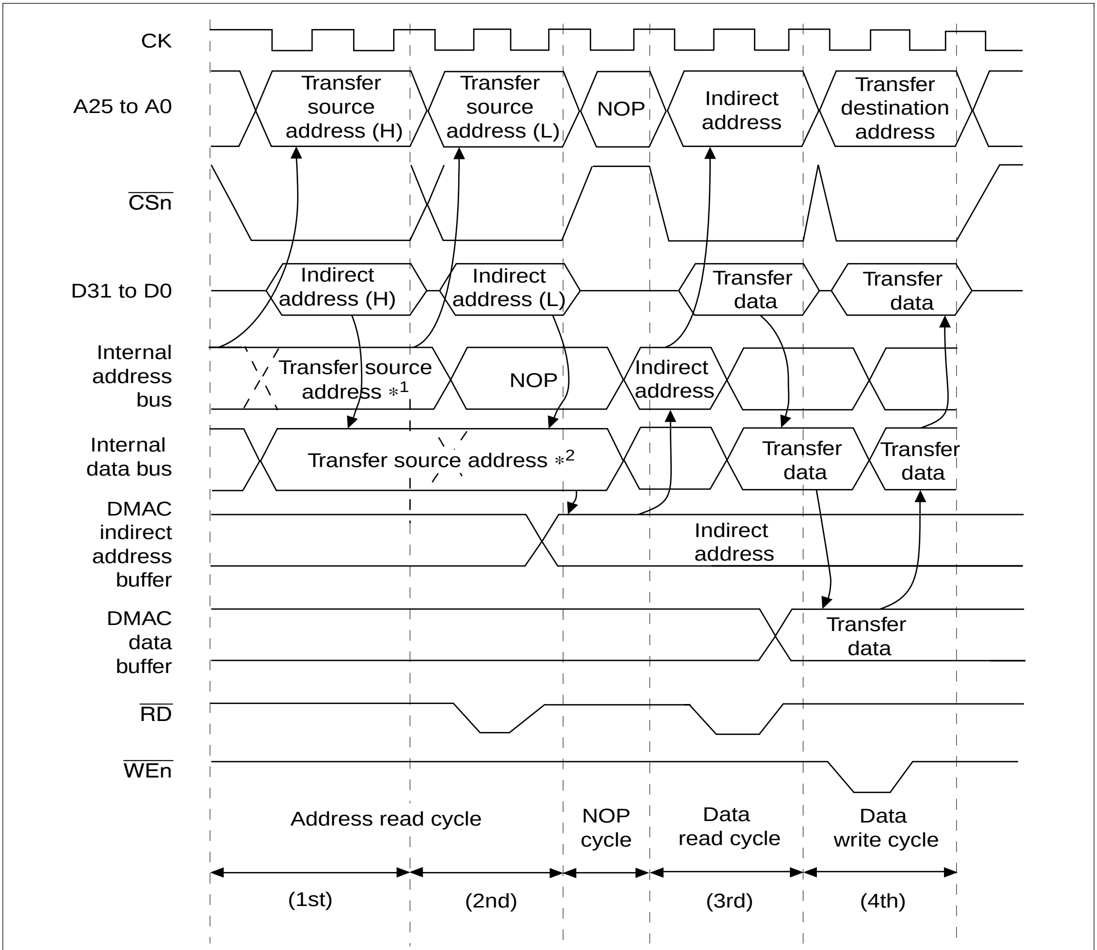
Third bus cycle

When the value in SAR3 is an address, the value in the data buffer is written to the transfer source module.

Fourth bus cycle

Note: The above description uses the memory, transfer source module, or transfer destination module; in practice, any module can be connected in the addressing space.

Figure 14.9 Operation of Indirect Address in the Dual Address Mode (When the External Memory Space has a 16-bit Width)



Notes: *1 The internal address bus value does not change, and controlled by the port.
*2 The DMAC does not fetch the value until 32-bit data is output to the internal data bus.

Figure 14.10 Example of Transfer Timing in the Indirect Address Mode in the Dual Address Mode (Transfer between External Memories (16-Bit-Width External Memory Space))

- Single Address Mode

  In single address mode, either the transfer source or transfer destination peripheral device is accessed (selected) by means of the DACK signal, and the other device is accessed by address. In this mode, the DMAC performs one DMA transfer in one bus cycle, accessing one of the external devices by outputting the DACK transfer request acknowledge signal to it, and at the same time outputting an address to the other device involved in the transfer. For example, in the case of transfer between external memory and an external device with DACK shown in figure 14.11, when the external device outputs data to the data bus, that data is written to the external memory in the same bus cycle.

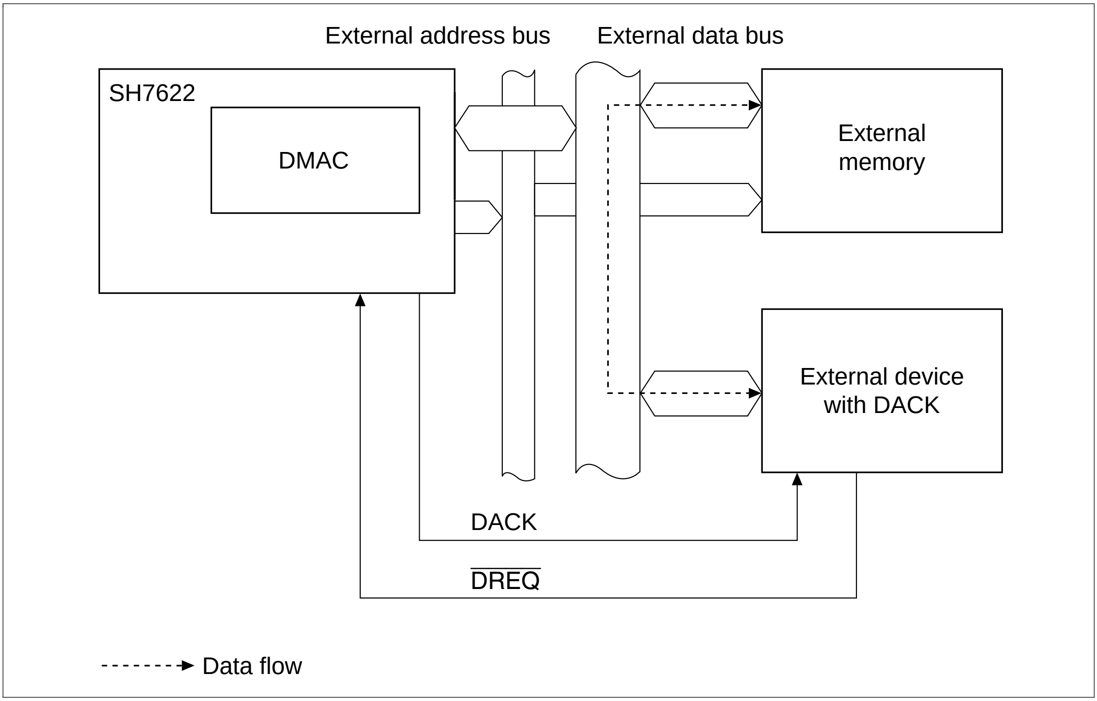
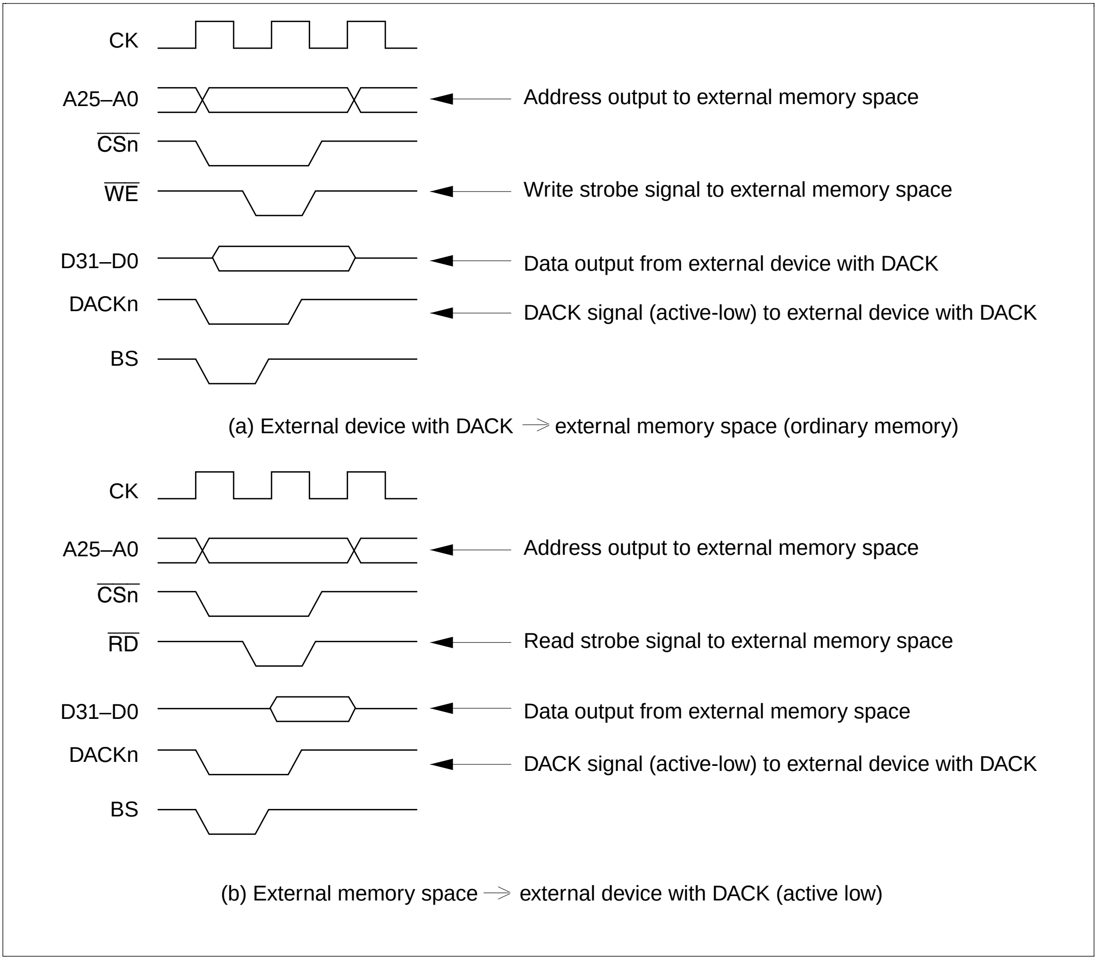


Figure 14.11 Data Flow in Single Address Mode

Two kinds of transfer are possible in single address mode: (1) transfer between an external device with DACK and a memory-mapped external device, and (2) transfer between an external device with DACK and external memory. In both cases, only the external request signal ($\overline{\text{DREQ}}$) is used for transfer requests.

Figures 14.12 to 14.14 show examples of DMA transfer timing in single address mode.


CK
A25–A0
$\overline{\text{CSn}}$
$\overline{\text{WE}}$
D31–D0
DACKn
BS
Address output to external memory space
Write strobe signal to external memory space
Data output from external device with DACK
DACK signal (active-low) to external device with DACK
CK
A25–A0
$\overline{\text{CSn}}$
$\overline{\text{RD}}$
D31–D0
DACKn
BS
Address output to external memory space
Read strobe signal to external memory space
Data output from external memory space
DACK signal (active-low) to external device with DACK

(a) External device with DACK → external memory space (ordinary memory)

(b) External memory space → external device with DACK (active low)

Figure 14.12 Example of Transfer Timing in Single Address Mode

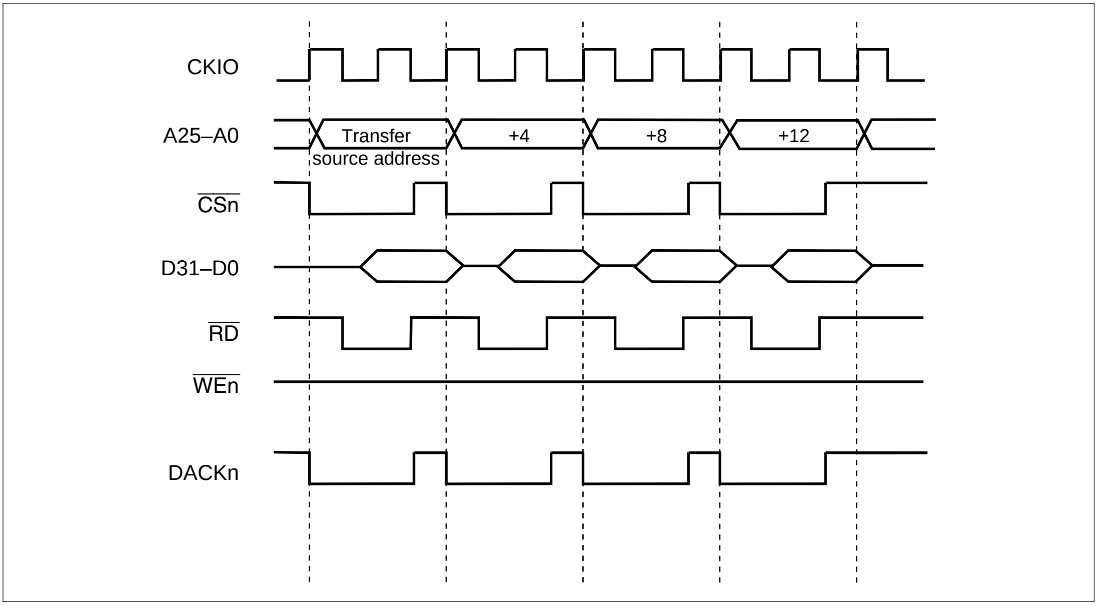


Figure 14.13 Example of Transfer Timing in Single Address Mode (External Memory Space (Ordinary Memory) → External Device with DACK)

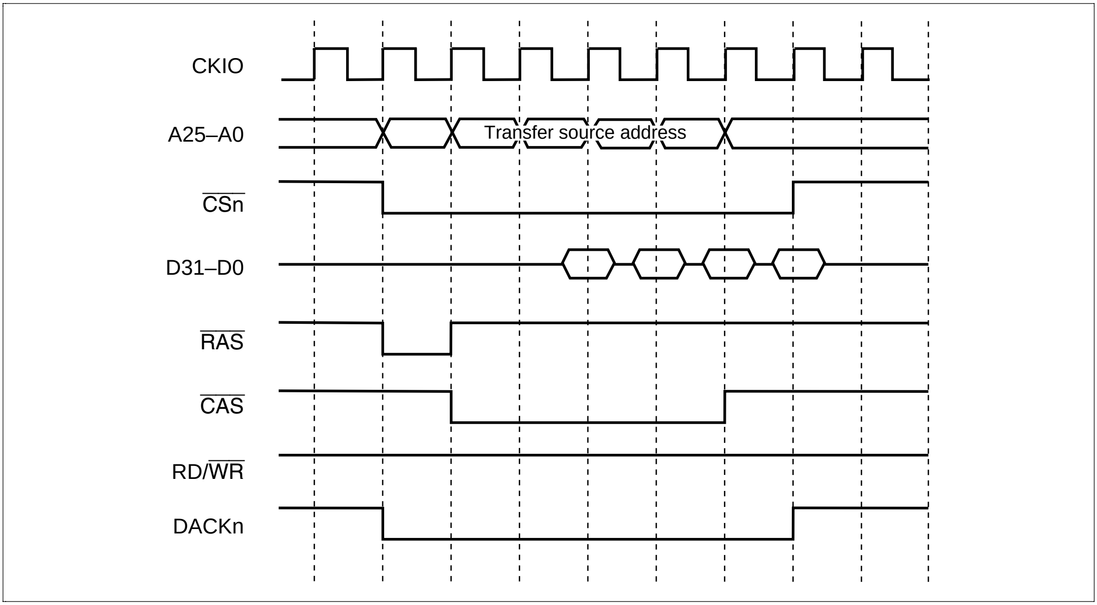


Figure 14.14 Example of Transfer Timing in Single Address Mode (External Memory Space (SDRAM) → External Device with DACK)

Bus Modes: There are two bus modes: cycle steal and burst. Select the mode in the TM bits of CHCR0–CHCR3.

- Cycle-Steal Mode

  In the cycle-steal mode, the bus right is given to another bus master after a one-transfer-unit (8-, 16-, or 32-bit unit) DMA transfer. When another transfer request occurs, the bus rights are obtained from the other bus master and a transfer is performed for one transfer unit. When that transfer ends, the bus right is passed to the other bus master. This is repeated until the transfer end conditions are satisfied.

  In the cycle-steal mode, transfer areas are not affected regardless of settings of the transfer request source, transfer source, and transfer destination. Figure 14.15 shows an example of DMA transfer timing in the cycle steal mode. Transfer conditions shown in the figure are

  — Dual address mode

  — $\overline{\text{DREQ}}$ level detection

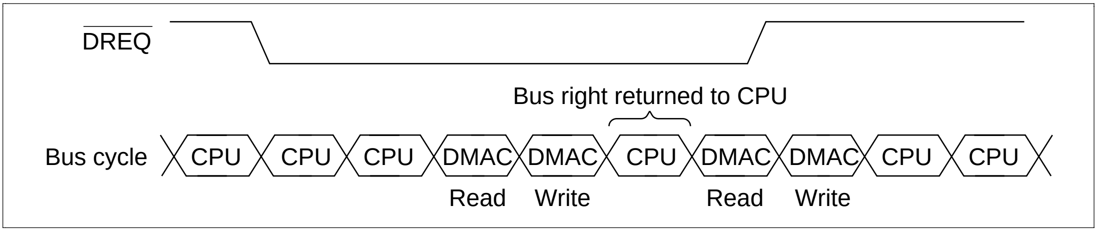


Figure 14.15 Transfer Example in the Cycle-Steal Mode

- Burst Mode

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

  The burst mode cannot be used when the A/D converter, SCIF2, or CMT1 is the transfer request source. Figure 14.16 shows a timing at this point.

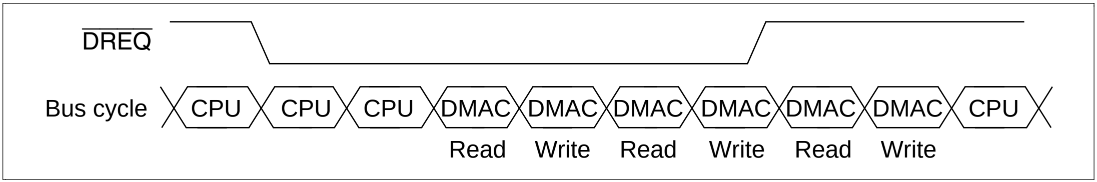


Figure 14.16 Transfer Example in the Burst Mode

Relationship between Request Modes and Bus Modes by DMA Transfer Category: Table 14.6 shows the relationship between request modes and bus modes by DMA transfer category.

Table 14.6 Relationship of Request Modes and Bus Modes by DMA Transfer Category

Address Mode	Transfer Category	Request Mode	Bus Mode	Transfer Size (bits)	Usable Channels
Dual	External device with DACK and external memory	External	B/C	8/16/32/128	0,1
	External device with DACK and memory-mapped external device	External	B/C	8/16/32/128	0, 1
	External memory and external memory	All*[1]	B/C	8/16/32/128	0–3*[4]
	External memory and memory-mapped external device	All*[1]	B/C	8/16/32/128	0–3*[4]
	Memory-mapped external device and memory-mapped external device	All*[1]	B/C	8/16/32/128	0–3*[4]
	External memory and on-chip peripheral module	All*[2]	B/C	8/16/32/128*[3]	0–3*[4]
	Memory-mapped external device and on-chip peripheral module	All*[2]	B/C	8/16/32/128*[3]	0–3*[4]
	On-chip peripheral module and on-chip peripheral module	All*[2]	B/C	8/16/32/128*[3]	0–3*[4]
	X/Y memory and X/Y memory	All	B/C	8/16/32/128	0–3
	X/Y memory and memory-mapped external device	All*[1]	B/C	8/16/32/128	0–3
	X/Y memory and on-chip peripheral module	All*[2]	B/C	8/16/32/128*[3]	0–3
	X/Y memory and external memory	All	B/C	8/16/32/128	0–3
Single	External device with DACK and external memory	External	B/C	8/16/32/128	0, 1
	External device with DACK and memory-mapped external device	External	B/C	8/16/32/128	0, 1

B: Burst, C: Cycle steal

Notes: *1 External requests, auto requests and on-chip peripheral module (CMT0, CMT1) requests are all available.

*2 External requests, auto requests, and on-chip peripheral module requests are all available. In the case of on-chip peripheral module requests, however, with the exception of CMT0 and CMT1, the module must be designated as the transfer request source or the transfer destination.

*3 Access size permitted for the on-chip peripheral module register functioning as the transfer source or transfer destination (With a 128-bit transfer size, the permitted access size is 32 bits only).

*4 If the transfer request is an external request, channels 0 and 1 are only available.

Bus Mode and Channel Priority Order: When a given channel 1 is transferring in burst mode and there is a transfer request to a channel 0 with a higher priority, the transfer of channel 0 will begin immediately.

At this time, if the priority is set in the fixed mode (CH0 > CH1), the channel 1 transfer will continue when the channel 0 transfer has completely finished, even if channel 0 is operating in the cycle steal mode or in the burst mode.

If the priority is set in the round-robin mode, channel 1 will begin operating again after channel 0 completes the transfer of one transfer unit, even if channel 0 is in the cycle steal mode or in the burst mode. The bus will then switch between the two in the order channel 1, channel 0, channel 1, channel 0.

Even if the priority is set in the fixed mode or in the round-robin mode, it will not give the bus to the CPU since channel 1 is in the burst mode. This example is illustrated in figure 14.17.

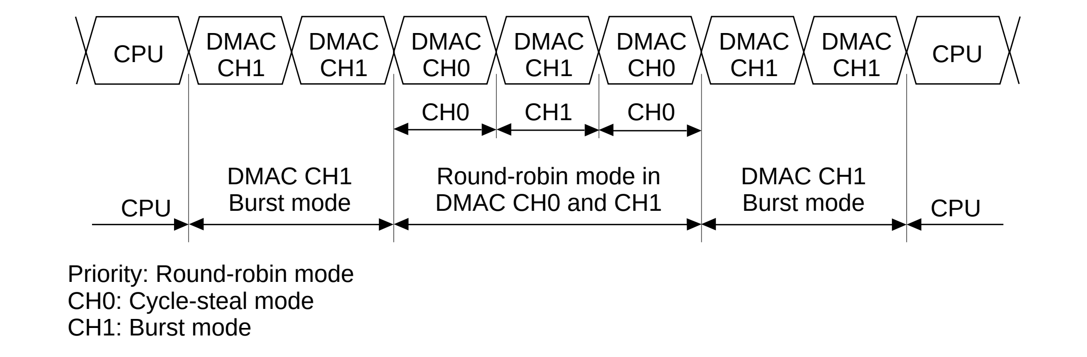


Figure 14.17 Bus State when Multiple Channels are Operating

14.3.5 Number of Bus Cycle States and $\overline{\text{DREQ}}$ Pin Sampling Timing

Number of Bus Cycle States: When the DMAC is the bus master, the number of bus cycle states is controlled by the bus state controller (BSC) in the same way as when the CPU is the bus master. For details, see section 13, Bus State Controller (BSC).

$\overline{\text{DREQ}}$ Pin Sampling Timing: In external request mode, the $\overline{\text{DREQ}}$ pin is sampled by clock pulse (CKIO) falling edge or low level detection. When $\overline{\text{DREQ}}$ input is detected, a DMAC bus cycle is generated and DMA transfer performed, at the earliest, three states later.

The second and subsequent $\overline{\text{DREQ}}$ sampling operations are started two cycles after the first sample.

Operation

- Cycle-Steal Mode

  In cycle-steal mode, the $\overline{\text{DREQ}}$ sampling timing is the same regardless of whether level or edge detection is used.

  For example, in figure 14.18 (cycle-steal mode, level detection), DMAC transfer begins, at the earliest, three cycles after the first sampling is performed. The second sampling is started two cycles after the first. If $\overline{\text{DREQ}}$ is not detected at this time, sampling is performed in each subsequent cycle.

  Thus, $\overline{\text{DREQ}}$ sampling is performed one step in advance. The third sampling operation is not performed until the idle cycle following the end of the first DMA transfer.

  The above conditions are the same whatever the number of CPU transfer cycles, as shown in figure 14.19. The above conditions are also the same whatever the number of DMA transfer cycles, as shown in figure 14.20.

  DACK is output in a read in the example in figure 14.18, and in a write in the example in figure 14.19. In both cases, DACK is output for the same duration as $\overline{\text{CSn}}$.

  Figure 14.21 shows an example in which sampling is executed in all subsequent cycles when $\overline{\text{DREQ}}$ cannot be detected.

  Figure 14.22 shows an example of operation in cycle-steal mode when using edge detection.

- Burst Mode, Level Detection

  In the case of burst mode with level detection, the $\overline{\text{DREQ}}$ sampling timing is the same as in the cycle-steal mode.

  For example, in figure 14.23, DMAC transfer begins, at the earliest, three cycles after the first sampling is performed. The second sampling is started two cycles after the first. Subsequent sampling operations are performed in the idle cycle following the end of the DMA transfer cycle.

  In the burst mode, also, the DACK output period is the same as in the cycle-steal mode.

- Burst Mode, Edge Detection

  In the case of burst mode with edge detection, $\overline{\text{DREQ}}$ sampling is only performed once.

  For example, in figure 14.24, DMAC transfer begins, at the earliest, three cycles after the first sampling is performed. After this, DMAC transfer is executed continuously until the number of data transfers set in the DMATCR register have been completed. $\overline{\text{DREQ}}$ is not sampled during this time.

  To restart DMA transfer after it has been suspended by an NMI, first clear NMIF, then input an edge request again.

  In the burst mode, also, the DACK output period is the same as in the cycle-steal mode.

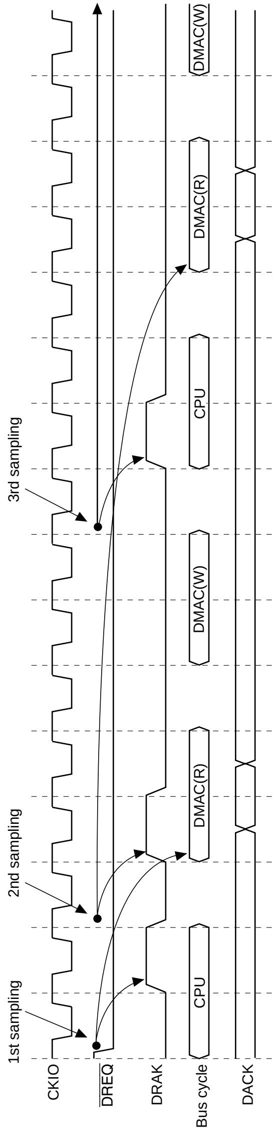


Figure 14.18 Cycle-Steal Mode, Level Input (CPU Access: 2 Cycles)

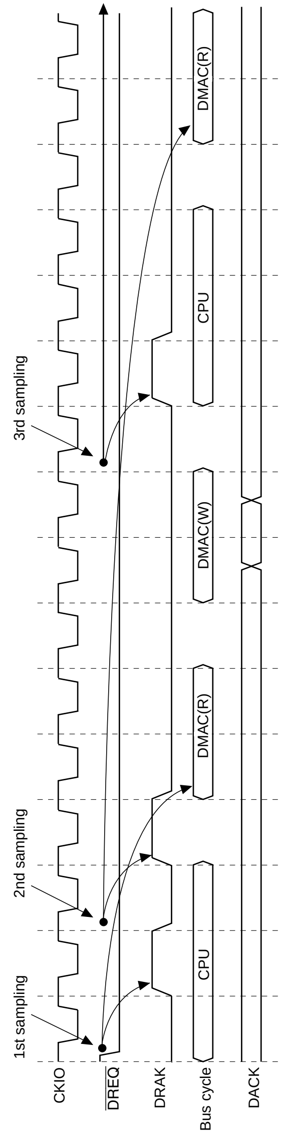


Figure 14.19 Cycle-Steal Mode, Level Input (CPU Access: 3 Cycles)

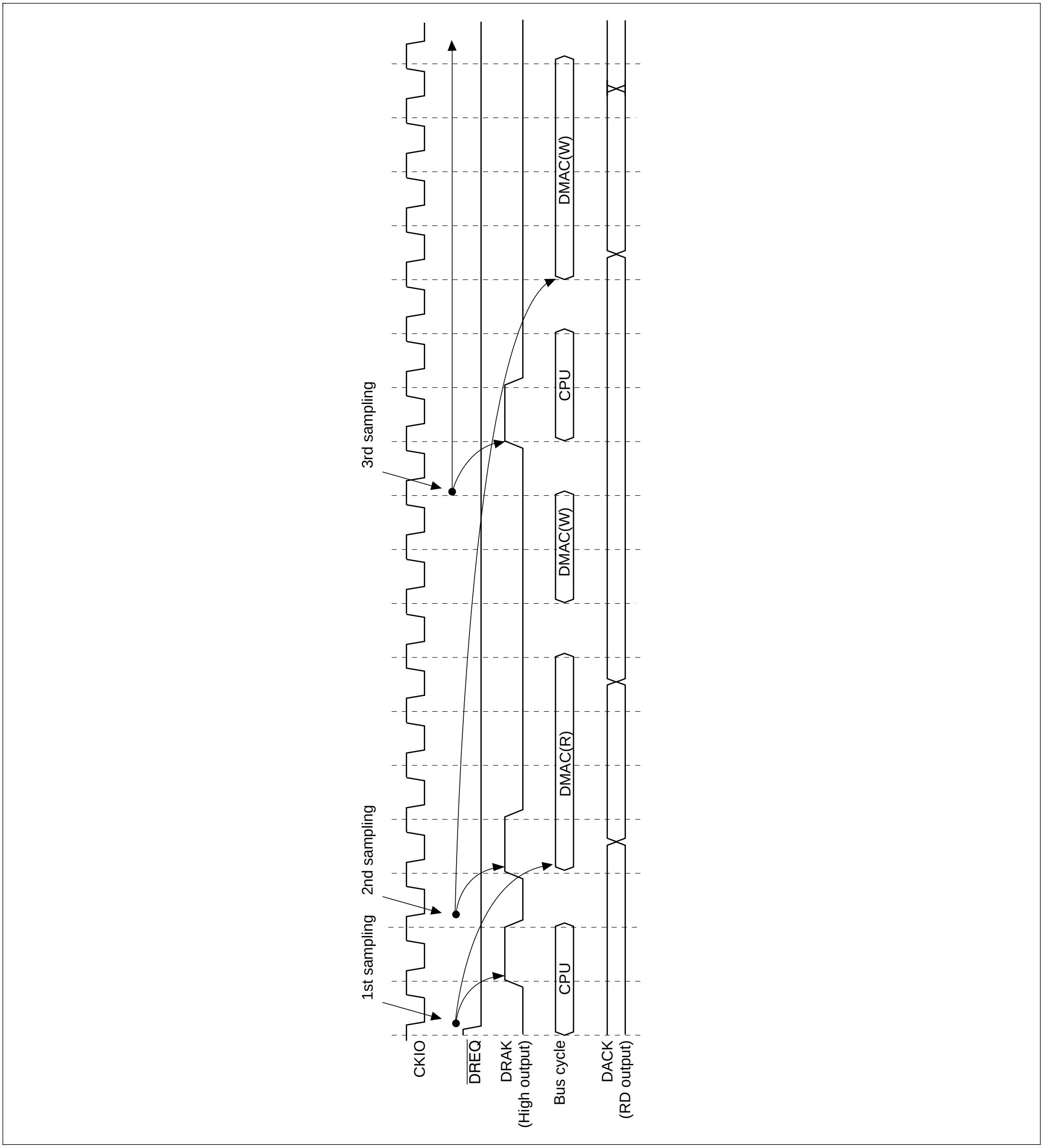


**Figure 14.20 Cycle-Steal Mode, Level Input
(CPU Access: 2 Cycles, DMA RD Access: 4 Cycles)**

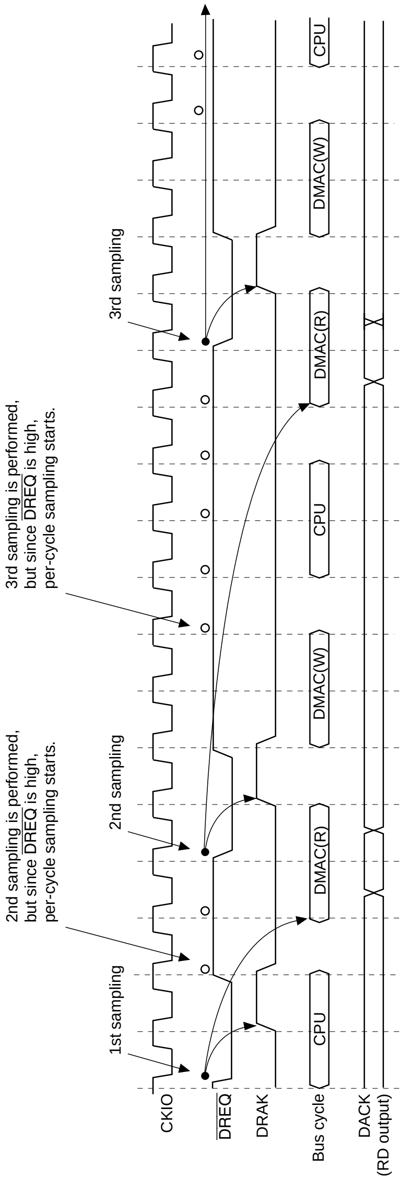
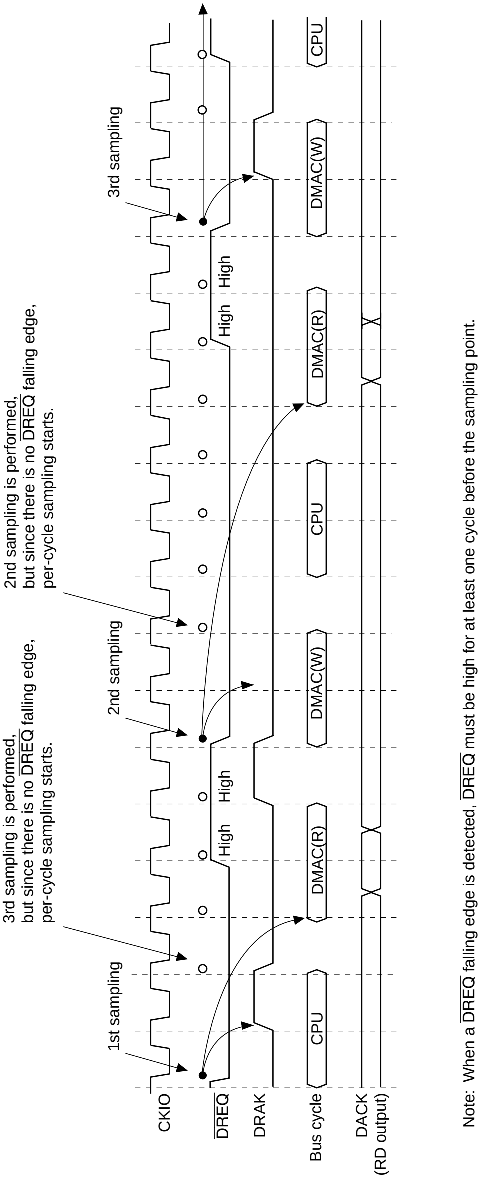


Figure 14.21 Cycle-Steal Mode, Level Input (CPU Access: 2 Cycles, DREQ Input Delayed)



3rd sampling is performed, but since there is no $\overline{\text{DREQ}}$ falling edge, per-cycle sampling starts.

2nd sampling is performed, but since there is no $\overline{\text{DREQ}}$ falling edge, per-cycle sampling starts.

Note: When a $\overline{\text{DREQ}}$ falling edge is detected, $\overline{\text{DREQ}}$ must be high for at least one cycle before the sampling point.

Figure 14.22 Cycle-Steal Mode, Edge Input (CPU Access: 2 Cycles)

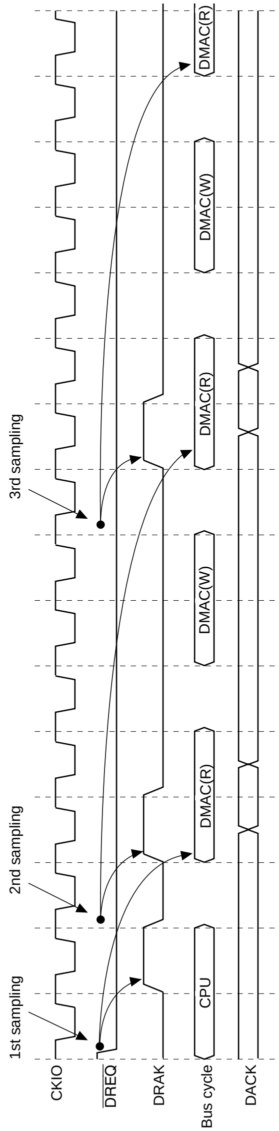


Figure 14.23 Burst Mode, Level Input

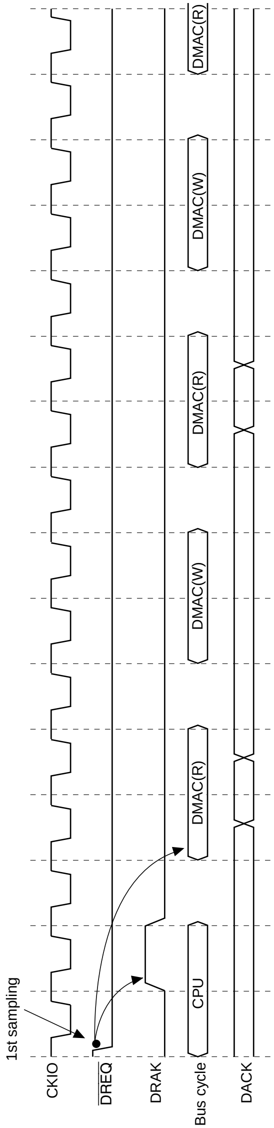


Figure 14.24 Burst Mode, Edge Input

14.3.6 Source Address Reload Function

Channel 2 includes a reload function, in which the value returns to the value set in the source address register (SAR2) for each four transfers by setting the RO bit in CHCR2. 16-byte transfer cannot be used. Figure 14.25 shows this operation. Figure 14.26 shows the timing chart of the source address reload function, which is under the following conditions: burst mode, auto request, 16-bit transfer data size, SAR2 count-up, DAR2 fixed, reload function on, and usage of only channel 2.

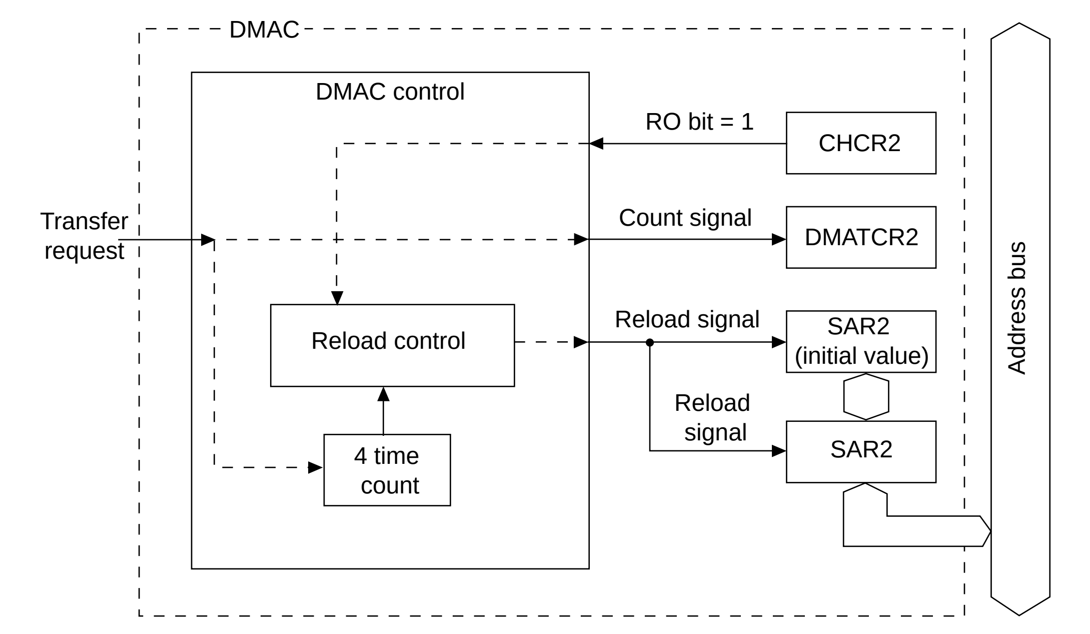


Figure 14.25 Source Address Reload Function Diagram

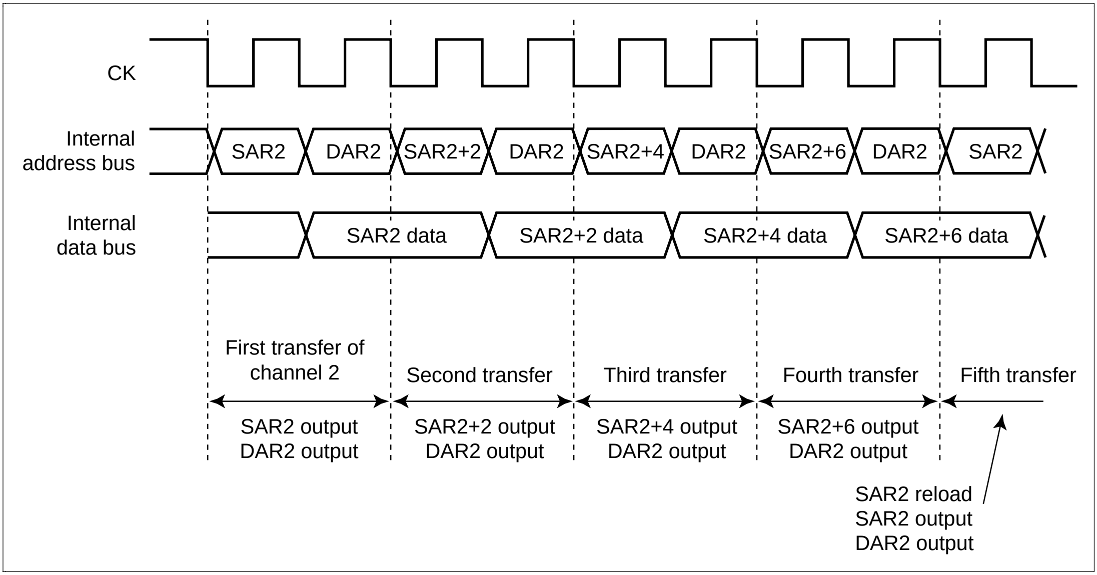


Figure 14.26 Timing Chart of Source Address Reload Function

Even if the transfer data size is 8, 16, or 32 bits, a reload function can be executed.

DMATCR2, which specifies a transfer count, increments 1 each time a transfer ends regardless of whether a reload function is on or off. Consequently, be sure to specify the value multiple of four in DMATCR2 when the reload function is on. Specifying other values does not guarantee the operation.

Though the counters that count transfers of four times for the reload function are reset by clearing the DME bit in DMAOR or the DE bit in CHCR2, by setting the transfer end flag (TE bit in CHCR2), by inputting NMI, besides by reset or standby, the SAR2, DAR2, DMATCR2 registers are not reset. Therefore, if these sources are generated, the counters that are initialized and are not initialized exist in the DMAC; malfunction will be caused by restarting the DMAC in that state. Consequently, if these sources occur except for setting the TE bit during the usage of the reload function, set SAR2, DAR2, and DMATCR2 again.

14.3.7 DMA Transfer Ending Conditions

The DMA transfer ending conditions vary for individual channels ending and all channels ending together. At transfer end, the following conditions are applied except the case where the value set in the DMA transfer count register (DMATCR) reaches 0.

(a) Cycle-steal mode (external request, internal request, and auto request)

When the transfer ending conditions are satisfied, DMAC transfer request acceptance is suspended. The DMAC stops operating after completing the number of transfers that it has accepted until the ending conditions are satisfied.

In the cycle-steal mode, the operation is the same regardless of whether the transfer request is detected by the level or at the edge.

(b) Burst mode, edge detection (external request, internal request, and auto request)

The timing from the point where the ending conditions are satisfied to the point where the DMAC stops operating does not differ from that in cycle steal mode. In the edge detection in the burst mode, though only one transfer request is generated to start up the DMAC, stop request sampling is performed in the same timing as transfer request sampling in the cycle-steal mode. As a result, the period when stop request is not sampled is regarded as the period when transfer request is generated, and after performing the DMA transfer for this period, the DMAC stops operating.

(c) Burst mode, level detection (external request)

Same as described in (a).

(d) Bus timing when transfers are suspended

The transfer is suspended when one transfer ends. Even if transfer ending conditions are satisfied during read in the direct address transfer in the dual address mode, the subsequent write process is executed, and after the transfer in (a) to (c) above has been executed, DMAC operation suspends.

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

- When DMATCR is 0: When the DMATCR value becomes 0 and the corresponding channel's DMA transfer ends, the transfer end flag bit (TE) is set in the CHCR. If the IE (interrupt enable) bit has been set, a DMAC interrupt (DEI) is requested to the CPU. This transfer ending does not apply to (a) to (d) described in above.

- When DE of CHCR is 0: Software can halt a DMA transfer by clearing the DE bit in the channel’s CHCR. The TE bit is not set when this happens. This transfer ending applies to (a) to (d) described above.

Conditions for Ending All Channels Simultaneously: Transfers on all channels end (1) when the NMIF (NMI flag) bit is set to 1 in the DMAOR, or (2) when the DME bit in the DMAOR is cleared to 0.

- Transfers ending when the NMIF bit is set to 1 in DMAOR: When an NMI interrupt occurs, the NMIF bit is set to 1 in the DMAOR and all channels stop their transfers according to the conditions in (a) to (d) described in 14.3.7, and pass the bus right to other bus masters. Consequently, even if the NMI bit is set to 1 during transfer, the SAR, DAR, DMATCR are updated. The TE bit is not set. To resume the transfers after NMI interrupt exception processing, clear the NMIF bit to 0. At this time, if there are channels that should not be restarted, clear the corresponding DE bit in the CHCR.

- Transfers ending when DME is cleared to 0 in DMAOR: Clearing the DME bit to 0 in the DMAOR forcibly aborts the transfers on all channels. The TE bit is not set. All channels aborts their transfers according to the conditions in (a) to (d) in 14.3.7 Transfer Ending Condition, as in NMI interrupt generation. In this case, the values in SAR, DAR, and DMATCR are also updated.

14.4 Compare Match Timer 0 (CMT0)

14.4.1 Overview

DMAC has an on-chip compare match timer 0 (CMT0) to generate DMA transfer request. The CMT has 16-bit counter.

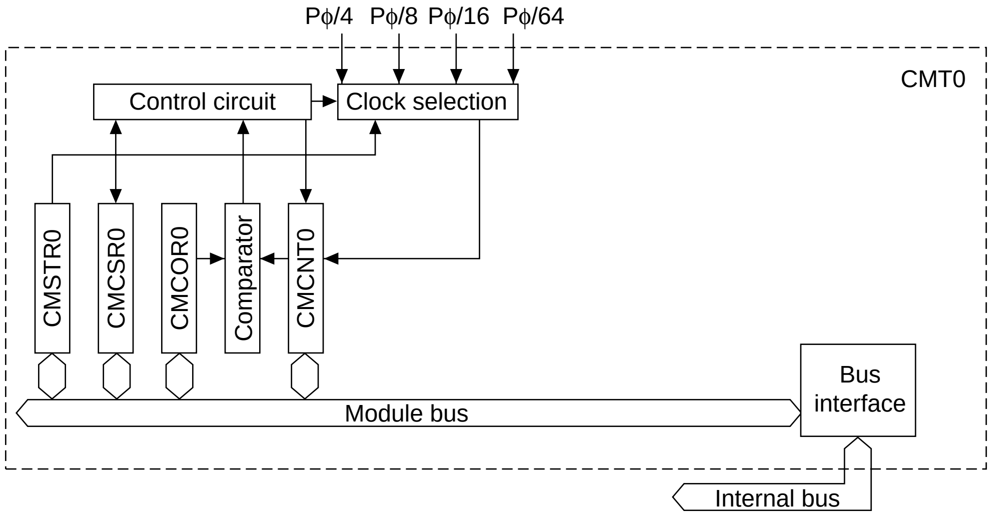
Features

The CMT has the following features:

- Four types of counter input clock can be selected.
  — One of four internal clocks (Pϕ/4, Pϕ/8, Pϕ/16, Pϕ/64) can be selected.
- Generates DMA transfer request when compare match occurs.

Block Diagram

Figure 14.27 shows a CMT0 block diagram.



CMSTR0: Compare match timer start register 0
CMCSR0: Compare match timer control/status register 0
CMCOR0: Compare match timer constant register 0
CMCNT0: Compare match timer counter 0

Figure 14.27 CMT0 Block Diagram

Register Configuration

Table 14.7 summarizes the CMT register configuration.

Table 14.7 Register Configuration

Name	Abbreviation	R/W	Initial Value	Address	Access Size (Bits)
Compare match timer start register 0	CMSTR0	R/(W)	H'0000	H'A4000070	8, 16, 32
Compare match timer control/status register 0	CMCSR0	R/(W)*	H'0000	H'A4000072	8, 16, 32
Compare match counter 0	CMCNT0	R/W	H'0000	H'A4000074	8, 16, 32
Compare match constant register 0	CMCOR0	R/W	H'FFFF	H'A4000076	8, 16, 32

Note: * The only value that can be written to CMF bits in CMCSR0 is a 0 to clear the flags.

14.4.2 Register Descriptions

Compare Match Timer Start Register 0 (CMSTR0)

The compare match timer start register (CMSTR) is a 16-bit register that selects whether compare match counter 0 (CMCNT0) is operated or halted. It is initialized to H'0000 by resets. It retains its previous value in standby mode.

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

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	STR0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

Bits 15 to 1—Reserved: These bits are always read as 0. The write value should alway be 0.

Bit 0—Count Start 0 (STR0): Selects whether to operate or halt compare match timer counter 0.

Bit 0: STR0	Description	
0	CMCNT0 count operation halted	(Initial value)
1	CMCNT0 count operation	

Compare Match Timer Control/Status Register 0 (CMCSR0)

The compare match timer control/status register 0 (CMCSR0) is a 16-bit register that indicates the occurrence of compare matches and establishes the clock used for incrementation. It is initialized to H'0000 by resets. It retains its previous value in standby mode.

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

Bit:	7	6	5	4	3	2	1	0
	CMF	—	—	—	—	—	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R	R	R	R	R	R/W	R/W

Note: * The only value that can be written is 0 to clear the flag.

Bits 15 to 8 and 6 to 2—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 7—Compare Match Flag (CMF): This flag indicates whether or not the compare match timer counter 0 (CMCNT0) and compare match timer constant 0 (CMCOR0) values have matched.

Bit 7: CMF	Description	
0	CMCNT0 and CMCOR0 values have not matched [Clearing condition] Write 0 to CMF after reading CMF = 1	(Initial value)
1	CMCNT0 and CMCOR0 values have matched	

Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0): These bits select the clock input to the CMCNT from the four internal clocks obtained by dividing the system clock (Pϕ). When the STR bit of the CMSTR is set to 1, the CMCNT0 begins incrementing with the clock selected by CKS1 and CKS0.

Bit 1: CKS1	Bit 0: CKS0	Description	
0	0	P ϕ/4	(Initial value)
	1	P ϕ/8	
1	0	P ϕ/16	
	1	P ϕ/64	

Compare Match Counter 0 (CMCNT0)

The compare match counter 0 (CMCNT0) is a 16-bit register used as an up-counter.

When an internal clock is selected with the CKS1 and CKS0 bits of the CMCSR0 register and the STR bit of the CMSTR is set to 1, the CMCNT0 begins incrementing with that clock. When the CMCNT0 value matches that of the compare match constant register 0 (CMCOR0), the CMCNT0 is cleared to H'0000 and the CMF flag of the CMCSR0 is set to 1.

The CMCNT0 is initialized to H'0000 by resets. It retains its previous value in standby mode.

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

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

Compare Match Constant Register 0 (CMCOR0)

The compare match constant register 0 (CMCOR0) is a 16-bit register that sets the compare match period with the CMCNT0.

The CMCOR0 is initialized to H'FFFF by resets. It retains its previous value in standby mode.

Bit:	15	14	13	12	11	10	9	8
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

14.4.3 Operation

Period Count Operation

When an internal clock is selected with the CKS1, CKS0 bits of the CMCSR0 register and the STR bit of the CMSTR is set to 1, the CMCNT0 begins incrementing with the selected clock. When the CMCNT counter value matches that of the CMCOR0, the CMCNT0 counter is cleared to H'0000 and the CMF flag of the CMCSR0 register is set to 1. The CMCNT0 counter begins counting up again from H'0000.

Figure 14.28 shows the compare match counter operation.

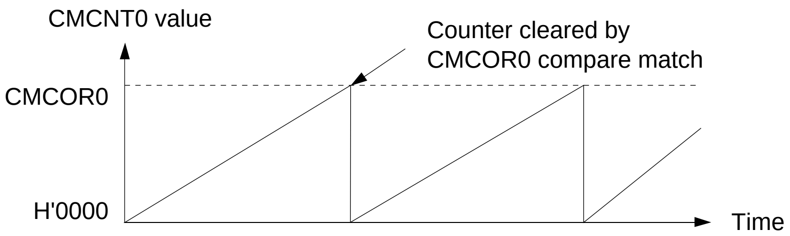


Figure 14.28 Counter Operation

CMCNT0 Count Timing

One of four clocks (Pϕ/4, Pϕ/8, Pϕ/16, Pϕ/64) obtained by dividing the clock (Pϕ) can be selected by the CKS1 and CKS0 bits of the CMCSR0. Figure 14.29 shows the timing.

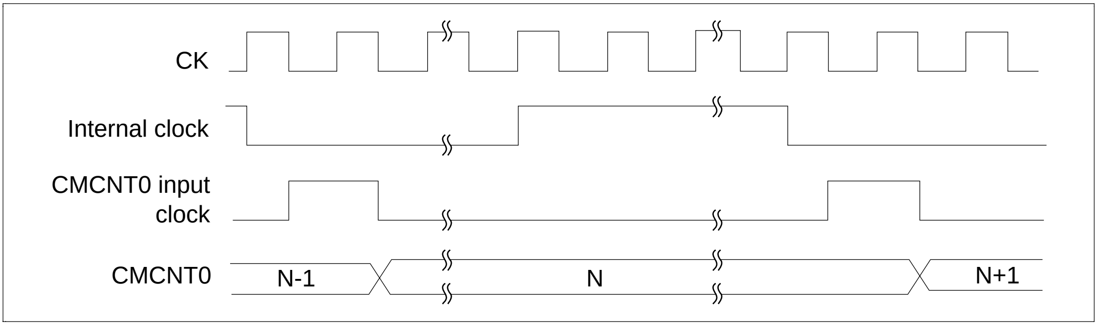


Figure 14.29 Count Timing

14.4.4 Compare Match

Compare Match Flag Set Timing

The CMF bit of the CMCSR0 register is set to 1 by the compare match signal generated when the CMCOR0 register and the CMCNT0 counter match. The compare match signal is generated upon the final state of the match (timing at which the CMCNT0 counter matching count value is updated). Consequently, after the CMCOR0 register and the CMCNT0 counter match, a compare match signal will not be generated until a CMCNT0 counter input clock occurs. Figure 14.30 shows the CMF bit set timing.

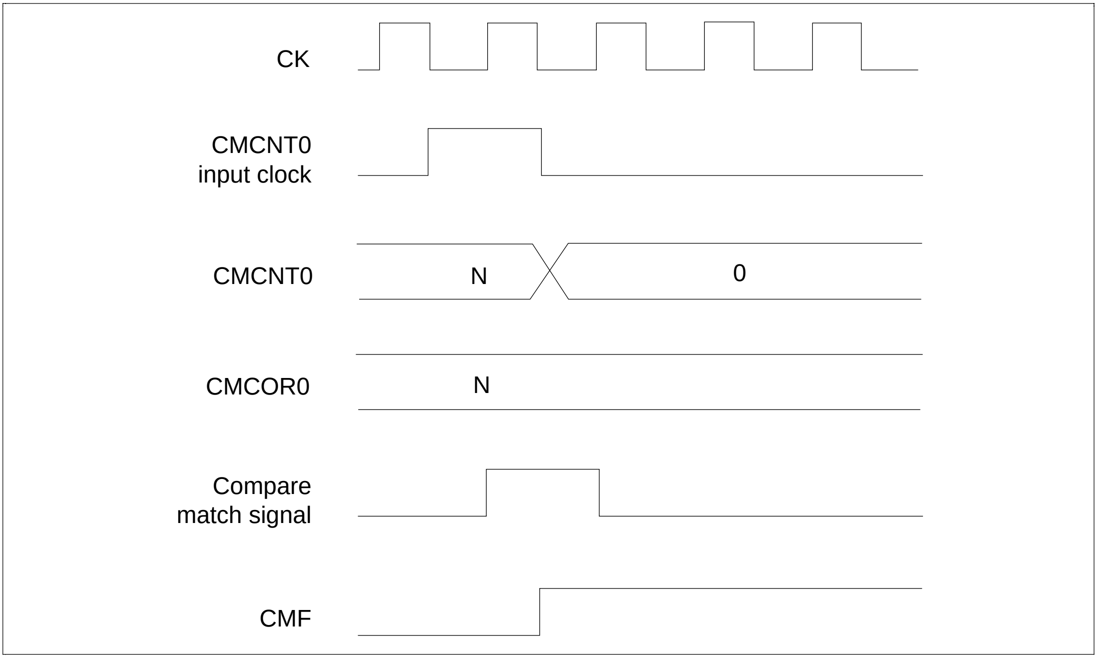


Figure 14.30 CMF Set Timing

Compare Match Flag Clear Timing

The CMF bit of the CMCSR0 register is cleared by writing 0 to it after reading 1. Figure 14.31 shows the timing when the CMF bit is cleared by the CPU.

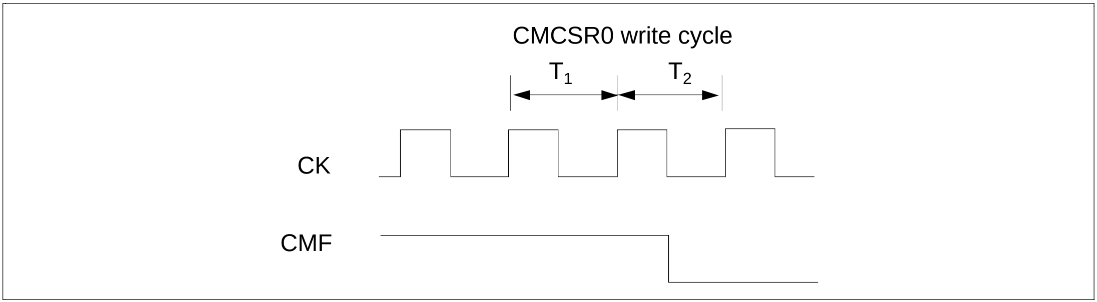


Figure 14.31 Timing of CMF Clear by the CPU

14.5 Examples of Use

14.5.1 Example of DMA Transfer between On-Chip SCIF0 and External Memory

In this example, receive data of on-chip SCIF0 is transferred to external memory using DMAC channel 3. Table 14.8 shows the transfer conditions and register settings. In addition, it is recommended that the trigger of the number of receive FIFO data in IrDA is set to 1 (RTRG1 = RTRG0 = 0 in SCFCR).

Table 14.8 Transfer Conditions and Register Settings for Transfer between On-Chip SCI and External Memory

Transfer Conditions	Register	Setting
Transfer source: SCFRDF0 of on-chip SCIF0	SAR3	H'0400014A
Transfer destination: external memory	DAR3	H'00400000
Number of transfers: 64	DMATCR3	H'00000040
Transfer source address: fixed	CHCR3	H'00004B05
Transfer destination address: incremented		
Transfer request source: SCIF0 (RXI0)		
Bus mode: cycle steal		
Transfer unit: byte		
Interrupt request generated at end of transfer		
Channel priority order: 0 > 2 > 3 > 1	DMAOR	H'0101

14.5.2 Example of DMA Transfer between A/D Converter and External Memory (Address Reload on)

In this example, DMA transfer is performed between the on-chip A/D converter (transfer source) and the external memory (transfer destination) with address reload function on. Table 14.9 shows the transfer conditions and register settings.

Table 14.9 Transfer Conditions and Register Settings for Transfer between On-Chip A/D Converter and External Memory

Transfer Conditions	Register	Setting
Transfer source: on-chip A/D converter	SAR2	H'04000080
Transfer destination: external memory	DAR2	H'00400000
Number of transfers: 128 (reloading 32 times)	DMATCR2	H'00000080
Transfer source address: incremented	CHCR2	H'00089E35
Transfer destination address: decremented		
Transfer request source: A/D converter		
Bus mode: burst		
Transfer unit: long word		
Interrupt request generated at end of transfer		
Channel priority order: 0 > 2 > 3 > 1	DMAOR	H'0101

When the address reload function is on, the value set in SAR returns to the initially set value at each four transfers. In this example, when an interrupt request is generated from AD converter, byte data is read from the register in address H'04000080 in A/D converter , and it is written to external memory address H'00400000. Since longword data has been transferred, the values in SAR and DAR are H'04000084 and H'003FFFFC, respectively. The bus right is maintained and data transfers are successively performed because this transfer is in the burst mode.

After four transfers end, fifth and sixth transfers are performed if the address reload function is off, and the value in SAR is incremented from H'0400008C, H'04000090, H'04000094,.... If the address reload function is on, the DMA transfer stops after the fourth transfer ends, the bus request signal to the CPU is cleared. At this time, the value stored in SAR is not incremented from H'0400008C to H'04000090, but returns to the initially set value H'04000080. The value in DAR continues being incremented regardless of whether the address reload function is on or off.

As a result, the values in the DMAC are as shown in table 14.10 when the fourth transfer ends, depending on whether the address reload function is on or off.

Table 14.10 Values in the DMAC after the Fourth Transfer Ends

Items	Address reload on	Address reload off
SAR	H'04000080	H'04000090
DAR	H'003FFFFC	H'003FFFFC
DMATCR	H'0000007C	H'0000007C
Bus right	Released	Held
DMAC operation	Stops	Keeps operating
Interrupt	Not generated	Not generated
Transfer request source flag clear	Executed	Not executed

Notes: 1. An interrupt is generated regardless of whether the address reload function is on or off, if transfers are executed until the value in DMATCR reaches 0 and the IE bit in CHCR has been set to 1.

2. The transfer request source flag is cleared regardless of whether the address reload function is on or off, if transfers are executed until the value in DMATCR reaches 0.

3. Specify the burst mode to use the address reload function. This function may not be correctly executed in the cycle steal mode.

4. Set the value multiple of four in DMATCR to use the address reload function. This function may not be correctly executed if other values are specified.

14.6 Cautions

1. The DMA channel control registers (CHCR0–CHCR3) can be accessed in any data size. The DMA operation register (DMAOR) must be accessed in byte (8 bits) or word (16 bits); other registers must be accessed in word (16 bits) or longword (32 bits).
2. Before rewriting the RS0–RS3 bits of CHCR0–CHCR3, first clear the DE bit to 0 (when rewriting CHCR with a byte address, be sure to set the DE bit to 0 in advance).
3. Even when the NMI interrupt is input when the DMAC is not operating, the NMIF bit of the DMAOR will be set.
4. When entering the standby mode, the DME bit in DMAOR must be cleared to 0 and the transfers accepted by the DMAC must end.
5. The on-chip peripheral modules which the DMAC can access are SCIF0/1/2, the A/D converter, USB, and the I/O ports. Do not use the DMAC to access any other on-chip peripheral modules.
6. When starting up the DMAC, set CHCR or DMAOR last. Specifying other registers last does not guarantee normal operation.
7. Even if the maximum number of transfers is performed in the same channel after the DMATCR count reaches 0 and the DMA transfer ends normally, write 0 to DMATCR. Otherwise, normal DMA transfer may not be performed.
8. When using the address reload function, specify the burst mode as a transfer mode. In the cycle-steal mode, normal DMA transfer may not be performed.
9. When using the address reload function, set the value multiple of four in DMATCR. Specifying other values does not guarantee normal operation.
10. When detecting an external request at the falling edge, keep the external request pin high when setting the DMAC.
11. Do not access the space ranging from H'A4000062 to H'A400006F, which is not used in the DMAC. Accessing that space may cause malfunctions.
12. When the X/Y memory is designated as the transfer source or transfer destination in DMA transfer, ensure that the X/Y memory is not accessed using the CPU or DSP during DMA transfer.
13. When a transfer size of 16 bytes is set for the DMAC, do not use the external wait ($\overline{\text{WAIT}}$) signal when the SRAM interface is designated as the transfer source.
14. Do not perform accesses to registers of the following modules simultaneously during DMA transfer, as operation cannot be guaranteed in this case.
    - Bus state controller (BSC)
    - Hitachi user debug interface (H-UDI)
    - Clock pulse generator (CPG)

Section 15 Timer (TMU)

15.1 Overview

This LSI uses a three-channel (channel 0–2) 32-bit timer unit (TMU).

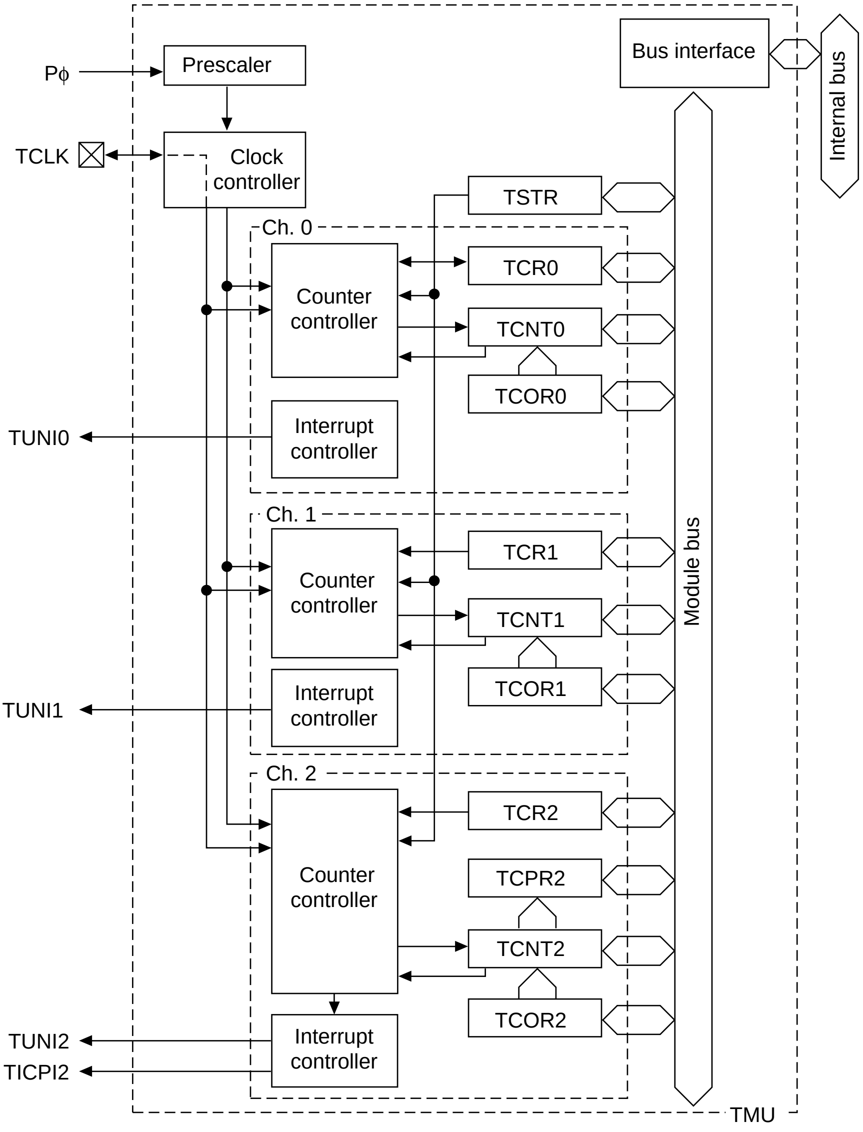
15.1.1 Features

The TMU has the following features:

- Each channel is provided with an auto-reload 32-bit down counter.
- Channel 2 is provided with an input capture function.
- All channels are provided with 32-bit constant registers and 32-bit down counters that can be read or written to at any time.
- All channels generate interrupt requests when the 32-bit down counter underflows (H'00000000 → H'FFFFFFFF).
- Allows selection between 5 counter input clocks: External clock (TCLK), Pϕ/4, Pϕ/16, Pϕ/64, Pϕ/256 (Pϕ is the internal clock for peripheral modules and can be selected as 1/4, 1/2, or the same frequency as that of the CPU operating clock ϕ. See section 11, Clock Pulse Generator (CPG), for more information on the clock pulse generator).
- Synchronized read: TCNT is a sequentially changing 32-bit register. Since the peripheral module used has an internal bus width of 16 bits, a time lag can occur between the time when the upper 16 bits and lower 16 bits are read. To correct the discrepancy in the counter read value caused by this time lag, a synchronization circuit is built into the TCNT so that the entire 32-bit data in the TCNT can be read at once.
- The maximum operating frequency of the 32-bit counter is 2 MHz on all channels: Operate the SH7622 so that the clock input to the timer counters of each channel (obtained by dividing the external clock and internal clock with the prescaler) does not exceed the maximum operating frequency.

15.1.2 Block Diagram

Figure 15.1 shows a block diagram of the TMU.



Legend

TSTR: Timer start register
TCR: Timer control register
TCNT: 32-bit timer counter
TCOR: 32-bit timer constant register
TCPR2: 32-bit input capture register

Figure 15.1 TMU Block Diagram

15.1.3 Pin Configuration

Table 15.1 shows the pin configuration of the TMU.

Table 15.1 Pin Configuration

Channel	Pin	I/O	Description
Clock input/clock output	TCLK	I/O	External clock input pin/input capture control input pin

15.1.4 Register Configuration

Table 15.2 shows the TMU register configuration.

Table 15.2 TMU Register Configuration

Channel	Register	Abbreviation	R/W	Initial Value*	Address	Access Size
Common	Timer start register	TSTR	R/W	H'00	H'FFFFFE92	8
0	Timer constant register 0	TCOR0	R/W	H'FFFFFFFF	H'FFFFFE94	32
	Timer counter 0	TCNT0	R/W	H'FFFFFFFF	H'FFFFFE98	32
	Timer control register 0	TCR0	R/W	H'0000	H'FFFFFE9C	16
1	Timer constant register 1	TCOR1	R/W	H'FFFFFFFF	H'FFFFFEA0	32
	Timer counter 1	TCNT1	R/W	H'FFFFFFFF	H'FFFFFEA4	32
	Timer control register 1	TCR1	R/W	H'0000	H'FFFFFEA8	16
2	Timer constant register 2	TCOR2	R/W	H'FFFFFFFF	H'FFFFFEAC	32
	Timer counter 2	TCNT2	R/W	H'FFFFFFFF	H'FFFFFEB0	32
	Timer control register 2	TCR2	R/W	H'0000	H'FFFFFEB4	16
	Input capture register 2	TCPR2	R	Undefined	H'FFFFFEB8	32

Note: * Initialized by power-on resets or manual resets.

15.2 TMU Registers

15.2.1 Timer Start Register (TSTR)

TSTR is an 8-bit read/write register that selects whether to run or halt the timer counters (TCNT) for channels 0–2. TSTR is initialized to H'00 by a power-on reset or manual reset, but is not initialized in standby mode when the input clock selected for the channel is the on-chip RTC clock (RTCCLK). It is initialized in standby mode, changing the multiplying ratio of PLL circuit 1 or MSTP2 bit in STBCR is set to a logic one only when an external clock (TCLK) or the peripheral clock (Pϕ) is used as the input clock.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	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. The write value should always be 0.

Bit 2—Counter Start 2 (STR2): Selects whether to run or halt timer counter 2 (TCNT2).

Bit 2: STR2	Description	
0	Halt TCNT2 count	(Initial value)
1	Start TCNT2 counting	

Bit 1—Counter Start 1 (STR1): Selects whether to run or halt timer counter 1 (TCNT1).

Bit 1: STR1	Description	
0	Halt TCNT1 count	(Initial value)
1	Start TCNT1 counting	

Bit 0—Counter Start 0 (STR0): Selects whether to run or halt timer counter 0 (TCNT0).

Bit 0: STR0	Description	
0	Halt TCNT0 count	(Initial value)
1	Start TCNT0 counting	

15.2.2 Timer Control Register (TCR)

The timer control registers (TCR) control the timer counters (TCNT) and interrupts. The TMU has three TCR registers for each channel.

The TCR registers are 16-bit read/write registers that control the issuance of interrupts when the flag indicating timer counter (TCNT) underflow has been set to 1, and also carry out counter clock selection. When the external clock has been selected, they also select its edge. Additionally, TCR2 controls the channel 2 input capture function and the issuance of interrupts during input capture. The TCRs are initialized to H'0000 by a power-on reset and manual reset. They are not initialized in standby mode.

Channel 0 and 1 TCR Bit Configuration

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	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

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	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 10, 9 (except TCR2), 7, and 6 (except TCR2)—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 9—Input Capture Interrupt Flag (ICPF): A function of channel 2 only: the flag is set when input capture is requested via the TCLK pin.

Bit 9: ICPF	Description
0	No input capture request has been issued [Clearing condition] When 0 is written to ICPF (Initial value)
1	Input capture has been requested via the TCLK pin [Setting condition] When an input capture is requested via the TCLK pin*

Note: * Contents do not change when 1 is written to ICPF.

Bit 8—Underflow Flag (UNF): Status flag that indicates occurrence of a TCNT underflow.

Bit 8: UNF	Description
0	TCNT has not underflowed [Clearing condition] When 0 is written to UNF (Initial value)
1	TCNT has underflowed (H'00000000 → H'FFFFFFFF) [Setting condition] When TCNT underflows*

Note: * Contents do not change when 1 is written to UNF.

Bits 7 and 6—Input Capture Control (ICPE1, ICPE0): A function of channel 2 only: determines whether the input capture function can be used, and when used, whether or not to enable interrupts.

When using this input capture function it is necessary to set the TCLK pin to input mode with the TCOE bit in the TOCR register. Additionally, use the CKEG bit to designate use of either the rising or falling edge of the TCLK pin to set the value in TCNT2 in the input capture register (TCPR2).

Bit 7: ICPE1	Bit 6: ICPE0	Description
0	0	Input capture function is not used (Initial value)
	1	Reserved (Setting disabled)
1	0	Input capture function is used. Interrupt due to ICPF (TICPI2) is not enabled
	1	Input capture function is used. Interrupt due to ICPF (TICPI2) is enabled

Bit 5—Underflow Interrupt Control (UNIE): Controls enabling of interrupt generation when the status flag (UNF) indicating TCNT underflow has been set to 1.

Bit 5: UNIE	Description	
0	Interrupt due to UNF (TUNI) is not enabled	(Initial value)
1	Interrupt due to UNF (TUNI) is enabled	

Bits 4 and 3—Clock Edge 1 and 0 (CKEG1, CKEG0): These bits select the external clock edge when the external clock is selected, or when the input capture function is used.

Bit 4: CKEG1	Bit 3: CKEG0	Description	
0	0	Count/capture register set on rising edge	(Initial value)
	1	Count/capture register set on falling edge	
1	X	Count/capture register set on both rising and falling edge	

Note: X means 0, 1, or 'don't care'.

Bits 2 to 0—Timer Prescalers 2–0 (TPSC2–TPSC0): These bits select the TCNT count clock.

Bit 2: TPSC2	Bit 1: TPSC1	Bit 0: TPSC0	Description	
0	0	0	Internal clock: count on Pϕ/4	(Initial value)
		1	Internal clock: count on Pϕ/16	
	1	0	Internal clock: count on Pϕ/64	
		1	Internal clock: count on Pϕ/256	
1	0	0	Reserved (Setting disabled)	
		1	External clock: count on TCLK pin input	
	1	0	Reserved (Setting disabled)	
		1	Reserved (Setting disabled)	

15.2.3 Timer Constant Register (TCOR)

The timer constant registers are 32-bit registers. The TMU has three TCOR registers, one for each of the three channels.

TCOR is a 32-bit read/write register. When a TCNT count-down results in an underflow, the TCOR value is set in TCNT and the count-down continues from that value. TCOR is initialized to H'FFFFFFFF by a power-on reset or manual reset; it is not initialized in standby mode, and retains its contents.

Bit:	31	30	29	28	27	26	25	24
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
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

15.2.4 Timer Counters (TCNT)

The timer counters are 32-bit read/write registers. The TMU has three timer counters, one for each channel.

TCNT counts down upon input of a clock. The clock input is selected using the TPSC2–TPSC0 bits in the timer control register (TCR).

When a TCNT count-down results in an underflow (H'00000000 → H'FFFFFFFF), the underflow flag (UNF) in the timer control register (TCR) of the relevant channel is set. The TCOR value is simultaneously set in TCNT itself and the count-down continues from that value.

Because the internal bus for the SH7622 on-chip supporting modules is 16 bits wide, a time lag can occur between the time when the upper 16 bits and lower 16 bits are read. Since TCNT counts sequentially, this time lag can create discrepancies between the data in the upper and lower halves. To correct the discrepancy, a buffer register is connected to TCNT so that upper and lower halves are not read separately. The entire 32-bit data in TCNT can thus be read at once.

TCNT is initialized to H'FFFFFFFF by a power-on reset or manual reset; it is not initialized in standby mode, and retains its contents.

Bit:	31	30	29	28	27	26	25	24
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
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	15	14	13	12	11	10	9	8
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

15.2.5 Input Capture Register (TCPR2)

The input capture register 2 (TCPR2) is a read-only 32-bit register built only into timer 2. Control of TCPR2 setting conditions due to the TCLK pin is affected by the input capture function bits (ICPE1/ICPE2 and CKEG1/CKEG0) in TCR2. When a TCPR2 setting indication due to the TCLK pin occurs, the value of TCNT2 is copied into TCPR2.

TCNT2 is not initialized by a power-on reset or manual reset, or in standby mode.

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

Bit:	23	22	21	20	19	18	17	16
Initial value:	—	—	—	—	—	—	—	—
R/W:	R	R	R	R	R	R	R	R

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

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

15.3 TMU Operation

15.3.1 Overview

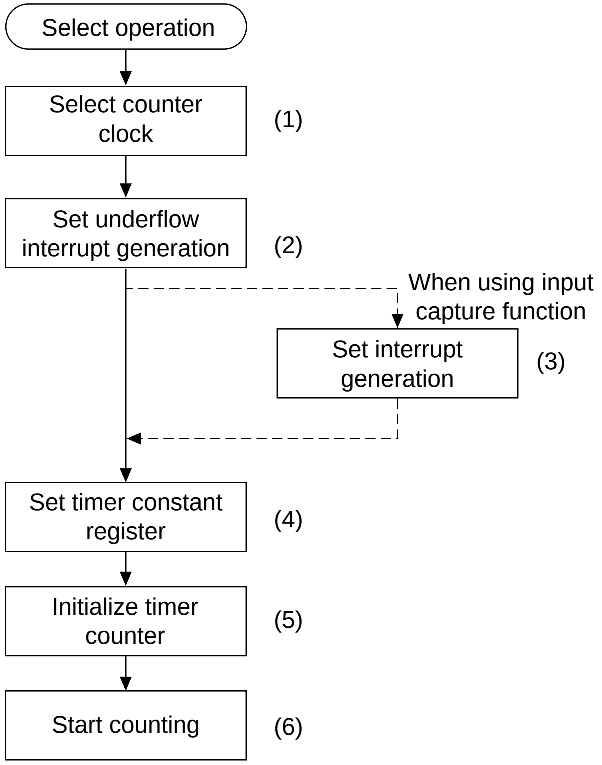
Each of three channels has a 32-bit timer counter (TCNT) and a 32-bit timer constant register (TCOR). The TCNT counts down. The auto-reload function enables synchronized counting and counting by external events. Channel 2 has an input capture function.

15.3.2 Basic Functions

Counter Operation: When the STR0–STR2 bits in the timer start register (TSTR) are set, the corresponding timer counter (TCNT) starts counting. When a TCNT underflows, the UNF flag of the corresponding timer control register (TCR) is set. At this time, if the UNIE bit in TCR is 1, an interrupt request is sent to the CPU. Also at this time, the value is copied from TCOR to TCNT and the down-count operation is continued.

The count operation is set as follows (figure 15.2):

1. Select the counter clock with the TPSC2–TPSC0 bits in the timer control register (TCR). If the external clock is selected, set the TCLK pin to input mode with the TOCE bit in TOCR, and select its edge with the CKEG1 and CKEG0 bits in TCR.
2. Use the UNIE bit in TCR to set whether to generate an interrupt when TCNT underflows.
3. When using the input capture function, set the ICPE bits in TCR, including the choice of whether or not to use the interrupt function (channel 2 only).
4. Set a value in the timer constant register (TCOR) (the cycle is the set value plus 1).
5. Set the initial value in the timer counter (TCNT).
6. Set the STR bit in the timer start register (TSTR) to 1 to start operation.



Note: When an interrupt has been generated, clear the flag in the interrupt handler that caused it. If interrupts are enabled without clearing the flag, another interrupt will be generated.

Figure 15.2 Setting the Count Operation

Auto-Reload Count Operation: Figure 15.3 shows the TCNT auto-reload operation.

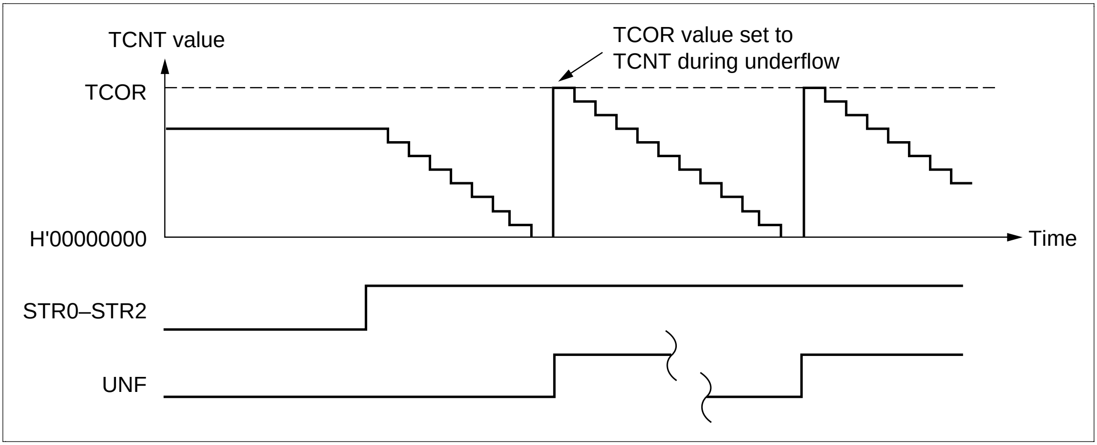


Figure 15.3 Auto-Reload Count Operation

TCNT Count Timing:

- Internal Clock Operation: Set the TPSC2–TPSC0 bits in TCR to select whether peripheral module clock Pϕ or one of the four internal clocks created by dividing it is used (Pϕ/4, Pϕ/16, Pϕ/64, Pϕ/256). Figure 15.4 shows the timing.

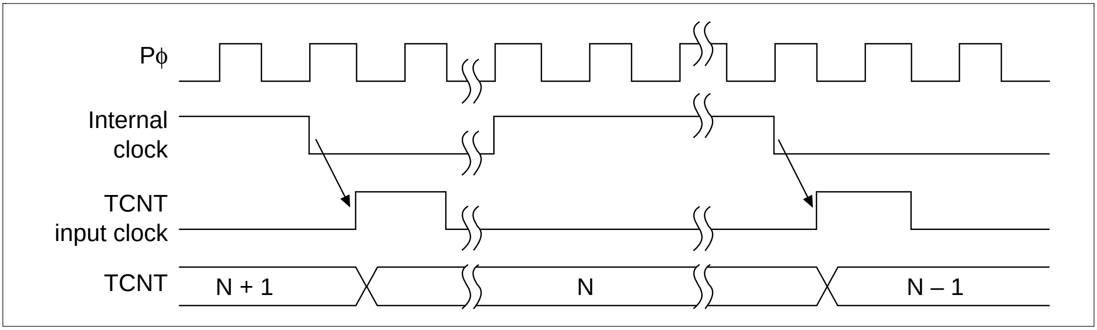


Figure 15.4 Count Timing when Internal Clock is Operating

- External Clock Operation: Set the TPSC2–TPSC0 bits in TCR to select the external clock (TCLK) as the timer clock. Use the CKEG1 and CKEG0 bits in TCR to select the detection edge. Rise, fall or both may be selected. The pulse width of the external clock must be at least 1.5 peripheral module clock cycles for single edges or 2.5 peripheral module clock cycles for both edges. A shorter pulse width will result in accurate operation. Figure 15.5 shows the timing for both-edge detection.

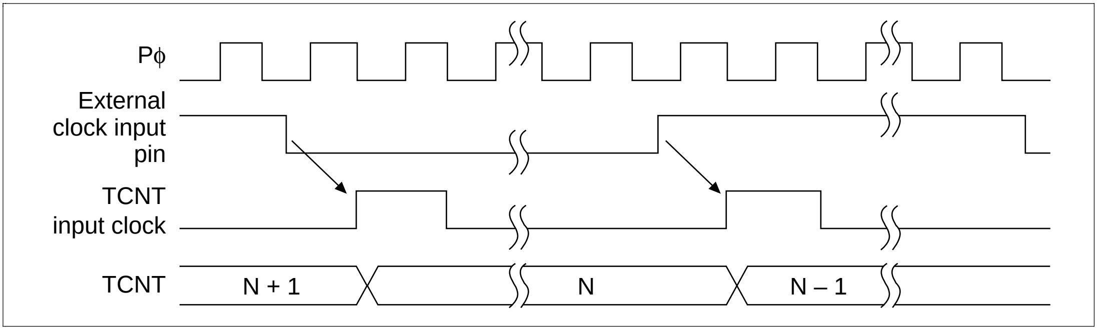


Figure 15.5 Count Timing when External Clock is Operating (Both Edges Detected)

Input Capture Function: Channel 2 has an input capture function (figure 15.6). When using the input capture function, set the TCLK pin to input mode with the TCOE bit in the timer output control register (TOCR) and set the timer operation clock to internal clock or on-chip RTC clock with the TPCS2–TPCS0 bits in the timer control register (TCR2). Also, designate use of the input capture function and whether to generate interrupts on using it with the IPCE1–IPCE0 bits in TCR2, and designate the use of either the rising or falling edge of the TCLK pin to set the timer counter (TCNT2) value into the input capture register (TCPR2) with the CKEG1–CKEG0 bits in TCR2.

The input capture function cannot be used in standby mode.

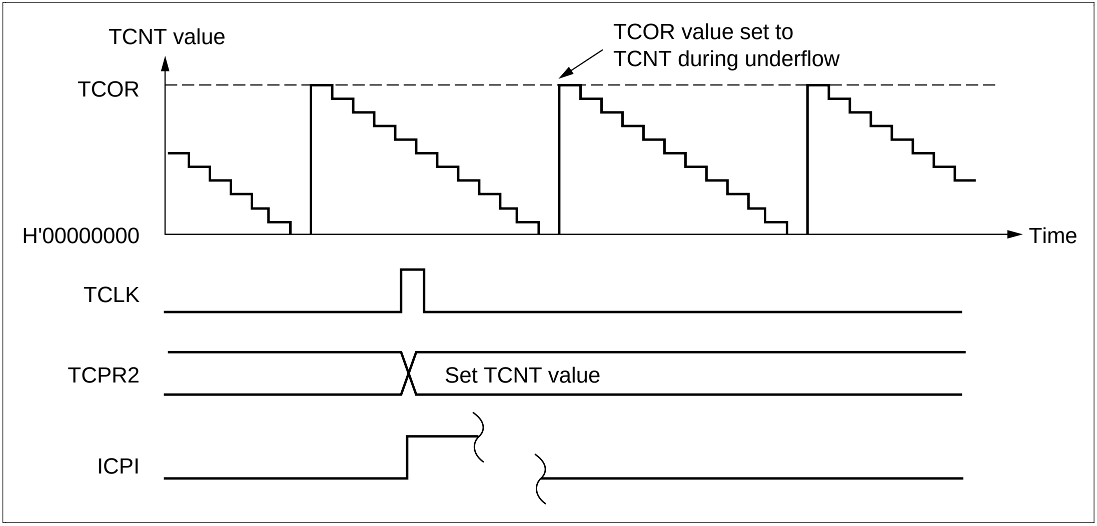


Figure 15.6 Operation Timing when Using the Input Capture Function (Using TCLK Rising Edge)

15.4 Interrupts

There are two sources of TMU interrupts: underflow interrupts (TUNI) and interrupts when using the input capture function (TICPI2).

15.4.1 Status Flag Set Timing

UNF is set to 1 when the TCNT underflows. Figure 15.7 shows the timing.

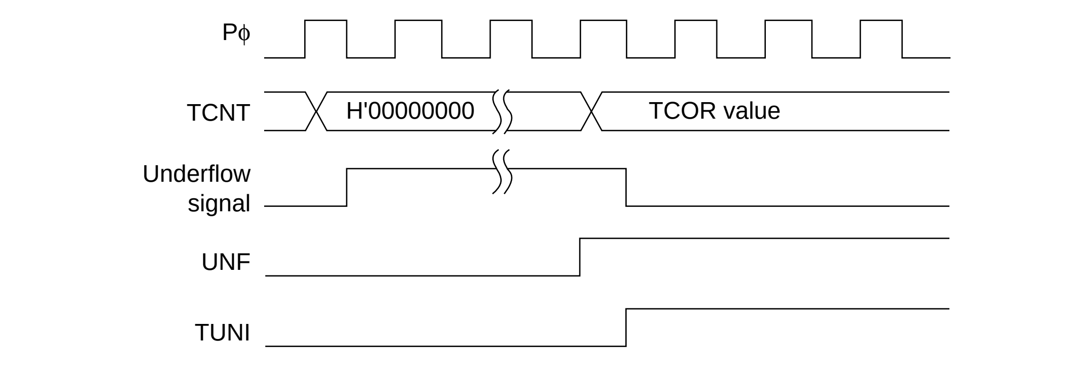


Figure 15.7 UNF Set Timing

15.4.2 Status Flag Clear Timing

The status flag can be cleared by writing 0 from the CPU. Figure 15.8 shows the timing.

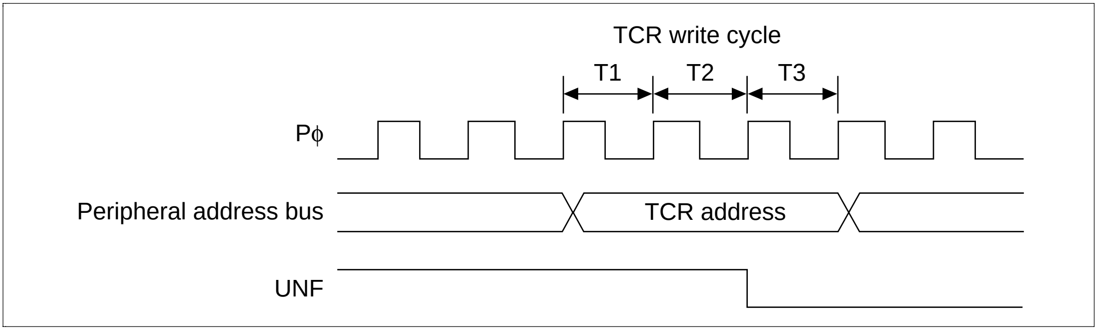


Figure 15.8 Status Flag Clear Timing

15.4.3 Interrupt Sources and Priorities

The TMU produces underflow interrupts for each channel. When the interrupt request flag and interrupt enable bit are both set to 1, the interrupt is requested. Codes are set in the interrupt event register (INTEVT, INTEVT2) for these interrupts and interrupt processing occurs according to the codes.

The relative priorities of channels can be changed using the interrupt controller (see section 5, Exception Processing, and section 8, Interrupt Controller (INTC)). Table 15.3 lists TMU interrupt sources.

Table 15.3 TMU Interrupt Sources

Channel	Interrupt Source	Description	Priority
0	TUNI0	Underflow interrupt 0	High
1	TUNI1	Underflow interrupt 1	↓
2	TUNI2	Underflow interrupt 2	
2	TICPI2	Input capture interrupt 2	Low

15.5 Usage Notes

15.5.1 Writing to Registers

Synchronization processing is not performed for timer counting during register writes. When writing to registers, always clear the appropriate start bits for the channel (STR2–STR0) in the timer start register (TSTR) to halt timer counting.

15.5.2 Reading Registers

Synchronization processing is performed for timer counting during register reads. When timer counting and register read processing are performed simultaneously, the register value before TCNT counting down (with synchronization processing) is read.

Section 16 Serial Communication Interface with FIFO (SCIF0)

16.1 Overview

SCIF0 is a serial communication interface with built-in FIFO buffers (Serial Communication Interface with FIFO: SCIF). SCIF0 can perform synchronous serial communication.

A 128-stage FIFO register is provided for transmission, and a 384-stage FIFO register for reception, enabling fast, efficient, and continuous communication.

16.1.1 Features

SCIF0 features are listed below.

- 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 communication format.

  — Data length: 8 bits

  — LSB-first transfer

  — Receive error detection: Overrun errors
- Full-duplex communication capability

  The transmitter and receiver are independent units, enabling transmission and reception to be performed simultaneously.

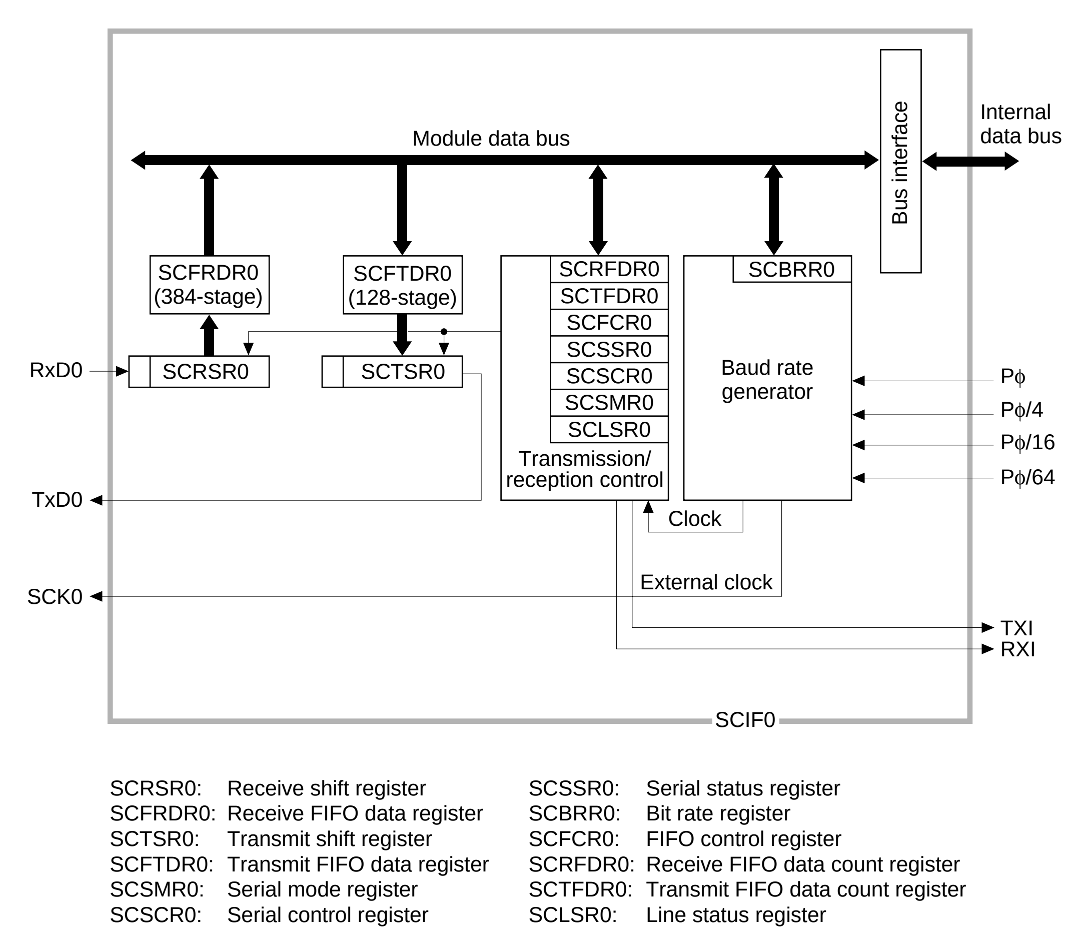
  The transmitter and receiver have a 128-stage and 384-stage FIFO buffer structure, respectively, enabling fast and continuous serial data transmission and reception.
- On-chip baud rate generator allows any bit rate to be selected.
- Choice of serial clock source: internal clock from baud rate generator or external clock from SCK0 pin
- Two interrupt sources

  There are two interrupt sources—the transmit-FIFO-data interrupt, receive-FIFO-data interrupt—that can issue requests independently. There are two kinds of receive-FIFO-data interrupt, discriminated by the value of the FSTE bit in the line status register (SCLSR0): receive-FIFO-data interrupt when FSTE = 0, and receive-data-stop interrupt when FSTE = 1.
- 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 or receive-FIFO-data interrupt.
- When not in use, SCIF0 can be stopped by halting its clock supply to reduce power consumption.
- The amount of data in the transmit/receive FIFO registers can be ascertained.

- The contents of the transmit FIFO data register (SCFTDR0) and receive FIFO data register (SCFRDR0) are undefined after a power-on or manual reset. Other registers are initialized by a power-on or manual reset, and retain their values in standby mode and in the module standby state. For details see section 16.1.4, Register Configuration.

16.1.2 Block Diagram

Figure 16.1 shows a block diagram of SCIF0.



SCRSR0:	Receive shift register	SCSSR0:	Serial status register
SCFRDR0:	Receive FIFO data register	SCBRR0:	Bit rate register
SCTSR0:	Transmit shift register	SCFCR0:	FIFO control register
SCFTDR0:	Transmit FIFO data register	SCRFDR0:	Receive FIFO data count register
SCSMR0:	Serial mode register	SCTFDR0:	Transmit FIFO data count register
SCSCR0:	Serial control register	SCLSR0:	Line status register

Figure 16.1 Block Diagram of SCIF0

16.1.3 Pin Configuration

Table 16.1 shows the SCIF0 pin configuration.

Table 16.1 SCIF0 Pins

	Pin Name	Abbreviation	I/O	Function
SCIF0	Serial clock pin	SCK0	I/O	Clock input/output
	Receive data pin	RxD0	Input	Receive data input
	Transmit data pin	TxD0	Output	Transmit data output

Note: These pins are made to function as serial pins by performing SCIF0 operation settings with the TE and RE bits in SCSCR0.

16.1.4 Register Configuration

SCIF0 has the internal registers shown in table 16.2. These registers are used to specify the bit rate, and to perform transmitter/receiver control.

Table 16.2 SCIF0 Registers

	Name	Abbreviation	R/W	Initial Value	Address	Access Size
SCIF0	Serial mode register	SCSMR0	R/W	H'00	H'A4002000	8
	Bit rate register	SCBRR0	R/W	H'FF	H'A4002002	8
	Serial control register	SCSCR0	R/W	H'00	H'A4002004	8
	Line status register	SCLSR0	R/(W)*2	H’0004	H'A4002006	16
	Serial status register	SCSSR0	R/(W)*1	H'0040	H'A4002008	16
	Receive FIFO data count register	SCRFDR0	R	H'0000	H'A400200A	16
	FIFO control register	SCFCR0	R/W	H'00	H'A400200C	8
	Transmit FIFO data register	SCFTDR0	W	Undefined	H'A4002010	8
	Receive FIFO data register	SCFRDR0	R	Undefined	H'A4002014	8
	Transmit FIFO data count register	SCTFDR0	R	H'0000	H'A4002018	16

Notes: *1 Only 0 can be written to bits 5 and 1, to clear the flags.

*2 Only 0 can be written to bit 0, to clear the flag.

16.2 Register Descriptions

16.2.1 Receive Shift Register (SCRSR0)

SCRSR0 is the register used to receive serial data.

SCIF0 sets serial data input from the RxD0 pin in SCRSR0 in the order received, starting with the LSB (bit 0), and converts it to parallel data. When 1 byte of data has been received, it is transferred to the receive FIFO data register, SCFRDR0, automatically.

SCRSR0 cannot be directly read or written to by the CPU.

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

16.2.2 Receive FIFO Data Register (SCFRDR0)

SCFRDR0 is a 384-stage FIFO register that stores received serial data.

When SCIF0 has received one byte of serial data, it transfers the received data from SCRSR0 to SCFRDR0 where it is stored, and completes the receive operation. SCRSR0 is then enabled for reception, and consecutive receive operations can be performed until the receive FIFO data register is full (384 data bytes).

SCFRDR0 is a read-only register, and cannot be written to by the CPU.

If a read is performed when there is no receive data in the receive FIFO data register, an undefined value will be returned. When the receive FIFO data register is full of receive data, subsequent serial data is lost.

The contents of SCFRDR0 are undefined after a power-on reset or manual reset.

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

16.2.3 Transmit Shift Register (SCTSR0)

SCTSR0 is the register used to transmit serial data.

To perform serial data transmission, SCIF0 first transfers transmit data from SCFTDR0 to SCTSR0, then sends the data to the TxD0 pin starting with the LSB (bit 0).

When transmission of one byte is completed, the next transmit data is transferred from SCFTDR0 to SCTSR0, and transmission started, automatically.

SCTSR0 cannot be directly read or written to by the CPU.

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

16.2.4 Transmit FIFO Data Register (SCFTDR0)

SCFTDR0 is a 128-stage FIFO data register that stores data for serial transmission.

If SCTSR0 is empty when transmit data has been written to SCFTDR0, SCIF0 transfers the transmit data written in SCFTDR0 to SCTSR0 and starts serial transmission.

SCFTDR0 is a write-only register, and cannot be read by the CPU.

The next data cannot be written when SCFTDR0 is filled with 128 bytes of transmit data. Data written in this case is ignored.

The contents of SCFTDR0 are undefined after a power-on reset or manual reset.

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

16.2.5 Serial Mode Register (SCSMR0)

SCSMR0 is an 8-bit register used to select the baud rate generator clock source.

SCSMR0 can be read or written to by the CPU at all times.

SCSMR0 is initialized to H'00 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Bits 7–2—Reserved: These bits are always read as 0. The write value should always be 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 (SCBRR0).

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 (SCSCR0)

The SCSCR0 register performs enabling or disabling of SCIF0 transfer operations and interrupt requests, and selection of the serial clock source.

SCSCR0 can be read or written to by the CPU at all times.

SCSCR0 is initialized to H'00 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

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	R	R/W	R/W

Bit 7—Transmit Interrupt Enable (TIE): Enables or disables transmit-FIFO-data interrupt (TXI) request generation when serial transmit data is transferred from SCFTDR0 to SCTSR0, and the TDFST flag in SCSSR0 is set to 1.

Bit 7: TIE	Description	
0	Transmit-FIFO-data interrupt (TXI) request disabled*	(Initial value)
1	Transmit-FIFO-data interrupt (TXI) request enabled	

Note: * TXI interrupt requests can be cleared by reading 1 from the TDFST flag, then clearing it to 0, or by clearing the TIE bit to 0.

When the transmit-FIFO-data interrupt is used, set the FSTE bit in SCLSR0 to 1.

Bit 6—Receive Interrupt Enable (RIE): Enables or disables generation of a receive-data interrupt (RXI) request when the RDFST flag in SCSSR0 is set to 1.

Bit 6: RIE	Description	
0	Receive-FIFO-data interrupt (RXI) request disabled*	(Initial value)
1	Receive-FIFO-data interrupt (RXI) request enabled	

Note: * An RXI interrupt request can be cleared by reading 1 from the RDFST 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 SCIF0.

Bit 5: TE	Description
0	Transmission disabled (Initial value)
1	Transmission enabled*

Note: * Serial mode register (SCSMR0) and FIFO control register (SCFCR0) settings must be made, the operating clock source 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 SCIF0.

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 RDFST flag, which retains its state.

*2 Serial mode register (SCSMR0) and FIFO control register (SCFCR0) settings must be made, the operating clock source decided, and the receive FIFO reset, before the RE bit is set to 1.

Bits 3 and 2—Reserved: These bits are always read as 0. The write value should always be 0.

Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0): These bits select the SCIF0 clock source. The CKE1 and CKE0 bits must be set before determining the SCIF0 operating mode with SCSMR0.

Bit 1: CKE1	Bit 0: CKE0	Description
0	0	Internal clock/SCK0 pin functions as input pin (input signal ignored) (Initial value)
	1	Internal clock/SCK0 pin functions as serial clock output
1	0	External clock/SCK0 pin functions as clock input
	1	External clock/SCK0 pin functions as clock input

16.2.7 Serial Status Register (SCSSR0)

SCSSR0 is a 16-bit register that can be read or written to by the CPU at all times. However, 1 cannot be written to the TDFST and RDFST flags. Also note that in order to clear these flags they must be read as 1 beforehand.

SCSSR0 is initialized to H'0040 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

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
	—	TEND	TDFST	—	—	—	RDFST	—
Initial value:	0	1	0	0	0	0	0	0
R/W:	R	R	R/(W)*	R	R	R	R/(W)*	R

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

Bits 15 to 7—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 6—Transmit End (TEND): Indicates that there is no valid data in SCFTDR0 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 condition] When data is written to SCFTDR0	
1	Transmission has been ended [Setting conditions] • Power-on reset or manual reset • When there is no transmit data in SCFTDR0 on transmission of a 1-byte serial transmit character	(Initial value)

Bit 5—Transmit FIFO Data Stop (TDFST): Indicates that data has been transferred from SCFTDR0 to SCTSR0, and the transfer data number is now equal to the number of transfer bytes specified by bits FST6–FST0 in SCLSR0. This bit is valid when FSTE = 1.

Bit 5: TDFST	Description
0	The number of transfer data bytes is less than the transfer byte number specified by bits FST6–FST0 (Initial value) [Clearing conditions] • Power-on reset or manual reset • When 0 is written to TDFST after reading TDFST = 1
1	The number of transfer data bytes is equal to the transfer byte number specified by bits FST6–FST0 [Setting condition] When the number of transfer data bytes becomes equal to the transfer byte number specified by bits FST6–FST0 due to a transmit operation

Bits 4 to 2—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 1—Receive FIFO Data Full/Stop (RDFST): The function of this bit depends on the setting of the FSTE bit in SCLSR0.

- When FSTE = 1

  Indicates that received data has been transferred from SCRSR0 to SCFRDR0, and the number of receive data bytes is now equal to the transfer byte number specified by bits FST6–FST0.

Bit 1: RDFST	Description
0	The number of receive data bytes in SCFRDR0 is less than the transfer byte number specified by bits FST6–FST0 (Initial value) [Clearing conditions] • Power-on reset or manual reset • When 0 is written to RDFST after reading RDFST = 1
1	The number of receive data bytes in SCFRDR0 is equal to the transfer byte number specified by bits FST6–FST0 [Setting condition] When the number of receive data bytes in SCFRDR0 becomes equal to the transfer byte number specified by bits FST6–FST0*

Note: * If data is read when SCFRDR0 is empty, an undefined value will be read. The number of receive data bytes in SCFRDR0 is indicated in SCRFDR0.

- When FSTE = 0

  Indicates that received data has been transferred from SCRSR0 to SCFRDR0, and SCFRDR0 is now full of receive data.

Bit 1: RDFST	Description
0	SCFRDR0 is not full of receive data (Initial value) [Clearing conditions] • Power-on reset or manual reset • When 0 is written to RDFST after reading RDFST = 1
1	SCFRDR0 is full of receive data [Setting condition] When SCFRDR0 becomes full of receive data*

Note: * If data is read when SCFRDR0 is empty, an undefined value will be read. The number of receive data bytes in SCFRDR0 is indicated in SCRFDR0.

Bit 0—Reserved: This bit is always read as 0. The write value should always be 0.

16.2.8 Bit Rate Register (SCBRR0)

SCBRR0 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 SCSMR0.

SCBRR0 can be read or written to by the CPU at all times.

SCBRR0 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.

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

The SCBRR0 setting is found from the following equation.

$$N = \frac{P\phi}{8 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Where		
	B:	Bit rate (bits/s)
	N:	SCBRR0 setting for baud rate generator (1 ≤ N ≤ 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.)

		SCSMR0 Setting	
n	**Clock**	**CKS1**	**CKS0**
0	Pϕ	0	0
1	Pϕ/4	0	1
2	Pϕ/16	1	0
3	Pϕ/64	1	1

16.2.9 FIFO Control Register (SCFCR0)

SCFCR0 performs data count resetting for the transmit and receive FIFO registers.

SCFCR0 can be read or written to by the CPU at all times.

SCFCR0 is initialized to H'00 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

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

Bits 7 to 3—Reserved: These bits are always read as 0. The write value should always be 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—Reserved: This bit is always read as 0. The write value should always be 0.

16.2.10 Receive FIFO Data Count Register (SCRFDR0)

SCRFDR0 is a 16-bit register that indicates the number of data bytes stored in the receive FIFO data register (SCFRDR0).

A value of H'0000 means that there is no receive data, and a value of H'0180 means that SCFRDR0 is full of receive data.

SCRFDR0 can be read by the CPU at all times.

SCRFDR0 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.

Bit:	15	14	13	12	11	10	9	8
	—	—	—	—	—	—	—	R8
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
	R7	R6	R5	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

16.2.11 Transmit FIFO Data Count Register (SCTFDR0)

SCTFDR0 is a 16-bit register that indicates the number of data bytes stored in the transmit FIFO data register (SCFTDR0).

A value of H'0000 means that there is no transmit data, and a value of H'0080 means that SCFTDR0 is full of transmit data.

SCTFDR0 can be read by the CPU at all times.

SCTFDR0 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.

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
	T7	T6	T5	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

16.2.12 Line Status Register (SCLSR0)

SCLSR0 is a 16-bit register that indicates the status of the FIFO stop function and the receive error (overrun error) status.

SCLSR0 can be read and written to by the CPU at all times. However, 1 cannot be written to the ORER bit; to clear this bit to 0, it must first be read as 1.

SCLSR0 is initialized to H'0004 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bit:	15	14	13	12	11	10	9	8
	—	FST6	FST5	FST4	FST3	FST2	FST1	FST0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	—	—	—	FSTE	—	—	—	ORER
Initial value:	0	0	0	0	0	1	0	0
R/W:	R	R	R	R/W	R	R	R	R/(W)*

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

Bit 15—Reserved: This bit is always read as 0. The write value should always be 0.

Bits 14 to 8—FIFO Stop Count (FST6 to FST0): Transmit/receive operations are stopped in accordance with the transfer byte number specified by these bits. Transmission/reception can be restarted by clearing the TDFST and RDFST bits to 0. These bits are valid only when FSTE = 1, and are invalid when FSTE = 0. The value set in these bits should be one less than the number of bytes to be transferred. A value from H'7F (128 bytes) to H'00 (1 byte) can be set. When the FIFO stop function is used in transmission or transmission/reception, only an even byte setting should be used (by setting an odd value in these bits). For details see section 16.6, Usage Notes.

Bits 7 to 5—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 4—FIFO Stop Count Enable (FSTE): Enables or disables the FIFO stop function during transmission/reception.

Bit 4: FSTE	Description	
0	FIFO stop function disabled	(Initial value)
1	FIFO stop function enabled	

Bits 3 and 1—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 2—Reserved: This bit is always read as 1, and the write value should always be 1.

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 [Clearing conditions] • Power-on reset or manual reset • When 0 is written to ORER after reading ORER = 1	(Initial value)
1	An overrun error occurred during reception*2 [Setting condition] When serial reception is performed while RDFST = 1	

Notes: *1 The ORER flag is not affected and retains its previous state when the RE bit in SCSCR0 is cleared to 0.

*2 The receive data prior to the overrun error is retained in SCFRDR0, 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

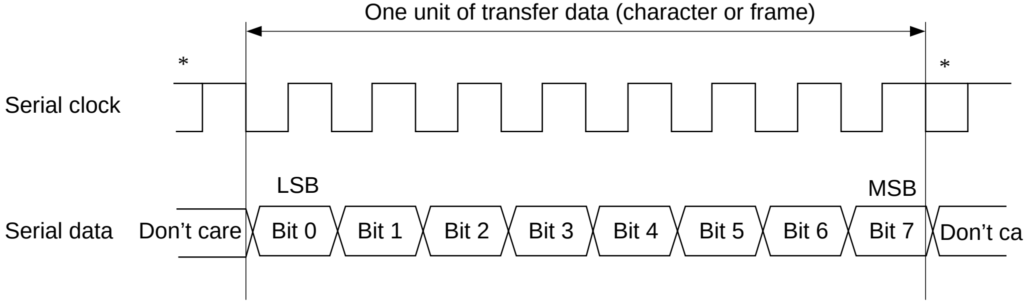
SCIF0 can carry out serial communication in synchronous mode, in which synchronization is achieved with clock pulses.

128-stage receive and 384-stage transmit FIFO buffers are provided, reducing the CPU overhead and enabling fast, continuous communication to be performed.

The operating clock source is selected with the serial mode register (SCSMR0), and the SCIF0 clock source is determined by the CKE1 and CKE0 bits in the serial control register (SCSCR0).

- Transmit/receive format: Fixed 8-bit data
- Indication of amount of data stored in transmit and receive FIFO registers
- Choice of internal or external clock as SCIF0 clock source
  - When internal clock is selected: SCIF0 operates on the baud rate generator clock and outputs a serial clock externally.
  - When external clock is selected: SCIF0 operates on the input serial clock (the on-chip baud rate generator is not used).

16.3.2 Serial Operation



Note: * High except in continuous transmission/reception

Figure 16.2 Data Format in Synchronous Communication

In synchronous serial communication, data on the communication line is output from one fall of the serial clock to the next. Data is guaranteed valid at the rise of the serial clock.

In serial communication, each character is output starting with the LSB and ending with the MSB. After the MSB is output, the communication line remains in the state of the MSB.

In synchronous mode, SCIF0 receives data in synchronization with the rise 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 SCK0 pin can be selected, according to the setting of the CKE1 and CKE0 bits in SCSCR0.

When SCIF0 is operated on an internal clock, the serial clock is output from the SCK0 pin.

Eight serial clock pulses are output in the transfer of one character, and when no transmission/reception is performed the clock is fixed high. In receive-only operation, when the on-chip clock source is selected, clock pulses are output for 1 to 128 bytes (set by bits FST6–FST0) when the FSTE bit in SCLSR0 is 1, and for up to 384 bytes when the FSTE bit is 0.

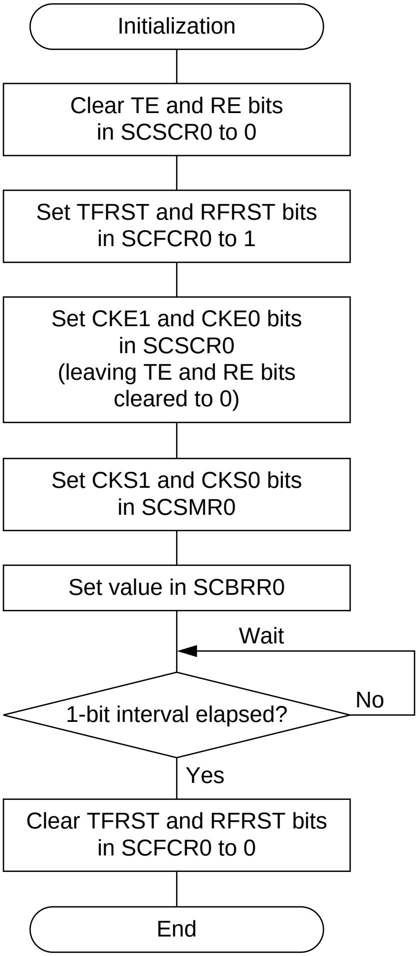
Data Transfer Operations

SCIF0 Initialization: Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR0 to 0, then initialize SCIF0 as described below.

When the clock source, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the transmit shift register (SCTSR0) is initialized. Do not clear the TE bit while transmission is in progress, or the RE bit while reception is in progress. Note that clearing the TE and RE bits to 0 does not change the contents of SCSSR0, SCFTDR0, or SCFRDR0. The TE bit should be cleared to 0 after all transmit data has been sent and the TDFST bit in SCSSR0 has been set to 1. When TE is cleared to 0, the TxD0 pin goes to the high-impedance state. Before setting TE to 1 again to start transmission, the TFRST bit in SCFCR0 should first be set to 1 to reset SCFTDR0.

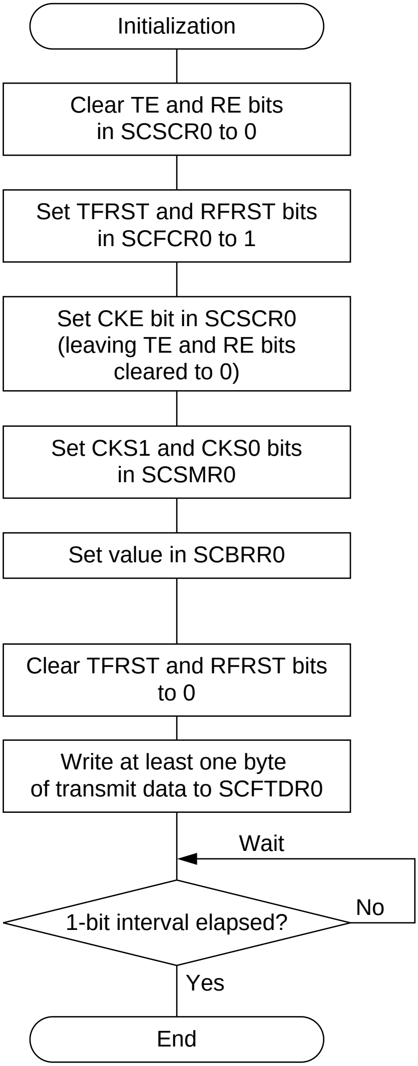
When an external clock is used the clock should not be stopped during operation, including initialization, since operation will be unreliable in this case.

Figure 16.3 shows a sample SCIF0 initialization flowchart.


Initialization
Clear TE and RE bits in SCSCR0 to 0
Set TFRST and RFRST bits in SCFCR0 to 1
Set CKE1 and CKE0 bits in SCSCR0 (leaving TE and RE bits cleared to 0)
Set CKS1 and CKS0 bits in SCSMR0
Set value in SCBRR0
Wait
1-bit interval elapsed?
No
Yes
Clear TFRST and RFRST bits in SCFCR0 to 0
End

1. Set the clock selection in SCSCR0.
   Be sure to clear bits RIE, TIE, TE, and RE to 0.
2. Be sure to set the TFRST and RFRST bits in SCFCR0 to 1, to reset the FIFOs.
3. Set the clock source selection in SCSMR0.
4. Write a value corresponding to the bit rate into SCBRR0 (Not necessary if an external clock is used).
5. Clear the TFRST and RFRST bits in SCFCR0 to 0.

Figure 16.3 Sample SCIF0 Initialization Flowchart (1) (Transmission or Reception)

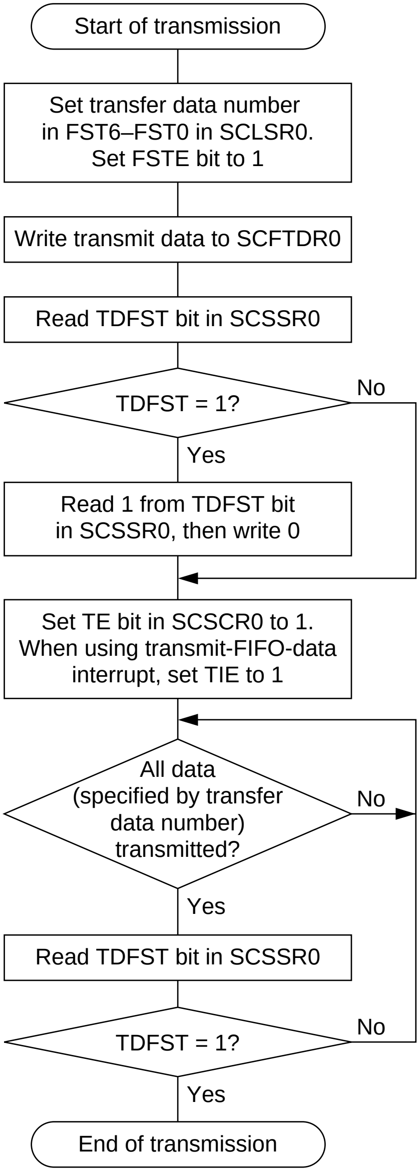

Initialization
Clear TE and RE bits in SCSCR0 to 0
Set TFRST and RFRST bits in SCFCR0 to 1
Set CKE bit in SCSCR0 (leaving TE and RE bits cleared to 0)
Set CKS1 and CKS0 bits in SCSMR0
Set value in SCBRR0
Clear TFRST and RFRST bits to 0
Write at least one byte of transmit data to SCFTDR0
Wait
1-bit interval elapsed?
No
Yes
End

1. Set the clock selection in SCSCR0. Be sure to clear bits RIE, TIE, TE, and RE to 0.
2. Be sure to set the TFRST and RFRST bits in SCFCR0 to 1, to reset the FIFOs.
3. Set the clock source selection in SCSMR0.
4. Write a value corresponding to the bit rate into SCBRR0.
5. Clear the TFRST and RFRST bits in SCFCR0 to 0.
6. Write at least one byte of transmit data to SCFTDR0.
7. Wait for a 1-bit interval.

Figure 16.3 Sample SCIF0 Initialization Flowchart (2) (Simultaneous Transmission and Reception)

Serial Data Transmission: Figure 16.4 shows sample flowcharts for serial transmission.

Example (1) shows the flowchart when transmit data is written for each transmit operation, and example (2) shows the flowchart when transmit data is written for multiple transmit operations. The FIFO stop function is used in both cases.

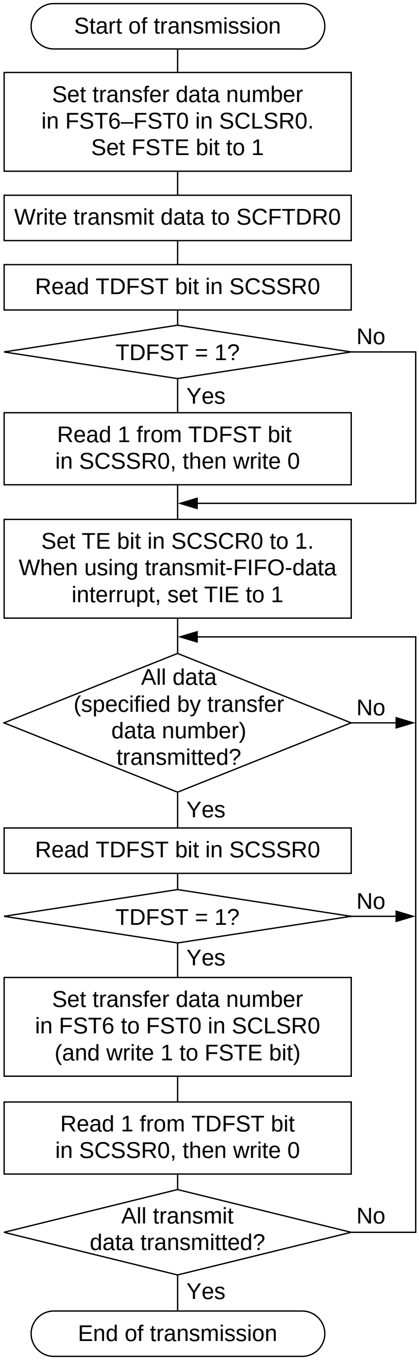

Start of transmission
Set transfer data number in FST6–FST0 in SCLSR0. Set FSTE bit to 1
Write transmit data to SCFTDR0
Read TDFST bit in SCSSR0
TDFST = 1?
No
Yes
Read 1 from TDFST bit in SCSSR0, then write 0
Set TE bit in SCSCR0 to 1. When using transmit-FIFO-data interrupt, set TIE to 1
All data (specified by transfer data number) transmitted?
No
Yes
Read TDFST bit in SCSSR0
TDFST = 1?
No
Yes
End of transmission

1. Set the transfer data number in the line status register (SCLSR0), and set the FSTE bit to 1.
2. Write transmit data to SCFTDR0, then read the TDFST bit, and if 1, clear to 0.
3. Transmission starts when the TE bit in SCSCR0 is set to 1, and when the data specified by the transfer data number has been transmitted, the TDFST bit is set to 1.

When writing more transmit data to the FIFO to continue transmission:

4. Set the TFRST bit in SCFCR0 to 1 to clear the transmit FIFO, then clear TFRST to 0.
5. Write transmit data to SCFTDR0.
6. Set the transfer data number in SCLSR0. Also write 1 to the FSTE bit.
7. Transmission starts when 1 is read from the TDFST bit in SCSSR0 and then TDFST is cleared to 0, and when the data specified by the transfer data number has been transmitted, the TDFST bit is set to 1.

Figure 16.4 Sample SCIF0 Transmission Flowchart (1)


Start of transmission
Set transfer data number in FST6–FST0 in SCLSR0. Set FSTE bit to 1
Write transmit data to SCFTDR0
Read TDFST bit in SCSSR0
TDFST = 1?
No
Yes
Read 1 from TDFST bit in SCSSR0, then write 0
Set TE bit in SCSCR0 to 1. When using transmit-FIFO-data interrupt, set TIE to 1
All data (specified by transfer data number) transmitted?
No
Yes
Read TDFST bit in SCSSR0
TDFST = 1?
No
Yes
Set transfer data number in FST6 to FST0 in SCLSR0 (and write 1 to FSTE bit)
Read 1 from TDFST bit in SCSSR0, then write 0
All transmit data transmitted?
No
Yes
End of transmission

1. Set the transfer data number in the line status register (SCLSR0), and set the FSTE bit to 1.
2. Write transmit data to SCFTDR0, then read the TDFST bit, and if 1, clear to 0.
3. Transmission starts when the TE bit in SCSCR0 is set to 1, and when the data specified by the transfer data number has been transmitted, the TDFST bit is set to 1.
4. Set the transfer data number in SCLSR0 while the TDFST bit in SCSSR0 is 1. Also write 1 to the FSTE bit.
5. Transmission starts when 1 is read from the TDFST bit in SCSSR0 and then TDFST is cleared to 0.

Note: When writing transmit data to SCFTDR0 again after the end of transmission, first set the TFRST bit in SCFCR0 to 1 and clear the transmit FIFO, then clear the TFRST bit to 0.

Figure 16.4 Sample SCIF0 Transmission Flowchart (2)

In serial transmission, SCIF0 operates as described below.

1. When the transfer data number is set in line status register 0 (SCLSR0), the FSTE bit is set, transmit data is written into SCFTDR0, and the TE bit in serial status register 0 (SCSSR0) is set to 1, data is transferred from SCFTDR0 to SCTSR0, and transmission is enabled. The transmit operation starts when the serial clock is output or input in this state.

2. The transmit data is output in LSB (bit 0) to MSB (bit 7) order, in synchronization with the clock. The data length is fixed at 8 bits.

3. When the last data (as specified by the transfer data number set in SCLSR0) is transferred from SCFTDR0 to SCTSR0, SCIF0 sets the TDFST bit, and after the MSB (bit 7) has been sent, the transmit data pin (TxD0) maintains its state.

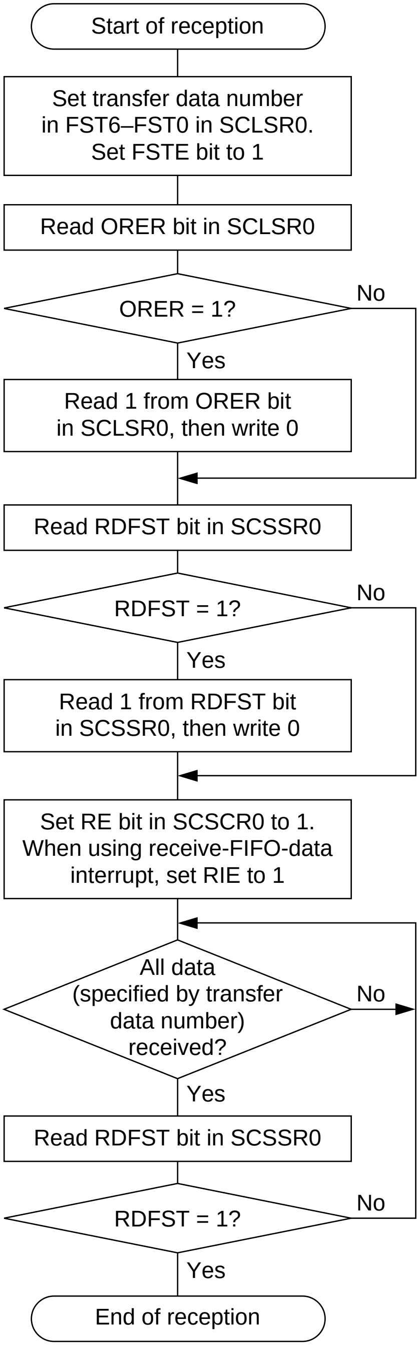
   After completion of serial transmission, the SCK0 pin is fixed high.

   If the TIE bit in the serial control register (SCSCR0) is set to 1, a transmit-FIFO-data interrupt request is generated.

Note: When transmit-FIFO-data interrupt requests are to be generated, the FIFO stop function must be used.

Serial Data Reception: Figure 16.5 shows sample flowcharts for serial reception.

Example (1) shows the flowchart when the FIFO stop function is used and receive data is read for each receive operation, example (2) shows the flowchart when the FIFO stop function is used and receive data is read for multiple receive operations, and example (3) shows the flowchart when the FIFO stop function is not used.

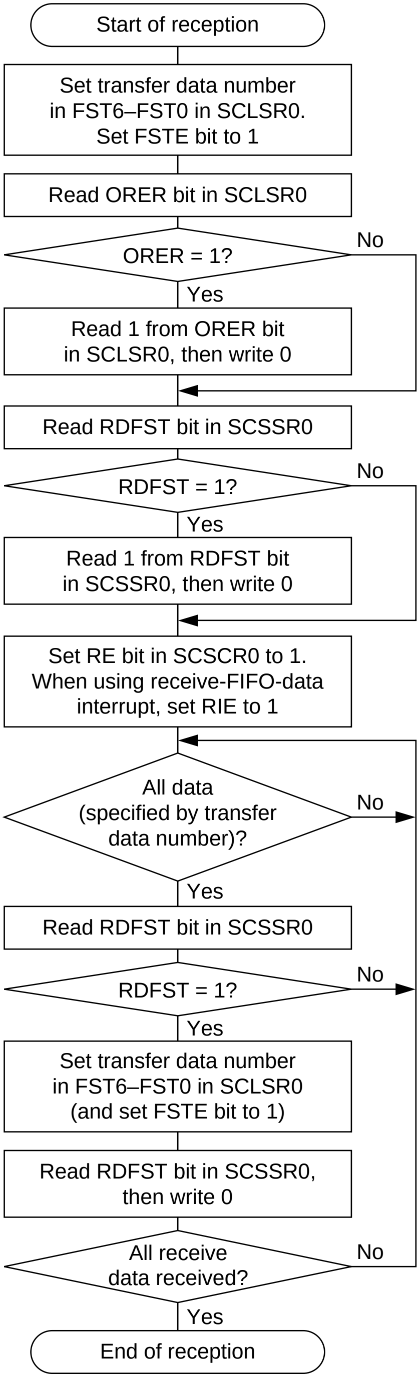

Start of reception
Set transfer data number in FST6–FST0 in SCLSR0. Set FSTE bit to 1
Read ORER bit in SCLSR0
ORER = 1?
No
Yes
Read 1 from ORER bit in SCLSR0, then write 0
Read RDFST bit in SCSSR0
RDFST = 1?
No
Yes
Read 1 from RDFST bit in SCSSR0, then write 0
Set RE bit in SCSCR0 to 1. When using receive-FIFO-data interrupt, set RIE to 1
All data (specified by transfer data number) received?
No
Yes
Read RDFST bit in SCSSR0
RDFST = 1?
No
Yes
End of reception

1. Set the transfer data number in the line status register (SCLSR0), and set the FSTE bit to 1.
   Read the ORER bit in SCLSR0, and if 1, clear to 0.
2. Read the RDFST bit in SCSSR0, and if 1, clear to 0.
3. Reception starts when the RE bit in SCSCR0 is set to 1, and when the data specified by the transfer data number has been received, the RDFST bit is set to 1.

When reading more receive data from the FIFO to continue reception:

4. Read receive data from SCFRDR0.
5. Set the RFRST bit in SCFCR0 to 1 to clear the receive FIFO, then clear RFRST to 0.
6. Set the transfer data number in SCLSR0. Also write 1 to the FSTE bit.
   If the ORER bit is 1, read 1 from it, then clear it to 0. Reception cannot be continued while the ORER bit is set to 1.
7. Reception starts when 1 is read from the RDFST bit in SCSSR0 and then RDFST is cleared to 0, and when the data specified by the transfer data number has been received, the RDFST bit is set to 1.

Figure 16.5 Sample SCIF0 Reception Flowchart (1)


Start of reception
Set transfer data number in FST6–FST0 in SCLSR0. Set FSTE bit to 1
Read ORER bit in SCLSR0
ORER = 1?
No
Yes
Read 1 from ORER bit in SCLSR0, then write 0
Read RDFST bit in SCSSR0
RDFST = 1?
No
Yes
Read 1 from RDFST bit in SCSSR0, then write 0
Set RE bit in SCSCR0 to 1. When using receive-FIFO-data interrupt, set RIE to 1
All data (specified by transfer data number)?
No
Yes
Read RDFST bit in SCSSR0
RDFST = 1?
No
Yes
Set transfer data number in FST6–FST0 in SCLSR0 (and set FSTE bit to 1)
Read RDFST bit in SCSSR0, then write 0
All receive data received?
No
Yes
End of reception

1. Set the transfer data number in the line status register (SCLSR0), and set the FSTE bit to 1. Read the ORER bit in SCLSR0, and if 1, clear to 0.

2. Read the RDFST bit in SCSSR0, and if 1, clear to 0.

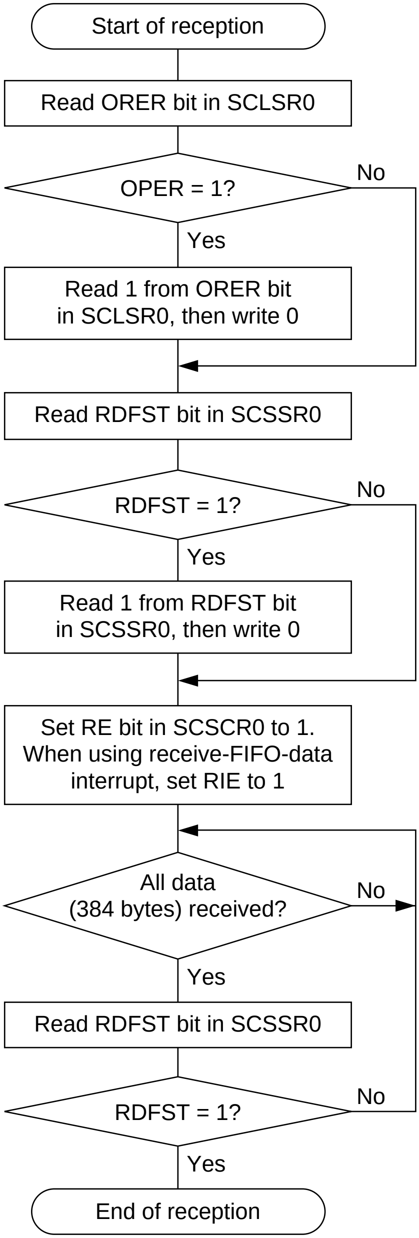
3. Reception starts when the RE bit in SCSCR0 is set to 1, and when the data specified by the transfer data number has been received, the RDFST bit is set to 1.

4. Set the transfer data number in SCLSR0 while the RDFST bit in SCSSR0 is set to 1.
   Also write 1 to the FSTE bit.
   If the ORER bit is 1, read 1 from it, then clear it to 0. Reception cannot be continued while the ORER bit is set to 1.

5. Reception starts when 1 is read from the RDFST bit in SCSSR0 and then RDFST is cleared to 0.

Note: When continuing receive operations after the end of reception, after reading the receive data from SCFRDR0, set the RFRST bit in SCFCR0 to 1 and clear the receive FIFO, then clear the RFRST bit to 0.

Figure 16.5 Sample SCIF0 Reception Flowchart (2)



Start of reception
Read ORER bit in SCLSR0
OPER = 1?
No
Yes
Read 1 from ORER bit in SCLSR0, then write 0
Read RDFST bit in SCSSR0
RDFST = 1?
No
Yes
Read 1 from RDFST bit in SCSSR0, then write 0
Set RE bit in SCSCR0 to 1. When using receive-FIFO-data interrupt, set RIE to 1
All data (384 bytes) received?
No
Yes
Read RDFST bit in SCSSR0
RDFST = 1?
No
Yes
End of reception

1. Read the ORER bit in the line status register (SCLSR0), and if 1, clear to 0.
2. Read the RDFST bit in SCSSR0, and if 1, clear to 0.
3. Reception starts when the RE bit in SCSCR0 is set to 1, and when all the data (384 bytes) has been received, the RDFST bit is set to 1 and reception ends.

Note: When continuing receive operations after the end of reception, after reading the receive data from SCFRDR0, set the RFRST bit in SCFCR0 to 1 and clear the receive FIFO, then clear the RFRST bit to 0.

Figure 16.5 Sample SCIF0 Reception Flowchart (3)

In serial reception, SCIF0 operates as described below.

1. When the transfer data number is set in line status register 0 (SCLSR0), and the RE bit in serial status register 0 (SCSSR0) is set to 1, reception is enabled. The receive operation starts when the serial clock is output or input in this state.

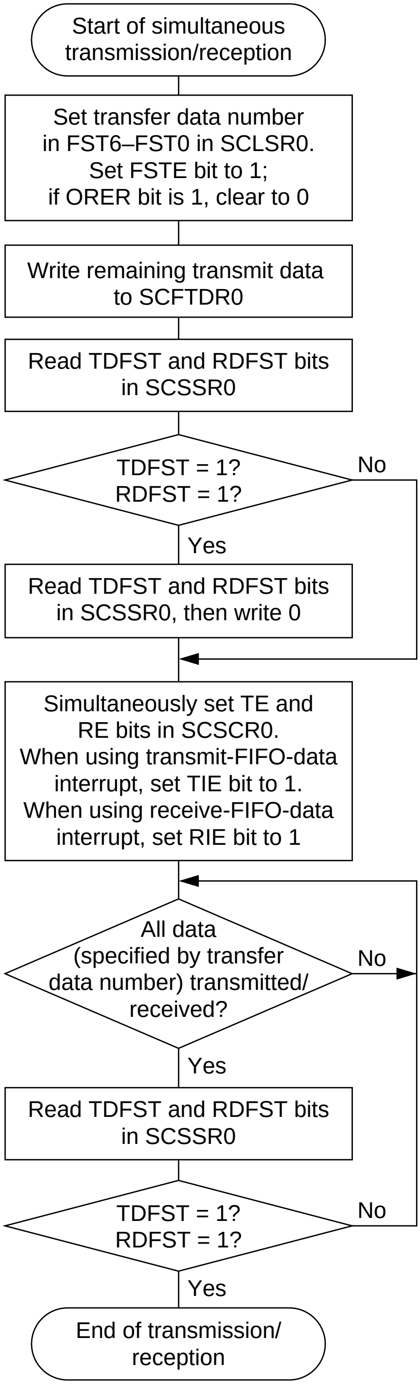
2. The received data is stored in receive shift register 0 (SCRSR0) LSB-to-MSB order.

3. The number of transfer data bytes set in SCLSR0 are received, and reception ends. If the FIFO stop function is used (FSTE = 0), reception ends when 384 bytes have been received.

   When the last receive data is transferred from SCRSR0 to SCFRDR0, the RDFST bit is set to 1, and if the RIE bit in serial control register 0 (SCSCR0) is set, a receive-FIFO-data interrupt request is generated.

Simultaneous Serial Data Transmission and Reception: Figure 16.6 shows sample flowcharts for simultaneous serial transmission and reception.

Example (1) shows the flowchart when data is read and written for each transmit/receive operation, and example (2) shows the flowchart when data is read and written for multiple transmit/receive operations.


Start of simultaneous transmission/reception
Set transfer data number in FST6–FST0 in SCLSR0. Set FSTE bit to 1; if ORER bit is 1, clear to 0
Write remaining transmit data to SCFTDR0
Read TDFST and RDFST bits in SCSSR0
TDFST = 1? RDFST = 1?
No
Yes
Read TDFST and RDFST bits in SCSSR0, then write 0
Simultaneously set TE and RE bits in SCSCR0. When using transmit-FIFO-data interrupt, set TIE bit to 1. When using receive-FIFO-data interrupt, set RIE bit to 1
All data (specified by transfer data number) transmitted/ received?
No
Yes
Read TDFST and RDFST bits in SCSSR0
TDFST = 1? RDFST = 1?
No
Yes
End of transmission/ reception

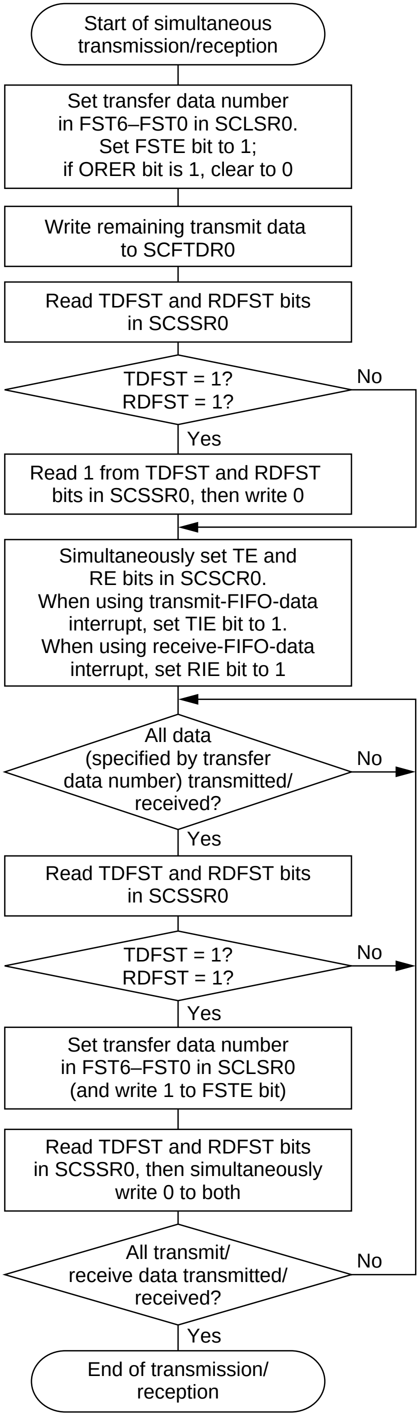
1. Set the transfer data number in the line status register (SCLSR0), and set the FSTE bit to 1.
   Read the ORER bit in SCLSR0, and if 1, clear to 0.
2. Write the remaining transmit data to SCFTDR0, then read the TDFST and RDFST bits in SCSSR0, and if 1, clear to 0.
3. Transmission/reception starts when the TE and RE bits in SCSCR0 are set to 1. The TE and RE bits must be set simultaneously.

   When the data specified by the transfer data number has been transmitted/received, the TDFST and RDFST bits are set to 1.

When writing more transmit data and reading more receive data to continue transmission/reception:

4. Read receive data from SCFRDR0.
5. Set the TFRST and RFRST bits in SCFCR0 to 1 to clear the transmit and receive FIFOs, then clear TFRST and RFRST to 0.
6. Write transmit data to SCFTDR0.
7. Set the transfer data number in SCLSR0. Also write 1 to the FSTE bit.
   If the ORER bit is 1, read 1 from it, then clear it to 0.
8. Transmission/reception starts when 1 is read from the TDFST and RDFST bits in SCSSR0 and then these bits are cleared to 0.
   The TDFST and RDFST bits must be cleared to 0 simultaneously.
   When the data specified by the transfer data number has been transmitted/ received, the TDFST and RDFST bits are set to 1.

Figure 16.6 Sample SCIF0 Simultaneous Transmission/Reception Flowchart (1)


Start of simultaneous transmission/reception
Set transfer data number in FST6–FST0 in SCLSR0. Set FSTE bit to 1; if ORER bit is 1, clear to 0
Write remaining transmit data to SCFTDR0
Read TDFST and RDFST bits in SCSSR0
TDFST = 1? RDFST = 1?
No
Yes
Read 1 from TDFST and RDFST bits in SCSSR0, then write 0
Simultaneously set TE and RE bits in SCSCR0. When using transmit-FIFO-data interrupt, set TIE bit to 1. When using receive-FIFO-data interrupt, set RIE bit to 1
All data (specified by transfer data number) transmitted/ received?
No
Yes
Read TDFST and RDFST bits in SCSSR0
TDFST = 1? RDFST = 1?
No
Yes
Set transfer data number in FST6–FST0 in SCLSR0 (and write 1 to FSTE bit)
Read TDFST and RDFST bits in SCSSR0, then simultaneously write 0 to both
All transmit/ receive data transmitted/ received?
No
Yes
End of transmission/ reception

1. Set the transfer data number in the line status register (SCLSR0), and set the FSTE bit to 1. Read the ORER bit in SCLSR0, and if 1, clear to 0.

2. Write the remaining transmit data to SCFTDR0, then read the TDFST and RDFST bits in SCSSR0, and if 1, clear to 0.

3. Transmission/reception starts when the TE and RE bits in SCSCR0 are set to 1. The TE and RE bits must be set simultaneously.
   When the data specified by the transfer data number has been transmitted/received, the TDFST and RDFST bits are set to 1.

4. Set the transfer data number in SCLSR0 while the TDFST and RDFST bits in SCSSR0 are set to 1.
   Also write 1 to the FSTE bit.
   If the ORER bit is 1, read 1 from it, then clear it to 0.

5. Transmission/reception starts when 1 is read from the TDFST and RDFST bits in SCSSR0 and then these bits are cleared to 0.
   The TDFST and RDFST bits must be cleared to 0 simultaneously.

Note: When continuing transmit/receive operations after the end of transmission/reception, before writing the transmit data and after reading the receive data, set the TFRST bit and RFRST bit in SCFCR0 to 1 and clear the transmit/ receive FIFOs, then clear the TFRST bit and RFRST bit to 0.

Figure 16.6 Sample SCIF0 Simultaneous Transmission/Reception Flowchart (2)

16.4 SCIF0 Interrupt Sources and the DMAC

SCIF0 has two interrupt sources: the transmit-FIFO-data interrupt (TXI) request and receive-FIFO-data interrupt (RXI) request.

Table 16.3 shows the interrupt sources and their order of priority. The interrupt sources can be enabled or disabled by means of the TIE and RIE bits in SCSCR0. A separate interrupt request is sent to the interrupt controller for each of these interrupt sources.

When the TDFST flag in SCSSR0 is set to 1, a TXI interrupt request is generated. The DMAC can be activated and data transfer performed on generation of a TXI interrupt request.

When the RDFST flag in SCSSR0 is set to 1, an RXI interrupt request is generated. The DMAC can be activated and data transfer performed on generation of an RXI interrupt request.

When using the DMAC for transmission/reception, set and enable the DMAC before making SCIF0 settings. See section 14, Direct Memory Access Controller (DMAC), for details of the DMAC setting procedure.

Table 16.3 SCIF0 Interrupt Sources

Interrupt Source	Description	DMAC Activation	Priority on Reset Release
RXI	Interrupt initiated by receive FIFO data flag (RDFST)	Possible	High ↕
TXI	Interrupt initiated by transmit FIFO data flag (TDFST)	Possible	Low

See section 5, Exception Handling, for priorities and the relationship with non-SCIF0 interrupts.

16.5 Timing of TDFST, RDFST, and TEND Bit Setting

Figure 16.7 shows the timing for the setting of bits TDFST, RDFST, and TEND.

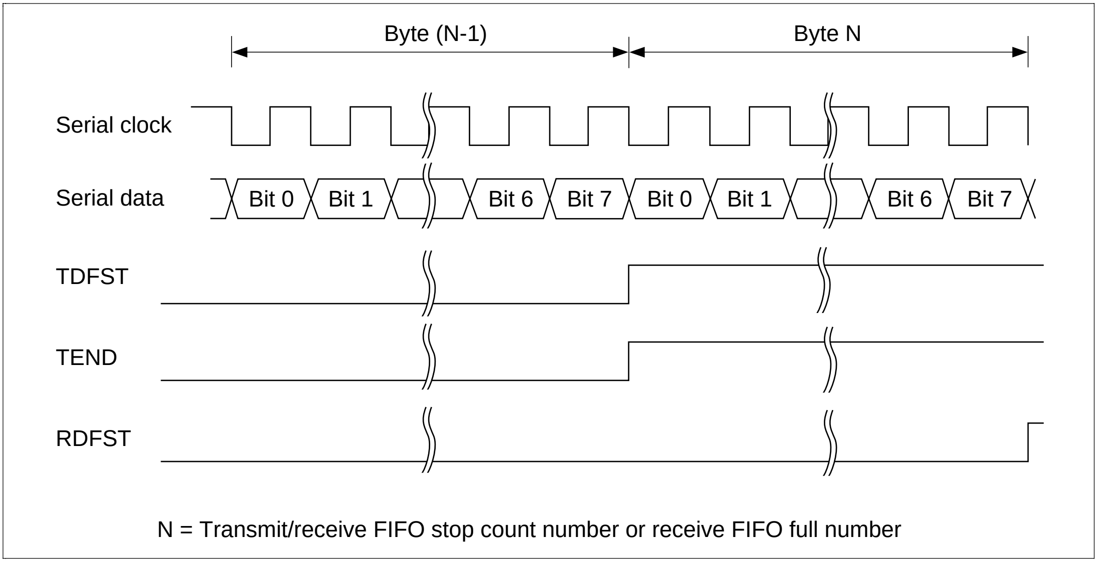


Figure 16.7 Timing of TDFST, RDFST, and TEND Bit Setting

The TDFST and TEND bits are set when the last data is transferred from SCFTDR0 to SCTSR0. The RDFST bit is set when the last data is transferred from SCRSR0 to SCFRDR0.

16.6 Usage Notes

Note the following when using SCIF0.

Interrupt Acceptance during DMAC Burst Transfer: SCIF0 interrupts (transmit-FIFO-data interrupt and receive-FIFO-data interrupt) are not accepted during DMAC burst transfer.

Reading/Writing in Transmit FIFO Full and Receive FIFO Empty States: SCFTDR0 is a write-only register, but write data is ignored after the transmit FIFO becomes full.

SCFRDR0 is a read-only register, but read data is undefined after the receive FIFO becomes empty.

When Using FIFO Stop Function:

1. When the FIFO stop function is used, set a value in the FIFO stop count bits (FST6–FST0) in the SCLSR register, then write H'00 to the FIFO stop count bits.

   Example:

   (Procedure for first transfer when using FIFO stop function)
   When the FSTE bit is set, simultaneously set a value in the FIFO stop count bits (FST6–FST0). Then clear the FIFO stop count bits to H'00.

```
MOV.W  #H'2718,R0
MOV.W  R0,@(SCLSR_OFFSET,GBR)  ; Set 40 bytes in FIFO stop count bits

MOV.W  #H'0018,R0
MOV.W  R0,@(SCLSR_OFFSET,GBR)  ; Clear FIFO stop count bits to 0
```

   (Procedure for second and subsequent transfers when using FIFO stop function)
   After completion of the preceding reception, set a value in the FIFO stop count bits (FST6–FST0) while the RDFST/TDFST bits are set to 1 (set 1 in the FSTE bit at this time). Next, clear the RDFST/TDFST bits and then clear the FIFO stop count bits to H'00.

```
MOV.B  #H'06,R0
MOV.B  R0,@(SCFCR_OFFSET,GBR)  ; Reset transmit/receive FIFO
MOV.B  #H'00,R0
MOV.B  R0,@(SCFCR_OFFSET,GBR)

MOV.W  #H'2718,R0
MOV.W  R0,@(SCLSR_OFFSET,GBR)  ; Set 40 bytes in FIFO stop count bits

MOV.W  @(SCSSR_OFFSET,GBR),R0
MOV    #H'00,R0
MOV.W  R0,@(SCSSR_OFFSET,GBR)  ; Clear RDFST/TDFST flags to 0

MOV.W  #H'0018,R0
MOV.W  R0,@(SCLSR_OFFSET,GBR)  ; Clear FIFO stop count bits to 0
```

   Supplementary Explanation:
   The timing for latching of FST6–FST0 as the transmit/receive FIFO stop counter is as follows.

   When using the transmit FIFO stop function:

   a. When the TDFST bit is 0 and the FSTE bit 0 value is changed to 1
   b. When the TDFST bit is 1

When using the receive FIFO stop function:

a. When the RDFST bit is 0 and the FSTE bit 0 value is changed to 1

b. When the RDFST bit is 1

2. The operation of the FIFO stop function (when an odd number of bytes are set) is as follows.

   The output data hold specification cannot be satisfied by switching to the LSB (first data) at the point at which the output data of the MSB (last data) of the last byte is (1) (see figure 16.8).

   When using the FIFO stop function in transmission or reception, only an even number of bytes should be set.

Table 16.4 Method-of-Use Matrix

	FIFO Stop Function		
	Used		
	Odd Byte Number Setting	Even Byte Number Setting	Not Used
Transmit operation	No	Yes	Yes
Receive operation	Yes	Yes	Yes
Transmit/receive operation	No	Yes	Yes

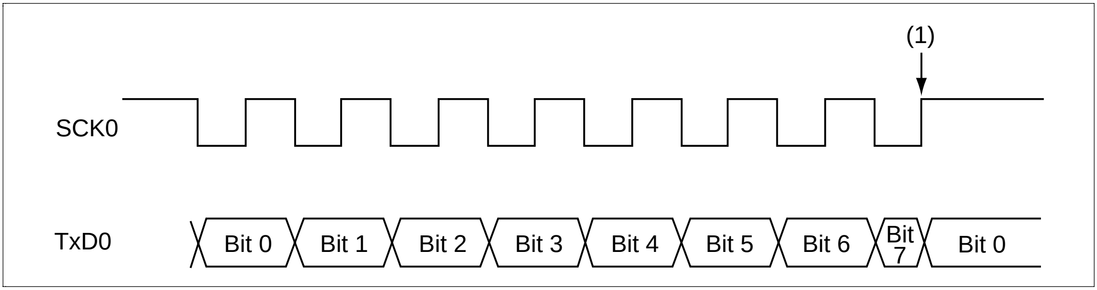


Figure 16.8 Transmission Chart when Using FIFO Stop Function (Odd Number of Bytes Set)

Section 17 Serial Communication Interface with FIFO (SCIF1)

17.1 Overview

SCIF1 is a serial communication interface with built-in FIFO buffers (Serial Communication Interface with FIFO: SCIF). SCIF1 can perform synchronous serial communication.

128-stage FIFO registers are provided for both transmission and reception, enabling fast, efficient, and continuous communication.

17.1.1 Features

SCIF1 features are listed below.

- 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 communication format.

  — Data length: 8 bits

  — LSB-first transfer

  — Receive error detection: Overrun errors
- Full-duplex communication capability

  The transmitter and receiver are independent units, enabling transmission and reception to be performed simultaneously.

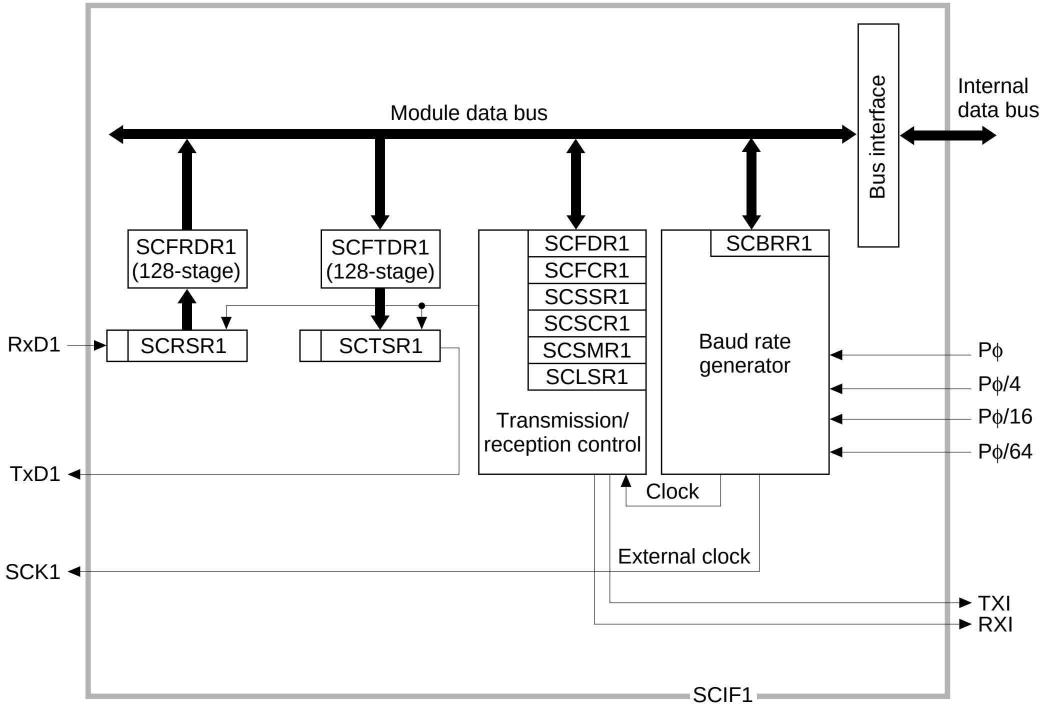
  The transmitter and receiver both have a 128-stage FIFO buffer structure, enabling fast and continuous serial data transmission and reception.
- On-chip baud rate generator allows any bit rate to be selected.
- Choice of serial clock source: internal clock from baud rate generator or external clock from SCK1 pin
- Two interrupt sources

  There are two interrupt sources—transmit-FIFO-data and receive-FIFO-data—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 or receive-FIFO-data interrupt.
- When not in use, SCIF1 can be stopped by halting its clock supply to reduce power consumption.
- The amount of data in the transmit/receive FIFO registers can be ascertained.

- The contents of the transmit FIFO data register (SCFTDR1) and receive FIFO data register (SCFRDR1) are undefined after a power-on or manual reset. Other registers are initialized by a power-on or manual reset, and retain their values in standby mode and in the module standby state. For details see section 17.1.4, Register Configuration.

17.1.2 Block Diagram

Figure 17.1 shows a block diagram of SCIF1.



SCRSR1:	Receive shift register	SCSSR1:	Serial status register
SCFRDR1:	Receive FIFO data register	SCBRR1:	Bit rate register
SCTSR1:	Transmit shift register	SCFCR1:	FIFO control register
SCFTDR1:	Transmit FIFO data register	SCFDR1:	FIFO data count register
SCSMR1:	Serial mode register	SCLSR1:	Line status register
SCSCR1:	Serial control register		

Figure 17.1 Block Diagram of SCIF1

17.1.3 Pin Configuration

Table 17.1 shows the SCIF1 pin configuration.

Table 17.1 SCIF1 Pins

	Pin Name	Abbreviation	I/O	Function
SCIF1	Serial clock pin	SCK1	I/O	Clock input/output
	Receive data pin	RxD1	Input	Receive data input
	Transmit data pin	TxD1	Output	Transmit data output

Note: These pins are made to function as serial pins by performing SCIF1 operation settings with the TE and RE bits in SCSCR1.

17.1.4 Register Configuration

SCIF1 has the internal registers shown in table 17.2. These registers are used to specify the bit rate, and to perform transmitter/receiver control.

Table 17.2 SCIF1 Registers

	Name	Abbreviation	R/W	Initial Value	Address	Access Size
SCIF1	Serial mode register	SCSMR1	R/W	H'00	H'A4002020	8
	Bit rate register	SCBRR1	R/W	H'FF	H'A4002022	8
	Serial control register	SCSCR1	R/W	H'00	H'A4002024	8
	Line status register	SCLSR1	R/(W)*2	H’0000	H'A4002026	16
	Serial status register	SCSSR1	R/(W)*1	H'0040	H'A4002028	16
	FIFO control register	SCFCR1	R/W	H'00	H'A400202A	8
	FIFO data count register	SCFDR1	R	H'0000	H'A400202C	16
	Transmit FIFO data register	SCFTDR1	W	Undefined	H'A4002030	8
	Receive FIFO data register	SCFRDR1	R	Undefined	H'A4002034	8

Notes: *1 Only 0 can be written to bits 5 and 1, to clear the flags.

*2 Only 0 can be written to bit 0, to clear the flag.

17.2 Register Descriptions

17.2.1 Receive Shift Register (SCRSR1)

SCRSR1 is the register used to receive serial data.

SCIF1 sets serial data input from the RxD1 pin in SCRSR1 in the order received, starting with the LSB (bit 0), and converts it to parallel data. When 1 byte of data has been received, it is transferred to the receive FIFO data register, SCFRDR1, automatically.

SCRSR1 cannot be directly read or written to by the CPU.

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

17.2.2 Receive FIFO Data Register (SCFRDR1)

SCFRDR1 is a 128-stage FIFO register that stores received serial data.

When SCIF1 has received one byte of serial data, it transfers the received data from SCRSR1 to SCFRDR1 where it is stored, and completes the receive operation. SCRSR1 is then enabled for reception, and consecutive receive operations can be performed until the receive FIFO data register is full (128 data bytes).

SCFRDR1 is a read-only register, and cannot be written to by the CPU.

If a read is performed when there is no receive data in the receive FIFO data register, an undefined value will be returned. When the receive FIFO data register is full of receive data, subsequent serial data is lost.

The contents of SCFRDR1 are undefined after a power-on reset or manual reset.

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

17.2.3 Transmit Shift Register (SCTSR1)

SCTSR1 is the register used to transmit serial data.

To perform serial data transmission, SCIF1 first transfers transmit data from SCFTDR1 to SCTSR1, then sends the data to the TxD1 pin starting with the LSB (bit 0).

When transmission of one byte is completed, the next transmit data is transferred from SCFTDR1 to SCTSR1, and transmission started, automatically.

SCTSR1 cannot be directly read or written to by the CPU.

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

17.2.4 Transmit FIFO Data Register (SCFTDR1)

SCFTDR1 is a 128-stage FIFO data register that stores data for serial transmission.

If SCTSR1 is empty when transmit data has been written to SCFTDR1, SCIF1 transfers the transmit data written in SCFTDR1 to SCTSR1 and starts serial transmission.

SCFTDR1 is a write-only register, and cannot be read by the CPU.

The next data cannot be written when SCFTDR1 is filled with 128 bytes of transmit data. Data written in this case is ignored.

The contents of SCFTDR1 are undefined after a power-on reset or manual reset.

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

17.2.5 Serial Mode Register (SCSMR1)

SCSMR1 is an 8-bit register used to 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. It is not initialized in standby mode or in the module standby state.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Bits 7–2—Reserved: These bits are always read as 0. The write value should always be 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 17.2.8, 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

17.2.6 Serial Control Register (SCSCR1)

The SCSCR1 register performs enabling or disabling of SCIF1 transfer operations 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. It is not initialized in standby mode or in the module standby state.

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	R	R/W	R/W

Bit 7—Transmit Interrupt Enable (TIE): Enables or disables transmit-FIFO-data interrupt (TXI) request generation when serial transmit data is transferred from SCFTDR1 to SCTSR1, and the TDFST flag in SCSSR1 is set to 1.

Bit 7: TIE	Description	
0	Transmit-FIFO-data interrupt (TXI) request disabled*	(Initial value)
1	Transmit-FIFO-data interrupt (TXI) request enabled	

Note: * TXI interrupt requests can be cleared by reading 1 from the TDFST flag, then clearing it to 0, or by clearing the TIE bit to 0.

When the transmit-FIFO-data interrupt is used, set the FSTE bit in SCLSR1 to 1.

Bit 6—Receive Interrupt Enable (RIE): Enables or disables generation of a receive-data interrupt (RXI) request when the RDFST flag in SCSSR1 is set.

Bit 6: RIE	Description	
0	Receive-FIFO-data interrupt (RXI) request disabled*	(Initial value)
1	Receive-FIFO-data interrupt (RXI) request enabled	

Note: * An RXI interrupt request can be cleared by reading 1 from the RDFST 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 SCIF1.

Bit 5: TE	Description
0	Transmission disabled (Initial value)
1	Transmission enabled*

Note: * Serial mode register (SCSMR1) and FIFO control register (SCFCR1) settings must be made, the operating clock source 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 SCIF1.

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 RDFST flag, which retains its state.

*2 Serial mode register (SCSMR1) and FIFO control register (SCFCR1) settings must be made, the operating clock source decided, and the receive FIFO reset, before the RE bit is set to 1.

Bits 3 and 2—Reserved: These bits are always read as 0. The write value should always be 0.

Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0): These bits select the SCIF1 clock source. The CKE1 and CKE0 bits must be set before determining the SCIF1 operating mode with SCSMR1.

Bit 1: CKE1	Bit 0: CKE0	Description
0	0	Internal clock/SCK1 pin functions as input pin (input signal ignored) (Initial value)
	1	Internal clock/SCK1 pin functions as serial clock output
1	0	External clock/SCK1 pin functions as clock input
	1	External clock/SCK1 pin functions as clock input

17.2.7 Serial Status Register (SCSSR1)

SCSSR1 is a 16-bit register that can be read or written to by the CPU at all times. However, 1 cannot be written to the TDFST and RDFST flags. Also note that in order to clear these flags they must be read as 1 beforehand.

SCSSR1 is initialized to H'0040 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

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
	—	TEND	TDFST	—	—	—	RDFST	—
Initial value:	0	1	0	0	0	0	0	0
R/W:	R	R	R/(W)*	R	R	R	R/(W)*	R

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

Bits 15 to 7—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 6—Transmit End (TEND): Indicates that there is no valid data in SCFTDR1 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 condition] When data is written to SCFTDR1	
1	Transmission has been ended [Setting conditions] • Power-on reset or manual reset • When there is no transmit data in SCFTDR1 on transmission of a 1-byte serial transmit character	(Initial value)

Bit 5—Transmit FIFO Data Stop (TDFST): Indicates that data has been transferred from SCFTDR1 to SCTSR1, and the transfer data number is now equal to the number of transfer bytes specified by bits FST6 to FST0 in SCLSR1. This bit is valid when FSTE = 1.

Bit 5: TDFST	Description
0	The number of transfer data bytes is less than the transfer byte number specified by bits FST6–FST0 (Initial value) [Clearing conditions] • Power-on reset or manual reset • When 0 is written to TDFST after reading TDFST = 1
1	The number of transfer data bytes is equal to the transfer byte number specified by bits FST6–FST0 [Setting condition] When the number of transfer data bytes becomes equal to the transfer byte number specified by bits FST6–FST0 due to a transmit operation

Bits 4 to 2—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 1—Receive FIFO Data Full/Stop (RDFST): The function of this bit depends on the setting of the FSTE bit in SCLSR1.

- When FSTE = 1

  Indicates that received data has been transferred from SCRSR1 to SCFRDR1, and the number of receive data bytes is now equal to the transfer byte number specified by bits FST6 to FST0.

Bit 1: RDFST	Description
0	The number of receive data bytes in SCFRDR1 is less than the transfer byte number specified by bits FST6–FST0 (Initial value) [Clearing conditions] • Power-on reset or manual reset • When 0 is written to RDFST after reading RDFST = 1
1	The number of receive data bytes in SCFRDR1 is equal to the transfer byte number specified by bits FST6–FST0 [Setting condition] When the number of receive data bytes in SCFRDR1 becomes equal to the transfer byte number specified by bits FST6–FST0*

Note: * If data is read when SCFRDR1 is empty, an undefined value will be read. The number of receive data bytes in SCFRDR1 is indicated in the lower bits of SCFDR1.

- When FSTE = 0

  Indicates that received data has been transferred from SCRSR1 to SCFRDR1, and SCFRDR1 is now full of receive data.

Bit 1: RDFST	Description
0	SCFRDR1 is not full of receive data (Initial value) [Clearing conditions] • Power-on reset or manual reset • When 0 is written to RDFST after reading RDFST = 1
1	SCFRDR1 is full of receive data [Setting condition] When SCFRDR1 becomes full of receive data*

Note: * If data is read when SCFRDR1 is empty, an undefined value will be read. The number of receive data bytes in SCFRDR1 is indicated in the lower bits of SCFDR1.

For a 128-byte receive operation, either clear FSTE to 0, or set FSTE to 1 and FST6–FST0 to H'7F.

Bit 0—Reserved: This bit is always read as 0. The write value should always be 0.

17.2.8 Bit Rate Register (SCBRR1)

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. It is not initialized in standby mode or in the module standby state.

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

The SCBRR1 setting is found from the following equation.

$$N = \frac{P\phi}{8 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Where		
	B:	Bit rate (bits/s)
	N:	SCBRR1 setting for baud rate generator (1 ≤ N ≤ 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

17.2.9 FIFO Control Register (SCFCR1)

SCFCR1 performs data count resetting for the transmit and receive FIFO registers.

SCFCR1 can be read or written to by the CPU at all times.

SCFCR1 is initialized to H'00 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

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

Bits 7 to 3—Reserved: These bits are always read as 0. The write value should always be 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—Reserved: This bit is always read as 0. The write value should always be 0.

17.2.10 FIFO Data Count Register (SCFDR1)

SCFDR1 is a 16-bit register that indicates the number of data bytes stored in the transmit FIFO data register (SCFTDR1) and receive FIFO data register (SCFRDR1).

The upper 8 bits show the number of transmit data bytes in SCFTDR1, and the lower 8 bits show the number of receive data bytes in SCFRDR1.

SCFDR1 can be read by the CPU at all times.

SCFDR1 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.

The upper 8 bits of SCFDR1 show the number of untransmitted data bytes in SCFTDR1.

A value of H'00 means that there is no transmit data, and a value of H'80 means that SCFTDR1 is full of transmit data.

The lower 8 bits of SCFDR1 show the number of receive data bytes in SCFRDR1.

A value of H'00 means that there is no receive data, and a value of H'80 means that SCFRDR1 is full of receive data.

Bit:	15	14	13	12	11	10	9	8
	T7	T6	T5	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

Bit:	7	6	5	4	3	2	1	0
	R7	R6	R5	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

17.2.11 Line Status Register (SCLSR1)

SCLSR1 is a 16-bit register that indicates the status of the FIFO stop function and the receive error (overrun error) status.

SCLSR1 can be read and written to by the CPU at all times. However, 1 cannot be written to the ORER bit; to clear this bit to 0, it must first be read as 1.

SCLSR1 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.

Bit:	15	14	13	12	11	10	9	8
	—	FST6	FST5	FST4	FST3	FST2	FST1	FST0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	—	—	—	FSTE	—	—	—	ORER
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R/(W)*

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

Bit 15—Reserved: This bit is always read as 0. The write value should always be 0.

Bits 14 to 8—FIFO Stop Count (FST6 to FST0): Transmit/receive operations are stopped in accordance with the transfer byte number specified by these bits. Transmission/reception can be restarted by clearing the TDFST and RDFST bits to 0. These bits are valid only when FSTE = 1 The value set in these bits should be one less than the number of bytes to be transferred. A value from H'7F (128 bytes) to H'00 (1 byte) can be set.

Bits 7 to 5—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 4—FIFO Stop Count Enable (FSTE): Enables or disables the FIFO stop function during transmission/reception.

Bit 4: FSTE	Description	
0	FIFO stop function disabled	(Initial value)
1	FIFO stop function enabled	

Bits 3 to 1—Reserved: These bits are always read as 0. The write value should always be 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 serial reception is performed while RDFST = 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 SCFRDR1, and the data received subsequently is lost. Serial reception cannot be continued while the ORER flag is set to 1.

17.3 Operation

17.3.1 Overview

SCIF1 can carry out serial communication in synchronous mode, in which synchronization is achieved with clock pulses.

128-stage transmit and receive FIFO buffers are provided, reducing the CPU overhead and enabling fast, continuous communication to be performed.

The operating clock source is selected with the serial mode register (SCSMR1), and the SCIF1 clock source is determined by the CKE1 and CKE0 bits in the serial control register (SCSCR1).

- Transmit/receive format: Fixed 8-bit data
- Indication of amount of data stored in transmit and receive FIFO registers
- Choice of internal or external clock as SCIF1 clock source
  - When internal clock is selected: SCIF1 operates on the baud rate generator clock and outputs a serial clock externally.
  - When external clock is selected: SCIF1 operates on the input serial clock (the on-chip baud rate generator is not used).

17.3.2 Serial Operation

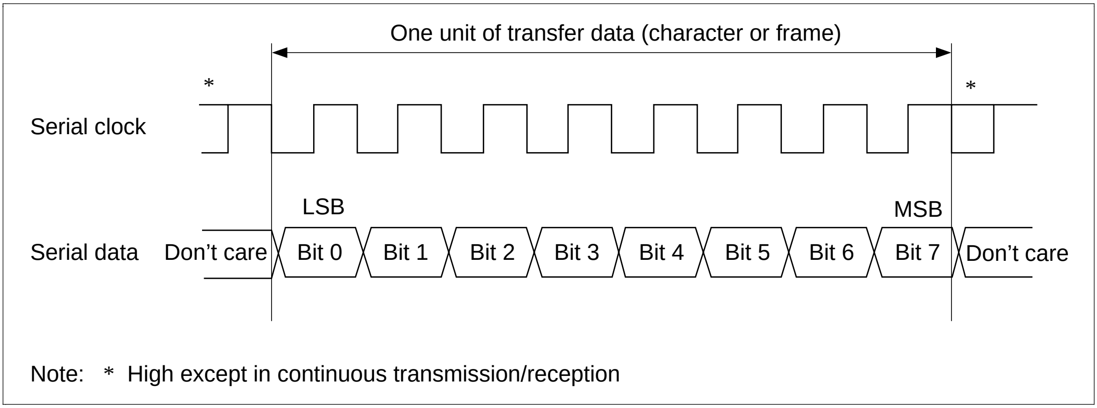

One unit of transfer data (character or frame)
*
*
Serial clock
LSB
MSB
Serial data
Don’t care
Bit 0
Bit 1
Bit 2
Bit 3
Bit 4
Bit 5
Bit 6
Bit 7
Don’t care
Note: * High except in continuous transmission/reception

Figure 17.2 Data Format in Synchronous Communication

In synchronous serial communication, data on the communication line is output from one fall of the serial clock to the next. Data is guaranteed valid at the rise of the serial clock.

In serial communication, each character is output starting with the LSB and ending with the MSB. After the MSB is output, the communication line remains in the state of the MSB.

In synchronous mode, SCIF1 receives data in synchronization with the rise 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 SCK1 pin can be selected, according to the setting of the CKE1 and CKE0 bits in SCSCR1.

When SCIF1 is operated on an internal clock, the serial clock is output from the SCK1 pin.

Eight serial clock pulses are output in the transfer of one character, and when no transmission/reception is performed the clock is fixed high. In receive-only operation, when the on-chip clock source is selected, clock pulses are output for 1 to 128 bytes (set by bits FST6 to FST0) when the FSTE bit in SCLSR1 is 1.

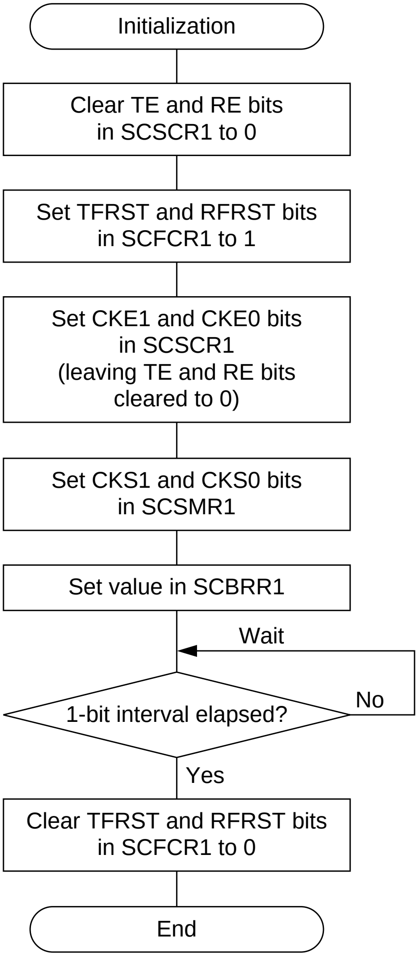
Data Transfer Operations

SCIF1 Initialization: Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR1 to 0, then initialize SCIF1 as described below.

When the clock source, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the transmit shift register (SCTSR1) is initialized. Do not clear the TE bit while transmission is in progress, or the RE bit while reception is in progress. Note that clearing the TE and RE bits to 0 does not change the contents of SCSSR1, SCFTDR1, or SCFRDR1. The TE bit should be cleared to 0 after all transmit data has been sent and the TDFST bit in SCSSR1 has been set to 1. When TE is cleared to 0, the TxD1 pin goes to the high-impedance state. Before setting TE to 1 again to start transmission, the TFRST bit in SCFCR1 should first be set to 1 to reset SCFTDR1.

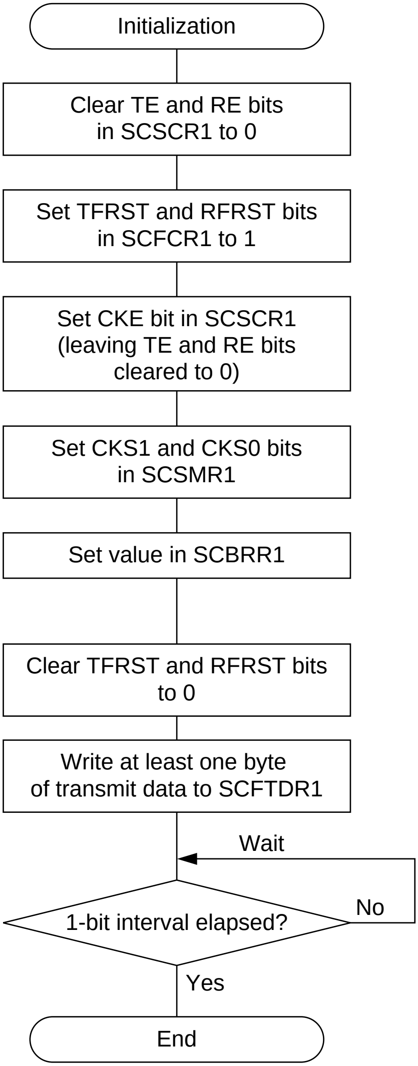
When an external clock is used the clock should not be stopped during operation, including initialization, since operation will be unreliable in this case.

Figure 17.3 shows a sample SCIF1 initialization flowchart.


Initialization
Clear TE and RE bits in SCSCR1 to 0
Set TFRST and RFRST bits in SCFCR1 to 1
Set CKE1 and CKE0 bits in SCSCR1 (leaving TE and RE bits cleared to 0)
Set CKS1 and CKS0 bits in SCSMR1
Set value in SCBRR1
Wait
1-bit interval elapsed?
No
Yes
Clear TFRST and RFRST bits in SCFCR1 to 0
End

1. Set the clock selection in SCSCR1. Be sure to clear bits RIE, TIE, TE, and RE to 0.
2. Be sure to set the TFRST and RFRST bits in SCFCR1 to 1, to reset the FIFOs.
3. Set the clock source selection in SCSMR1.
4. Write a value corresponding to the bit rate into SCBRR1 (Not necessary if an external clock is used).
5. Clear the TFRST and RFRST bits in SCFCR1 to 0.

Figure 17.3 Sample SCIF1 Initialization Flowchart (1) (Transmission or Reception)

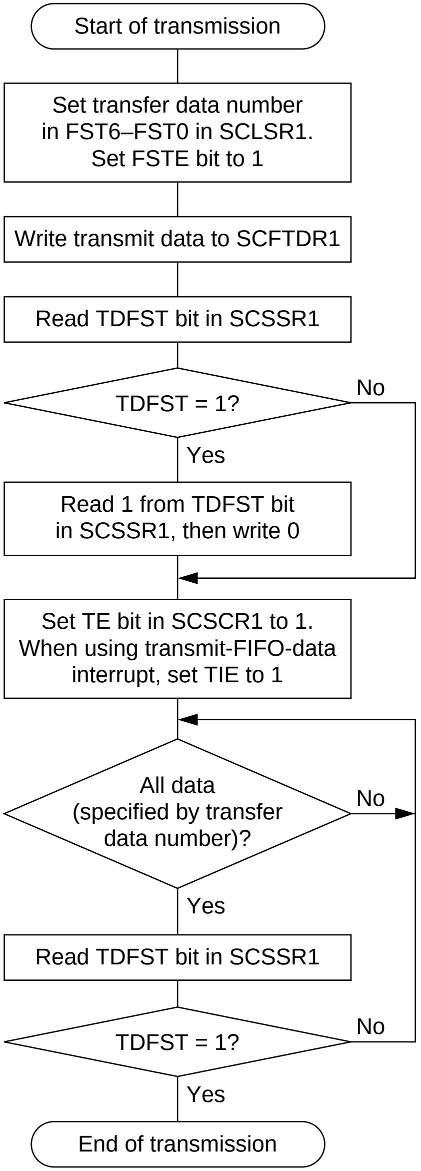

Initialization
Clear TE and RE bits in SCSCR1 to 0
Set TFRST and RFRST bits in SCFCR1 to 1
Set CKE bit in SCSCR1 (leaving TE and RE bits cleared to 0)
Set CKS1 and CKS0 bits in SCSMR1
Set value in SCBRR1
Clear TFRST and RFRST bits to 0
Write at least one byte of transmit data to SCFTDR1
Wait
1-bit interval elapsed?
No
Yes
End

1. Set the clock selection in SCSCR1. Be sure to clear bits RIE, TIE, TE, and RE to 0.
2. Be sure to set the TFRST and RFRST bits in SCFCR1 to 1, to reset the FIFOs.
3. Set the clock source selection in SCSMR1.
4. Write a value corresponding to the bit rate into SCBRR1.
5. Clear the TFRST and RFRST bits in SCFCR1 to 0.
6. Write at least one byte of transmit data to SCFTDR1.
7. Wait for a 1-bit interval.

Figure 17.3 Sample SCIF1 Initialization Flowchart (2) (Simultaneous Transmission and Reception)

Serial Data Transmission: Figure 17.4 shows sample flowcharts for serial transmission.

Example (1) shows the flowchart when transmit data is written for each transmit operation, and example (2) shows the flowchart when transmit data is written for multiple transmit operations. The FIFO stop function is used in both cases.



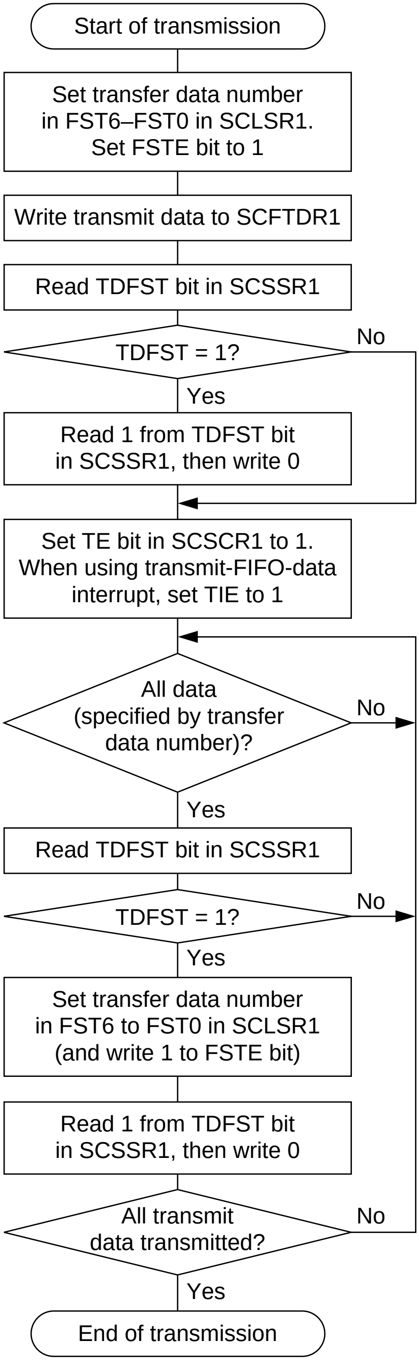
Start of transmission
Set transfer data number in FST6–FST0 in SCLSR1. Set FSTE bit to 1
Write transmit data to SCFTDR1
Read TDFST bit in SCSSR1
TDFST = 1?
No
Yes
Read 1 from TDFST bit in SCSSR1, then write 0
Set TE bit in SCSCR1 to 1. When using transmit-FIFO-data interrupt, set TIE to 1
All data (specified by transfer data number)?
No
Yes
Read TDFST bit in SCSSR1
TDFST = 1?
No
Yes
End of transmission

1. Set the transfer data number in the line status register (SCLSR1), and set the FSTE bit to 1.
2. Write transmit data to SCFTDR1, then read the TDFST bit, and if 1, clear to 0.
3. Transmission starts when the TE bit in SCSCR1 is set to 1, and when the data specified by the transfer data number has been transmitted, the TDFST bit is set to 1.

When writing more transmit data to the FIFO to continue transmission:

4. Set the TFRST bit in SCFCR1 to 1 to clear the transmit FIFO, then clear TFRST to 0.
5. Write transmit data to SCFTDR1.
6. Set the transfer data number in SCLSR1. Also write 1 to the FSTE bit.
7. Transmission starts when 1 is read from the TDFST bit in SCSSR1 and then TDFST is cleared to 0, and when the data specified by the transfer data number has been transmitted, the TDFST bit is set to 1.

Figure 17.4 Sample SCIF1 Transmission Flowchart (1)



Start of transmission
Set transfer data number in FST6–FST0 in SCLSR1. Set FSTE bit to 1
Write transmit data to SCFTDR1
Read TDFST bit in SCSSR1
TDFST = 1?
No
Yes
Read 1 from TDFST bit in SCSSR1, then write 0
Set TE bit in SCSCR1 to 1. When using transmit-FIFO-data interrupt, set TIE to 1
All data (specified by transfer data number)?
No
Yes
Read TDFST bit in SCSSR1
TDFST = 1?
No
Yes
Set transfer data number in FST6 to FST0 in SCLSR1 (and write 1 to FSTE bit)
Read 1 from TDFST bit in SCSSR1, then write 0
All transmit data transmitted?
No
Yes
End of transmission

1. Set the transfer data number in the line status register (SCLSR1), and set the FSTE bit to 1.
2. Write transmit data to SCFTDR1, then read the TDFST bit, and if 1, clear to 0.
3. Transmission starts when the TE bit in SCSCR1 is set to 1, and when the data specified by the transfer data number has been transmitted, the TDFST bit is set to 1.
4. Set the transfer data number in SCLSR1 while the TDFST bit in SCSSR1 is 1. Also write 1 to the FSTE bit.
5. Transmission starts when 1 is read from the TDFST bit in SCSSR1 and then TDFST is cleared to 0.

Note: When writing transmit data to SCFTDR0 again after the end of transmission, first set the TFRST bit in SCFCR0 to 1 and clear the transmit FIFO, then clear the TFRST bit to 0.

Figure 17.4 Sample SCIF1 Transmission Flowchart (2)

In serial transmission, SCIF1 operates as described below.

1. When the transfer data number is set in line status register 1 (SCLSR1), the FSTE bit is set, transmit data is written into SCFTDR1, and the TE bit in serial status register 1 (SCSSR1) is set to 1, data is transferred from SCFTDR1 to SCTSR1, and transmission is enabled. The transmit operation starts when the serial clock is output or input in this state.

2. The transmit data is output in LSB (bit 0) to MSB (bit 7) order, in synchronization with the clock. The data length is fixed at 8 bits.

3. When the last data (as specified by the transfer data number set in SCLSR1) is transferred from SCFTDR1 to SCTSR1, SCIF1 sets the TDFST bit, and after the MSB (bit 7) has been sent, the transmit data pin (TxD1) maintains its state.

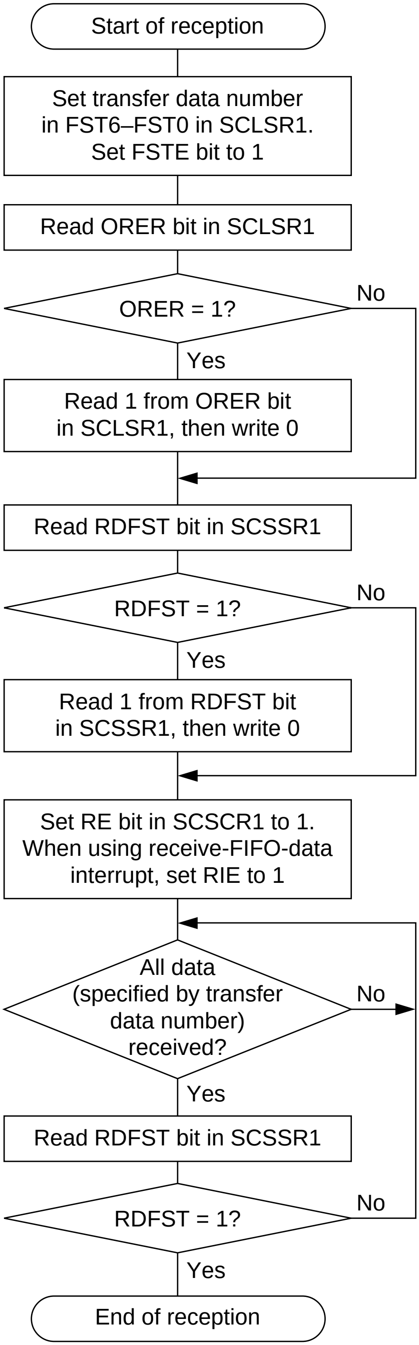
   After completion of serial transmission, the SCK1 pin is fixed high.

   If the TIE bit in the serial control register (SCSCR1) is set to 1, a transmit-FIFO-data interrupt request is generated.

Note: When transmit-FIFO-data interrupt requests are to be generated, the FIFO stop function must be used.

Serial Data Reception: Figure 17.5 shows sample flowcharts for serial reception.

Example (1) shows the flowchart when the FIFO stop function is used and receive data is read for each receive operation, and example (2) shows the flowchart when the FIFO stop function is used and receive data is read for multiple receive operations.



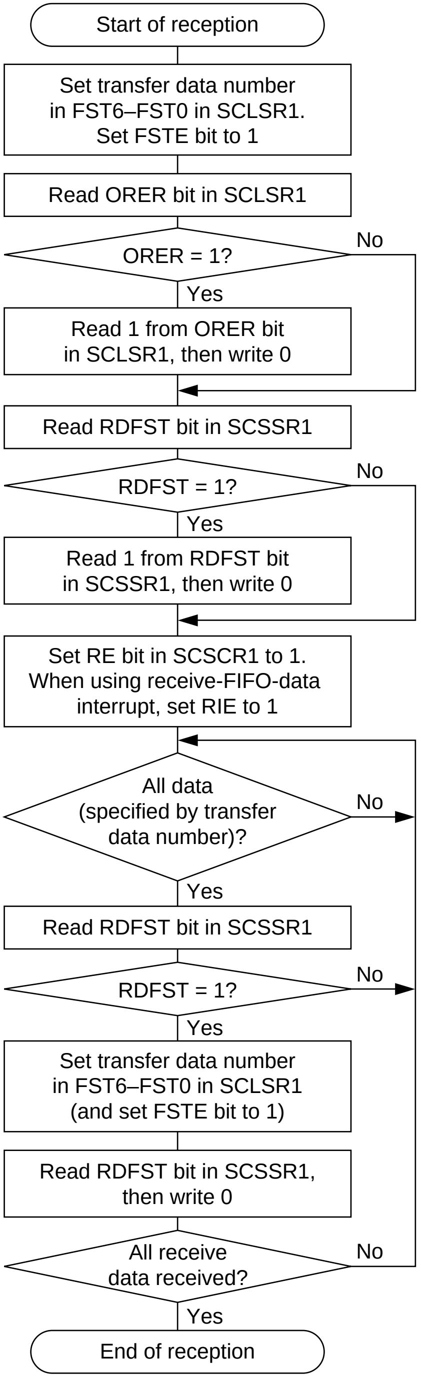
Start of reception
Set transfer data number in FST6–FST0 in SCLSR1. Set FSTE bit to 1
Read ORER bit in SCLSR1
ORER = 1?
No
Yes
Read 1 from ORER bit in SCLSR1, then write 0
Read RDFST bit in SCSSR1
RDFST = 1?
No
Yes
Read 1 from RDFST bit in SCSSR1, then write 0
Set RE bit in SCSCR1 to 1. When using receive-FIFO-data interrupt, set RIE to 1
All data (specified by transfer data number) received?
No
Yes
Read RDFST bit in SCSSR1
RDFST = 1?
No
Yes
End of reception

1. Set the transfer data number in the line status register (SCLSR1), and set the FSTE bit to 1.
   Read the ORER bit in SCLSR1, and if 1, clear to 0.
2. Read the RDFST bit in SCSSR1, and if 1, clear to 0.
3. Reception starts when the RE bit in SCSCR1 is set to 1, and when the data specified by the transfer data number has been received, the RDFST bit is set to 1.

When reading more receive data from the FIFO to continue reception:

4. Read receive data from SCFRDR1.
5. Set the RFRST bit in SCFCR1 to 1 to clear the receive FIFO, then clear RFRST to 0.
6. Set the transfer data number in SCLSR1. Also write 1 to the FSTE bit.
   If the ORER bit is 1, read 1 from it, then clear it to 0. Reception cannot be continued while the ORER bit is set to 1.
7. Reception starts when 1 is read from the RDFST bit in SCSSR1 and then RDFST is cleared to 0, and when the data specified by the transfer data number has been received, the RDFST bit is set to 1.

Figure 17.5 Sample SCIF1 Reception Flowchart (1)


Start of reception
Set transfer data number in FST6–FST0 in SCLSR1. Set FSTE bit to 1
Read ORER bit in SCLSR1
ORER = 1?
No
Yes
Read 1 from ORER bit in SCLSR1, then write 0
Read RDFST bit in SCSSR1
RDFST = 1?
No
Yes
Read 1 from RDFST bit in SCSSR1, then write 0
Set RE bit in SCSCR1 to 1. When using receive-FIFO-data interrupt, set RIE to 1
All data (specified by transfer data number)?
No
Yes
Read RDFST bit in SCSSR1
RDFST = 1?
No
Yes
Set transfer data number in FST6–FST0 in SCLSR1 (and set FSTE bit to 1)
Read RDFST bit in SCSSR1, then write 0
All receive data received?
No
Yes
End of reception

1. Set the transfer data number in the line status register (SCLSR1), and set the FSTE bit to 1. Read the ORER bit in SCLSR1, and if 1, clear to 0.
2. Read the RDFST bit in SCSSR1, and if 1, clear to 0.
3. Reception starts when the RE bit in SCSCR1 is set to 1, and when the data specified by the transfer data number has been received, the RDFST bit is set to 1.
4. Set the transfer data number in SCLSR1 while the RDFST bit in SCSSR1 is set to 1.
   Also write 1 to the FSTE bit.
   If the ORER bit is 1, read 1 from it, then clear it to 0. Reception cannot be continued while the ORER bit is set to 1.
5. Reception starts when 1 is read from the RDFST bit in SCSSR1 and then RDFST is cleared to 0.

Note: When continuing receive operations after the end of reception, after reading the receive data from SCFRDR0, set the RFRST bit in SCFCR0 to 1 and clear the receive FIFO, then clear the RFRST bit to 0.

Figure 17.5 Sample SCIF1 Reception Flowchart (2)

In serial reception, SCIF1 operates as described below.

1. When the transfer data number is set in line status register 1 (SCLSR1), and the RE bit in serial status register 1 (SCSSR1) is set to 1, reception is enabled. The receive operation starts when the serial clock is output or input in this state.

2. The received data is stored in receive shift register 1 (SCRSR1) LSB-to-MSB order.

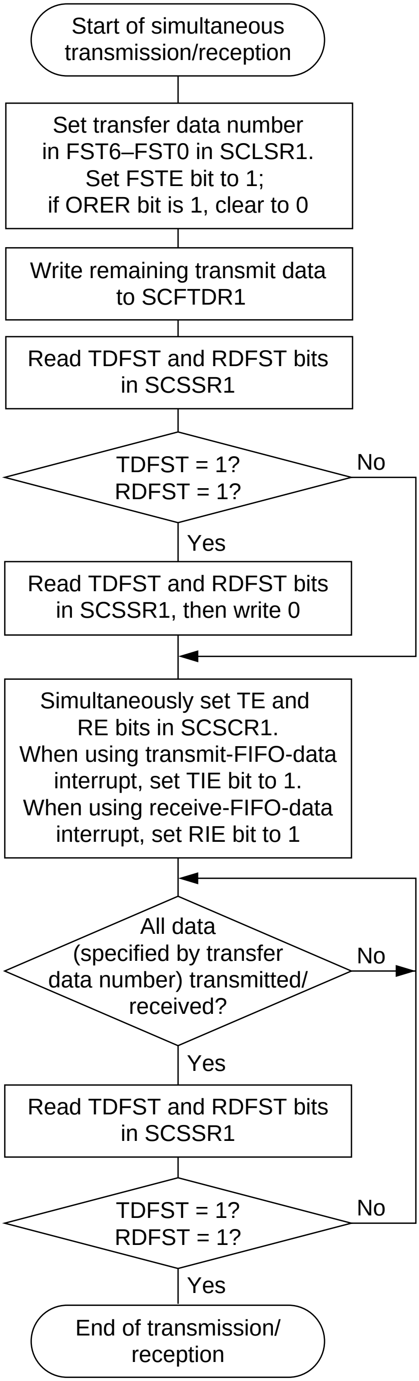
3. The number of transfer data bytes set in SCLSR1 are received, and reception ends.

   When the last receive data is transferred from SCRSR1 to SCFRDR1, the RDFST bit is set to 1, and if the RIE bit in serial control register 1 (SCSCR1) is set, a receive-FIFO-data interrupt request is generated.

Note: When receive-FIFO-data interrupt requests are to be generated, the FIFO stop function must be used.

Simultaneous Serial Data Transmission and Reception: Figure 17.6 shows sample flowcharts for simultaneous serial transmission and reception.

Example (1) shows the flowchart when data is read and written for each transmit/receive operation, and example (2) shows the flowchart when data is read and written for multiple transmit/receive operations.

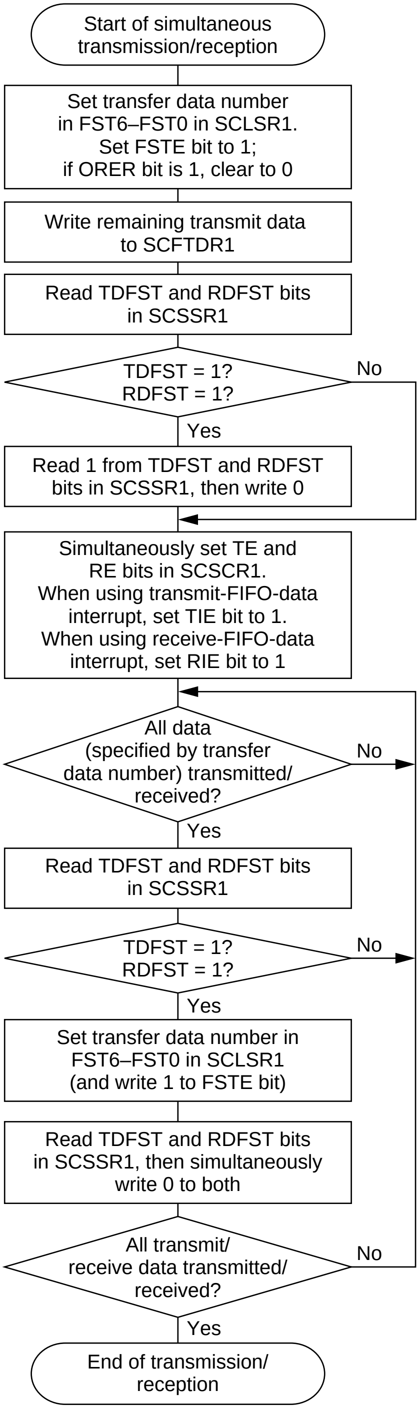

Start of simultaneous transmission/reception
Set transfer data number in FST6–FST0 in SCLSR1. Set FSTE bit to 1; if ORER bit is 1, clear to 0
Write remaining transmit data to SCFTDR1
Read TDFST and RDFST bits in SCSSR1
TDFST = 1? RDFST = 1?
No
Yes
Read TDFST and RDFST bits in SCSSR1, then write 0
Simultaneously set TE and RE bits in SCSCR1. When using transmit-FIFO-data interrupt, set TIE bit to 1. When using receive-FIFO-data interrupt, set RIE bit to 1
All data (specified by transfer data number) transmitted/ received?
No
Yes
Read TDFST and RDFST bits in SCSSR1
TDFST = 1? RDFST = 1?
No
Yes
End of transmission/ reception

1. Set the transfer data number in the line status register (SCLSR1), and set the FSTE bit to 1.
   Read the ORER bit in SCLSR1, and if 1, clear to 0.
2. Write the remaining transmit data to SCFTDR1, then read the TDFST and RDFST bits in SCSSR1, and if 1, clear to 0.
3. Transmission/reception starts when the TE and RE bits in SCSCR1 are set to 1. The TE and RE bits must be set simultaneously. When the data specified by the transfer data number has been transmitted/received, the TDFST and RDFST bits are set to 1.

When writing more transmit data and reading more receive data to continue transmission/reception:

4. Read receive data from SCFRDR1.
5. Set the TFRST and RFRST bits in SCFCR1 to 1 to clear the transmit and receive FIFOs, then clear TFRST and RFRST to 0.
6. Write transmit data to SCFTDR1.
7. Set the transfer data number in SCLSR1. Also write 1 to the FSTE bit.
   If the ORER bit is 1, read 1 from it, then clear it to 0.
8. Transmission/reception starts when 1 is read from the TDFST and RDFST bits in SCSSR1 and then these bits are cleared to 0.
   The TDFST and RDFST bits must be cleared to 0 simultaneously.
   When the data specified by the transfer data number has been transmitted/ received, the TDFST and RDFST bits are set to 1.

Figure 17.6 Sample Simultaneous Serial Data Transmission/Reception Flowchart (1)


Start of simultaneous transmission/reception
Set transfer data number in FST6–FST0 in SCLSR1. Set FSTE bit to 1; if ORER bit is 1, clear to 0
Write remaining transmit data to SCFTDR1
Read TDFST and RDFST bits in SCSSR1
TDFST = 1? RDFST = 1?
No
Yes
Read 1 from TDFST and RDFST bits in SCSSR1, then write 0
Simultaneously set TE and RE bits in SCSCR1. When using transmit-FIFO-data interrupt, set TIE bit to 1. When using receive-FIFO-data interrupt, set RIE bit to 1
All data (specified by transfer data number) transmitted/ received?
No
Yes
Read TDFST and RDFST bits in SCSSR1
TDFST = 1? RDFST = 1?
No
Yes
Set transfer data number in FST6–FST0 in SCLSR1 (and write 1 to FSTE bit)
Read TDFST and RDFST bits in SCSSR1, then simultaneously write 0 to both
All transmit/ receive data transmitted/ received?
No
Yes
End of transmission/ reception

1. Set the transfer data number in the line status register (SCLSR1), and set the FSTE bit to 1. Read the ORER bit in SCLSR1, and if 1, clear to 0.

2. Write the remaining transmit data to SCFTDR1, then read the TDFST and RDFST bits in SCSSR1, and if 1, clear to 0.

3. Transmission/reception starts when the TE and RE bits in SCSCR1 are set to 1. The TE and RE bits must be set simultaneously.
When the data specified by the transfer data number has been transmitted/received, the TDFST and RDFST bits are set to 1.

4. Set the transfer data number in SCLSR1 while the TDFST and RDFST bits in SCSSR1 are set to 1.
Also write 1 to the FSTE bit.
If the ORER bit is 1, read 1 from it, then clear it to 0.

5. Transmission/reception starts when 1 is read from the TDFST and RDFST bits in SCSSR1 and then these bits are cleared to 0.
The TDFST and RDFST bits must be cleared to 0 simultaneously.

Note: When continuing transmit/receive operations after the end of transmission/reception, before writing the transmit data and after reading the receive data, set the TFRST bit and RFRST bit in SCFCR0 to 1 and clear the transmit/receive FIFOs, then clear the TFRST bit and RFRST bit to 0.

Figure 17.6 Sample Simultaneous Serial Data Transmission/Reception Flowchart (2)

17.4 SCIF1 Interrupt Sources and the DMAC

SCIF1 has two interrupt sources: the transmit-FIFO-data interrupt (TXI) request and the receive-FIFO-data interrupt (RXI) request.

Table 17.3 shows the interrupt sources and their order of priority. The interrupt sources can be enabled or disabled by means of the TIE and RIE bits in SCSCR1. A separate interrupt request is sent to the interrupt controller for each of these interrupt sources.

When the TDFST flag in SCSSR1 is set to 1, a TXI interrupt request is generated. The DMAC can be activated and data transfer performed on generation of a TXI interrupt request.

When the RDFST flag in SCSSR1 is set to 1, an RXI interrupt request is generated. The DMAC can be activated and data transfer performed on generation of an RXI interrupt request.

When using the DMAC for transmission/reception, set and enable the DMAC before making SCIF1 settings. See section 14, Direct Memory Access Controller (DMAC), for details of the DMAC setting procedure.

Table 17.3 SCIF1 Interrupt Sources

Interrupt Source	Description	DMAC Activation	Priority on Reset Release
RXI	Interrupt initiated by receive FIFO data flag (RDFST)	Possible	High ↕
TXI	Interrupt initiated by transmit FIFO data flag (TDFST)	Possible	Low

See section 5, Exception Handling, for priorities and the relationship with non-SCIF1 interrupts.

17.5 Timing of TDFST, RDFST, and TEND Bit Setting

Figure 17.7 shows the timing for the setting of bits TDFST, RDFST, and TEND.

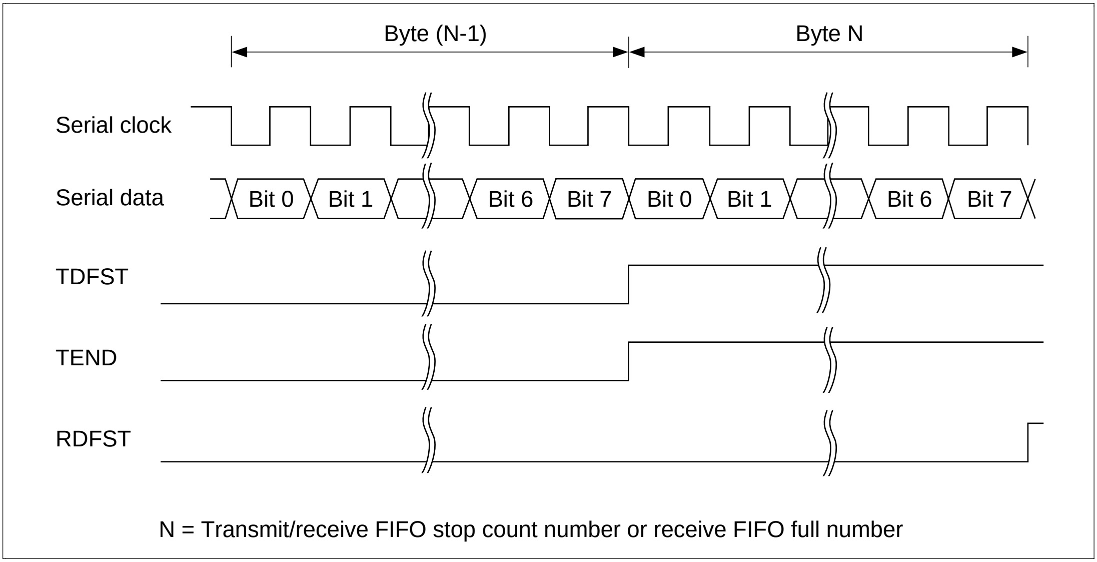


Figure 17.7 Timing of TDFST, RDFST, and TEND Bit Setting

The TDFST and TEND bits are set when the last data is transferred from SCFTDR1 to SCTSR1. The RDFST bit is set when the last data is transferred from SCRSR1 to SCFRDR1.

17.6 Usage Notes

Note the following when using SCIF1.

Interrupt Acceptance during DMAC Burst Transfer: SCIF1 interrupts (transmit-FIFO-data interrupt and receive-FIFO-data interrupt) are not accepted during DMAC burst transfer.

Reading/Writing in Transmit FIFO Full and Receive FIFO Empty States: SCFTDR1 is a write-only register, but write data is ignored after the transmit FIFO becomes full.

SCFRDR1 is a read-only register, but read data is undefined after the receive FIFO becomes empty.

When Using FIFO Stop Function: When the FIFO stop function is used, set a value in the FIFO stop count bits (FST6–FST0) in the SCLSR register, then write H'00 to the FIFO stop count bits.

Example:

(Procedure for first transfer when using FIFO stop function)
When the FSTE bit is set, simultaneously set a value in the FIFO stop count bits (FST6–FST0). Then clear the FIFO stop count bits to H'00.

```
  MOV.W  #H'2718,R0
  MOV.W  R0,@(SCLSR_OFFSET,GBR)    ; Set 40 bytes in FIFO stop count bits

  MOV.W  #H'0018,R0
  MOV.W  R0,@(SCLSR_OFFSET,GBR)     ; Clear FIFO stop count bits to 0
```

(Procedure for second and subsequent transfers when using FIFO stop function)
After completion of the preceding reception, set a value in the FIFO stop count bits (FST6–FST0) while the RDFST/TDFST bits are set to 1 (set 1 in the FSTE bit at this time). Next, clear the RDFST/TDFST bits and then clear the FIFO stop count bits to H'00.

```
  MOV.B  #H'06,R0
  MOV.B  R0,@(SCFCR_OFFSET,GBR)     ; Reset transmit/receive FIFO
  MOV.B  #H'00,R0
  MOV.B  R0,@(SCFCR_OFFSET,GBR)

  MOV.W  #H'2718,R0
  MOV.W  R0,@(SCLSR_OFFSET,GBR)     ; Set 40 bytes in FIFO stop count bits

  MOV.W  @(SCSSR_OFFSET,GBR),R0
  MOV    #H'00,R0
  MOV.W  R0,@(SCSSR_OFFSET,GBR)     ; Clear RDFST/TDFST flags to 0

  MOV.W  #H'0018,R0
  MOV.W  R0,@(SCLSR_OFFSET,GBR)     ; Clear FIFO stop count bits to 0
```

Supplementary Explanation:
The timing for latching of FST6–FST0 as the transmit/receive FIFO stop counter is as follows.

When using the transmit FIFO stop function:
1. When the TDFST bit is 0 and the FSTE bit 0 value is changed to 1
2. When the TDFST bit is 1

When using the receive FIFO stop function:
1. When the RDFST bit is 0 and the FSTE bit 0 value is changed to 1
2. When the RDFST bit is 1

Section 18 Serial Communication Interface with FIFO (SCIF2)

18.1 Overview

SCIF2 is a serial communication interface with on-chip FIFO buffers (Serial Communication Interface with FIFO: SCIF). SCIF2 can perform asynchronous and synchronous serial communication.

A 16-stage FIFO register is provided for both transmission and reception, enabling fast, efficient, and continuous communication.

18.1.1 Features

SCIF2 features are listed below.

- Asynchronous mode

  Serial data communication is executed using an asynchronous system in which synchronization is achieved character by character. Serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA).

  There is a choice of 8 serial data communication formats.

  — Data length: 7 or 8 bits

  — Stop bit length: 1 or 2 bits

  — Parity: Even/odd/none

  — LSB-first transfer

  — Receive error detection: Parity, overrun, and timeout errors

  — Break detection: If a framing error is following by at least one frame at the space “0” (low) level, a break is detected.

- 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 communication format.

  — Data length: 8 bits

  — LSB-first transfer

- Full-duplex communication capability

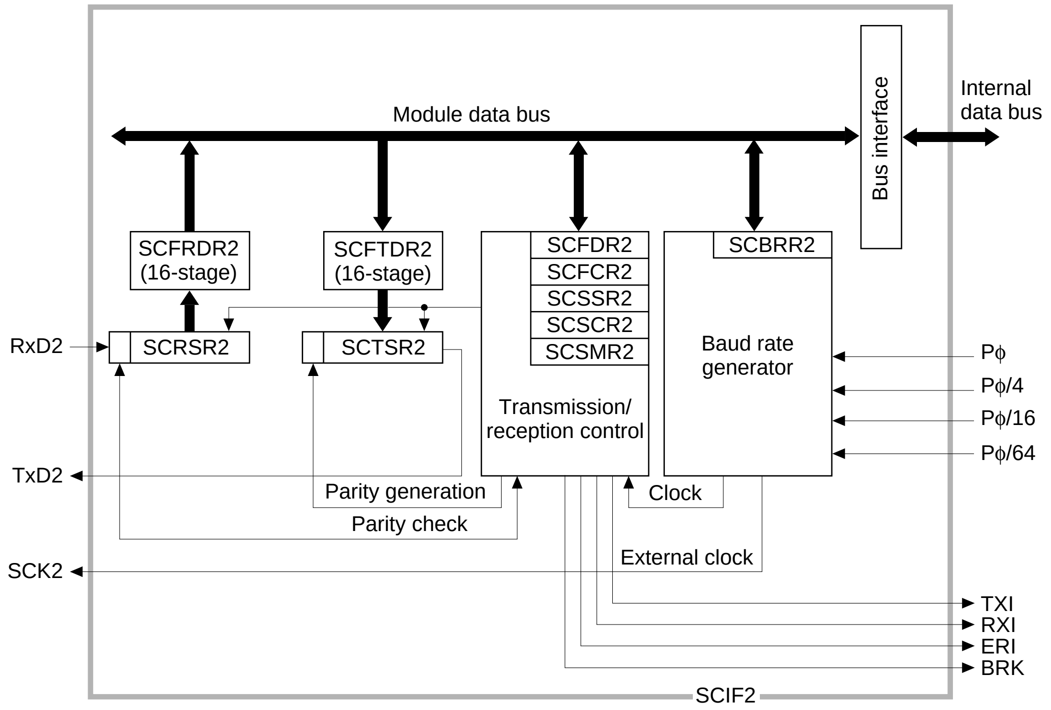
  The transmitter and receiver are independent units, enabling transmission and reception to be performed simultaneously.

  The transmitter and receiver both have a 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, SCIF2 can be stopped by halting its clock supply to reduce power consumption.
- The amount of data in the transmit/receive FIFO registers, and the number of receive errors in the receive data in the receive FIFO register, can be ascertained.
- The contents of the transmit FIFO data register (SCFTDR2) and receive FIFO data register (SCFRDR2) are undefined after a power-on or manual reset. Other registers are initialized by a power-on or manual reset, and retain their values in standby mode and in the module standby state. For details see section 18.1.4, Register Configuration.

18.1.2 Block Diagrams

Figure 18.1 shows a block diagram of SCIF2.



SCRSR2:	Receive shift register	SCSCR2:	Serial control register
SCFRDR2:	Receive FIFO data register	SCSSR2:	Serial status register
SCTSR2:	Transmit shift register	SCBRR2:	Bit rate register
SCFTDR2:	Transmit FIFO data register	SCFCR2:	FIFO control register
SCSMR2:	Serial mode register	SCFDR2:	FIFO data count register

Figure 18.1 Block Diagram of SCIF2

18.1.3 Pin Configuration

Table 18.1 shows the SCIF2 pin configuration.

Table 18.1 SCIF2 Pins

	Pin Name	Abbreviation	I/O	Function
SCIF2	Serial clock pin	SCK2	I/O	Clock input/output
	Receive data pin	RxD2	Input	Receive data input
	Transmit data pin	TxD2	Output	Transmit data output

Note: These pins are made to function as serial pins by performing SCIF2 operation settings with the TE and RE bits in SCSCR2.

18.1.4 Register Configuration

SCIF2 has the internal registers shown in table 18.2. These registers are used to specify the data format and bit rate, and to perform transmitter/receiver control.

Table 18.2 SCIF2 Registers

	Name	Abbreviation	R/W	Initial Value	Address	Access Size
SCIF2	Serial mode register	SCSMR2	R/W	H'00	H'A4002040	8
	Bit rate register	SCBRR2	R/W	H'FF	H'A4002042	8
	Serial control register	SCSCR2	R/W	H'00	H'A4002044	8
	Transmit FIFO data register	SCFTDR2	W	Undefined	H'A4002046	8
	Serial status register	SCSSR2	R/(W)*	H'0060	H'A4002048	16
	Receive FIFO data register	SCFRDR2	R	Undefined	H'A400204A	8
	FIFO control register	SCFCR2	R/W	H'00	H'A400204C	8
	FIFO data count register	SCFDR2	R	H'0000	H'A400204E	16

Note: * Only 0 can be written to bits 7, 5, 4, and 1, to clear the flags.

18.2 Register Descriptions

18.2.1 Receive Shift Register (SCRSR2)

SCRSR2 is the register used to receive serial data.

SCIF2 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 data register, SCFRDR2, automatically.

SCRSR2 cannot be directly read or written to by the CPU.

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

18.2.2 Receive FIFO Data Register (SCFRDR2)

SCFRDR2 is a 16-stage FIFO register that stores received serial data.

When SCIF2 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 data 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 data register, an undefined value will be returned. When the receive FIFO data 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.

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

18.2.3 Transmit Shift Register (SCTSR2)

SCTSR2 is the register used to transmit serial data.

To perform serial data transmission, SCIF2 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.

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

18.2.4 Transmit FIFO Data Register (SCFTDR2)

SCFTDR2 is a 16-stage FIFO data register that stores data for serial transmission.

If SCTSR2 is empty when transmit data has been written to SCFTDR2, SCIF2 transfers the transmit data written in SCFTDR2–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.

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

18.2.5 Serial Mode Register (SCSMR2)

SCSMR2 is an 8-bit register used to set SCIF2'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'00 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bit:	7	6	5	4	3	2	1	0
	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	—	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	R/W	R/W

Bit 7—Communication Mode (C/$\overline{A}$): Selects asynchronous mode or synchronous mode as the SCIF2 operating mode. This bit should be written to in the initialization procedure after a power-on reset, and its value should not subsequently be changed.

Bit 7: C/$\overline{A}$	Description	
0	Asynchronous mode	(Initial value)
1	Synchronous mode	

Bit 6—Character Length (CHR): Selects 7 or 8 bits as the asynchronous mode data length. 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 the transmit FIFO data register (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. 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/$\overline{E}$ 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/$\overline{E}$): Selects either even or odd parity for use in parity addition and checking. The O/$\overline{E}$ bit setting is only valid when the PE bit is set to 1, enabling parity bit addition and checking. The O/$\overline{E}$ bit setting is invalid in synchronous mode, and when parity addition and checking is disabled in asynchronous mode.

Bit 4: O/$\overline{E}$	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. The STOP bit setting is only valid in asynchronous mode. When 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—Reserved: This bit is always read as 0. The write value should always be 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 relationship between the clock source, the bit rate register setting, and the baud rate, see section 18.2.8, 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

18.2.6 Serial Control Register (SCSCR2)

The SCSCR2 register performs enabling or disabling of SCIF2 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'00 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

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	R	R/W	R/W

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 is set to 1 in the serial status 1 register (SCSSR2).

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): In asynchronous mode, this bit enables or disables generation of a receive-FIFO-data-full interrupt (RXI) request when the RDF flag in SCSSR2 is set to 1, a receive-error interrupt (ERI) request when the ER flag in SCSSR2 is set to 1, and a break interrupt (BRI) request when the BRK flag in SCSSR2 is set to 1. In synchronous mode, this bit enables or disables generation of a receive-data-full interrupt (RXI) request when the RDF flag in SCSSR2 is set.

Bit 6: RIE	Description
0	Receive-FIFO-data-full interrupt (RXI) request, receive-error interrupt (ERI) request, and break interrupt (BRI) request disabled* (Initial value)
1	Receive-FIFO-data-full interrupt (RXI) request, receive-error interrupt (ERI) request, and break interrupt (BRI) request enabled

Note: * An RXI interrupt requests 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 or BRK 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 SCIF2.

Bit 5: TE	Description
0	Transmission disabled (Initial value)
1	Transmission enabled*

Note: * Serial mode register (SCSMR2) and FIFO control register (SCFCR2) settings must be made, the transfer 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 SCIF2.

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, and ORER flags, which retain their states.

*2 Serial mode register (SCSMR2) and FIFO control register (SCFCR2) settings must be made, the receive format decided, and the receive FIFO reset, before the RE bit is set to 1.

Bits 3 and 2—Reserved: These bits are always read as 0. The write value should always be 0.

Bits 1 and 0—Clock Enable 1 and 0 (CKE1, CKE0): These bits select the SCIF2 clock source. The CKE1 and CKE0 bits must be set before determining the SCIF2 operating mode with SCSMR2.

Bit 1: CKE1	Bit 0: CKE0	Description
0	0	Internal clock/SCK2 pin functions as input pin (input signal ignored) (Initial value)
	1	Internal clock/SCK2 pin functions as serial clock output*1
1	0	External clock/SCK2 pin functions as clock input*2
	1	External clock/SCK2 pin functions as clock input*2

Notes: *1 In asynchronous mode, outputs a clock with a frequency of 16 times the bit rate.
In synchronous mode, outputs a clock with a frequency equal to the bit rate.
*2 In asynchronous mode, inputs a clock with a frequency of 16 times the bit rate.
In synchronous mode, inputs a clock with a frequency equal to the bit rate.
When an external clock is not input, set bits CKE1 and CKE0 to 00 or 01.

18.2.7 Serial Status Register (SCSSR2)

SCSSR2 is a 16-bit register. The lower 8 bits consist of status flags that indicate the operating status of SCIF2, and the upper 8 bits indicate the number of receive errors in the data in the receive FIFO register.

SCSSR2 can be read or written to at all times. However, 1 cannot be written to the ER, TDFE, BRK, RDF, and DR status flags. Also note that in order to clear these flags to 0, they must be read as 1 beforehand. The TEND, FER, PER, FER3 to FER0, and PER3 to PER0 flags are read-only flags and cannot be modified.

SCSSR2 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.

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	R/(W)*	R/(W)*	R	R	R/(W)*	R/(W)*

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

Bits 15–12—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 SCSSR2 is set, the value indicated by bits 15–12 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–PER0 will be 0.

Bits 11–8—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 SCSSR2 is set, the value indicated by bits 11–8 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–FER0 will be 0.

Bit 7—Receive Error (ER): In asynchronous mode, indicates that a framing error or parity error occurred during reception.*

Note: * 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 SCSSR2 can be used to determine whether there is a receive error in the data read from SCFRDR2.

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 SCIF2 checks whether the stop bit at the end of the receive data is 1 when reception ends, and the stop bit is 0* • 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/$\overline{\text{E}}$ bit in SCSMR2

Note: * 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 condition] When data is written to SCFTDR2
1	Transmission has been ended (Initial value) [Setting conditions] When there is no transmit data in SCFTDR2 on transmission 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 = 0 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): In asynchronous mode, 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): In asynchronous mode, 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 data is read when SCFRDR2 is empty, an undefined value will be returned. 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
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)

18.2.8 Bit Rate Register (SCBRR2)

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.

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

The SCBRR2 setting is found from the following equation.

Asynchronous mode:

$$N = \frac{P\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Synchronous mode:

$$N = \frac{P\phi}{8 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Where B: Bit rate (bits/s)

N: Asynchronous mode = SCBRR2 setting for baud rate generator ($0 \leq N \leq 255$)
Synchronous mode = SCBRR2 setting for baud rate generator ($1 \leq N \leq 255$)

$P\phi$: 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:

$$\text{Error (\%)} = \left\{ \frac{P\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

Tables 18.3 and 18.4 show sample SCBRR2 settings in asynchronous mode and synchronous mode.

Table 18.3 Examples of Bit Rates and SCBRR2 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

Blank: No setting is available.

—: A setting is available but error occurs.

Table 18.3 Examples of Bit Rates and SCBRR2 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 18.3 Examples of Bit Rates and SCBRR2 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			26		
Bit Rate (Bits/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	106	–0.44	3	108	0.08	3	114	0.36
150	3	77	0.16	3	79	0.00	3	84	–0.43
300	2	155	0.16	2	159	0.00	2	168	0.16
600	2	77	0.16	2	79	0.00	2	84	–0.43
1200	1	155	0.16	1	159	0.00	1	168	0.16
2400	1	77	0.16	1	79	0.00	1	84	–0.43
4800	0	155	0.16	0	159	0.00	0	168	0.16
9600	0	77	0.16	0	79	0.00	0	84	–0.43
19200	0	38	0.16	0	39	0.00	0	41	0.76
31250	0	23	0.00	0	24	–1.70	0	25	0.00
38400	0	19	–2.34	0	19	0.00	0	20	0.76

Table 18.4 Examples of Bit Rates and SCBRR2 Settings in Synchronous Mode

Bit Rate (Bits/s)	Pφ (MHz) 4		8		16		26	
	n	N	n	N	n	N	n	N
110	—	—	—	—	—	—	—	—
250	2	249	3	124	3	249	3	405
500	2	124	2	249	3	124	3	202
1 k	1	249	2	124	2	249	2	405
2.5 k	1	99	1	199	2	99	2	161
5 k	0	199	1	99	1	199	2	80
10 k	0	99	0	199	1	99	1	161
25 k	0	39	0	79	0	159	1	64
50 k	0	19	0	39	0	79	0	129
100 k	0	9	0	19	0	39	0	64
250 k	0	3	0	7	0	15	0	25
500 k	0	1	0	3	0	7	0	12
1 M	0	0*	0	1	0	3	—	—
2 M			0	0*	0	1	—	—

Note: As far as possible, the setting should be made so that the error is within 1%.

Blank: No setting is available.
—: A setting is available but error occurs.
*: Continuous transmission/reception is not possible.

Table 18.5 shows the maximum bit rate for various frequencies in asynchronous mode when using the baud rate generator. Tables 18.6 and 18.7 show the maximum bit rates when using external clock input.

Table 18.5 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
26	812500	0	0

Table 18.6 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
26	6.5000	406250

Table 18.7 Maximum Bit Rate with External Clock Input (Synchronous Mode)

Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (Bits/s)
8	0.6666	666666.6
16	1.3333	1333333.3
24	2.0000	2000000
26	2.1667	2166666.7

18.2.9 FIFO Control Register (SCFCR2)

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'00 by a power-on reset or manual reset. It is not initialized in standby mode or in the module standby state.

Bit:	7	6	5	4	3	2	1	0
	RTRG1	RTRG0	TTRG1	TTRG0	—	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

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 (SCSSR2).

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 (Initial value)
	1	4
1	0	8
	1	14

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 (SCSSR2).

The TDFE flag is set when, as the result of a transmit operation, the number of transmit data bytes in the transmit FIFO data register (SCFTDR2) falls to or below the trigger set number shown in the following table.

Bit 5: TTRG1	Bit 4: TTRG0	Transmit Trigger Number	
0	0	8 (8)	(Initial value)
	1	4 (12)	
1	0	2 (14)	
	1	0 (16)	

Note: Figures in parentheses are the number of empty bytes in SCFTDR2 when the flag is set.

Bit 3—Reserved: This bit is always read as 0. The write value should always be 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), enabling loopback testing.

Bit 0: LOOP	Description	
0	Loopback test disabled	(Initial value)
1	Loopback test enabled	

18.2.10 FIFO Data Count Register (SCFDR2)

SCFDR2 is a 16-bit register that indicates the number of data bytes stored in the transmit FIFO data register (SCFTDR2) and receive FIFO data register (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.

SCFDR2 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.

The upper 8 bits of SCFDR2 show the number of untransmitted data bytes in SCFTDR2.

A value of H'00 means that there is no transmit data, and a value of H'10 means that SCFTDR2 is full of transmit data.

The lower 8 bits of SCFDR2 show the number of receive data bytes in SCFRDR2.

A value of H'00 means that there is no receive data, and a value of H'10 means that SCFRDR2 is full of receive data.

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

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

18.3 Operation

18.3.1 Overview

SCIF2 can carry out serial communication in asynchronous mode, in which synchronization is achieved character by character, and synchronous mode, in which synchronization is achieved with clock pulses.

16-stage FIFO buffers are provided for both transmission and reception, reducing the CPU overhead and enabling fast, continuous communication to be performed.

18.3.2 Asynchronous Mode

The transmission format is selected using the serial mode register (SCSMR2), as shown in table 18.8. The SCIF2 clock source is determined by the CKE1 and CKE0 bits in the serial control register (SCSCR2).

- 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, 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 SCIF2 clock source
  - — When internal clock is selected: SCIF2 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 18.8 SCSMR2 Settings for Serial Transfer Format Selection

SCSMR2 Settings				SCIF2 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	None	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

18.3.3 Serial Operation in Asynchronous Mode

Data Transfer Format: Table 18.9 shows the transfer formats that can be used in asynchronous mode. Any of 8 transfer formats can be selected according to the SCSMR2 settings.

Table 18.9 Serial Transfer Formats

SCSMR2 Settings			Serial Transfer Format and Frame Length											
CHR	**MP**	**STOP**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
0	0	0	S	8-bit data								STOP		
		1	S	8-bit data								STOP	STOP	
	1	0	S	8-bit data								P	STOP	
		1	S	8-bit data								P	STOP	STOP
1	0	0	S	7-bit data							STOP			
		1	S	7-bit data							STOP	STOP		
	1	0	S	7-bit data							P	STOP		
		1	S	7-bit data							P	STOP	STOP	

S: Start bit
STOP: Stop bit
P: Parity bit

Clock: Either an internal clock generated by the on-chip baud rate generator or an external clock input at the SCK2 pin can be selected as the serial clock for SCIF2, according to the setting of the CKE1 and CKE0 bits in SCSCR2.

When an external clock is input at the SCK2 pin, the input clock frequency should be 16 times the bit rate used.

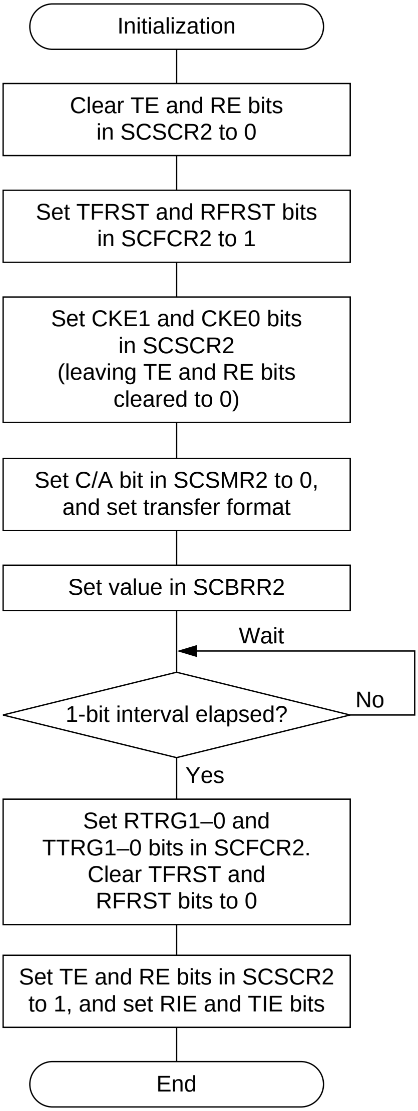
Data Transfer Operations

SCIF2 Initialization: Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR2 to 0, then initialize SCIF2 as described below.

When the transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the transmit shift register (SCTSR2) is initialized. Note that clearing the TE and RE bits to 0 does not change the contents of SCSSR2, SCFTDR2, or SCFRDR2. The TE bit should be cleared to 0 after all transmit data has been sent and the TEND bit in SCSSR2 has been set to 1. Clearing to 0 can also be performed during transmission, but the data being transmitted will go to the high-impedance state after the clearance. Before setting TE to 1 again to start transmission, the TFRST bit in 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 18.2 shows a sample SCIF2 initialization flowchart.

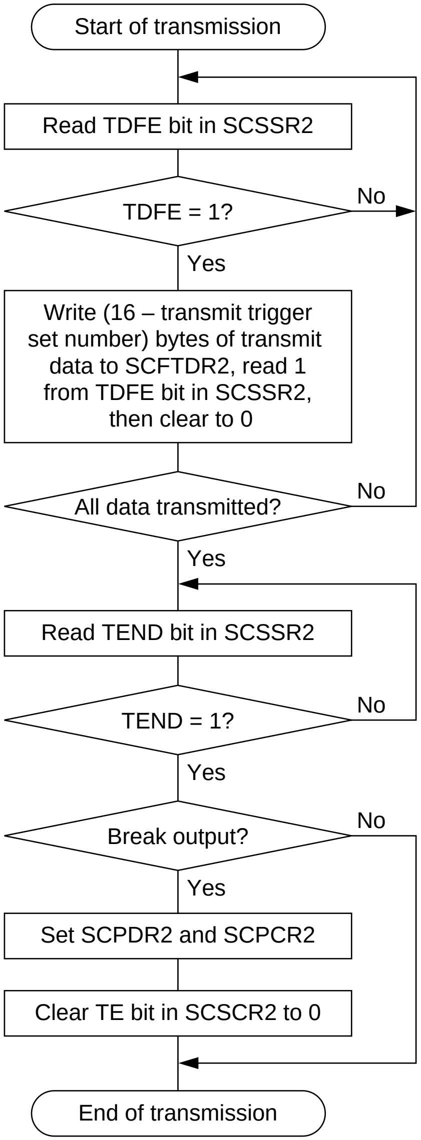

Initialization
Clear TE and RE bits in SCSCR2 to 0
Set TFRST and RFRST bits in SCFCR2 to 1
Set CKE1 and CKE0 bits in SCSCR2 (leaving TE and RE bits cleared to 0)
Set C/A bit in SCSMR2 to 0, and set transfer format
Set value in SCBRR2
Wait
1-bit interval elapsed?
No
Yes
Set RTRG1–0 and TTRG1–0 bits in SCFCR2. Clear TFRST and RFRST bits to 0
Set TE and RE bits in SCSCR2 to 1, and set RIE and TIE bits
End

1. Set the clock selection in SCSCR2.
   Be sure to clear bits RIE, TIE, TE, and RE to 0.
2. Set the 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 and TIE bits.
   Setting the TE and RE bits enables the TxD2 and RxD2 pins to be used.

Figure 18.2 Sample SCIF2 Initialization Flowchart

Serial Data Transmission: Figure 18.3 shows a sample flowchart for serial transmission.

Use the following procedure for serial data transmission after enabling SCIF2 for transmission.



1. SCIF2 status check and transmit data write: Read the serial status register (SCSSR2) and check that the TDFE flag is set to 1, then write transmit data to SCFTDR2, read 1 from the TDFE, then clear this flag to 0. The number of 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 bit to 0.
3. Break output at the end of serial transmission:
   To output a break in serial transmission, set the port SC data register (SCPDR) and port SC control register (SCPCR), then clear the TE bit in 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 18.3 Sample Serial Transmission Flowchart

In serial transmission, SCIF2 operates as described below.

1. When data is written into SCFTDR2, SCIF2 transfers the data from SCFTDR2 to SCTSR2 and starts transmitting. Confirm that the TDFE flag in the serial status register (SCSSR2) 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 set number).

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 SCIF2 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 SCSSR2 is set to 1, the stop bit is sent, and then the line goes to the mark state in which 1 is output.

Figure 18.4 shows an example of the operation for transmission in asynchronous mode.

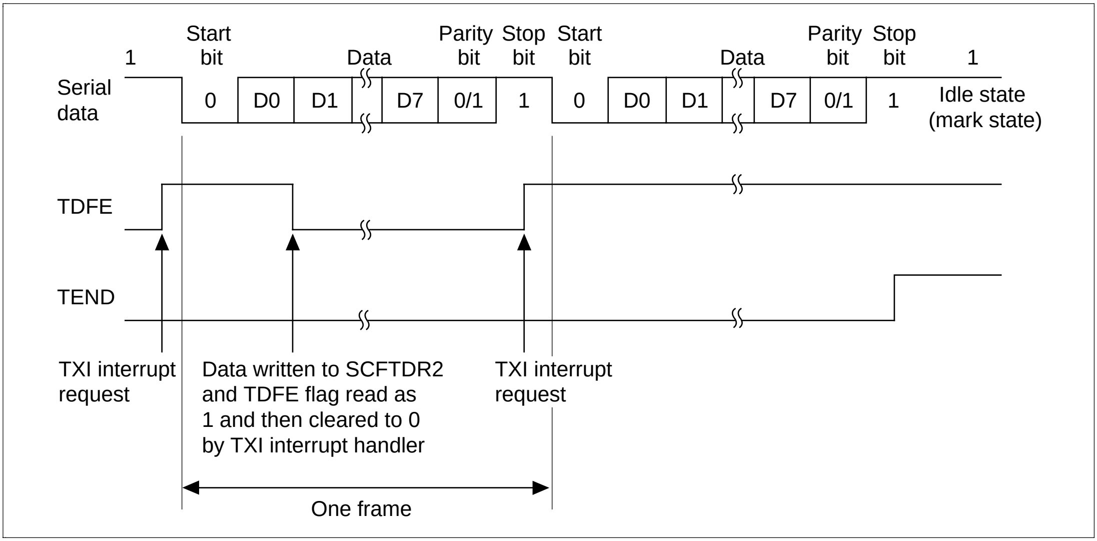
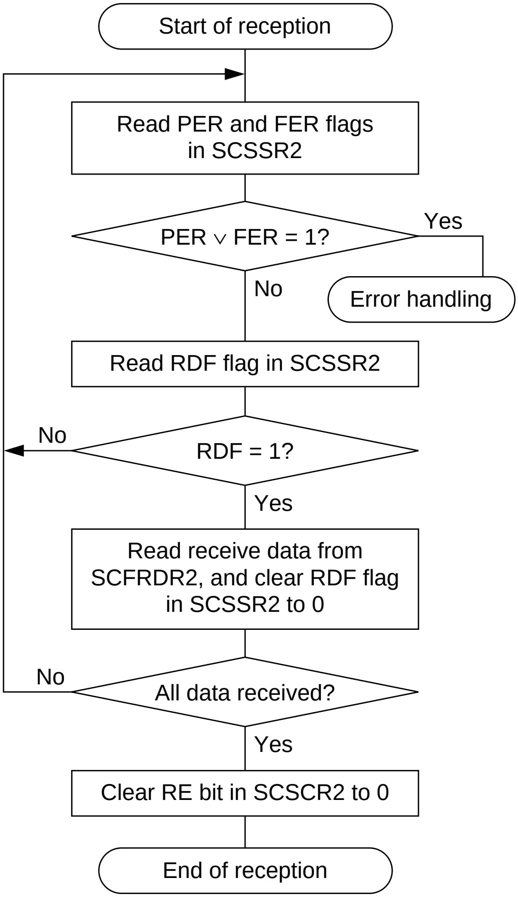


Figure 18.4 Example of Transmit Operation (Example with 8-Bit Data, Parity, One Stop Bit)

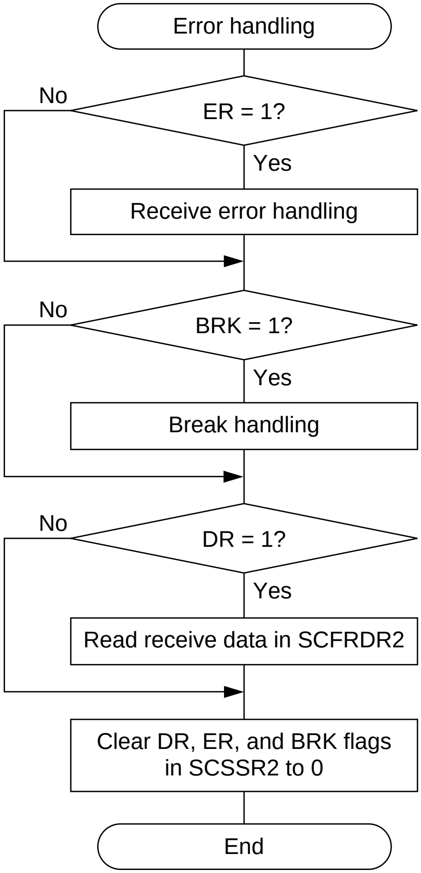
Serial Data Reception: Figures 18.5 and 18.6 show a sample flowchart for serial reception.

Use the following procedure for serial data reception after enabling SCIF2 for reception.



1. Receive error handling and break detection: Read the DR, ER, and BRK flags in SCSSR2 to identify any error, perform the appropriate error handling, then clear the DR, ER, and BRK flags to 0. In the case of a framing error, a break can also be detected by reading the value of the RxD pin.
2. SCIF2 status check and receive data read: Read SCSSR2 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 data bytes from SCFRDR2, read 1 from the RDF flag, and 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 18.5 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 SCSSR2.

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 18.6 Sample Serial Reception Flowchart (2)

In serial reception, SCIF2 operates as described below.

1. SCIF2 monitors the communication line, and if a 0 start bit is detected, performs internal synchronization and starts reception.

2. The received data is stored in SCR2SR2 in LSB-to-MSB order.

3. The parity bit and stop bit are received.

   After receiving these bits, SCIF2 carries out the following checks.

   a. Stop bit check: SCIF2 checks whether the stop bit is 1. If there are two stop bits, only the first is checked.
   b. SCIF2 checks whether receive data can be transferred from the receive shift register (SCRSR2) to SCFRDR2.
   c. Break check: SCIF2 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 a receive error occurs.

4. If the RIE bit in SCSCR2 is set to 1 when the RDF flag changes to 1, a receive-FIFO-data-full interrupt (RXI) request is generated.

   If the RIE 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 in SCSCR2 is set to 1 when the BRK flag changes to 1, a break reception interrupt (BRI) request is generated.

Figure 18.7 shows an example of the operation for reception in asynchronous mode.

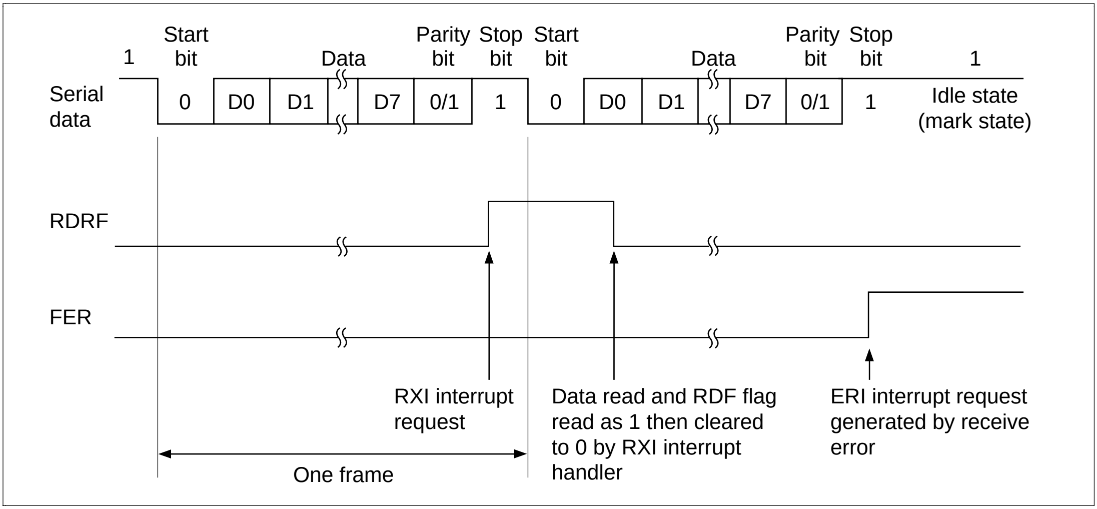


Figure 18.7 Example of SCIF2 Receive Operation (Example with 8-Bit Data, Parity, One Stop Bit)

18.3.4 Synchronous Mode

16-stage FIFO buffers are provided for both transmission and reception, reducing the CPU overhead and enabling fast, continuous communication to be performed.

The operating clock source is selected with the serial mode register (SCSMR2), and the SCIF2 clock source is determined by the CKE1 and CKE0 bits in the serial control register (SCSCR2).

- Transmit/receive format: Fixed 8-bit data
- Indication of amount of data stored in transmit and receive FIFO registers
- Choice of internal or external clock as SCIF2 clock source
  - When internal clock is selected: SCIF2 operates on the baud rate generator clock and outputs a serial clock externally.
  - When external clock is selected: SCIF2 operates on the input serial clock (the on-chip baud rate generator is not used).

18.3.5 Serial Operation in Synchronous Mode

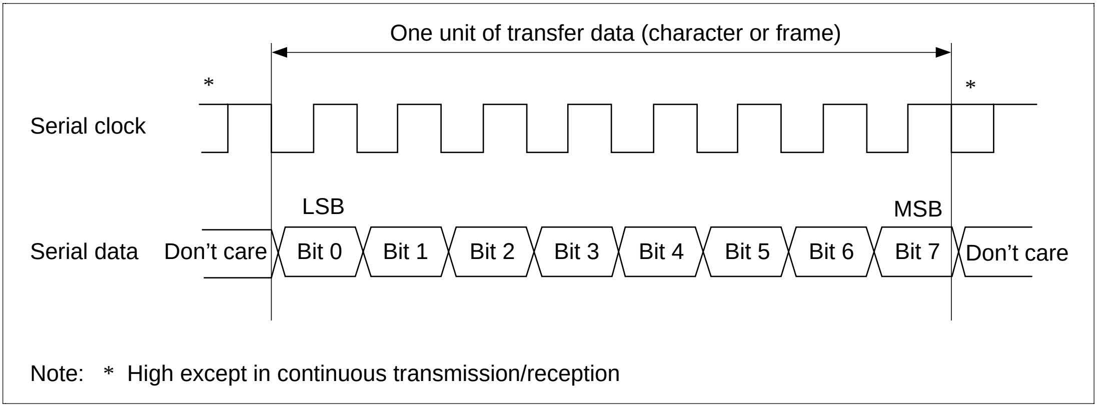

One unit of transfer data (character or frame)
*
*
Serial clock
LSB
MSB
Serial data
Don’t care
Bit 0
Bit 1
Bit 2
Bit 3
Bit 4
Bit 5
Bit 6
Bit 7
Don’t care
Note: * High except in continuous transmission/reception

Figure 18.8 Data Format in Synchronous Communication

In synchronous serial communication, data on the communication line is output from one fall of the serial clock to the next. Data is guaranteed valid at the rise of the serial clock.

In serial communication, each character is output starting with the LSB and ending with the MSB. After the MSB is output, the communication line remains in the state of the MSB.

In synchronous mode, SCIF2 receives data in synchronization with the rise 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 SCK2 pin can be selected, according to the setting of the CKE1 and CKE0 bits in SCSCR2.

When SCIF2 is operated on an internal clock, the serial clock is output from the SCK2 pin.

Eight serial clock pulses are output in the transfer of one character, and when no transmission/reception is performed the clock is fixed high. In receive-only operation, when the on-chip clock source is selected, clock pulses are output until the receive FIFO is full of data.

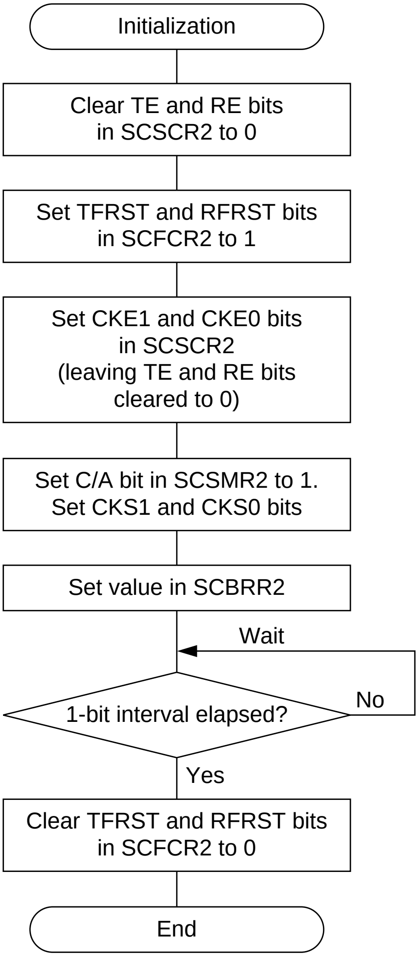
Data Transfer Operations

- **SCIF2 Initialization:** Before transmitting and receiving data, it is necessary to clear the TE and RE bits in SCSCR2 to 0, then initialize SCIF2 as described below.

When the clock source, etc., is changed, the TE and RE bits must be cleared to 0 before making the change using the following procedure. When the TE bit is cleared to 0, the transmit shift register (SCTSR2) is initialized. Note that clearing the TE and RE bits to 0 does not change the contents of SCSSR2, SCFTDR2, or SCFRDR2. The TE bit should be cleared to 0 after all transmit data has been sent and the TEND bit in SCSSR2 has been set to 1. When TE is cleared to 0, the TxD2 pin goes to the high-impedance state. 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 to 1 again to start transmission, the TFRST bit in SCFCR2 should first be set to 1 to reset SCFTDR2.

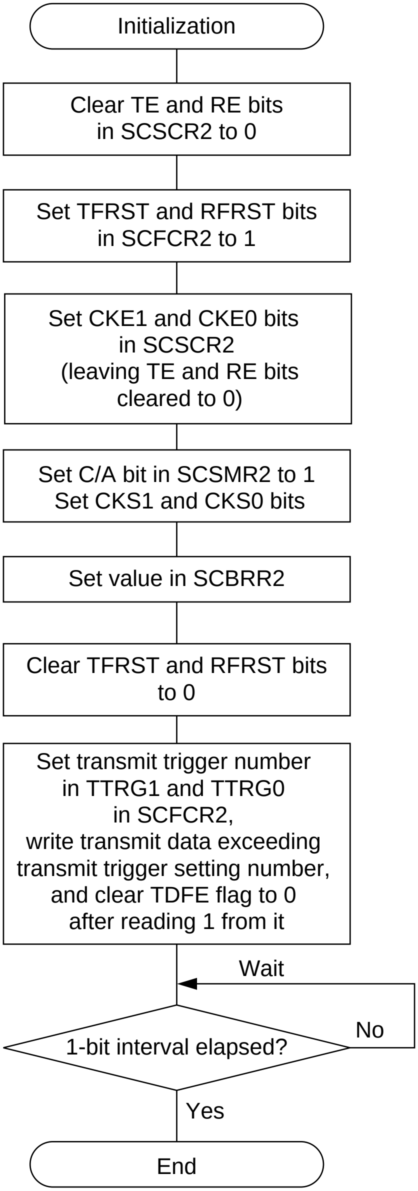
When an external clock is used the clock should not be stopped during operation, including initialization, since operation will be unreliable in this case.

Figure 18.9 shows a sample SCIF2 initialization flowchart.


Initialization
Clear TE and RE bits in SCSCR2 to 0
Set TFRST and RFRST bits in SCFCR2 to 1
Set CKE1 and CKE0 bits in SCSCR2 (leaving TE and RE bits cleared to 0)
Set C/A bit in SCSMR2 to 1. Set CKS1 and CKS0 bits
Set value in SCBRR2
Wait
1-bit interval elapsed?
No
Yes
Clear TFRST and RFRST bits in SCFCR2 to 0
End

1. Set the clock selection in SCSCR2. Be sure to clear bits RIE, TIE, TE, and RE to 0.
2. Be sure to set the TFRST and RFRST bits in SCFCR2 to 1, to reset the FIFOs.
3. Set the clock mode and clock source selection in SCSMR2.
4. Write a value corresponding to the bit rate into SCBRR2 (Not necessary if an external clock is used).
5. Clear the TFRST and RFRST bits in SCFCR2 to 0.

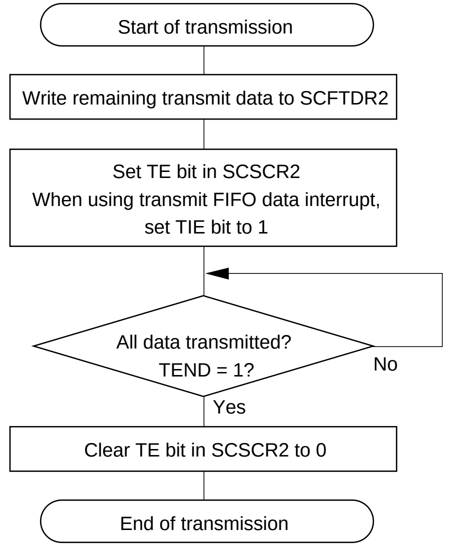
Figure 18.9 Sample SCIF2 Initialization Flowchart (1) (Reception)


Initialization
Clear TE and RE bits in SCSCR2 to 0
Set TFRST and RFRST bits in SCFCR2 to 1
Set CKE1 and CKE0 bits in SCSCR2 (leaving TE and RE bits cleared to 0)
Set C/A bit in SCSMR2 to 1 Set CKS1 and CKS0 bits
Set value in SCBRR2
Clear TFRST and RFRST bits to 0
Set transmit trigger number in TTRG1 and TTRG0 in SCFCR2, write transmit data exceeding transmit trigger setting number, and clear TDFE flag to 0 after reading 1 from it
Wait
No
1-bit interval elapsed?
Yes
End

1. Set the clock selection in SCSCR2. Be sure to clear bits RIE, TIE, TE, and RE to 0.
2. Be sure to set the TFRST and RFRST bits in SCFCR2 to 1, to reset the FIFOs.
3. Set the clock source selection in SCSMR2.
4. Write a value corresponding to the bit rate into SCBRR2.
5. Clear the TFRST and RFRST bits in SCFCR2 to 0.
6. Set the transmit trigger number, write transmit data exceeding the transmit trigger setting number, and clear the TDFE flag to 0 after reading it.
7. Wait for a 1-bit interval.

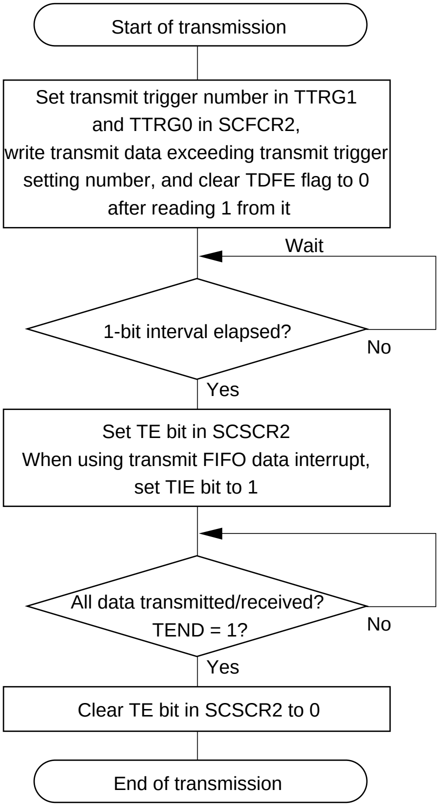
Figure 18.9 Sample SCIF2 Initialization Flowchart (2) (Transmission, or Simultaneous Transmission and Reception)

- **Serial Data Transmission:** Figure 18.10, 18.11 shows a sample flowchart for serial transmission.


Start of transmission
Write remaining transmit data to SCFTDR2
Set TE bit in SCSCR2
When using transmit FIFO data interrupt, set TIE bit to 1
All data transmitted?
TEND = 1?
No
Yes
Clear TE bit in SCSCR2 to 0
End of transmission

1. Write the remaining transmit data to SCFTDR2.
2. Transmission is started when the TE bit in SCSCR2 is set to 1.
3. After the end of transmission, clear the TE bit to 0.

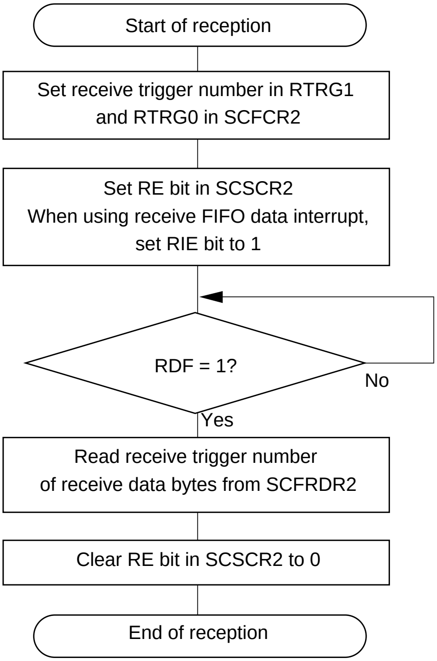
Figure 18.10 Sample SCIF2 Transmission Flowchart (1) (First Transmit Operation)


Start of transmission
Set transmit trigger number in TTRG1 and TTRG0 in SCFCR2, write transmit data exceeding transmit trigger setting number, and clear TDFE flag to 0 after reading 1 from it
Wait
1-bit interval elapsed?
No
Yes
Set TE bit in SCSCR2
When using transmit FIFO data interrupt, set TIE bit to 1
All data transmitted/received? TEND = 1?
No
Yes
Clear TE bit in SCSCR2 to 0
End of transmission

1. Set the transmit trigger number in SCFCR2.
2. Write transmit data to SCFTDR2, and clear the TDFE flag to 0 after reading 1 from it.
3. Wait for a 1-bit interval.
4. Transmission is started when the TE bit in SCSCR2 is set to 1.
5. After the end of transmission, clear the TE bit to 0.

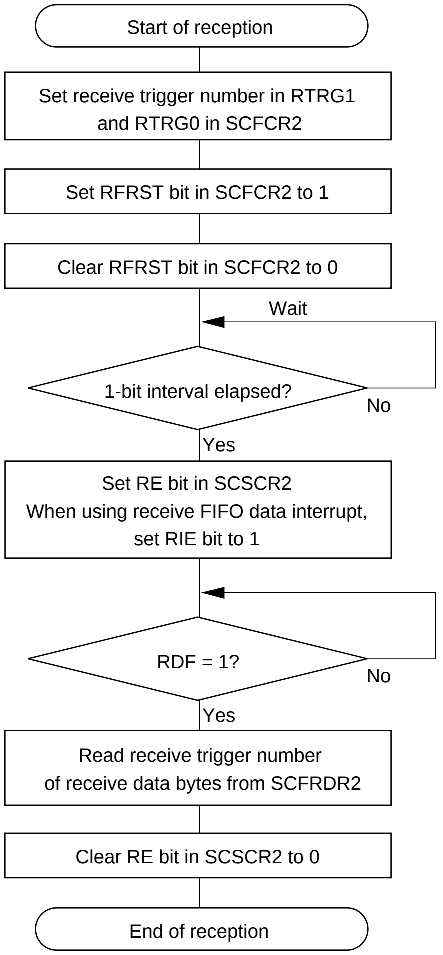
Figure 18.11 Sample SCIF2 Transmission Flowchart (2) (Second and Subsequent Transmit Operations)

- **Serial Data Reception:** Figure 18.12, 18.13 shows a sample flowchart for serial reception.


Start of reception
Set receive trigger number in RTRG1 and RTRG0 in SCFCR2
Set RE bit in SCSCR2
When using receive FIFO data interrupt, set RIE bit to 1
RDF = 1?
No
Yes
Read receive trigger number of receive data bytes from SCFRDR2
Clear RE bit in SCSCR2 to 0
End of reception

1. Set the receive trigger number in SCFCR2.
2. Reception is started when the RE bit in SCSCR2 is set to 1.
3. Read receive data while the RDF bit is 1.
4. After the end of reception, clear the RE bit to 0.

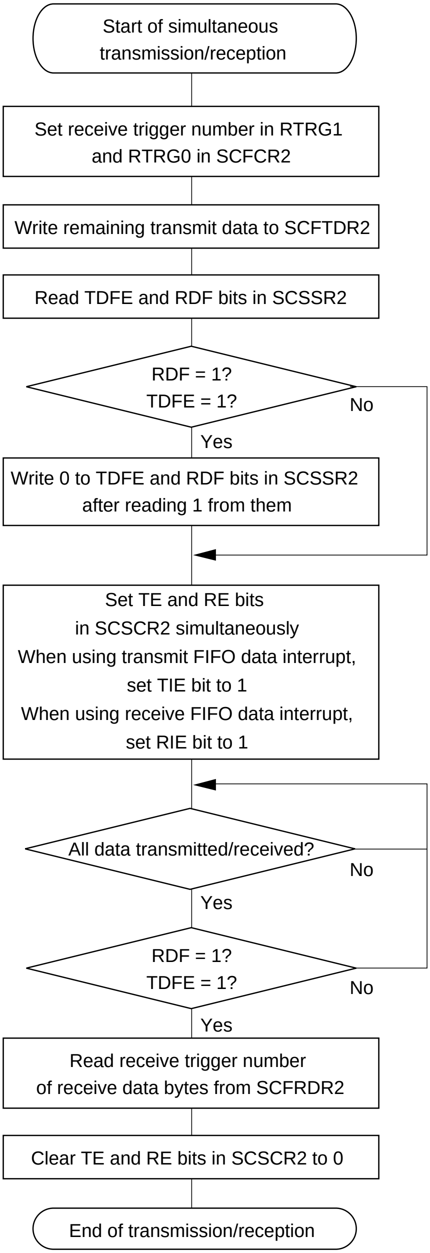
Figure 18.12 Sample SCIF2 Reception Flowchart (1) (First Receive Operation)


Start of reception
Set receive trigger number in RTRG1 and RTRG0 in SCFCR2
Set RFRST bit in SCFCR2 to 1
Clear RFRST bit in SCFCR2 to 0
Wait
1-bit interval elapsed?
No
Yes
Set RE bit in SCSCR2
When using receive FIFO data interrupt, set RIE bit to 1
RDF = 1?
No
Yes
Read receive trigger number of receive data bytes from SCFRDR2
Clear RE bit in SCSCR2 to 0
End of reception

1. Set the receive trigger number in SCFCR2.
2. Reset the receive FIFO.
3. Wait for a 1-bit interval.
4. Reception is started when the RE bit in SCSCR2 is set to 1.
5. Read receive data while the RDF bit is 1.
6. After the end of reception, clear the RE bit to 0.

Figure 18.13 Sample SCIF2 Reception Flowchart (2)
(Second and Subsequent Receive Operations)

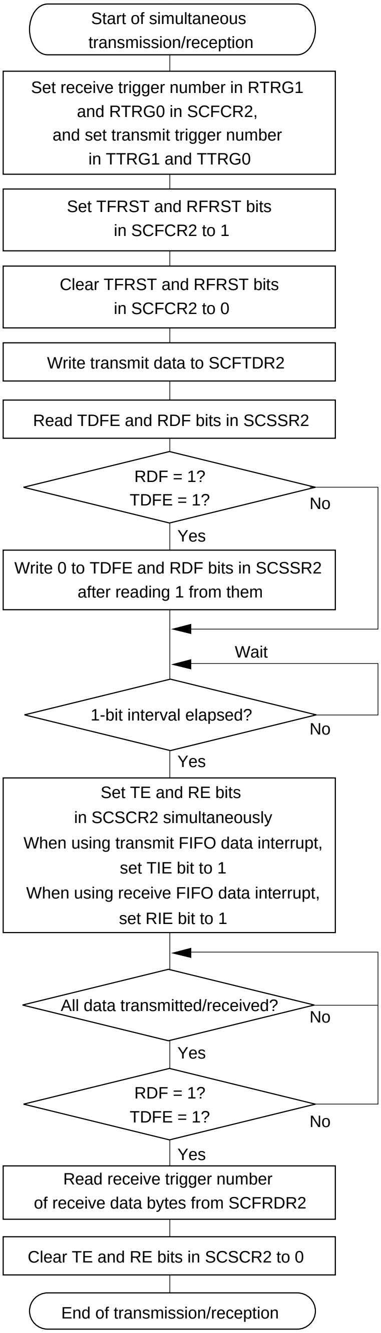
- **Simultaneous Serial Data Transmission and Reception:** Figure 18.14, 18.15 show a sample flowchart for simultaneous serial transmission and reception.



Start of simultaneous transmission/reception
Set receive trigger number in RTRG1 and RTRG0 in SCFCR2
Write remaining transmit data to SCFTDR2
Read TDFE and RDF bits in SCSSR2
RDF = 1?
TDFE = 1?
No
Yes
Write 0 to TDFE and RDF bits in SCSSR2 after reading 1 from them
Set TE and RE bits in SCSCR2 simultaneously
When using transmit FIFO data interrupt, set TIE bit to 1
When using receive FIFO data interrupt, set RIE bit to 1
All data transmitted/received?
No
Yes
RDF = 1?
TDFE = 1?
No
Yes
Read receive trigger number of receive data bytes from SCFRDR2
Clear TE and RE bits in SCSCR2 to 0
End of transmission/reception

1. Set the receive trigger number in SCFCR2.
2. Write the remaining transmit data to SCFTDR2, and if there is receive data in the FIFO, read receive data until there is less than the receive trigger setting number, read the TDFE and RDF bits in SCSSR2, and if 1, clear to 0.
3. Transmission/reception is started when the TE and RE bits in SCSCR2 are set to 1. The TE and RE bits must be set simultaneously.
4. After the end of transmission/reception, clear the TE and RE bits to 0.

Figure 18.14 Sample SCIF2 Simultaneous Transmission/Reception Flowchart (1) (First Transmit/Receive Operation)


Start of simultaneous transmission/reception
Set receive trigger number in RTRG1 and RTRG0 in SCFCR2, and set transmit trigger number in TTRG1 and TTRG0
Set TFRST and RFRST bits in SCFCR2 to 1
Clear TFRST and RFRST bits in SCFCR2 to 0
Write transmit data to SCFTDR2
Read TDFE and RDF bits in SCSSR2
RDF = 1? TDFE = 1?
No
Yes
Write 0 to TDFE and RDF bits in SCSSR2 after reading 1 from them
Wait
1-bit interval elapsed?
No
Yes
Set TE and RE bits in SCSCR2 simultaneously
When using transmit FIFO data interrupt, set TIE bit to 1
When using receive FIFO data interrupt, set RIE bit to 1
All data transmitted/received?
No
Yes
RDF = 1? TDFE = 1?
No
Yes
Read receive trigger number of receive data bytes from SCFRDR2
Clear TE and RE bits in SCSCR2 to 0
End of transmission/reception

1. Set the receive trigger number and transmit trigger number in SCFCR2.
2. Reset the receive FIFO and transmit FIFO.
3. Write transmit data to SCFTDR2, and if there is receive data in the FIFO, read receive data until there is less than the receive trigger setting number, read the TDFE and RDF bits in SCSSR2, and if 1, clear to 0.
4. Wait for a 1-bit interval.
5. Transmission/reception is started when the TE and RE bits in SCSCR2 are set to 1. The TE and RE bits must be set simultaneously.
6. After the end of transmission/reception, clear the TE and RE bits to 0.

Figure 18.15 Sample SCIF2 Simultaneous Transmission/Reception Flowchart (2) (Second and Subsequent Transmit/Receive Operations)

18.4 SCIF2 Interrupt Sources and the DMAC

SCIF2 supports four interrupts—transmit-FIFO-data-empty (TXI), receive-error (ERI), receive-FIFO-data-full (RXI), and break (BRI)—in asynchronous mode, and two interrupts—transmit-FIFO-data-empty (TXI) and receive-FIFO-data-full (RXI)—in synchronous mode.

Table 18.10 shows the interrupt sources and their order of priority. The interrupt sources can be enabled or disabled by means of the TIE and RIE bits in SCSCR2. A separate interrupt request is sent to the interrupt controller for each of these interrupt sources.

When the TDFE flag in SCSSR2 is set to 1, a TXI interrupt request is generated. The DMAC can be activated and data transfer performed on generation of a TXI interrupt request. When data exceeding the transmit trigger set number is written to SCFTDR2 by the DMAC, the TDFE flag is automatically cleared to 0.

When the RDF flag in SCSSR2 is set to 1, an RXI interrupt request is generated. The DMAC can be activated and data transfer performed on generation of an RXI interrupt request. When receive data in SCFRDR2 is read by the DMAC until the amount left is less than the receive trigger set number, the RDF flag is automatically cleared to 0.

When using the DMAC for transmission/reception, set and enable the DMAC before making SCIF2 settings. See section 14, Direct Memory Access Controller (DMAC), for details of the DMAC setting procedure.

When the BRK flag in SCSSR2 is set to 1, a BRI interrupt request is generated.

When the receive-FIFO-data-full interrupt (RXI interrupt by the RDF flag) is used to activate the DMAC, a value of 4 or more should be used for the receive FIFO data quantity trigger setting.

A receive trigger value of 1 is provided for the case where one-byte transfer is to be performed by DMA. For single-byte DMA transfer using the SCIF, therefore, set a receive trigger value of 1 and set 1 in the DMATCR register.

Table 18.10 SCIF2 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)	Possible	↕
BRI	Interrupt initiated by break flag (BRK)	Not possible	
TXI	Interrupt initiated by transmit FIFO data empty flag (TDFE)	Possible	Low

See section 5, Exception Handling, for priorities and the relationship with non-SCIF2 interrupts.

18.5 Usage Notes

Note the following when using SCIF2.

SCFTDR2 Writing and the TDFE Flag: The TDFE flag in the serial status register (SCSSR2) 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 (SCSSR2) 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 still equal to or greater than the trigger number after a read, the RDF flag will be set to 1 again if it is cleared to 0. RDF should therefore be cleared to 0 after being read as 1 after all receive data has been read.

The number of receive data bytes in SCFRDR2 can be found from the lower 8 bits of the FIFO data count register (SCFDR2).

Note on DMAC Transfer Operation in Case of Activation by Receive-FIFO-Full Interrupt: If the number of receive triggers is set at 4 or more (4, 8, or 14), the DMAC will not transfer all the receive data in the FIFO.

When the DMAC is activated by the receive-FIFO-full interrupt, use the following procedure to ensure that all the receive data in the FIFO is transferred.

- Example

  Conditions: 128-byte reception, four receive triggers set, all receive data to be transferred by the DMAC

1. Set D'124 in DMATCR.

   Receive data quantity (128) – number of receive triggers (4) = DMATCR set value (124)

2. Enable DMA transfer end interrupts, enable DMAC transfer, and then start the SCIF2 receive operation.
3. Clear the DE and TE bits in CHCR and the RDF flag in SCSSR2 in the DMA transfer end interrupt routine.

   When using CH1 or CH3, set bits 15 to 8 in CHCRA to H'0000; when using CH0 or CH2, set bits 7 to 0 in CHCRA to H'0000. The RIE bit in SCSCR2 is left set.
4. Perform a 4-byte read by software in the receive-FIFO-full interrupt routine.

   Receive data quantity (128) – DMATCR set value (124) = 4

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 SCIF2 stops transferring receive data to SCFRDR2 after receiving a break, the receive operation continues.

Receive Data Sampling Timing and Receive Margin: SCIF2 operates on a base clock with a frequency of 16 times the transfer rate.

In reception, SCIF2 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 18.16.

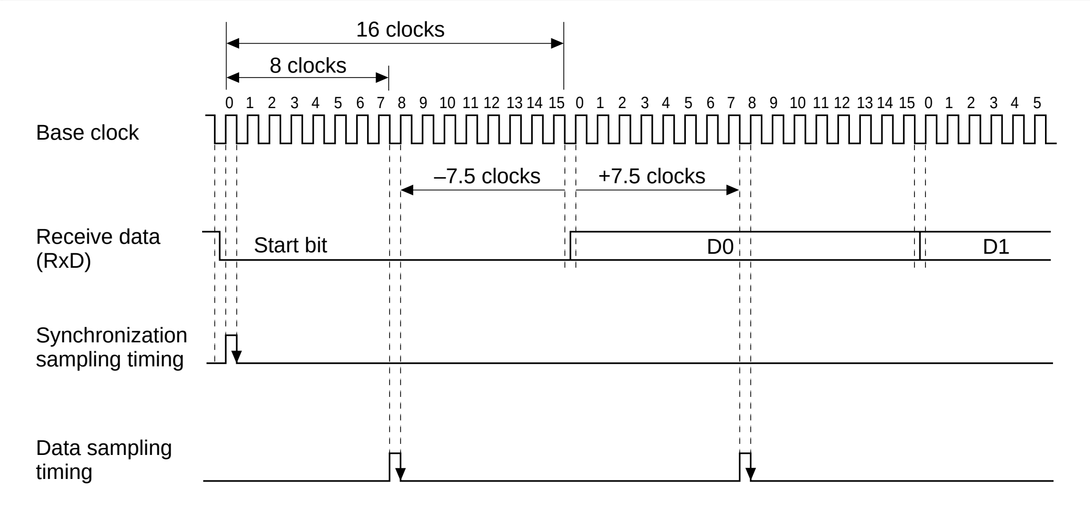


Figure 18.16   Receive Data Sampling Timing in Asynchronous Mode

The receive margin in asynchronous mode can therefore be expressed as shown in equation (1).

$$M = \left| \left(0.5 - \frac{1}{2N}\right) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100\% \quad \text{..................} \quad (1)$$

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:

$$M = (0.5 - 1/(2 \times 16)) \times 100\% = 46.875\% \quad \text{...} \quad (2)$$

This is a theoretical value. A reasonable margin to allow in system designs is 20% to 30%.

Notes on Use of SCIF2 in Asynchronous Mode: The TxD2 pin goes to the high-impedance state when the TE bit in SCSCR2 is cleared to 0. There are three methods of driving the TxD2 pin high during a transmit operation, as follows.

1. Pull the TxD2 pin up externally.
2. Set the TE bit to 1 in the initialization procedure, and do not clear it to 0 subsequently. At the start of the transmit operation, write the transmit data to SCFTDR2 and clear the TDFE flag.
3. Set the TxD2 pin as an output port and select high drive (by setting SCPCR[9:8] to B'01 and SCPDR[4] to B'1). As a result, the TxD2 pin will be driven high when TE = 0, and will function as a serial pin when TE = 1.

Note on Simultaneous Transmission/Reception in Synchronous Mode: Transmission is performed until the FIFO is empty, while reception is performed until the FIFO is full. The operation is also the same in asynchronous mode.

When performing simultaneous transmission/reception in synchronous mode, SCK2 output matches the transmit data quantity (8 pulses for 1 byte). In this case, reception is continuous until the FIFO is full. The valid number of receive data bytes is the same as the number of transmit data bytes.

After transmission/reception ends, transmission/reception is restarted by clearing the TE and RE bits to 0, writing transmit data, reading receive data, and resetting the transmit/receive FIFO, and then setting the TE and RE bits simultaneously and starting transmit/receive operation.

For details of the setting procedure, see figure 18.14, 18.15, Sample SCF2 Transmission/Reception Flowchart.

Note on Serial Mode Register (SCSMR2) Setting: Bit 7 (C/A) in the serial mode register (SCSMR2) should be written to in the initialization procedure after a power-on reset, and its value should not subsequently be changed.

Section 19 USB Function Module

19.1 Features

- Incorporates UDC (USB device controller) conforming to USB1.0

  Automatic processing of USB protocol

  Automatic processing of USB standard commands for endpoint 0 (some commands and class/vendor commands require decoding and processing by firmware)
- Transfer speed: Full-speed
- Endpoint configuration

Endpoint Name	Abbreviation	Transfer Type	Maximum Packet Size	FIFO Buffer Capacity (Byte)	DMA Transfer
Endpoint 0	EP0s	Setup	8	8	—
	EP0i	Control-in	8	8	—
	EP0o	Control-out	8	8	—
Endpoint 1	EP1	Bulk-out	64	128	Possible
Endpoint 2	EP2	Bulk-in	64	128	Possible
Endpoint 3	EP3	Interrupt	8	8	—

```
Configuration 1—Interface 0—Alternate setting 0 ─┬─ Endpoint 1
                                                  ├─ Endpoint 2
                                                  └─ Endpoint 3
```

- Interrupt requests: Generates various interrupt signals necessary for USB transmission/reception.
- Clock: Selection of internal system clock/external input (48 MHz) by means of EXCPG
- Power-down mode

  Power consumption can be reduced by stopping UDC internal clock when USB cable is disconnected.

  Automatic transition to/recovery from suspend state
- Can be connected to a Philips PDIUSBP11 Series transceiver or compatible product (when using a compatible product, carry out evaluation and investigation with the manufacturer supplying the transceiver beforehand).
- Power mode: Self-powered

19.2 Block Diagram

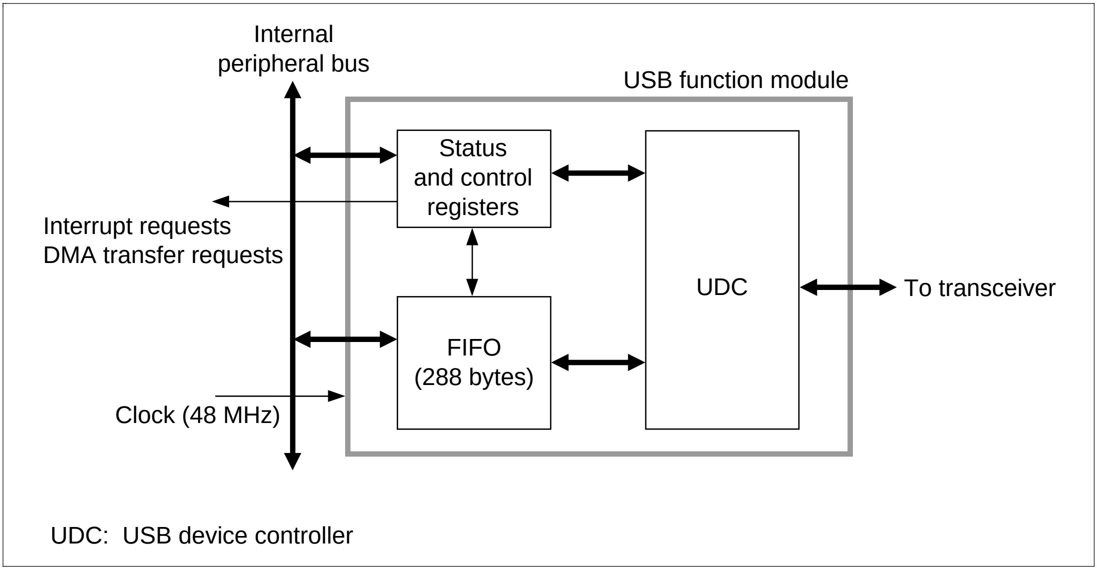


Figure 19.1 Block Diagram of USB

19.3 Pin Configuration

Table 19.1 Pin Configuration and Functions

Pin Name	I/O	Function
XVDATA	Input	Input pin for receive data from differential receiver
DPLS	Input	Input pin to driver for D+ signal from receiver
DMNS	Input	Input pin to driver for D– signal from receiver
TXDPLS	Output	D+ transmit output pin to driver
TXDMNS	Output	D– transmit output pin to driver
TXENL	Output	Driver output enable pin
VBUS	Input	USB cable connection monitor pin
SUSPND	Output	Transceiver suspend state output pin
UCLK	Input	USB clock input pin (48 MHz input)

Can be connected to a Philips PDIUSBP11 Series transceiver or compatible product (when using a compatible product, carry out evaluation and investigation with the manufacturer supplying the transceiver beforehand).

19.4 Register Configuration

Table 19.2 USB Function Module Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
USBEP0i data register	USBEPDR0I	W	—	H'A4008000	8
USBEP0o data register	USBEPDR0O	R	—	H'A4008004	8
USBEP0s data register	USBEPDR0S	R	—	H'A4008008	8
USBEP1 data register	USBEPDR1	W	—	H'A400800C	8/32*
USBEP2 data register	USBEPDR2	W	—	H'A4008010	8/32*
USBEP3 data register	USBEPDR3	W	—	H'A4008014	8
Interrupt flag register 0	USBIFR0	R/W	H'10	H'A4008018	8
Interrupt flag register 1	USBIFR1	R/W	H'00	H'A400801A	8
Trigger register	USBTRG	W	—	H'A400801C	8
FIFO clear register	USBFCLR	W	—	H'A400801E	8
USBEP0o receive data size register	USBEPSZ0O	R	H'00	H'A4008020	8
Data status register	USBDASTS	R	H'00	H'A4008022	8
Endpoint stall register	USBEPSTL	R/W	H'00	H'A4008024	8
Interrupt enable register 0	USBIER0	R/W	H'00	H'A4008026	8
Interrupt enable register 1	USBIER1	R/W	H'00	H'A4008028	8
USBEP1 receive data size register	USBEPSZ1	R	H'00	H'A400802A	8
USBDMA setting register	USBDMAR	R/W	H'00	H'A400802C	8
Interrupt select register 0	USBISR0	R/W	H'00	H'A400802E	8
Interrupt select register 1	USBISR1	R/W	H'07	H'A4008030	8

Note: * The USBEP1 data register and USBEP2 data register can be accessed in longword units. For the access method using longword units, see section 19.10, Usage Notes 1.

19.5 Register Descriptions

19.5.1 USBEP0i Data Register (USBEPDR0I)

USBEPDR0I is an 8-byte FIFO buffer for endpoint 0, holding 1 packet of transmit data for control-in. Transmit data is fixed by writing 1 packet of data and setting bit 0 in the USB trigger register. When an ACK handshake is returned from the host after the data has been transmitted, bit 0 in USB interrupt flag register 0 is set. This FIFO buffer can be initialized by means of bit 0 in the USBFIFO clear register.

19.5.2 USBEP0o Data Register (USBEPDR0O)

USBEPDR0O is an 8-byte receive FIFO buffer for endpoint 0. USBEPDR0O holds endpoint 0 receive data other than setup commands. When data is received normally, bit 2 in USB interrupt flag register 0 is set, and the number of receive bytes is indicated in the EP0o receive data size register. After the data has been read, setting bit 1 in the USB trigger register enables the next packet to be received. This FIFO buffer can be initialized by means of bit 1 in the USBFIFO clear register.

19.5.3 USBEP0s Data Register (USBEPDR0S)

USBEPDR0S is an 8-byte FIFO buffer specifically for endpoint 0 setup command reception. USBEPDR0S receives only setup commands requiring processing on the application side. When command data is received normally, bit 3 in USB interrupt flag register 0 is set. As a setup command must be received without fail, if data is left in this buffer, it will be overwritten with new data. If reception of the next command is started while the current command is being read, command reception has priority, the read by the application is forcibly terminated, and the read data is invalid.

19.5.4 USBEP1 Data Register (USBEPDR1)

USBEPDR1 is a 128-byte receive FIFO buffer for endpoint 1. USBEPDR1 has a dual-FIFO configuration, and has a capacity of twice the maximum packet size. When 1 packet of data is received normally from the host, bit 6 in USB interrupt flag register 0 is set. The number of receive bytes is indicated in the USBEP1 receive data size register. After the data has been read, the buffer that was read is enabled to receive again by writing 1 to bit 5 in the USB trigger register. The receive data in this FIFO buffer can be transferred by DMA (see section 19.5.19, USBDMA Setting Register (USBDMAR)). This FIFO buffer can be initialized by means of bit 1 in the USBFIFO clear register.

19.5.5 USBEP2 Data Register (USBEPDR2)

USBEPDR2 is a 128-byte transmit FIFO buffer for endpoint 2. USBEPDR2 has a dual-buffer configuration, and has a capacity of twice the maximum packet size. When transmit data is written to this FIFO buffer and bit 4 in the USB trigger register is set, 1 packet of transmit data is fixed, and the dual-FIFO buffer is switched over. Transmit data for this FIFO buffer can be transferred by DMA (see section 19.5.19, USBDMA Setting Register (USBDMAR)). This FIFO buffer can be initialized by means of bit 4 in the USBFIFO clear register.

19.5.6 USBEP3 Data Register (USBEPDR3)

USBEPDR3 is an 8-byte transmit FIFO buffer for endpoint 3, holding 1 packet of transmit data in endpoint 3 interrupt transfer. Transmit data is fixed by writing 1 packet of data and setting bit 6 in the USB trigger register. When an ACK handshake is received from the host after 1 packet of data has been transmitted normally, bit 1 in the USB interrupt flag register is set. This FIFO buffer can be initialized by means of bit 6 in the USBFCLR register.

19.5.7 USB Interrupt Flag Register 0 (USBIFR0)

Together with USB interrupt flag register 1, USBIFR0 indicates interrupt status information required by the application. When an interrupt source occurs, the corresponding bit is set to 1 and an interrupt request is sent to the CPU according to the combination with USB interrupt enable register 0. Clearing is performed by writing 0 to the bit to be cleared, and 1 to the other bits. However, bits 6 and 4 are status bits, and cannot be cleared.

Bit:	7	6	5	4	3	2	1	0
	BRST	EP1 FULL	EP2 TR	EP2 EMPTY	SETUP TS	EP0o TS	EP0i TR	EP0i TS
Initial value:	0	0	0	1	0	0	0	0
R/W:	R/W	R	R/W	R	R/W	R/W	R/W	R/W

Bit 7—Bus Reset (BRST): Set to 1 when the bus reset signal is detected on the USB bus.

Bit 6—EP1 FIFO Full (EP1 FULL): This bit is set when endpoint 1 receives 1 packet of data normally from the host, and holds a value of 1 as long as there is valid data in the FIFO buffer. EP1 FULL is a status bit, and cannot be cleared.

Bit 5—EP2 Transfer Request (EP2 TR): This bit is set if there is no valid transmit data in the FIFO buffer when an IN token for endpoint 2 is received from the host. A NACK handshake is returned to the host until data is written to the FIFO buffer and packet transmission is enabled.

Bit 4—EP2 FIFO Empty (EP2 EMPTY): This bit is set when at least one of the dual endpoint 2 transmit FIFO buffers is ready for transmit data to be written. EP2 EMPTY is a status bit, and cannot be cleared.

Bit 3—Setup Command Receive Complete (SETUP TS): This bit is set to 1 when endpoint 0 receives normally a setup command requiring decoding on the application side, and returns an ACK handshake to the host.

Bit 2—EP0o Receive Complete (EP0o TS): This bit is set to 1 when endpoint 0 receives data from the host normally, stores the data in the FIFO buffer, and returns an ACK handshake to the host.

Bit 1—EP0i Transfer Request (EP0i TR): This bit is set if there is no valid transmit data in the FIFO buffer when an IN token for endpoint 0 is received from the host. A NACK handshake is returned to the host until data is written to the FIFO buffer and packet transmission is enabled.

Bit 0—EP0i Transmit Complete (EP0i TS): This bit is set when data is transmitted to the host from endpoint 0 and an ACK handshake is returned.

19.5.8 USB Interrupt Flag Register 1 (USBIFR1)

Together with USB interrupt flag register 0, USBIFR1 indicates interrupt status information required by the application. When an interrupt source occurs, the corresponding bit is set to 1 and an interrupt request is sent to the CPU according to the combination with USB interrupt enable register 1. Clearing is performed by writing 0 to the bit to be cleared, and 1 to the other bits.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	EP3 TR	EP3 TS	BVUS
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

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

Bit 2—EP3 Transfer Request (EP3 TR): This bit is set if there is no valid transmit data in the FIFO buffer when an IN token for endpoint 3 is received from the host. A NACK handshake is returned to the host until data is written to the FIFO buffer and packet transmission is enabled.

Bit 1—EP3 Transmit Complete (EP3 TS): This bit is set when data is transmitted to the host from endpoint 3 and an ACK handshake is returned.

Bit 0—USB Bus Connect (VBUS): This bit is set by a rising edge at the VBUS pin. By connecting the VBUS monitor signal to the VBUS pin, an interrupt request can be sent to the CPU when power is supplied to the VBUS.

The VBUS pin must be connected, as it is needed inside the module.

19.5.9 USB Trigger Register (USBTRG)

USBTRG generates one-shot triggers to control the transmit/receive sequence for each endpoint.

Bit:	7	6	5	4	3	2	1	0
	—	EP3 PKTE	EP1 RDFN	EP2 PKTE	—	PE0s RDFN	EP0o RDFN	EP0i PKTE
R/W:	W	W	W	W	W	W	W	W

Bit 7—Reserved

Bit 6—EP3 Packet Enable (EP3 PKTE): After one packet of data has been written to the endpoint 3 transmit FIFO buffer, the transmit data is fixed by writing 1 to this bit.

Bit 5—EP1 Read Complete (EP1 RDFN): Write 1 to this bit after one packet of data has been read from the endpoint 1 FIFO buffer. The endpoint 1 receive FIFO buffer has a dual-FIFO configuration. Writing 1 to this bit initializes the FIFO that was read, enabling the next packet to be received.

Bit 4—EP2 Packet Enable (EP2 PKTE): After one packet of data has been written to the endpoint 2 FIFO buffer, the transmit data is fixed by writing 1 to this bit.

Bit 3—Reserved

Bit 2—EP0s Read Complete (EP0s RDFN): Write 1 to this bit after EP0s command FIFO data has been read. Writing 1 to this bit enables transmission/reception of data in the following data stage. A NACK handshake is returned in response to transmit/receive requests from the host in the data stage until 1 is written to this bit.

Bit 1—EP0o Read Complete (EP0o RDFN): Writing 1 to this bit after one packet of data has been read from the endpoint 0 transmit FIFO buffer initializes the FIFO buffer, enabling the next packet to be received.

Bit 0—EP0i Packet Enable (EP0i PKTE): After one packet of data has been written to the endpoint 0 transmit FIFO buffer, the transmit data is fixed by writing 1 to this bit.

19.5.10 USBFIFO Clear Register (USBFCLR)

USBFCLR is provided to initialize the FIFO buffers for each endpoint. Writing 1 to a bit clears all the data in the corresponding FIFO buffer. The corresponding interrupt flag is not cleared. Do not clear a FIFO buffer during transmission/reception.

Bit:	7	6	5	4	3	2	1	0
	—	EP3 CLR	EP1 CLR	EP2 CLR	—	—	EP0o CLR	EP0i CLR
R/W:	W	W	W	W	W	W	W	W

Bit 7—Reserved

Bit 6—EP3 Clear (EP3 CLR): When 1 is written to this bit, the endpoint 3 transmit FIFO buffer is initialized.

Bit 5—EP1 Clear (EP1 CLR): When 1 is written to this bit, both FIFOs in the endpoint 1 receive FIFO buffer are initialized.

Bit 4—EP2 Clear (EP2 CLR): When 1 is written to this bit, both FIFOs in the endpoint 2 transmit FIFO buffer are initialized.

Bits 3 and 2—Reserved

Bit 1—EP0o Clear (EP0o CLR): When 1 is written to this bit, the endpoint 0 receive FIFO buffer is initialized.

Bit 0—EP0i Clear (EP0i CLR): When 1 is written to this bit, the endpoint 0 transmit FIFO buffer is initialized.

19.5.11 USBEP0o Receive Data Size Register (USBEPSZ0O)

USBEPSZ0O indicates, in bytes, the amount of data received from the host by endpoint 0.

19.5.12 USB Data Status Register (USBDASTS)

USBDASTS indicates whether the transmit FIFO buffers contain valid data. A bit is set when data is written to the corresponding FIFO buffer and the packet enable state is set, and cleared when all data has been transmitted to the host.

Bit:	7	6	5	4	3	2	1	0
	—	—	EP3 DE	EP2 DE	—	—	—	EP0i DE
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Bits 7 and 6—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 5—EP3 Data Present (EP3 DE): This bit is set when the endpoint 3 FIFO buffer contains valid data.

Bit 4—EP2 Data Present (EP2 DE): This bit is set when the endpoint 2 FIFO buffer contains valid data.

Bits 3 to 1—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 0—EP0i Data Present (EP0i DE): This bit is set when the endpoint 0 FIFO buffer contains valid data.

19.5.13 USB Endpoint Stall Register (USBEPSTL)

The bits in USBEPSTL are used to forcibly stall the endpoints on the application side. While a bit is set to 1, the corresponding endpoint returns a stall handshake to the host. The stall bit for endpoint 0 (EP0 STL) is cleared automatically on reception of 8-bit command data for which decoding is performed by the function. When the SETUPTS flag in IFR0 is set, a write of 1 to the EP0 STL bit is ignored. For details see section 19.8, Stall Operations.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	EP3 STL	EP2 STL	EP1 STL	EP0 STL
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R/W	R/W	R/W	R/W

Bits 7 to 4—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 3—EP3 Stall (EP3 STL): When this bit is set to 1, endpoint 3 is placed in the stall state.

Bit 2—EP2 Stall (EP2 STL): When this bit is set to 1, endpoint 2 is placed in the stall state.

Bit 1—EP1 Stall (EP1 STL): When this bit is set to 1, endpoint 1 is placed in the stall state.

Bit 0—EP0 Stall (EP0 STL): When this bit is set to 1, endpoint 0 is placed in the stall state.

19.5.14 USB Interrupt Enable Register 0 (USBIER0)

USBIER0 enables the interrupt requests indicated in interrupt flag register 0 (USBIFR0). When an interrupt flag is set while the corresponding bit in USBIER0 is set to 1, an interrupt request is sent to the CPU. The interrupt vector number is determined by the contents of interrupt select register 0 (USBISR0).

Bit:	7	6	5	4	3	2	1	0
	BRST	EP1 FULL	EP2 TR	EP2 TMPTY	SETUP TS	EP0o TS	EP0i TS	EP0i TS
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

19.5.15 USB Interrupt Enable Register 1 (USBIER1)

USBIER1 enables the interrupt requests indicated in interrupt flag register 1 (USBIFR1). When an interrupt flag is set while the corresponding bit in USBIER1 is set to 1, an interrupt request is sent to the CPU. The interrupt vector number is determined by the contents of interrupt select register 1 (USBISR1).

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	EP3 TR	EP3 TS	VBUS
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W

19.5.16 USBEP1 Receive Data Size Register (USBEPSZ1)

USBEPSZ1 is the endpoint 1 receive data size register, indicating the amount of data received from the host. The endpoint 1 FIFO buffer has a dual-FIFO configuration; the receive data size indicated by this register refers to the currently selected FIFO.

19.5.17 USB Interrupt Select Register 0 (USBISR0)

USBISR0 selects the vector numbers of the interrupt requests indicated in interrupt flag register 0. If the USB issues an interrupt request to the INTC when the corresponding bit in USBISR0 is cleared to 0, the interrupt will be USI1 (USB interrupt 1), with an interrupt vector number of 160. If the USB issues an interrupt request to the INTC when the corresponding bit in USBISR0 is set to 1, the interrupt will be USI2 (USB interrupt 2), with an interrupt vector number of 161. The initial value designates vector number 160. If interrupts occur simultaneously, USI0 has priority by default. Bits must be assigned so as to prevent endpoint 0 related interrupt requests.

Bit:	7	6	5	4	3	2	1	0
	BSRT	EP1 FULL	EP2 TR	EP2 EMPTY	SETUP TS	EP0o TS	EP0i TR	EP0i TS
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

19.5.18 USB Interrupt Select Register 1 (USBISR1)

USBISR1 selects the vector numbers of the interrupt requests indicated in interrupt flag register 1. If the USB issues an interrupt request to the INTC when the corresponding bit in USBISR1 is cleared to 0, the interrupt will be USI1 (USB interrupt 1), with an interrupt vector number of 160. If the USB issues an interrupt request to the INTC when the corresponding bit in USBISR1 is set to 1, the interrupt will be USI2 (USB interrupt 2), with an interrupt vector number of 161. The initial value designates vector number 161. If interrupts occur simultaneously, USI0 has priority by default. Bits must be assigned so as to prevent endpoint 0 related interrupt requests.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	EP3 TR	EP3 TS	VBUS
Initial value:	0	0	0	0	0	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W

19.5.19 USBDMA Setting Register (USBDMAR)

DMA transfer can be carried out between the endpoint 1 and endpoint 2 data registers by means of the on-chip DMA controller. Dual address transfer is performed, using byte transfer units. In order to start DMA transfer, DMA control settings must be made in addition to the settings in this register.

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	EP2 DMAE	EP1 DMAE
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

Bits 7 to 2—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 1—Endpoint 2 DMA Transfer Enable (EP2 DMAE): When this bit is set, DMA transfer is enabled from memory to the endpoint 2 transmit FIFO buffer. If there is at least one byte of space in the FIFO buffer, a transfer request is asserted for the DMA controller. In DMA transfer, when 64 bytes are written to the FIFO buffer the EP2 packet enable bit is set automatically, allowing 64 bytes of data to be transferred, and if there is still space in the other of the two FIFOs, a transfer request is asserted for the DMA controller again. However, if the size of the data packet to be transmitted is less than 64 bytes, the EP2 packet enable bit is not set automatically, and so should be set by the CPU with a DMA transfer end interrupt.

Use a 1-packet unit as the DMA controller transfer unit. The DMA controller transfer count must therefore be set to 64 bytes or less.

Also, as EP2-related interrupt requests to the CPU are not automatically masked, interrupt requests should be masked as necessary in the interrupt enable register.

Operating procedure 1: Continuous transfer of maximum packet size (64 bytes)

1. Write of 1 to the USBDMAR/EP2 DMAE bit
2. Transfer count setting for the maximum packet size (64 bytes) in the DMA controller
3. DMA controller activation
4. DMA transfer (transfer of 64 bytes)
5. DMA control re-activation by the DMA transfer end interrupt

Steps 3 to 5 are subsequently repeated.

Operating procedure 2: Data size smaller than maximum packet size (64 bytes)

1. Write of 1 to the USBDMAR/EP2 DMAE bit
2. Transfer count setting for less than the maximum packet size (64 bytes) in the DMA controller
3. DMA controller activation
4. DMA transfer (transfer of fewer than 64 bytes)
5. Write of 1 to the USBTRG/EP2 PKTE bit by the DMA transfer end interrupt

Bit 0—Endpoint 1 DMA Transfer Enable (EP1 DMAE): When this bit is set, DMA transfer is enabled from the endpoint 1 receive FIFO buffer to memory. If there is at least one byte of receive data in the FIFO buffer, a transfer request is asserted for the DMA controller. In DMA transfer, when all the received data is read, EP1 is read automatically and the completion trigger operates.

A one-packet unit must be used as the DMA controller transfer unit. Therefore, use the EP1 FIFO full interrupt to check the size received from the host (USBEPSZ1), and set that received size (up to 64 bytes) as the DMA controller transfer count.

EP1-related interrupt requests to the CPU are not automatically masked.

Operating procedure:

1. Write of 1 to the USBDMAR/EP1 DMAE bit
2. EP1 FIFO full interrupt reception
3. USBEP1 receive size register (USBEPSZ1) read
4. Transfer count setting for the USBEP1 receive size register size (up to 64 bytes) in the DMA controller
5. DMA controller activation
6. DMA transfer (transfer of up to 64 bytes)
7. After the DMA transfer end interrupt, wait for the next EP1 FIFO full interrupt

Steps 2 to 7 are subsequently repeated.

19.6 Operation

19.6.1 Cable Connection

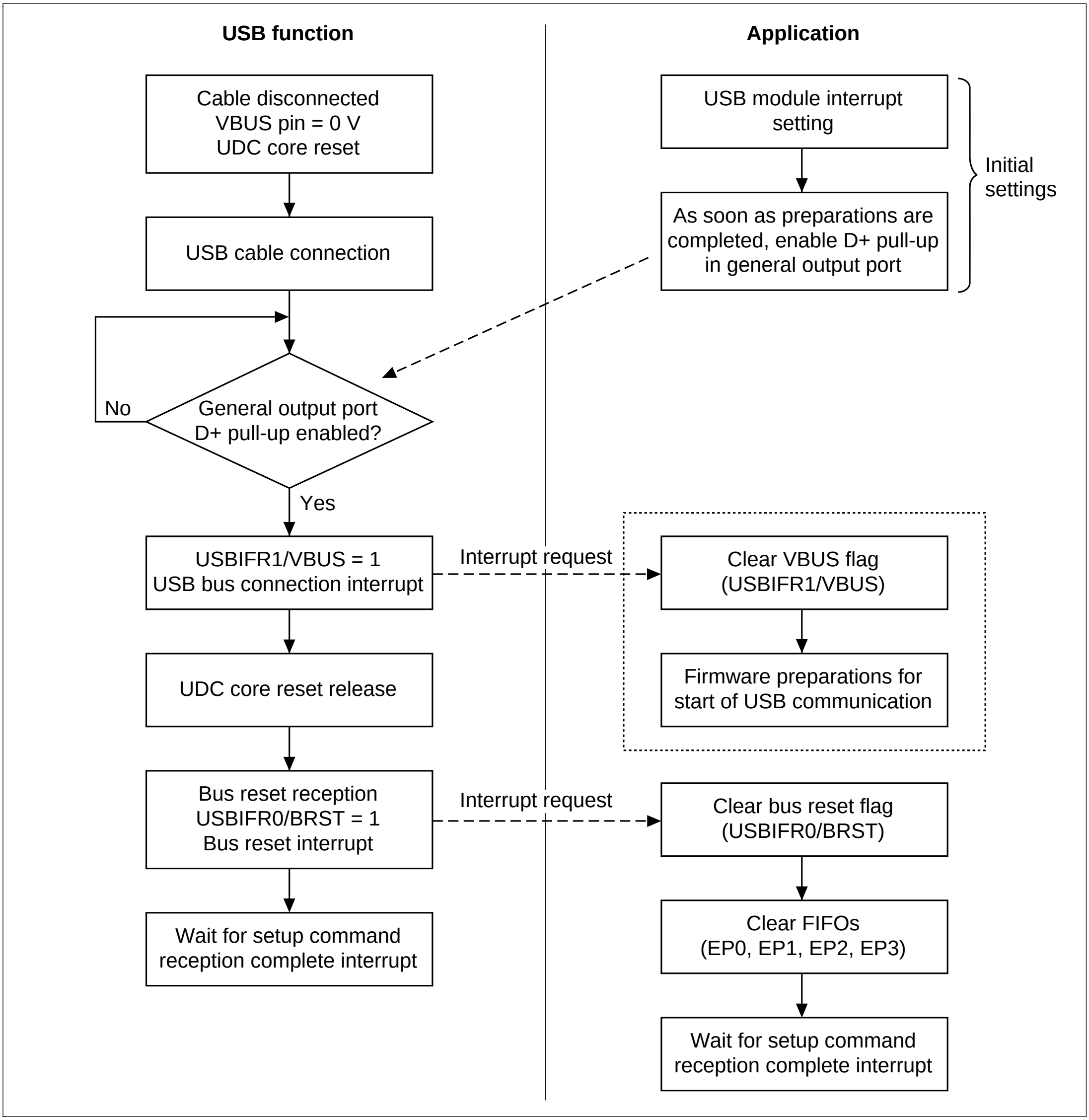


Figure 19.2 Cable Connection Operation

The above flowchart shows the operation in the case of figure 19.15 in section 19.9, Example of USB External Circuitry (*1 and *2 not used).

In applications that do not require USB cable connection to be detected, processing by the USB bus connection interrupt is not necessary. Preparations should be made with the bus reset interrupt.

Also, in applications that require connection detection regardless of D+ pull-up control, detection should be carried out using IRQx or a general input port. For details, see section 19.9, Example of USB External Circuitry.

19.6.2 Cable Disconnection

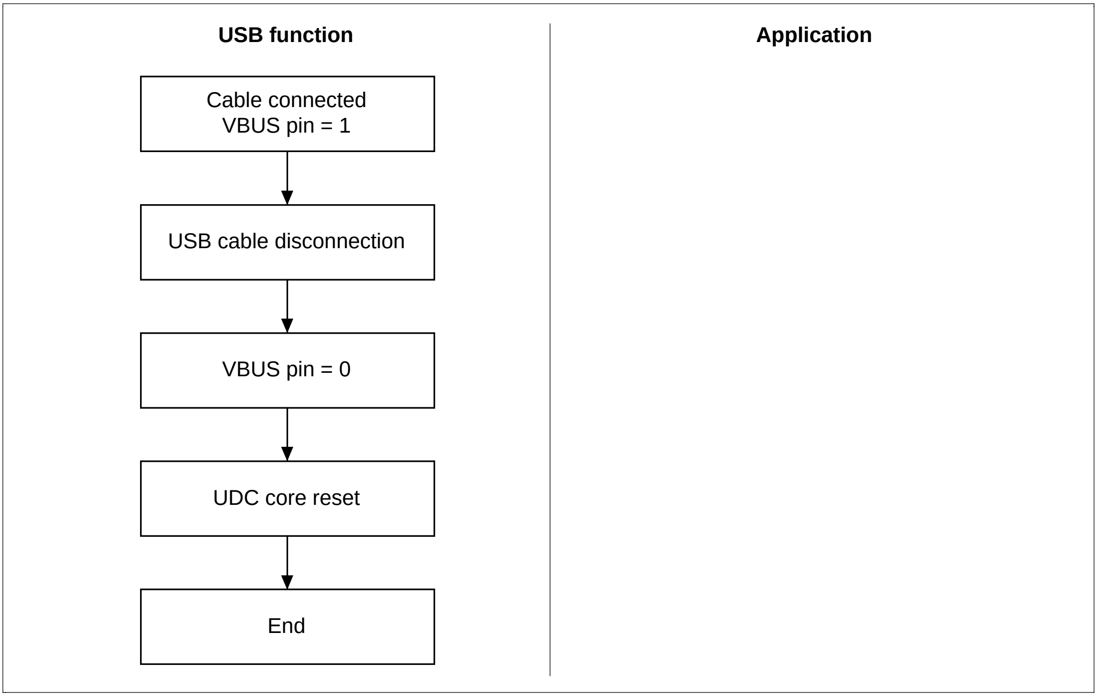


Figure 19.3 Cable Disconnection Operation

The above flowchart shows the operation in the case of figure 19.15, Example of USB External Circuitry, in section 19.9, Example of USB External Circuitry (*1 and *2 not used).

As the USB bus connection interrupt is detected at the rising edge, disconnection detection is not possible by means of this interrupt (Also, a USB bus connection interrupt may be generated if chattering occurs during disconnection). Therefore, in applications that require disconnection to be detected, or applications that require connection/disconnection detection regardless of D+ pull-up control, detection should be carried out using IRQx or a general input port. For details, see section 19.9, Example of USB External Circuitry.

19.6.3 Control Transfer

Control transfer consists of three stages: setup, data (not always included), and status (figure 19.4). The data stage comprises a number of bus transactions. Operation flowcharts for each stage are shown below.

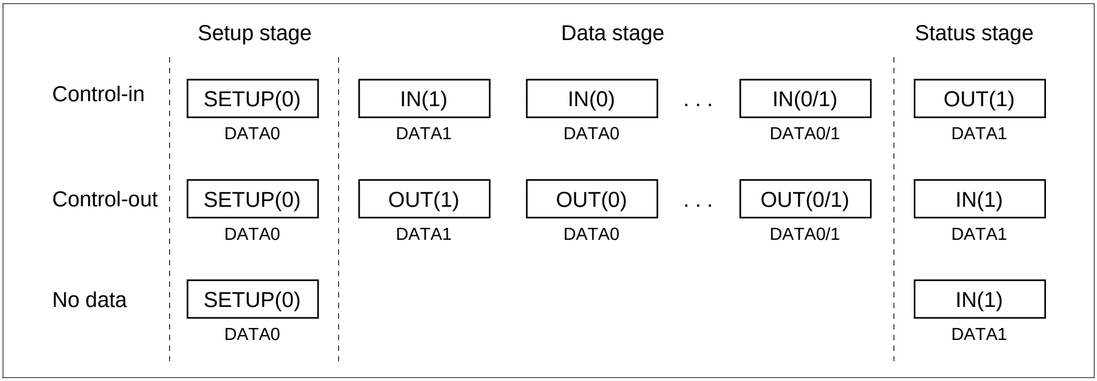
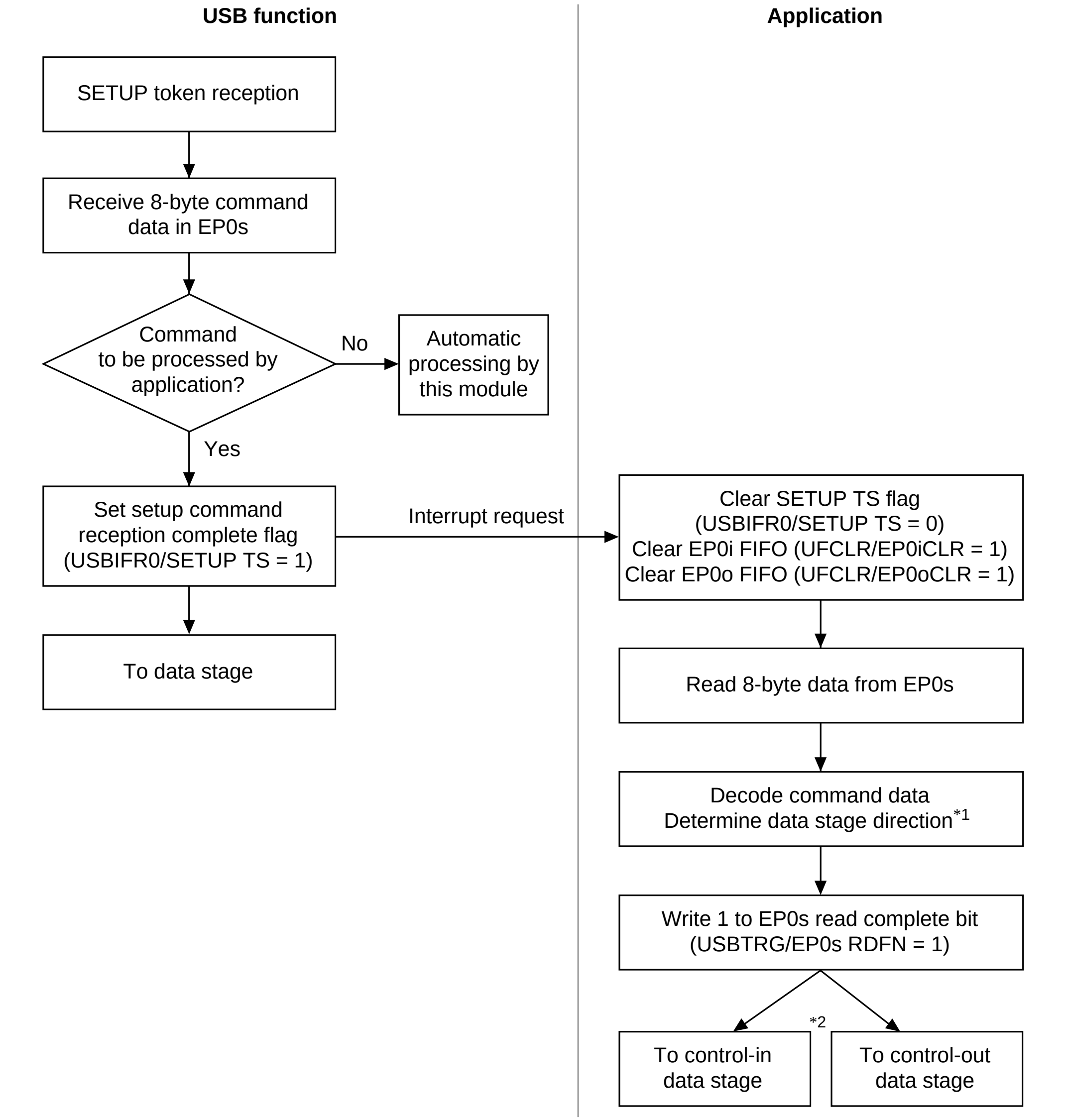


Figure 19.4 Transfer Stages in Control Transfer

Setup Stage

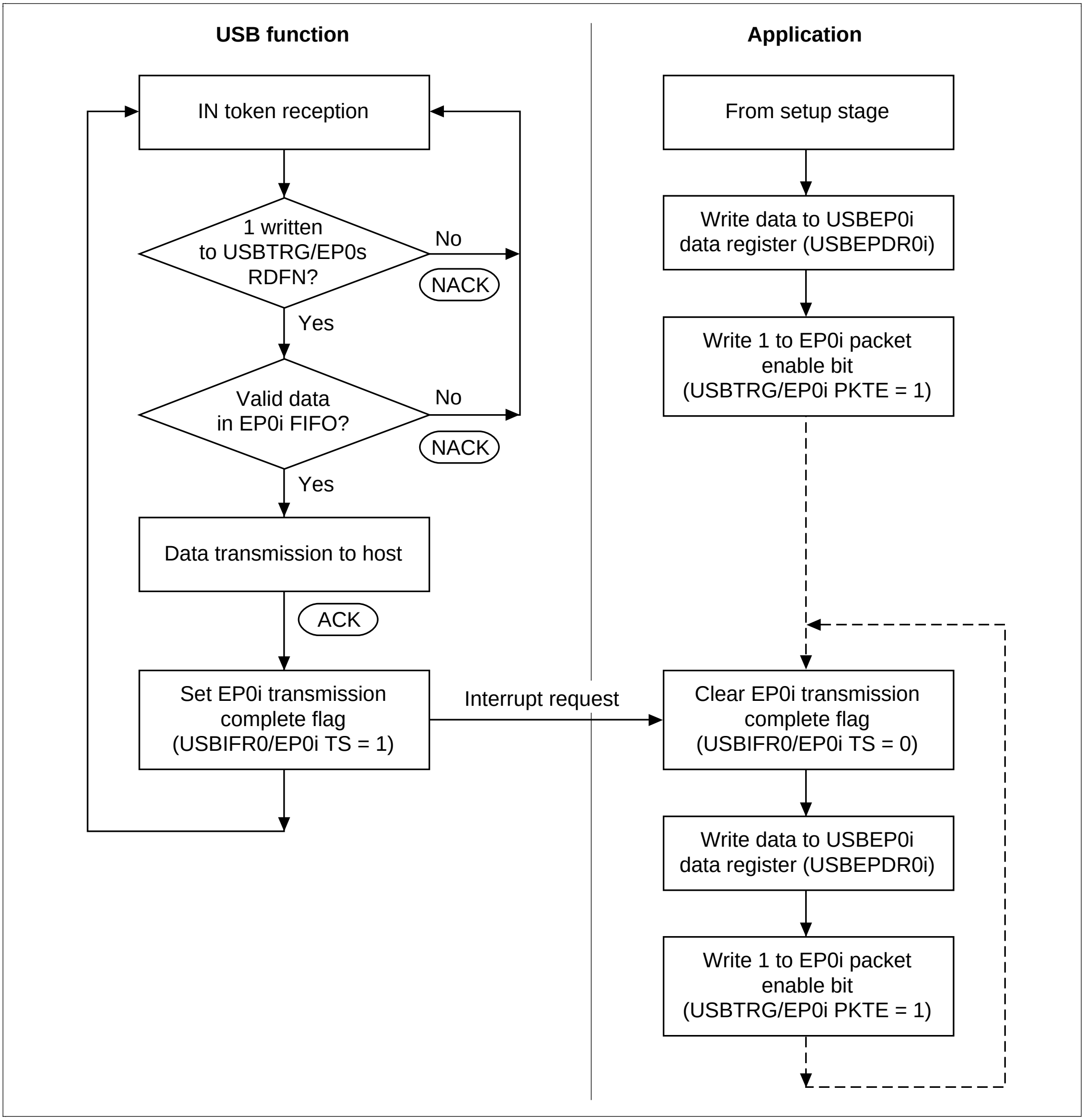

USB function
Application
SETUP token reception
Receive 8-byte command data in EP0s
Command to be processed by application?
No
Automatic processing by this module
Yes
Set setup command reception complete flag (USBIFR0/SETUP TS = 1)
Interrupt request
To data stage
Clear SETUP TS flag (USBIFR0/SETUP TS = 0)
Clear EP0i FIFO (UFCLR/EP0iCLR = 1)
Clear EP0o FIFO (UFCLR/EP0oCLR = 1)
Read 8-byte data from EP0s
Decode command data
Determine data stage direction*1
Write 1 to EP0s read complete bit (USBTRG/EP0s RDFN = 1)
*2
To control-in data stage
To control-out data stage

Notes: *1 In the setup stage, the application analyzes command data from the host requiring processing by the application, and determines the subsequent processing (for example, data stage direction, etc.).

*2 When the transfer direction is control-out, the EP0i transfer request interrupt required in the status stage should be enabled here. When the transfer direction is control-in, this interrupt is not required and should be disabled.

Figure 19.5 Setup Stage Operation

Data Stage (Control-In)



USB function
Application
IN token reception
1 written to USBTRG/EP0s RDFN?
No
NACK
Yes
Valid data in EP0i FIFO?
No
NACK
Yes
Data transmission to host
ACK
Set EP0i transmission complete flag (USBIFR0/EP0i TS = 1)
Interrupt request
From setup stage
Write data to USBEP0i data register (USBEPDR0i)
Write 1 to EP0i packet enable bit (USBTRG/EP0i PKTE = 1)
Clear EP0i transmission complete flag (USBIFR0/EP0i TS = 0)
Write data to USBEP0i data register (USBEPDR0i)
Write 1 to EP0i packet enable bit (USBTRG/EP0i PKTE = 1)

Figure 19.6 Data Stage (Control-In) Operation

The application first analyzes command data from the host in the setup stage, and determines the subsequent data stage direction. If the result of command data analysis is that the data stage is in-transfer, 1 packet of data to be sent to the host is written to the FIFO. If there is more data to be sent, this data is written to the FIFO after the data written first has been sent to the host (USBIFR0/EP0i TS = 1).

The end of the data stage is identified when the host transmits an OUT token and the status stage is entered.

Note: If the size of the data transmitted by the function is smaller than the data size requested by the host, the function indicates the end of the data stage by returning to the host a packet shorter than the maximum packet size. If the size of the data transmitted by the function is an integral multiple of the maximum packet size, the function indicates the end of the data stage by transmitting a 0-length packet.

Data Stage (Control-Out)

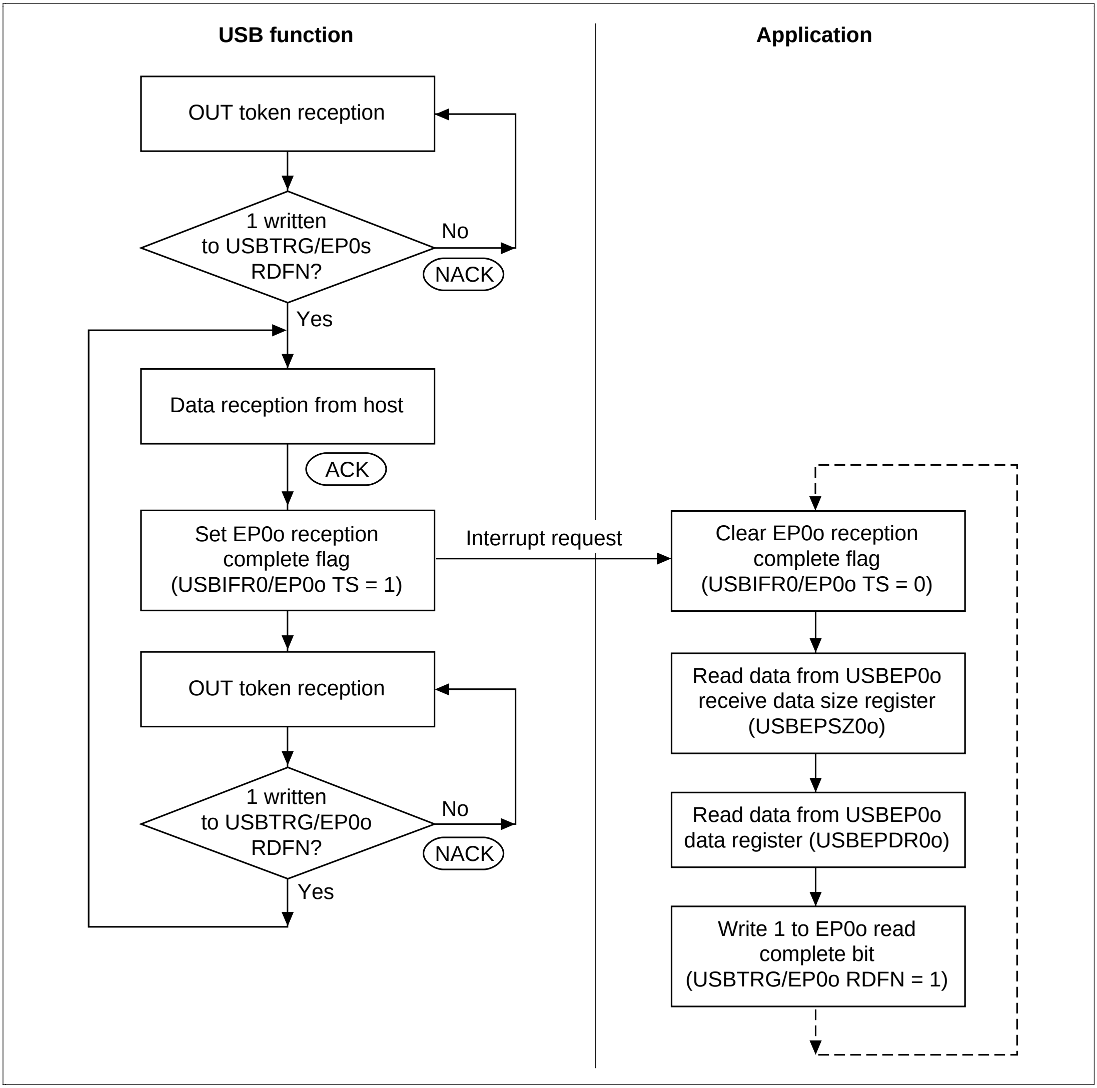

USB function
Application
OUT token reception
1 written to USBTRG/EP0s RDFN?
No
NACK
Yes
Data reception from host
ACK
Set EP0o reception complete flag (USBIFR0/EP0o TS = 1)
Interrupt request
OUT token reception
1 written to USBTRG/EP0o RDFN?
No
NACK
Yes
Clear EP0o reception complete flag (USBIFR0/EP0o TS = 0)
Read data from USBEP0o receive data size register (USBEPSZ0o)
Read data from USBEP0o data register (USBEPDR0o)
Write 1 to EP0o read complete bit (USBTRG/EP0o RDFN = 1)

Figure 19.7 Data Stage (Control-Out) Operation

The application first analyzes command data from the host in the setup stage, and determines the subsequent data stage direction. If the result of command data analysis is that the data stage is out-transfer, the application waits for data from the host, and after data is received (USBIFR0/EP0o TS = 1), reads data from the FIFO. Next, the application writes 1 to the EP0o read complete bit, empties the receive FIFO, and waits for reception of the next data.

The end of the data stage is identified when the host transmits an IN token and the status stage is entered.

Status Stage (Control-In)

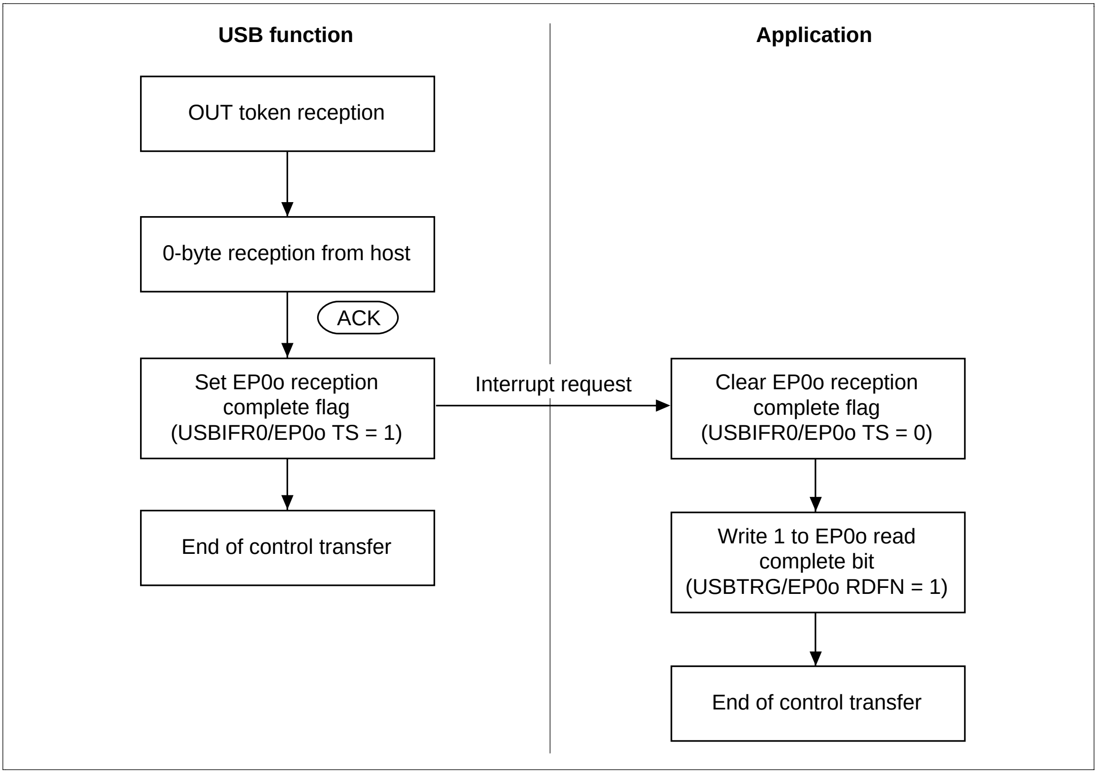

USB function
Application
OUT token reception
0-byte reception from host
ACK
Set EP0o reception complete flag (USBIFR0/EP0o TS = 1)
Interrupt request
Clear EP0o reception complete flag (USBIFR0/EP0o TS = 0)
End of control transfer
Write 1 to EP0o read complete bit (USBTRG/EP0o RDFN = 1)
End of control transfer

Figure 19.8 Status Stage (Control-In) Operation

The control-in status stage starts with an OUT token from the host. The application receives 0-byte data from the host, and ends control transfer.

Status Stage (Control-Out)

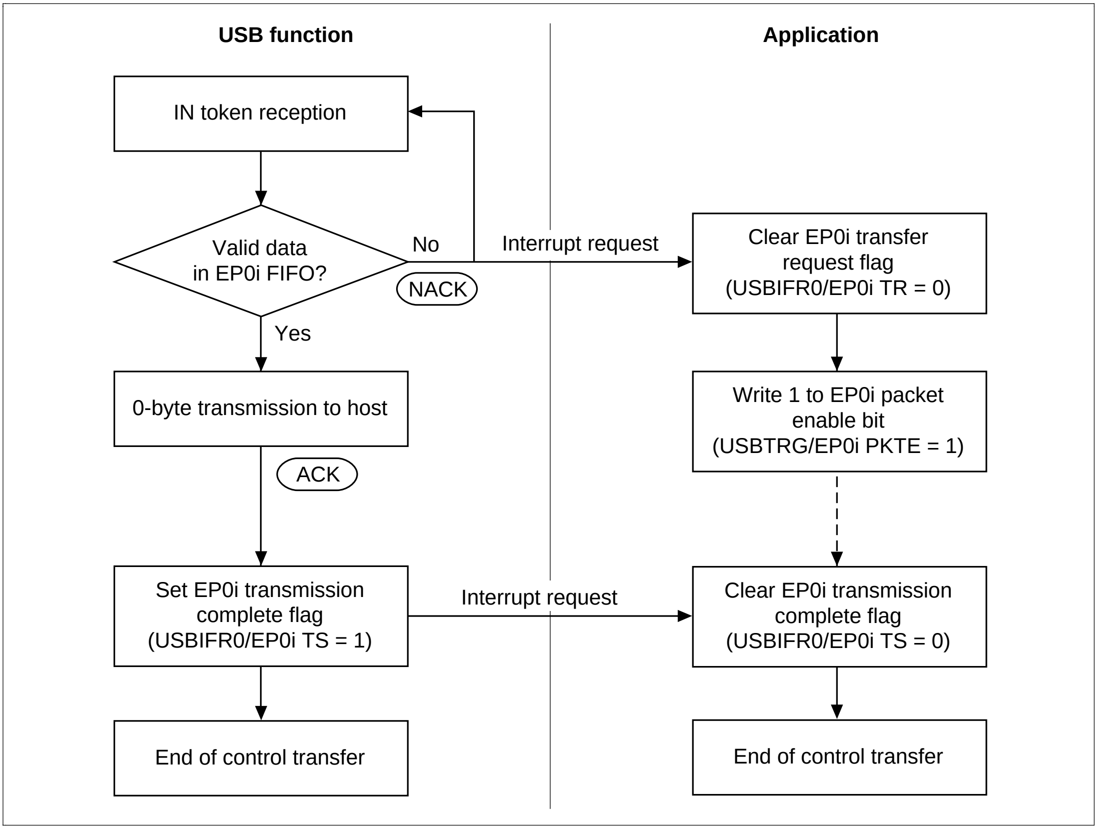

USB function
Application
IN token reception
Valid data in EP0i FIFO?
No
NACK
Interrupt request
Clear EP0i transfer request flag (USBIFR0/EP0i TR = 0)
Yes
0-byte transmission to host
Write 1 to EP0i packet enable bit (USBTRG/EP0i PKTE = 1)
ACK
Set EP0i transmission complete flag (USBIFR0/EP0i TS = 1)
Interrupt request
Clear EP0i transmission complete flag (USBIFR0/EP0i TS = 0)
End of control transfer
End of control transfer

Figure 19.9 Status Stage (Control-Out) Operation

The control-out status stage starts with an IN token from the host. When an IN-token is received at the start of the status stage, there is not yet any data in the EP0i FIFO, and so an EP0i transfer request interrupt is generated. The application recognizes from this interrupt that the status stage has started. Next, in order to transmit 0-byte data to the host, 1 is written to the EP0i packet enable bit but no data is written to the EP0i FIFO. As a result, the next IN token causes 0-byte data to be transmitted to the host, and control transfer ends.

After the application has finished all processing relating to the data stage, 1 should be written to the EP0i packet enable bit.

19.6.4 EP1 Bulk-Out Transfer (Dual FIFOs)

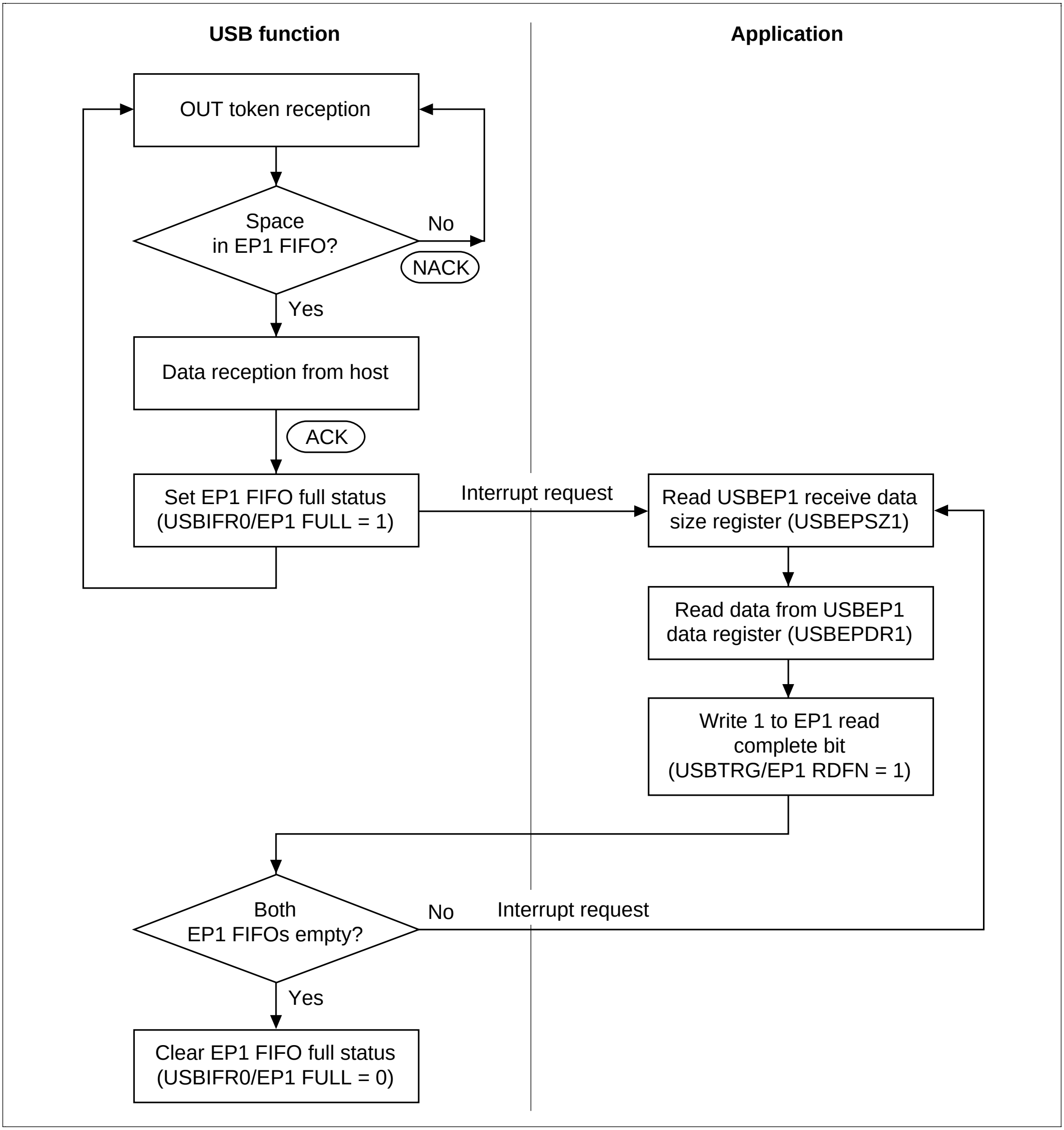

USB function
Application
OUT token reception
Space in EP1 FIFO?
No
NACK
Yes
Data reception from host
ACK
Set EP1 FIFO full status (USBIFR0/EP1 FULL = 1)
Interrupt request
Read USBEP1 receive data size register (USBEPSZ1)
Read data from USBEP1 data register (USBEPDR1)
Write 1 to EP1 read complete bit (USBTRG/EP1 RDFN = 1)
Both EP1 FIFOs empty?
No
Interrupt request
Yes
Clear EP1 FIFO full status (USBIFR0/EP1 FULL = 0)

Figure 19.10 EP1 Bulk-Out Transfer Operation

EP1 has two 64-byte FIFOs, but the user can perform data reception and receive data reads without being aware of this dual-FIFO configuration.

When one FIFO is full after reception is completed, the USBIFR0/EP1 FULL bit is set. After the first receive operation into one of the FIFOs when both FIFOs are empty, the other FIFO is empty, and so the next packet can be received immediately. When both FIFOs are full, NACK is returned to the host automatically. When reading of the receive data is completed following data reception, 1 is written to the USBTRG/EP1 RDFN bit. This operation empties the FIFO that has just been read, and makes it ready to receive the next packet.

19.6.5 EP2 Bulk-In Transfer (Dual FIFOs)

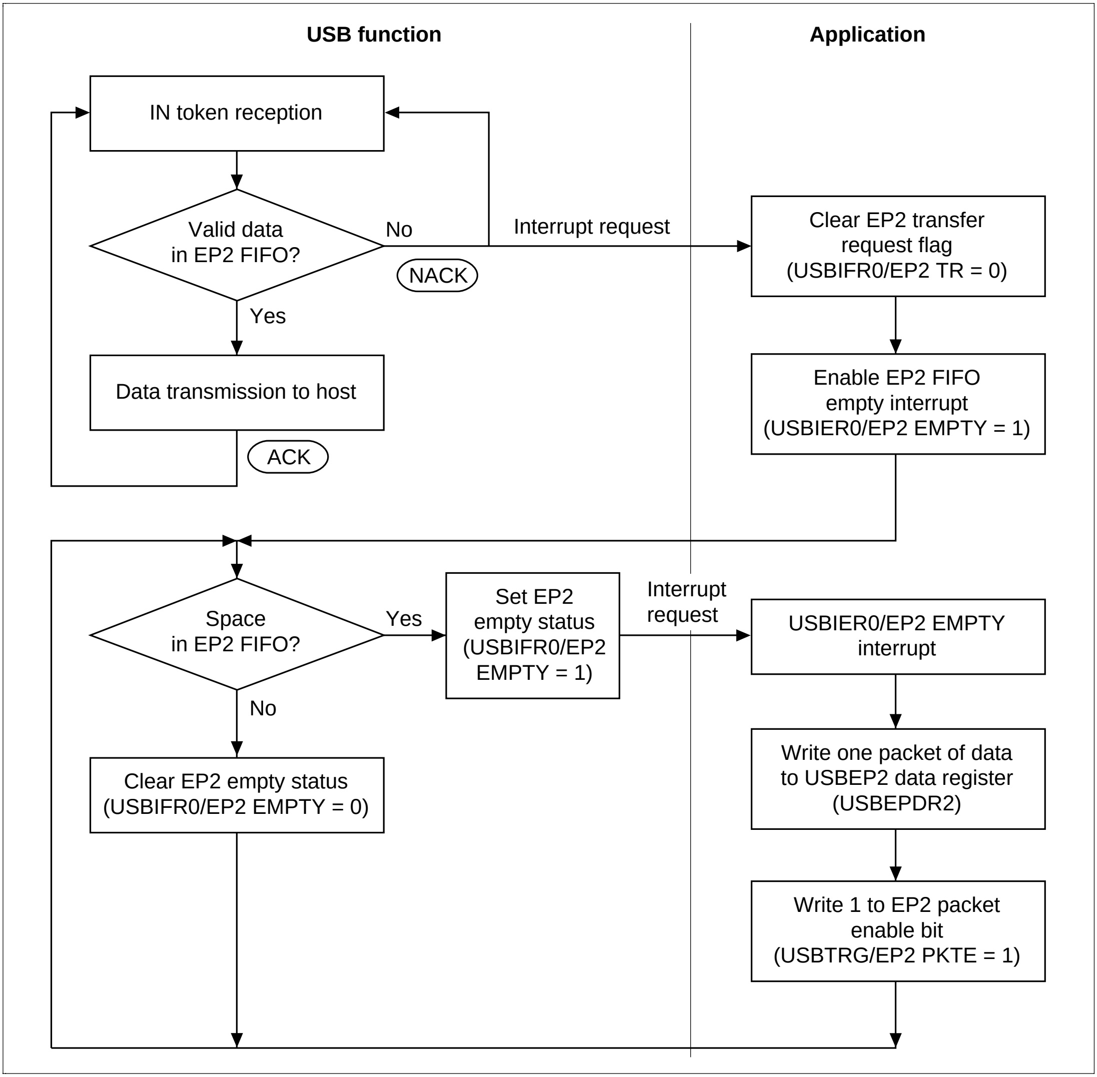

USB function
Application
IN token reception
Valid data in EP2 FIFO?
No
Interrupt request
NACK
Yes
Clear EP2 transfer request flag (USBIFR0/EP2 TR = 0)
Data transmission to host
ACK
Enable EP2 FIFO empty interrupt (USBIER0/EP2 EMPTY = 1)
Space in EP2 FIFO?
Yes
Set EP2 empty status (USBIFR0/EP2 EMPTY = 1)
Interrupt request
USBIER0/EP2 EMPTY interrupt
No
Clear EP2 empty status (USBIFR0/EP2 EMPTY = 0)
Write one packet of data to USBEP2 data register (USBEPDR2)
Write 1 to EP2 packet enable bit (USBTRG/EP2 PKTE = 1)

Figure 19.11 EP2 Bulk-In Transfer Operation

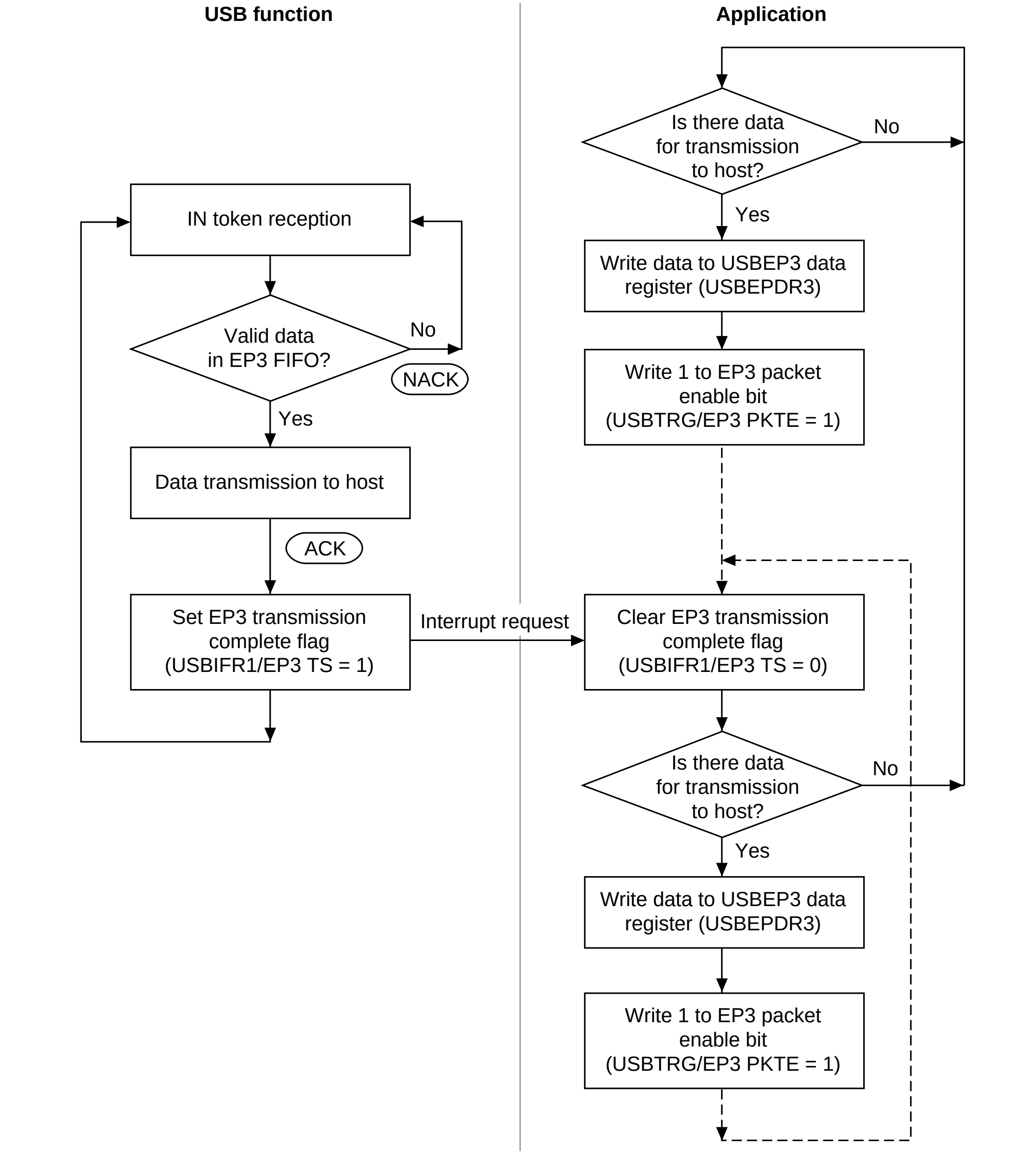
EP2 has two 64-byte FIFOs, but the user can perform data transmission and transmit data writes without being aware of this dual-FIFO configuration. However, one data write is performed for one FIFO. For example, even if both FIFOs are empty, it is not possible to perform EP2/PKTE at one time after consecutively writing 128 bytes of data. EP2/PKTE must be performed for each 64-byte write.

When performing bulk-in transfer, as there is no valid data in the FIFOs on reception of the first IN token, a USBIFR0/EP2 TR interrupt is requested. With this interrupt, 1 is written to the USBIER0/EP2 EMPTY bit, and the EP2 FIFO empty interrupt is enabled. At first, both EP2 FIFOs are empty, and so an EP2 FIFO empty interrupt is generated immediately.

The data to be transmitted is written to the data register using this interrupt. After the first transmit data write for one FIFO, the other FIFO is empty, and so the next transmit data can be written to the other FIFO immediately. When both FIFOs are full, EP2 EMPTY is cleared to 0. If at least one FIFO is empty, USBIFR0/EP2 EMPTY is set to 1. When ACK is returned from the host after data transmission is completed, the FIFO used in the data transmission becomes empty. If the other FIFO contains valid transmit data at this time, transmission can be continued.

When transmission of all data has been completed, write 0 to USBIER0/EP2 EMPTY and disable interrupt requests.

19.6.6 EP3 Interrupt-In Transfer


USB function
Application
Is there data for transmission to host?
No
Yes
Write data to USBEP3 data register (USBEPDR3)
Write 1 to EP3 packet enable bit (USBTRG/EP3 PKTE = 1)
IN token reception
Valid data in EP3 FIFO?
No
NACK
Yes
Data transmission to host
ACK
Set EP3 transmission complete flag (USBIFR1/EP3 TS = 1)
Interrupt request
Clear EP3 transmission complete flag (USBIFR1/EP3 TS = 0)
Is there data for transmission to host?
No
Yes
Write data to USBEP3 data register (USBEPDR3)
Write 1 to EP3 packet enable bit (USBTRG/EP3 PKTE = 1)

Note: This flowchart shows just one example of interrupt transfer processing. Other possibilities include an operation flow in which, if there is data to be transferred, the EP3 DE bit in the USB data status register is referenced to confirm that the FIFO is empty, and then data is written to the FIFO.

Figure 19.12 EP3 Interrupt-In Transfer Operation

19.7 Processing of USB Standard Commands and Class/Vendor Commands

19.7.1 Processing of Commands Transmitted by Control Transfer

A command transmitted from the host by control transfer may require decoding and execution of command processing on the application side. Whether command decoding is required on the application side is indicated in table 19.3 below.

Table 19.3 Command Decoding on Application Side

Decoding not Necessary on Application Side	Decoding Necessary on Application Side
Clear feature	Get descriptor
Get configuration	Class/Vendor command
Get interface	
Get status	
Set address	
Set configuration	
Set feature	
Set interface	

If decoding is not necessary on the application side, command decoding and data stage and status stage processing are performed automatically. No processing is necessary by the user. An interrupt is not generated in this case.

If decoding is necessary on the application side, the USB function module stores the command in the EP0s FIFO. After normal reception is completed, the USBIER0/SETUP TS flag is set and an interrupt request is generated. In the interrupt routine, 8 bytes of data must be read from the EP0s data register (USBEPDR0S) and decoded by firmware. The necessary data stage and status stage processing should then be carried out according to the result of the decoding operation.

The other standard commands (Synch frame and Set descriptor) are not supported. If a Synch frame or Set descriptor command is received from the host, an ACK handshake will be returned to the host in the setup stage, but a setup command reception complete interrupt will not be generated. In the data stage of these commands, a STALL handshake is returned to the host.

19.8 Stall Operations

19.8.1 Overview

This section describes stall operations in the USB function module. There are two cases in which the USB function module stall function is used:

- When the application forcibly stalls an endpoint for some reason
- When a stall is performed automatically within the USB function module due to a USB specification violation

The USB function module has internal status bits that hold the status (stall or non-stall) of each endpoint. When a transaction is sent from the host, the module references these internal status bits and determines whether to return a stall to the host. These bits cannot be cleared by the application; they must be cleared with a Clear Feature command from the host.

19.8.2 Forcible Stall by Application

The application uses the USBEPSTL register to issue a stall request for the USB function module. When the application wishes to stall a specific endpoint, it sets the corresponding bit in USBEPSTL (1-1 in figure 19.13). The internal status bits are not changed. When a transaction is sent from the host for the endpoint for which the USBEPSTL bit was set, the USB function module references the internal status bit, and if this is not set, references the corresponding bit in USBEPSTL (1-2 in figure 19.13). If the corresponding bit in USBEPSTL is set, the USB function module sets the internal status bit and returns a stall handshake to the host (1-3 in figure 19.13). If the corresponding bit in USBEPSTL is not set, the internal status bit is not changed and the transaction is accepted.

Once an internal status bit is set, it remains set until cleared by a Clear Feature command from the host, without regard to the USBEPSTL register. Even after a bit is cleared by the Clear Feature command (3-1 in figure 19.13), the USB function module continues to return a stall handshake while the bit in USBEPSTL is set, since the internal status bit is set each time a transaction is executed for the corresponding endpoint (1-2 in figure 19.13). To clear a stall, therefore, it is necessary for the corresponding bit in USBEPSTL to be cleared by the application, and also for the internal status bit to be cleared with a Clear Feature command (2-1, 2-2, and 2-3 in figure 19.13).

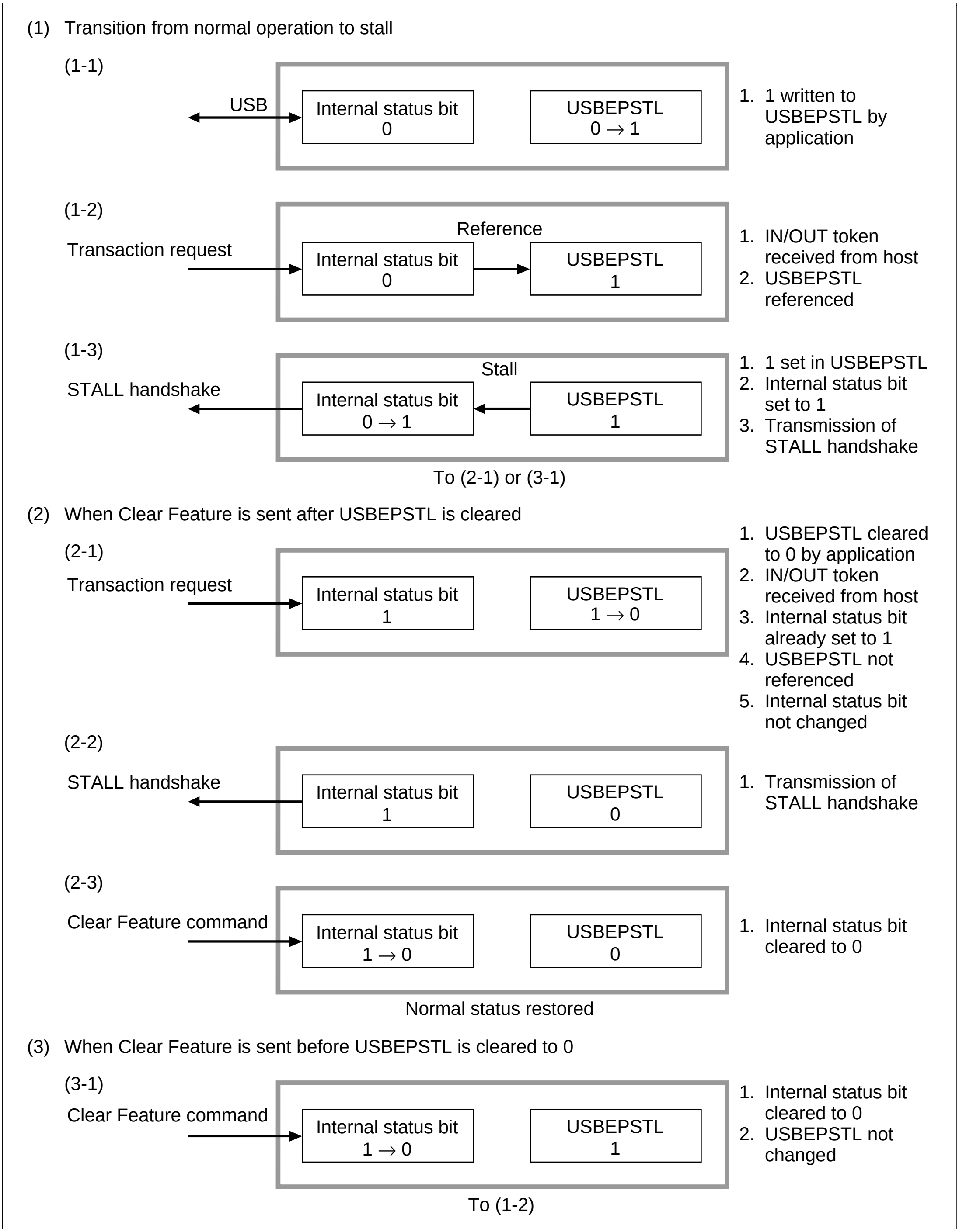

(1) Transition from normal operation to stall
(1-1)
USB
Internal status bit 0
USBEPSTL 0 → 1
1. 1 written to USBEPSTL by application
(1-2)
Transaction request
Reference
Internal status bit 0
USBEPSTL 1
1. IN/OUT token received from host
2. USBEPSTL referenced
(1-3)
STALL handshake
Stall
Internal status bit 0 → 1
USBEPSTL 1
1. 1 set in USBEPSTL
2. Internal status bit set to 1
3. Transmission of STALL handshake
To (2-1) or (3-1)
(2) When Clear Feature is sent after USBEPSTL is cleared
(2-1)
Transaction request
Internal status bit 1
USBEPSTL 1 → 0
1. USBEPSTL cleared to 0 by application
2. IN/OUT token received from host
3. Internal status bit already set to 1
4. USBEPSTL not referenced
5. Internal status bit not changed
(2-2)
STALL handshake
Internal status bit 1
USBEPSTL 0
1. Transmission of STALL handshake
(2-3)
Clear Feature command
Internal status bit 1 → 0
USBEPSTL 0
1. Internal status bit cleared to 0
Normal status restored
(3) When Clear Feature is sent before USBEPSTL is cleared to 0
(3-1)
Clear Feature command
Internal status bit 1 → 0
USBEPSTL 1
1. Internal status bit cleared to 0
2. USBEPSTL not changed
To (1-2)

Figure 19.13 Forcible Stall by Application

19.8.3 Automatic Stall by USB Function Module

When a stall setting is made with the Set Feature command, or in the event of a USB specification violation, the USB function module automatically sets the internal status bit for the relevant endpoint without regard to the USBEPSTL register, and returns a stall handshake (1-1 in figure 19.14).

Once an internal status bit is set, it remains set until cleared by a Clear Feature command from the host, without regard to the USBEPSTL register. After a bit is cleared by the Clear Feature command, USBEPSTL is referenced (3-1 in figure 19.14). The USB function module continues to return a stall handshake while the internal status bit is set, since the internal status bit is set even if a transaction is executed for the corresponding endpoint (2-1 and 2-2 in figure 19.14). To clear a stall, therefore, the internal status bit must be cleared with a Clear Feature command (3-1 in figure 19.14). If set by the application, USBEPSTL should also be cleared (2-1 in figure 19.14).

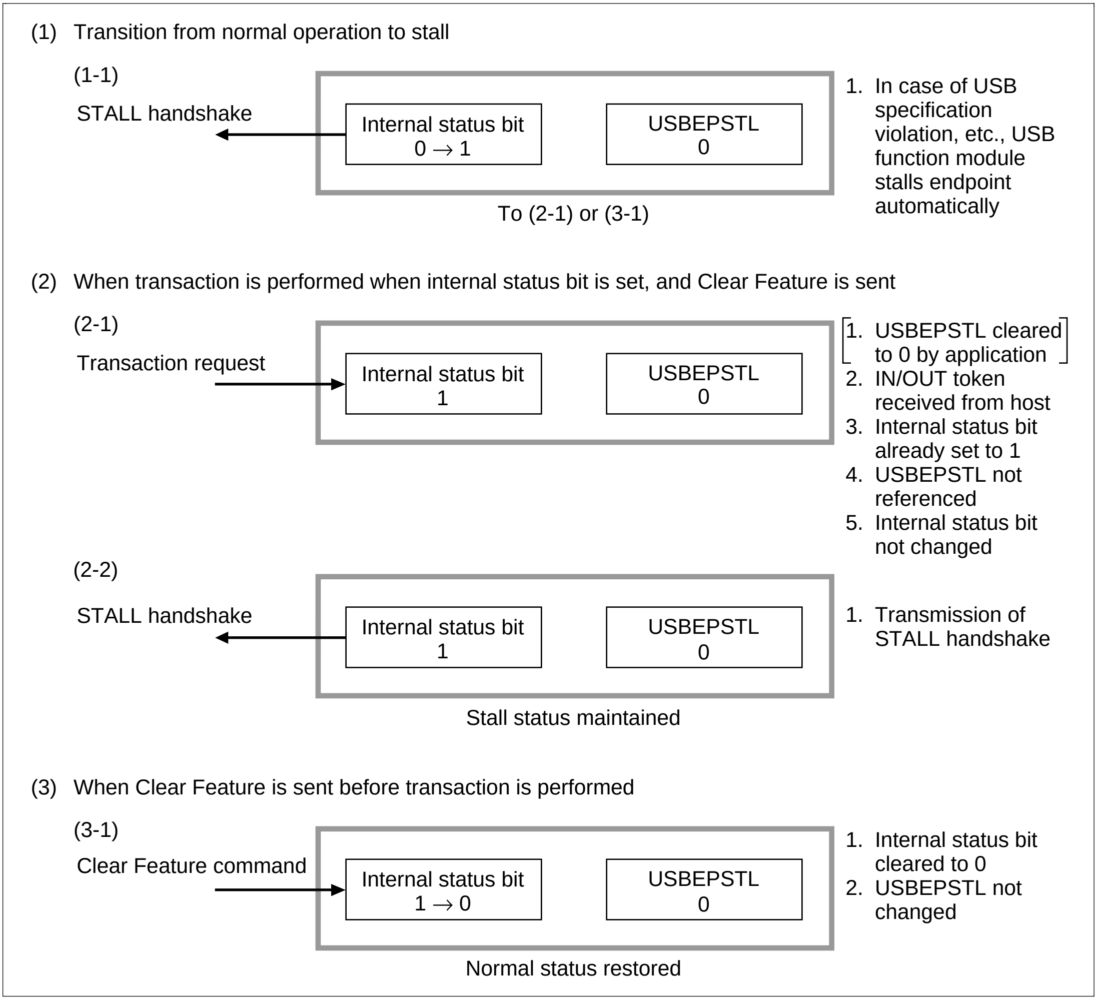
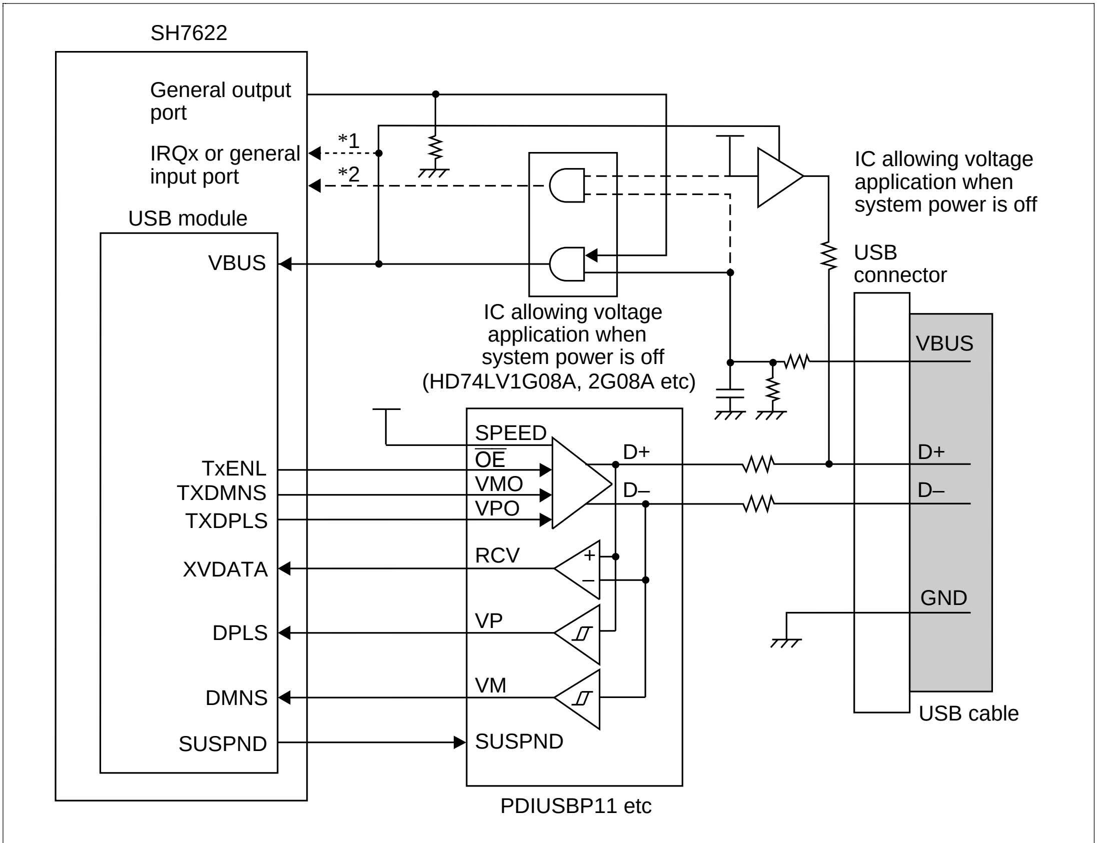

(1) Transition from normal operation to stall
(1-1)
STALL handshake
Internal status bit
0 → 1
USBEPSTL
0
1. In case of USB specification violation, etc., USB function module stalls endpoint automatically
To (2-1) or (3-1)
(2) When transaction is performed when internal status bit is set, and Clear Feature is sent
(2-1)
Transaction request
Internal status bit
1
USBEPSTL
0
[1. USBEPSTL cleared to 0 by application]
2. IN/OUT token received from host
3. Internal status bit already set to 1
4. USBEPSTL not referenced
5. Internal status bit not changed
(2-2)
STALL handshake
Internal status bit
1
USBEPSTL
0
1. Transmission of STALL handshake
Stall status maintained
(3) When Clear Feature is sent before transaction is performed
(3-1)
Clear Feature command
Internal status bit
1 → 0
USBEPSTL
0
1. Internal status bit cleared to 0
2. USBEPSTL not changed
Normal status restored

Figure 19.14 Automatic Stall by USB Function Module

19.9 Example of USB External Circuitry


SH7622
General output port
IRQx or general input port
*1
*2
USB module
VBUS
IC allowing voltage application when system power is off
USB connector
IC allowing voltage application when system power is off (HD74LV1G08A, 2G08A etc)
VBUS
SPEED
OE
VMO
VPO
D+
D–
TxENL
TXDMNS
TXDPLS
D+
D–
RCV
+
–
XVDATA
GND
VP
DPLS
VM
DMNS
USB cable
SUSPND
SUSPND
PDIUSBP11 etc

Notes: *1------ : When USB cable disconnection is detected.
*2– – – : When USB cable connection/disconnection is detected regardless of D+ pull-up control.

Figure 19.15 Example of USB External Circuitry

USB Transceiver: The USB function module in the SH7622 does not include a USB transceiver. Therefore, a USB transceiver IC (such as a PDUSBP11) should be connected externally. The USB transceiver manufacturer should be consulted concerning the recommended circuit from the USB transceiver to the USB connector, etc.

D+ Pull-Up Control: In a system where it is wished to disable USB host/hub connection notification (D+ pull-up) (during high-priority processing or initialization processing, for example), D+ pull-up should be controlled using a general output port. However, if a USB cable is already connected to the host/hub and D+ pull-up is prohibited, D+ and D– will both go low (both pulled down on the host/hub side) and the USB module will mistakenly identify this as reception of a USB bus reset from the host. Therefore, the D+ pull-up control signal and VBUS pin input signal should be controlled using a general output port and the USB cable VBUS (AND circuit) as shown in the circuit example above. (The UDC core in the SH7622 maintains the powered state when the VBUS pin is low, regardless of the D+/D– state.)

Detection of USB Cable Connection/Disconnection: As USB states, etc., are managed by hardware in this module, a VBUS signal that recognizes connection/disconnection is necessary. The power supply signal (VBUS) in the USB cable is used for this purpose. However, if the cable is connected to the USB host/hub when the function (SH7622-installed system) power is off, a voltage (5 V) will be applied from the USB host/hub. Therefore, an IC (such as an HD74LV1G08A or 2G08A) that allows voltage application when the system power is off should be connected externally.

This module incorporates a USB bus connection interrupt function, but this interrupt is generated on detection of a rising edge on the VBUS pin. Therefore, in an application that requires disconnection to be recognized, disconnection should be detected using IRQx or a general input port (*1 in figure 19.15).

Also, in an application that requires VBUS connection/disconnection to be recognized regardless of D+ pull-up control, this should be detected using IRQx or a general input port (*2 in figure 19.15).

Note: Both-edge detection is not supported for SH7622 external interrupts. Therefore, when performing connection/disconnection detection using IRQx, before setting the interrupt function, either falling edge detection or rising edge detection should be selected after the state has been recognized using the port function of the IRQx pin.

Note on sample USB external circuit: This USB external circuit is only an example for reference purposes, and it is necessary to confirm that there are no problems in terms of the system before undertaking board design. Operation is not guaranteed with this sample circuit.

Also, if external surge and ESD noise countermeasures are required for the system, a protective diode or the like should be used for this purpose.

19.10 Usage Notes

1. 32-bit access to USBEP1 data register

   The USBEP1 data register (USBEPDR1) and USBEP2 data register (USBEPDR2) can be accessed in byte or longword units.

   If the receive data size in bytes is not a multiple of 4, when performing longword access to the USBEP1 data register, reading should be carried out as shown in the examples below. Figure 19.16 shows examples in which 7 bytes are received.

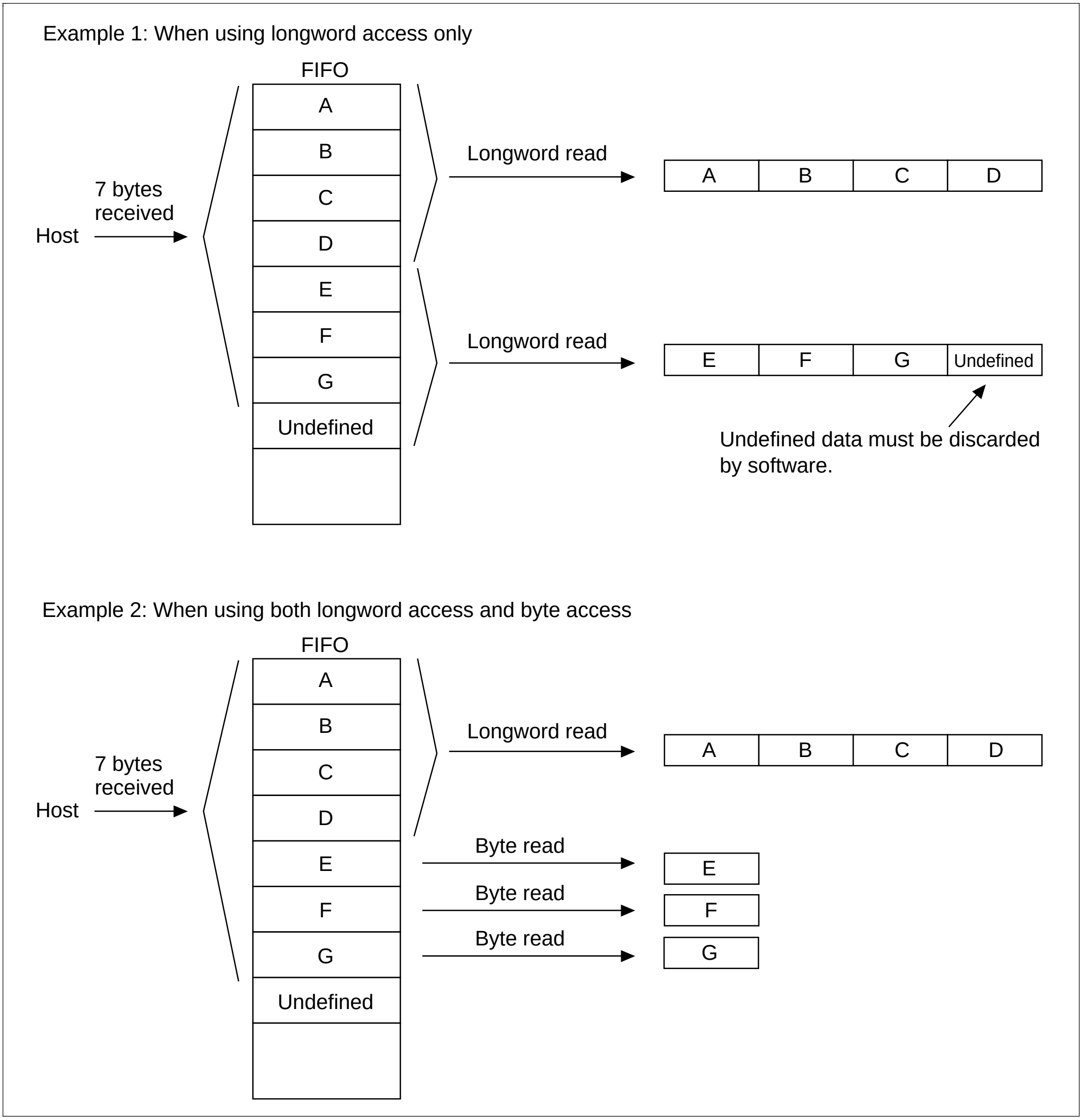


Figure 19.16 Examples of Read Operation when 7 Bytes are Received

Section 20 Compare Match Timer 1 (CMT1)

20.1 Overview

CMT1 is a compare match timer (CMT) that generates DMA transfer requests or interrupt requests. CMT1 is a 16-bit counter.

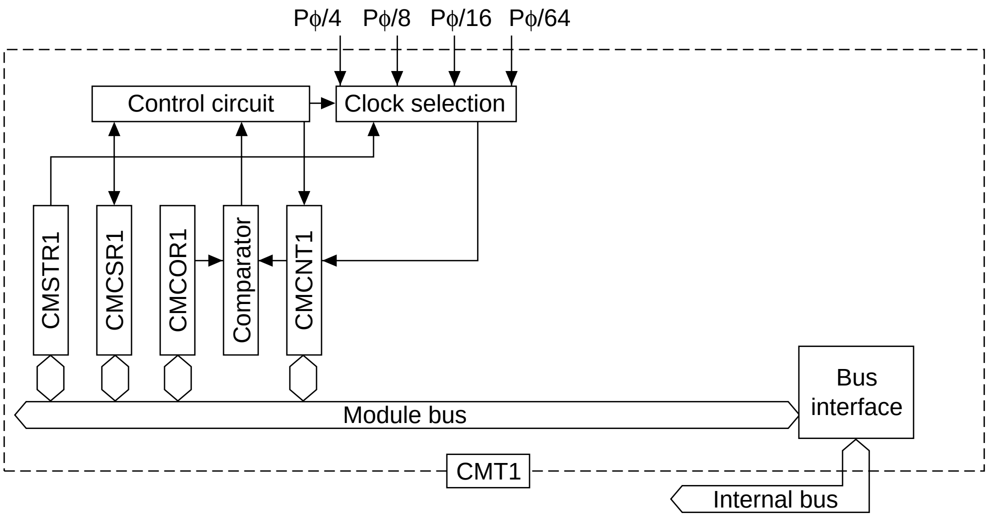
20.1.1 Features

CMT1 has the following features.

- Selection of four counter input clocks
  - Any of four internal clocks (Pϕ/4, Pϕ/8, Pϕ/16, Pϕ/64) can be selected.
- Selection of DMA transfer request or interrupt request generation on compare match
- When not in use, CMT1 can be stopped by halting its clock supply to reduce power consumption.

20.1.2 Block Diagram

Figure 20.1 shows a block diagram of CMT1.



CMSTR1: Compare match timer start register 1
CMCSR1: Compare match timer control/status register 1
CMCOR1: Compare match timer constant register 1
CMCNT1: Compare match counter 1

Figure 20.1 Block Diagram of Compare Match Timer 1

20.1.3 Register Configuration

Table 20.1 summarizes the CMT1 registers.

Table 20.1 CMT1 Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Compare match timer start register 1	CMSTR1	R/W	H'0000	H'A4002070	16
Compare match timer control/status register 1	CMCSR1	R/(W)*	H'0000	H'A4002072	16
Compare match counter 1	CMCNT1	R/W	H'0000	H'A4002074	16
Compare match timer constant register 1	CMCOR1	R/W	H'FFFF	H'A4002076	16

Note: * Only 0 can be written to the CMF bit in CMCSR1, to clear the flag.

20.2 Register Descriptions

20.2.1 Compare Match Timer Start Register 1 (CMSTR1)

CMSTR1 is a 16-bit register that selects whether compare match counter 1 (CMCNT1) operates or is stopped.

CMSTR1 is initialized to H'0000 by a reset, but is not initialized in standby mode.

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

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

Bits 15 to 1—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 0—Count Start (STR): Specifies whether compare match counter 1 operates or is stopped.

Bit 0: STR	Description	
0	CMCNT1 count is stopped	(Initial value)
1	CMCNT1 count is started	

20.2.2 Compare Match Timer Control/Status Register 1 (CMCSR1)

CMCSR1 is a 16-bit register that indicates compare match generation, enables interrupts or DMA transfer requests, and selects the counter input clock.

CMCSR1 is initialized to H'0000 by a reset, but is not initialized in standby mode.

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

Bit:	7	6	5	4	3	2	1	0
	CMF	—	CMR1	CMR0	—	—	CKS1	CKS0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R	R/W	R/W	R	R	R/W	R/W

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

Bits 15 to 8, 6, 3, and 2—Reserved: These bits are always read as 0. The write value should always be 0.

Bit 7—Compare Match Flag (CMF): Indicates whether or not the values of CMCNT1 and CMCOR1 match.

Bit 7: CMF	Description
0	CMCNT1 and CMCOR1 values do not match (Initial value) [Clearing condition] When 0 is written to CMF after reading CMF = 1
1	CMCNT1 and CMCOR1 values match

Bits 5 and 4—Compare Match Request 1 and 0 (CMR1, CMR0): These bits enables or disable DMA transfer request or interrupt request generation when a compare match occurs.

Bit 5: CMR1	Bit 4: CMR0	Description
0	0	DMA transfer request/interrupt request disabled (Initial value)
	1	DMA transfer request enabled
1	0	Interrupt request enabled
	1	Reserved (Do not set)

Bits 1 and 0—Clock Select 1 and 0 (CKS1, CKS0): These bits select the clock to be input to CMCNT1 from four internal clocks obtained by dividing the peripheral operating clock (Pφ). When the STR bit in CMSTR1 is set to 1, CMCNT1 starts counting on the clock selected with bits CKS1 and CKS0.

Bit 1: CKS1	Bit 0: CKS0	Description	
0	0	Pφ/4	(Initial value)
	1	Pφ/8	
1	0	Pφ/16	
	1	Pφ/64	

20.2.3 Compare Match Counter 1 (CMCNT1)

CMCNT1 is a 16-bit register used as an up-counter. When the counter input clock is selected with bits CKS1 and CKS0 in CMCSR1 and the STR bit in CMSTR1 is set to 1, CMCNT1 starts counting using the selected clock.

When the value in CMCNT1 and the value in compare match constant register 1 (CMCOR1) match, CMCNT1 is cleared to H'0000 and the CMF flag in CMCSR1 is set to 1.

CMCNT1 is initialized to H'0000 by a reset, but is not initialized in standby mode.

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

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

20.2.4 Compare Match Constant Register 1 (CMCOR1)

CMCOR1 is a 16-bit register that sets the interval up to a compare match with CMCNT1.

CMCOR1 is initialized to H'FFFF by a reset, but is not initialized in standby mode.

Bit:	15	14	13	12	11	10	9	8
Initial value:	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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

20.3 Operation

20.3.1 Interval Count Operation

When an internal clock is selected with bits CKS1 and CKS0 in CMCSR1 and the STR bit in CMSTR1 is set to 1, CMCNT1 starts incrementing using the selected clock. When the values in CMCNT1 and CMCOR1 match, CMCNT1 is cleared to H'0000 and the CMF flag in CMCSR1 is set to 1. CMCNT1 then starts counting up again from H'0000.

Figure 20.2 shows the operation of the compare match counter.

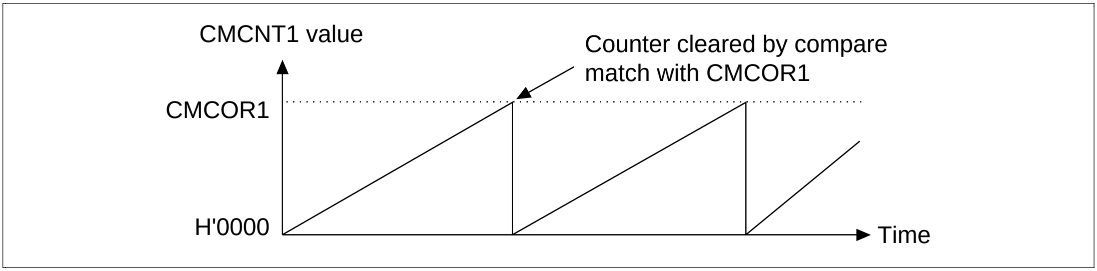


Figure 20.2 Counter Operation

20.3.2 CMCNT Count Timing

One of four internal clocks (Pϕ/4, Pϕ/8, Pϕ/16, Pϕ/64) obtained by dividing the Pϕ clock can be selected with bits CKS1 and CKS0 in CMCSR1. Figure 20.3 shows the timing.

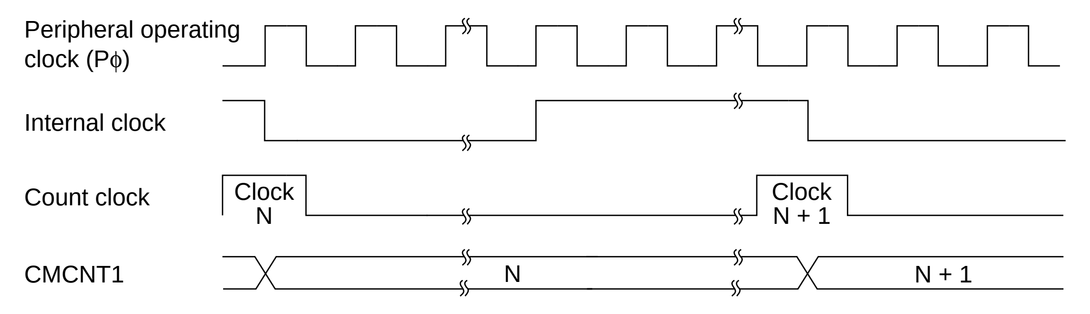


Figure 20.3 Count Timing

20.4 Compare Matches

20.4.1 Timing of Compare Match Flag Setting

When CMCOR1 and CMCNT1 match, a compare match signal is generated and the CMF bit in CMCSR1 is set to 1. The compare match signal is generated in the last state in which the values match (when the CMCNT1 value is updated to H'0000). That is, after a match between CMCOR1 and CMCNT1, the compare match signal is not generated until the next CMCNT1 counter clock input. Figure 20.4 shows the timing of CMF bit setting.

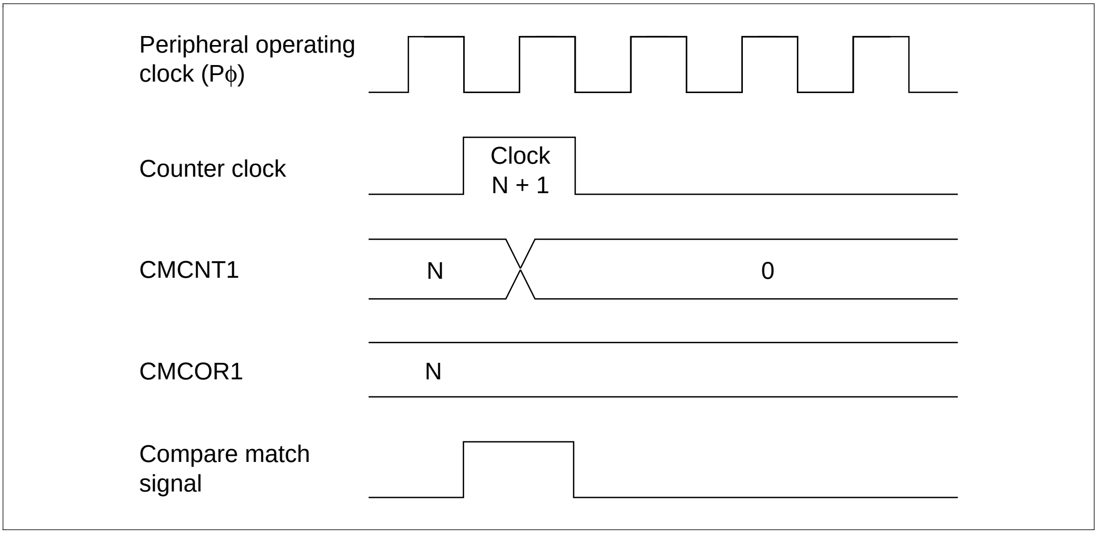


Figure 20.4 Timing of CMF Setting

20.4.2 DMA Transfer Requests and Interrupt Requests

Generation of a DMA transfer request or an interrupt request when a compare match occurs can be selected with bits CMR1 and CMR0 in CMCSR1.

With a DMA transfer request, the request signal is cleared automatically when the DMAC accepts the request. However, the CMF bit in CMCSR1 is not cleared to 0.

An interrupt request is cleared by writing 0 to the CMF bit in CMCSR1. Therefore, an operation to set CMF = 0 must be performed by the user in the exception handling routine. If this operation is not carried out, another interrupt will be generated.

20.4.3 Timing of Compare Match Flag Clearing

The CMF bit in CMCSR1 is cleared by reading 1 from this bit, then writing 0.

Section 21 Pin Function Controller (PFC)

21.1 Overview

The pin function controller (PFC) consists of registers for selecting multiplex pin functions and their input/output direction. The pin function and input/output direction can be selected for individual pins irrespective of the operating mode of the SH7622. Table 21.1 shows the SH7622's multiplex pins.

Table 21.1 Multiplex Pins

Port	Port Function (Related Module)	Other Function(s) (Related Module)
A	PTA7 input/output (port)	D23 input/output (data bus)
A	PTA6 input/output (port)	D22 input/output (data bus)
A	PTA5 input/output (port)	D21 input/output (data bus)
A	PTA4 input/output (port)	D20 input/output (data bus)
A	PTA3 input/output (port)	D19 input/output (data bus)
A	PTA2 input/output (port)	D18 input/output (data bus)
A	PTA1 input/output (port)	D17 input/output (data bus)
A	PTA0 input/output (port)	D16 input/output (data bus)
B	PTB7 input/output (port)	D31 input/output (data bus)
B	PTB6 input/output (port)	D30 input/output (data bus)
B	PTB5 input/output (port)	D29 input/output (data bus)
B	PTB4 input/output (port)	D28 input/output (data bus)
B	PTB3 input/output (port)	D27 input/output (data bus)
B	PTB2 input/output (port)	D26 input/output (data bus)
B	PTB1 input/output (port)	D25 input/output (data bus)
B	PTB0 input/output (port)	D24 input/output (data bus)
C	PTC7 input/output (port)	IRQ6 input (INTC)
C	PTC6 input/output (port)	IRQ7 input (INTC)
C	PTC5 input/output (port)	XVDATA input (USB)
C	PTC4 input/output (port)	$\overline{\text{TXENL}}$ output (USB)
C	PTC3 output (port)	NF
C	PTC2 output (port)	NF
C	PTC1 input (port)	NF
C	PTC0 output (port)	—

Table 21.1 Multiplex Pins (cont)

Port	Port Function (Related Module)	Other Function(s) (Related Module)
D	PTD7 input/output (port)	DACK1 output (DMAC)
D	PTD6 input (port)	$\overline{\text{DREQ1}}$ input (DMAC)
D	PTD5 input/output (port)	DACK0 output (DMAC)
D	PTD4 input (port)	$\overline{\text{DREQ0}}$ input (DMAC)
D	PTD3 input/output (port)	VBUS input (USB)
D	PTD2 input/output (port)	SUSPND output (USB)
D	PTD1 input/output (port)	DRAK0 output (DMAC)
D	PTD0 input/output (port)	DRAK1 output (DMAC)
E	PTE7 input/output (port)	$\overline{\text{AUDSYNC}}$ output (AUD)
E	PTE6 input/output (port)	—
E	PTE5 input/output (port)	—
E	PTE4 input/output (port)	—
E	PTE3 input/output (port)	—
E	PTE2 input/output (port)	$\overline{\text{RAS3U}}$ output (BSC)
E	PTE1 input/output (port)	—
E	PTE0 input/output (port)	TDO output (H-UDI)
F	PTF7 input (port)	$\overline{\text{TRST}}$ input (AUD, H-UDI)
F	PTF6 input (port)	TMS input (H-UDI)
F	PTF5 input (port)	TDI input (H-UDI)
F	PTF4 input (port)	TCK input (H-UDI)
F	PTF3 input (port)	DMNS input (USB)
F	PTF2 input (port)	DPLS input (USB)
F	PTF1 input (port)	TXDPLS output (USB)
F	PTF0 input (port)	TXDMNS output (USB)
G	PTG7 input (port)	—
G	PTG6 input (port)	$\overline{\text{ASEMD0}}$ input (AUD, H-UDI)
G	PTG5 input (port)	$\overline{\text{ASEBRKAK}}$ output (AUD)
G	PTG4 input (port)	UCLK input (USB)
G	PTG3 input (port)	AUDATA3 output (AUD)
G	PTG2 input (port)	AUDATA2 output (AUD)
G	PTG1 input (port)	AUDATA1 output (AUD)
G	PTG0 input (port)	AUDATA0 output (AUD)

Table 21.1 Multiplex Pins (cont)

Port	Port Function (Related Module)	Other Function(s) (Related Module)
H	PTH7 input/output (port)	TCLK input/output (timer)
H	PTH6 input (port)	AUDCK input (AUD)
H	PTH5 input (port)	$\overline{\text{ADTRG}}$ input (ADC)
H	PTH4 input (port)	IRQ4 input (INTC)
H	PTH3 input (port)	IRQ3 input (INTC)
H	PTH2 input (port)	IRQ2 input (INTC)
H	PTH1 input (port)	IRQ1 input (INTC)
H	PTH0 input (port)	IRQ0 input (INTC)
J	PTJ7 input/output (port)	STATUS1 output (CPG)
J	PTJ6 input/output (port)	STATUS0 output (CPG)
J	PTJ5 output (port)	NF
J	PTJ4 output (port)	NF
J	PTJ3 input/output (port)	CASU output (BSC)
J	PTJ2 input/output (port)	CASL output (BSC)
J	PTJ1 output (port)	NF
J	PTJ0 input/output (port)	$\overline{\text{RAS3L}}$ output (BSC)
K	PTK7 input/output (port)	$\overline{\text{WE3}}$ output (BSC) / DQMUU output (BSC)
K	PTK6 input/output (port)	$\overline{\text{WE2}}$ output (BSC) / DQMUL output (BSC)
K	PTK5 input/output (port)	CKE output (BSC)
K	PTK4 input/output (port)	$\overline{\text{BS}}$ output (BSC)
K	PTK3 input/output (port)	$\overline{\text{CS5}}$ output (BSC)
K	PTK2 input/output (port)	$\overline{\text{CS4}}$ output (BSC)
K	PTK1 input/output (port)	$\overline{\text{CS3}}$ output (BSC)
K	PTK0 input/output (port)	$\overline{\text{CS2}}$ output (BSC)
L	PTL7 input (port)	—
L	PTL6 input (port)	—
L	PTL5 input (port)	—
L	PTL4 input (port)	—
L	PTL3 input (port)	AN3 input (ADC)
L	PTL2 input (port)	AN2 input (ADC)
L	PTL1 input (port)	AN1 input (ADC)
L	PTL0 input (port)	AN0 input (ADC)

Table 21.1 Multiplex Pins (cont)

Port	Port Function (Related Module)	Other Function(s) (Related Module)
SCPT	SCPT7 input (port)	IRQ5 input (INTC)
SCPT	SCPT6 input/output (port)	—
SCPT	SCPT5 input/output (port)	SCK2 input/output (SCIF2)
SCPT	SCPT4 input (port)	RxD2 input (SCIF2)
	SCPT4 output (Port)	TxD2 output (SCIF2)
SCPT	SCPT3 input/output (port)	SCK1 input/output (SCIF1)
SCPT	SCPT2 input (port)	RxD1 input (SCIF1)
	SCPT2 output (Port)	TxD1 output (SCIF1)
SCPT	SCPT1 input/output (port)	SCK0 input/output (SCIF0)
SCPT	SCPT0 input (port)	RxD0 input (SCIF0)
	SCPT0 output (Port)	TxD0 output (SCIF0)

21.2 Register Configuration

PFC registers are listed in table 21.2.

Table 21.2 PFC Registers

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port A control register	PACR	R/W	H'0000	H'A4000100	16
Port B control register	PBCR	R/W	H'0000	H'A4000102	16
Port C control register	PCCR	R/W	H'AA00	H'A4000104	16
Port D control register	PDCR	R/W	H'AAAA	H'A4000106	16
Port E control register	PECR	R/W	H'AAAA/ H'2AA8	H'A4000108	16
Port F control register	PFCR	R/W	H'AAAA/ H'00AA	H'A400010A	16
Port G control register	PGCR	R/W	H'AAAA/ H'A200	H'A400010C	16
Port H control register	PHCR	R/W	H'AAAA/ H'8AAA	H'A400010E	16
Port J control register	PJCR	R/W	H'0000	H'A4000110	16
Port K control register	PKCR	R/W	H'0000	H'A4000112	16
Port L control register	PLCR	R/W	H'AA00	H'A4000114	16
SC port control register	SCPCR	R/W	H'A888	H'A4000116	16

Note: The initial value of the port E, F, G, and H control registers depends on the state of the $\overline{\text{ASEMD0}}$ pin. If a low level is input at the $\overline{\text{ASEMD0}}$ pin while the $\overline{\text{RESETP}}$ pin is asserted, debugger mode (H-UDI/AUD) will be entered; if a high level is input, the normal usage state (main chip mode) will be entered. See section 24, Hitachi User Debug Interface (H-UDI)/ Advanced User Debugger (AUD), for more information on the H-UDI/AUD.

21.3 Register Descriptions

21.3.1 Port A Control Register (PACR)

PACR is a 16-bit readable/writable register that selects port A pin functions. PACR 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
	PA7MD1	PA7MD0	PA6MD1	PA6MD0	PA5MD1	PA5MD0	PA4MD1	PA4MD0
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
	PA3MD1	PA3MD0	PA2MD1	PA2MD0	PA1MD1	PA1MD0	PA0MD1	PA0MD0
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—PA7 Mode 1 and 0 (PA7MD1, PA7MD0)
Bits 13 and 12—PA6 Mode 1 and 0 (PA6MD1, PA6MD0)
Bits 11 and 10—PA5 Mode 1 and 0 (PA5MD1, PA5MD0)
Bits 9 and 8—PA4 Mode 1 and 0 (PA4MD1, PA4MD0)
Bits 7 and 6—PA3 Mode 1 and 0 (PA3MD1, PA3MD0)
Bits 5 and 4—PA2 Mode 1 and 0 (PA2MD1, PA2MD0)
Bits 3 and 2—PA1 Mode 1 and 0 (PA1MD1, PA1MD0)
Bits 1 and 0—PA0 Mode 1 and 0 (PA0MD1, PA0MD0)
These bits select the pin function and control MOS input pull-up.

Bit (2n + 1): **PAnMD1**	**Bit 2n:** **PAnMD0**	**Pin Function**	
0	0	Other function (see table 21.1)	(Initial value)
	1	Port output	
1	0	Port input (MOS pull-up on)	
	1	Port input (MOS pull-up off)	

(n = 0 to 7)

21.3.2 Port B Control Register (PBCR)

PBCR is a 16-bit readable/writable register that selects port B pin functions. PBCR 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
	PB7MD1	PB7MD0	PB6MD1	PB6MD0	PB5MD1	PB5MD0	PB4MD1	PB4MD0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PB3MD1	PB3MD0	PB2MD1	PB2MD0	PB1MD1	PB1MD0	PB0MD1	PB0MD0
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—PB7 Mode 1 and 0 (PB7MD1, PB7MD0)
Bits 13 and 12—PB6 Mode 1 and 0 (PB6MD1, PB6MD0)
Bits 11 and 10—PB5 Mode 1 and 0 (PB5MD1, PB5MD0)
Bits 9 and 8—PB4 Mode 1 and 0 (PB4MD1, PB4MD0)
Bits 7 and 6—PB3 Mode 1 and 0 (PB3MD1, PB3MD0)
Bits 5 and 4—PB2 Mode 1 and 0 (PB2MD1, PB2MD0)
Bits 3 and 2—PB1 Mode 1 and 0 (PB1MD1, PB1MD0)
Bits 1 and 0—PB0 Mode 1 and 0 (PB0MD1, PB0MD0)
These bits select the pin function and control MOS input pull-up.

Bit (2n + 1):	**Bit 2n:**	
PBnMD1	**PBnMD0**	**Pin Function**
0	0	Other function (see table 21.1) (Initial value)
	1	Port output
1	0	Port input (MOS pull-up on)
	1	Port input (MOS pull-up off)

(n = 0 to 7)

21.3.3 Port C Control Register (PCCR)

PCCR is a 16-bit readable/writable register that selects port C pin functions. PCCR is initialized to H'AA00 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
	PC7MD1	PC7MD0	PC6MD1	PC6MD0	PC5MD1	PC5MD0	PC4MD1	PC4MD0
Initial value:	1	0	1	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PC3MD1	PC3MD0	PC2MD1	PC2MD0	PC1MD1	PC1MD0	PC0MD1	PC0MD0
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—PC7 Mode 1 and 0 (PC7MD1, PC7MD0)
Bits 13 and 12—PC6 Mode 1 and 0 (PC6MD1, PC6MD0)
Bits 11 and 10—PC5 Mode 1 and 0 (PC5MD1, PC5MD0)
Bits 9 and 8—PC4 Mode 1 and 0 (PC4MD1, PC4MD0)
Bits 7 and 6—PC3 Mode 1 and 0 (PC3MD1, PC3MD0)
Bits 5 and 4—PC2 Mode 1 and 0 (PC2MD1, PC2MD0)
Bits 3 and 2—PC1 Mode 1 and 0 (PC1MD1, PC1MD0)
Bits 1 and 0—PC0 Mode 1 and 0 (PC0MD1, PC0MD0)
These bits select the pin function and control MOS input pull-up.

Bit (2n + 1): PCnMD1	Bit 2n: PCnMD0	Pin Function	
0	0	NF*2	(Initial value)
	1	Port output	
1	—*1	Reserved*3	

(n = 0, 2, 3)

Notes: *1 0 or 1
*2 The “port output” setting must be selected when this port is used. When not used as a port (when designated as NF), the pin must be left open.
*3 Operation is not guaranteed if a reserved setting is selected.

Bit (2n + 1): PCnMD1	Bit 2n: PCnMD0	Pin Function	
0	0	NF*[1]	(Initial value)
	1	Reserved*[2]	
1	0	Port input (MOS pull-up on)	
	1	Port input (MOS pull-up off)	

(n = 1)

Notes: *1 The “port output” setting must be selected when this port is used. When not used as a port (when designated as NF), use a pull-down connection.

*2 Operation is not guaranteed if a reserved setting is selected.

Bit (2n + 1): PCnMD1	Bit 2n: PCnMD0	Pin Function	
0	0	Other function	
	1	Port output	
1	0	Port input (MOS pull-up on)	(Initial value)
	1	Port input (MOS pull-up off)	

(n = 4 to 7)

21.3.4 Port D Control Register (PDCR)

PDCR is a 16-bit readable/writable register that selects port D pin functions. PDCR is initialized to H'AAAA 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
	PD7MD1	PD7MD0	PD6MD1	PD6MD0	PD5MD1	PD5MD0	PD4MD1	PD4MD0
Initial value:	1	0	1	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PD3MD1	PD3MD0	PD2MD1	PD2MD0	PD1MD1	PD1MD0	PD0MD1	PD0MD0
Initial value:	1	0	1	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 and 14—PD7 Mode 1 and 0 (PD7MD1, PD7MD0)
Bits 11 and 10—PD5 Mode 1 and 0 (PD5MD1, PD5MD0)
Bits 7 and 6—PD3 Mode 1 and 0 (PD3MD1, PD3MD0)
Bits 5 and 4—PD2 Mode 1 and 0 (PD2MD1, PD2MD0)
Bits 3 and 2—PD1 Mode 1 and 0 (PD1MD1, PD1MD0)
Bits 1 and 0—PD0 Mode 1 and 0 (PD0MD1, PD0MD0)

These bits select the pin function and control MOS input pull-up.

Bit (2n + 1): PDnMD1	Bit 2n PDnMD0	Pin Function	
0	0	Other function	
	1	Port output	
1	0	Port input (MOS pull-up on)	(Initial value)
	1	Port input (MOS pull-up off)	

(n = 0 to 3)

Bit (2n + 1): PDnMD1	Bit 2n PDnMD0	Pin Function	
0	0	Other function	
	1	Port output	
1	—*	Port input (MOS pull-up off)	(Initial value)

(n = 5, 7)

Note: * 0 or 1

Bits 13 and 12—PD6 Mode 1 and 0 (PD6MD1, PD6MD0)
Bits 9 and 8—PD4 Mode 1 and 0 (PD4MD1, PD4MD0)

These bits select the pin function and control MOS input pull-up.

Bit (2n + 1): PDnMD1	Bit 2n: PDnMD0	Pin Function	
0	0	Other function (see table 21.1)	
	1	Reserved*	
1	0	Port input (MOS pull-up on)	(Initial value)
	1	Port input (MOS pull-up off)	

(n = 4, 6)

Note: * Operation is not guaranteed if the reserved setting is selected.

21.3.5 Port E Control Register (PECR)

PECR is a 16-bit readable/writable register that selects port E pin functions. PECR is initialized to H'AAAA ($\overline{\text{ASEMD0}}$ = 1) or H'2AA8 ($\overline{\text{ASEMD0}}$ = 0) 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
	PE7MD1	PE7MD0	PE6MD1	PE6MD0	PE5MD1	PE5MD0	PE4MD1	PE4MD0
Initial value:	1/0	0	1	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PE3MD1	PE3MD0	PE2MD1	PE2MD0	PE1MD1	PE1MD0	PE0MD1	PE0MD0
Initial value:	1	0	1	0	1	0	1/0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 and 14—PE7 Mode 1 and 0 (PE7MD1, PE7MD0)
Bits 13 and 12—PE6 Mode 1 and 0 (PE6MD1, PE6MD0)
Bits 11 and 10—PE5 Mode 1 and 0 (PE5MD1, PE5MD0)
Bits 9 and 8—PE4 Mode 1 and 0 (PE4MD1, PE4MD0)
Bits 7 and 6—PE3 Mode 1 and 0 (PE3MD1, PE3MD0)
Bits 5 and 4—PE2 Mode 1 and 0 (PE2MD1, PE2MD0)
Bits 3 and 2—PE1 Mode 1 and 0 (PE1MD1, PE1MD0)
Bits 1 and 0—PE0 Mode 1 and 0 (PE0MD1, PE0MD0)
These bits select the pin function and control MOS input pull-up.

Bit (2n + 1): PEnMD1	**Bit 2n: PEnMD0**	**Pin Function**	
0	0	Other function (n = 0, 7)*	(Initial value) $\overline{\text{ASEMD0}}$ = 0
	1	Port output	
1	0	Port input (MOS pull-up on)	(Initial value) $\overline{\text{ASEMD0}}$ = 1
	1	Port input (MOS pull-up off)	

(n = 0, 7)

Note: * Do not set when $\overline{\text{ASEMD0}}$ = 1. Operation is not guaranteed if this setting is made.

Bit (2n + 1): PEnMD1	Bit 2n: PEnMD0	Pin Function	
0	0	Reserved*[2]	
	1	Port output	
1	—*[1]	Port input (MOS pull-up off)	(Initial value)

(n = 1, 3 to 6)

Notes: *1 0 or 1

*2 Operation is not guaranteed if the reserved setting is selected.

Bit (2n + 1): PEnMD1	Bit 2n: PEnMD0	Pin Function	
0	0	Other function	
	1	Port output	
1	—*	Port input (MOS pull-up off)	(Initial value)

(n = 2)

Note: * 0 or 1

21.3.6 Port F Control Register (PFCR)

PFCR is a 16-bit readable/writable register that selects port F pin functions. PFCR is initialized to H'AAAA ($\overline{\text{ASEMD0}}$ = 1) or H'00AA ($\overline{\text{ASEMD0}}$ = 0) 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
	PF7MD1	PF7MD0	PF6MD1	PF6MD0	PF5MD1	PF5MD0	PF4MD1	PF4MD0
Initial value:	1/0	0	1/0	0	1/0	0	1/0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PF3MD1	PF3MD0	PF2MD1	PF2MD0	PF1MD1	PF1MD0	PF0MD1	PF0MD0
Initial value:	1	0	1	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 and 14—PF7 Mode 1 and 0 (PF7MD1, PF7MD0)
Bits 13 and 12—PF6 Mode 1 and 0 (PF6MD1, PF6MD0)
Bits 11 and 10—PF5 Mode 1 and 0 (PF5MD1, PF5MD0)
Bits 9 and 8—PF4 Mode 1 and 0 (PF4MD1, PF4MD0)
Bits 7 and 6—PF3 Mode 1 and 0 (PF3MD1, PF3MD0)
Bits 5 and 4—PF2 Mode 1 and 0 (PF2MD1, PF2MD0)
Bits 3 and 2—PF1 Mode 1 and 0 (PF1MD1, PF1MD0)
Bits 1 and 0—PF0 Mode 1 and 0 (PF0MD1, PF0MD0)

These bits select the pin function and control MOS input pull-up.

Bit (2n + 1):	**Bit 2n:**		
PFnMD1	**PFnMD0**	**Pin Function**	
0	0	Other function	(Initial value) $\overline{\text{ASEMD0}} = 0$
	1	Reserved*	
1	0	Port input (MOS pull-up on)	(Initial value) $\overline{\text{ASEMD0}} = 1$
	1	Port input (MOS pull-up off)	

(n = 4 to 7)

Note: * Operation is not guaranteed if the reserved setting is selected.

Bit (2n + 1):	**Bit 2n:**		
PFnMD1	**PFnMD0**	**Pin Function**	
0	0	Other function	
	1	Reserved*	
1	0	Port input (MOS pull-up on)	(Initial value)
	1	Port input (MOS pull-up off)	

(n = 0 to 3)

Note: * Operation is not guaranteed if the reserved setting is selected.

21.3.7 Port G Control Register (PGCR)

PGCR is a 16-bit readable/writable register that selects port G pin functions. PGCR is initialized to H'AAAA ($\overline{\text{ASEMD0}}$ = 1) or H'A200 ($\overline{\text{ASEMD0}}$ = 0) 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
	PG7MD1	PG7MD0	PG6MD1	PG6MD0	PG5MD1	PG5MD0	PG4MD1	PG4MD0
Initial value:	1	0	1	0	1/0	0	1	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
	PG3MD1	PG3MD0	PG2MD1	PG2MD0	PG1MD1	PG1MD0	PG0MD1	PG0MD0
Initial value:	1/0	0	1/0	0	1/0	0	1/0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 and 14—PG7 Mode 1 and 0 (PG7MD1, PG7MD0)
Bits 13 and 12—PG6 Mode 1 and 0 (PG6MD1, PG6MD0)
Bits 11 and 10—PG5 Mode 1 and 0 (PG5MD1, PG5MD0)
Bits 9 and 8—PG4 Mode 1 and 0 (PG4MD1, PG4MD0)
Bits 7 and 6—PG3 Mode 1 and 0 (PG3MD1, PG3MD0)
Bits 5 and 4—PG2 Mode 1 and 0 (PG2MD1, PG2MD0)
Bits 3 and 2—PG1 Mode 1 and 0 (PG1MD1, PG1MD0)
Bits 1 and 0—PG0 Mode 1 and 0 (PG0MD1, PG0MD0)
These bits select the pin function and control MOS input pull-up.

Bit (2n + 1): **PGnMD1**	**Bit 2n:** **PGnMD0**	**Pin Function**	
0	0	Other function ($\overline{\text{ASEMD0}}$ = 0) Reserved*[1] ($\overline{\text{ASEMD0}}$ = 1)	 (Initial value) $\overline{\text{ASEMD0}}$ = 0
	1	Reserved*[2]	
1	0	Port input (MOS pull-up on)	(Initial value) $\overline{\text{ASEMD0}}$ = 1
	1	Port input (MOS pull-up off)	

(n = 0 to 3, 5)

Notes: *1 Do not set when $\overline{\text{ASEMD0}}$ = 1. Operation is not guaranteed if this setting is made.
*2 Operation is not guaranteed if the reserved setting is selected.

Bit (2n + 1): PGnMD1	Bit 2n: PGnMD0	Pin Function	
0	0	Other function (n = 4, 6), reserved* (n = 7)	
	1	Reserved*	
1	0	Port input (MOS pull-up on)	(Initial value)
	1	Port input (MOS pull-up off)	

(n = 4, 6, 7)

Note: * Operation is not guaranteed if the reserved setting is selected.

21.3.8 Port H Control Register (PHCR)

PHCR is a 16-bit readable/writable register that selects port H pin functions. PHCR is initialized to H'AAAA ($\overline{\text{ASEMD0}}$ = 1) or H'8AAA ($\overline{\text{ASEMD0}}$ = 0) 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
	PH7MD1	PH7MD0	PH6MD1	PH6MD0	PH5MD1	PH5MD0	PH4MD1	PH4MD0
Initial value:	1	0	1/0	0	1	0	1	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
	PH3MD1	PH3MD0	PH2MD1	PH2MD0	PH1MD1	PH1MD0	PH0MD1	PH0MD0
Initial value:	1	0	1	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 and 14—PH7 Mode 1 and 0 (PH7MD1, PH7MD0)
These bits select the pin function and control MOS input pull-up.

Bit 15: PH7MD1	Bit 14: PH7MD0	Pin Function	
0	0	Other function	
	1	Port output	
1	0	Port input (MOS pull-up on)	(Initial value)
	1	Port input (MOS pull-up off)	

Bits 13 and 12—PH6 Mode 1 and 0 (PH6MD1, PH6MD0)
Bits 11 and 10—PH5 Mode 1 and 0 (PH5MD1, PH5MD0)
Bits 9 and 8—PH4 Mode 1 and 0 (PH4MD1, PH4MD0)
Bits 7 and 6—PH3 Mode 1 and 0 (PH3MD1, PH3MD0)
Bits 5 and 4—PH2 Mode 1 and 0 (PH2MD1, PH2MD0)
Bits 3 and 2—PH1 Mode 1 and 0 (PH1MD1, PH1MD0)
Bits 1 and 0—PH0 Mode 1 and 0 (PH0MD1, PH0MD0)

These bits select the pin function and control MOS input pull-up.

Bit (2n + 1): **PH6MD1**	**Bit 2n:** **PH6MD0**	**Pin Function**	
0	0	Other function*[1]	(Initial value) $\overline{\text{ASEMD0}}$ = 0
	1	Reserved*[2]	
1	0	Port input (MOS pull-up on)	(Initial value) $\overline{\text{ASEMD0}}$ = 1
	1	Port input (MOS pull-up off)	

(n = 6)

Notes: *1 Do not set when $\overline{\text{ASEMD0}}$ = 1. Operation is not guaranteed if this setting is made.
*2 Operation is not guaranteed if a reserved setting is selected.

Bit (2n + 1): **PHnMD1**	**Bit 2n:** **PHnMD0**	**Pin Function**	
0	0	Other function	
	1	Reserved*	
1	0	Port input (MOS pull-up on)	(Initial value)
	1	Port input (MOS pull-up off)	

(n = 0 to 5)

Note: * Operation is not guaranteed if a reserved setting is selected.

21.3.9 Port J Control Register (PJCR)

PJCR is a 16-bit readable/writable register that selects port J pin functions. PJCR 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
	PJ7MD1	PJ7MD0	PJ6MD1	PJ6MD0	PJ5MD1	PJ5MD0	PJ4MD1	PJ4MD0
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
	PJ3MD1	PJ3MD0	PJ2MD1	PJ2MD0	PJ1MD1	PJ1MD0	PJ0MD1	PJ0MD0
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—PJ7 Mode 1 and 0 (PJ7MD1, PJ7MD0)
Bits 13 and 12—PJ6 Mode 1 and 0 (PJ6MD1, PJ6MD0)
Bits 11 and 10—PJ5 Mode 1 and 0 (PJ5MD1, PJ5MD0)
Bits 9 and 8—PJ4 Mode 1 and 0 (PJ4MD1, PJ4MD0)
Bits 7 and 6—PJ3 Mode 1 and 0 (PJ3MD1, PJ3MD0)
Bits 5 and 4—PJ2 Mode 1 and 0 (PJ2MD1, PJ2MD0)
Bits 3 and 2—PJ1 Mode 1 and 0 (PJ1MD1, PJ1MD0)
Bits 1 and 0—PJ0 Mode 1 and 0 (PJ0MD1, PJ0MD0)
These bits select the pin function and control MOS input pull-up.

Bit (2n + 1): PJnMD1	Bit 2n: PJnMD0	Pin Function	
0	0	Other function	(Initial value)
	1	Port output	
1	0	Port input (MOS pull-up on)	(n = 0, 2, 6, 7)
	1	Port input (MOS pull-up off)	(n = 0, 2, 6, 7)
	—*	Port input (MOS pull-up off)	(n = 3)

(n = 0, 2, 3, 6, 7)

Note: * 0 or 1

Bit (2n + 1): PJnMD1	Bit 2n: PJnMD0	Pin Function	
0	0	NF*2	(Initial value)
	1	Port output	
1	—*1	Reserved*3	

(n = 1, 4, 5)

Notes: *1 0 or 1

*2 The “port output” setting must be selected when this port is used. When not used as a port (when designated as NF), the pin must be left open.

*3 Operation is not guaranteed if a reserved setting is selected.

21.3.10 Port K Control Register (PKCR)

PKCR is a 16-bit readable/writable register that selects port K pin functions. PKCR 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
	PK7MD1	PK7MD0	PK6MD1	PK6MD0	PK5MD1	PK5MD0	PK4MD1	PK4MD0
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
	PK3MD1	PK3MD0	PK2MD1	PK2MD0	PK1MD1	PK1MD0	PK0MD1	PK0MD0
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—PK7 Mode 1 and 0 (PK7MD1, PK7MD0)
Bits 13 and 12—PK6 Mode 1 and 0 (PK6MD1, PK6MD0)
Bits 11 and 10—PK5 Mode 1 and 0 (PK5MD1, PK5MD0)
Bits 9 and 8—PK4 Mode 1 and 0 (PK4MD1, PK4MD0)
Bits 7 and 6—PK3 Mode 1 and 0 (PK3MD1, PK3MD0)
Bits 5 and 4—PK2 Mode 1 and 0 (PK2MD1, PK2MD0)
Bits 3 and 2—PK1 Mode 1 and 0 (PK1MD1, PK1MD0)
Bits 1 and 0—PK0 Mode 1 and 0 (PK0MD1, PK0MD0)

These bits select the pin function and control MOS input pull-up.

Bit (2n + 1): PKnMD1	Bit 2n: PKnMD0	Pin Function	
0	0	Other function	(Initial value)
	1	Port output	
1	0	Port input (MOS pull-up on)	
	1	Port input (MOS pull-up off)	

(n = 0 to 7)

21.3.11 Port L Control Register (PLCR)

PLCR is a 16-bit readable/writable register that selects port L pin functions. PLCR is initialized to H'AA00 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
	PL7MD1	PL7MD0	PL6MD1	PL6MD0	PL5MD1	PL5MD0	PL4MD1	PL4MD0
Initial value:	1	0	1	0	1	0	1	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit:	7	6	5	4	3	2	1	0
	PL3MD1	PL3MD0	PL2MD1	PL2MD0	PL1MD1	PL1MD0	PL0MD1	PL0MD0
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—PL7 Mode 1 and 0 (PL7MD1, PL7MD0)
Bits 13 and 12—PL6 Mode 1 and 0 (PL6MD1, PL6MD0)
Bits 11 and 10—PL5 Mode 1 and 0 (PL5MD1, PL5MD0)
Bits 9 and 8—PL4 Mode 1 and 0 (PL4MD1, PL4MD0)
Bits 7 and 6—PL3 Mode 1 and 0 (PL3MD1, PL3MD0)
Bits 5 and 4—PL2 Mode 1 and 0 (PL2MD1, PL2MD0)
Bits 3 and 2—PL1 Mode 1 and 0 (PL1MD1, PL1MD0)
Bits 1 and 0—PL0 Mode 1 and 0 (PL0MD1, PL0MD0)

These bits select the pin function.

Bit (2n + 1): PLnMD1	Bit 2n: PLnMD0	Pin Function	
0	0	Other function	(Initial value)
	1	Reserved*2	
1	—*1	Port input (MOS pull-up off)	

(n = 0 to 3)

Notes: *1 0 or 1
*2 Operation is not guaranteed if a reserved setting is selected.

Bit (2n + 1): PLnMD1	Bit 2n: PLnMD0	Pin Function	
0	0	Reserved*2	
	1		
1	—*1	Port input (MOS pull-up off)	(Initial value)

(n = 4 to 7)

Notes: *1 0 or 1
*2 Operation is not guaranteed if a reserved setting is selected.

21.3.12 SC Port Control Register (SCPCR)

SCPCR is a 16-bit readable/writable register that selects SC port pin functions. SCPCR settings are valid only when transmit/receive operations are disabled by settings in the SCSCR register. SCPCR is initialized to H'A888 by a power-on reset, but is not initialized by a manual reset or in standby mode.

When the TE bit in SCSCR is set to 1, the “Other function” output state has priority over the SCPCR settings for pins TxD[2:0].

When the RE bit in SCSCR is set to 1, the input state has priority over the SCPCR settings for pins RxD[2:0].

Bit:	15	14	13	12	11	10	9	8
	SCP7MD1	SCP7MD0	SCP6MD1	SCP6MD0	SCP5MD1	SCP5MD0	SCP4MD1	SCP4MD0
Initial value:	1	0	1	0	1	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
	SCP3MD1	SCP3MD0	SCP2MD1	SCP2MD0	SCP1MD1	SCP1MD0	SCP0MD1	SCP0MD0
Initial value:	1	0	0	0	1	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 15 and 14—SCP7 Mode 1 and 0 (SCP7MD1, SCP7MD0)
Bits 13 and 12—SCP6 Mode 1 and 0 (SCP6MD1, SCP6MD0)
Bits 11 and 10—SCP5 Mode 1 and 0 (SCP5MD1, SCP5MD0)
Bits 9 and 8—SCP4 Mode 1 and 0 (SCP4MD1, SCP4MD0)
Bits 7 and 6—SCP3 Mode 1 and 0 (SCP3MD1, SCP3MD0)
Bits 5 and 4—SCP2 Mode 1 and 0 (SCP2MD1, SCP2MD0)
Bits 3 and 2—SCP1 Mode 1 and 0 (SCP1MD1, SCP1MD0)
Bits 1 and 0—SCP0 Mode 1 and 0 (SCP0MD1, SCP0MD0)
These bits select the pin function and control MOS input pull-up.

Bit (2n + 1): SCPnMD1	**Bit 2n: SCPnMD0**	**Pin Function**	
0	0	Other function	
	1	Reserved*	
1	0	Port input (MOS pull-up on)	(Initial value)
	1	Port input (MOS pull-up off)	

(n = 7)

Note: * Operation is not guaranteed if the reserved setting is selected.

Bit (2n + 1): SCPnMD1	**Bit 2n: SCPnMD0**	**Pin Function**	
0	0	Other function	
	1	Port output	
1	0	Port input (MOS pull-up on)	(Initial value)
	1	Port input (MOS pull-up off)	

(n = 1, 3, 5)

Bit (2n + 1): SCPnMD1	**Bit 2n: SCPnMD0**	**Pin Function**	
0	0	Reserved*	
	1	Port output	
1	0	Port input (MOS pull-up on)	(Initial value)
	1	Port input (MOS pull-up off)	

(n = 6)

Note: * Operation is not guaranteed if the reserved setting is selected.

TxD Pin

Bit (2n + 1): SCPnMD1	Bit 2n: SCPnMD0	Pin Function	
0	0	Other (TxD pin) function	(Initial value)
	1	Port output	
1	—*	Output high impedance	

(n = 0, 2, 4)

Note: * 0 or 1

RxD Pin

Bit (2n + 1): SCPnMD1	Bit 2n: SCPnMD0	Pin Function	
0	0	Other (RxD pin) function	(Initial value)
	1	Input (input level ignored)	
1	0	Port input (MOS pull-up on)*	
	1	Port input (MOS pull-up off)*	

(n = 0, 2, 4)

Notes: As one bit (SCP4DT) is accessed using two pins, TxD2 and RxD2, there is no SCPT4 simultaneous input/output combination.

* When SCSCR[2–0] bits [RE, TE] are set to 1, RxD[2–0] and TxD[2–0] input/output is performed regardless of the SCPCR settings.

When "port input" or "other function" is set, the TxD pin will go to the output state when the TE bit in SCSCR is set to 1, and to the high-impedance state when the TE bit is cleared to 0.

Section 22 I/O Ports

22.1 Overview

The SH7622 has twelve 8-bit ports (ports A to L and SC). All the pins in each port are multiplexed as port pins and special function pins. Pin function selection and MOS pull-up control is performed by means of the pin function controller (PFC). Each port is provided with a data register for storing the pin data.

22.2 Port A

Port A is an 8-bit input/output port with the pin configuration shown in figure 22.1. Each pin is provided with a MOS input pull-up, controlled by the port A control register (PACR) in the PFC.

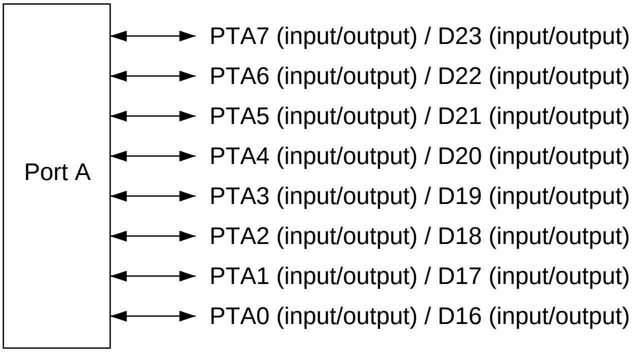

Port A
PTA7 (input/output) / D23 (input/output)
PTA6 (input/output) / D22 (input/output)
PTA5 (input/output) / D21 (input/output)
PTA4 (input/output) / D20 (input/output)
PTA3 (input/output) / D19 (input/output)
PTA2 (input/output) / D18 (input/output)
PTA1 (input/output) / D17 (input/output)
PTA0 (input/output) / D16 (input/output)

Figure 22.1 Port A

22.2.1 Register Description

Table 22.1 summarizes the port A register.

Table 22.1 Port A Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port A data register	PADR	R/W	H'00	H'A4000120	8

22.2.2 Port A Data Register (PADR)

PADR is an 8-bit readable/writable register that stores data for pins PTA7–PTA0. Bits PA7DT to PA0DT correspond to pins PTA7–PTA0. When a pin functions as a general output port, if port A is read the value of the corresponding PADR bit is read directly. When a pin functions as a general input port, if port A is read the corresponding pin level is read. Table 22.2 summarizes the functions of PADR.

PADR is initialized to H'00 by a power-on reset. In a manual reset and in standby mode it retains its contents.

Bit:	7	6	5	4	3	2	1	0
	PA7DT	PA6DT	PA5DT	PA4DT	PA3DT	PA2DT	PA1DT	PA0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22.2 Port A Data Register (PADR) Read/Write Operations

PAnMD1	PAnMD0	Pin State	Read	Write
0	0	Other function	PADR value	Value can be written to PADR, but does not affect pin state
	1	Output	PADR value	Write value is output from pin
1	0	Input (MOS pull-up on)	Pin state	Value can be written to PADR, but does not affect pin state
	1	Input (MOS pull-up off)	Pin state	Value can be written to PADR, but does not affect pin state

(n = 0 to 7)

22.3 Port B

Port B is an 8-bit input/output port with the pin configuration shown in figure 22.2. Each pin is provided with a MOS input pull-up, controlled by the port B control register (PBCR) in the PFC.

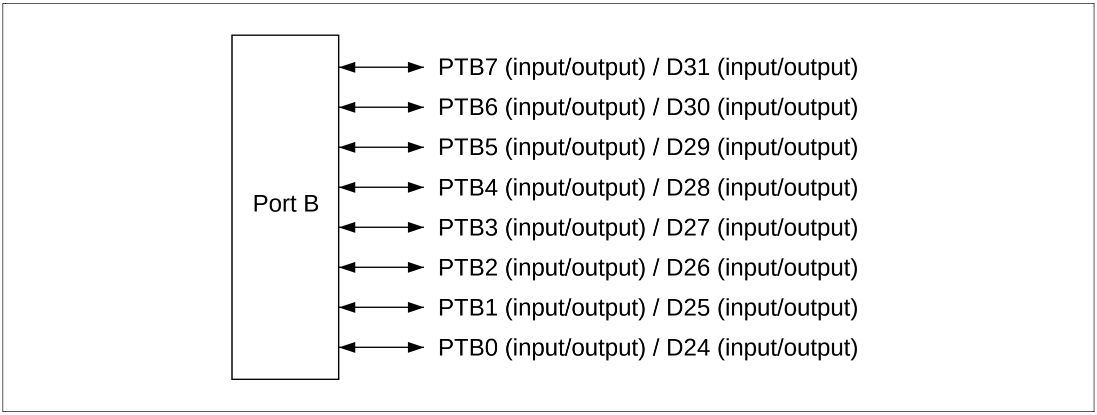

Port B
PTB7 (input/output) / D31 (input/output)
PTB6 (input/output) / D30 (input/output)
PTB5 (input/output) / D29 (input/output)
PTB4 (input/output) / D28 (input/output)
PTB3 (input/output) / D27 (input/output)
PTB2 (input/output) / D26 (input/output)
PTB1 (input/output) / D25 (input/output)
PTB0 (input/output) / D24 (input/output)

Figure 22.2 Port B

22.3.1 Register Description

Table 22.3 summarizes the port B register.

Table 22.3 Port B Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port B data register	PBDR	R/W	H'00	H'A4000122	8

22.3.2 Port B Data Register (PBDR)

PBDR is an 8-bit readable/writable register that stores data for pins PTB7–PTB0. Bits PB7DT–PB0DT correspond to pins PTB7–PTB0. When a pin functions as a general output port, if port B is read the value of the corresponding PBDR bit is read directly. When a pin functions as a general input port, if port B is read the corresponding pin level is read. Table 22.4 summarizes the functions of PBDR.

PBDR is initialized to H'00 by a power-on reset. In a manual reset and in standby mode it retains its contents.

Bit:	7	6	5	4	3	2	1	0
	PB7DT	PB6DT	PB5DT	PB4DT	PB3DT	PB2DT	PB1DT	PB0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22.4 Port B Data Register (PBDR) Read/Write Operations

PBnMD1	PBnMD0	Pin State	Read	Write
0	0	Other function	PBDR value	Value can be written to PBDR, but does not affect pin state
	1	Output	PBDR value	Write value is output from pin
1	0	Input (MOS pull-up on)	Pin state	Value can be written to PBDR, but does not affect pin state
	1	Input (MOS pull-up off)	Pin state	Value can be written to PBDR, but does not affect pin state

(n = 0 to 7)

22.4 Port C

Port C is a 4-bit input/output port, 3-bit output port, and 1-bit input port with the pin configuration shown in figure 22.3. Each pin is provided with a MOS input pull-up, controlled by the port C control register (PCCR) in the PFC.

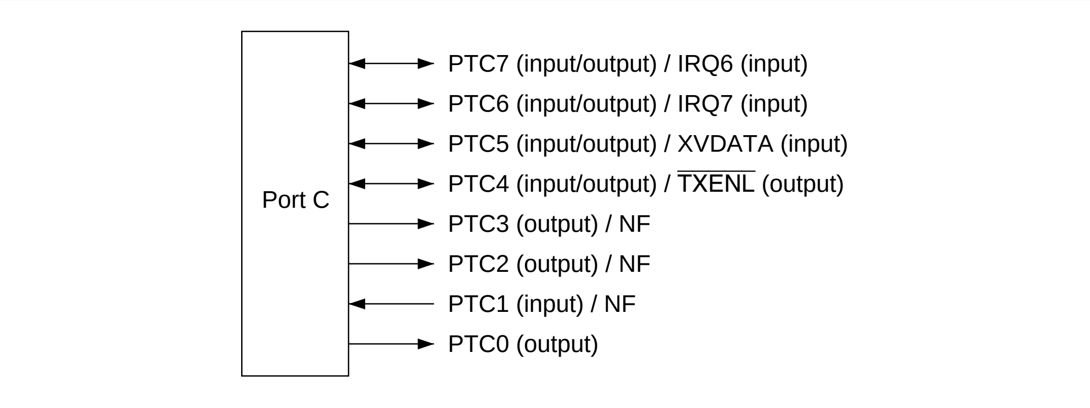

Port C
PTC7 (input/output) / IRQ6 (input)
PTC6 (input/output) / IRQ7 (input)
PTC5 (input/output) / XVDATA (input)
PTC4 (input/output) / $\overline{\text{TXENL}}$ (output)
PTC3 (output) / NF
PTC2 (output) / NF
PTC1 (input) / NF
PTC0 (output)

Figure 22.3 Port C

22.4.1 Register Description

Table 22.5 summarizes the port C register.

Table 22.5 Port C Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port C data register	PCDR	R/W	H'00	H'A4000124	8

22.4.2 Port C Data Register (PCDR)

PCDR is an 8-bit readable/writable register that stores data for pins PTC7–PTC0. Bits PC7DT–PC0DT correspond to pins PTC7–PTC0. When a pin functions as a general output port, if port C is read the value of the corresponding PCDR bit is read directly. When a pin functions as a general input port, if port C is read the corresponding pin level is read. Table 22.6 summarizes the functions of PCDR.

PCDR is initialized to H'00 by a power-on reset. For PTC[7:4], the initial pin function is general input port (with MOS pull-up on), and the corresponding pin level is read. In a manual reset and in standby mode, PCDR retains its contents.

Bit:	7	6	5	4	3	2	1	0
	PC7DT	PC6DT	PC5DT	PC4DT	PC3DT	PC2DT	PC1DT	PC0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22.6 Port C Data Register (PCDR) Read/Write Operations

PCnMD1	PCnMD0	Pin State	Read	Write
0	0	NF	PCDR value	Value can be written to PCDR, but does not affect pin state
	1	Output	PCDR value	Write value is output from pin
1	—*1	Reserved*2	Low level	Ignored (does not affect pin state)

(n = 0, 2, 3)

Notes: *1 0 or 1
*2 Operation is not guaranteed if the reserved setting is selected.

PCnMD1	PCnMD0	Pin State	Read	Write
0	0	NF	PCDR value	Value can be written to PCDR, but does not affect pin state
	1	Reserved*	Low level	Ignored (does not affect pin state)
1	0	Input (MOS pull-up on)	Pin state	Value can be written to PCDR, but does not affect pin state
	1	Input (MOS pull-up off)	Pin state	Value can be written to PCDR, but does not affect pin state

(n = 1)

Note: * Operation is not guaranteed if the reserved setting is selected.

PCnMD1	PCnMD0	Pin State	Read	Write
0	0	Other function	PCDR value	Value can be written to PCDR, but does not affect pin state
	1	Output	PCDR value	Write value is output from pin
1	0	Input (MOS pull-up on)	Pin state	Value can be written to PCDR, but does not affect pin state
	1	Input (MOS pull-up off)	Pin state	Value can be written to PCDR, but does not affect pin state

(n = 4 to 7)

22.5 Port D

Port D comprises a 6-bit input/output port and 2-bit input port with the pin configuration shown in figure 22.4. Each pin is provided with a MOS input pull-up, controlled by the port D control register (PDCR) in the PFC.

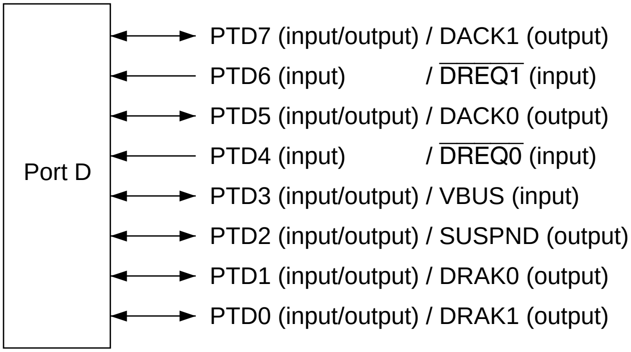

Port D
PTD7 (input/output) / DACK1 (output)
PTD6 (input) / $\overline{\text{DREQ1}}$ (input)
PTD5 (input/output) / DACK0 (output)
PTD4 (input) / $\overline{\text{DREQ0}}$ (input)
PTD3 (input/output) / VBUS (input)
PTD2 (input/output) / SUSPND (output)
PTD1 (input/output) / DRAK0 (output)
PTD0 (input/output) / DRAK1 (output)

Figure 22.4 Port D

22.5.1 Register Description

Table 22.7 summarizes the port D register.

Table 22.7 Port D Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port D data register	PDDR	R/W or R	H'00	H'A4000126	8

22.5.2 Port D Data Register (PDDR)

PDDR is an 8-bit register, comprising 6 readable/writable bits and 2 readable bits, that stores data for pins PTD7–PTD0. Bits PD7DT–PD0DT correspond to pins PTD7–PTD0. When a pin functions as a general output port, if port D is read the value of the corresponding PDDR bit is read directly. When a pin functions as a general input port, if port D is read the corresponding pin level is read. Table 22.8 summarizes the functions of PDDR.

PDDR is initialized to H'00 by a power-on reset. In a manual reset and in standby mode it retains its contents.

Note that a low level will be read if bit 6 or 4 is read when the general input function has not been selected.

Bit:	7	6	5	4	3	2	1	0
	PD7DT	PD6DT	PD5DT	PD4DT	PD3DT	PD2DT	PD1DT	PD0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R/W	R	R/W	R/W	R/W	R/W

Table 22.8 Port D Data Register (PDDR) Read/Write Operations

PDnMD1	PDnMD0	Pin State	Read	Write
0	0	Other function	PDDR value	Value can be written to PDDR, but does not affect pin state
	1	Output	PDDR value	Write value is output from pin
1	0 (n = 0 to 3)	Input (MOS pull-up on)	Pin state	Value can be written to PDDR, but does not affect pin state
	1 (n = 0 to 3)	Input (MOS pull-up off)	Pin state	Value can be written to PDDR, but does not affect pin state
	—* (n = 5, 7)	Input (MOS pull-up off)	Pin state	Value can be written to PDDR, but does not affect pin state

(n = 0 to 3, 5, 7)

Note: * 0 or 1

PDnMD1	PDnMD0	Pin State	Read	Write
0	0	Other function	Low level	Ignored (does not affect pin state)
	1	Reserved*	Low level	Ignored (does not affect pin state)
1	0	Input (MOS pull-up on)	Pin state	Ignored (does not affect pin state)
	1	Input (MOS pull-up off)	Pin state	Ignored (does not affect pin state)

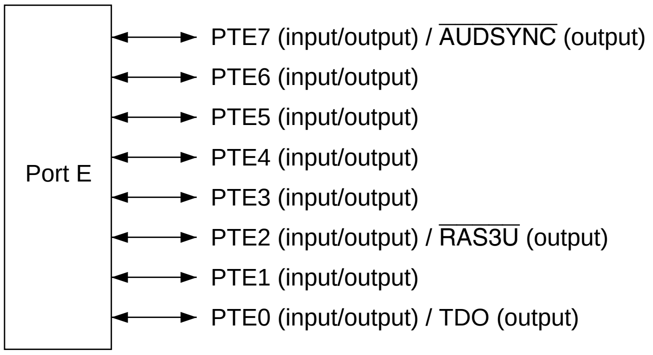
(n = 4, 6)

Note: * Operation is not guaranteed if the reserved setting is selected.

Bits 3 to 0—Reserved: These bits are always read as 0. The write value should always be 0.

22.6 Port E

Port E is an 8-bit input/output port with the pin configuration shown in figure 22.5. Each pin is provided with a MOS input pull-up, controlled by the port E control register (PECR) in the PFC.



Port E
PTE7 (input/output) / $\overline{\text{AUDSYNC}}$ (output)
PTE6 (input/output)
PTE5 (input/output)
PTE4 (input/output)
PTE3 (input/output)
PTE2 (input/output) / $\overline{\text{RAS3U}}$ (output)
PTE1 (input/output)
PTE0 (input/output) / TDO (output)

Figure 22.5 Port E

22.6.1 Register Description

Table 22.9 summarizes the port E register.

Table 22.9 Port E Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port E data register	PEDR	R/W	H'00	H'A4000128	8

22.6.2 Port E Data Register (PEDR)

PEDR is an 8-bit readable/writable register that stores data for pins PTE7–PTE0. Bits PE7DT–PE0DT correspond to pins PTE7–PTE0. When a pin functions as a general output port, if port E is read the value of the corresponding PEDR bit is read directly. When a pin functions as a general input port, if port E is read the corresponding pin level is read. Table 22.10 summarizes the functions of PEDR.

PEDR is initialized to H'00 by a power-on reset, after which the initial pin function is general input port (with MOS pull-up on for PTE[0,7] and off for PTE[1–6]), and the corresponding pin level is read. In a manual reset and in standby mode, PEDR retains its contents.

Bit:	7	6	5	4	3	2	1	0
	PE7DT	PE6DT	PE5DT	PE4DT	PE3DT	PE2DT	PE1DT	PE0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22.10 Port E Data Register (PEDR) Read/Write Operations

PEnMD1	PEnMD0	Pin State	Read	Write
0	0	Other function	PEDR value	Value can be written to PEDR, but does not affect pin state
	1	Output	PEDR value	Write value is output from pin
1	0	Input (MOS pull-up on)	Pin state	Value can be written to PEDR, but does not affect pin state
	1	Input (MOS pull-up off)	Pin state	Value can be written to PEDR, but does not affect pin state

(n = 0, 7)

PEnMD1	PEnMD0	Pin State	Read	Write
0	0	Reserved*2	Low level	Ignored (does not affect pin state)
	1	Output	PEDR value	Write value is output from pin
1	—*1	Input (MOS pull-up off)	Pin state	Value can be written to PEDR, but does not affect pin state

(n = 1, 3 to 6)

Notes: *1 0 or 1

*2 Operation is not guaranteed if the reserved setting is selected.

PEnMD1	PEnMD0	Pin State	Read	Write
0	0	Other function	PEDR value	Value can be written to PEDR, but does not affect pin state
	1	Output	PEDR value	Write value is output from pin
1	—*	Input (MOS pull-up off)	Pin state	Value can be written to PEDR, but does not affect pin state

(n = 2)

Note: * 0 or 1

22.7 Port F

Port F is an 8-bit input port with the pin configuration shown in figure 22.6. Each pin is provided with a MOS input pull-up, controlled by the port F control register (PFCR) in the PFC.

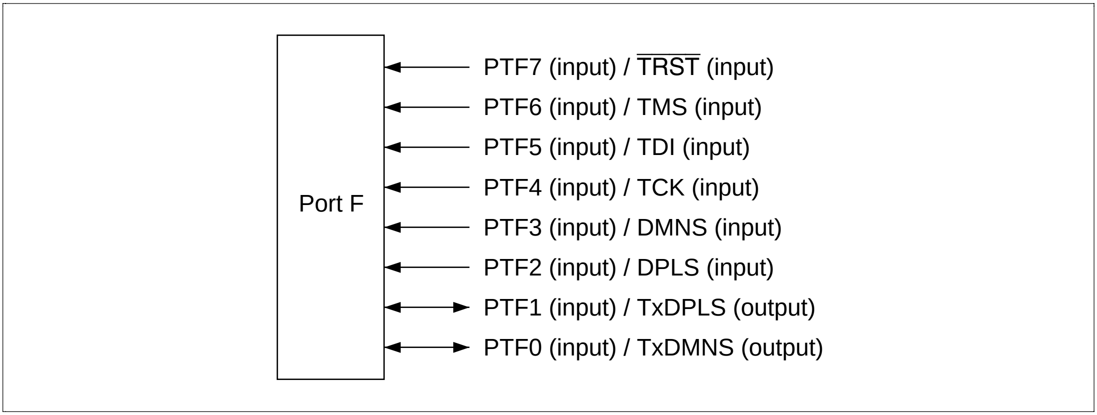

Port F
PTF7 (input) / $\overline{\text{TRST}}$ (input)
PTF6 (input) / TMS (input)
PTF5 (input) / TDI (input)
PTF4 (input) / TCK (input)
PTF3 (input) / DMNS (input)
PTF2 (input) / DPLS (input)
PTF1 (input) / TxDPLS (output)
PTF0 (input) / TxDMNS (output)

Figure 22.6 Port F

22.7.1 Register Description

Table 22.11 summarizes the port F register.

Table 22.11 Port F Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port F data register	PFDR	R	H'00	H'A400012A	8

22.7.2 Port F Data Register (PFDR)

PFDR is an 8-bit readable register that stores data for pins PTF7–PTF0. Bits PF7DT–PF0DT correspond to pins PTF7–PTF0. When a pin functions as a general input port, if port F is read the corresponding pin level is read. Table 22.12 summarizes the functions of PFDR.

Bit:	7	6	5	4	3	2	1	0
	PF7DT	PF6DT	PF5DT	PF4DT	PF3DT	PF2DT	PF1DT	PF0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Table 22.12 Port F Data Register (PFDR) Read/Write Operations

PFnMD1	PFnMD0	Pin State	Read	Write
0	0	Other function	H'00	Ignored (does not affect pin state)
	1	Reserved*	H'00	Ignored (does not affect pin state)
1	0	Input (MOS pull-up on)	Pin state	Ignored (does not affect pin state)
	1	Input (MOS pull-up off)	Pin state	Ignored (does not affect pin state)

(n = 0 to 7)

Note: * Operation is not guaranteed if the reserved setting is selected.

22.8 Port G

Port G comprises an 8-bit input port with the pin configuration shown in figure 22.7. Each pin is provided with a MOS input pull-up, controlled by the port G control register (PGCR) in the PGC.

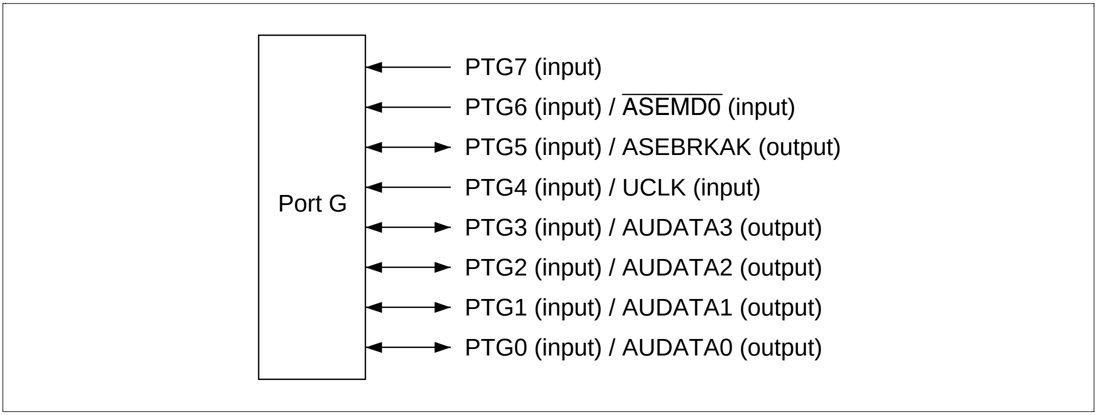

Port G
PTG7 (input)
PTG6 (input) / $\overline{\text{ASEMD0}}$ (input)
PTG5 (input) / ASEBRKAK (output)
PTG4 (input) / UCLK (input)
PTG3 (input) / AUDATA3 (output)
PTG2 (input) / AUDATA2 (output)
PTG1 (input) / AUDATA1 (output)
PTG0 (input) / AUDATA0 (output)

Figure 22.7 Port G

22.8.1 Register Description

Table 22.13 summarizes the port G register.

Table 22.13 Port G Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port G data register	PGDR	R/W	H'00	H'A400012C	8

22.8.2 Port G Data Register (PGDR)

PGDR is an 8-bit readable register that stores data for pins PTG7–PTG0. Bits PG7DT–PG0DT correspond to pins PTG7–PTG0. When a pin functions as a general input port, if port G is read the corresponding pin level is read. Table 22.14 summarizes the functions of PGDR.

Bit:	7	6	5	4	3	2	1	0
	PG7DT	PG6DT	PG5DT	PG4DT	PG3DT	PG2DT	PG1DT	PG0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Table 22.14 Port G Data Register (PGDR) Read/Write Operations

PGnMD1	PGnMD0	Pin State	Read	Write
0	0	Other function	H'00	Ignored (does not affect pin state)
	1	Reserved*	H'00	Ignored (does not affect pin state)
1	0	Input (MOS pull-up on)	Pin state	Ignored (does not affect pin state)
	1	Input (MOS pull-up off)	Pin state	Ignored (does not affect pin state)

(n = 0 to 7)

Note: * Operation is not guaranteed if the reserved setting is selected.

22.9 Port H

Port H comprises a 1-bit input/output port and 7-bit input port with the pin configuration shown in figure 22.8. Each pin is provided with a MOS input pull-up, controlled by the port H control register (PHCR) in the PFC.

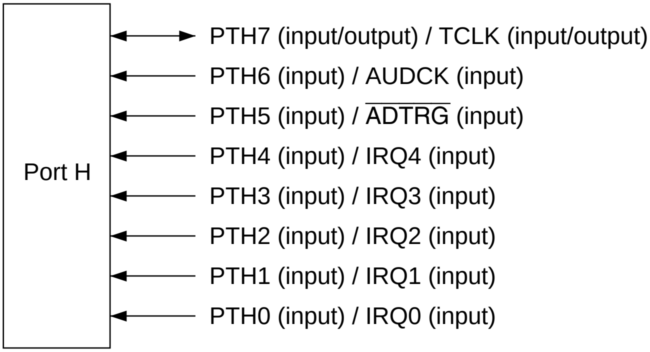

Port H
PTH7 (input/output) / TCLK (input/output)
PTH6 (input) / AUDCK (input)
PTH5 (input) / $\overline{\text{ADTRG}}$ (input)
PTH4 (input) / IRQ4 (input)
PTH3 (input) / IRQ3 (input)
PTH2 (input) / IRQ2 (input)
PTH1 (input) / IRQ1 (input)
PTH0 (input) / IRQ0 (input)

Figure 22.8 Port H

22.9.1 Register Description

Table 22.15 summarizes the port H register.

Table 22.15 Port H Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port H data register	PHDR	R/W or R	H'00	H'A400012E	8

22.9.2 Port H Data Register (PHDR)

PHDR is an 8-bit register, comprising 1 readable/writable bit and 7 readable bits, that stores data for pins PTH7–PTH0. Bits PH7DT–PH0DT correspond to pins PTH7–PTH0. When a pin functions as a general output port, if port H is read the value of the corresponding PHDR bit is read directly. When a pin functions as a general input port, if port H is read the corresponding pin level is read. Table 22.16 summarizes the functions of PHDR.

PHDR is initialized to H'00 by a power-on reset, after which the initial pin function is general input port (with MOS pull-up on), and the corresponding pin level is read. In a manual reset and in standby mode, PHDR retains its contents.

Note that a low level will be read if bits 6 to 0 are read when the general input function has not been selected.

Bit:	7	6	5	4	3	2	1	0
	PH7DT	PH6DT	PH5DT	PH4DT	PH3DT	PH2DT	PH1DT	PH0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R	R	R	R	R	R	R

Table 22.16 Port H Data Register (PHDR) Read/Write Operations

PHnMD1	PHnMD0	Pin State	Read	Write
0	0	Other function	PHDR value	Value can be written to PHDR, but does not affect pin state
	1	Output	PHDR value	Write value is output from pin
1	0	Input (MOS pull-up on)	Pin state	Value can be written to PHDR, but does not affect pin state
	1	Input (MOS pull-up off)	Pin state	Value can be written to PHDR, but does not affect pin state

(n = 7)

PHnMD1	PHnMD0	Pin State	Read	Write
0	0	Other function	Low level	Ignored (does not affect pin state)
	1	Reserved*	Low level	Ignored (does not affect pin state)
1	0	Input (MOS pull-up on)	Pin state	Ignored (does not affect pin state)
	1	Input (MOS pull-up off)	Pin state	Ignored (does not affect pin state)

(n = 0 to 6)

Note: * Operation is not guaranteed if the reserved setting is selected.

22.10 Port J

Port J comprises a 5-bit input/output port and 3-bit output port with the pin configuration shown in figure 22.9. Pins PTJ[0,2,6,7] are provided with a MOS input pull-up, controlled by the port J control register (PJCR) in the PFC.

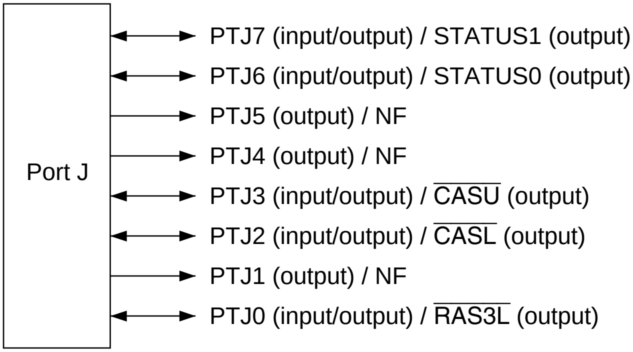

Port J
PTJ7 (input/output) / STATUS1 (output)
PTJ6 (input/output) / STATUS0 (output)
PTJ5 (output) / NF
PTJ4 (output) / NF
PTJ3 (input/output) / $\overline{\text{CASU}}$ (output)
PTJ2 (input/output) / $\overline{\text{CASL}}$ (output)
PTJ1 (output) / NF
PTJ0 (input/output) / $\overline{\text{RAS3L}}$ (output)

Figure 22.9 Port J

22.10.1 Register Description

Table 22.17 summarizes the port J register.

Table 22.17 Port J Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port J data register	PJDR	R/W	H'00	H'A4000130	8

22.10.2 Port J Data Register (PJDR)

PJDR is an 8-bit readable/writable register that stores data for pins PTJ7–PTJ0. Bits PJ7DT–PJ0DT correspond to pins PTJ7 to PTJ0. When a pin functions as a general output port, if port J is read the value of the corresponding PJDR bit is read directly. When a pin functions as a general input port, if port J is read the corresponding pin level is read. Table 22.18 summarizes the functions of PJDR.

PJDR retains its contents in a power-on or manual reset, and in standby mode.

Bit:	7	6	5	4	3	2	1	0
	PJ7DT	PJ6DT	PJ5DT	PJ4DT	PJ3DT	PJ2DT	PJ1DT	PJ0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22.18 Port J Data Register (PJDR) Read/Write Operations

PJnMD1	PJnMD0	Pin State	Read	Write
0	0	Other function	PJDR value	Value can be written to PJDR, but does not affect pin state
	1	Output	PJDR value	Write value is output from pin
1	0 (n = 0, 2, 6, 7)	Input (MOS pull-up on)	Pin state	Value can be written to PJDR, but does not affect pin state
	1 (n = 0, 2, 6, 7)	Input (MOS pull-up off)	Pin state	Value can be written to PJDR, but does not affect pin state
	—* (n = 3)			

(n = 0, 2, 3, 6, 7)

Note: * 0 or 1

PJnMD1	PJnMD0	Pin State	Read	Write
0	0	NF	PJDR value	Value can be written to PJDR, but does not affect pin state
	1	Output	PJDR value	Write value is output from pin
1	—*1	Reserved*2	Low level	Ignored (does not affect pin state)

(n = 1, 4, 5)

Notes: *1 0 or 1

*2 Operation is not guaranteed if the reserved setting is selected.

22.11 Port K

Port K is an 8-bit input/output port with the pin configuration shown in figure 22.10. Each pin is provided with a MOS input pull-up, controlled by the port K control register (PKCR) in the PFC.

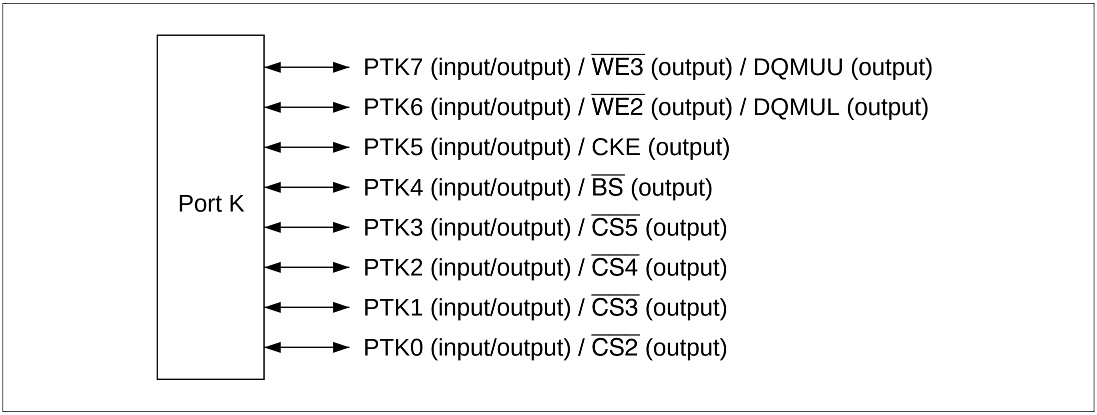

Port K
PTK7 (input/output) / $\overline{\text{WE3}}$ (output) / DQMUU (output)
PTK6 (input/output) / $\overline{\text{WE2}}$ (output) / DQMUL (output)
PTK5 (input/output) / CKE (output)
PTK4 (input/output) / $\overline{\text{BS}}$ (output)
PTK3 (input/output) / $\overline{\text{CS5}}$ (output)
PTK2 (input/output) / $\overline{\text{CS4}}$ (output)
PTK1 (input/output) / $\overline{\text{CS3}}$ (output)
PTK0 (input/output) / $\overline{\text{CS2}}$ (output)

Figure 22.10 Port K

22.11.1 Register Description

Table 22.19 summarizes the port K register.

Table 22.19 Port K Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port K data register	PKDR	R/W	H'00	H'A4000132	8

22.11.2 Port K Data Register (PKDR)

PKDR is an 8-bit readable/writable register that stores data for pins PTK7–PTK0. Bits PK7DT–PK0DT correspond to pins PTK7–PTK0. When a pin functions as a general output port, if port K is read the value of the corresponding PKDR bit is read directly. When a pin functions as a general input port, if port K is read the corresponding pin level is read. Table 22.20 summarizes the functions of PKDR.

PKDR is initialized to H'00 by a power-on reset. In a manual reset and in standby mode it retains its contents.

Bit:	7	6	5	4	3	2	1	0
	PD7DT	PD6DT	PD5DT	PD4DT	PD3DT	PD2DT	PD1DT	PD0DT
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 22.20 Port K Data Register (PKDR) Read/Write Operations

PKnMD1	PKnMD0	Pin State	Read	Write
0	0	Other function	PKDR value	Value can be written to PKDR, but does not affect pin state
	1	Output	PKDR value	Write value is output from pin
1	0	Input (MOS pull-up on)	Pin state	Value can be written to PKDR, but does not affect pin state
	1	Input (MOS pull-up off)	Pin state	Value can be written to PKDR, but does not affect pin state

(n = 0 to 7)

22.12 Port L

Port L is an 8-bit input port with the pin configuration shown in figure 22.11.

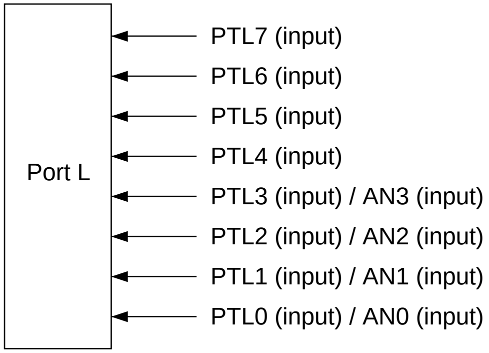

Port L
PTL7 (input)
PTL6 (input)
PTL5 (input)
PTL4 (input)
PTL3 (input) / AN3 (input)
PTL2 (input) / AN2 (input)
PTL1 (input) / AN1 (input)
PTL0 (input) / AN0 (input)

Figure 22.11 Port L

22.12.1 Register Description

Table 22.21 summarizes the port L register.

Table 22.21 Port L Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
Port L data register	PLDR	R	H'00	H'A4000134	8

22.12.2 Port L Data Register (PLDR)

PLDR is an 8-bit readable register that stores data for pins PTL7–PTL0. Bits PL7DT–PL0DT correspond to pins PTL7–PTL0. When a pin functions as a general input port, if port L is read the corresponding pin level is read. Table 22.22 summarizes the functions of PLDR.

PLDR is initialized to H'00 by a power-on reset. In a manual reset and in standby mode it retains its contents.

Bit:	7	6	5	4	3	2	1	0
	PL7DT	PL6DT	PL5DT	PL4DT	PL3DT	PL2DT	PL1DT	PL0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Table 22.22 Port L Data Register (PLDR) Read/Write Operations

PLnMD1	PLnMD0	Pin State	Read	Write
0	0	Other function	Low level	Ignored (does not affect pin state)
	1	Reserved*	Low level	Ignored (does not affect pin state)
1	0	Input (MOS pull-up on)	Pin state	Ignored (does not affect pin state)
	1	Input (MOS pull-up off)	Pin state	Ignored (does not affect pin state)

(n = 0 to 3)

Nots: * Operation is not guaranteed if a reserved setting is selected.

PLnMD1	PLnMD0	Pin State	Read	Write
0	—*1	Reserved*2	Low level	Ignored (does not affect pin state)
1	—*1	Input (MOS pull-up off)	Low level	Ignored (does not affect pin state)

(n = 4 to 7)

Notes: *1 1 or 0

*2 Operation is not guaranteed if a reserved setting is selected.

22.13 SC Port

The SC port comprises a 4-bit input/output port, 3-bit output port, and 4-bit input port with the pin configuration shown in figure 22.12. Each pin is provided with a MOS input pull-up, controlled by the SC port control register (SCPCR) in the PFC.

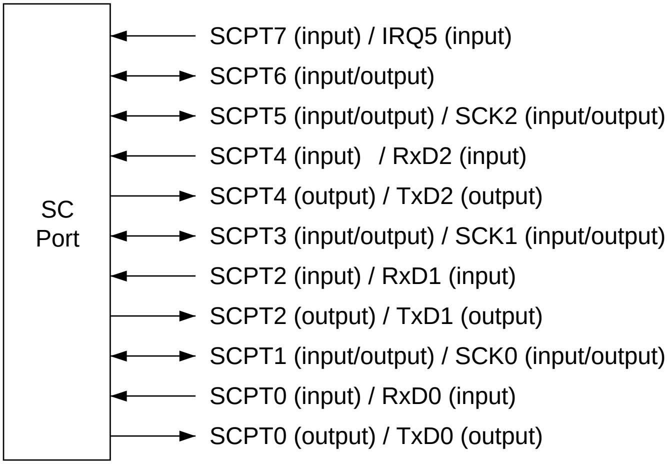

SC
Port
SCPT7 (input) / IRQ5 (input)
SCPT6 (input/output)
SCPT5 (input/output) / SCK2 (input/output)
SCPT4 (input) / RxD2 (input)
SCPT4 (output) / TxD2 (output)
SCPT3 (input/output) / SCK1 (input/output)
SCPT2 (input) / RxD1 (input)
SCPT2 (output) / TxD1 (output)
SCPT1 (input/output) / SCK0 (input/output)
SCPT0 (input) / RxD0 (input)
SCPT0 (output) / TxD0 (output)

Figure 22.12 SC Port

22.13.1 Register Description

Table 22.23 summarizes the SC port register.

Table 22.23 SC Port Register

Name	Abbreviation	R/W	Initial Value	Address	Access Size
SC port data register	SCPDR	R/W or R	H'00	H'A4000136	8

22.13.2 SC Port Data Register (SCPDR)

SCPDR is an 8-bit register, comprising 7 readable/writable bits and 1 readable bit, that stores data for pins SCPT7–SCPT0. Bits SCP7DT–SCP0DT correspond to pins SCPT7–SCPT0. When a pin functions as a general output port, if the SC port is read the value of the corresponding SCPDR bit is read directly. When a pin functions as a general input port, if the SC port is read the corresponding pin level is read. Table 22.24 summarizes the functions of SCPDR.

SCPDR is initialized by a power-on reset. In a manual reset and in standby mode it retains its contents.

Note that a low level will be read if bit 7 is read when the general input function has not been selected.

When reading the state of pins RxD2–RxD0 in bits SCP4DT, SCP2DT, and SCP0DT in SCPDTR without clearing the TE bit or RE bit in SCSCR to 0, the RE bit in SCSCR should be set to 1. When the RE bit is set to 1, the RxD pin becomes an input, and the pin state can be read, taking precedence over the SCPCR settings.

Bit:	7	6	5	4	3	2	1	0
	SCP7DT	SCP6DT	SCP5DT	SCP4DT	SCP3DT	SCP2DT	SCP1DT	SCP0DT
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22.24 SC Port Data Register (SCPDR) Read/Write Operations

SCPnMD1	SCPnMD0	Pin State	Read	Write
0	0	Reserved*	Low level	Ignored (does not affect pin state)
	1	Output	SCPDR value	Write value is output from pin
1	0	Input (MOS pull-up on)	Pin state	Value can be written to SCPDR, but does not affect pin state
	1	Input (MOS pull-up off)	Pin state	Value can be written to SCPDR, but does not affect pin state

(n = 6)

Note: * Operation is not guaranteed if a reserved setting is selected.

SCPnMD1	SCPnMD0	Pin State	Read	Write
0	0	Other function	Reading prohibited	Writes prohibited
	1	TxD: Output RxD: Input (input level ignored)	SCPDR value	Write value is output from TxD pin
1	0	TxD: Output high impedance RxD: Input (MOS pull-up on)	RxD pin state	Value can be written to SCPDR, but does not affect pin state
	1	TxD: Output high impedance RxD: Input (MOS pull-up off)	RxD pin state	Value can be written to SCPDR, but does not affect pin state

(n = 0, 2, 4)

SCPnMD1	SCPnMD0	Pin State	Read	Write
0	0	Other function	Reading prohibited	Writes prohibited
	1	Output	SCPDR value	Write value is output from pin
1	0	Input (MOS pull-up on)	Pin state	Value can be written to SCPDR, but does not affect pin state
	1	Input (MOS pull-up off)	Pin state	Value can be written to SCPDR, but does not affect pin state

(n = 1, 3, 5)

SCPnMD1	SCPnMD0	Pin State	Read	Write
0	0	Other function	Reading prohibited	Writes prohibited
	1	Reserved*	Low level	Ignored (does not affect pin state)
1	0	Input (MOS pull-up on)	Pin state	Ignored (does not affect pin state)
	1	Input (MOS pull-up off)	Pin state	Ignored (does not affect pin state)

(n = 7)

Note: * Operation is not guaranteed if a reserved setting is selected.

Section 23 A/D Converter

23.1 Overview

This LSI includes a 10-bit successive-approximation A/D converter with a selection of up to four analog input channels.

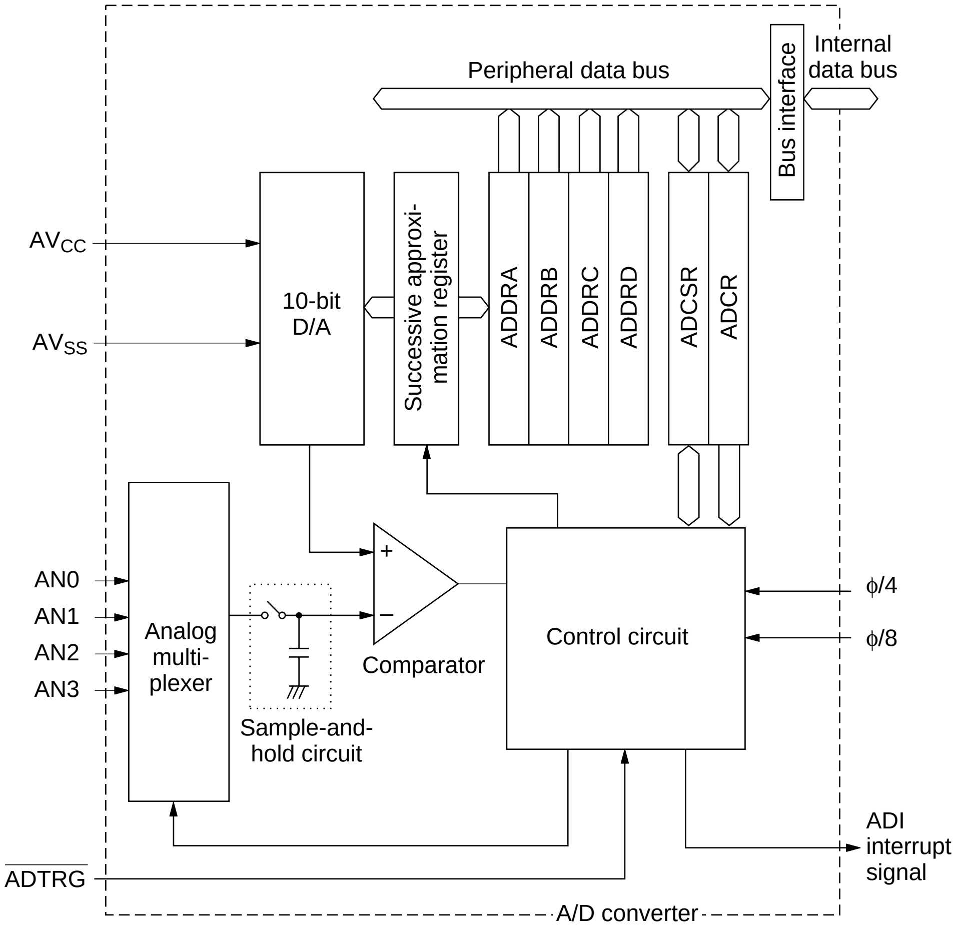
23.1.1 Features

A/D converter features are listed below.

- 10-bit resolution
- Four input channels
- High-speed conversion
  - Conversion time: maximum 8.9 µs per channel (with 15-MHz peripheral clock)
- Three conversion modes
  - Single mode: A/D conversion of one channel
  - Multi mode: A/D conversion on one to four channels
  - Scan mode: Continuous A/D conversion on one to four channels
- Four 16-bit data registers
  - A/D conversion results are transferred for storage into data registers corresponding to the channels.
- Sample-and-hold function
- A/D conversion can be externally triggered
- A/D interrupt requested at the end of conversion
  - At the end of A/D conversion, an A/D end interrupt (ADI) can be requested.

23.1.2 Block Diagram

Figure 23.1 shows a block diagram of the A/D converter.



Legend

ADCR: A/D control register

ADCSR: A/D control/status register

ADDRA: A/D data register A

ADDRB: A/D data register B

ADDRC: A/D data register C

ADDRD: A/D data register D

Figure 23.1 A/D Converter Block Diagram

23.1.3 Input Pins

Table 23.1 summarizes the A/D converter’s input pins.

Table 23.1 A/D Converter Pins

Pin Name	Abbreviation	I/O	Function
Analog power-supply pin	AVcc	Input	Analog power supply
Analog ground pin	AVss	Input	Analog ground and reference voltage
Analog input pin 0	AN0	Input	Analog inputs
Analog input pin 1	AN1	Input	
Analog input pin 2	AN2	Input	
Analog input pin 3	AN3	Input	
A/D external trigger input pin	$\overline{\text{ADTRG}}$	Input	External trigger input for starting A/D conversion

23.1.4 Register Configuration

Table 23.2 summarizes the A/D converter’s registers.

Table 23.2 A/D Converter Registers

Name	Abbreviation	R/W	Initial Value	Address
A/D data register A (high)	ADDRAH	R	H'00	H'A4000080
A/D data register A (low)	ADDRAL	R	H'00	H'A4000082
A/D data register B (high)	ADDRBH	R	H'00	H'A4000084
A/D data register B (low)	ADDRBL	R	H'00	H'A4000086
A/D data register C (high)	ADDRCH	R	H'00	H'A4000088
A/D data register C (low)	ADDRCL	R	H'00	H'A400008A
A/D data register D (high)	ADDRDH	R	H'00	H'A400008C
A/D data register D (low)	ADDRDL	R	H'00	H'A400008E
A/D control/status register	ADCSR	R/(W)*	H'00	H'A4000090
A/D control register	ADCR	R/W	H'07	H'A4000092

Note: * Only 0 can be written to bit 7, to clear the flag.

23.2 Register Descriptions

23.2.1 A/D Data Registers A–D (ADDRA–ADDRD)

The four A/D data registers (ADDRA–ADDRD) are 16-bit read-only registers that store the results of A/D conversion.

An A/D conversion produces 10-bit data, which is transferred for storage into the A/D data register corresponding to the selected channel. The upper 8 bits of the result are stored in the upper byte (bits 15–8) of the A/D data register. The lower 2 bits are stored in the lower byte (bits 7 and 6). Bits 5–0 of an A/D data register are reserved bits that always read 0. Table 23.3 indicates the pairings of analog input channels and A/D data registers.

The A/D data registers are initialized to H'0000 by a reset and in standby mode.

Upper register: H

Bit:	15	14	13	12	11	10	9	8
	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

Lower register: L

Bit:	7	6	5	4	3	2	1	0
	AD1	AD0	—	—	—	—	—	—
Initial value:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

n = A to D

Table 23.3 Analog Input Channels and A/D Data Registers

Analog Input Channel	A/D Data Register
AN0	ADDRA
AN1	ADDRB
AN2	ADDRC
AN3	ADDRD

23.2.2 A/D Control/Status Register (ADCSR)

ADCSR is an 8-bit read/write register that selects the mode and controls the A/D converter. ADCSR is initialized to H'00 by a reset and in standby mode.

Bit:	7	6	5	4	3	2	1	0
	ADF	ADIE	ADST	MULTI	CKS	—	CH1	CH0
Initial value:	0	0	0	0	0	0	0	0
R/W:	R/(W)*	R/W	R/W	R/W	R/W	R	R/W	R/W

Note: * Write 0 to clear the flag.

Bit 7—A/D End Flag (ADF): Indicates the end of A/D conversion.

Bit 7: ADF	Description
0	[Clear condition] (Initial value) • Cleared by reading ADF while ADF = 1, then writing 0 in ADF • Cleared when DMAC is activated by ADI interrupt and ADDR is read
1	[Set conditions] Single mode: A/D conversion ends Multi mode: A/D conversion ends in all selected channels

Bit 6—A/D Interrupt Enable (ADIE): Enables or disables the interrupt (ADI) requested at the end of A/D conversion.

Bit 6: ADIE	Description
0	A/D end interrupt request (ADI) is disabled (Initial value)
1	A/D end interrupt request (ADI) is enabled

Bit 5—A/D Start (ADST): Starts or stops A/D conversion. The ADST bit remains set to 1 during A/D conversion. It can also be set to 1 by external trigger input at the $\overline{\text{ADTRG}}$ pin.

Bit 5: ADST	Description
0	A/D conversion is stopped (Initial value)
1	Single mode: A/D conversion starts; ADST is automatically cleared to 0 when conversion ends. Multi mode: A/D conversion starts; ADST is automatically cleared to 0 when conversion ends after a circuit of all the specified channels. Scan mode: A/D conversion starts and continues, cycling among the selected channels, until ADST is cleared to 0 by software, by a reset, or by a transition to standby mode.

Bit 4—Multi Mode (MULTI): Selects single mode, multi mode, or scan mode. For further information on operation in these modes, see section 23.4, Operation.

	ADCR		
Bit 4: MULTI	**Bit 5: SCN**	**Description**	
0	0	Single mode	(Initial value)
	1		
1	0	Multi mode	
	1	Scan mode	

Bit 3—Clock Select (CKS): Selects the A/D conversion time. Clear the ADST bit to 0 before switching the conversion time.

Bit 3:CKS	Description	
0	Conversion time = 266 states (maximum)	(Initial value)
1	Conversion time = 134 states (maximum)	

Bit 2—Reserved: This bit is always read as 0. The write value should always be 0.

Bits 1 and 0—Channel Select 2–0 (CH2–CH0): These bits and the MULTI bit select the analog input channels. Clear the ADST bit to 0 before changing the channel selection.

Channel Selection		Description	
CH1	**CH0**	**Single Mode (MULTI = 0)**	**Multi Mode (MULTI = 1)**
0	0	AN0 (Initial value)	AN0
	1	AN1	AN0, AN1
1	0	AN2	AN0 to AN2
	1	AN3	AN0 to AN3

23.2.3 A/D Control Register (ADCR)

ADCR is an 8-bit read/write register that enables or disables external triggering of A/D conversion. ADCR is initialized to H'07 by a reset and in standby mode.

Bit:	7	6	5	4	3	2	1	0
	TRGE1	TRGE0	SCN	RESVD1	RESVD2	—	—	—
Initial value:	0	0	0	0	0	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R	R	R

Bit 7 and 6—Trigger Enable (TRGE1, TRGE0): Enables or disables external triggering of A/D conversion.

The TRGE1 and TRGE0 bits should only be set when conversion is not in progress.

Bit 7: TRGE1	Bit 6: TRGE0	Description
0	0	When an external trigger is input, the A/D conversion does not start (Initial value)
0	1	
1	0	
1	1	The A/D conversion starts at the falling edge of an input signal from the external trigger pin ($\overline{\text{ADTRG}}$).

Bit 5—Scan Mode (SCN): Selects multi mode or scan mode when the MULTI bit is set to 1. See the description of bit 4 in section 23.2.2, A/D Control/Status Register (ADCSR).

Bits 4 and 3—Reserved (RESVD1, RESVD2): These bits are always read as 0. The write value should always be 0.

Bits 2–0—Reserved: These bits are always read as 1. The write value should always be 1.

23.3 Bus Master Interface

ADDRA to ADDRD are 16-bit registers, but they are connected to the bus master by the upper 8 bits of the 16-bit peripheral data bus. Therefore, although the upper byte can be accessed directly by the bus master, the lower byte is read through an 8-bit temporary register (TEMP).

An A/D data register is read as follows. When the upper byte is read, the upper-byte value is transferred directly to the bus master and the lower-byte value is transferred into TEMP. Next, when the lower byte is read, the TEMP contents are transferred to the bus master.

When reading an A/D data register, always read the upper byte before the lower byte. It is possible to read only the upper byte, but if only the lower byte is read, the read value is not guaranteed.

Figure 23.2 shows the data flow for access to an A/D data register.

See section 23.7.3, Access Size and Read Data.

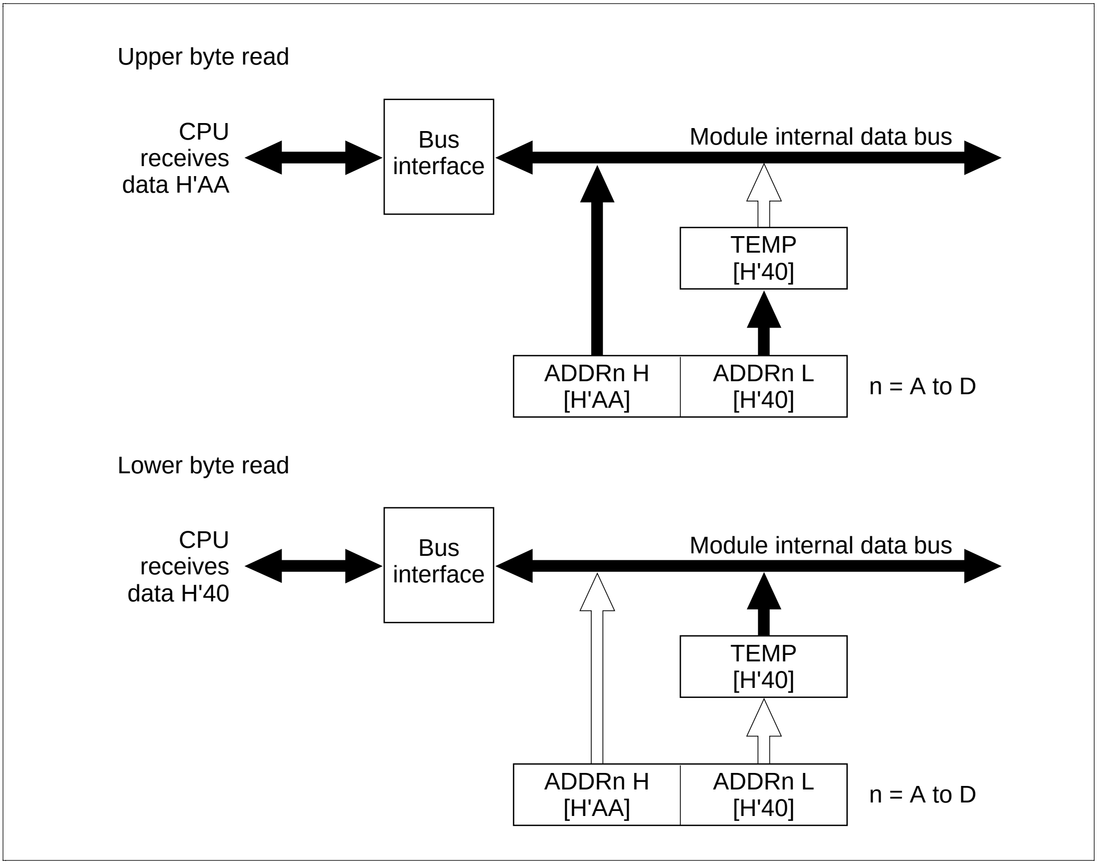


Figure 23.2 A/D Data Register Access Operation (Reading H'AA40)

23.4 Operation

The A/D converter operates by successive approximations with 10-bit resolution. It has two operating modes: single mode and scan mode.

23.4.1 Single Mode (MULTI = 0)

Single mode should be selected when only one A/D conversion on one channel is required. A/D conversion starts when the ADST bit in A/D control/status register (ADCSR) is set to 1 by software, or by external trigger input. The ADST bit remains set to 1 during A/D conversion and is automatically cleared to 0 when conversion ends.

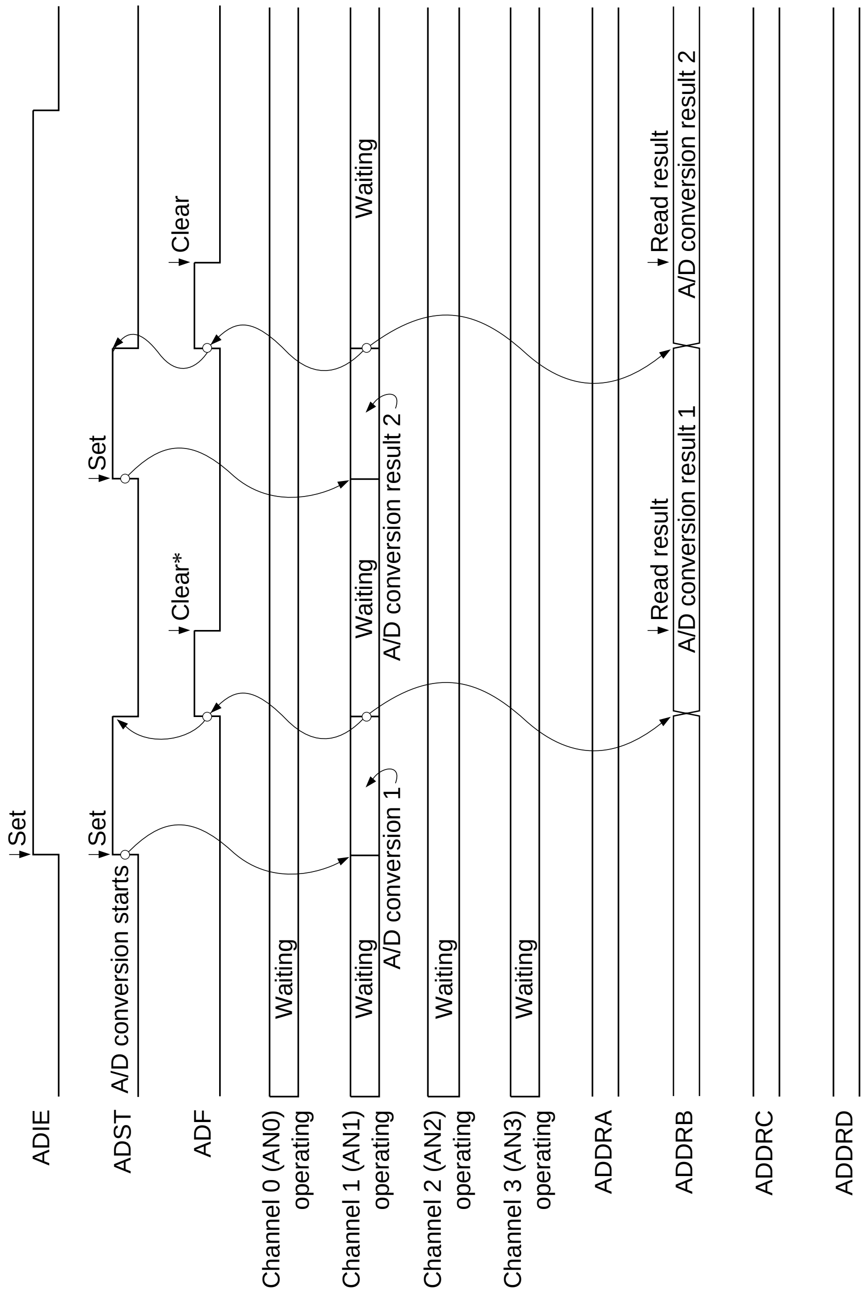
When conversion ends the ADF bit is set to 1. If the ADIE bit is also set to 1, an ADI interrupt is requested at this time. To clear the ADF flag to 0, first read ADCSR, then write 0 in ADF.

When the mode or analog input channel must be switched during A/D conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1 to start A/D conversion again. The ADST bit can be set at the same time as the mode or channel is changed.

Typical operations when channel 1 (AN1) is selected in single mode are described next.

Figure 23.3 shows a timing diagram for this example.

1. Single mode is selected (MULTI = 0), input channel AN1 is selected (CH2 = CH1 = 0, CH0 = 1), the A/D interrupt is enabled (ADIE = 1), and A/D conversion is started (ADST = 1).
2. When A/D conversion is completed, the result is transferred into ADDRB. At the same time the ADF flag is set to 1, the ADST bit is cleared to 0, and the A/D converter becomes idle.
3. Since ADF = 1 and ADIE = 1, an ADI interrupt is requested.
4. The A/D interrupt processing routine starts.
5. The routine reads ADCSR, then writes 0 in the ADF flag.
6. The routine reads and processes the conversion result (ADDRB = 0).
7. Execution of the A/D interrupt processing routine ends. Then, when the ADST bit is set to 1, A/D conversion starts to execute 2 to 7 above.



Note: * Downward arrows (↓) indicate instruction execution.

Figure 23.3 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)

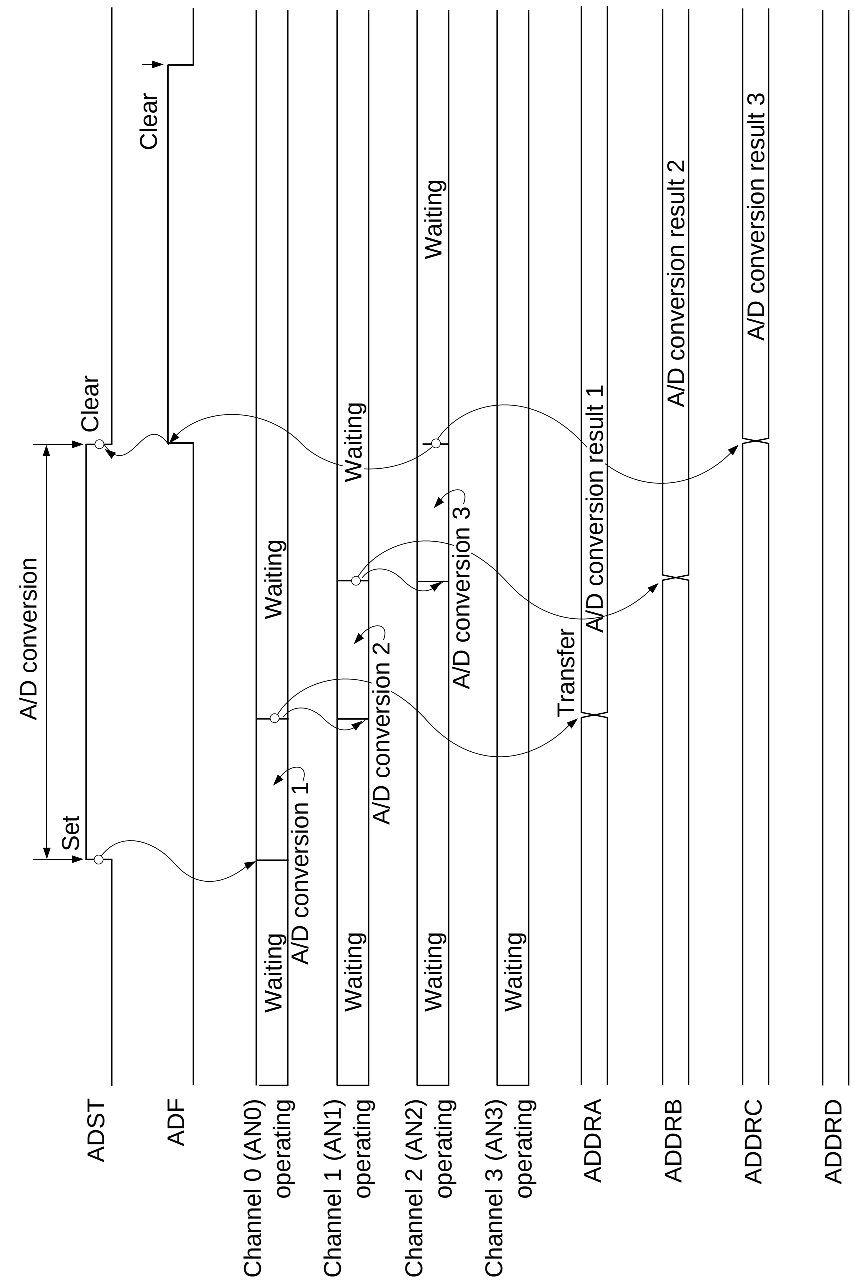
23.4.2 Multi Mode (MULTI = 1, SCN = 0)

Multi mode should be selected when performing multi channel A/D conversions on one or more channels. When the ADST bit in A/D control/status register (ADCSR) is set to 1 by software or external trigger input, A/D conversion starts on the first channel (AN0). When two or more channels are selected, after conversion of the first channel ends, conversion of the second channel (AN1) starts immediately. When A/D conversions end on the selected channels, the ADST bit is cleared to 0. The conversion results are transferred for storage into the A/D data registers corresponding to the channels.

When the mode or analog input channel selection must be changed during A/D conversion, to prevent incorrect operation, first clear the ADST bit to 0 in ADCSR to halt A/D conversion. After making the necessary changes, set the ADST bit to 1. A/D conversion will start again from the first channel in the group. The ADST bit can be set at the same time as the mode or channel selection is changed.

Typical operations when three channels in group 0 (AN0–AN2) are selected in scan mode are described next. Figure 23.4 shows a timing diagram for this example.

1. Multi mode is selected (MULTI = 1, SCN = 0), analog input channels AN0–AN2 are selected (CH1 = 1, CH0 = 0), and A/D conversion is started (ADST = 1).
2. When A/D conversion of the first channel (AN0) is completed, the result is transferred into ADDRA. Next, conversion of the second channel (AN1) starts automatically.
3. Conversion proceeds in the same way through the third channel (AN2).
4. When conversion of all selected channels (AN0–AN2) is completed, the ADF flag is set to 1 and ADST bit is cleared to 0. If the ADIE bit is set to 1, an ADI interrupt is requested at this time.



Note: Downward arrows (↓) indicate instruction executed by software.

Figure 23.4 Example of A/D Converter Operation (Multi Mode, Channels AN0 to AN2 Selected)

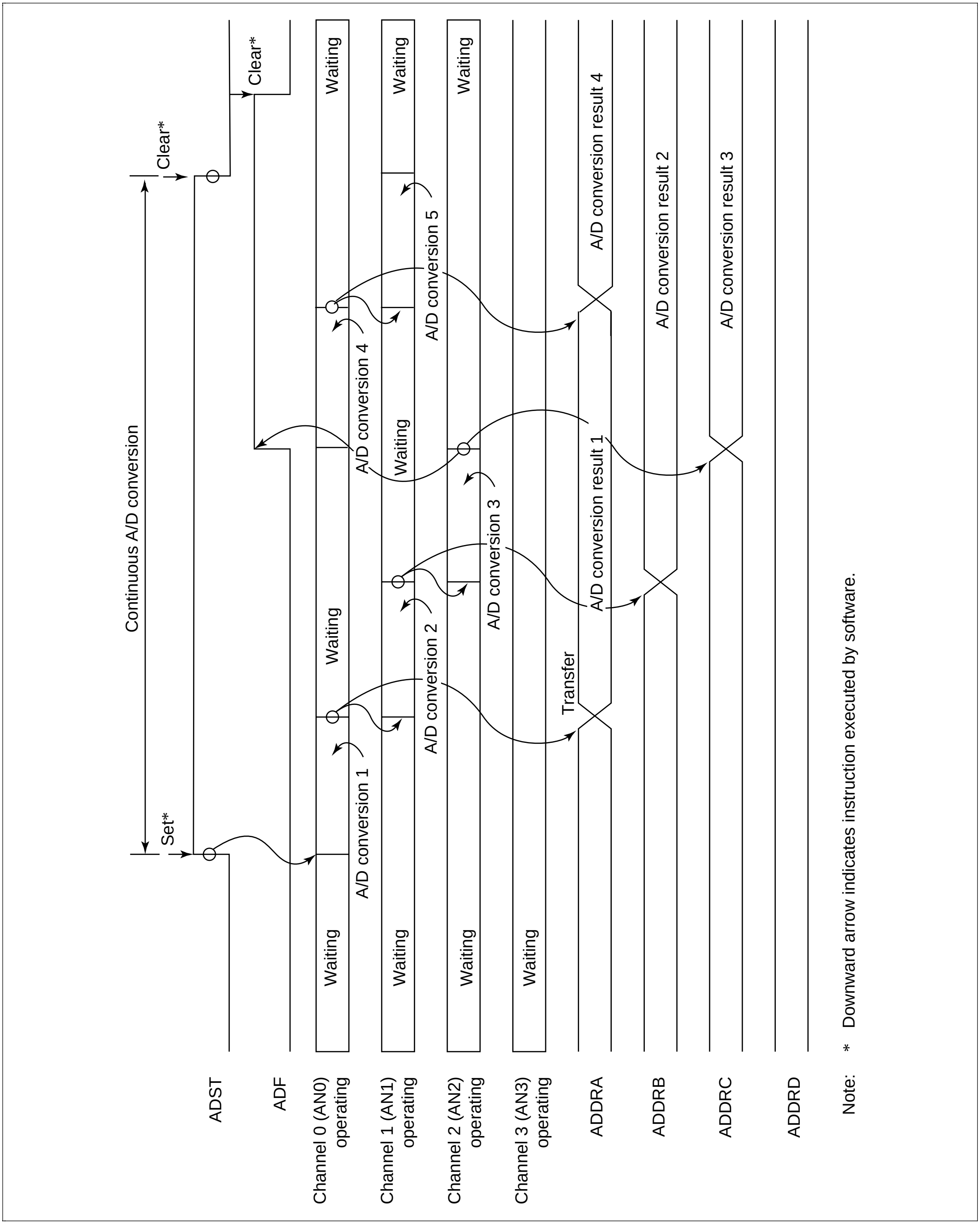
23.4.3 Scan Mode (MULTI = 1, SCN = 1)

Scan mode is useful for monitoring analog inputs in a group of one or more channels. When the ADST bit in the A/D control/status register (ADCSR) is set to 1 by software or external trigger input, A/D conversion starts on the first channel (AN0). When two or more channels are selected, after conversion of the first channel ends, conversion of the second channel (AN1) starts immediately. A/D conversion continues cyclically on the selected channels until the ADST bit is cleared to 0. The conversion results are transferred for storage into the A/D data registers corresponding to the channels.

When the mode or analog input channel must be changed during analog conversion, to prevent incorrect operation, first clear the ADST bit to 0 to halt A/D conversion. After making the necessary changes, set the ADST bit to 1. A/D conversion will start again from the first channel in the group. The ADST bit can be set at the same time as the mode or channel selection is changed.

Typical operations when three channels (AN0–AN2) are selected in scan mode are described next. Figure 23.5 shows a timing diagram for this example.

1. Scan mode is selected (MULTI = 1, SCN = 1), analog input channels AN0–AN2 are selected (CH1 = 1, CH0 = 0), and A/D conversion is started (ADST = 1).
2. When A/D conversion of the first channel (AN0) is completed, the result is transferred into ADDRA. Next, conversion of the second channel (AN1) starts automatically.
3. Conversion proceeds in the same way through the third channel (AN2).
4. When conversion of all the selected channels (AN0–AN2) is completed, the ADF flag is set to 1 and conversion of the first channel (AN0) starts again. If the ADIE bit is set to 1, an ADI interrupt is requested at this time.
5. Steps 2 to 4 are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops. After that, if the ADST bit is set to 1, A/D conversion starts again from the first channel (AN0).



Note: * Downward arrow indicates instruction executed by software.

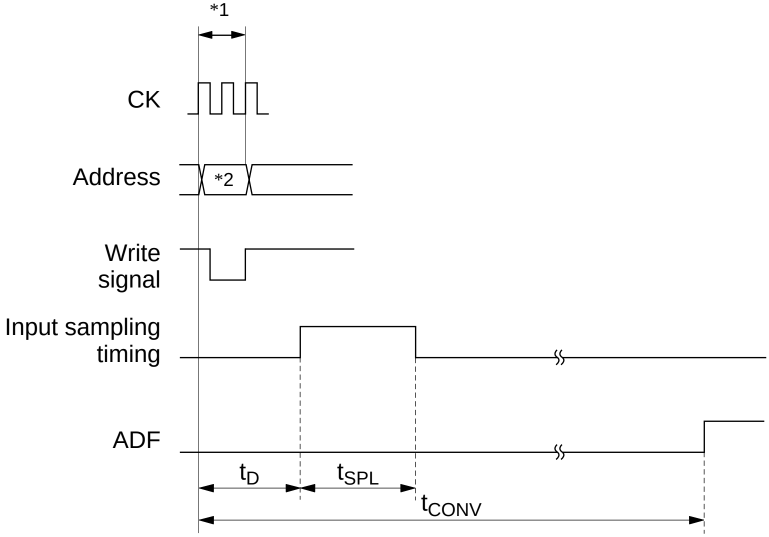
Figure 23.5 Example of A/D Converter Operation (Scan Mode, Channels AN0 to AN2 Selected)

23.4.4 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit. The A/D converter samples the analog input at a time t_D after the ADST bit in A/D control/status register (ADCSR) is set to 1, then starts conversion. Figure 23.6 shows the A/D conversion timing. Table 23.4 indicates the A/D conversion time.

As indicated in figure 23.6, the A/D conversion time includes t_D and the input sampling time. The length of t_D varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 23.4.

In multi mode and scan mode, the values given in table 23.4 apply to the first conversion. In the second and subsequent conversions the conversion time is fixed at 256 states when CKS = 0 or 128 states when CKS = 1.



t_D A/D conversion start delay

t_{SPL} Input sampling time

t_{CONV} A/D conversion time

Notes: *1 ADCSR write cycle
*2 ADCSR address

Figure 23.6 A/D Conversion Timing

Table 23.4 A/D Conversion Time (Single Mode)

		CKS = 0			CKS = 1		
	Symbol	Min	Typ	Max	Min	Typ	Max
A/D conversion start delay	t_D	10	—	17	6	—	9
Input sampling time	t_{SPL}	—	65	—	—	32	—
A/D conversion time	t_{CONV}	259	—	266	131	—	134

Note: Values in the table are numbers of states (t_{cyc}).

23.4.5 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGE1, TRGE0 bits are set to 1 in A/D control register (ADCR), external trigger input is enabled at the $\overline{\text{ADTRG}}$ pin. A high-to-low transition at the $\overline{\text{ADTRG}}$ pin sets the ADST bit to 1 in A/D control/status register (ADCSR), starting A/D conversion. Other operations, regardless of the conversion mode, are the same as if the ADST bit had been set to 1 by software. Figure 23.7 shows the timing.

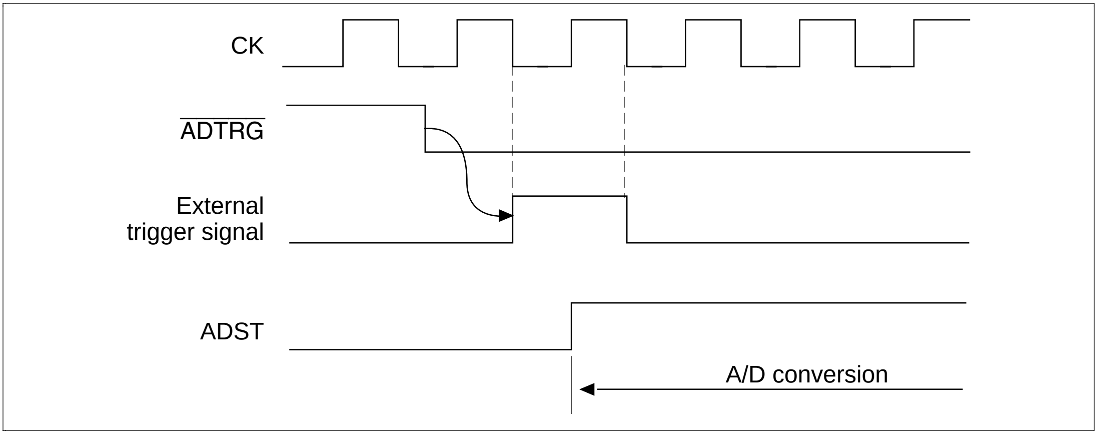


Figure 23.7 External Trigger Input Timing

23.5 Interrupts

The A/D converter generates an interrupt (ADI) at the end of A/D conversion. The ADI interrupt request can be enabled or disabled by the ADIE bit in ADCSR.

23.6 Definitions of A/D Conversion Accuracy

The A/D converter compares an analog value input from an analog input channel to its analog reference value and converts it into 10-bit digital data. The absolute accuracy of this A/D conversion is the deviation between the input analog value and the output digital value. It includes the following errors:

- Offset error
- Full-scale error
- Quantization error
- Nonlinearity error

These four error quantities are explained below using figure 23.8. In the figure, the 10 bits of the A/D converter have been simplified to 3 bits.

Offset error is the deviation between actual and ideal A/D conversion characteristics when the digital output value changes from the minimum (zero voltage) 0000000000 (000 in the figure) to 000000001 (001 in the figure)(figure 23.8, item (1)). Full-scale error is the deviation between actual and ideal A/D conversion characteristics when the digital output value changes from the 1111111110 (110 in the figure) to the maximum 1111111111 (111 in the figure)(figure 23.8, item (2)). Quantization error is the intrinsic error of the A/D converter and is expressed as 1/2 LSB (figure 23.8, item (3)). Nonlinearity error is the deviation between actual and ideal A/D conversion characteristics between zero voltage and full-scale voltage (figure 23.8, item (4)). Note that it does not include offset, full-scale or quantization error.

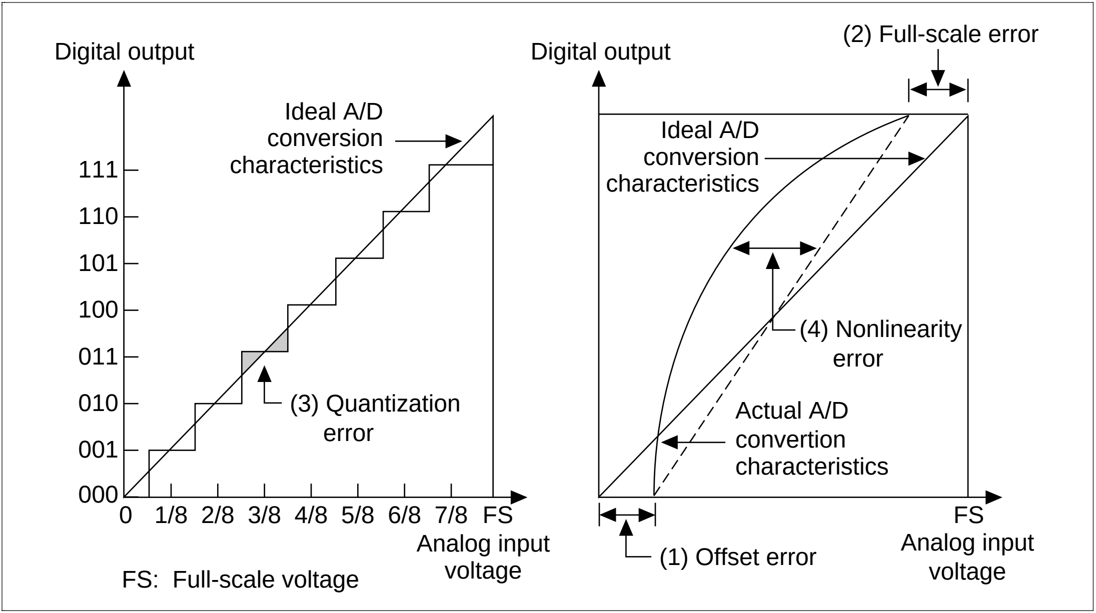


Figure 23.8 Definitions of A/D Conversion Accuracy

23.7 A/D Converter Usage Notes

When using the A/D converter, note the points listed in section 23.7.1 below.

23.7.1 Setting Analog Input Voltage

- Analog Input Voltage Range: During A/D conversion, the voltages input to the analog input pins ANn should be in the range $AV_{SS} \leq ANn \leq AV_{CC}$ (n = 0 to 3).
- Relationships of AV_{CC} and AV_{SS}: AV_{CC} and AV_{SS} should be related as follows: $AV_{CC} = V_{CC}Q \pm 0.2$ V and $AV_{SS} = V_{SS}$.

23.7.2 Processing of Analog Input Pins

To prevent damage from voltage surges at the analog input pins (AN0–AN3), connect an input protection circuit like the one shown in figure 23.9. The circuit shown also includes an RC filter to suppress noise. This circuit is shown as an example; The circuit constants should be selected according to actual application conditions. Table 23.5 lists the analog input pin specifications and figure 23.10 shows an equivalent circuit diagram of the analog input ports.

23.7.3 Access Size and Read Data

Table 23.6 shows the relationship between access size and read data. Note the read data obtained with different access sizes, bus widths, and endian modes.

The case is shown here in which H'3FF is obtained when AV_{CC} is input as an analog input. FF is the data containing the upper 8 bits of the conversion result, and C0 is the data containing the lower 2 bits.

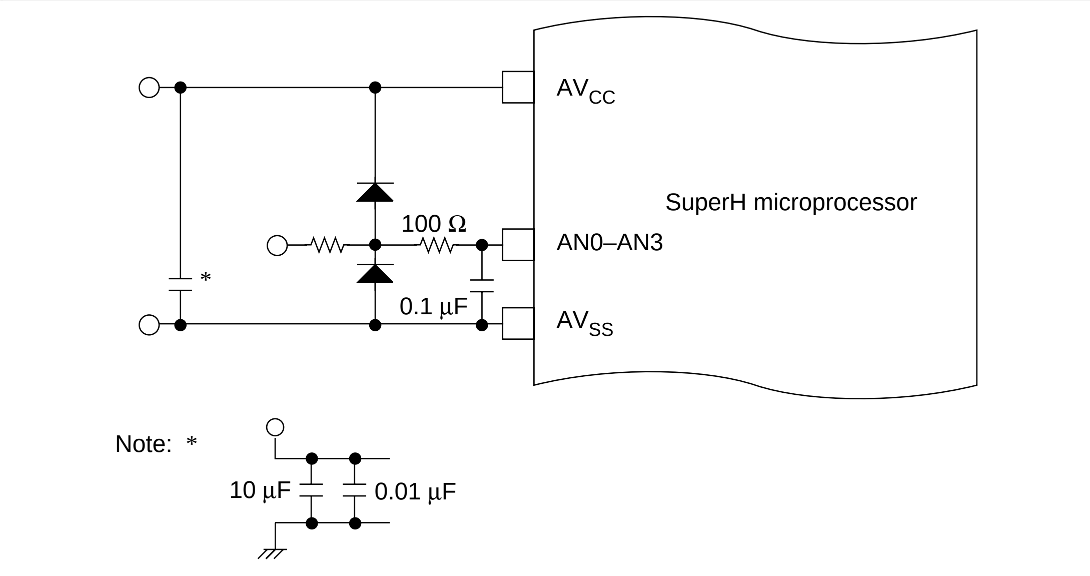


Figure 23.9 Example of Analog Input Protection Circuit

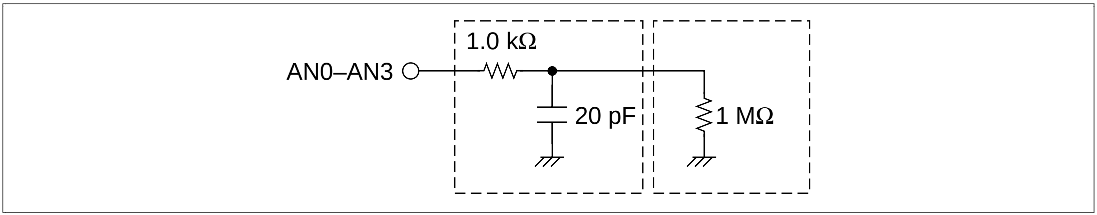


Figure 23.10 Analog Input Pin Equivalent Circuit (Reference Values)

Table 23.5 Analog Input Pin Ratings (Reference Values)

Item	Min	Max	Unit
Analog input capacitance	—	20	pF
Allowable signal-source impedance	—	5	kΩ

Table 23.6 Relationship between Access Size and Read Data

Access		Bus Width	32 Bits (D31—D0)		16 Bits (D15—D0)		8 Bits (D7—D0)	
Size	Command	Endian	Big	Little	Big	Little	Big	Little

Byte access	`MOV.L`
	`MOV.B`
		`#ADDRAH,R9`
		`@R9,R8`
			FFFFFFFF	FFFFFFFF	FFFF	FFFF	FF	FF

	`MOV.L`
	`MOV.B`
		`#ADDRAL,R9`
		`@R9,R8`
			C0C0C0C0	C0C0C0C0	C0C0	C0C0	C0	C0

Word access	`MOV.L`
	`MOV.W`
		`#ADDRAH,R9`
		`@R9,R8`
			FFxxFFxx	FFxxFFxx	FFxx	FFxx	FFxx	FFxx

	`MOV.L`
	`MOV.W`
		`#ADDRAL,R9`
		`@R9,R8`
			C0xxC0xx	C0xxC0xx	C0xx	C0xx	C0xx	C0xx

Longword access	`MOV.L`
	`MOV.L`
		`#ADDRAH,R9`
		`@R9,R8`
			FFxxC0xx	FFxxC0xx	FFxxC0xx	FFxxC0xx	FFxxC0xx	FFxxC0xx

In this table:

```
#ADDRAH   .EQU   H'04000080
#ADDRAL   .EQU   H'04000082
```

Values are shown in hexadecimal for the case where read data is output to an external device via R8.

Section 24 Hitachi User Debug Interface (H-UDI)

24.1 Overview

The SH7622 incorporates a Hitachi user debug interface (H-UDI) and advanced user debugger (AUD) for program debugging.

24.2 Hitachi User Debug Interface (H-UDI)

The H-UDI performs on-chip debugging which is supported by the SH7622 The H-UDI described here is a serial interface which is pi-compatible with JTAG (Joint Test Action Group, IEEE Standard 1149.1 and IEEE Standard Test Access Port and Boundary-Scan Architecture) specifications.

The H-UDI in the SH7622 supports a boundary scan mode, and is also used for emulator connection.

When using an emulator, H-UDI functions should not be used. Refer to the each emulator manual for the method of connecting the emulator.

24.2.1 Pin Description

TCK: H-UDI serial data input/output clock pin. Data is serially supplied to the H-UDI from the data input pin (TDI), and output from the data output pin (TDO), in synchronization with this clock.

TMS: Mode select input pin. The state of the TAP control circuit is determined by changing this signal in synchronization with TCK. The protocol conforms to the JTAG standard (IEEE Std. 1140.1).

$\overline{\textbf{TRST}}$: H-UDI reset input pin. Input is accepted asynchronously with respect to TCK, and when low, the H-UDI is reset. See section 24.4.2, Reset Configuration, for more information.

TDI: H-UDI serial data input pin. Data transfer to the H-UDI is executed by changing this signal in synchronization with TCK.

TDO: H-UDI serial data output pin. Data output from the H-UDI is executed by reading this signal in synchronization with TCK.

$\overline{\textbf{ASEMD0}}$: ASE mode select pin. If a low level is input at the $\overline{\text{ASEMD0}}$ pin while the $\overline{\text{RESETP}}$ pin is asserted, ASE mode is entered; if a high level is input, normal mode is entered. In ASE mode, boundary scan and emulator functions can be used. The input level at the $\overline{\text{ASEMD0}}$ pin should be held for at least 1 cycle after $\overline{\text{RESETP}}$ negation.

$\overline{\text{ASEBRKAK}}$: Dedicated emulator pin

24.2.2 Block Diagram

Figure 24.1 shows the block diagram of the H-UDI.

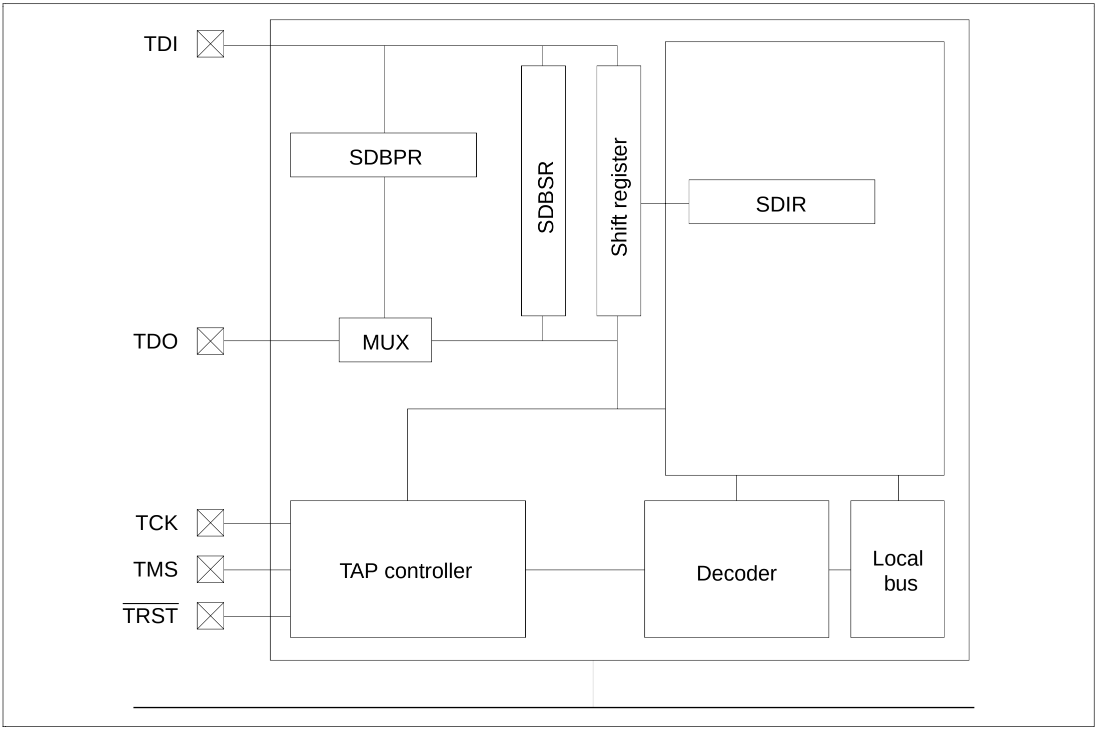


Figure 24.1 H-UDI Block Diagram

24.3 Register Descriptions

The H-UDI has the following registers.

- SDBPR: Bypass register
- SDIR: Instruction register
- SDBSR: Boundary scan register

Table 24.1 shows H-UDI register configuration.

Table 24.1 H-UDI Registers

		CPU Side			H-UDI Side		Initial
Name	**Abbreviation**	**R/W**	**Size**	**Address**	**R/W**	**Size**	**Value***
Bypass register	SDBPR	—	—	—	R/W	1	Undefined
Instruction register	SDIR	R	16	H'A4000200	R/W	16	H'FFFF
Boundary register	SDBSR	—	—	—	R/W	—	Undefined

Note: * Initialized when $\overline{\text{TRST}}$ pin is low or when TAP is in the test-logic-reset state.

24.3.1 Bypass Register (SDBPR)

The bypass register is a 1-bit register that cannot be accessed by the CPU. When the SDIR is set to the bypass mode, the SDBPR is connected between H-UDI pins TDI and TDO.

24.3.2 Instruction Register (SDIR)

The instruction register (SDIR) is a 16-bit read-only register. The register is in bypass mode in its initial state. It is initialized by $\overline{\text{TRST}}$ or in the TAP test-logic-reset state, and can be written by the H-UDI irrespective of the CPU mode. Operation is not guaranteed when a reserved command is set to this register

Bit:	15	14	13	12	11	10	9	8
	TI3	TI2	TI1	TI0	—	—	—	—
Initial value:	1	1	1	1	1	1	1	1

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—
Initial value:	1	1	1	1	1	1	1	1

Bits 15 to 12—Test Instruction Bits (TI3–TI0): Cannot be written by the CPU.

Table 24.2 H-UDI Commands

TI3	TI2	TI1	TI0	Description
0	0	0	0	EXTEST
0	1	0	0	SAMPLE/PRELOAD
0	1	0	1	Reserved
0	1	1	0	H-UDI reset negate
0	1	1	1	H-UDI reset assert
1	0	0	—	Reserved
1	0	1	—	H-UDI interrupt
1	1	0	—	Reserved
1	1	1	0	Reserved
1	1	1	1	BYPASS
0	0	0	1	Reserved

Bits 11 to 0—Reserved: Always read 1.

24.3.3 Boundary Scan Register (SDBSR)

The boundary scan register (SDBSR) is a shift register, located on the PAD, for controlling the input/output pins of the SH7622.

Using the EXTEST and SAMPLE/PRELOAD commands, a boundary scan test conforming to the JTAG standard can be carried out. Table 24.3 shows the correspondence between SH7622 pins and boundary scan register bits.

Table 24.3 SH7622 Pins and Boundary Scan Register Bits

Bit	Pin Name	I/O	Bit	Pin Name	I/O
from TDI			316	D1	IN
346	D31/PTB7	IN	315	D0	IN
345	D30/PTB6	IN	314	MD1	IN
344	D29/PTB5	IN	313	MD2	IN
343	D28/PTB4	IN	312	NMI	IN
342	D27/PTB3	IN	311	IRQ0/PTH0	IN
341	D26/PTB2	IN	310	IRQ1/PTH1	IN
340	D25/PTB1	IN	309	IRQ2/PTH2	IN
339	D24/PTB0	IN	308	IRQ3/PTH3	IN
338	D23/PTA7	IN	307	IRQ4/PTH4	IN
337	D22/PTA6	IN	306	D31/PTB7	OUT
336	D21/PTA5	IN	305	D30/PTB6	OUT
335	D20/PTA4	IN	304	D29/PTB5	OUT
334	D19/PTA3	IN	303	D28/PTB4	OUT
333	D18/PTA2	IN	302	D27/PTB3	OUT
332	D17/PTA1	IN	301	D26/PTB2	OUT
331	D16/PTA0	IN	300	D25/PTB1	OUT
330	D15	IN	299	D24/PTB0	OUT
329	D14	IN	298	D23/PTA7	OUT
328	D13	IN	297	D22/PTA6	OUT
327	D12	IN	296	D21/PTA5	OUT
326	D11	IN	295	D20/PTA4	OUT
325	D10	IN	294	D19/PTA3	OUT
324	D9	IN	293	D18/PTA2	OUT
323	D8	IN	292	D17/PTA1	OUT
322	D7	IN	291	D16/PTA0	OUT
321	D6	IN	290	D15	OUT
320	D5	IN	289	D14	OUT
319	D4	IN	288	D13	OUT
318	D3	IN	287	D12	OUT
317	D2	IN	286	D11	OUT

Table 24.3 SH7622 Pins and Boundary Scan Register Bits (cont)

Bit	Pin Name	I/O	Bit	Pin Name	I/O
285	D10	OUT	255	D12	Control
284	D9	OUT	254	D11	Control
283	D8	OUT	253	D10	Control
282	D7	OUT	252	D9	Control
281	D6	OUT	251	D8	Control
280	D5	OUT	250	D7	Control
279	D4	OUT	249	D6	Control
278	D3	OUT	248	D5	Control
277	D2	OUT	247	D4	Control
276	D1	OUT	246	D3	Control
275	D0	OUT	245	D2	Control
274	D31/PTB7	Control	244	D1	Control
273	D30/PTB6	Control	243	D0	Control
272	D29/PTB5	Control	242	$\overline{\text{BS}}$/PTK4	IN
271	D28/PTB4	Control	241	$\overline{\text{WE2}}$/DQMUL/PTK6	IN
270	D27/PTB3	Control	240	$\overline{\text{WE3}}$/DQMUU/PTK7	IN
269	D26/PTB2	Control	239	$\overline{\text{AUDSYNC}}$/PTE7	IN
268	D25/PTB1	Control	238	$\overline{\text{CS2}}$/PTK0	IN
267	D24/PTB0	Control	237	$\overline{\text{CS3}}$/PTK1	IN
266	D23/PTA7	Control	236	$\overline{\text{CS4}}$/PTK2	IN
265	D22/PTA6	Control	235	$\overline{\text{CS5}}$/PTK3	IN
264	D21/PTA5	Control	234	PTE4	IN
263	D20/PTA4	Control	233	PTE5	IN
262	D19/PTA3	Control	232	A0	OUT
261	D18/PTA2	Control	231	A1	OUT
260	D17/PTA1	Control	230	A2	OUT
259	D16/PTA0	Control	229	A3	OUT
258	D15	Control	228	A4	OUT
257	D14	Control	227	A5	OUT
256	D13	Control	226	A6	OUT

Table 24.3 SH7622 Pins and Boundary Scan Register Bits (cont)

Bit	Pin Name	I/O	Bit	Pin Name	I/O
225	A7	OUT	195	$\overline{\text{CS4}}$/PTK2	OUT
224	A8	OUT	194	$\overline{\text{CS5}}$/CE1A/PTK3	OUT
223	A9	OUT	193	$\overline{\text{CS6}}$	OUT
222	A10	OUT	192	PTE4	OUT
221	A11	OUT	191	PTE5	OUT
220	A12	OUT	190	A0	Control
219	A13	OUT	189	A1	Control
218	A14	OUT	188	A2	Control
217	A15	OUT	187	A3	Control
216	A16	OUT	186	A4	Control
215	A17	OUT	185	A5	Control
214	A18	OUT	184	A6	Control
213	A19	OUT	183	A7	Control
212	A20	OUT	182	A8	Control
211	A21	OUT	181	A9	Control
210	A22	OUT	180	A10	Control
209	A23	OUT	179	A11	Control
208	A24	OUT	178	A12	Control
207	A25	OUT	177	A13	Control
206	BS/PTK4	OUT	176	A14	Control
205	$\overline{\text{RD}}$	OUT	175	A15	Control
204	$\overline{\text{WE0}}$/DQMLL	OUT	174	A16	Control
203	$\overline{\text{WE1}}$/DQMLU/$\overline{\text{WE}}$	OUT	173	A17	Control
202	$\overline{\text{WE2}}$/DQMUL/PTK6	OUT	172	A18	Control
201	$\overline{\text{WE3}}$/DQMUU/PTK7	OUT	171	A19	Control
200	RD/$\overline{\text{WR}}$	OUT	170	A20	Control
199	$\overline{\text{AUDSYNC}}$/PTE7	OUT	169	A21	Control
198	$\overline{\text{CS0}}$	OUT	168	A22	Control
197	$\overline{\text{CS2}}$/PTK0	OUT	167	A23	Control
196	$\overline{\text{CS3}}$/PTK1	OUT	166	A24	Control

Table 24.3 SH7622 Pins and Boundary Scan Register Bits (cont)

Bit	Pin Name	I/O	Bit	Pin Name	I/O
165	A25	Control	135	$\overline{\text{BREQ}}$	IN
164	$\overline{\text{BS}}$/PTK4	Control	134	$\overline{\text{WAIT}}$	IN
163	$\overline{\text{RD}}$	Control	133	AUDCK/PTH6	IN
162	$\overline{\text{WE0}}$/DQMLL	Control	132	PTG7	IN
161	$\overline{\text{WE1}}$/DQMLU/$\overline{\text{WE}}$	Control	131	$\overline{\text{ASEBRKAK}}$/PTG5	IN
160	$\overline{\text{WE2}}$/DQMUL/PTK6	Control	130	UCLK/PTG4	IN
159	$\overline{\text{WE3}}$/DQMUU/PTK7	Control	129	AUDATA3/PTG3	IN
158	RD/$\overline{\text{WR}}$	Control	128	AUDATA2/PTG2	IN
157	$\overline{\text{AUDSYNC}}$/PTE7	Control	127	AUDATA1/PTG1	IN
156	$\overline{\text{CS0}}$	Control	126	AUDATA0/PTG0	IN
155	$\overline{\text{CS2}}$/PTK0	Control	125	$\overline{\text{ADTRG}}$/PTH5	IN
154	$\overline{\text{CS3}}$/PTK1	Control	124	DMNS/PTF3	IN
153	$\overline{\text{CS4}}$/PTK2	Control	123	DPLS/PTF2	IN
152	$\overline{\text{CS5}}$/$\overline{\text{CE1A}}$/PTK3	Control	122	TXDPLS/PTF1	IN
151	$\overline{\text{CS6}}$	Control	121	TXDMNS/PTF0	IN
150	PTE4	Control	120	MD0	IN
149	PTE5	Control	119	CKE/PTK5	OUT
148	CKE/PTK5	IN	118	$\overline{\text{RAS3L}}$/PTJ0	OUT
147	$\overline{\text{RAS3L}}$/PTJ0	IN	117	PTJ1	OUT
146	PTJ1	IN	116	$\overline{\text{CASL}}$/PTJ2	OUT
145	$\overline{\text{CASL}}$/PTJ2	IN	115	$\overline{\text{CASU}}$/PTJ3	OUT
144	$\overline{\text{CASU}}$/PTJ3	IN	114	PTJ4	OUT
143	PTJ4	IN	113	PTJ5	OUT
142	PTJ5	IN	112	DACK0/PTD5	OUT
141	DACK0/PTD5	IN	111	DACK1/PTD7	OUT
140	DACK1/PTD7	IN	110	PTE6	OUT
139	PTE6	IN	109	PTE3	OUT
138	PTE3	IN	108	$\overline{\text{RAS3U}}$/PTE2	OUT
137	$\overline{\text{RAS3U}}$/PTE2	IN	107	PTE1	OUT
136	PTE1	IN	106	$\overline{\text{BACK}}$	OUT

Table 24.3 SH7622 Pins and Boundary Scan Register Bits (cont)

Bit	Pin Name	I/O	Bit	Pin Name	I/O
105	$\overline{\text{ASEBRKAK}}$/PTG5	OUT	73	$\overline{\text{SCK1}}$/SCPT3	IN
104	$\overline{\text{AUDATA3}}$/PTG3	OUT	72	$\overline{\text{SCK2}}$/SCPT5	IN
103	$\overline{\text{AUDATA2}}$/PTG2	OUT	71	SCPT6	IN
102	$\overline{\text{AUDATA1}}$/PTG1	OUT	70	RxD0/SCPT0	IN
101	$\overline{\text{AUDATA0}}$/PTG0	OUT	69	RxD2/SCPT4	IN
100	TXDPLS/PTF1	OUT	68	VBUS/PTD3	IN
99	TXDMNS/PTF0	OUT	67	SUSPND/PTD2	IN
98	CKE/PTK5	Control	66	DRAK0/PTD1	IN
97	$\overline{\text{RAS3L}}$/PTJ0	Control	65	DRAK1/PTD0	IN
96	PTJ1	Control	64	$\overline{\text{DREQ0}}$/PTD4	IN
95	$\overline{\text{CASL}}$/PTJ2	Control	63	$\overline{\text{DREQ1}}$/PTD6	IN
94	$\overline{\text{CASU}}$/PTJ3	Control	62	RxD1/SCPT2	IN
93	PTJ4	Control	61	IRQ5/SCPT7	IN
92	PTJ5	Control	60	IRQ6/PTC7	IN
91	DACK0/PTD5	Control	59	IRQ7/PTC6	IN
90	DACK1/PTD7	Control	58	XVDATA/PTC5	IN
89	PTE6	Control	57	$\overline{\text{TXENL}}$/PTC4	IN
88	PTE3	Control	56	PTC3^{*2}	IN
87	$\overline{\text{RAS3U}}$/PTE2	Control	55	PTC2^{*2}	IN
86	PTE1	Control	54	PTC1	IN
85	$\overline{\text{BACK}}$	Control	53	PTC0^{*2}	IN
84	$\overline{\text{ASEBRKAK}}$/PTG5	Control	52	MD3	IN
83	$\overline{\text{AUDATA3}}$/PTG3	Control	51	MD4	IN
82	$\overline{\text{AUDATA2}}$/PTG2	Control	50	^{*1}	IN
81	$\overline{\text{AUDATA1}}$/PTG1	Control	49	PTL4	IN
80	$\overline{\text{AUDATA0}}$/PTG0	Control	48	PTL5	IN
79	TXDPLS/PTF1	Control	47	PTL6	IN
78	TXDMNS/PTF0	Control	46	PTL7	IN
77	$\overline{\text{STATUS0}}$/PTJ6	IN	45	STATUS0/PTJ6	OUT
76	$\overline{\text{STATUS1}}$/PTJ7	IN	44	STATUS1/PTJ7	OUT
75	TCLK/PTH7	IN	43	TCLK/PTH7	OUT
74	$\overline{\text{SCK0}}$/SCPT1	IN	42	IRQOUT	OUT

Table 24.3 SH7622 Pins and Boundary Scan Register Bits (cont)

Bit	Pin Name	I/O	Bit	Pin Name	I/O
41	TxD0/SCPT0	OUT	19	$\overline{\text{IRQOUT}}$	Control
40	$\overline{\text{SCK0}}$/SCPT1	OUT	18	TxD0/SCPT0	Control
39	TxD1/SCPT2	OUT	17	SCK0/SCPT1	Control
38	$\overline{\text{SCK1}}$/SCPT3	OUT	16	TxD1/SCPT2	Control
37	TxD2/SCPT4	OUT	15	SCK1/SCPT3	Control
36	$\overline{\text{SCK2}}$/SCPT5	OUT	14	TxD2/SCPT4	Control
35	SCPT6	OUT	13	SCK2/SCPT5	Control
34	IRQ6/PTC7	OUT	12	SCPT6	Control
33	IRQ7/PTC6	OUT	11	IRQ6/PTC7	Control
32	XVDATA/PTC5	OUT	10	IRQ7/PTC6	Control
31	$\overline{\text{TXENL}}$/PTC4	OUT	9	XVDATA/PTC5	Control
30	VBUS/PTD3	OUT	8	$\overline{\text{TXENL}}$/PTC4	Control
29	SUSPND/PTD2	OUT	7	VBUS/PTD3	Control
28	PTC3	OUT	6	SUSPND/PTD2	Control
27	PTC2	OUT	5	PTC3	Control
26	PTC1*3	OUT	4	PTC2	Control
25	PTC0	OUT	3	PTC1	Control
24	DRAK0/PTD1	OUT	2	PTC0	Control
23	DRAK1/PTD0	OUT	1	DRAK0/PTD1	Control
22	STATUS0/PTJ6	Control	0	DRAK1/PTD0	Control
21	STATUS1/PTJ7	Control	to TDO		
20	TCLK/PTH7	Control			

Notes: Control is an active-low signal.

When Control is driven low, the corresponding pin is driven by the value of OUT.

*1 Not available to the user, but subject to boundary scan. Must be pulled down.

*2 Output-only pin, but input pin side also included as object of boundary scan.

*3 Output-only pin, but output pin side also included as object of boundary scan.

24.4 H-UDI Operations

24.4.1 TAP Controller

Figure 24.2 shows the internal states of TAP controller. State transitions basically conform with the JTAG standard.

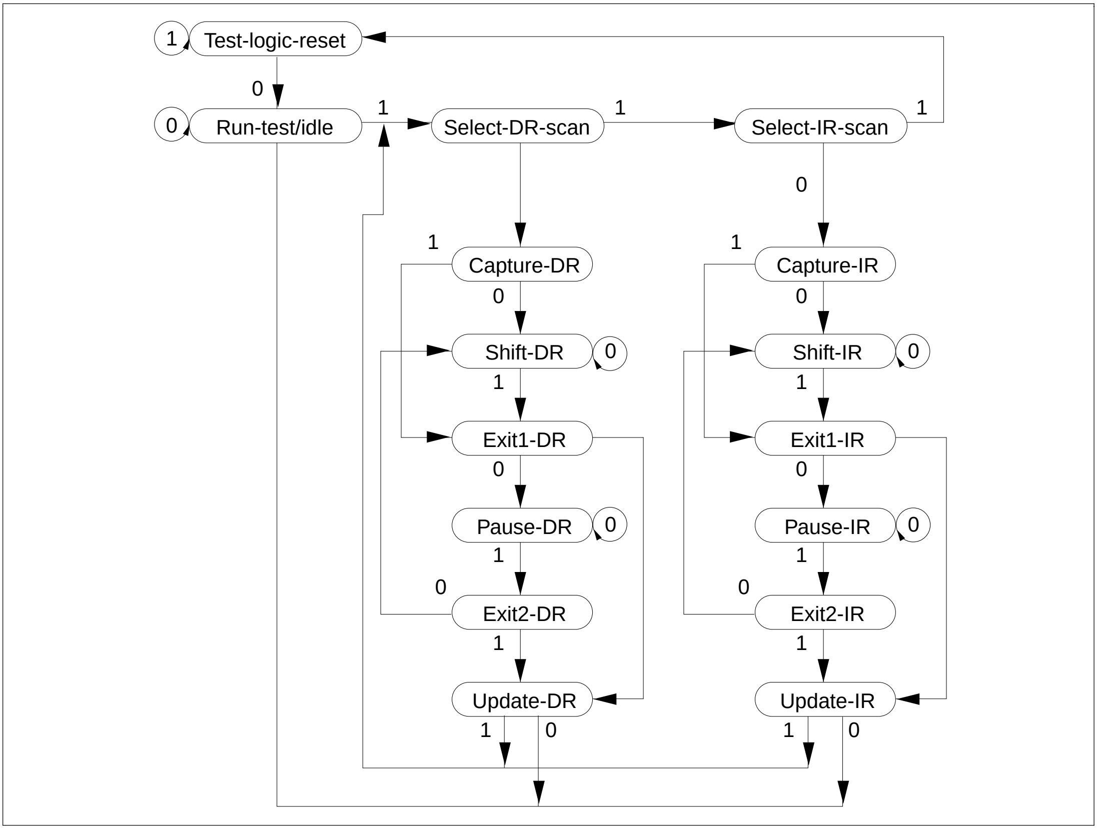


Figure 24.2 TAP Controller State Transitions

Note: The transition condition is the TMS value on the rising edge of TCK. The TDI value is sampled on the rising edge of TCK; shifting occurs on the falling edge of TCK. The TDO value changes on the TCK falling edge. The TDO is at high impedance, except with shift-DR (shift-SR) and shift-IR states. During the change to $\overline{\text{TRST}}$ = 0, there is a transition to test-logic-reset asynchronously with TCK.

24.4.2 Reset Configuration

Table 24.4 Reset Configuration

$\overline{\text{ASDMD0}}$*[1]	$\overline{\text{RESETP}}$	$\overline{\text{TRST}}$	Chip State
H	L	L	Normal reset and H-UDI reset
		H	Normal reset
	H	L	H-UDI reset only
		H	Normal operation
L	L	L	Reset hold*[2]
		H	Normal reset
	H	L	H-UDI reset only
		H	Normal operation

Notes: *1 Performs main chip mode and ASE mode settings

$\overline{\text{ASEMD0}}$ = 1, main chip mode

$\overline{\text{ASEMD0}}$ = 0, ASE mode

When using the user system alone without using an emulator or the H-UDI, set $\overline{\text{ASEMD0}}$ = H.

*2 During ASE mode, reset hold is enabled by setting $\overline{\text{RESETP}}$ and $\overline{\text{TRST}}$ pins at low level for a constant cycle. In this state, the CPU does not start up, even if $\overline{\text{RESETP}}$ is set to high level. When $\overline{\text{TRST}}$ is set to high level, H-UDI operation is enabled, but the CPU does not start up. The reset hold state is cancelled by the following:

- Boot request from H-UDI (boot sequence)
- Another $\overline{\text{RESETP}}$ assert (power-on reset)

24.4.3 H-UDI Reset

An H-UDI reset is executed by setting an H-UDI reset assert command in SDIR. An H-UDI reset is of the same kind as a power-on reset. An H-UDI reset is released by inputting an H-UDI reset negate command.

The interval required between the H-UDI reset assert command and the H-UDI reset negate command is the same as the time for which the $\overline{\text{RESETP}}$ pin is held low in order to execute a power-on reset.

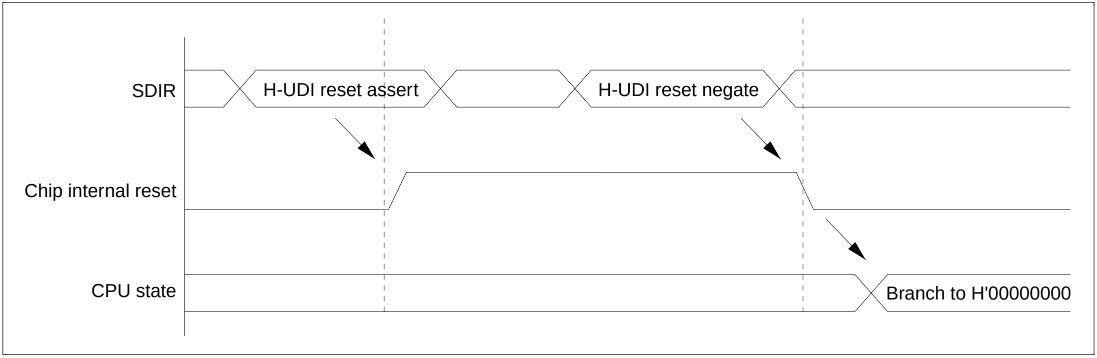


Figure 24.3 H-UDI Reset

24.4.4 H-UDI Interrupt

The H-UDI interrupt function generates an interrupt by setting a command from the H-UDI in the SDIR. An H-UDI interrupt is a general exception/interrupt operation, resulting in a branch to an address based on the VBR value plus offset, and return by the RTE instruction. This interrupt request has a fixed priority level of 15.

H-UDI interrupts are not accepted in sleep mode or standby mode.

24.4.5 Bypass

The JTAG-based bypass mode for the H-UDI pins can be selected by setting a command from the H-UDI in the SDIR.

24.5 Boundary Scan

A command can be set in SDIR by the H-UDI to place the H-UDI pins in the boundary scan mode stipulated by JTAG.

24.5.1 Supported Instructions

The SH7622 supports the three essential instructions defined in the JTAG standard (BYPASS, SAMPLE/PRELOAD, and EXTEST).

BYPASS: The BYPASS instruction is an essential standard instruction that operates the bypass register. This instruction shortens the shift path to speed up serial data transfer involving other chips on the printed circuit board. While this instruction is executing, the test circuit has no effect on the system circuits. The instruction code is 1111.

SAMPLE/PRELOAD: The SAMPLE/PRELOAD instruction inputs values from the SH7622's internal circuitry to the boundary scan register, outputs values from the scan path, and loads data onto the scan path. When this instruction is executing, the SH7622's input pin signals are transmitted directly to the internal circuitry, and internal circuit values are directly output externally from the output pins. The SH7622's system circuits are not affected by execution of this instruction. The instruction code is 0100.

In a SAMPLE operation, a snapshot of a value to be transferred from an input pin to the internal circuitry, or a value to be transferred from the internal circuitry to an output pin, is latched into the boundary scan register and read from the scan path. Snapshot latching is performed in synchronization with the rise of TCK in the Capture-DR state. Snapshot latching does not affect normal operation of the SH7622.

In a PRELOAD operation, an initial value is set in the parallel output latch of the boundary scan register from the scan path prior to the EXTEST instruction. Without a PRELOAD operation, when the EXTEST instruction was executed an undefined value would be output from the output pin until completion of the initial scan sequence (transfer to the output latch) (with the EXTEST instruction, the parallel output latch value is constantly output to the output pin).

EXTEST: This instruction is provided to test external circuitry when the SH7622 is mounted on a printed circuit board. When this instruction is executed, output pins are used to output test data (previously set by the SAMPLE/PRELOAD instruction) from the boundary scan register to the printed circuit board, and input pins are used to latch test results into the boundary scan register from the printed circuit board. If testing is carried out by using the EXTEST instruction N times, the Nth test data is scanned-in when test data (N–1) is scanned out.

Data loaded into the output pin boundary scan register in the Capture-DR state is not used for external circuit testing (it is replaced by a shift operation).

The instruction code is 0000.

24.5.2 Notes on Use

1. Boundary scan mode does not cover clock-related signals (EXTAL, XTAL, CKIO).
2. Boundary scan mode does not cover reset-related signals (RESETP, RESETM).
3. Boundary scan mode does not cover H-UDI-related signals (TCK, TDI, TDO, TMS, TRST).
4. When a boundary scan test is carried out. Either input a clock to EXTAL from off-chip or operate the EXTAL resonator, and set the state in which the CPU clocks (I, B, and P clocks) operate constantly.

   The EXTAL frequency range is as follows:

   Minimum: 1 MHz

   Maximum: Maximum frequency for respective clock mode specified in the CPG section

   Set pins MD to the clock mode to be used.

   Drive RESETP low (for approximately 100 µs with external clock input, or approximately 10 ms when using the oscillator). Then perform a boundary scan test with RESETP low or high.
5. Fix the ASEMD0 pin low.

24.6 Notes on Use

1. An H-UDI command other than an H-UDI interrupt, once set, will not be modified as long as another command is not re-issued from the H-UDI. An H-UDI interrupt command, however, will be changed to a bypass command once set.
2. Because chip operations are suspended in standby mode, H-UDI commands are not accepted. However, the TAP controller remains in operation at this time.
3. The H-UDI is used for emulator connection. Therefore, H-UDI functions cannot be used when using an emulator.
4. If the function is not used and open processing is required for the relevant pins, it is essential to check that the corresponding port control register setting is “pull-up MOS on”.

24.7 Advanced User Debugger (AUD)

The AUD is a function exclusively for use by an emulator. Refer to the User's Manual for the relevant emulator for details of the AUD. If the function is not used and open processing is required for the relevant pins, it is essential to check that the corresponding port control register setting is “pull-up MOS on”.

Section 25 Electrical Characteristics (80 MHz)

25.1 Absolute Maximum Ratings

Table 25.1 shows the absolute maximum ratings.

Table 25.1 Absolute Maximum Ratings

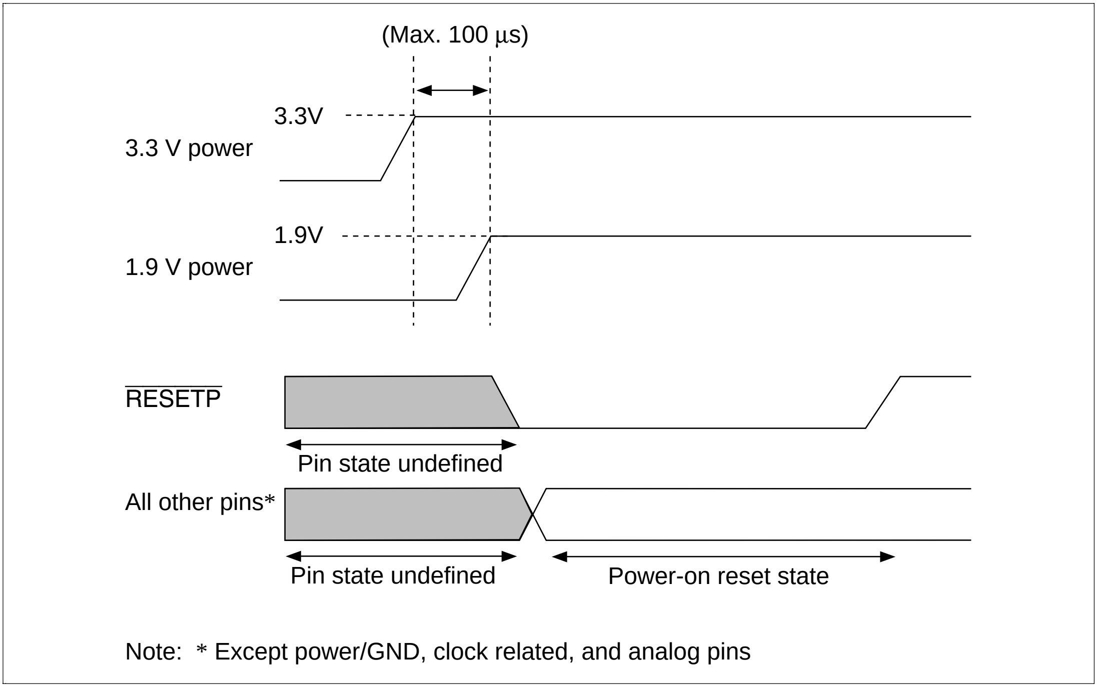
Item	Symbol	Rating	Unit
Power supply voltage (I/O)	$V_{CC}Q$	–0.3 to 4.2	V
Power supply voltage (internal)	V_{CC} V_{CC} – PLL1 V_{CC} – PLL2	–0.3 to 2.5	V
Input voltage (except port L)	Vin	–0.3 to $V_{CC}Q$ + 0.3	V
Input voltage (port L)	Vin	–0.3 to AV_{CC} + 0.3	V
Analog power-supply voltage	AV_{CC}	–0.3 to 4.6	V
Analog input voltage	V_{AN}	–0.3 to AV_{CC} + 0.3	V
Operating temperature	Topr	–20 to 75	°C
Storage temperature	Tstg	–55 to 125	°C

1. Exceeding the absolute maximum ratings may permanently damage the chip.
2. Order of turning on 1.9 V power (Vcc, Vcc-PLL1, Vcc-PLL2) and 3.3 V power (VccQ, AVcc):

   (1) First turn on the 3.3 V power, then turn on the 1.9 V power within 100 µs. This interval should be as short as possible.

   (2) Until voltage is applied to all power supplies and a low level is input at the $\overline{\text{RESETP}}$ pin, internal circuits remain unsettled, and so pin states are also undefined. The system design must ensure that these undefined states do not cause erroneous system operation.

Waveforms at power-on are shown in the following figure.



Note: * Except power/GND, clock related, and analog pins

Power-On Sequence

3. Power-off order

   (1) Reversing the order of powering-on, first turn off the 1.9 V power, then turn off the 3.3 V power within 100 µs. This interval should be as short as possible.

   (2) Pin states are undefined while only the 1.9 V power is off. The system design must ensure that these undefined states do not cause erroneous system operation.

25.2 DC Characteristics

Tables 25.2 and 25.3 list DC characteristics.

Table 25.2 DC Characteristics (1) [Common Items] (T_a = –20 to 75°C)

Item		Symbol	Min	Typ	Max	Unit	Measurement Conditions
Power supply voltage		$V_{CC}Q$	3.0	3.3	3.6	V	
		V_{CC}, V_{CC} – PLL1 V_{CC} – PLL2	1.75	1.9	2.05	V	
Current dissipation	Normal operation	I_{CC}	—	120	240	mA	V_{CC} = 1.9 V Iϕ = 80 MHz
		$I_{CC}Q$	—	20	60	mA	$V_{CC}Q$ = 3.3 V Bϕ = 33 MHz
	In standby mode	I_{stby}	—	100	300	µA	T_a = 25°C $V_{CC}Q$ = 3.3 V V_{CC} = 1.9 V
Input leak current	All input pins	$\lvert I_{in} \rvert$	—	—	1.0	µA	V_{in} = 0.5 to $V_{CC}Q$ – 0.5 V
Three-state leak current	I/O, all output pins (off condition)	$\lvert I_{TSI} \rvert$	—	—	1.0	µA	Vin = 0.5 to $V_{CC}Q$ – 0.5 V
Pull-up resistance	Port pin	P_{pull}	30	60	120	kΩ	
Pin capacity	All pins	C	—	—	10	pF	
Analog power-supply voltage		AV_{CC}	3.0	3.3	3.6	V	
Analog power-supply current	During A/D conversion	AI_{CC}	—	0.8	2	mA	
	Idle		—	0.01	5.0	µA	

Table 25.2 DC Characteristics (2-a) [Excluding USB-Related Pins] (T_a = –20 to 75°C)

Item		Symbol	Min	Typ	Max	Unit	Measurement Conditions
Input high voltage	$\overline{\text{RESETP}}$, $\overline{\text{RESETM}}$, NMI	V_{IH}	$V_{CC}Q \times 0.9$	—	$V_{CC}Q + 0.3$	V	
	$\overline{\text{BREQ}}$, IRQ7 to IRQ0, MD0 to MD4		$V_{CC}Q – 0.4$	—	$V_{CC}Q + 0.3$	V	
	EXTAL, CKIO		$V_{CC}Q – 0.4$	—	$V_{CC}Q + 0.3$	V	
	Port L		2.0	—	$AV_{CC} + 0.3$	V	
	Other input pins		2.0	—	$V_{CC}Q + 0.3$	V	
Input low voltage	$\overline{\text{RESETP}}$, $\overline{\text{RESETM}}$, NMI	V_{IL}	–0.3	—	$V_{CC}Q \times 0.1$	V	
	$\overline{\text{BREQ}}$, IRQ7 to IRQ0, MD0 to MD4		–0.3	—	$V_{CC}Q \times 0.2$	V	
	Port L		–0.3	—	$AV_{CC} \times 0.2$	V	
	Other input pins		–0.3	—	$V_{CC}Q \times 0.2$	V	
Output high voltage	All output pins	V_{OH}	2.4	—	—	V	$V_{CC}Q$ = 3.0 V I_{OH} = –200 µA
			2.0	—	—	V	$V_{CC}Q$ = 3.0 V I_{OH} = –2 mA
Output low voltage	All output pins	V_{OL}	—	—	0.55	V	$V_{CC}Q$ = 3.6 V I_{OL} = 1.6 mA

Notes: 1. The V_{CC} pins must be connected to V_{CC}, and the V_{SS} pins to V_{SS}.

2. AV_{CC} must satisfy the condition: $V_{CC}Q – 0.2\ V \le AV_{CC} \le V_{CC}Q + 0.2\ V$. Do not leave the AV_{CC} and AV_{SS} pins open if the A/D converter is not used; connect AV_{CC} to $V_{CC}Q$, and AV_{SS} to $V_{SS}Q$.

3. Current dissipation values are for V_{IH}min = $V_{CC}Q$ – 0.5 V and V_{IL}max = 0.5 V with all output pins unloaded.

Table 25.2 DC Characteristics (2-b) [USB-Related Pins*] (T_a = –20 to 75°C)

Item	Symbol	Min	Typ	Max	Unit	Measurement Conditions
Power supply voltage	$V_{CC}Q$	3.0	3.3	3.6	V	
Input high voltage	V_{IH}	2.0	—	$V_{CC}Q$ + 0.3	V	
Input low voltage	V_{IL}	–0.3	—	$V_{CC}Q \times 0.2$	V	
Output higt voltage	V_{OH}	2.4	—	—	V	$V_{CC}Q$ = 3.0 V I_{OH} = –200 µA
		2.0	—	—		$V_{CC}Q$ = 3.0 V I_{OH} = –2 mA
Output low voltage	V_{OL}	—	—	0.55	V	$V_{CC}Q$ = 3.6 V I_{OL} = 1.6 mA

Note: * Pins XVDATA, DPLS, DMNS, TXDPLS, TXDMNS, TXENL, VBUS, SUSPND, and UCLK

Table 25.3 Permitted Output Current Values ($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Item	Symbol	Min	Typ	Max	Unit
Output low-level permissible current (per pin)	I_{OL}	—	—	2.0	mA
Output low-level permissible current (total)	$\sum I_{OL}$	—	—	120	mA
Output high-level permissible current (per pin)	$-I_{OH}$	—	—	2.0	mA
Output high-level permissible current (total)	$\sum (-I_{OH})$	—	—	40	mA

Caution: To ensure LSI reliability, do not exceed the value for output current given in table 25.3.

25.3 AC Characteristics

In general, inputting for this LSI should be clock synchronous. Keep the setup and hold times for each input signal unless otherwise specified.

Table 25.4 Maximum Operating Frequencies ($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

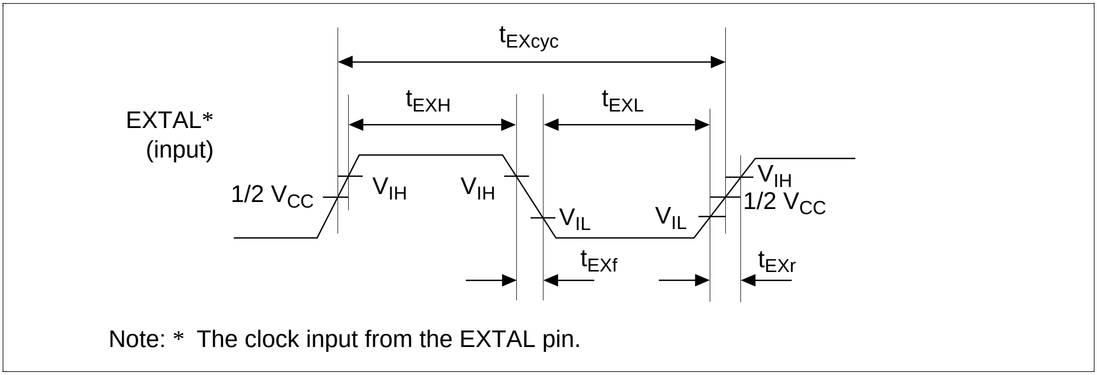
Item		Symbol	Min	Typ	Max	Unit	Remarks
Operating frequency	CPU, cache, (Iϕ)	f	20	—	80	MHz	
	External bus (Bϕ)		5	—	26.6		
	Peripheral module (Pϕ)		5	—	26.6		

25.3.1 Clock Timing

Table 25.5 Clock Timing ($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Item	Symbol	Min	Max	Unit	Figure
EXTAL clock input frequency	f_{EX}	5	26.6	MHz	25.1
EXTAL clock input cycle time	t_{EXcyc}	37.5	200	ns	
EXTAL clock input low pulse width	t_{EXL}	7	—	ns	
EXTAL clock input high pulse width	t_{EXH}	7	—	ns	
EXTAL clock input rise time	t_{EXr}	—	4	ns	
EXTAL clock input fall time	t_{EXf}	—	4	ns	
CKIO clock input frequency	f_{CKI}	20	26.6	MHz	25.2 (1)
CKIO clock input cycle time	t_{CKIcyc}	37.5	50	ns	
CKIO clock input low pulse width	t_{CKIL}	7	—	ns	
CKIO clock input high pulse width	t_{CKIH}	7	—	ns	
CKIO clock input rise time	t_{CKIR}	—	3	ns	
CKIO clock input fall time	t_{CKIF}	—	3	ns	
CKIO clock input frequency	f_{CKI}	5	26.6	MHz	25.2 (2)
CKIO clock input cycle time	t_{CKIcyc}	37.5	200	ns	
CKIO clock input low pulse width	t_{CKIL}	8	—	ns	
CKIO clock input high pulse width	t_{CKIH}	8	—	ns	
CKIO clock input rise time	t_{CKIr}	—	6	ns	
CKIO clock input fall time	t_{CKIf}	—	6	ns	
Power-on oscillation settling time	t_{OSC1}	10	—	ms	25.3
$\overline{\text{RESETP}}$ setup time	t_{RESPS}	20	—	ns	25.3, 25.4
$\overline{\text{RESETM}}$ setup time	t_{RESMS}	0	—	ns	
$\overline{\text{RESETP}}$ assert time	t_{RESPW}	20	—	tcyc	
$\overline{\text{RESETM}}$ assert time	t_{RESMW}	20	—	tcyc	
Standby return oscillation settling time 1	t_{OSC2}	10	—	ms	25.4
Standby return oscillation settling time 2	t_{OSC3}	10	—	ms	25.5
PLL synchronization settling time	t_{PLL}	100	—	µs	25.6

Note: The maximum internal data bus operating frequency is 80 MHz. Set the multiplication factor in line with this condition.



Note: * The clock input from the EXTAL pin.

Figure 25.1 EXTAL Clock Input Timing

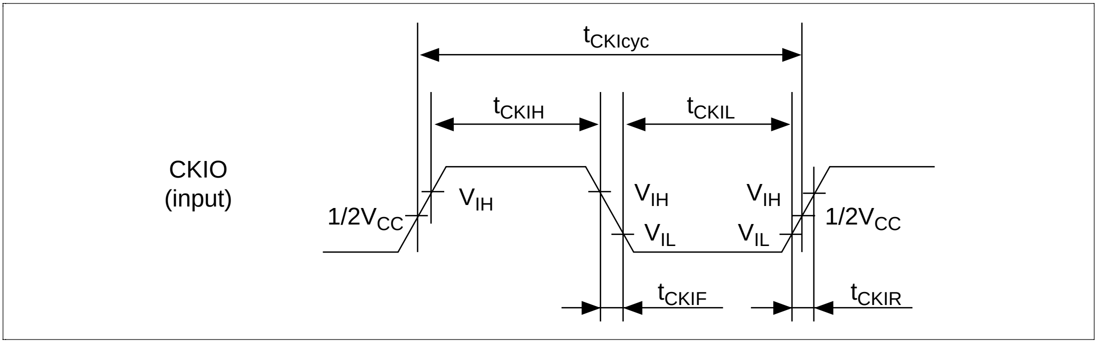


Figure 25.2(1) CKIO Clock Input Timing

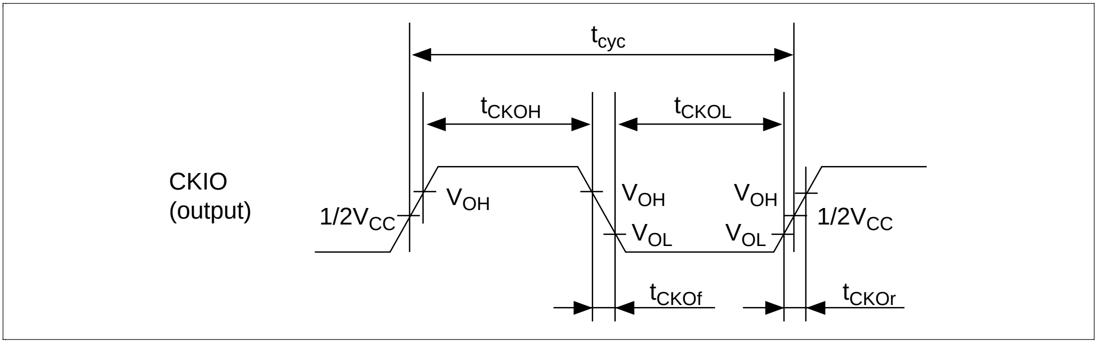


Figure 25.2(2) CKIO Clock Output Timing

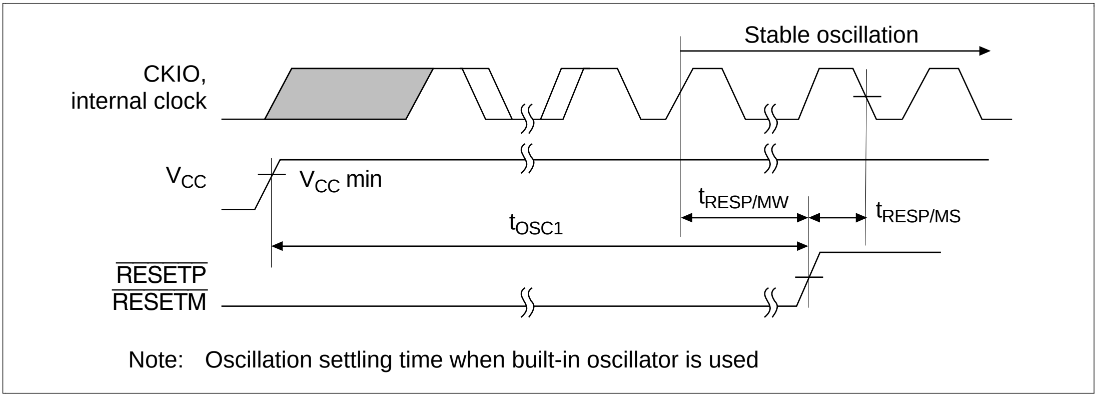


Figure 25.3 Power-on Oscillation Settling Time

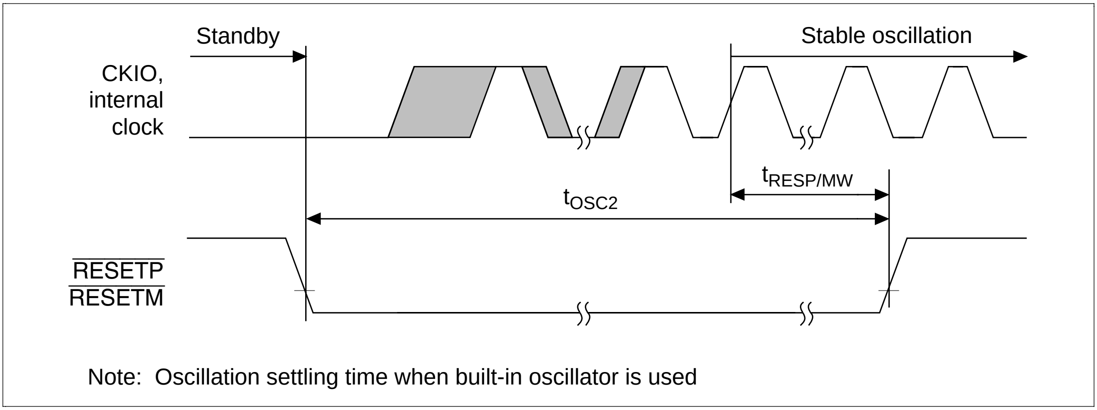


Figure 25.4 Oscillation Settling Time at Standby Return (Return by Reset)

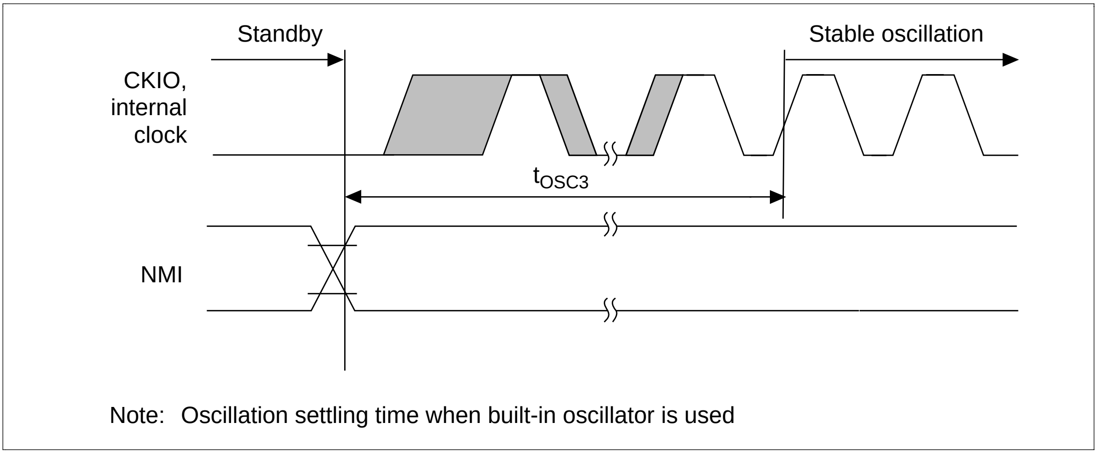
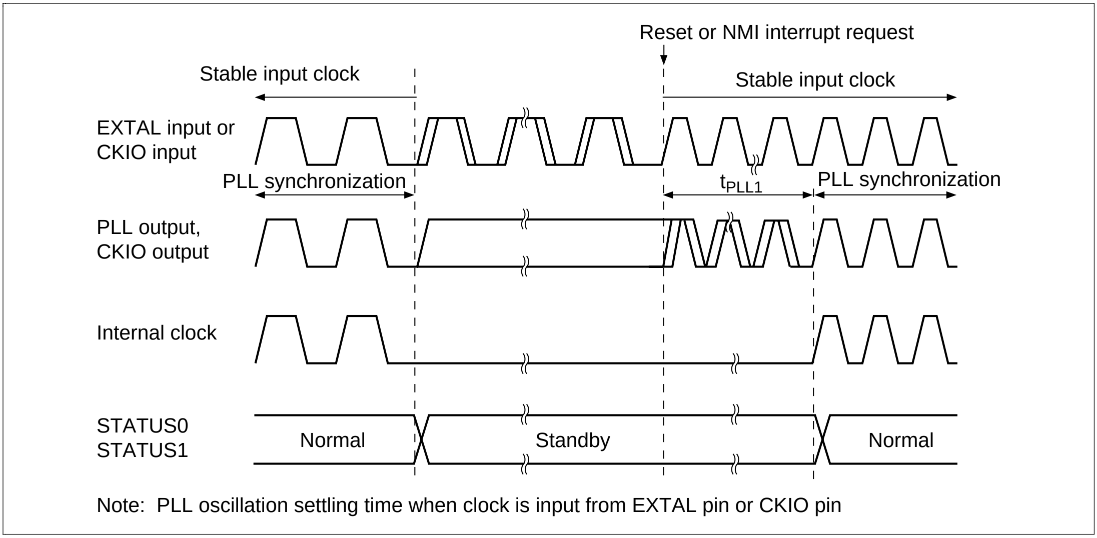

Standby
Stable oscillation
CKIO,
internal
clock
tOSC3
NMI
Note: Oscillation settling time when built-in oscillator is used

Figure 25.5 Oscillation Settling Time at Standby Return (Return by NMI)



Note: PLL oscillation settling time when clock is input from EXTAL pin or CKIO pin

Figure 25.6 PLL Synchronization Settling Time by Reset or NMI

25.3.2 Control Signal Timing

Table 25.6 Control Signal Timing
(V_{CC} = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

		26.6*2			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Figure**
$\overline{\text{RESETP}}$ pulse width	t_{RESPW}	20 *3	—	tcyc	25.7, 25.8
$\overline{\text{RESETP}}$ setup time*1	t_{RESPS}	23	—	ns	
$\overline{\text{RESETP}}$ hold time	t_{RESPH}	2	—	ns	
$\overline{\text{RESETM}}$ pulse width	t_{RESMW}	12 *4	—	tcyc	
$\overline{\text{RESETM}}$ setup time	t_{RESMS}	3	—	ns	
$\overline{\text{RESETM}}$ hold time	t_{RESMH}	34	—	ns	
$\overline{\text{BREQ}}$ setup time	t_{BREQS}	12	—	ns	25.10
$\overline{\text{BREQ}}$ hold time	t_{BREQH}	3	—	ns	
NMI setup time *1	t_{NMIS}	12	—	ns	25.8
NMI hold time	t_{NMIH}	4	—	ns	
IRQ7–IRQ0 setup time *1	t_{IRQS}	12	—	ns	
IRQ7–IRQ0 hold time	t_{IRQH}	4	—	ns	
$\overline{\text{IRQOUT}}$ delay time	t_{IRQOD}	—	14	ns	25.9
$\overline{\text{BACK}}$ delay time	t_{BACKD}	—	14	ns	25.10, 25.11
STATUS1, STATUS0 delay time	t_{STD}	—	18	ns	
Bus tri-state delay time 1	t_{BOFF1}	0	30	ns	
Bus tri-state delay time 2	t_{BOFF2}	0	30	ns	
Bus tri-state delay time 3	t_{BOFF3}	$2B_{cyc}$*5	—	ns	
Bus buffer-on time 1	t_{BON1}	0	30	ns	
Bus buffer-on time 2	t_{BON2}	0	30	ns	
Bus buffer-on time 3	t_{BON3}	$2B_{cyc}$*5	—	ns	

Notes: *1 $\overline{\text{RESETP}}$, NMI, and IRQ7–IRQ0 are asynchronous. Changes are detected at the clock fall when the setup shown is used. When the setup cannot be used, detection can be delayed until the next clock falls.

*2 The upper limit of the external bus clock is 26.6 MHz.

*3 In the standby mode, t_{RESPW} = t_{OSC2} (10 ms).
When the clock multiplication ratio is changed, t_{RESPW} = t_{PLL1} (100 µs).

*4 In the standby mode, t_{RESMW} = t_{OSC2} (10 ms).
When the clock multiplication ratio is changed, $\overline{\text{RESETM}}$ must be kept low until STATUS (0–1) changes to reset (HH).

*5 B_{cyc} is the external bus clock cycle (B clock cycle).

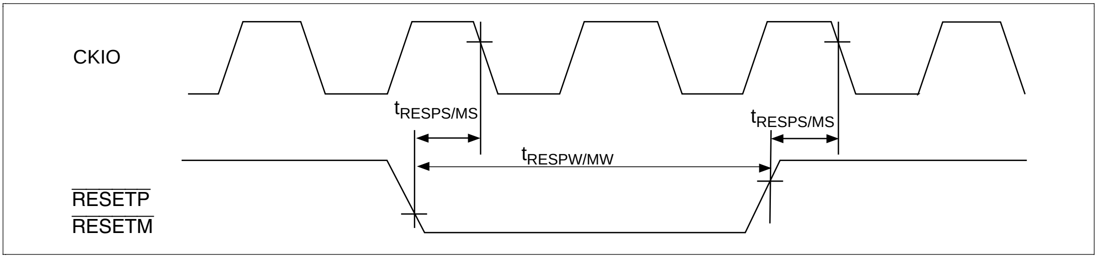


Figure 25.7 Reset Input Timing

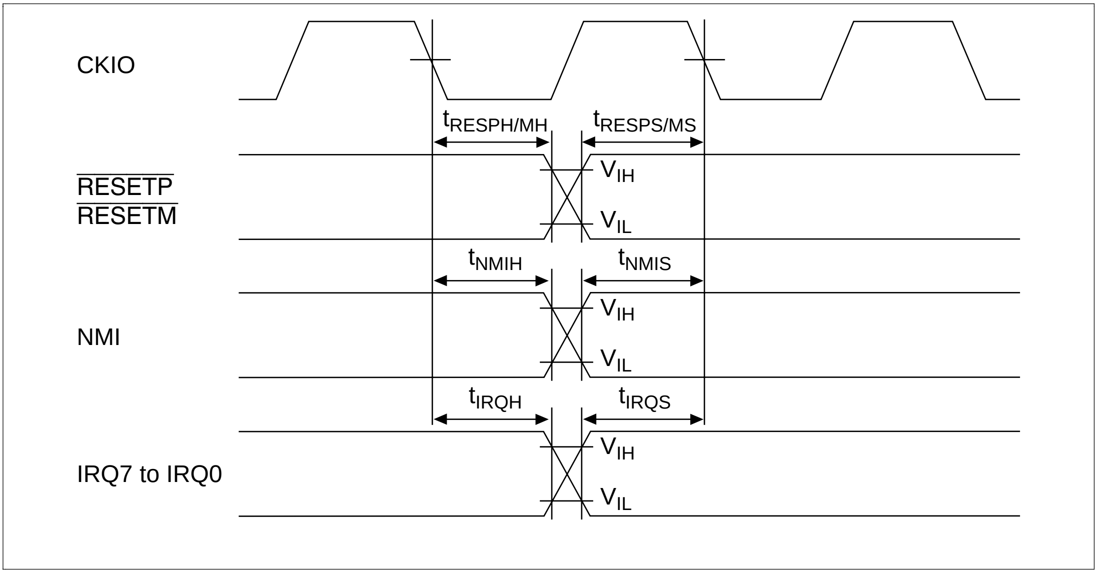


Figure 25.8 Interrupt Signal Input Timing

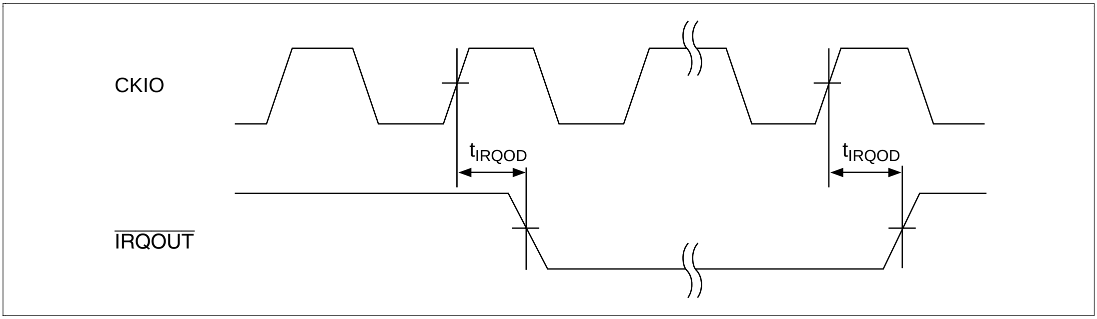


Figure 25.9 $\overline{\text{IRQOUT}}$ Timing

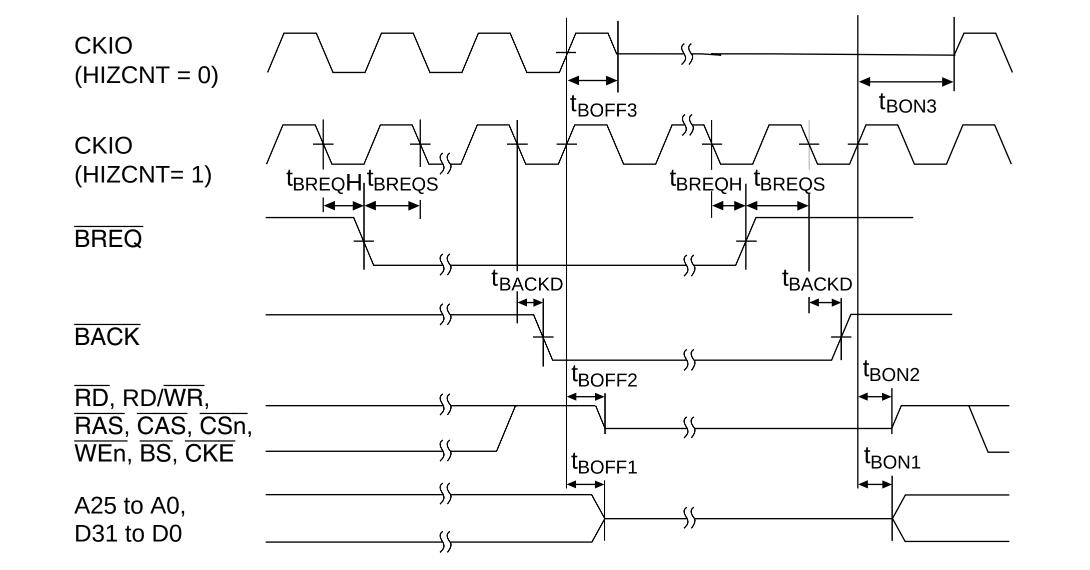


Figure 25.10 Bus Release Timing

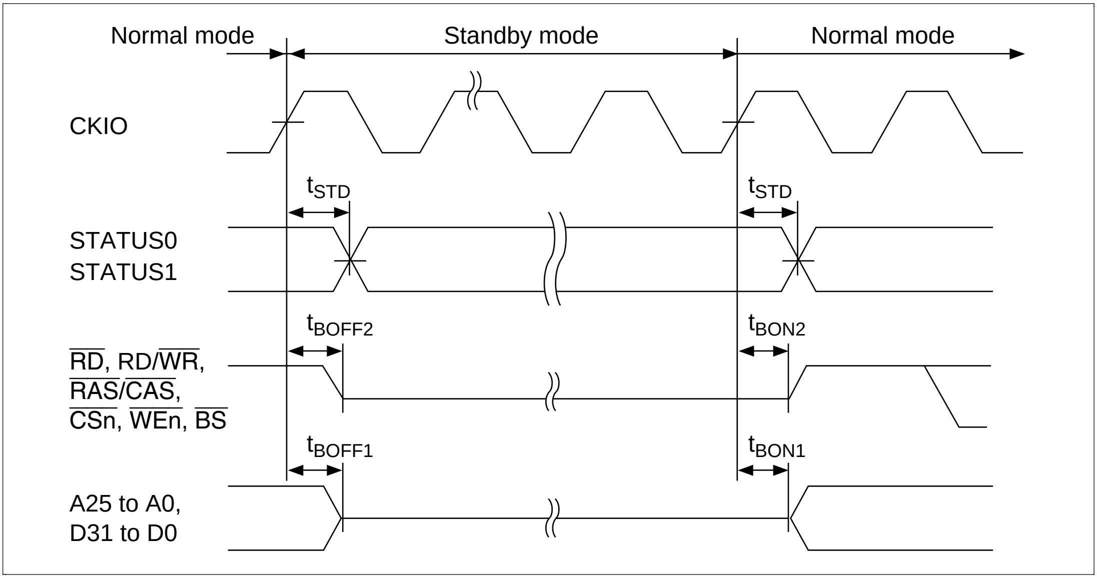


Figure 25.11 Pin Drive Timing at Standby

25.3.3 AC Bus Timing

Table 25.7 Bus Timing
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

		–26.6			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Figure**
Address delay time	t_{AD}	1	15	ns	25.12–25.32, 25.35
Address hold time	t_{AH}	10	—	ns	25.12–25.17
$\overline{\text{BS}}$ delay time	t_{BSD}	—	14	ns	25.12–25.32, 25.35
$\overline{\text{CS}}$ delay time 1	t_{CSD1}	1	14	ns	25.12–25.32, 25.35
$\overline{\text{CS}}$ delay time 2	t_{CSD2}	1	14	ns	25.12–25.17
Read/write delay time	t_{RWD}	2	12	ns	25.12–25.32, 25.35
Read/write hold time	t_{RWH}	0	—	ns	25.12–25.17
Read strobe delay time	t_{RSD}	—	12	ns	25.12–25.17
Read data setup time 1	t_{RDS1}	12	—	ns	25.12–25.17
Read data setup time 2	t_{RDS2}	7	—	ns	25.18–25.21
Read data hold time 1	t_{RDH1}	0	—	ns	25.12–25.17
Read data hold time 2	t_{RDH2}	2	—	ns	25.18–25.21, 25.26–25.30
Write enable delay time	t_{WED}	1	14	ns	25.12–25.14
Write data delay time 1	t_{WDD1}	—	17	ns	25.12–25.14
Write data delay time 2	t_{WDD2}	—	16	ns	25.22–25.25
Write data hold time 1	t_{WDH1}	2	—	ns	25.12–25.14
Write data hold time 2	t_{WDH2}	2	—	ns	25.22–25.25
Write data hold time 3	t_{WDH3}	2	—	ns	25.12–25.14
$\overline{\text{WAIT}}$ setup time	t_{WTS}	12	—	ns	25.13–25.17
$\overline{\text{WAIT}}$ setup time 2	t_{WTS2}	7	—	ns	25.14(2)
$\overline{\text{WAIT}}$ hold time	t_{WTH}	4	—	ns	25.13–25.17, 25.29
$\overline{\text{RAS}}$ delay time 2	t_{RASD2}	1	15	ns	25.18–25.35
$\overline{\text{CAS}}$ delay time 2	t_{CASD2}	1	15	ns	25.18–25.35
$\overline{\text{DQM}}$ delay time	t_{DQMD}	1	15	ns	25.18–25.32
DACK delay time 1	t_{DAKD1}	—	15	ns	25.12–25.32

25.3.4 Basic Timing

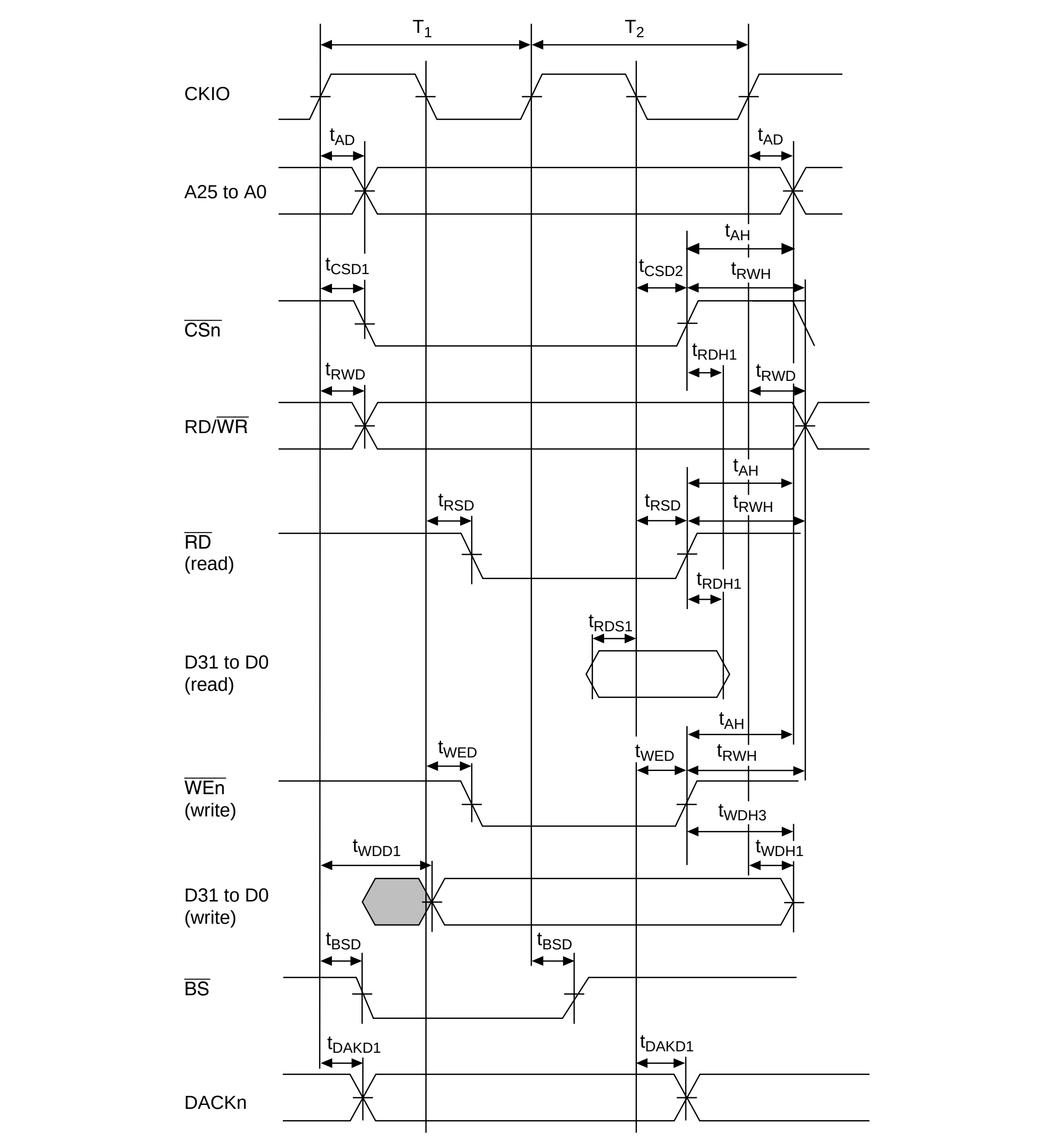


Figure 25.12 Basic Bus Cycle (No Wait)

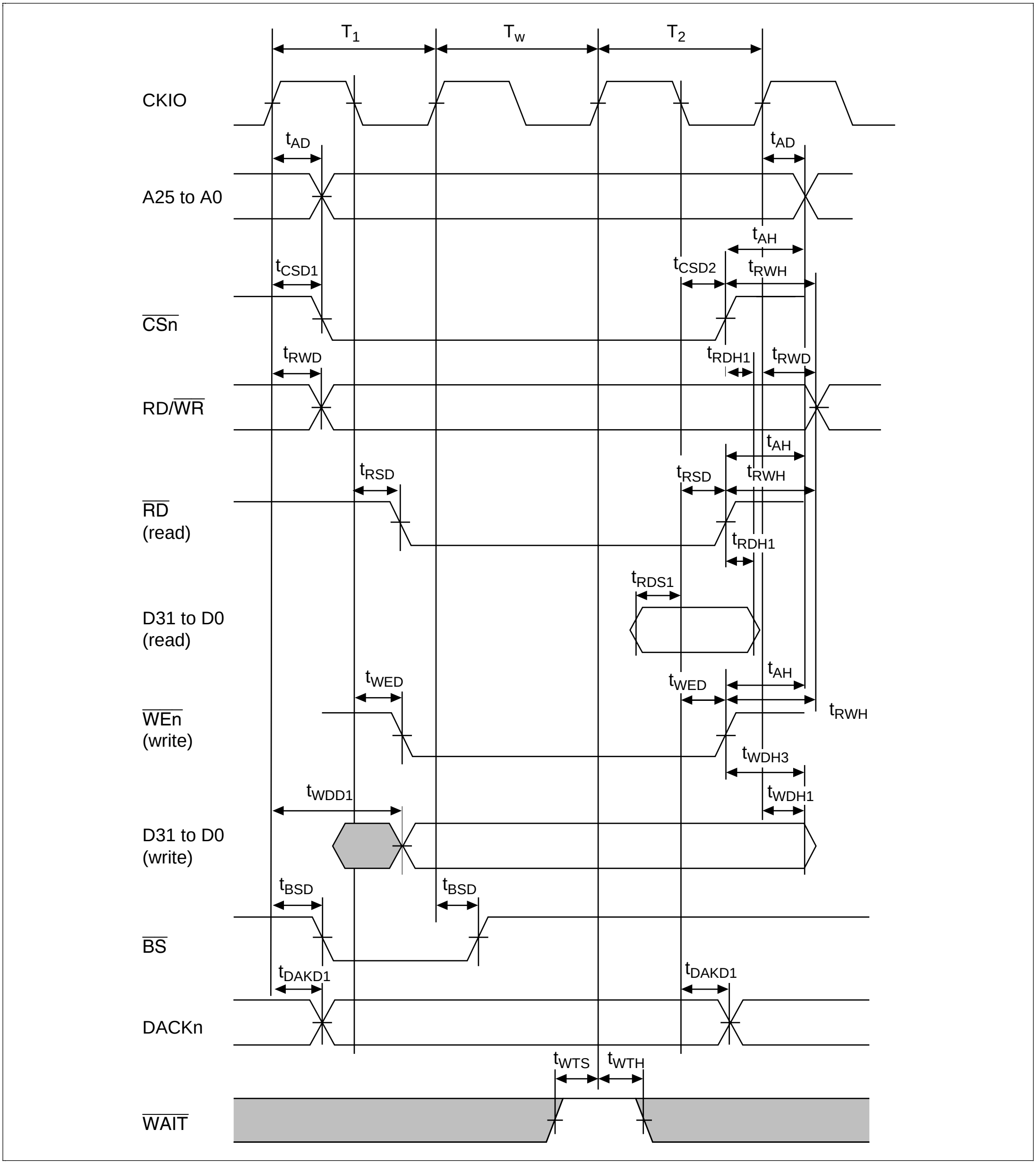


Figure 25.13 Basic Bus Cycle (One Wait)

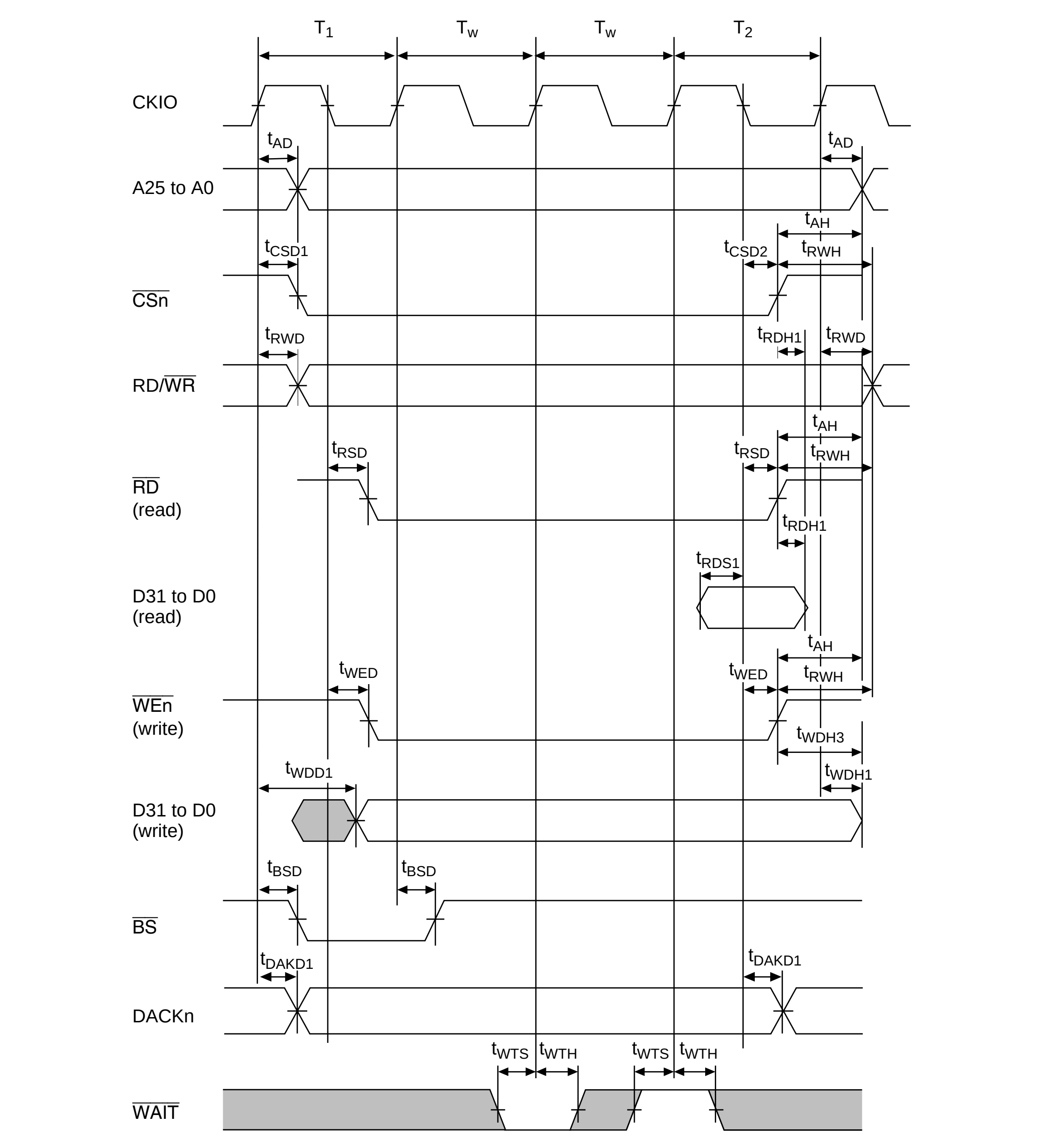


Figure 25.14 (1) Basic Bus Cycle (External Wait, WAITSEL = 0)

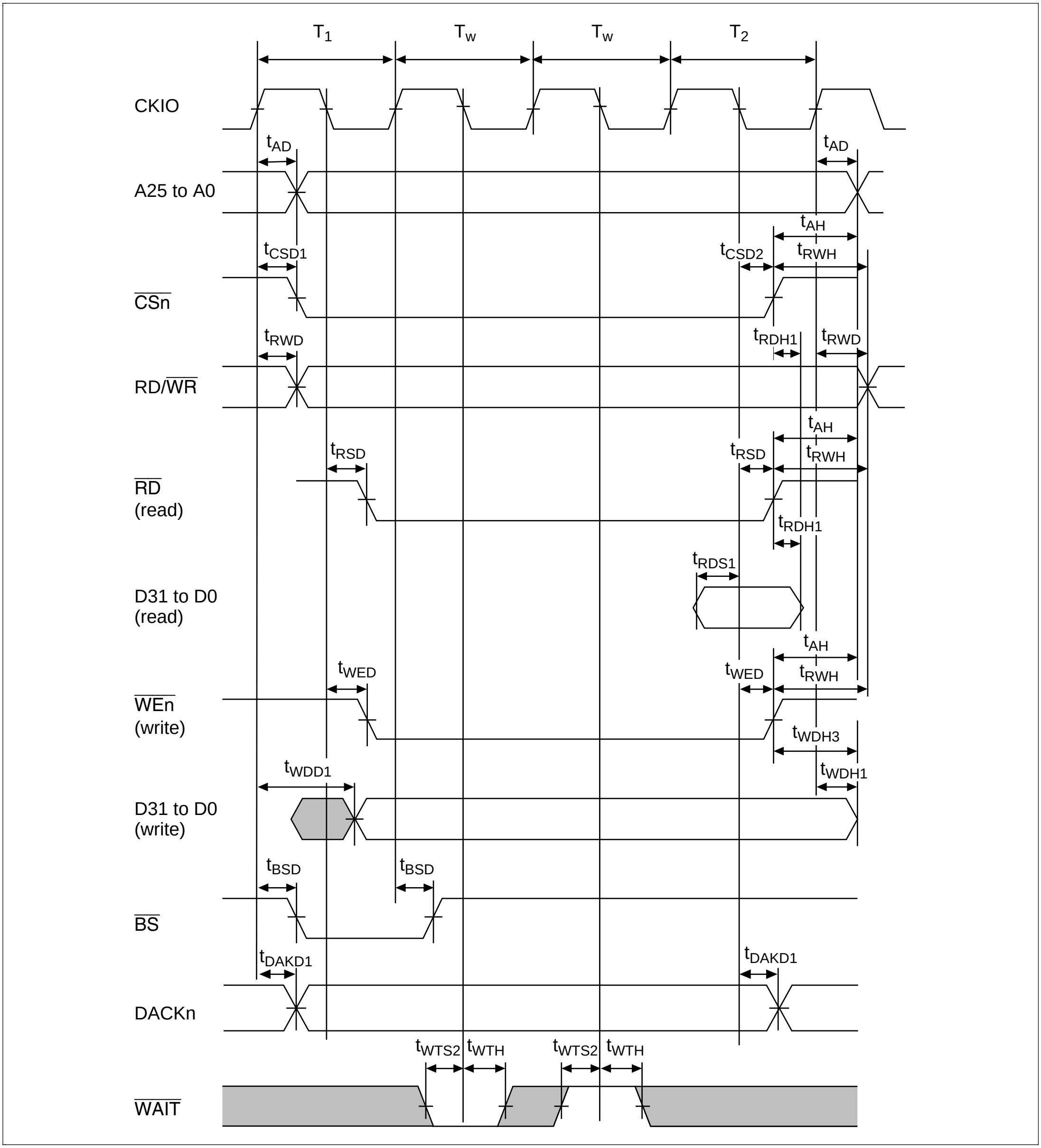
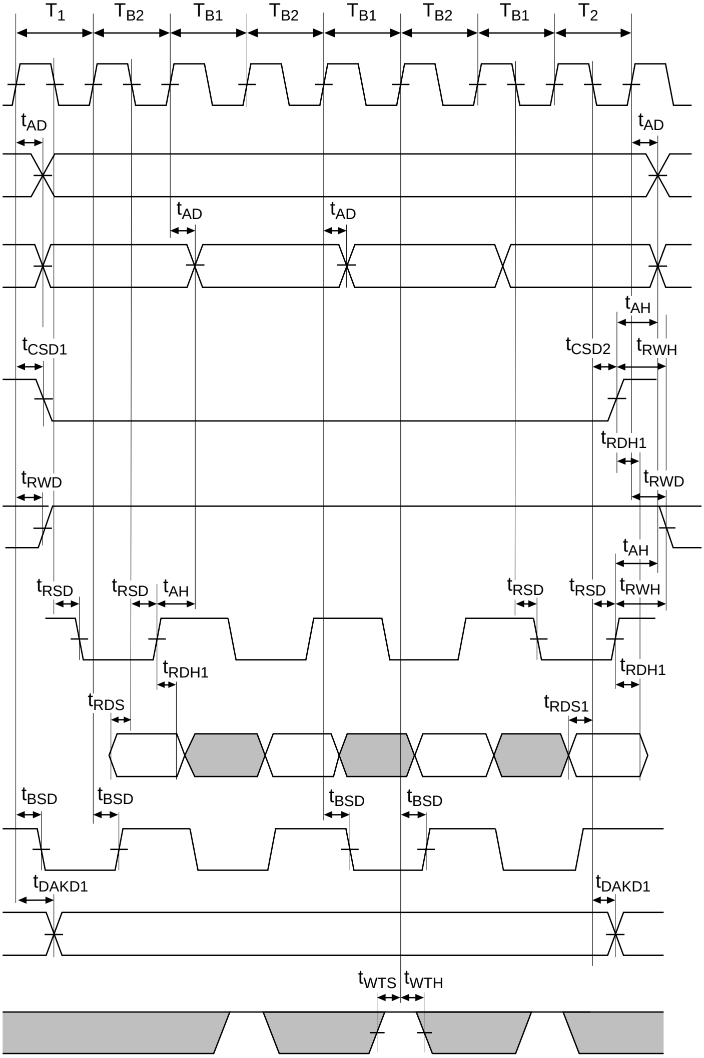


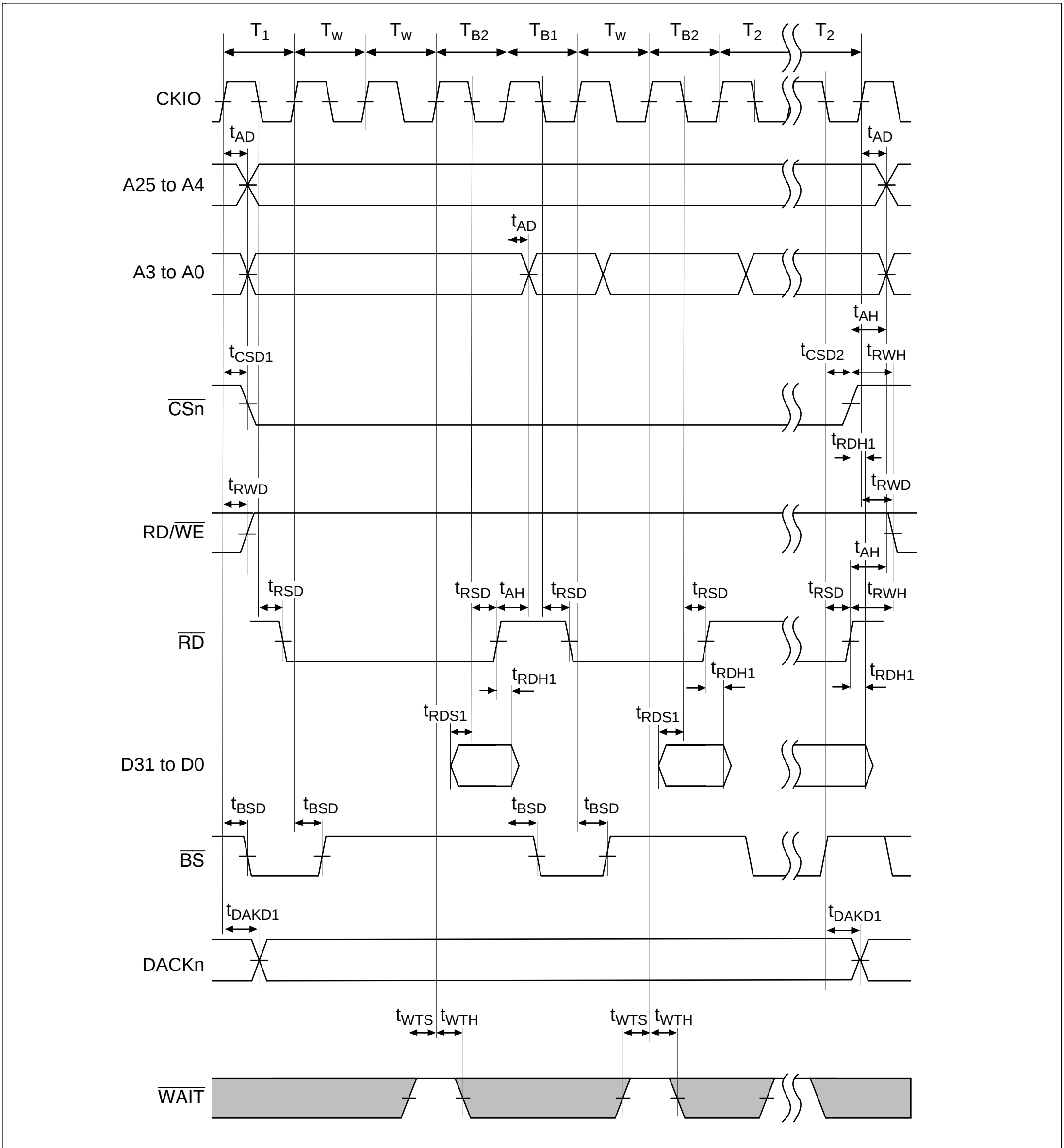
Figure 25.14 (2) Basic Bus Cycle (External Wait, WAITSEL = 1)

25.3.5 Burst ROM Timing



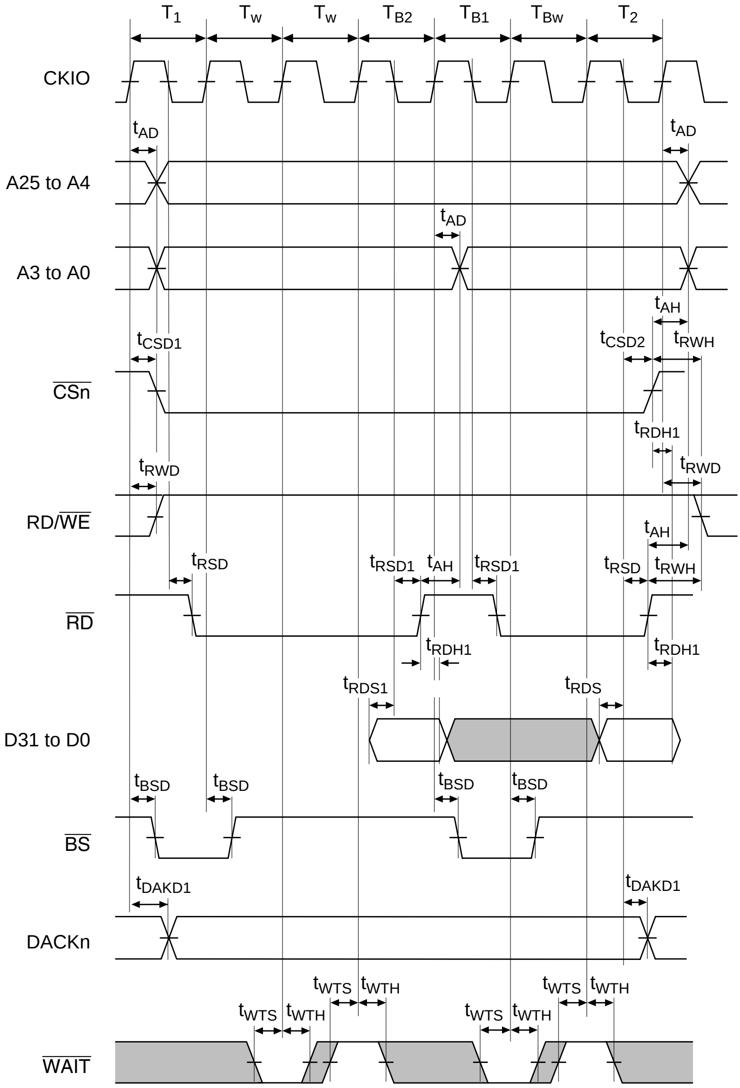
Note: In the write cycle, the basic bus cycle is performed.

Figure 25.15 Burst ROM Bus Cycle (No Wait)



Note: In the write cycle, the basic bus cycle is performed.

Figure 25.16 Burst ROM Bus Cycle (Two Waits)



Note: In the write cycle, the basic bus cycle is performed.

Figure 25.17 Burst ROM Bus Cycle (External Wait, WAITSEL = 0)

25.3.6 Synchronous DRAM Timing

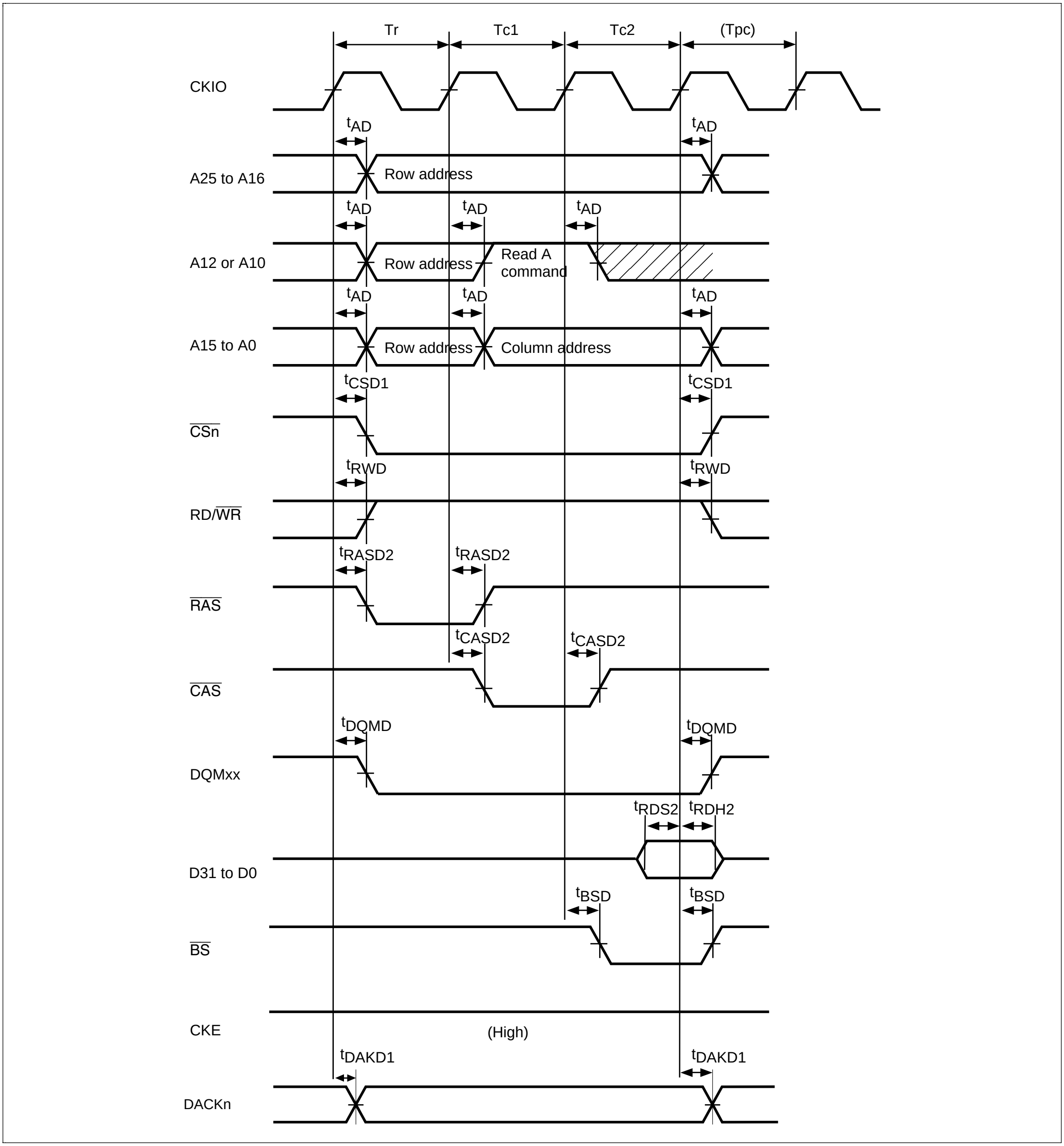


Figure 25.18 Synchronous DRAM Read Bus Cycle (RCD = 0, CAS Latency = 1, TPC = 0)

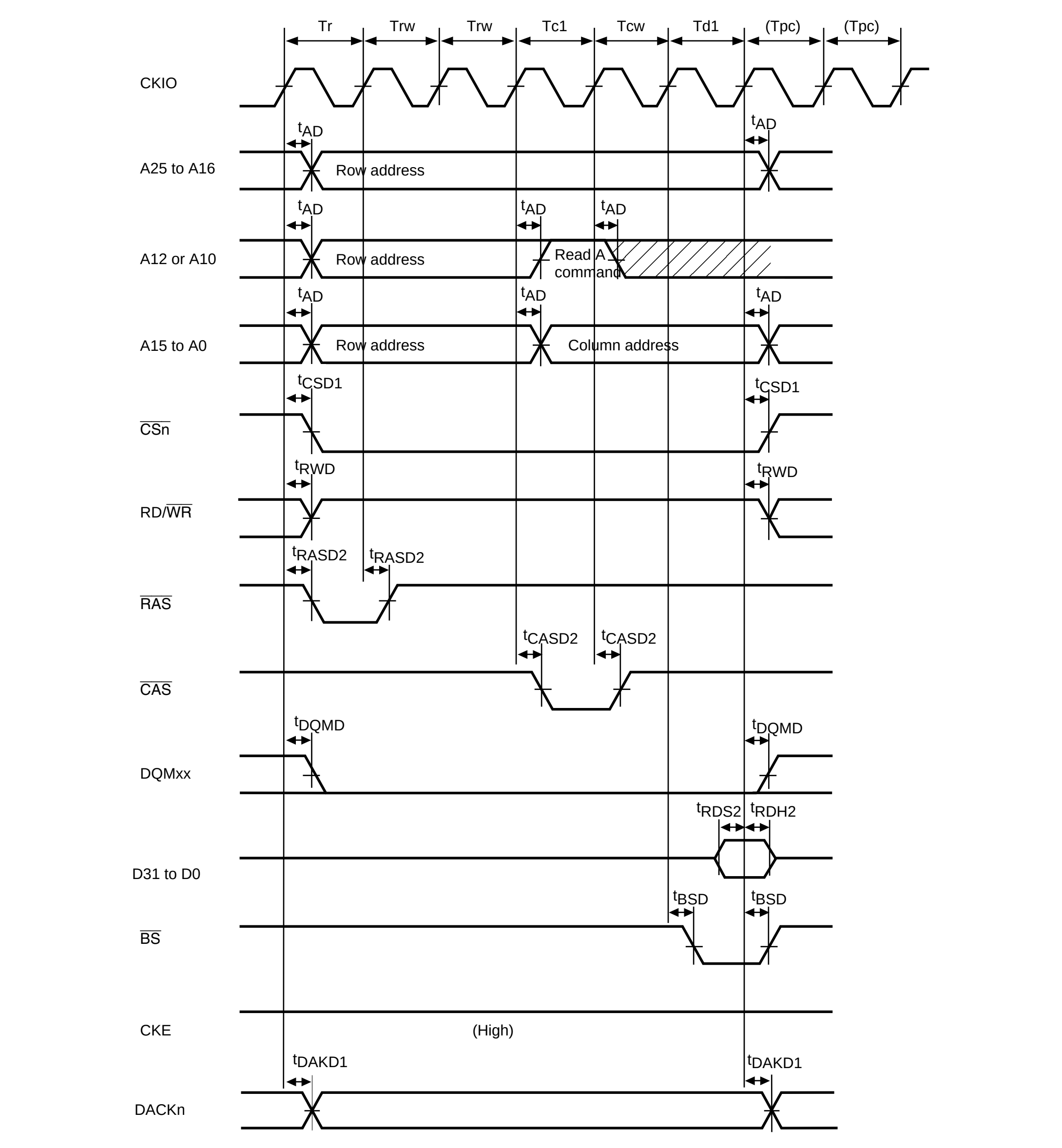


Figure 25.19 Synchronous DRAM Read Bus Cycle (RCD = 2, CAS Latency = 2, TPC = 1)

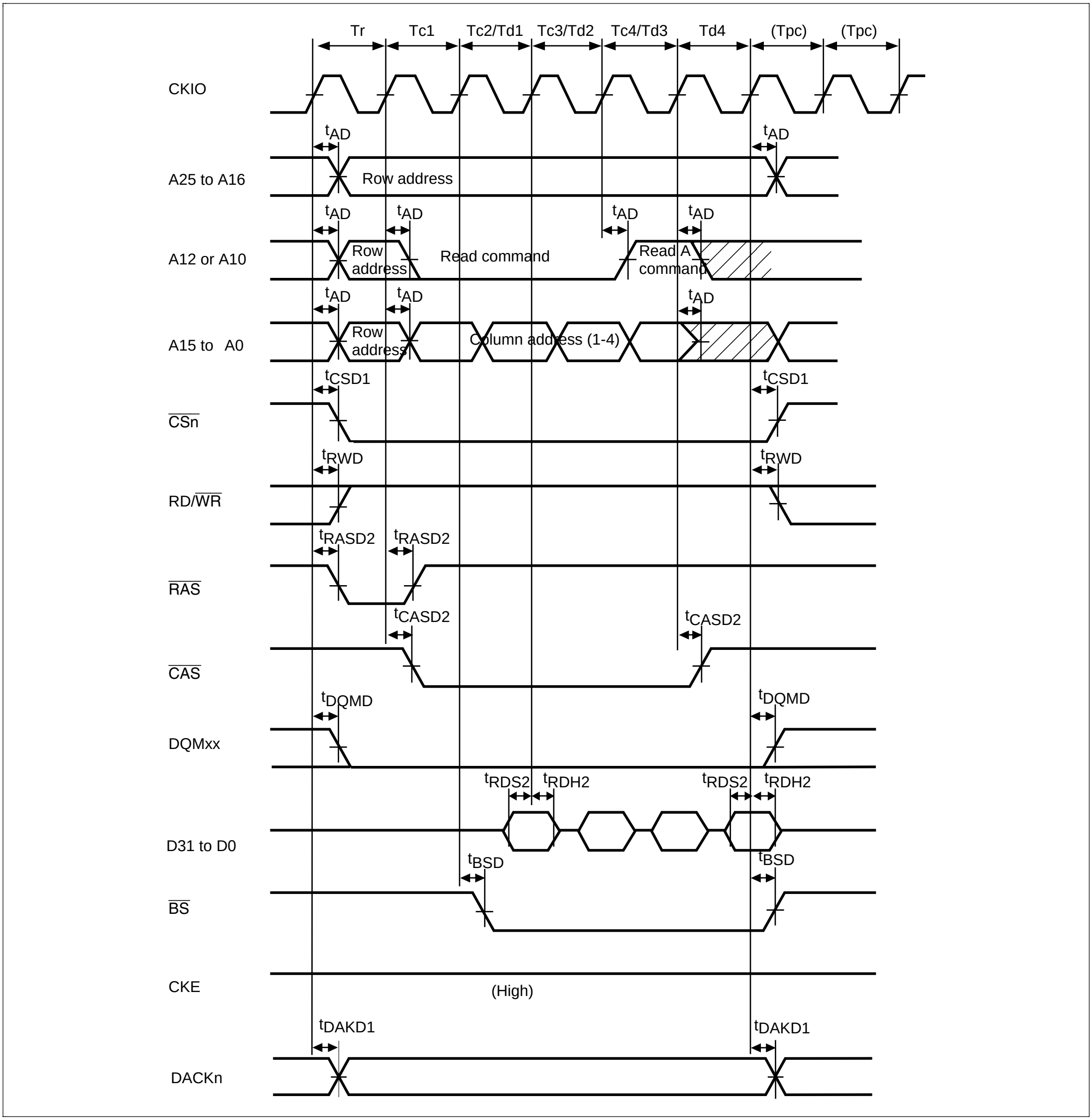


Figure 25.20 Synchronous DRAM Read Bus Cycle
(Burst Read (Single Read × 4), RCD = 0, CAS Latency = 1, TPC = 1)

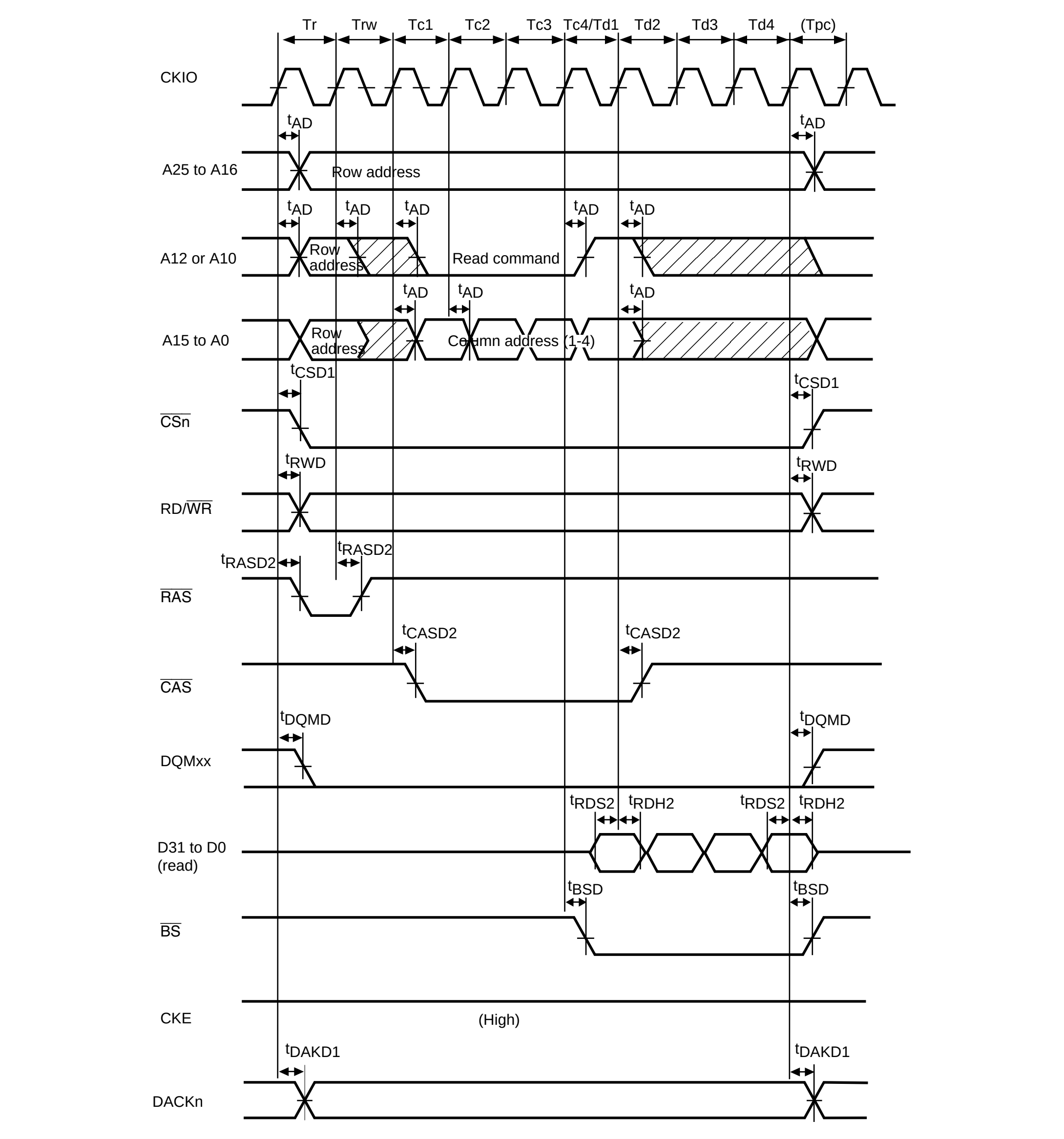


Figure 25.21 Synchronous DRAM Read Bus Cycle (Burst Read (Single Read × 4), RCD = 1, CAS Latency = 3, TPC = 0)

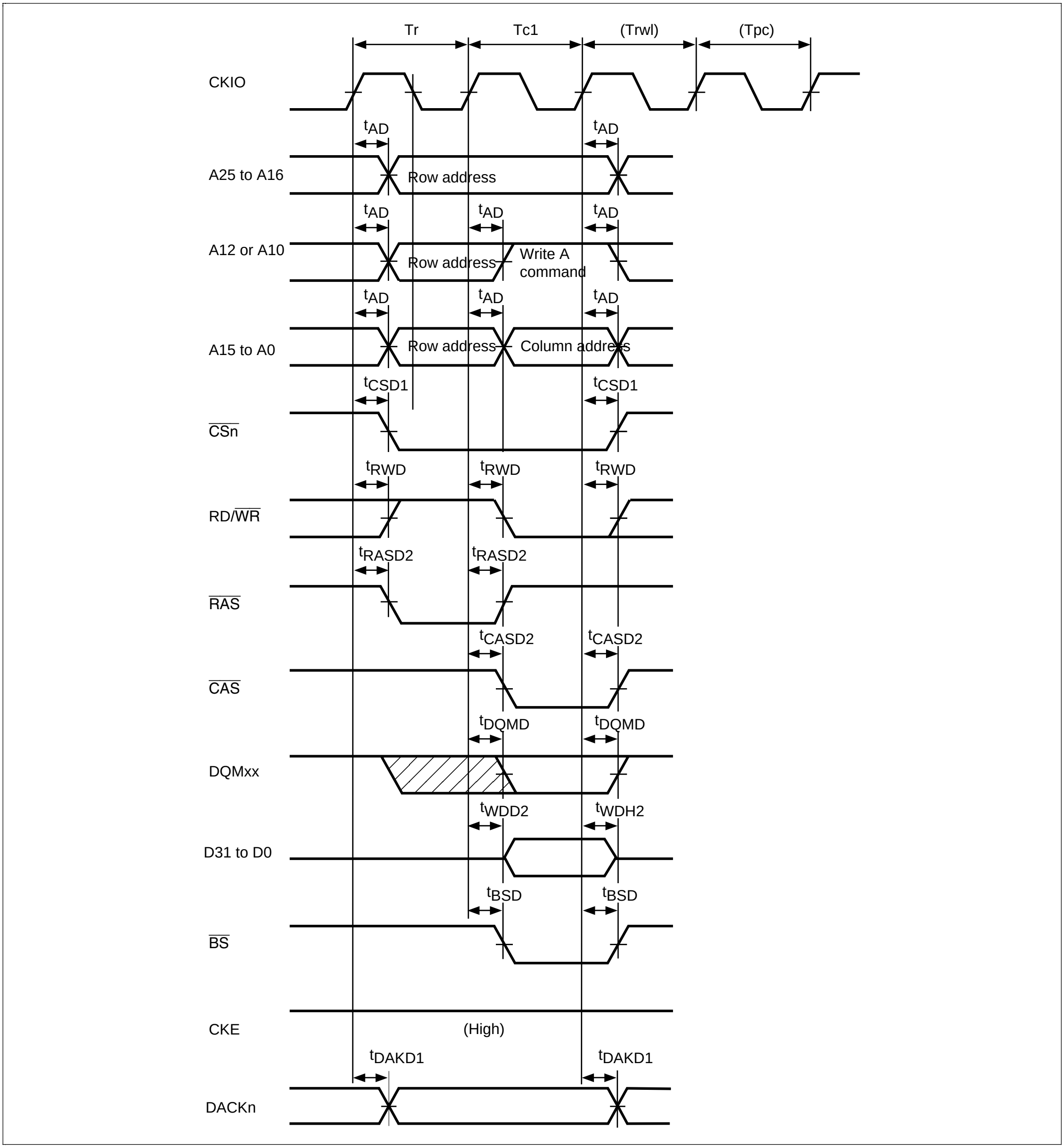


Figure 25.22 Synchronous DRAM Write Bus Cycle (RCD = 0, TPC = 0, TRWL = 0)

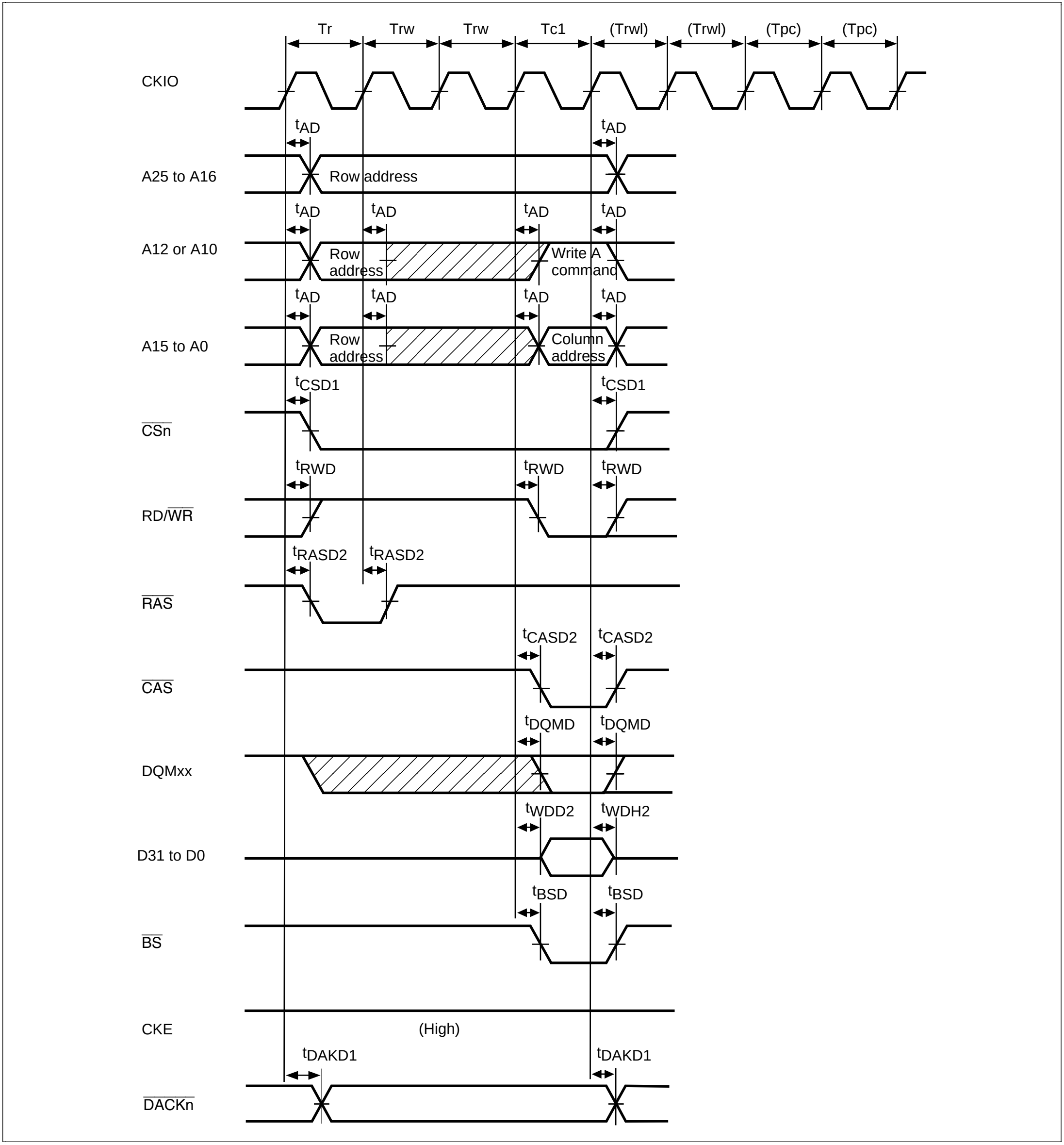


Figure 25.23 Synchronous DRAM Write Bus Cycle (RCD = 2, TPC = 1, TRWL = 1)

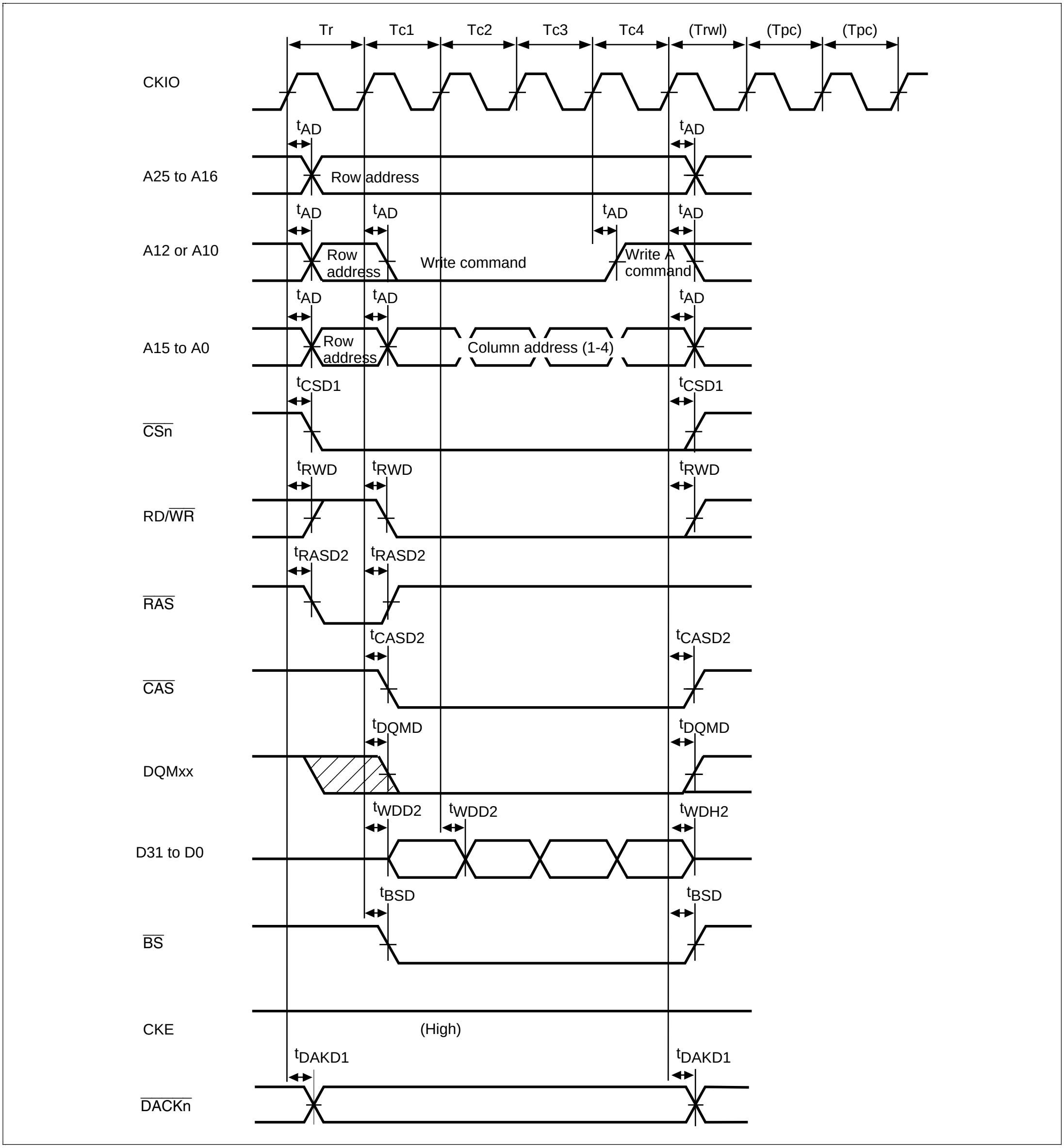


Figure 25.24 Synchronous DRAM Write Bus Cycle (Burst Mode (Single Write × 4), RCD = 0, TPC = 1, TRWL = 0)

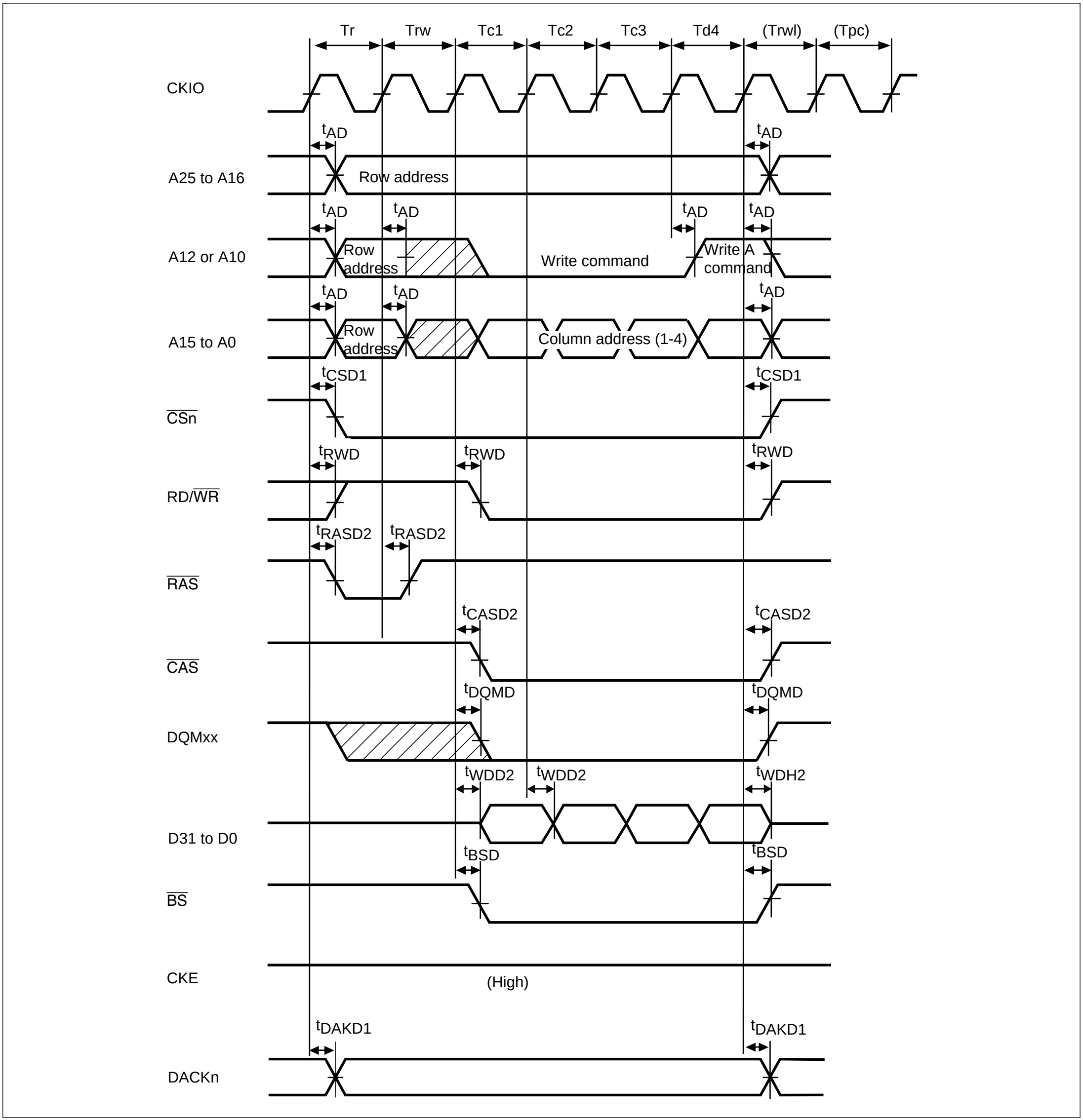


Figure 25.25 Synchronous DRAM Write Bus Cycle (Burst Mode (Single Write × 4), RCD = 1, TPC = 0, TRWL = 0)

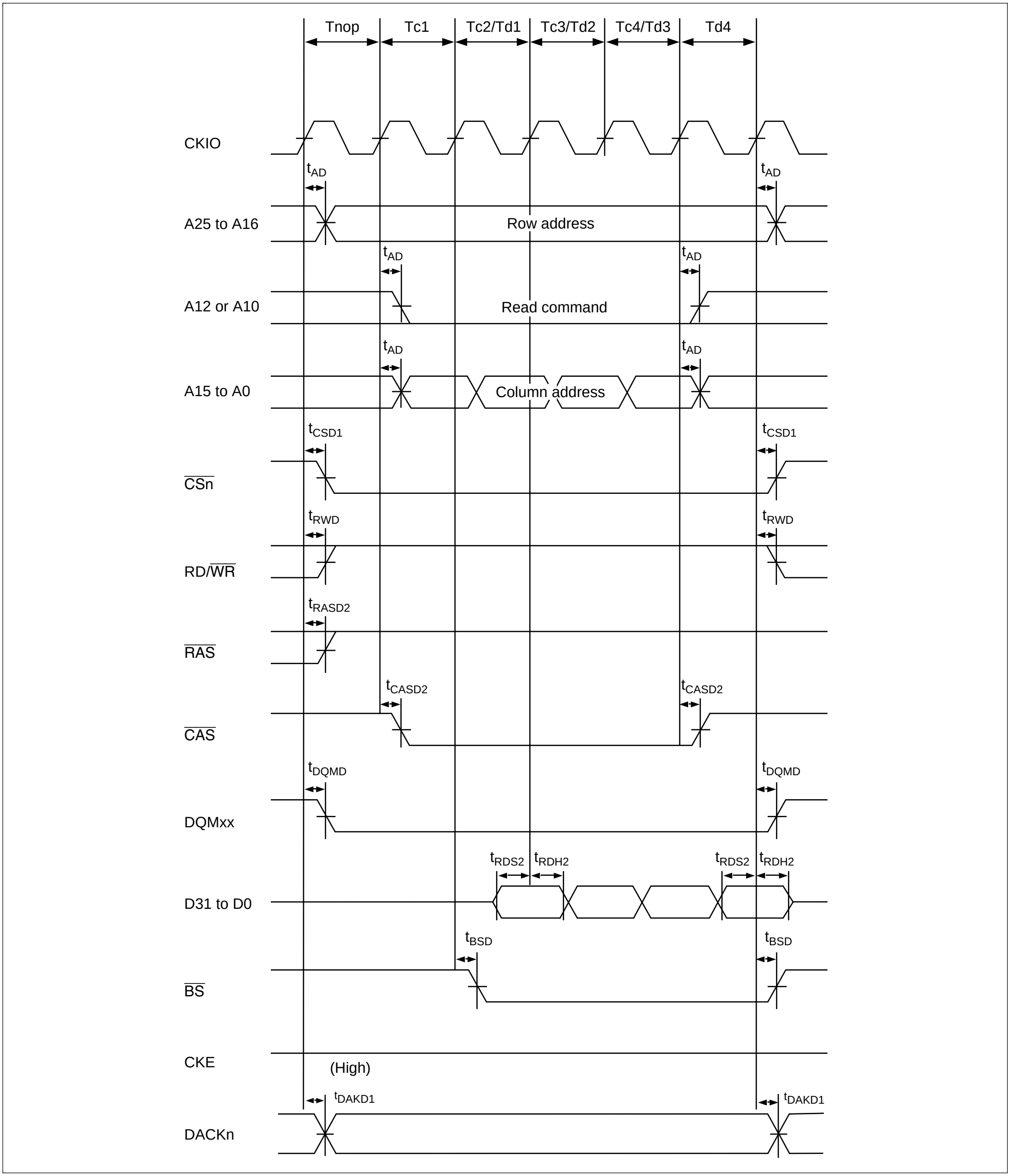


Figure 25.26 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Same Row Address, CAS Latency = 1)

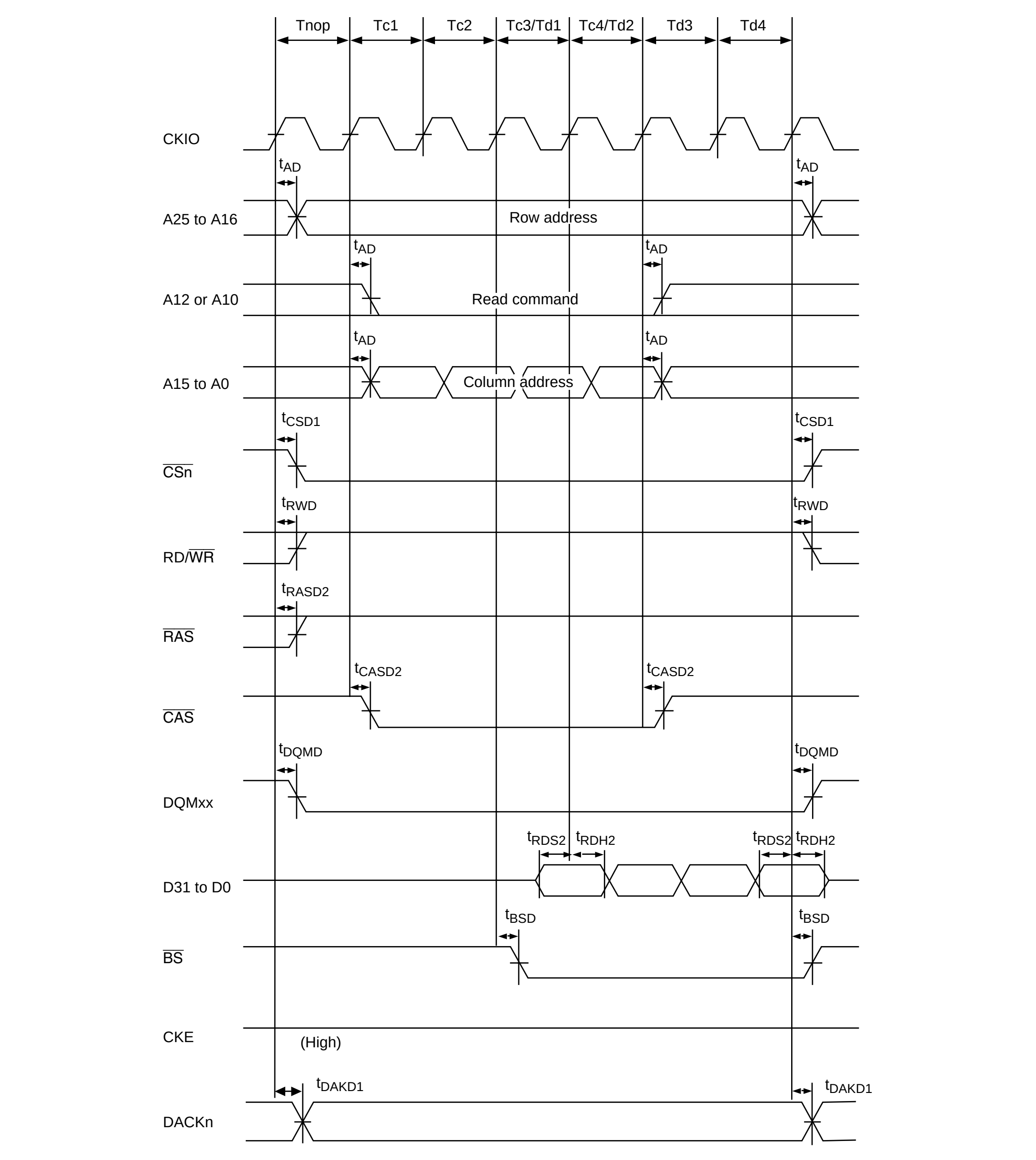


Figure 25.27 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Same Row Address, CAS Latency = 2)

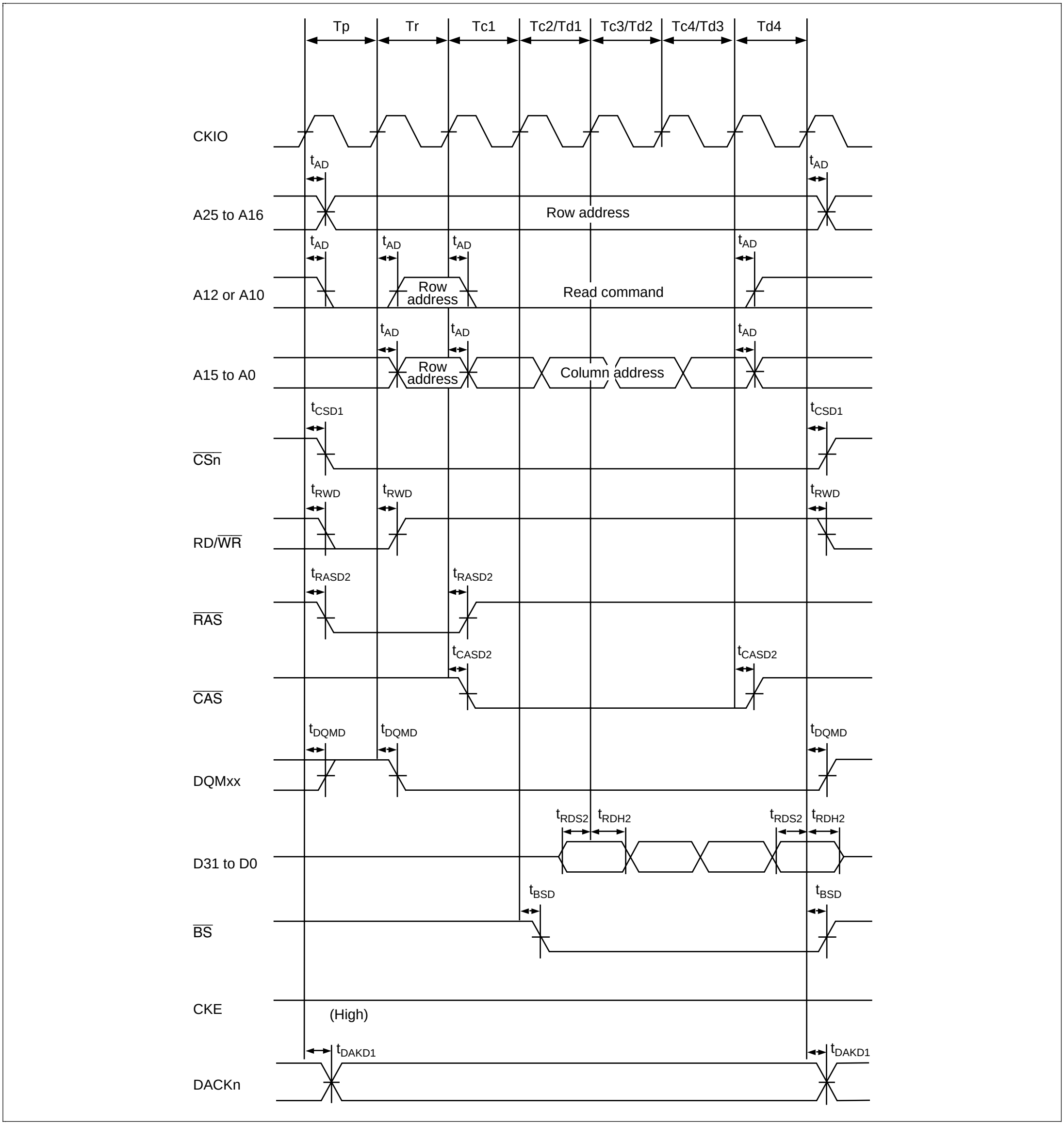


Figure 25.28 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Different Row Address, TPC = 0, RCD = 0, CAS Latency = 1)

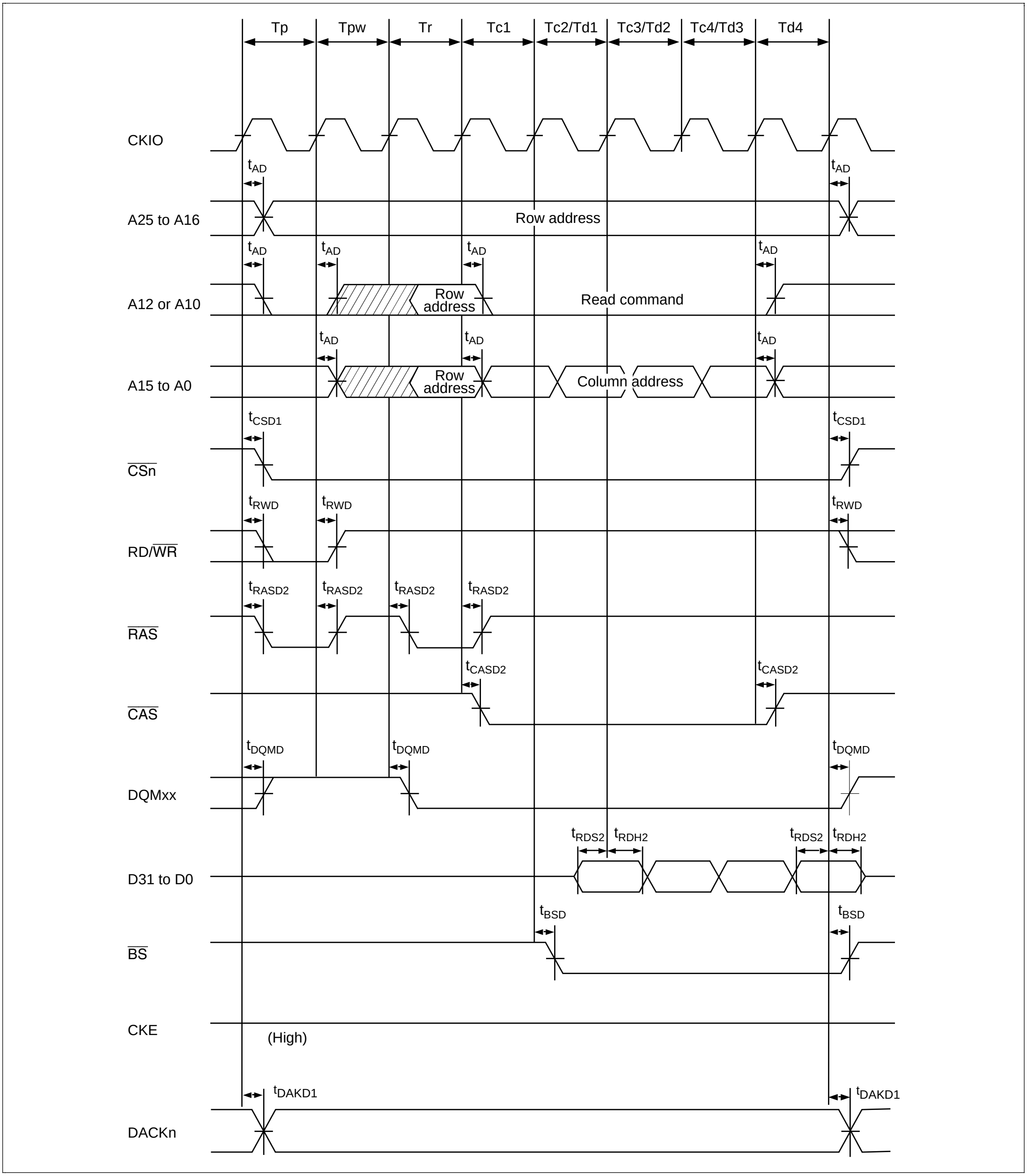


Figure 25.29 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Different Row Address, TPC = 1, RCD = 0, CAS Latency = 1)

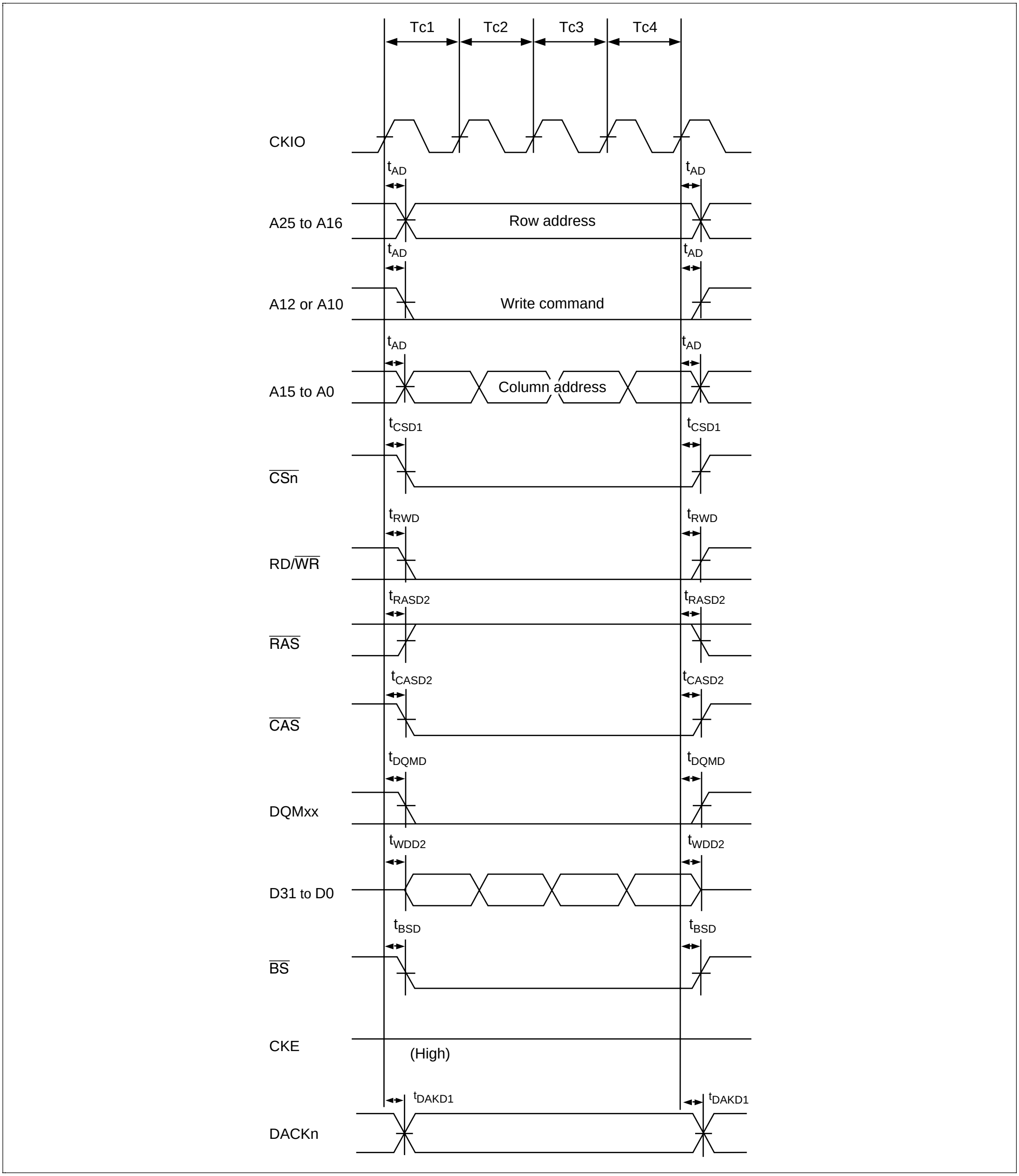


Figure 25.30 Synchronous DRAM Burst Write Bus Cycle (RAS Down, Same Row Address)

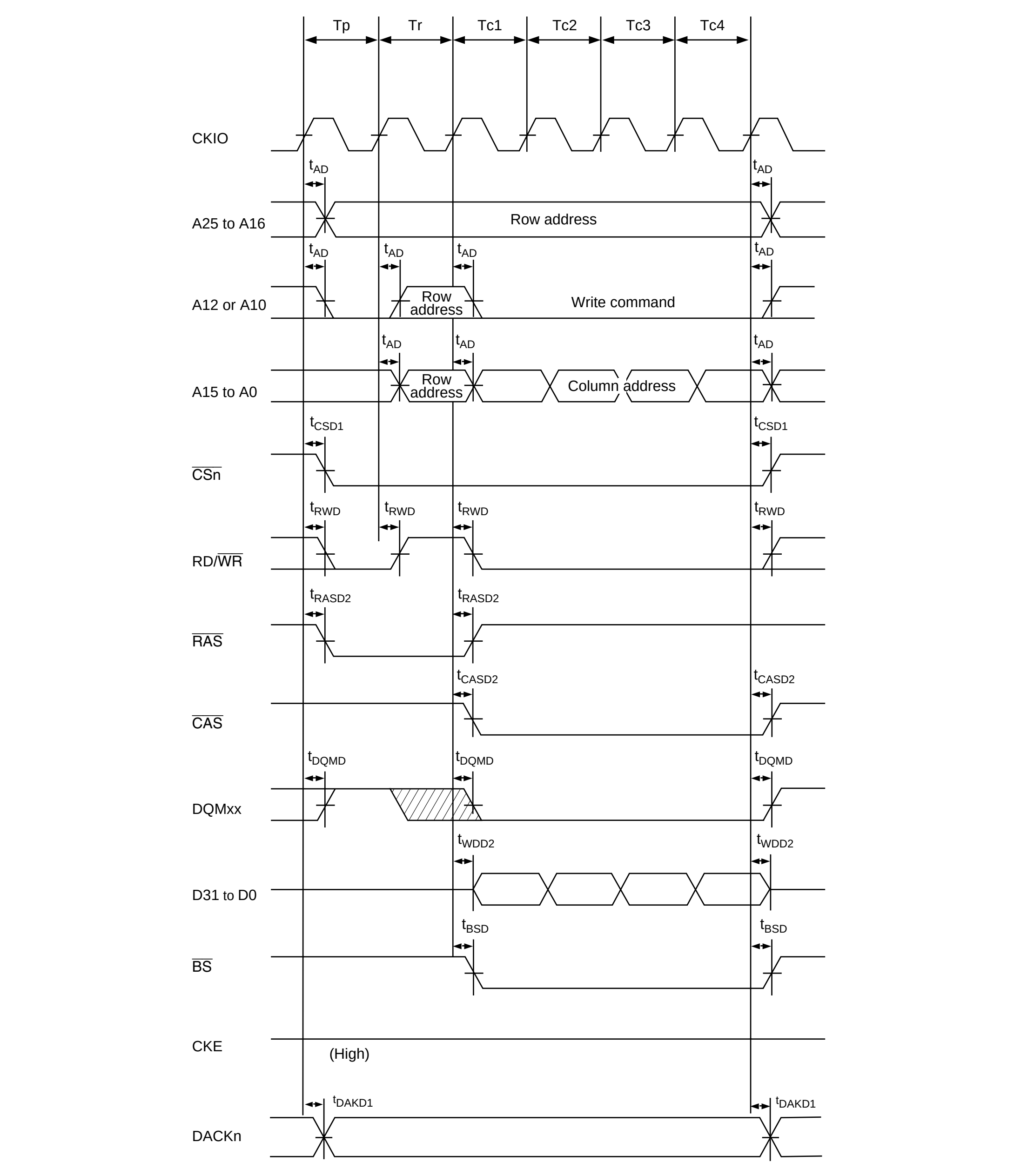


Figure 25.31 Synchronous DRAM Burst Write Bus Cycle (RAS Down, Different Row Address, TPC = 0, RCD = 0)

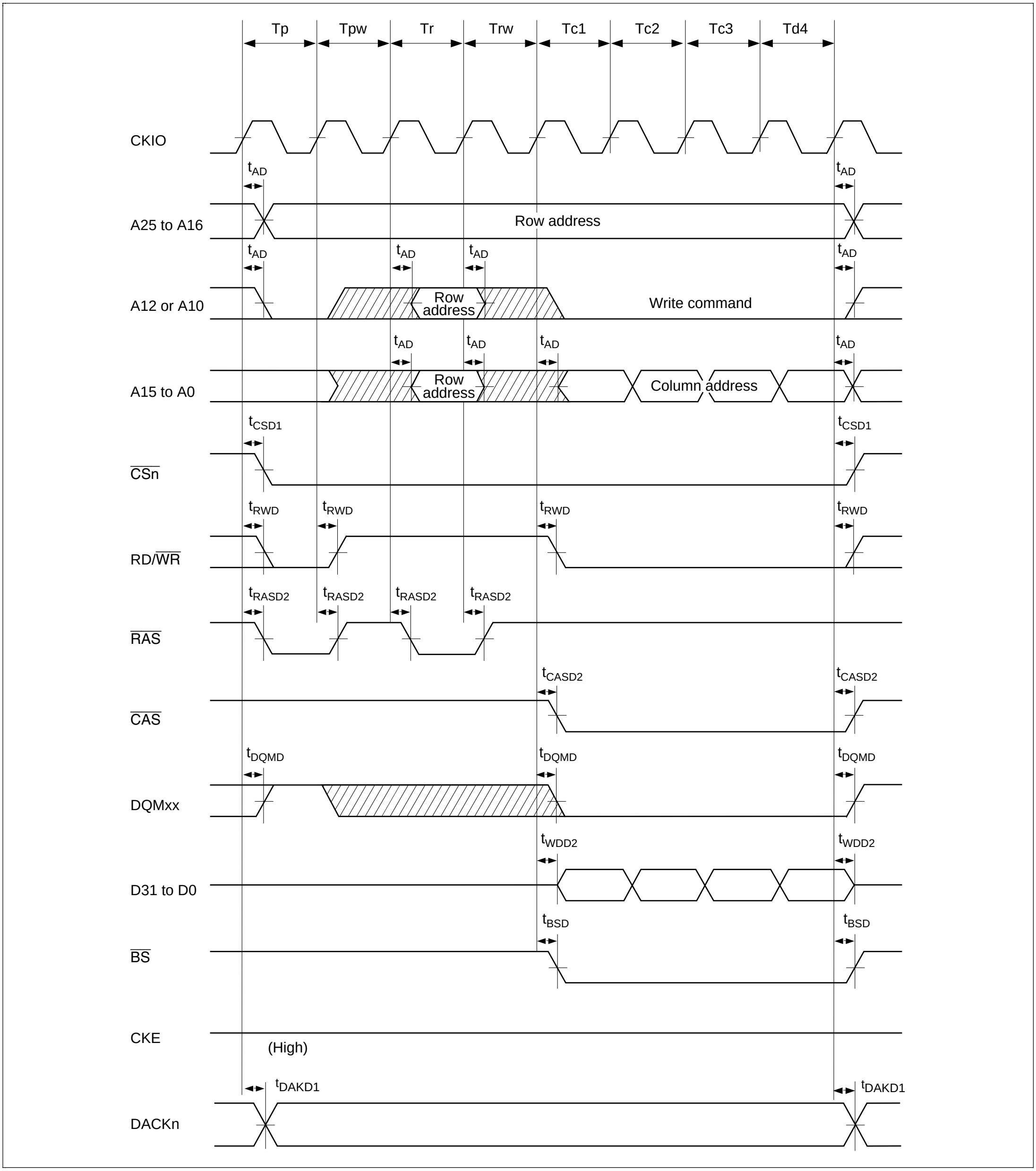


Figure 25.32 Synchronous DRAM Burst Write Bus Cycle (RAS Down, Different Row Address, TPC = 1, RCD = 1)

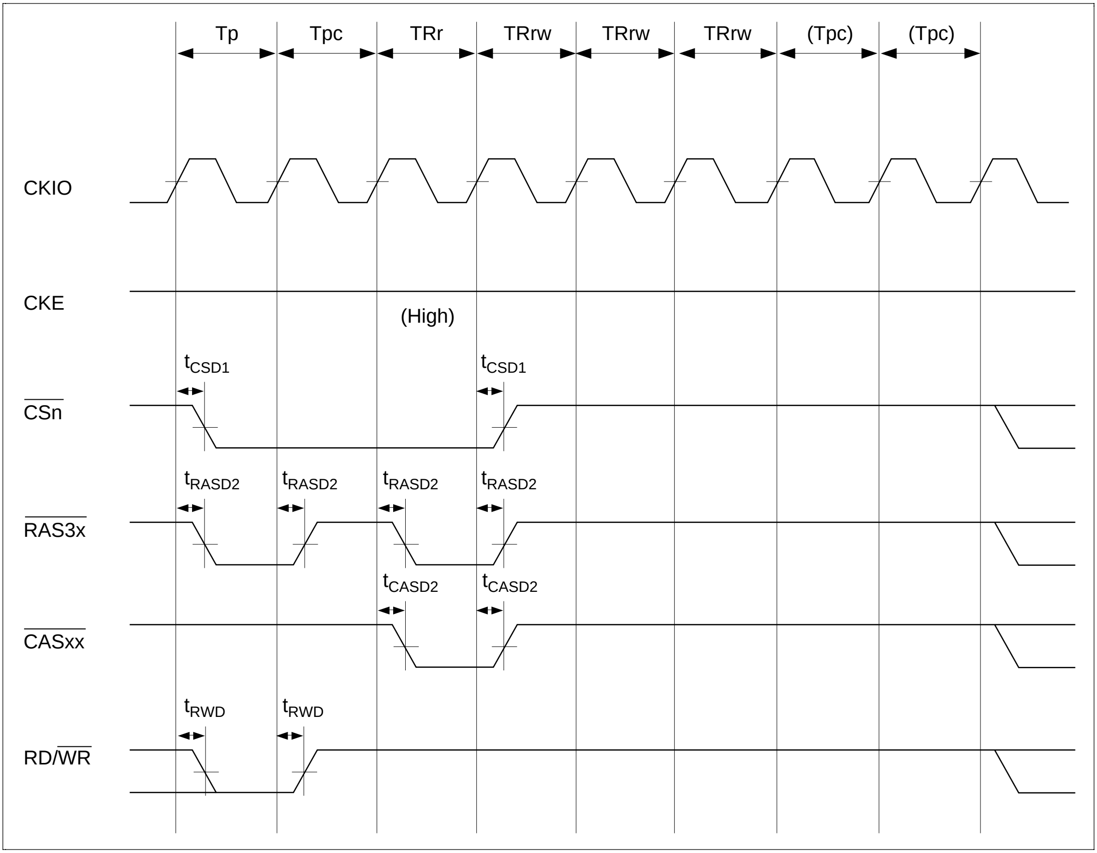


Figure 25.33 Synchronous DRAM Auto-Refresh Timing (TRAS = 1, TPC = 1)

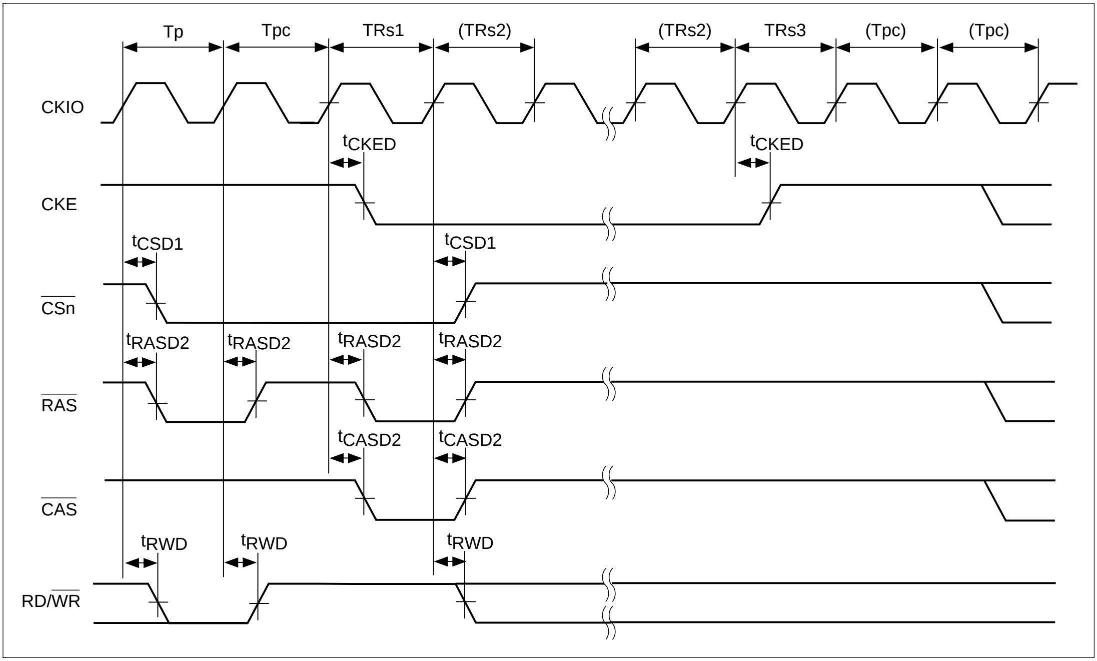


Figure 25.34 Synchronous DRAM Self-Refresh Cycle (TPC = 0, TPC = 1)

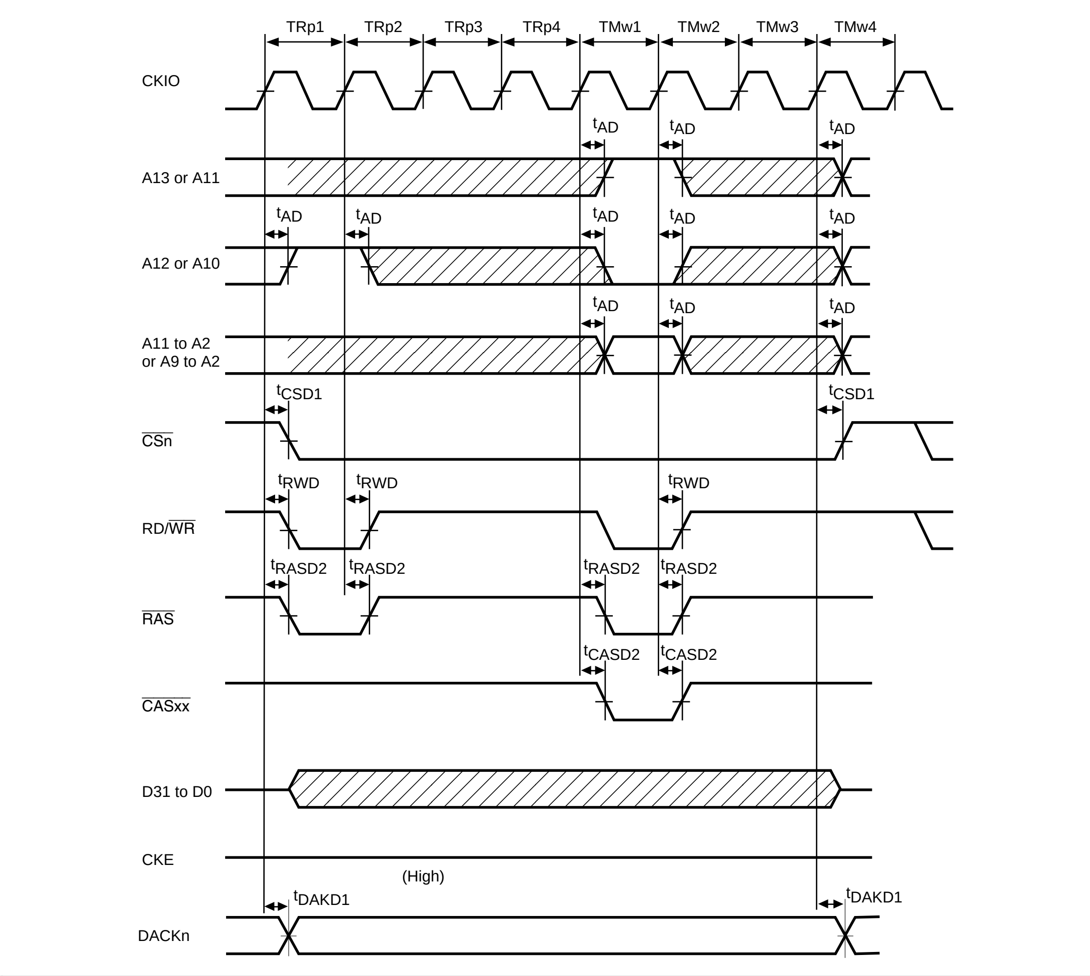


Figure 25.35 Synchronous DRAM Mode Register Write Cycle

25.3.7 Peripheral Module Signal Timing

Table 25.8 Peripheral Module Signal Timing
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Module	Item		Symbol	Min	Max	Unit	Figure
TMU	Timer input setup time		t_{TCLKS}	15	—	ns	25.36
	Timer clock input setup time		t_{TCKS}	15	—		25.37
	Timer clock pulse width	Edge specification	t_{TCKWH}	1.5	—	t_{cyc}	
		Both edge specification	t_{TCKWL}	2.5	—		
SCIF0	Input clock cycle	Clock synchronization	t_{Scyc}	12	—	P_{cyc}	25.38 25.39
	Input clock rise time		t_{SCKr}	—	1.5		25.38
	Input clock fall time		t_{SCKf}	—	1.5		
	Input clock pulse width		t_{SCKW}	0.4	0.6	tscyc	
	Transmission data delay time		t_{TXD}	—	3pcyc* + 50	ns	25.39
	Receive data setup time (clock synchronization)		t_{RXS}	2pcyc*	—		
	Receive data hold time (clock synchronization)		t_{RXH}	2pcyc*	—		
SCIF1	Input clock cycle	Clock synchronization	t_{Scyc}	12	—	t_{cyc}	25.38 25.39
	Input clock rise time		t_{SCKr}	—	1.5		25.38
	Input clock fall time		t_{SCKf}	—	1.5		
	Input clock pulse width		t_{SCKW}	0.4	0.6	tscyc	
	Transmission data delay time		t_{TXD}	—	3pcyc* + 50	ns	25.39
	Receive data setup time (clock synchronization)		t_{RXS}	2pcyc*	—		
	Receive data hold time (clock synchronization)		t_{RXH}	2pcyc*	—		

Note: * pcyc indicates a P clock cycle.

Table 25.8 Peripheral Module Signal Timing (cont) ($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Module	Item		Symbol	Min	Max	Unit	Figure
SCIF2	Input clock cycle	Asynchronization	t_{Scyc}	4	—	t_{cyc}	25.38 25.39
		Clock synchronization		12	—		
	Input clock rise time		t_{SCKr}	—	1.5		25.38
	Input clock fall time		t_{SCKf}	—	1.5		
	Input clock pulse width		t_{SCKW}	0.4	0.6	tscyc	
	Transmission data delay time		t_{TXD}	—	3pcyc* + 50	ns	25.39
	Receive data setup time (clock synchronization)		t_{RXS}	2pcyc*	—		
	Receive data hold time (clock synchronization)		t_{RXH}	2pcyc*	—		
Port	Output data delay time		t_{PORTD}	—	17	ns	25.40
	Input data setup time		t_{PORTS1}	15	—		
	Input data hold time		t_{PORTH1}	8	—		
	Input data setup time		t_{PORTS2}	17	—		
	Input data hold time		t_{PORTH2}	10	—		
DMAC	$\overline{\text{DREQ}}$ setup time		t_{DREQ}	12	—	ns	25.41
	$\overline{\text{DREQ}}$ hold time		t_{DREQH}	8	—		
	DRAK delay time		t_{DRAKD}	—	14		25.42

Note: * pcyc indicates a P clock cycle.

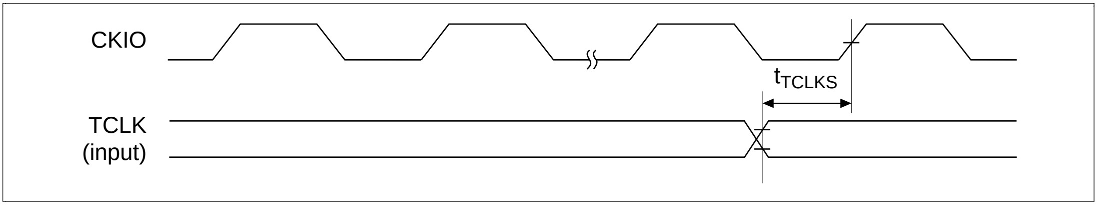


Figure 25.36 TCLK Input Timing

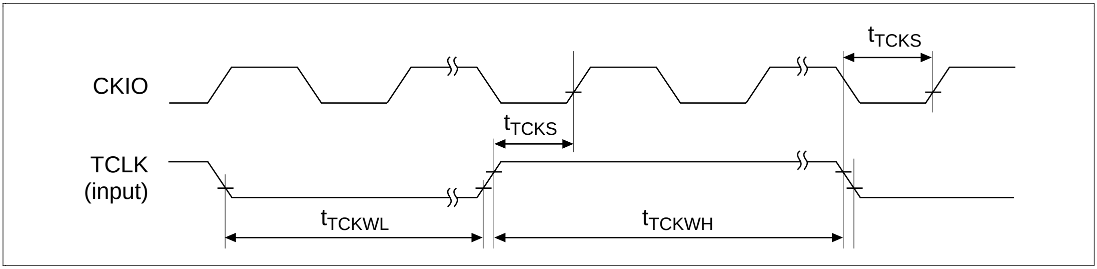


Figure 25.37 TCLK Clock Input Timing

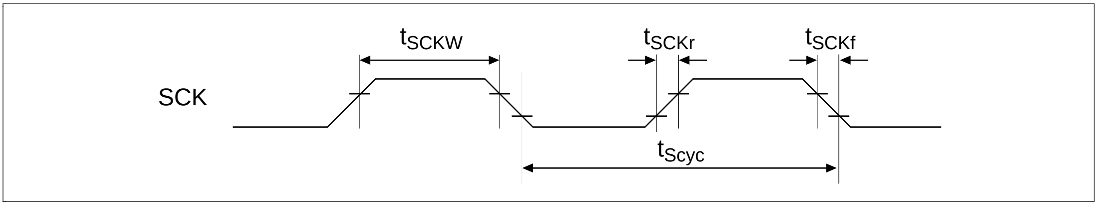


Figure 25.38 SCK Clock Input Timing

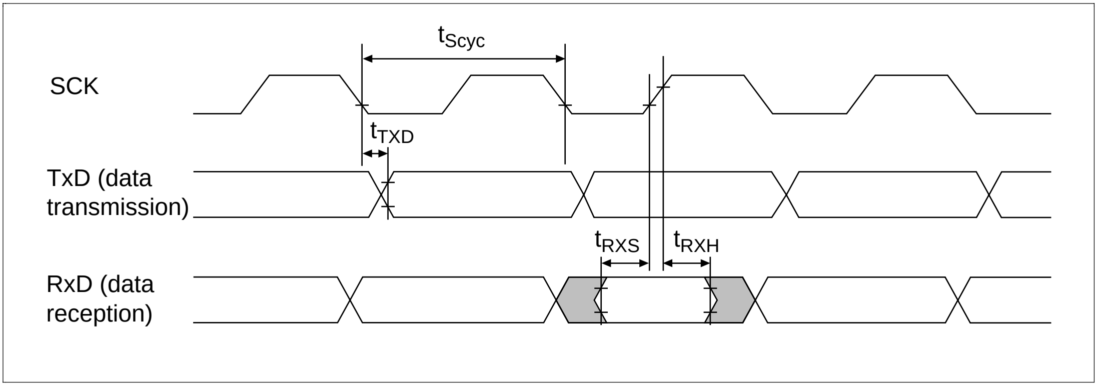


Figure 25.39 SCIF I/O Timing in Clock Synchronous Mode

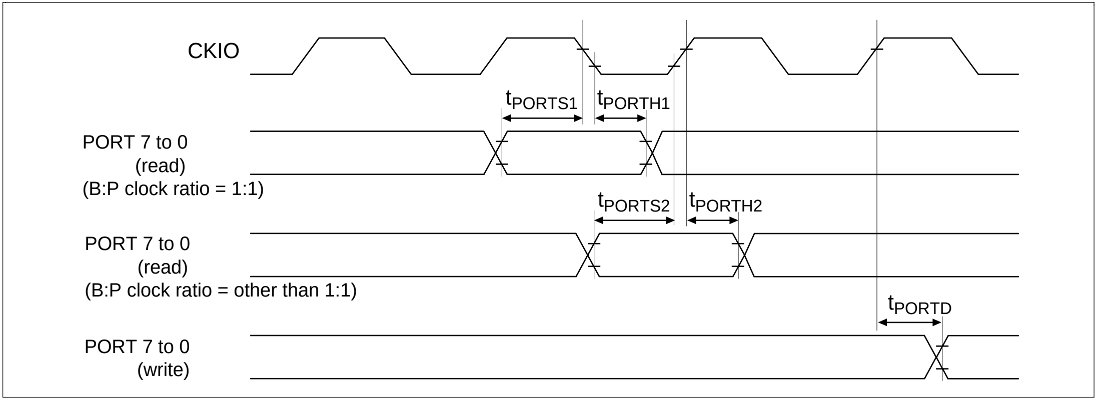


Figure 25.40 I/O Port Timing

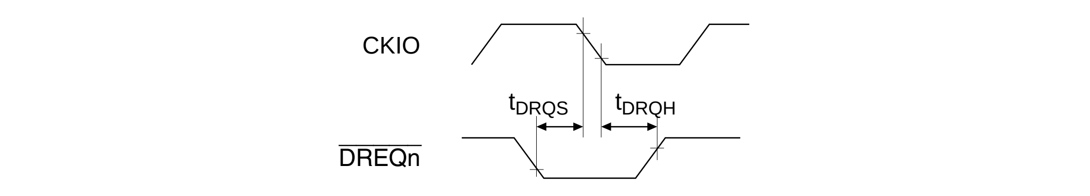


Figure 25.41 $\overline{\text{DREQ}}$ Input Timing

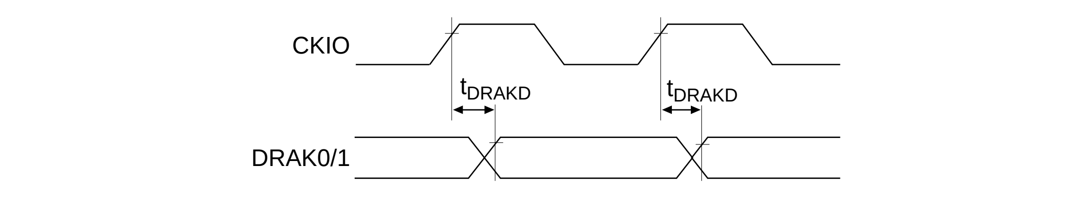


Figure 25.42 DRAK Output Timing

25.3.8 USB Module Signal Timing

Table 25.9 USB Module Signal Timing
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Item	Symbol	Min	Max	Unit	Figure
Frequency (48 MHz)	t_{FREQ}	47.9	48.1	MHz	25.43
Clock rise time	t_{R48}	—	2	ns	
Clock fall time	t_{F48}	—	2	ns	
Duty (t_{HIGH}/t_{LOW})	t_{DUTY}	90	110	%	

Note: When the USB is operated by supplying a clock to the UCLK pin from off-chip, and EXCPG is not used, the supplied clock must satisfy the above clock specifications.

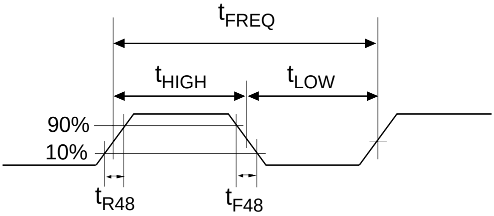


Figure 25.43 USB Clock Timing

Table 25.10 USB Module Pin Input/Output Timing
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Item	Symbol	Min	Max	Unit	Figure
Input data rise time	t_{RI}	—	4	ns	25.44
Input data fall time	t_{FI}	—	4	ns	
Rise/fall time matching (t_{RI}/t_{FI})	t_{RFMI}	90	110	%	
Output data rise time	t_{RO}	4	10	ns	
Output data fall time	t_{FO}	4	10	ns	
Rise/fall time matching (t_{RO}/t_{FO})	t_{RFMO}	90	110	%	

Note: The USB module in the SH7622 must be used together with a USB transceiver as a set. Transceivers that can be used are the Philips PDIUSBP11 Series or compatible models. See section 19, USB Function Module, for details of pin connections.

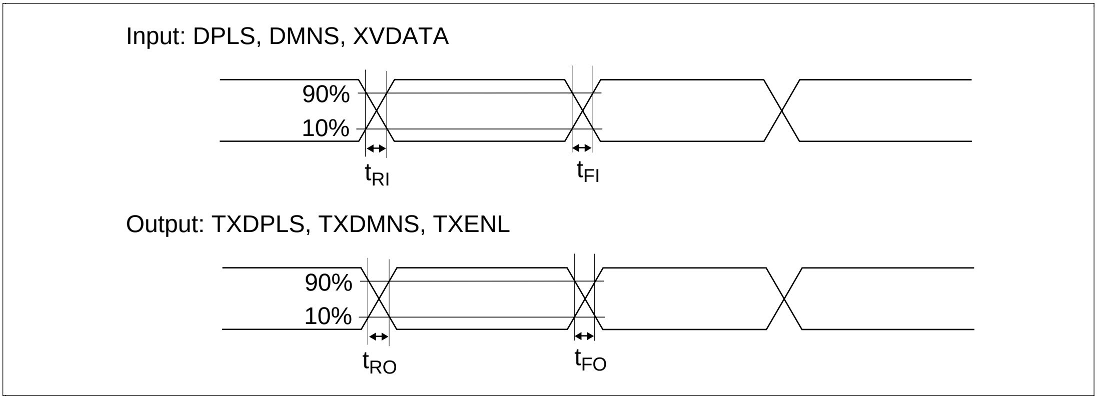


Figure 25.44 USB Module Pin Input/Output Timing

25.3.9 H-UDI-Related Pin Timing

Table 25.11 H-UDI-Related Pin Timing (VccQ = 3.3 ± 0.3 V, Vcc = 1.9/1.8 ± 0.15 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

Item	Symbol	Min	Max	Unit	Figure
TCK cycle time	t_{TCKcyc}	50	—	ns	Figure 25.45
TCK high pulse width	t_{TCKH}	12	—	ns	
TCK low pulse width	t_{TCKL}	12	—	ns	
TCK rise/fall time	t_{TCKf}	—	4	ns	
TRST setup time	t_{TRSTS}	12	—	ns	Figure 25.46
TRST hold time	t_{TRSTH}	50	—	t_{cyc}	
TDI setup time	t_{TDIS}	10	—	ns	Figure 25.47
TDI hold time	t_{TDIH}	10	—	ns	
TMS setup time	t_{TMSS}	10	—	ns	
TMS hold time	t_{TMSH}	10	—	ns	
TDO delay time	t_{TDOD}	—	16	ns	
$\overline{\text{ASEMD0}}$ setup time	t_{ASEMDH}	12	—	ns	Figure 25.48
$\overline{\text{ASEMD0}}$ hold time	t_{ASEMDS}	12	—	ns	

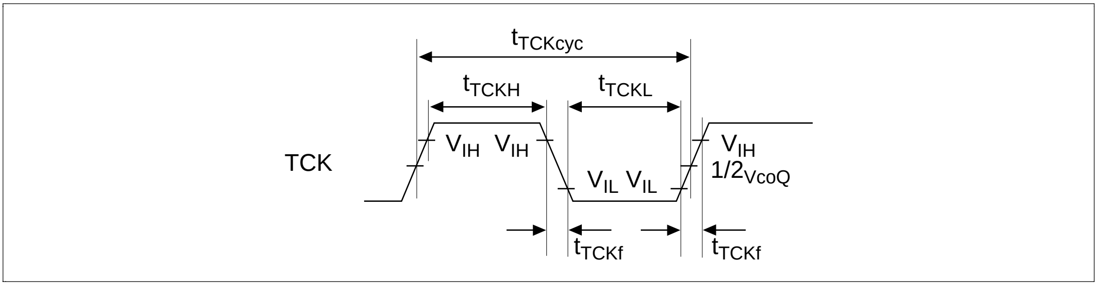


Figure 25.45 TCK Input Timing

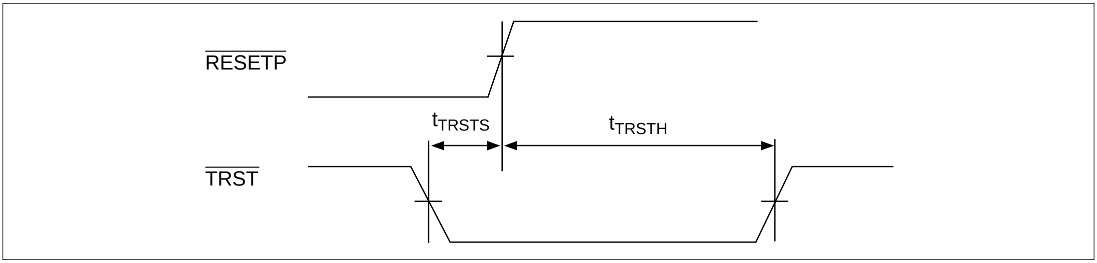


Figure 25.46 $\overline{\text{TRST}}$ Input Timing (Reset Hold)

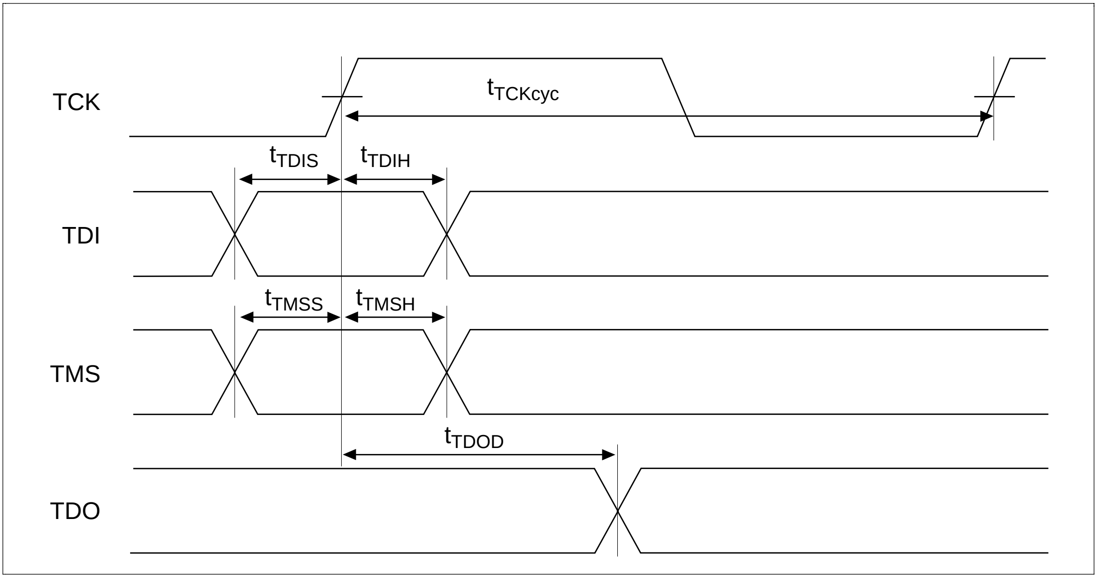


Figure 25.47 H-UDI Data Transfer Timing

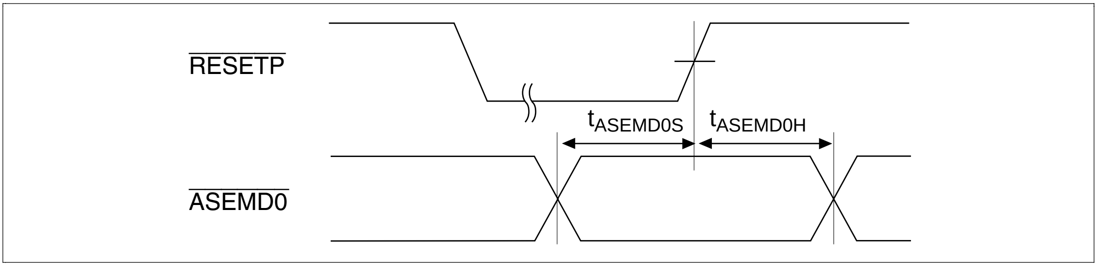


Figure 25.48 $\overline{\text{ASEMD0}}$ Input Timing

25.3.10 A/D Converter Timing

Table 25.12 A/D Converter Timing (VccQ = 3.3 ± 0.3 V, Vcc = 1.9/1.8 ± 0.15 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

Item		Symbol	Min	Typ	Max	Unit	Figure
External trigger input pulse width		t_{TRGW}	2	—	—	tcyc	25.49
External trigger input start delay time		t_{TRGS}	50	—	—	ns	
Input sampling time	(CKS = 0)	t_{SPL}	—	65	—	tcyc	25.50
	(CKS = 1)		—	32	—		
A/D conversion start delay time	(CKS = 0)	t_D	10	—	17	tcyc	
	(CKS = 1)		6	—	9		
A/D conversion time	(CKS = 0)	t_{CONV}	259	—	266	tcyc	
	(CKS = 1)		131	—	134		

tcyc: Pφ cycle

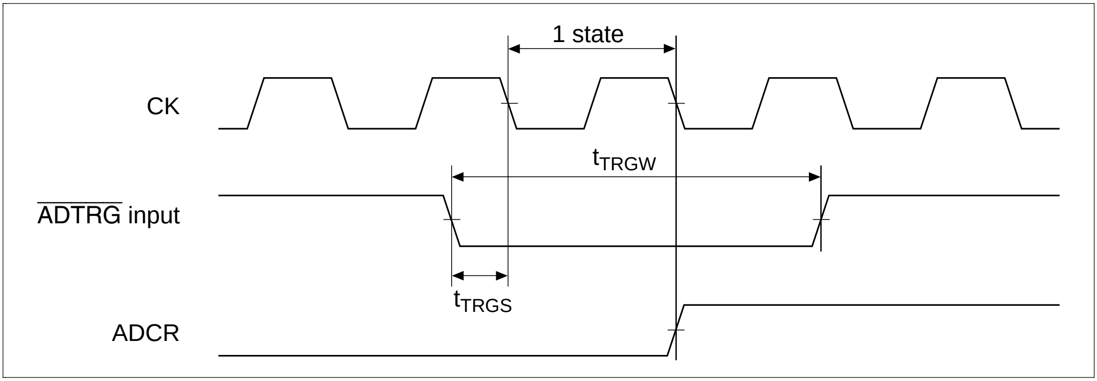
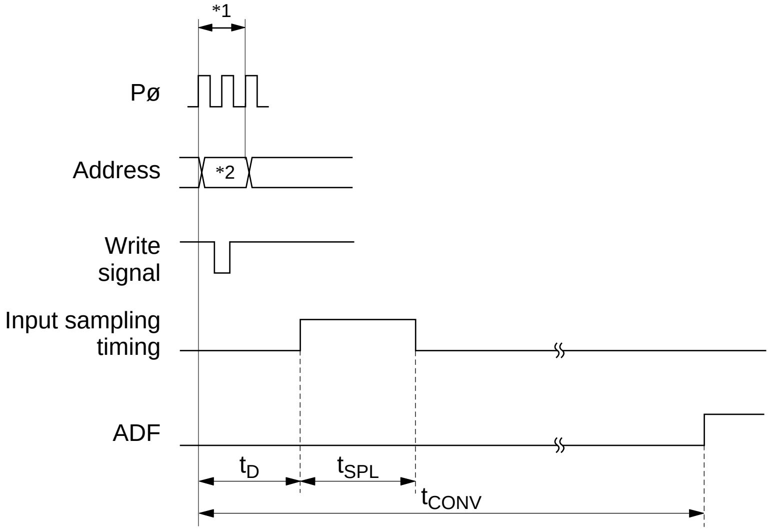


Figure 25.49 External Trigger Input Timing



t_D: A/D conversion start delay

t_{SPL}: Input sampling time

t_{CONV}: A/D conversion time

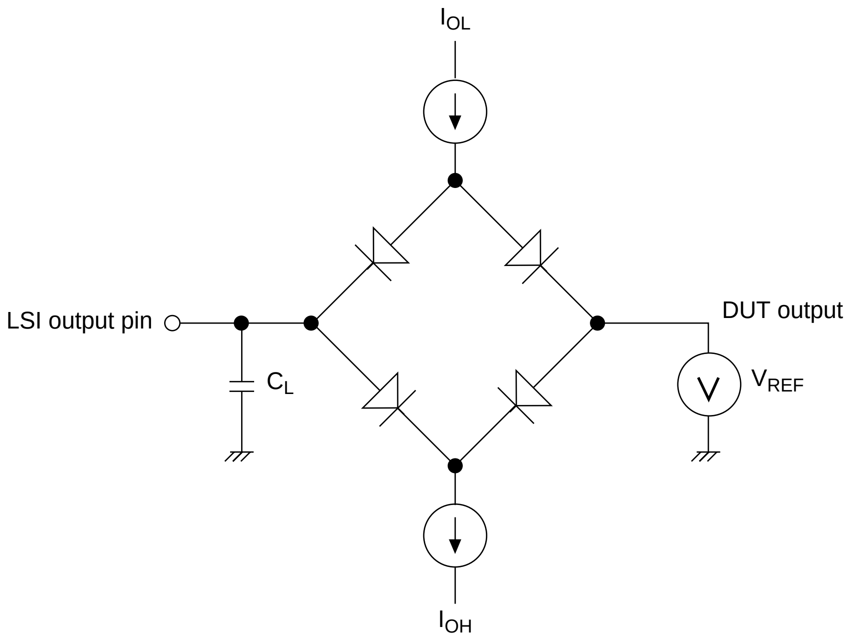
Notes: *1 ADCSR write cycle

*2 ADCSR address

Figure 25.50 A/D Conversion Timing

25.3.11 AC Characteristics Measurement Conditions

- I/O signal reference level: 1.5 V ($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V)
- Input pulse level: V_{SS} to 3.0 V (where $\overline{\text{RESETP}}$, $\overline{\text{RESETM}}$, NMI, $\overline{\text{IRQ5}}$–$\overline{\text{IRQ0}}$, CKIO, and MD4–MD0 are within V_{SS} to V_{CC})
- Input rise and fall times: 1 ns



Notes: 1. C_L is the total value that includes the capacitance of measurement instruments, etc., and is set as follows for each pin.
30 pF: CKIO, $\overline{\text{RAS}}$, $\overline{\text{CASxx}}$, $\overline{\text{CS0}}$, $\overline{\text{CS2}}$–$\overline{\text{CS6}}$, $\overline{\text{BACK}}$
50 pF: All other pins
2. I_{OL} and I_{OH} are the values shown in table 25.3.

Figure 25.51 Output Load Circuit

25.3.12 Delay Time Variation Due to Load Capacitance (Reference Values)

A graph (reference data) of the variation in delay time when a load capacitance greater than that stipulated (30 pF) is connected to the SH7622's pins is shown below. The graph shown in figure 25.52 should be taken into consideration in the design process if the stipulated capacitance is exceeded in connecting an external device.

If the connected load capacitance exceeds the range shown in figure 25.52, the graph will not be a straight line.

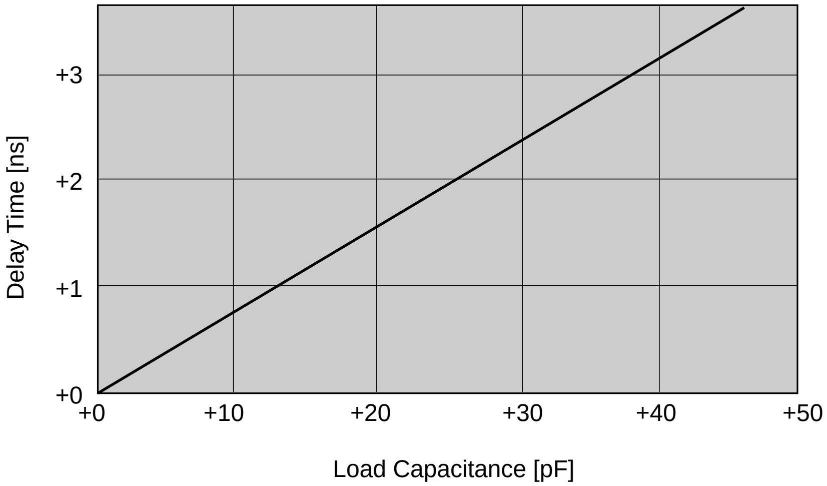


Figure 25.52 Load Capacitance vs. Delay Time

25.4 A/D Converter Characteristics

Table 25.13 lists the A/D converter characteristics.

Table 25.13 A/D Converter Characteristics
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Item	Min	Typ	Max	Unit
Resolution	10	10	10	bits
Conversion time	—	—	8.9	µs
Analog input capacitance	—	—	20*	pF
Permissible signal-source (single-source) impedance	—	—	5*	kΩ
Nonlinearity error	—	—	±3.0*	LSB
Offset error	—	—	±2.0*	LSB
Full-scale error	—	—	±2.0*	LSB
Quantization error	—	—	±0.5*	LSB
Absolute accuracy	—	—	±4.0	LSB

Note: * Reference values

Section 26 Electrical Characteristics (100 MHz)

26.1 Absolute Maximum Ratings

Table 26.1 shows the absolute maximum ratings.

Table 26.1 Absolute Maximum Ratings

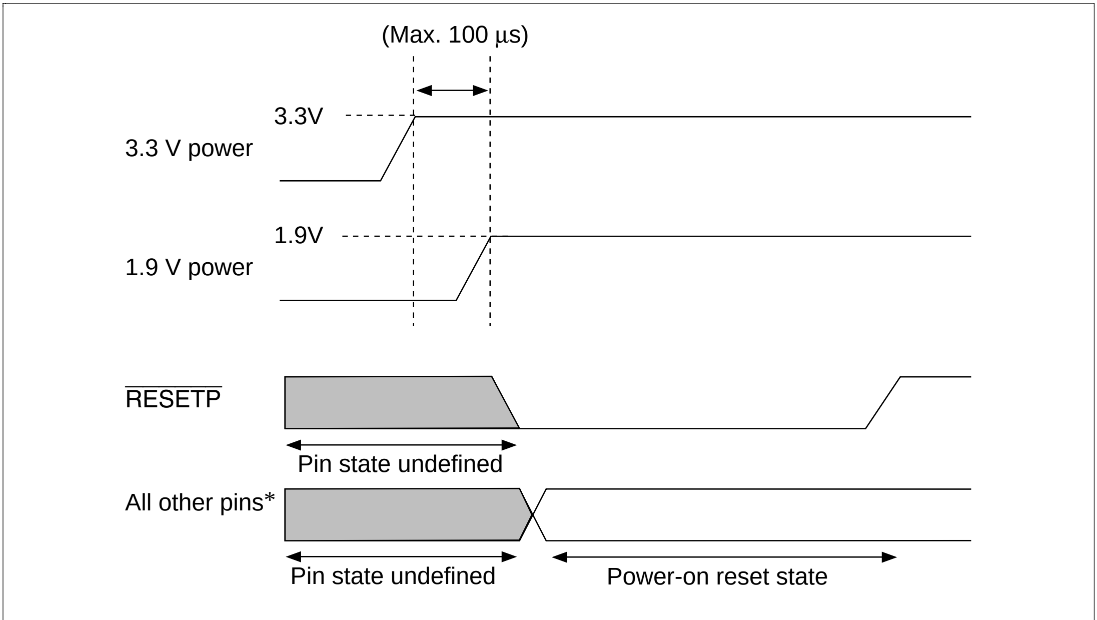
Item	Symbol	Rating	Unit
Power supply voltage (I/O)	$V_{CC}Q$	–0.3 to 4.2	V
Power supply voltage (internal)	V_{CC} V_{CC} – PLL1 V_{CC} – PLL2	–0.3 to 2.5	V
Input voltage (except port L)	Vin	–0.3 to $V_{CC}Q$ + 0.3	V
Input voltage (port L)	Vin	–0.3 to AV_{CC} + 0.3	V
Analog power-supply voltage	AV_{CC}	–0.3 to 4.6	V
Analog input voltage	V_{AN}	–0.3 to AV_{CC} + 0.3	V
Operating temperature	Topr	–20 to 75	°C
Storage temperature	Tstg	–55 to 125	°C

1. Exceeding the absolute maximum ratings may permanently damage the chip.
2. Order of turning on 1.9 V power (Vcc, Vcc-PLL1, Vcc-PLL2) and 3.3 V power (VccQ, AVcc):

   (1) First turn on the 3.3 V power, then turn on the 1.9 V power within 100 µs. This interval should be as short as possible.

   (2) Until voltage is applied to all power supplies and a low level is input at the $\overline{\text{RESETP}}$ pin, internal circuits remain unsettled, and so pin states are also undefined. The system design must ensure that these undefined states do not cause erroneous system operation.

Waveforms at power-on are shown in the following figure.



Note: * Except power/GND, clock related, and analog pins

Power-On Sequence

3. Power-off order

   (1) Reversing the order of powering-on, first turn off the 1.9 V power, then turn off the 3.3 V power within 100 µs. This interval should be as short as possible.

   (2) Pin states are undefined while only the 1.9 V power is off. The system design must ensure that these undefined states do not cause erroneous system operation.

26.2 DC Characteristics

Tables 26.2 and 26.3 list DC characteristics.

Table 26.2 DC Characteristics (1) [Common Items] (T_a = –20 to 75°C)

Item		Symbol	Min	Typ	Max	Unit	Measurement Conditions
Power supply voltage		$V_{CC}Q$	3.0	3.3	3.6	V	
		V_{CC}, V_{CC} – PLL1 V_{CC} – PLL2	1.75	1.9	2.05	V	
Current dissipation	Normal operation	I_{CC}	—	150	250	mA	V_{CC} = 1.9 V Iφ = 100 MHz
		$I_{CC}Q$	—	30	60	mA	$V_{CC}Q$ = 3.3 V Bφ = 33 MHz
	In standby mode	I_{stby}	—	100	300	µA	T_a = 25°C $V_{CC}Q$ = 3.3 V V_{CC} = 1.9 V
Input leak current	All input pins	$\| I_{in} \|$	—	—	1.0	µA	V_{in} = 0.5 to $V_{CC}Q$ – 0.5 V
Three-state leak current	I/O, all output pins (off condition)	$\| I_{TSI} \|$	—	—	1.0	µA	Vin = 0.5 to $V_{CC}Q$ – 0.5 V
Pull-up resistance	Port pin	P_{pull}	30	60	120	kΩ	
Pin capacity	All pins	C	—	—	10	pF	
Analog power-supply voltage		AV_{CC}	3.0	3.3	3.6	V	
Analog power-supply current	During A/D conversion	AI_{CC}	—	0.8	2	mA	
	Idle		—	0.01	5.0	µA	

Table 26.2 DC Characteristics (2-a) [Excluding USB-Related Pins] (T_a = –20 to 75°C)

Item		Symbol	Min	Typ	Max	Unit	Measurement Conditions
Input high voltage	$\overline{\text{RESETP}}$, $\overline{\text{RESETM}}$, NMI	V_{IH}	$V_{CC}Q \times 0.9$	—	$V_{CC}Q + 0.3$	V	
	$\overline{\text{BREQ}}$, IRQ7 to IRQ0, MD0 to MD4		$V_{CC}Q - 0.4$	—	$V_{CC}Q + 0.3$	V	
	EXTAL, CKIO		$V_{CC}Q - 0.4$	—	$V_{CC}Q + 0.3$	V	
	Port L		2.0	—	$AV_{CC} + 0.3$	V	
	Other input pins		2.0	—	$V_{CC}Q + 0.3$	V	
Input low voltage	$\overline{\text{RESETP}}$, $\overline{\text{RESETM}}$, NMI	V_{IL}	–0.3	—	$V_{CC}Q \times 0.1$	V	
	$\overline{\text{BREQ}}$, IRQ7 to IRQ0, MD0 to MD4		–0.3	—	$V_{CC}Q \times 0.2$	V	
	Port L		–0.3	—	$AV_{CC} \times 0.2$	V	
	Other input pins		–0.3	—	$V_{CC}Q \times 0.2$	V	
Output high voltage	All output pins	V_{OH}	2.4	—	—	V	$V_{CC}Q$ = 3.0 V I_{OH} = –200 µA
			2.0	—	—	V	$V_{CC}Q$ = 3.0 V I_{OH} = –2 mA
Output low voltage	All output pins	V_{OL}	—	—	0.55	V	$V_{CC}Q$ = 3.6 V I_{OL} = 1.6 mA

Notes: 1. The V_{CC} pins must be connected to V_{CC}, and the V_{SS} pins to V_{SS}.

2. AV_{CC} must satisfy the condition: $V_{CC}Q - 0.2 \text{ V} \leq AV_{CC} \leq V_{CC}Q + 0.2 \text{ V}$. Do not leave the AV_{CC} and AV_{SS} pins open if the A/D converter is not used; connect AV_{CC} to $V_{CC}Q$, and AV_{SS} to $V_{SS}Q$.

3. Current dissipation values are for V_{IH}min = $V_{CC}Q$ – 0.5 V and V_{IL}max = 0.5 V with all output pins unloaded.

Table 26.2 DC Characteristics (2-b) [USB-Related Pins*] (T_a = –20 to 75°C)

Item	Symbol	Min	Typ	Max	Unit	Measurement Conditions
Power supply voltage	$V_{CC}Q$	3.0	3.3	3.6	V	
Input high voltage	V_{IH}	2.0	—	$V_{CC}Q$ + 0.3	V	
Input low voltage	V_{IL}	–0.3	—	$V_{CC}Q \times 0.2$	V	
Output high voltage	V_{OH}	2.4	—	—	V	$V_{CC}Q$ = 3.0 V I_{OH} = –200 µA
		2.0	—	—		$V_{CC}Q$ = 3.0 V I_{OH} = –2 mA
Output low voltage	V_{OL}	—	—	0.55	V	$V_{CC}Q$ = 3.6 V I_{OL} = 1.6 mA

Note: * Pins XVDATA, DPLS, DMNS, TXDPLS, TXDMNS, TXENL, VBUS, SUSPND, and UCLK

Table 26.3 Permitted Output Current Values
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Item	Symbol	Min	Typ	Max	Unit
Output low-level permissible current (per pin)	I_{OL}	—	—	2.0	mA
Output low-level permissible current (total)	ΣI_{OL}	—	—	120	mA
Output high-level permissible current (per pin)	$-I_{OH}$	—	—	2.0	mA
Output high-level permissible current (total)	$\Sigma (-I_{OH})$	—	—	40	mA

Caution: To ensure LSI reliability, do not exceed the value for output current given in table 26.3.

26.3 AC Characteristics

In general, inputting for this LSI should be clock synchronous. Keep the setup and hold times for each input signal unless otherwise specified.

Table 26.4 Maximum Operating Frequencies
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

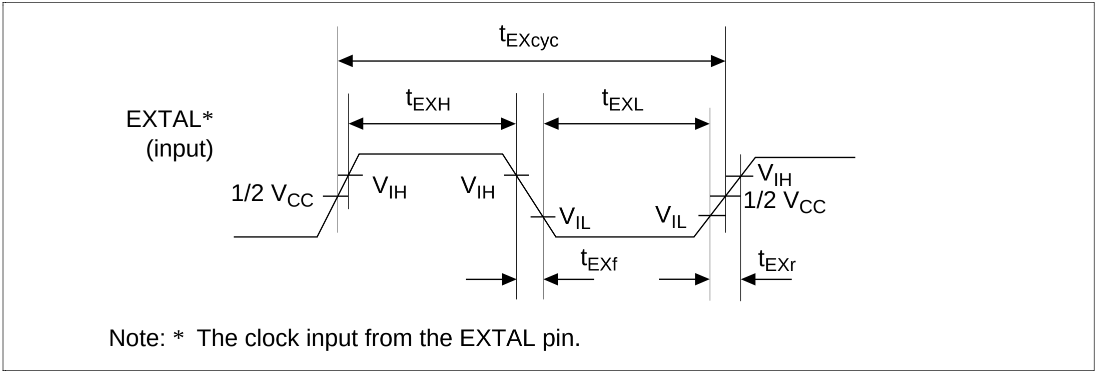
Item		Symbol	Min	Typ	Max	Unit	Remarks
Operating frequency	CPU, cache, (Iϕ)	f	20	—	100	MHz	
	External bus (Bϕ)		5	—	33		
	Peripheral module (Pϕ)		5	—	33		

26.3.1 Clock Timing

Table 26.5 Clock Timing
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Item	Symbol	Min	Max	Unit	Figure
EXTAL clock input frequency	f_{EX}	5	33	MHz	26.1
EXTAL clock input cycle time	t_{EXcyc}	30.3	200	ns	
EXTAL clock input low pulse width	t_{EXL}	7	—	ns	
EXTAL clock input high pulse width	t_{EXH}	7	—	ns	
EXTAL clock input rise time	t_{EXr}	—	4	ns	
EXTAL clock input fall time	t_{EXf}	—	4	ns	
CKIO clock input frequency	f_{CKI}	20	33	MHz	26.2 (1)
CKIO clock input cycle time	t_{CKIcyc}	30.3	50	ns	
CKIO clock input low pulse width	t_{CKIL}	7	—	ns	
CKIO clock input high pulse width	t_{CKIH}	7	—	ns	
CKIO clock input rise time	t_{CKIR}	—	3	ns	
CKIO clock input fall time	t_{CKIF}	—	3	ns	
CKIO clock input frequency	f_{CKI}	5	33	MHz	26.2 (2)
CKIO clock input cycle time	t_{CKIcyc}	30.3	200	ns	
CKIO clock input low pulse width	t_{CKIL}	8	—	ns	
CKIO clock input high pulse width	t_{CKIH}	8	—	ns	
CKIO clock input rise time	t_{CKIr}	—	6	ns	
CKIO clock input fall time	t_{CKIf}	—	6	ns	
Power-on oscillation settling time	t_{OSC1}	10	—	ms	26.3
$\overline{\text{RESETP}}$ setup time	t_{RESPS}	20	—	ns	26.3, 26.4
$\overline{\text{RESETM}}$ setup time	t_{RESMS}	0	—	ns	
$\overline{\text{RESETP}}$ assert time	t_{RESPW}	20	—	tcyc	
$\overline{\text{RESETM}}$ assert time	t_{RESMW}	20	—	tcyc	
Standby return oscillation settling time 1	t_{OSC2}	10	—	ms	26.4
Standby return oscillation settling time 2	t_{OSC3}	10	—	ms	26.5
PLL synchronization settling time	t_{PLL}	100	—	µs	26.6

Note: The maximum internal data bus operating frequency is 100 MHz. Set the multiplication factor in line with this condition.



Note: * The clock input from the EXTAL pin.

Figure 26.1 EXTAL Clock Input Timing

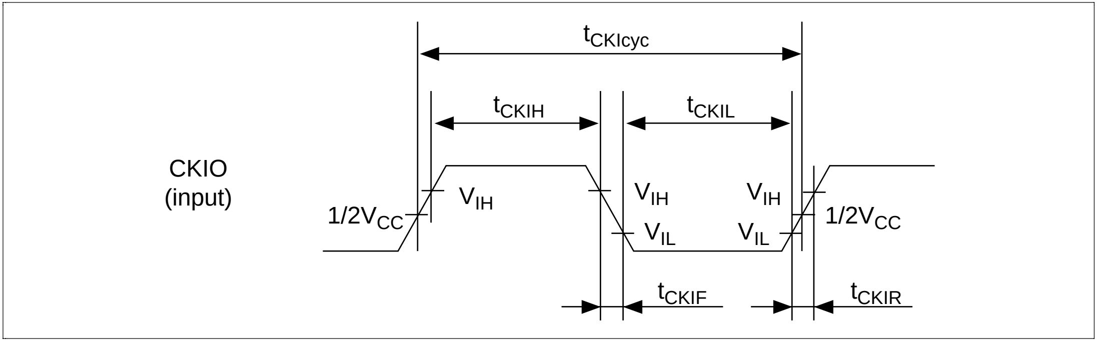


Figure 26.2 (1) CKIO Clock Input Timing

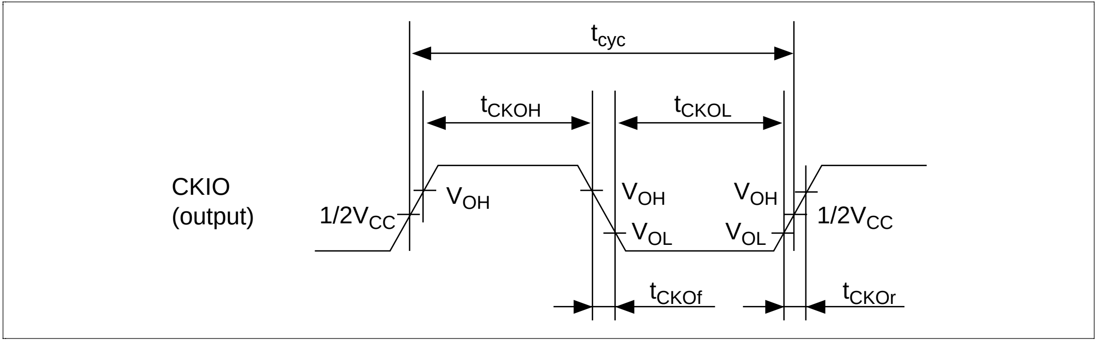
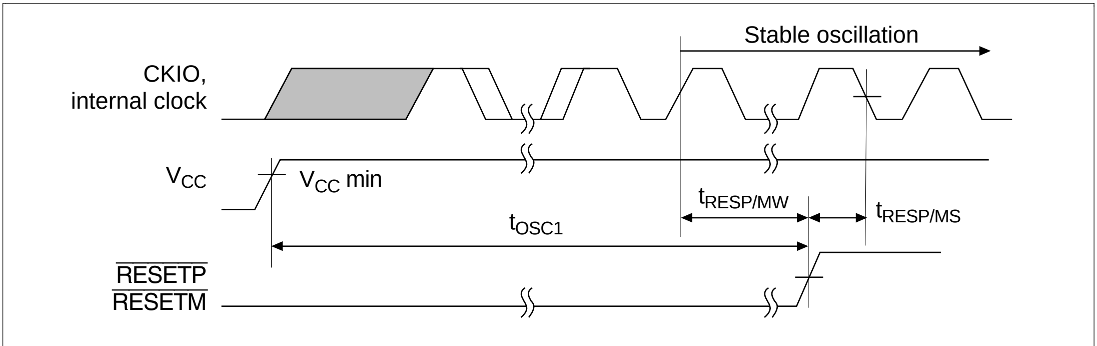
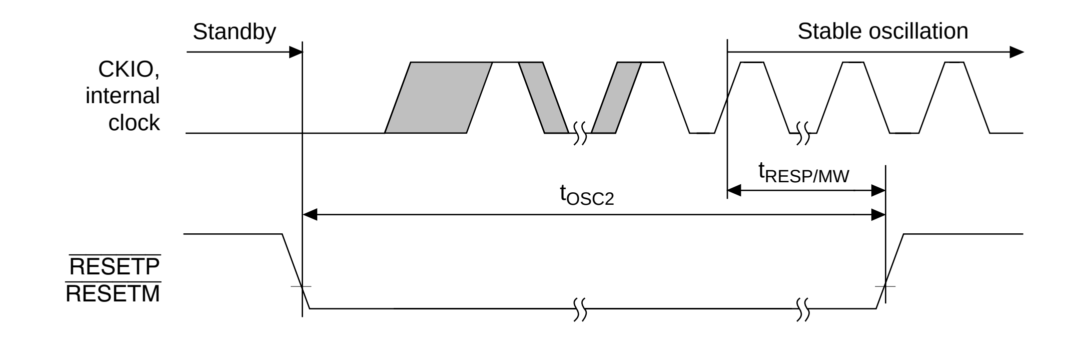


Figure 26.2 (2) CKIO Clock Output Timing



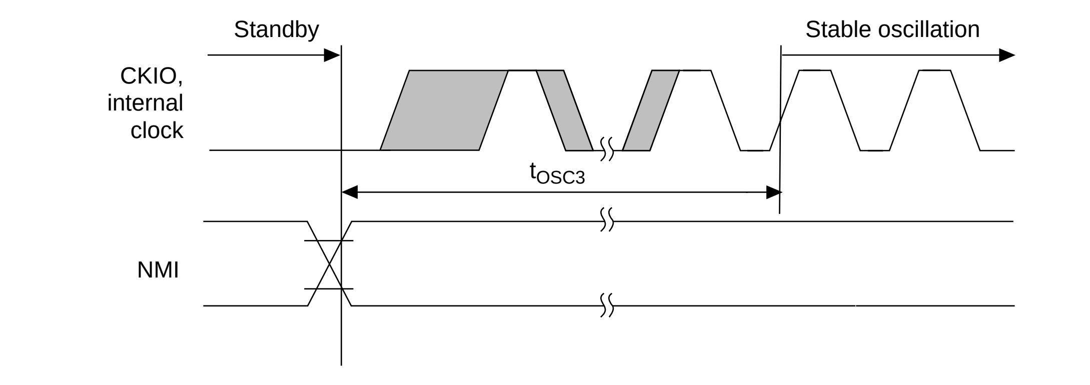
Note: Oscillation settling time when built-in oscillator is used

Figure 26.3 Power-on Oscillation Settling Time



Note: Oscillation settling time when built-in oscillator is used

Figure 26.4 Oscillation Settling Time at Standby Return (Return by Reset)



Note: Oscillation settling time when built-in oscillator is used

Figure 26.5 Oscillation Settling Time at Standby Return (Return by NMI)

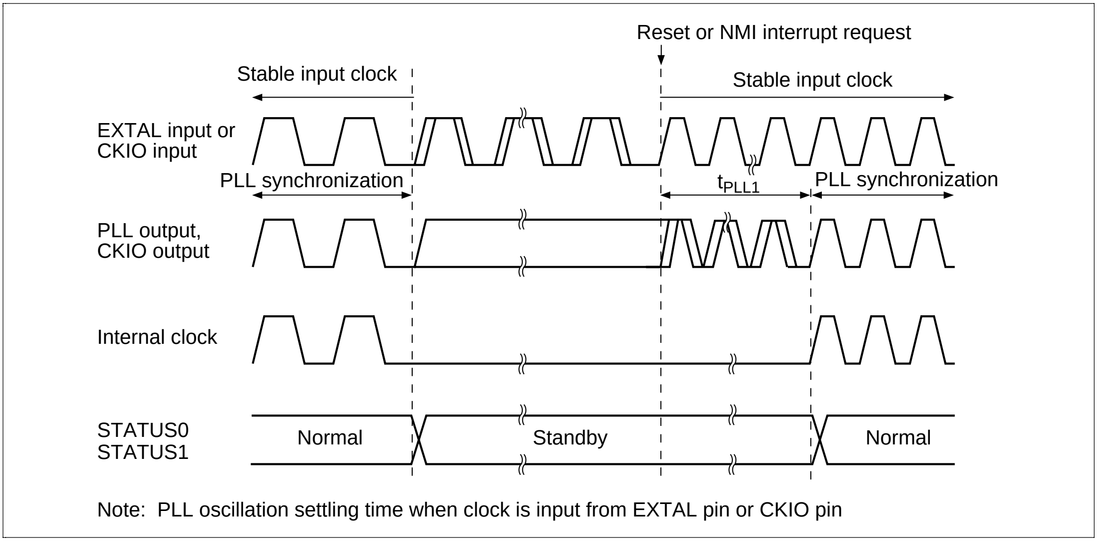


Figure 26.6   PLL Synchronization Settling Time by Reset or NMI

26.3.2 Control Signal Timing

Table 26.6 Control Signal Timing
(V_{CC} = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

		33*2			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Figure**
$\overline{\text{RESETP}}$ pulse width	t_{RESPW}	20 *3	—	tcyc	26.7,
$\overline{\text{RESETP}}$ setup time*1	t_{RESPS}	23	—	ns	26.8
$\overline{\text{RESETP}}$ hold time	t_{RESPH}	2	—	ns	
$\overline{\text{RESETM}}$ pulse width	t_{RESMW}	12 *4	—	tcyc	
$\overline{\text{RESETM}}$ setup time	t_{RESMS}	3	—	ns	
$\overline{\text{RESETM}}$ hold time	t_{RESMH}	34	—	ns	
$\overline{\text{BREQ}}$ setup time	t_{BREQS}	12	—	ns	26.10
$\overline{\text{BREQ}}$ hold time	t_{BREQH}	3	—	ns	
NMI setup time *1	t_{NMIS}	12	—	ns	26.8
NMI hold time	t_{NMIH}	4	—	ns	
IRQ7–IRQ0 setup time *1	t_{IRQS}	12	—	ns	
IRQ7–IRQ0 hold time	t_{IRQH}	4	—	ns	
$\overline{\text{IRQOUT}}$ delay time	t_{IRQOD}	—	14	ns	26.9
$\overline{\text{BACK}}$ delay time	t_{BACKD}	—	14	ns	26.10,
STATUS1, STATUS0 delay time	t_{STD}	—	18	ns	26.11
Bus tri-state delay time 1	t_{BOFF1}	0	30	ns	
Bus tri-state delay time 2	t_{BOFF2}	0	30	ns	
Bus tri-state delay time 3	t_{BOFF3}	$2B_{cyc}$*5	—	ns	
Bus buffer-on time 1	t_{BON1}	0	30	ns	
Bus buffer-on time 2	t_{BON2}	0	30	ns	
Bus buffer-on time 3	t_{BON3}	$2B_{cyc}$*5	—	ns	

Notes: *1 $\overline{\text{RESETP}}$, NMI, and IRQ7–IRQ0 are asynchronous. Changes are detected at the clock fall when the setup shown is used. When the setup cannot be used, detection can be delayed until the next clock falls.

*2 The upper limit of the external bus clock is 33 MHz.

*3 In the standby mode, t_{RESPW} = t_{OSC2} (10 ms).
When the clock multiplication ratio is changed, t_{RESPW} = t_{PLL1} (100 µs).

*4 In the standby mode, t_{RESMW} = t_{OSC2} (10 ms).
When the clock multiplication ratio is changed, $\overline{\text{RESETM}}$ must be kept low until STATUS (0–1) changes to reset (HH).

*5 B_{cyc} is the external bus clock cycle (B clock cycle).

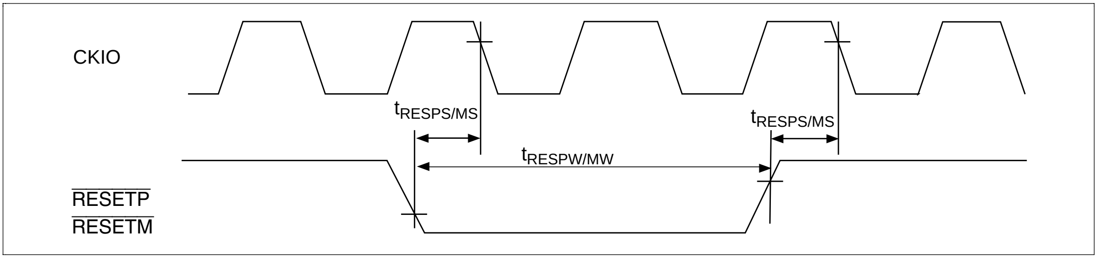


Figure 26.7 Reset Input Timing

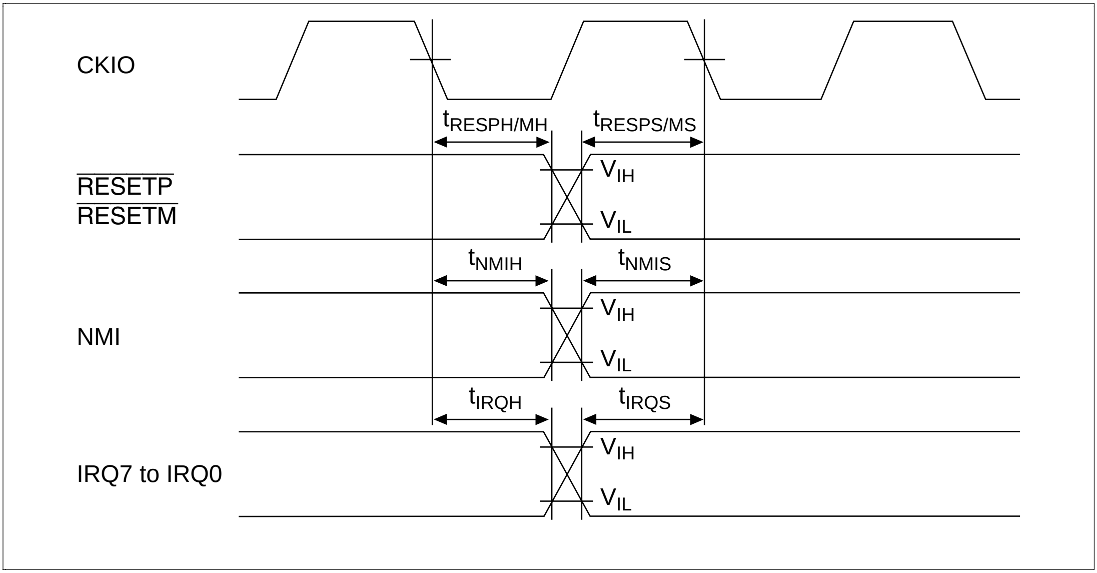


Figure 26.8 Interrupt Signal Input Timing

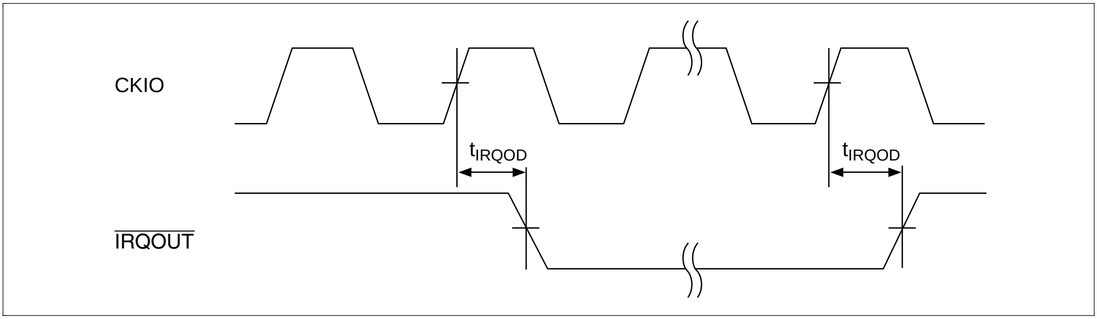


Figure 26.9 $\overline{\text{IRQOUT}}$ Timing

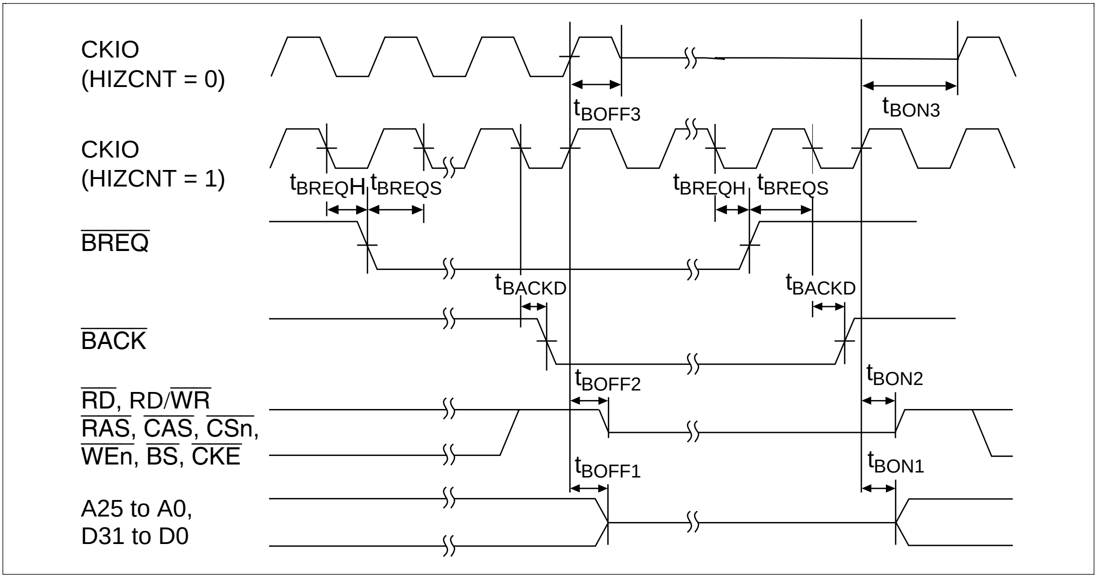


Figure 26.10 Bus Release Timing

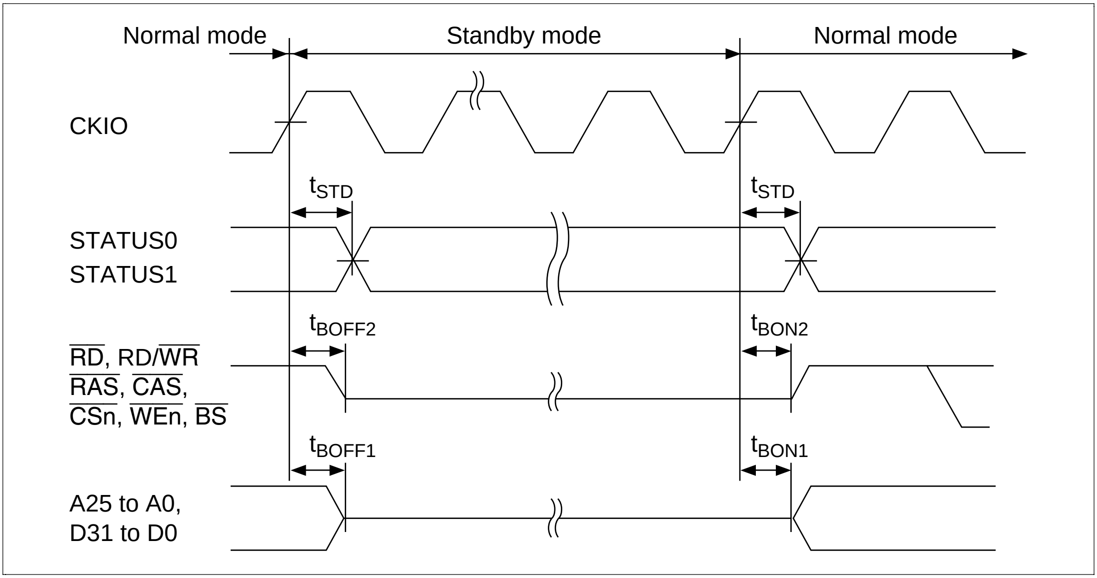


Figure 26.11 Pin Drive Timing at Standby

26.3.3 AC Bus Timing

Table 26.7 Bus Timing
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

		–33			
Item	**Symbol**	**Min**	**Max**	**Unit**	**Figure**
Address delay time	t_{AD}	1	15	ns	26.12–26.32, 26.35
Address hold time	t_{AH}	10	—	ns	26.12–26.17
$\overline{\text{BS}}$ delay time	t_{BSD}	—	14	ns	26.12–26.32, 26.35
$\overline{\text{CS}}$ delay time 1	t_{CSD1}	1	14	ns	26.12–26.32, 26.35
$\overline{\text{CS}}$ delay time 2	t_{CSD2}	1	14	ns	26.12–26.17
Read/write delay time	t_{RWD}	2	12	ns	26.12–26.32, 26.35
Read/write hold time	t_{RWH}	0	—	ns	26.12–26.17
Read strobe delay time	t_{RSD}	—	12	ns	26.12–26.17
Read data setup time 1	t_{RDS1}	12	—	ns	26.12–26.17
Read data setup time 2	t_{RDS2}	7	—	ns	26.18–26.21
Read data hold time 1	t_{RDH1}	0	—	ns	26.12–26.17
Read data hold time 2	t_{RDH2}	2	—	ns	26.18–26.21, 26.26–26.30
Write enable delay time	t_{WED}	1	14	ns	26.12–26.14
Write data delay time 1	t_{WDD1}	—	17	ns	26.12–26.14
Write data delay time 2	t_{WDD2}	—	16	ns	26.22–26.25
Write data hold time 1	t_{WDH1}	2	—	ns	26.12–26.14
Write data hold time 2	t_{WDH2}	2	—	ns	26.22–26.25
Write data hold time 3	t_{WDH3}	2	—	ns	26.12–26.14
$\overline{\text{WAIT}}$ setup time	t_{WTS}	12	—	ns	26.13–26.17
$\overline{\text{WAIT}}$ setup time 2	t_{WTS2}	7	—	ns	26.14(2)
$\overline{\text{WAIT}}$ hold time	t_{WTH}	4	—	ns	26.13–26.17, 26.29
$\overline{\text{RAS}}$ delay time 2	t_{RASD2}	1	15	ns	26.18–26.35
$\overline{\text{CAS}}$ delay time 2	t_{CASD2}	1	15	ns	26.18–26.35
$\overline{\text{DQM}}$ delay time	t_{DQMD}	1	15	ns	26.18–26.32
DACK delay time 1	t_{DAKD1}	—	15	ns	26.12–26.32

26.3.4 Basic Timing

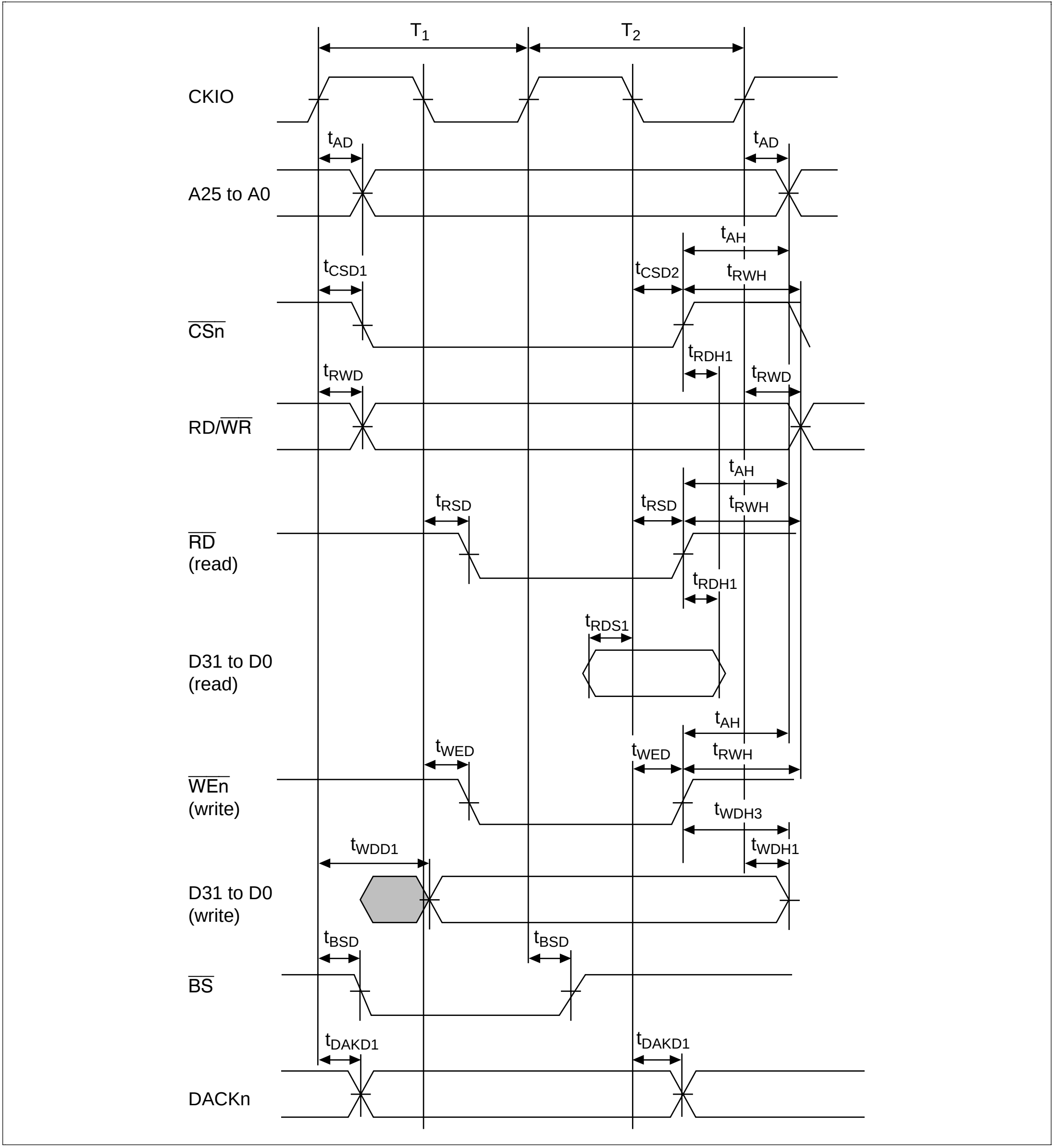


Figure 26.12 Basic Bus Cycle (No Wait)

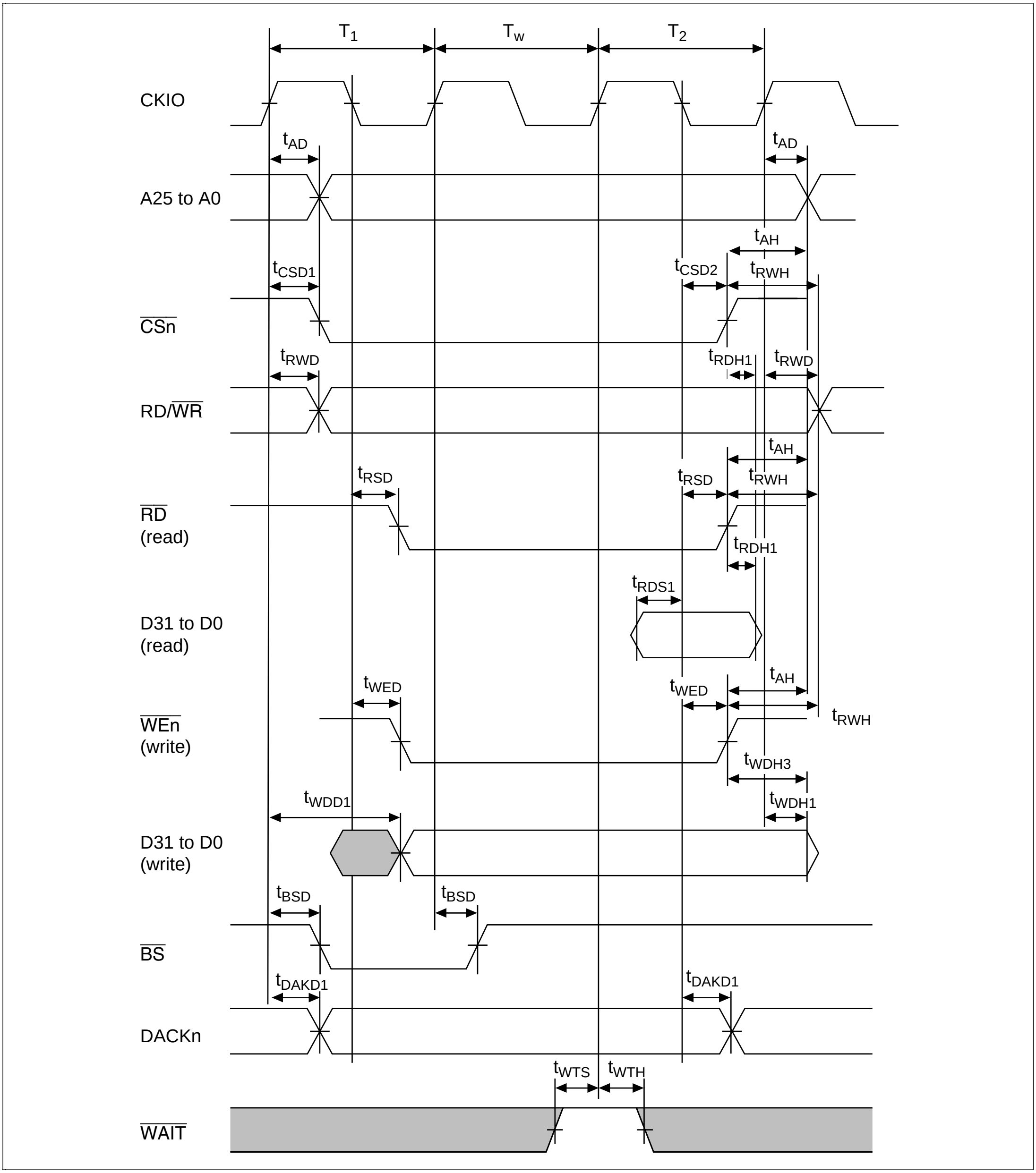


Figure 26.13 Basic Bus Cycle (One Wait)

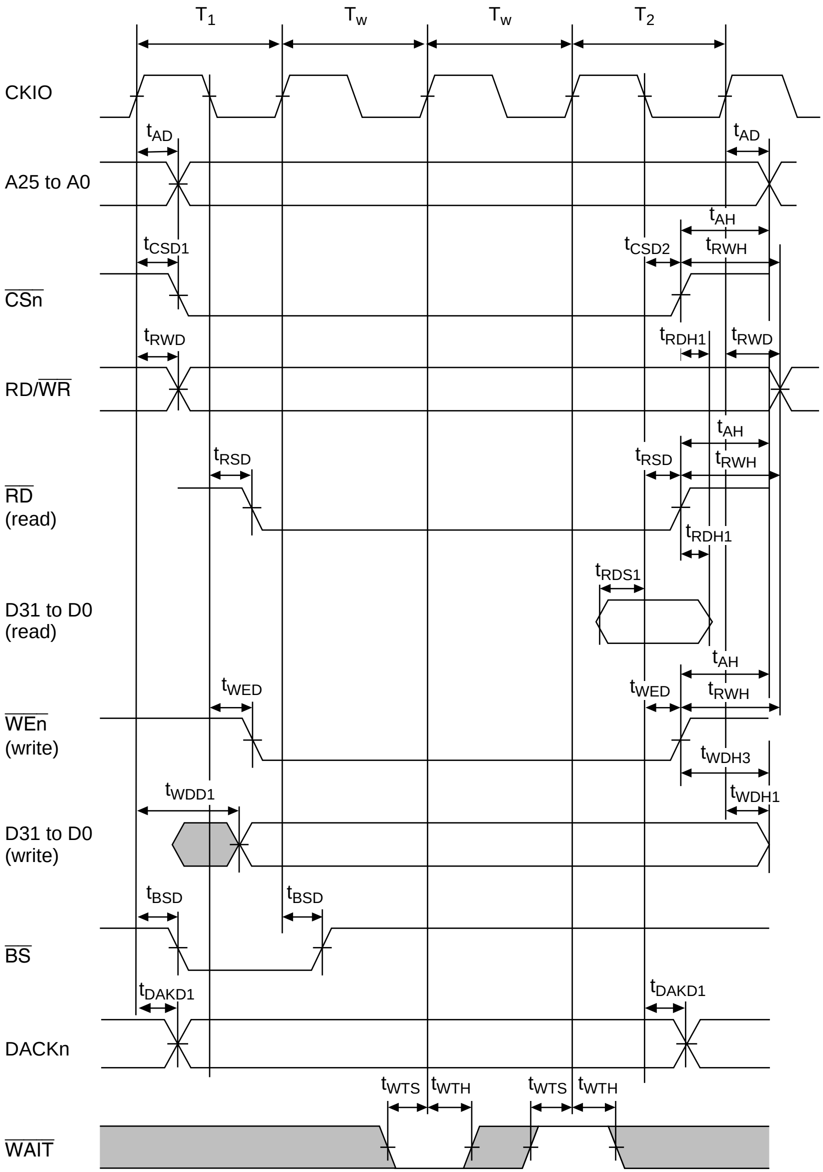


Figure 26.14 (1) Basic Bus Cycle (External Wait, WAITSEL = 0)

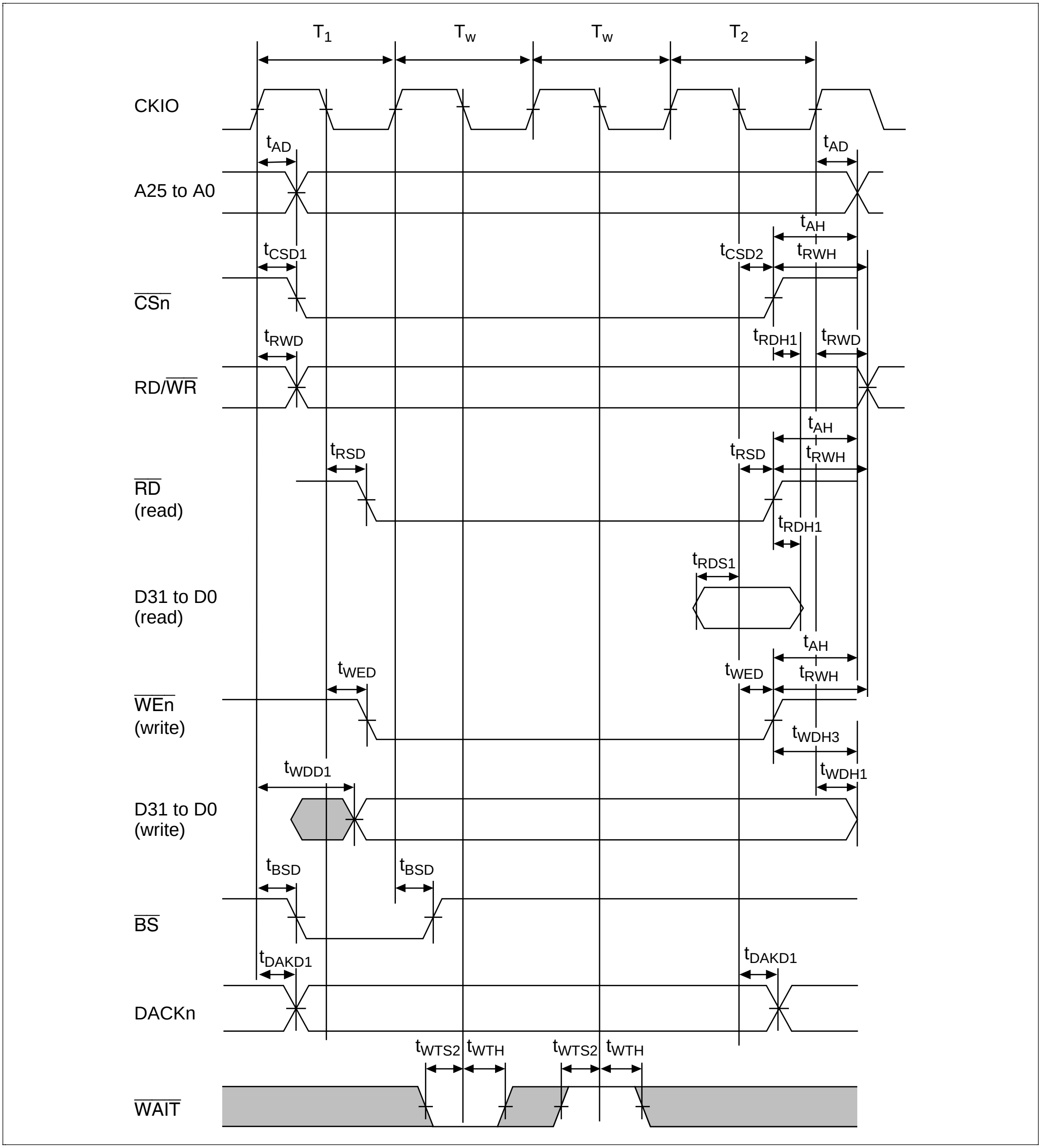
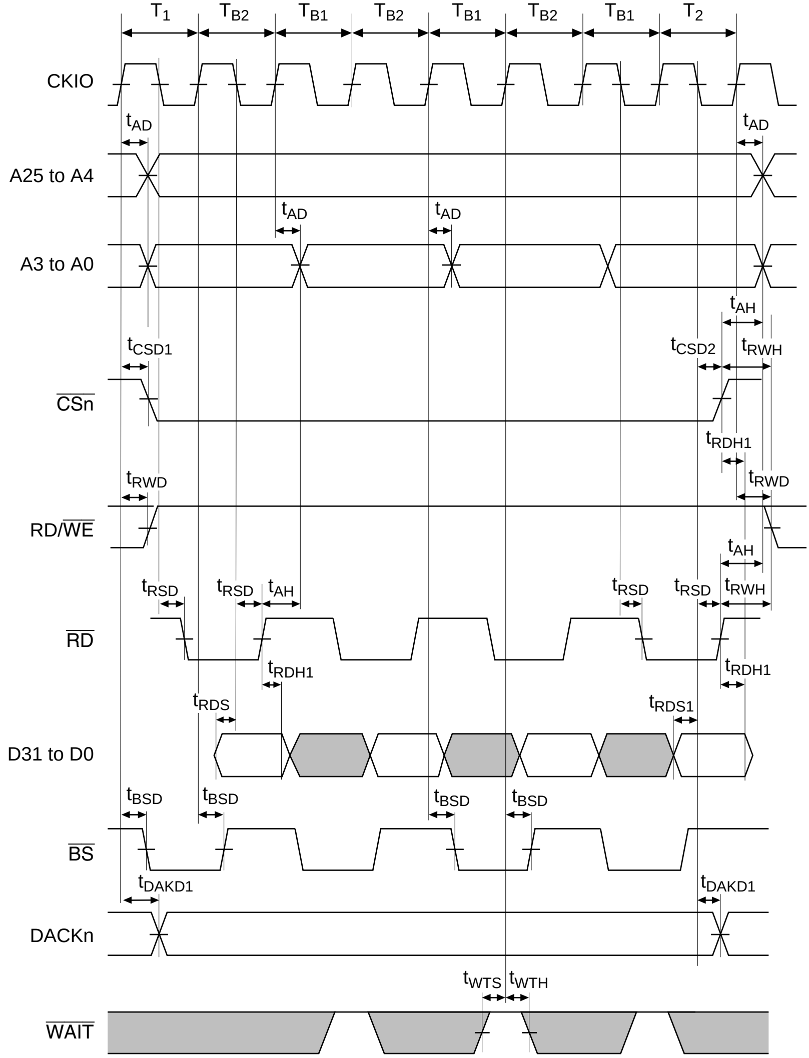


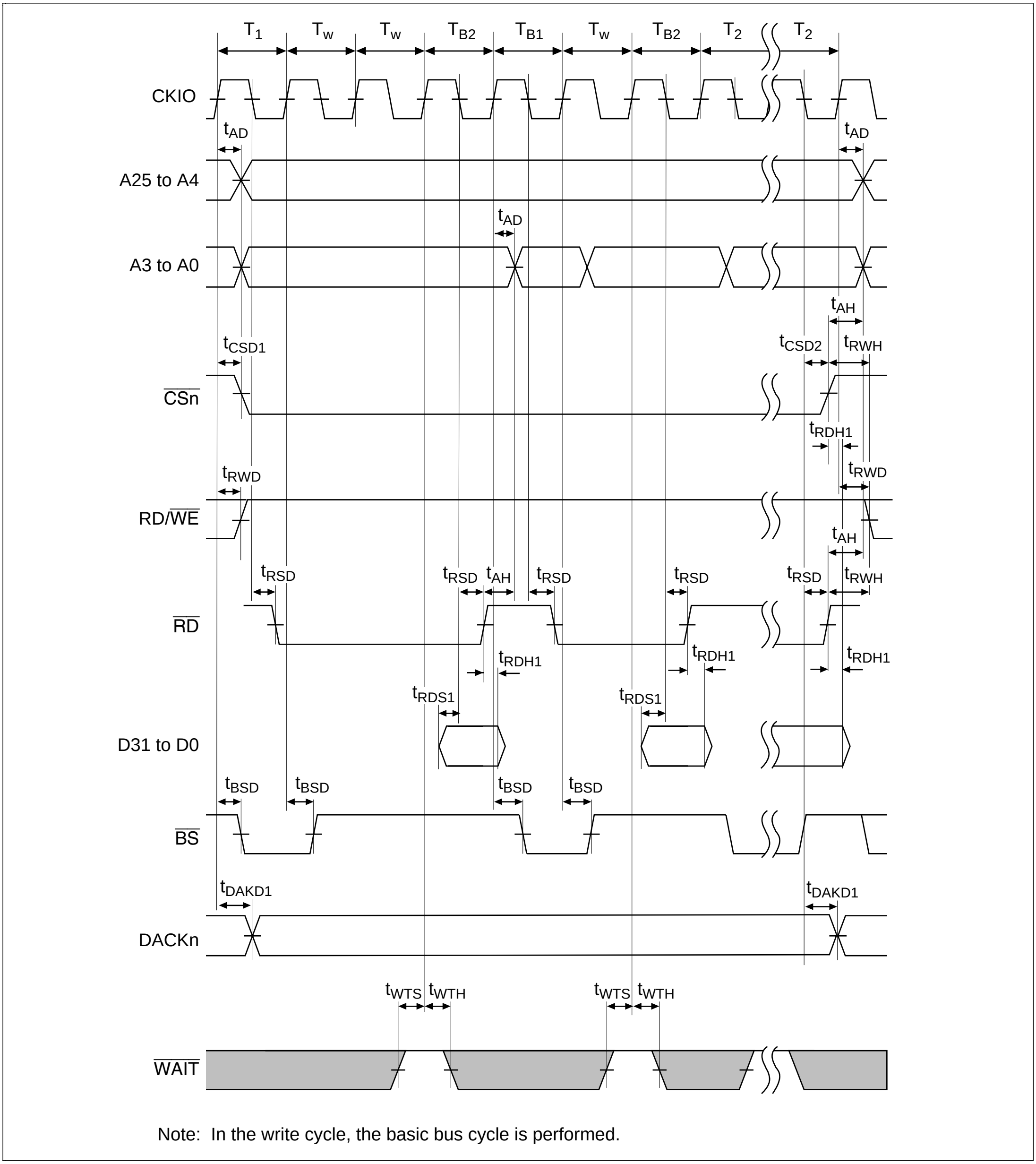
Figure 26.14 (2) Basic Bus Cycle (External Wait, WAITSEL = 1)

26.3.5 Burst ROM Timing



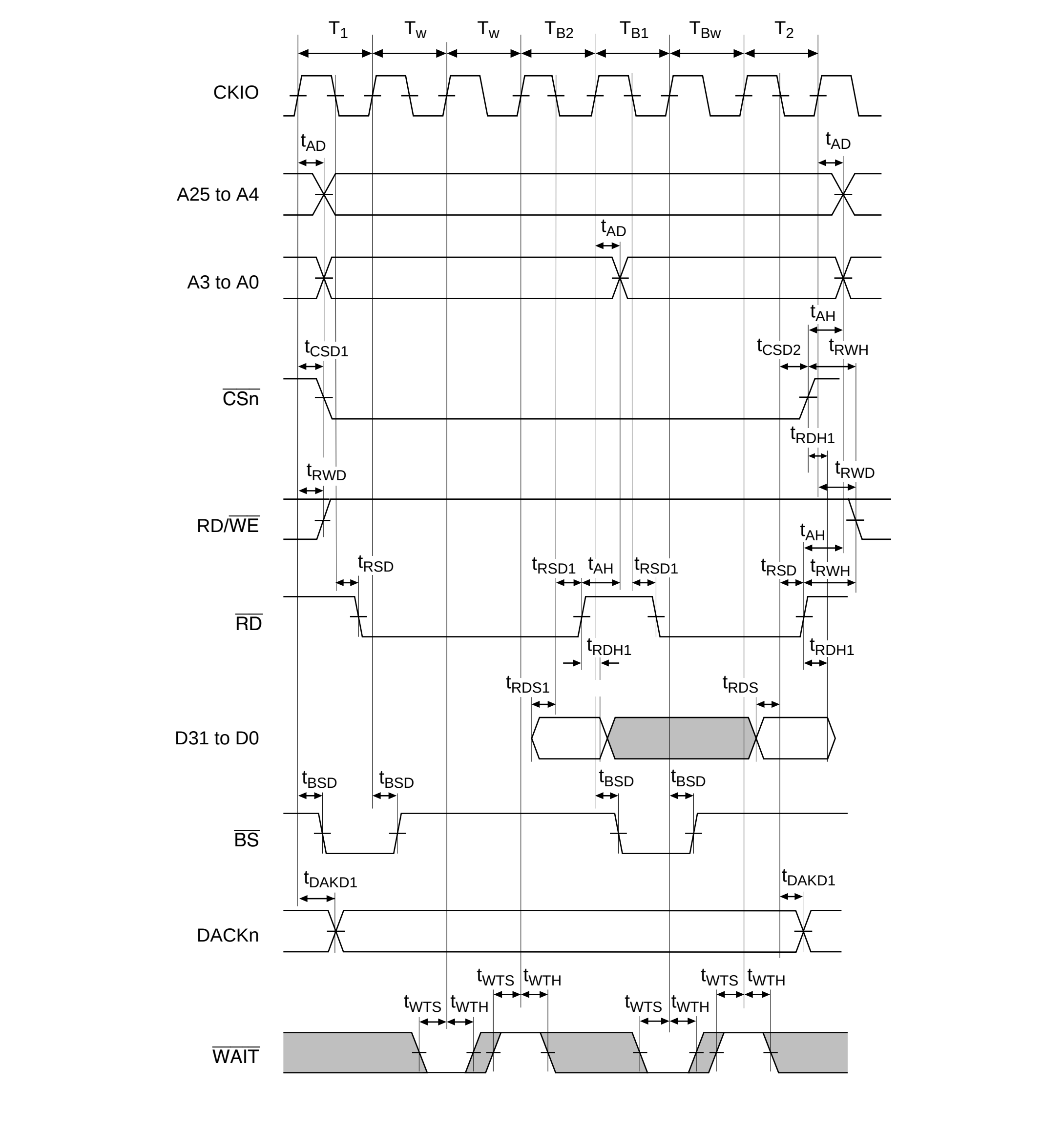
Note: In the write cycle, the basic bus cycle, the basic bus cycle is performed.

Figure 26.15 Burst ROM Bus Cycle (No Wait)



Note: In the write cycle, the basic bus cycle is performed.

Figure 26.16 Burst ROM Bus Cycle (Two Waits)



Note: In the write cycle, the basic bus cycle is performed.

Figure 26.17 Burst ROM Bus Cycle (External Wait, WAITSEL = 0)

26.3.6 Synchronous DRAM Timing

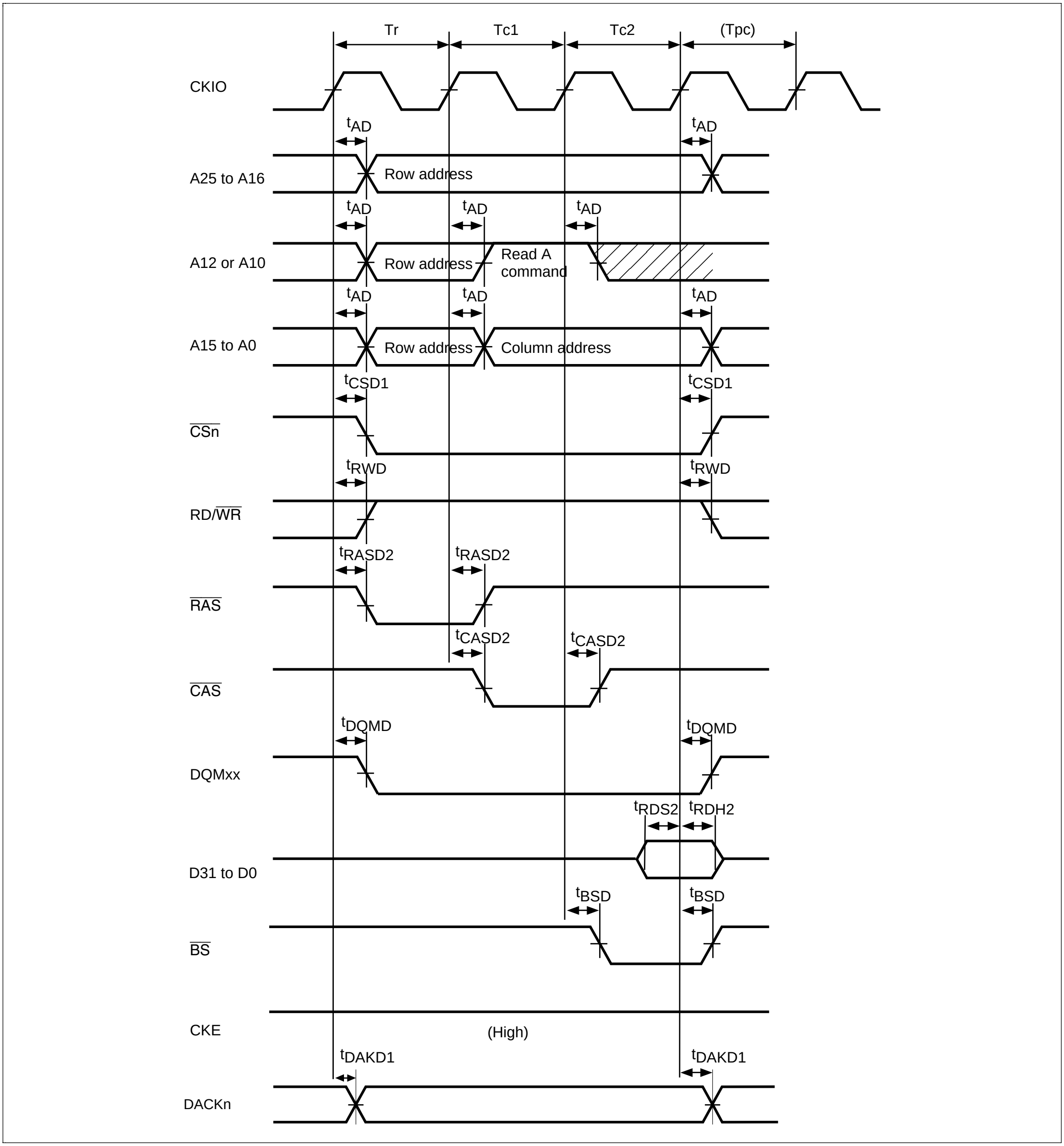


Figure 26.18 Synchronous DRAM Read Bus Cycle (RCD = 0, CAS Latency = 1, TPC = 0)

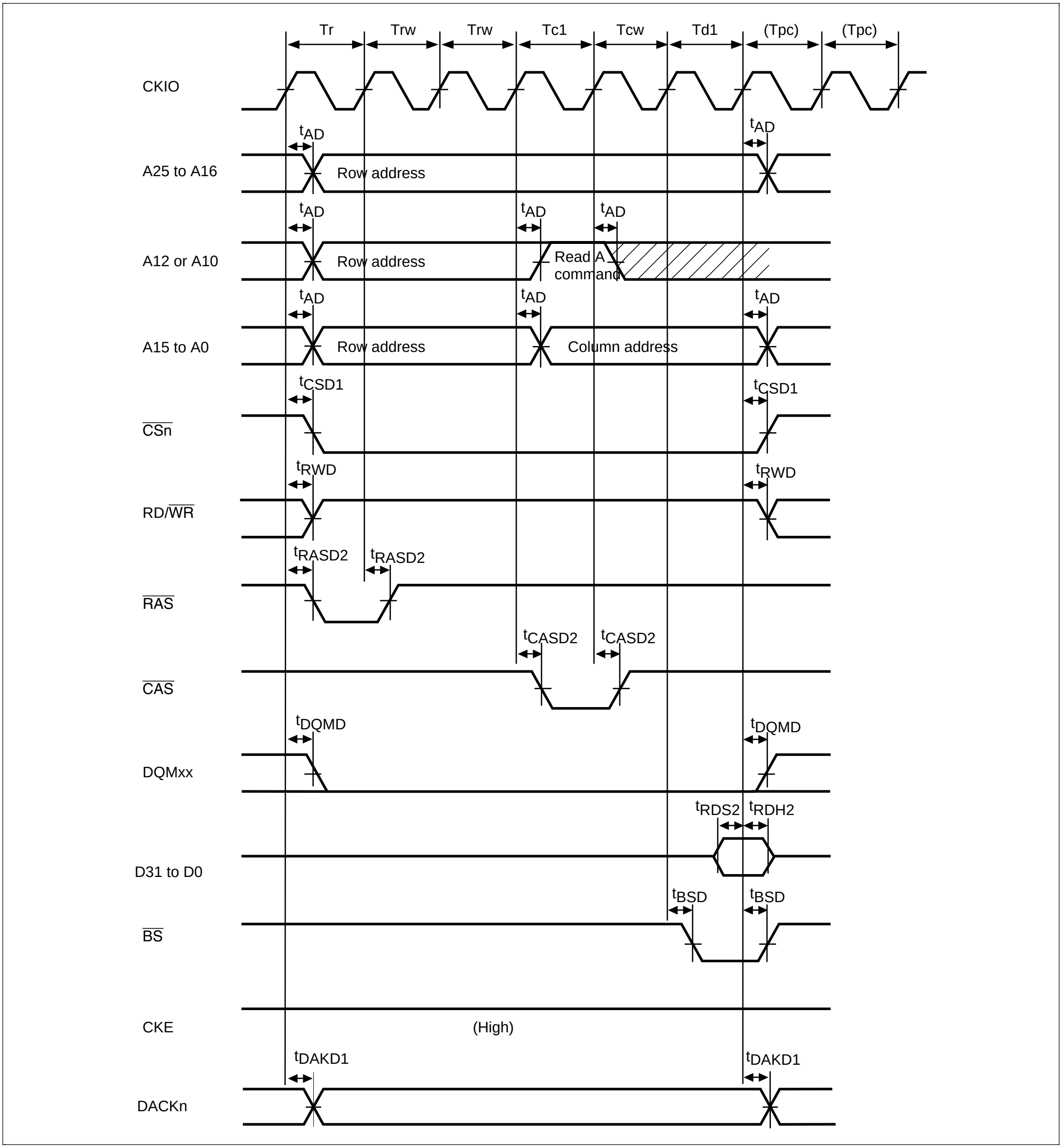


Figure 26.19 Synchronous DRAM Read Bus Cycle (RCD = 2, CAS Latency = 2, TPC = 1)

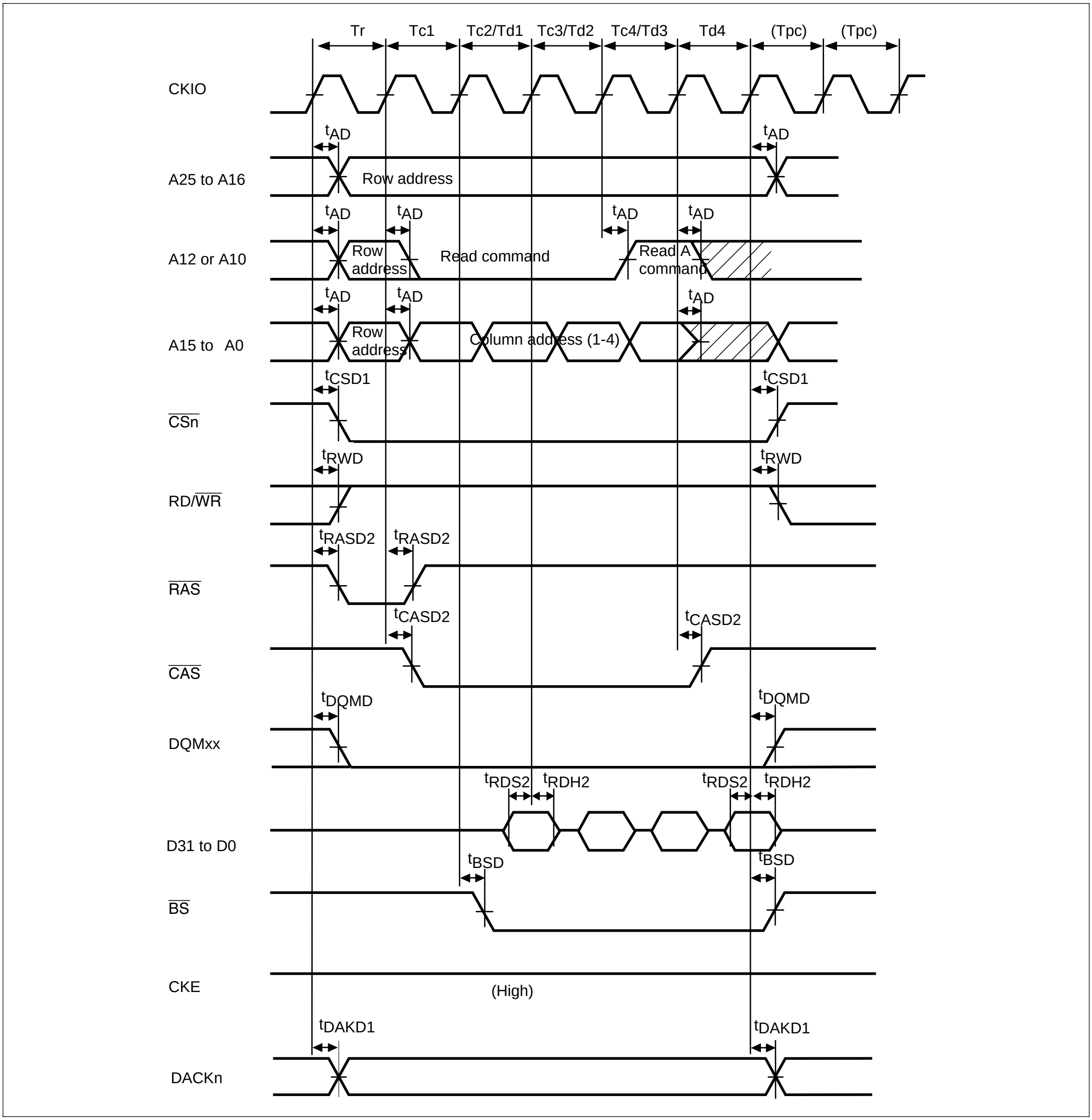


Figure 26.20 Synchronous DRAM Read Bus Cycle (Burst Read (Single Read × 4), RCD = 0, CAS Latency = 1, TPC = 1)

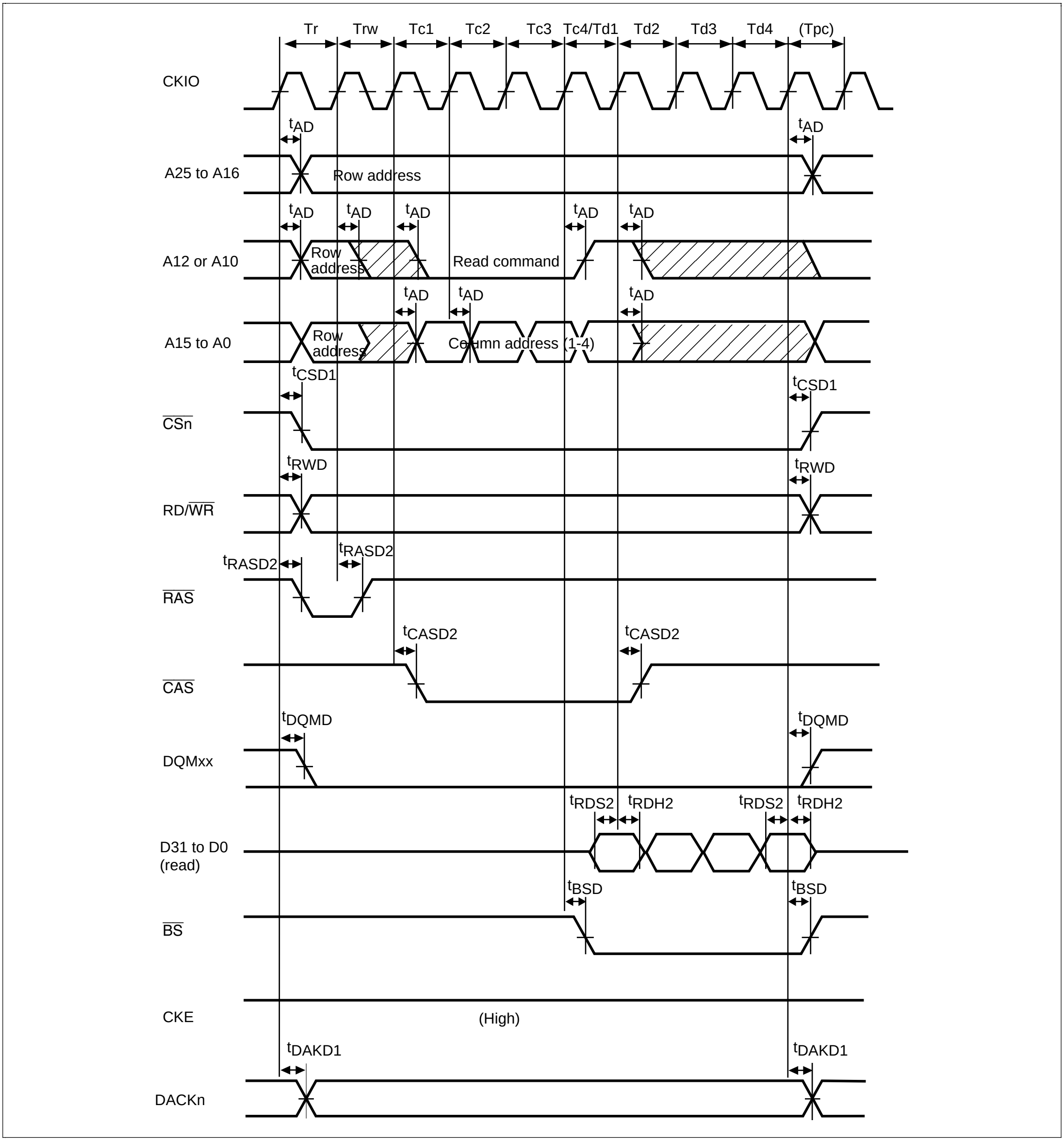


Figure 26.21 Synchronous DRAM Read Bus Cycle (Burst Read (Single Read × 4), RCD = 1, CAS Latency = 3, TPC = 0)

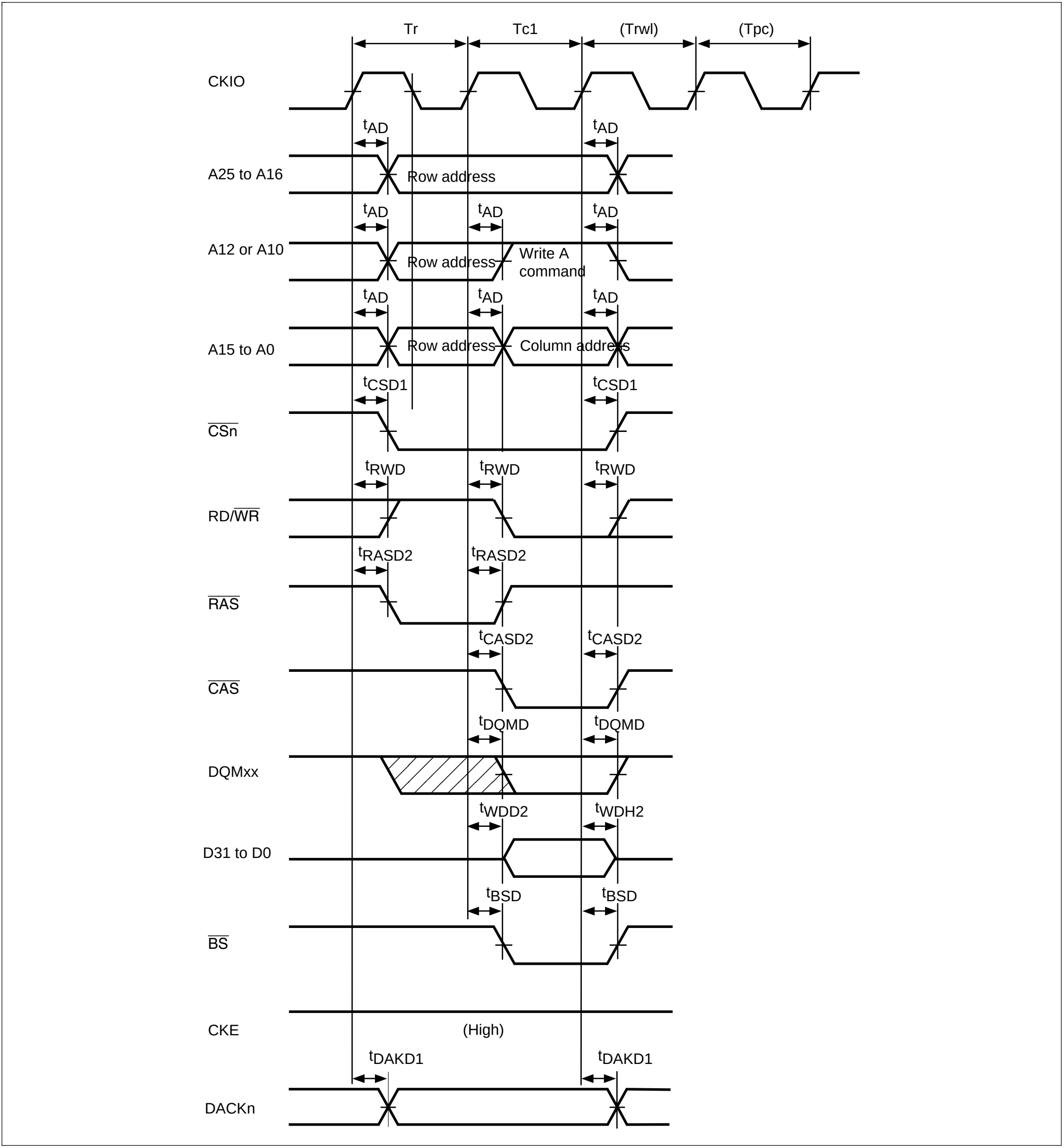


Figure 26.22 Synchronous DRAM Write Bus Cycle (RCD = 0, TPC = 0, TRWL = 0)

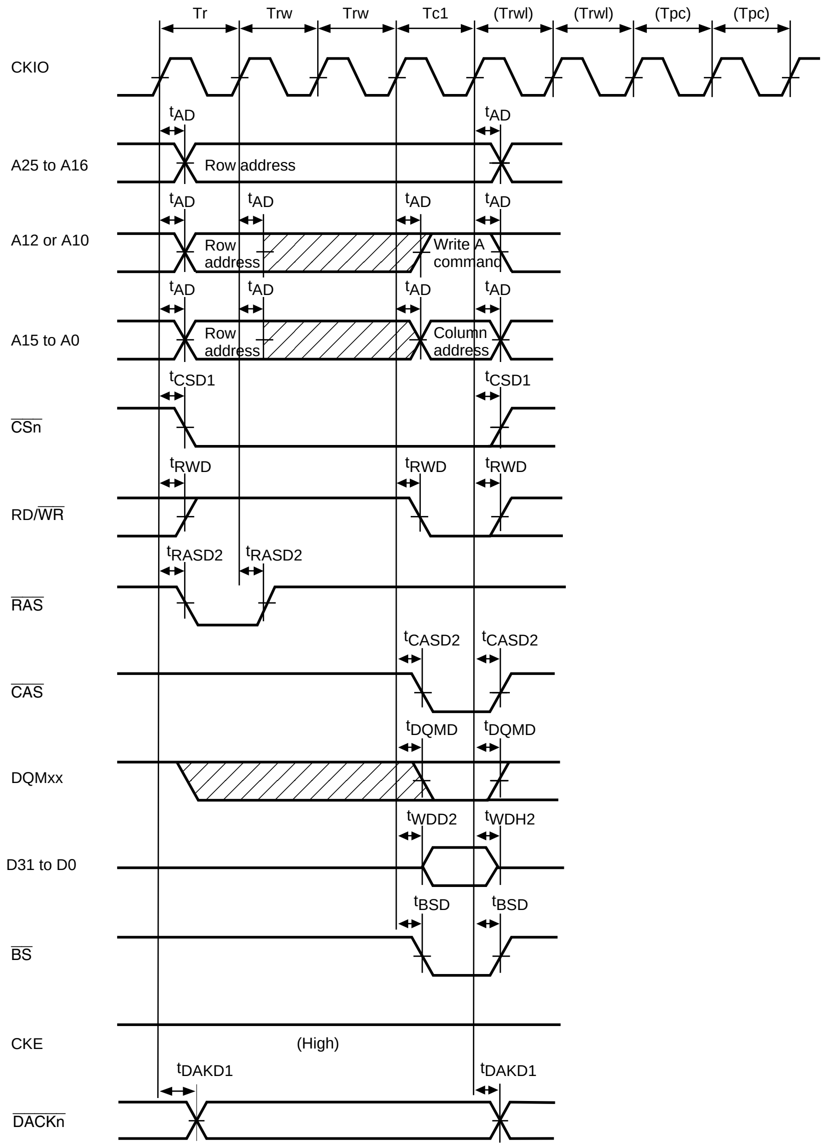


Figure 26.23 Synchronous DRAM Write Bus Cycle (RCD = 2, TPC = 1, TRWL = 1)

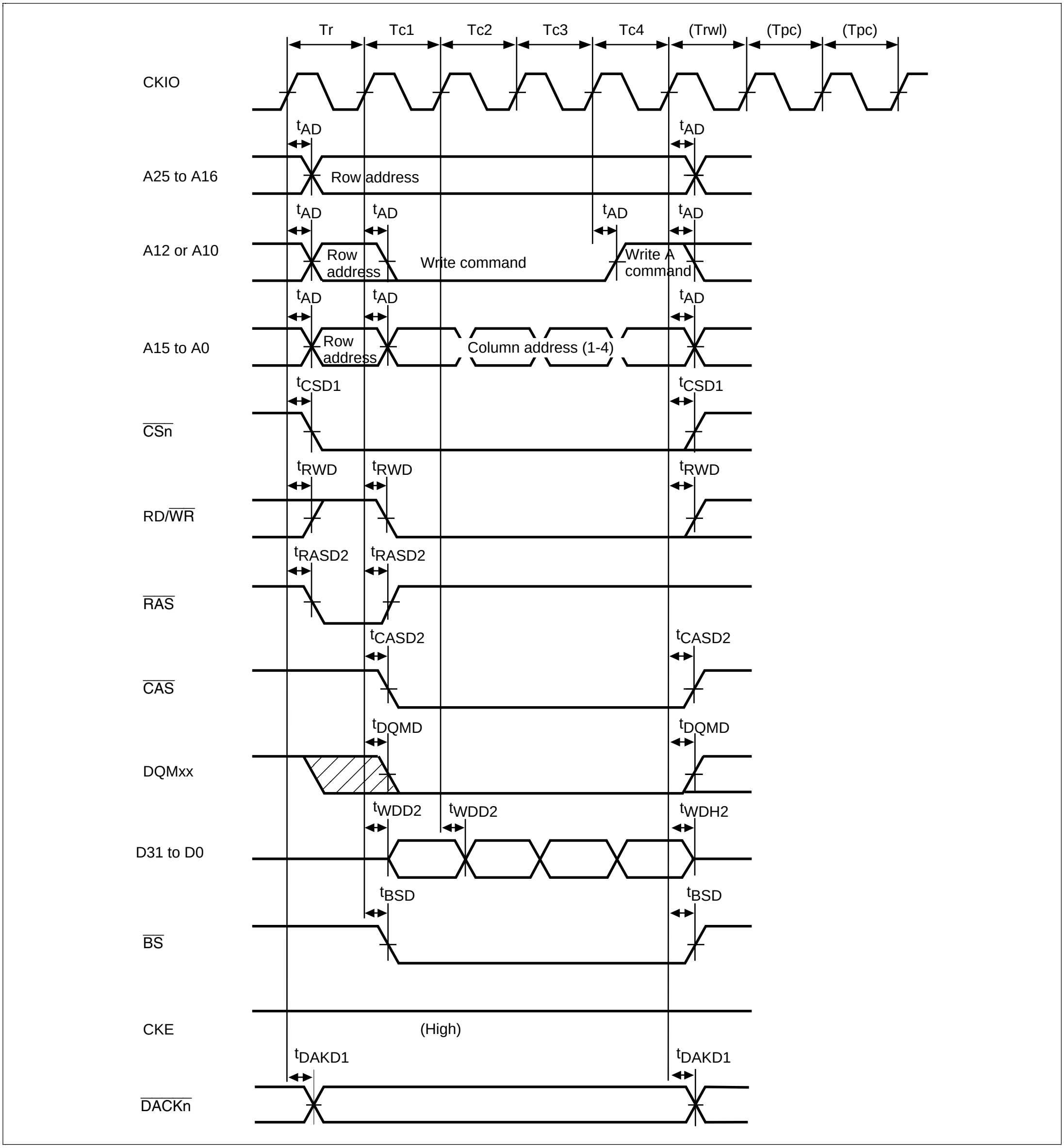


Figure 26.24 Synchronous DRAM Write Bus Cycle (Burst Mode (Single Write × 4), RCD = 0, TPC = 1, TRWL = 0)

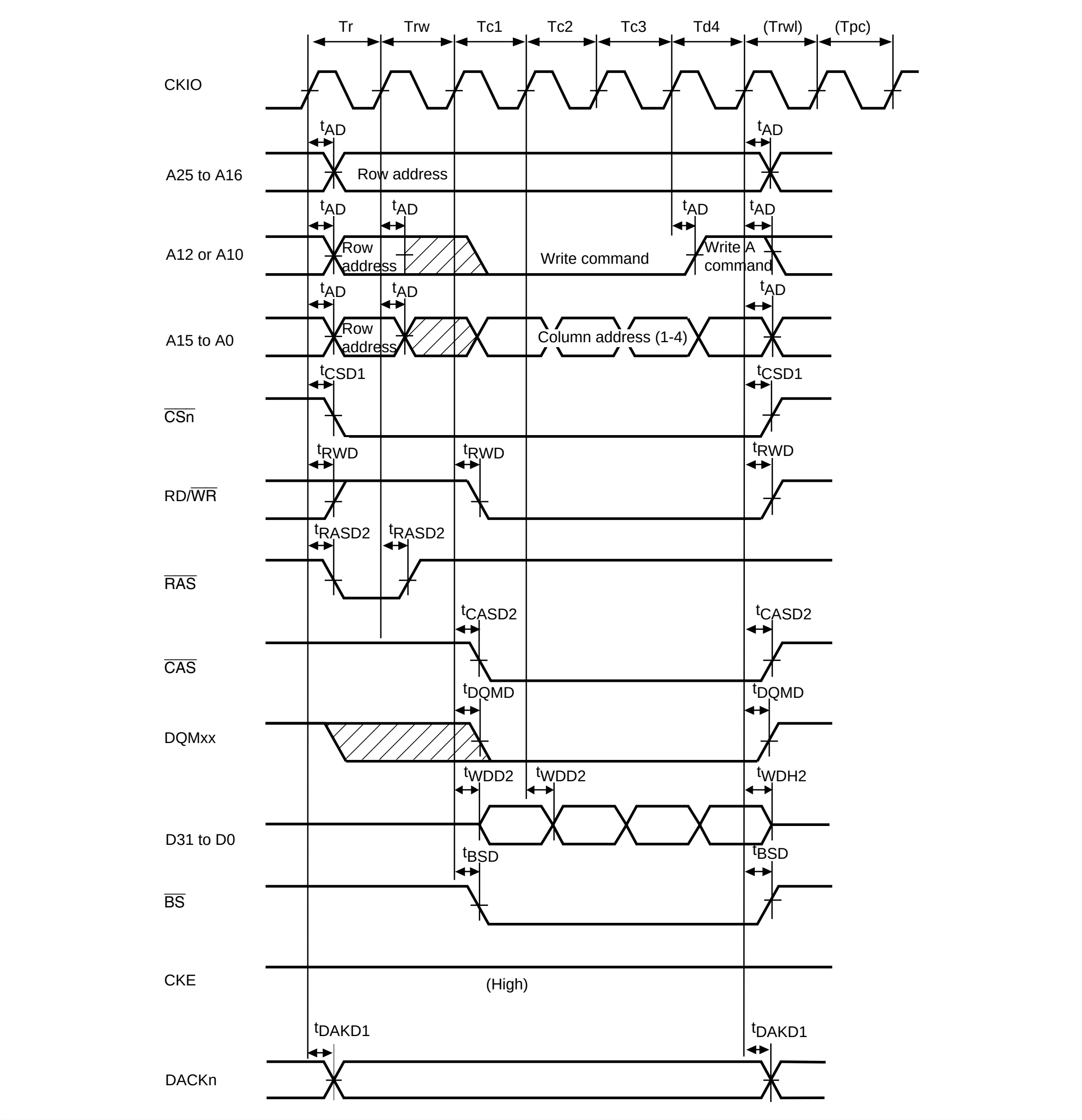


Figure 26.25 Synchronous DRAM Write Bus Cycle (Burst Mode (Single Write × 4), RCD = 1, TPC = 0, TRWL = 0)

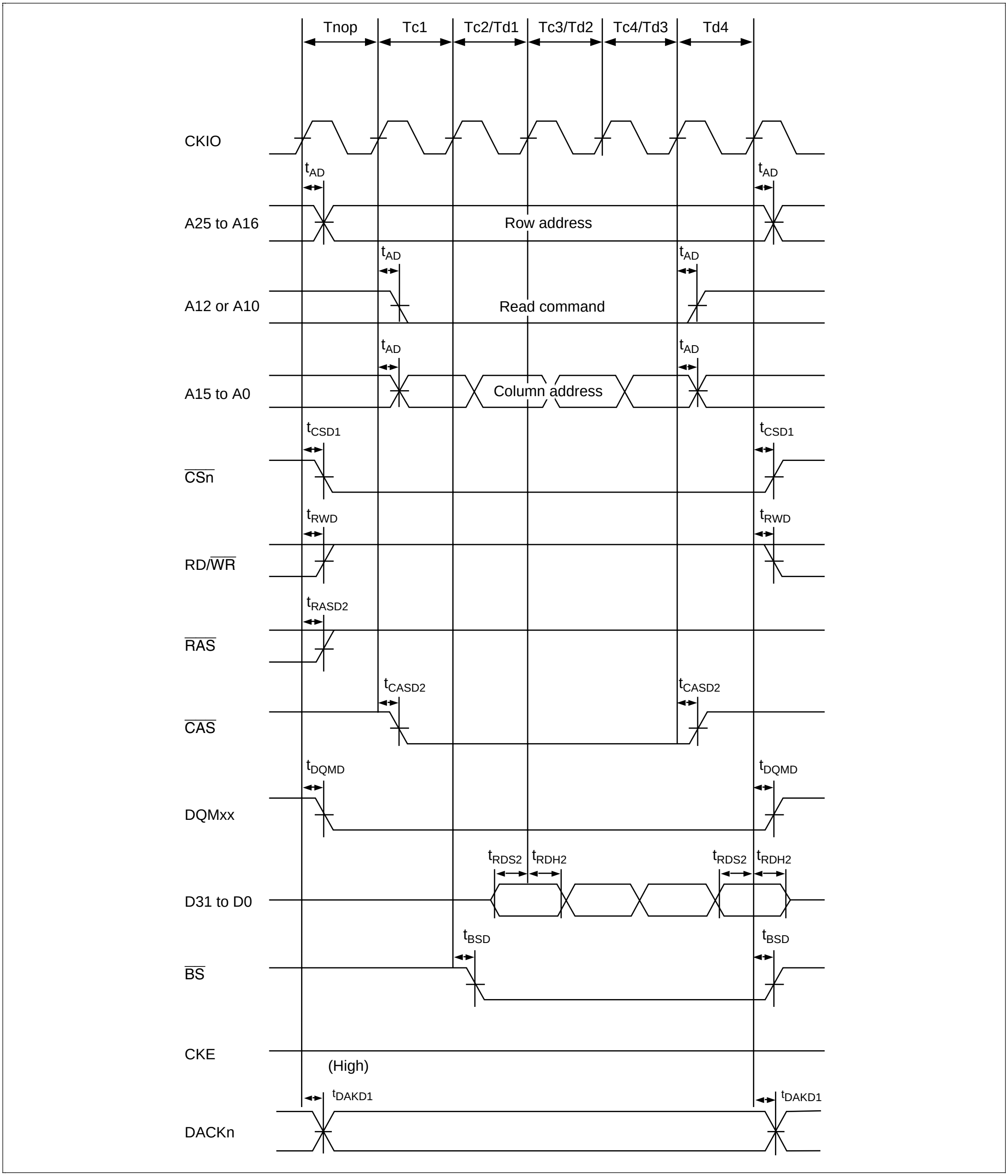


Figure 26.26 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Same Row Address, CAS Latency = 1)

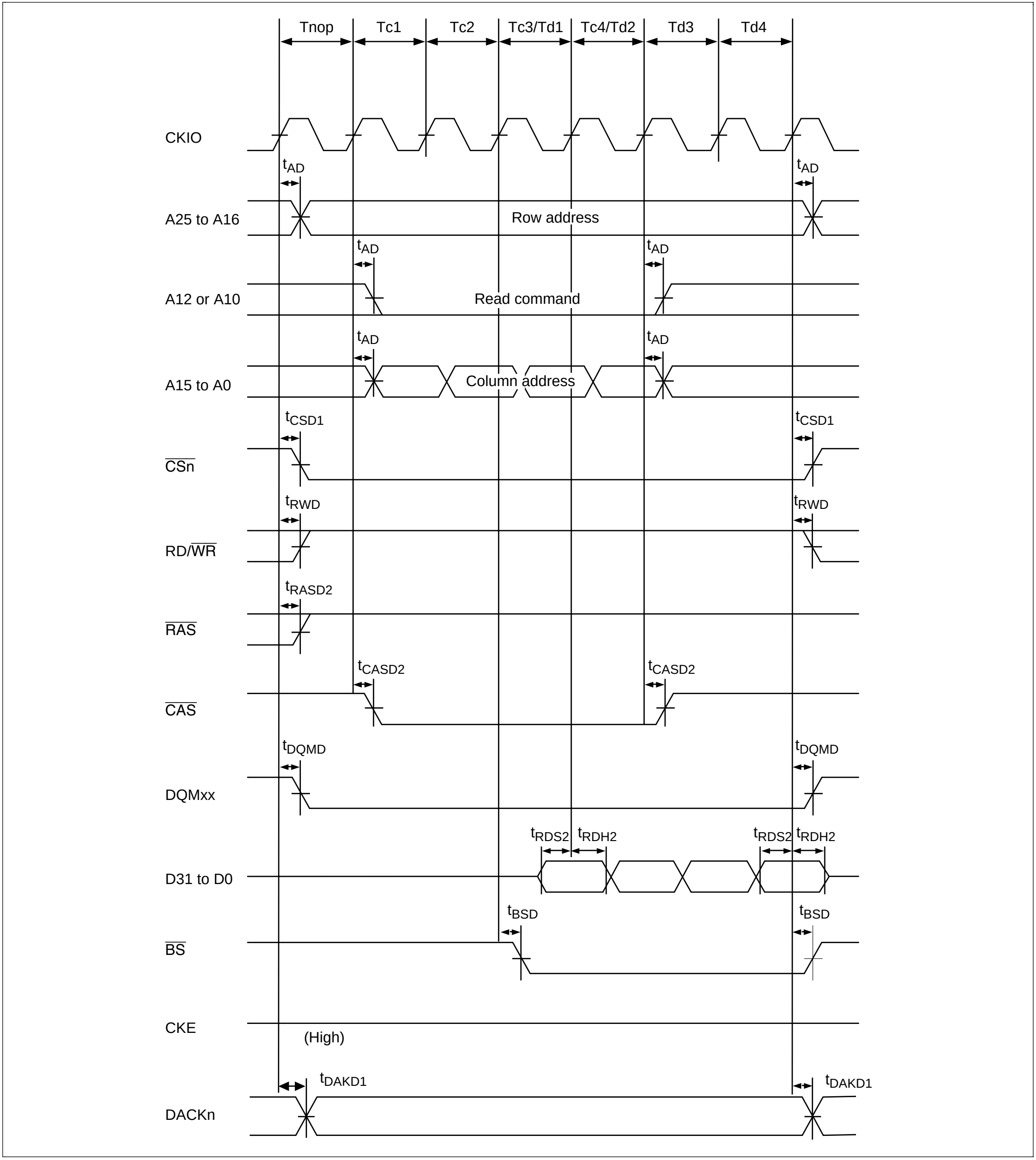


Figure 26.27 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Same Row Address, CAS Latency = 2)

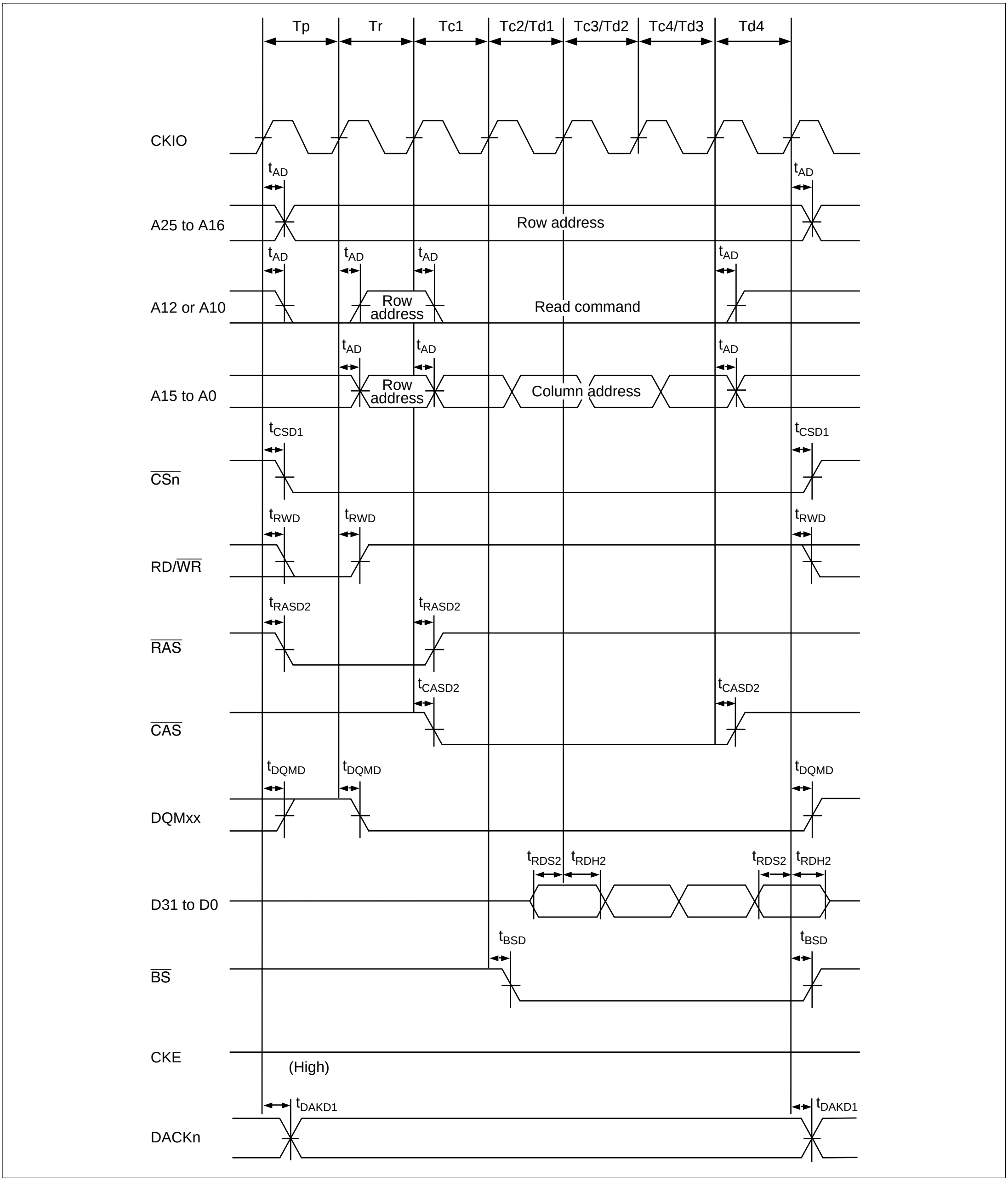


Figure 26.28 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Different Row Address, TPC = 0, RCD = 0, CAS Latency = 1)

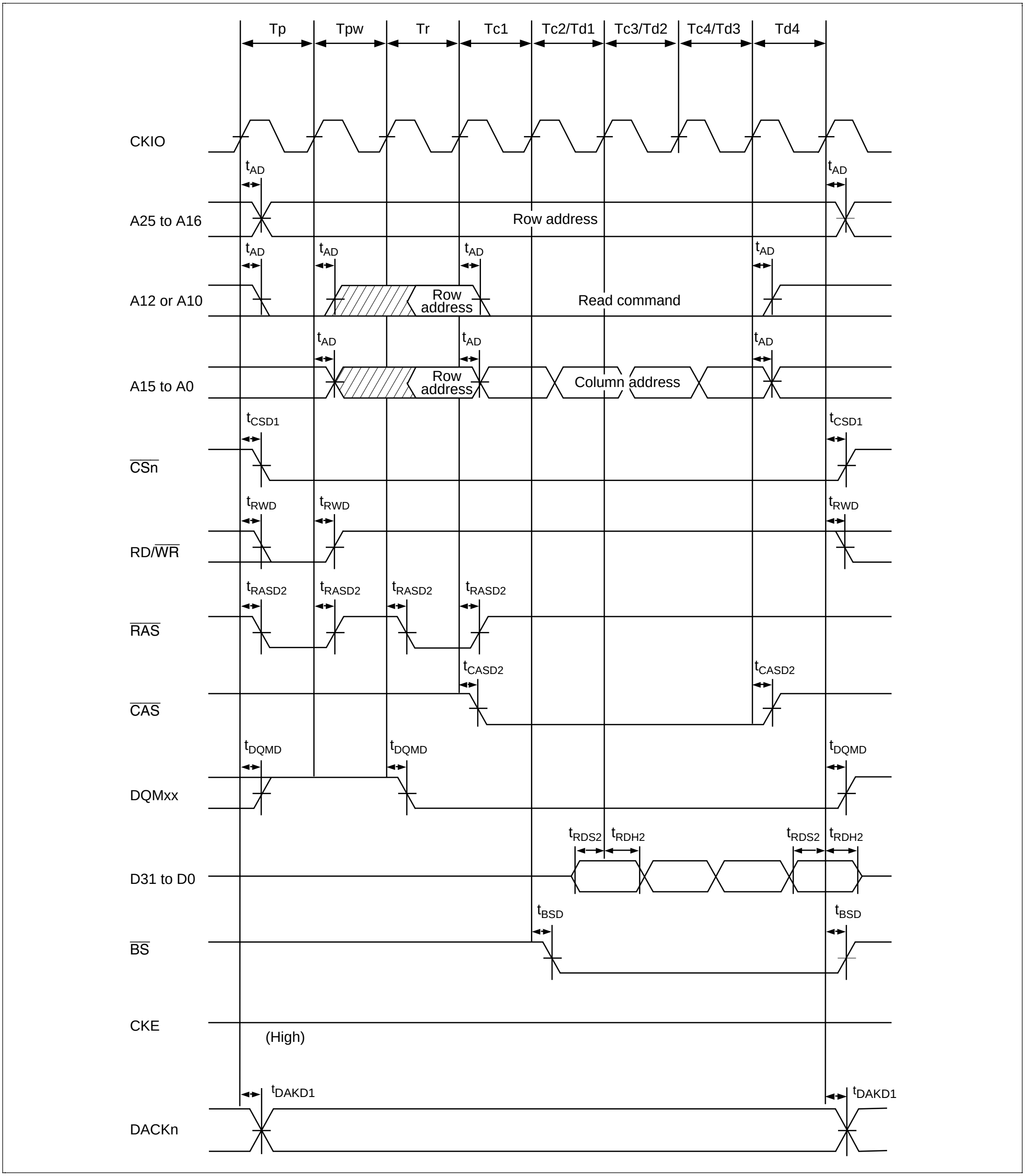


Figure 26.29 Synchronous DRAM Burst Read Bus Cycle (RAS Down, Different Row Address, TPC = 1, RCD = 0, CAS Latency = 1)

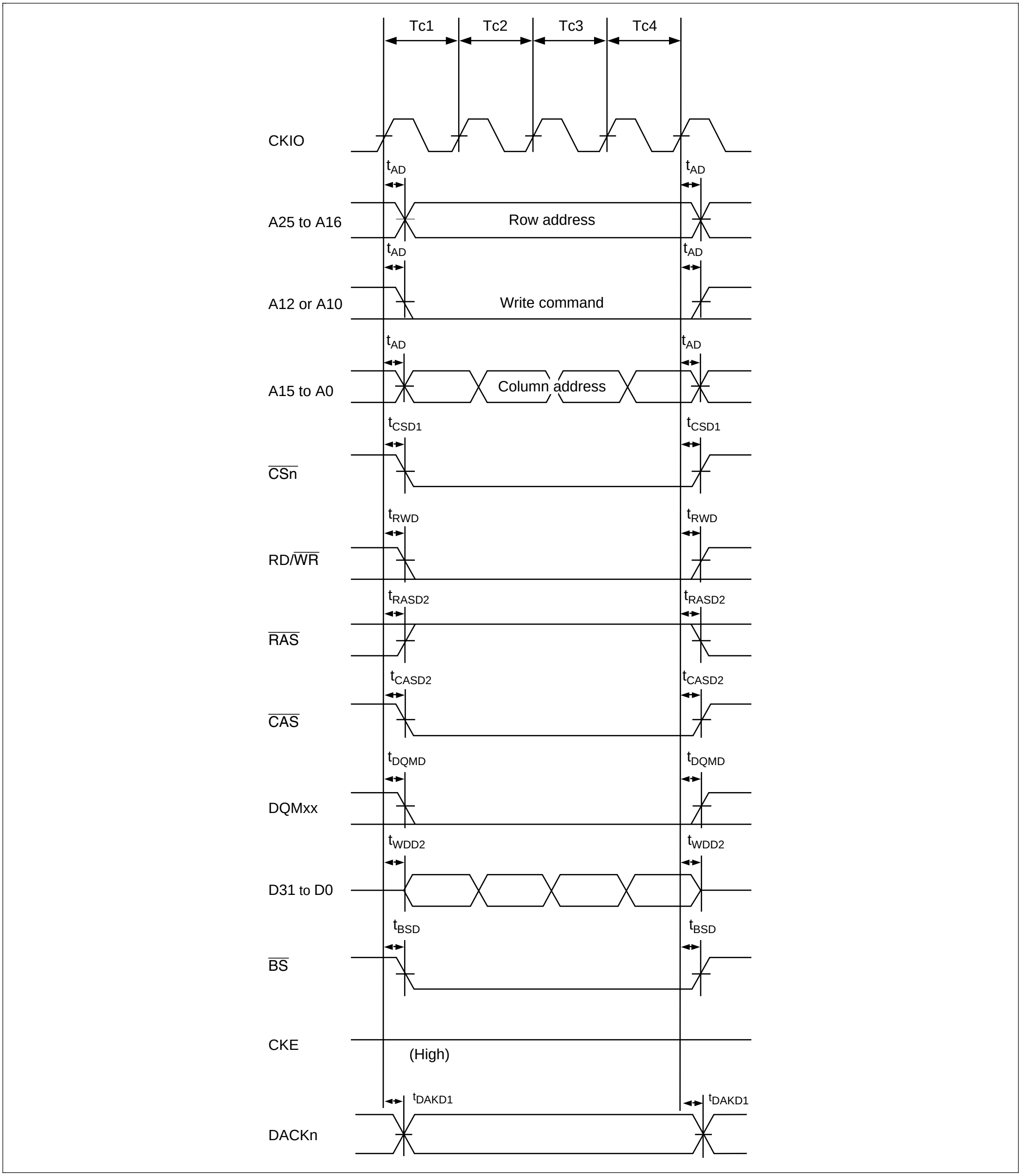


Figure 26.30 Synchronous DRAM Burst Write Bus Cycle (RAS Down, Same Row Address)

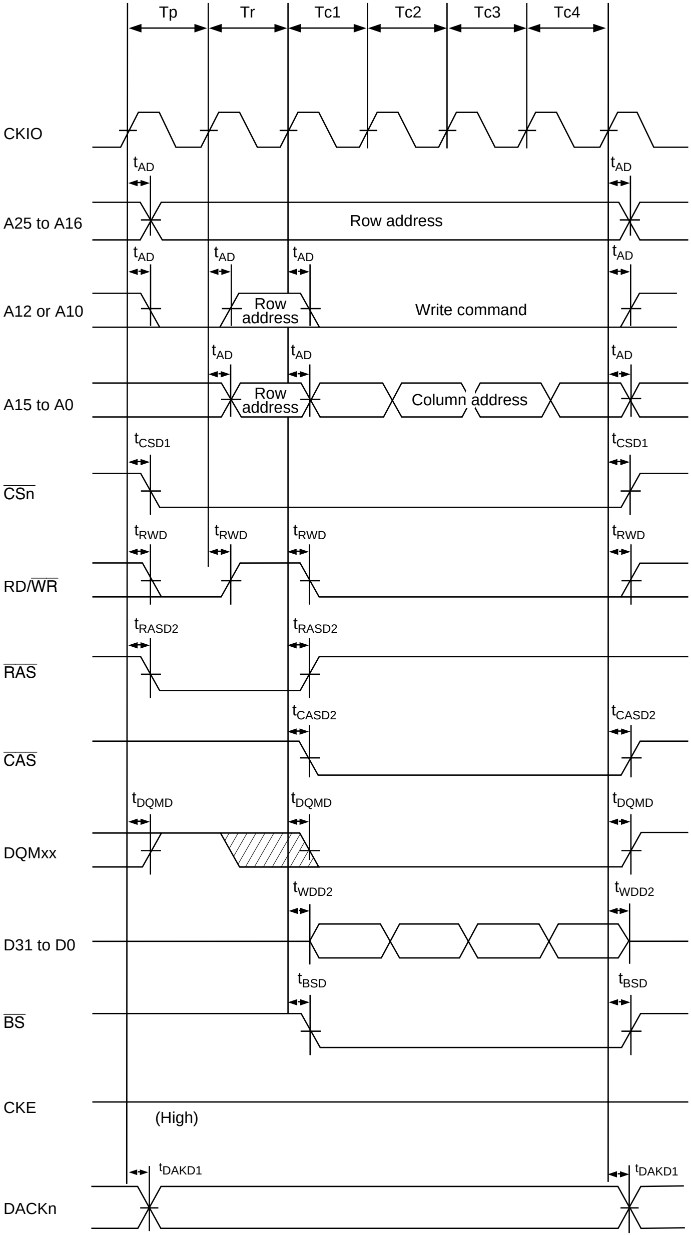


Figure 26.31 Synchronous DRAM Burst Write Bus Cycle (RAS Down, Different Row Address, TPC = 0, RCD = 0)

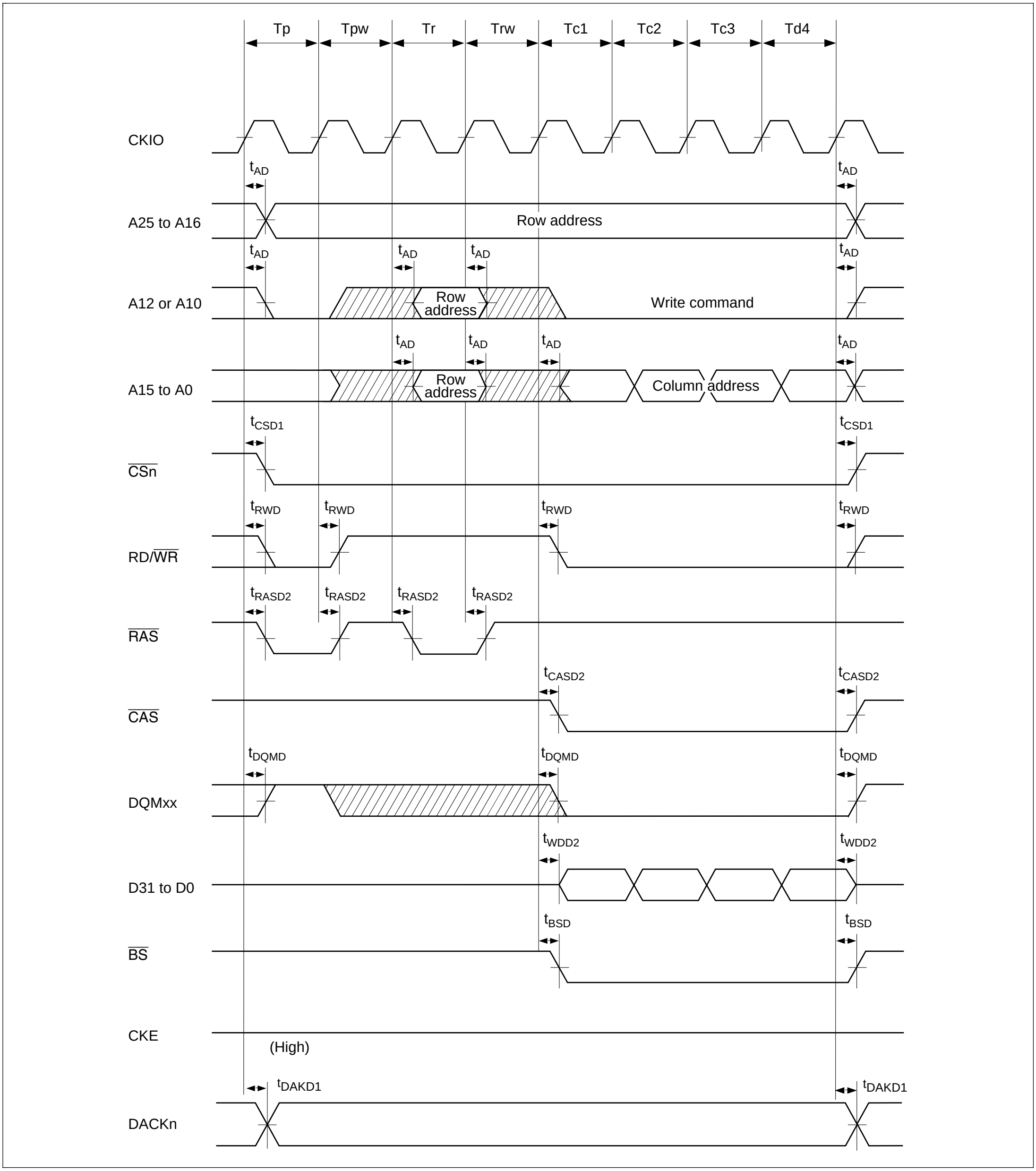


Figure 26.32 Synchronous DRAM Burst Write Bus Cycle (RAS Down, Different Row Address, TPC = 1, RCD = 1)

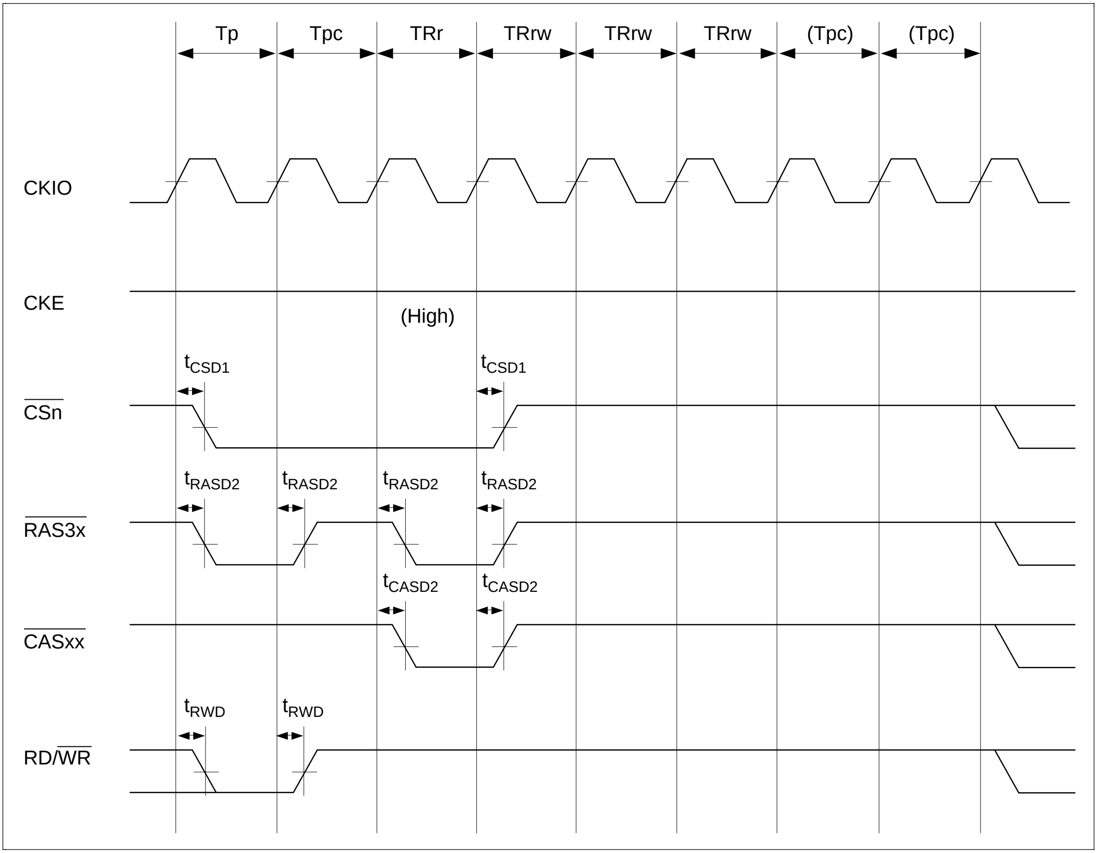


Figure 26.33 Synchronous DRAM Auto-Refresh Timing (TRAS = 1, TPC = 1)

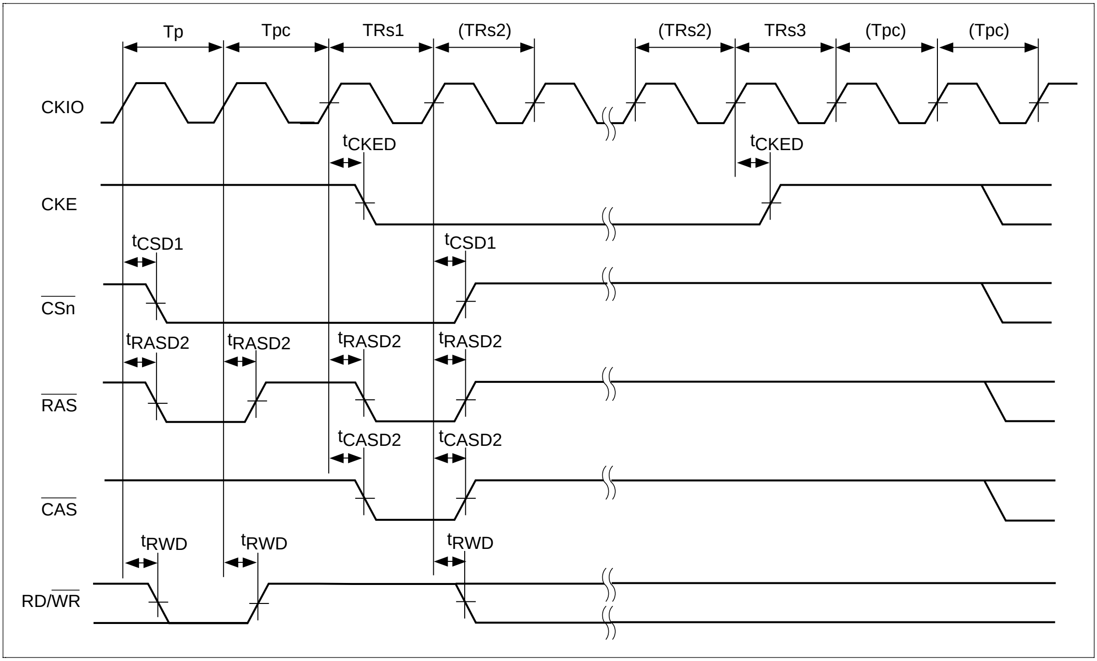


Figure 26.34 Synchronous DRAM Self-Refresh Cycle (TPC = 0, TPC = 1)

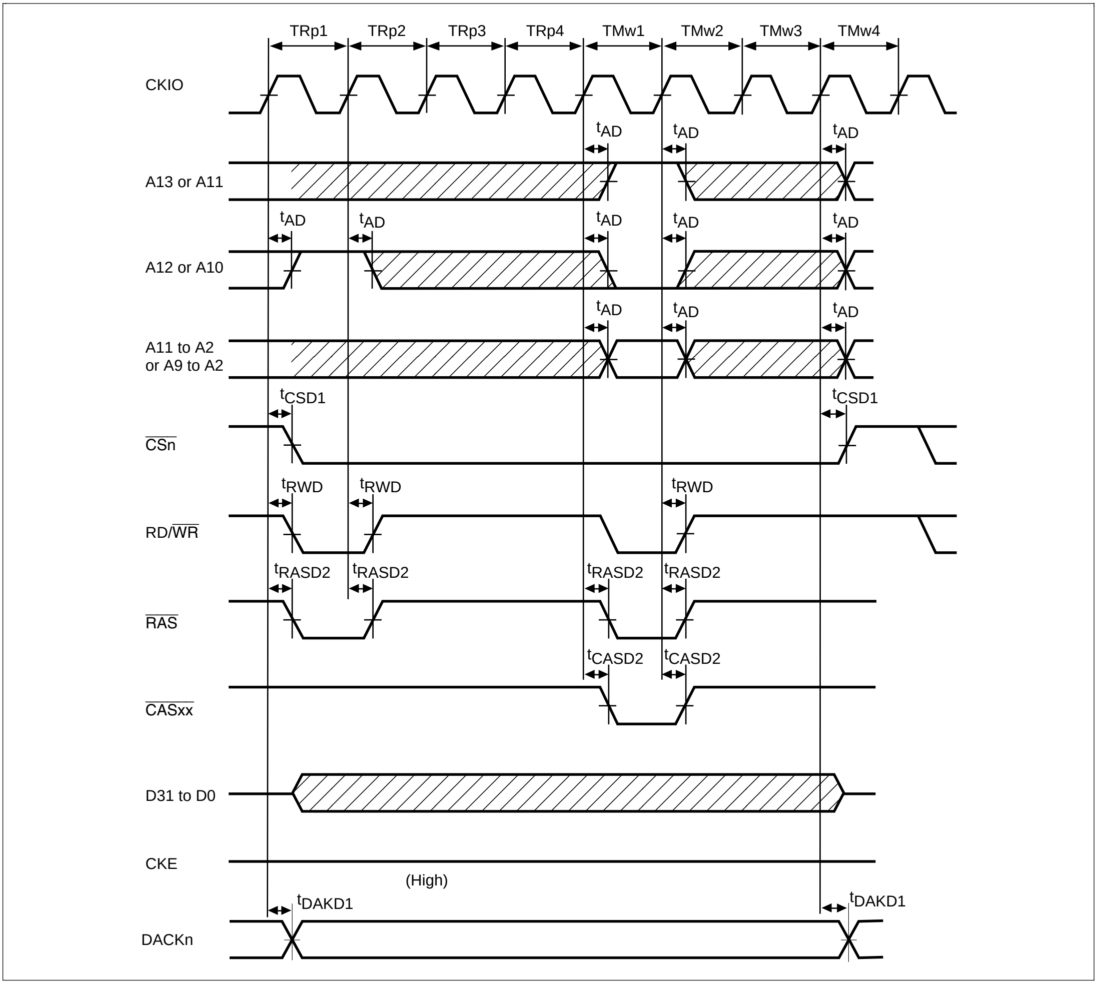


Figure 26.35 Synchronous DRAM Mode Register Write Cycle

26.3.7 Peripheral Module Signal Timing

Table 26.8 Peripheral Module Signal Timing
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Module	Item		Symbol	Min	Max	Unit	Figure
TMU	Timer input setup time		t_{TCLKS}	15	—	ns	26.36
	Timer clock input setup time		t_{TCKS}	15	—		26.37
	Timer clock pulse width	Edge specification	t_{TCKWH}	1.5	—	t_{cyc}	
		Both edge specification	t_{TCKWL}	2.5	—		
SCIF0	Input clock cycle	Clock synchronization	t_{Scyc}	12	—	P_{cyc}	26.38 26.39
	Input clock rise time		t_{SCKr}	—	1.5		26.38
	Input clock fall time		t_{SCKf}	—	1.5		
	Input clock pulse width		t_{SCKW}	0.4	0.6	tscyc	
	Transmission data delay time		t_{TXD}	—	3pcyc* + 50	ns	26.39
	Receive data setup time (clock synchronization)		t_{RXS}	2pcyc*	—		
	Receive data hold time (clock synchronization)		t_{RXH}	2pcyc*	—		
SCIF1	Input clock cycle	Clock synchronization	t_{Scyc}	12	—	t_{cyc}	26.38 26.39
	Input clock rise time		t_{SCKr}	—	1.5		26.38
	Input clock fall time		t_{SCKf}	—	1.5		
	Input clock pulse width		t_{SCKW}	0.4	0.6	tscyc	
	Transmission data delay time		t_{TXD}	—	3pcyc* + 50	ns	26.39
	Receive data setup time (clock synchronization)		t_{RXS}	2pcyc*	—		
	Receive data hold time (clock synchronization)		t_{RXH}	2pcyc*	—		

Note: * pcyc indicates a P clock cycle.

Table 26.8 Peripheral Module Signal Timing (cont) ($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Module	Item		Symbol	Min	Max	Unit	Figure
SCIF2	Input clock cycle	Asynchronization	t_{Scyc}	4	—	t_{cyc}	26.38 26.39
		Clock synchronization		12	—		
	Input clock rise time		t_{SCKr}	—	1.5		26.38
	Input clock fall time		t_{SCKf}	—	1.5		
	Input clock pulse width		t_{SCKW}	0.4	0.6	tscyc	
	Transmission data delay time		t_{TXD}	—	3pcyc* + 50	ns	26.39
	Receive data setup time (clock synchronization)		t_{RXS}	2pcyc*	—		
	Receive data hold time (clock synchronization)		t_{RXH}	2pcyc*	—		
Port	Output data delay time		t_{PORTD}	—	17	ns	26.40
	Input data setup time		t_{PORTS1}	15	—		
	Input data hold time		t_{PORTH1}	8	—		
	Input data setup time		t_{PORTS2}	17	—		
	Input data hold time		t_{PORTH2}	10	—		
DMAC	$\overline{\text{DREQ}}$ setup time		t_{DREQ}	12	—	ns	26.41
	$\overline{\text{DREQ}}$ hold time		t_{DREQH}	8	—		
	DRAK delay time		t_{DRAKD}	—	14		26.42

Note: * pcyc indicates a P clock cycle.

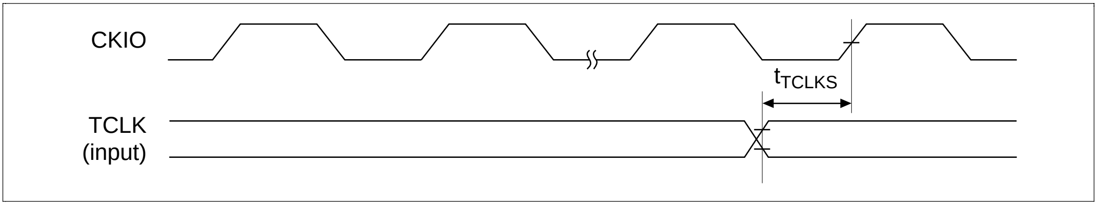


Figure 26.36 TCLK Input Timing

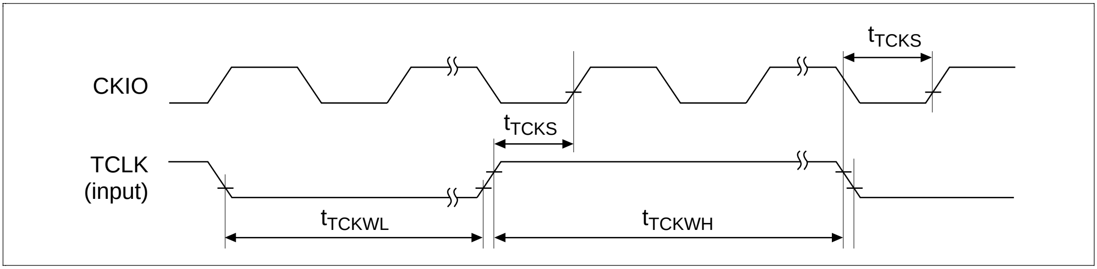


Figure 26.37 TCLK Clock Input Timing

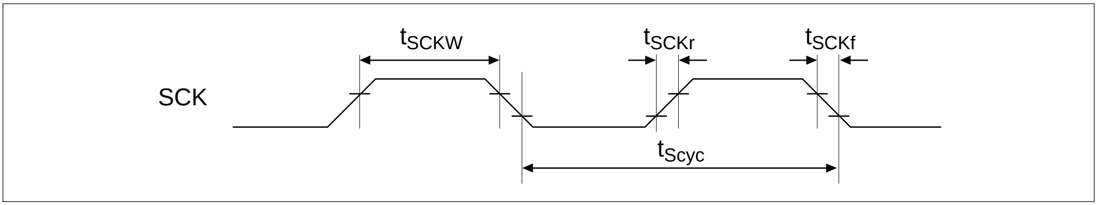


Figure 26.38 SCK Clock Input Timing

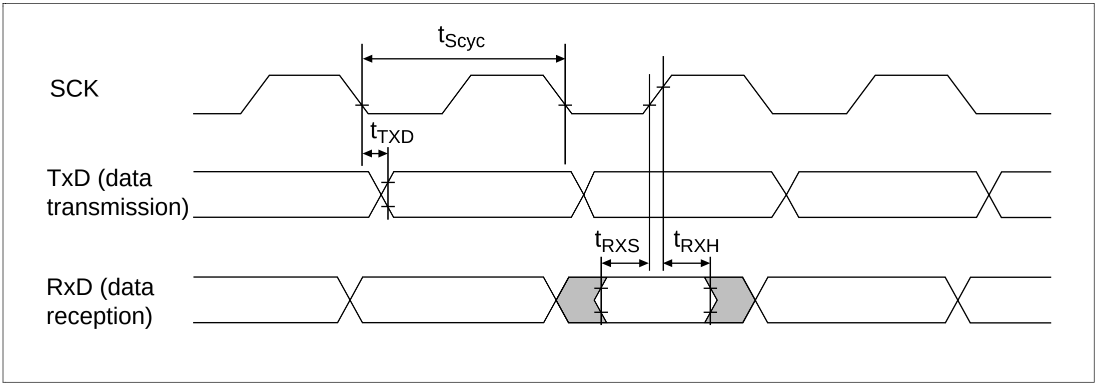


Figure 26.39 SCIF I/O Timing in Clock Synchronous Mode

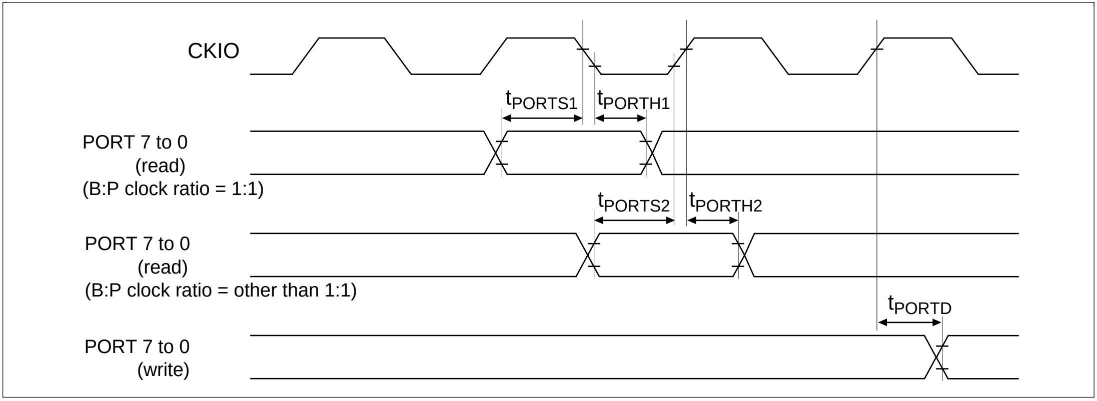


Figure 26.40 I/O Port Timing

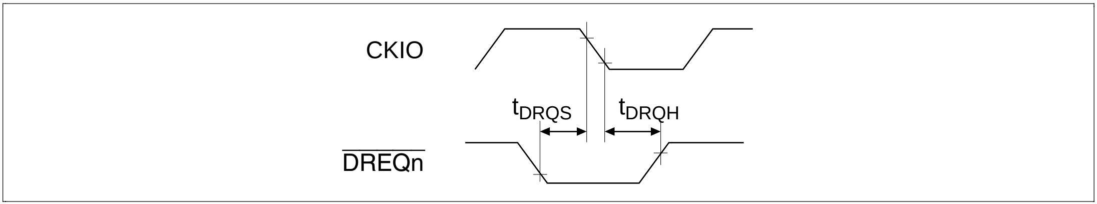


Figure 26.41 $\overline{\text{DREQ}}$ Input Timing

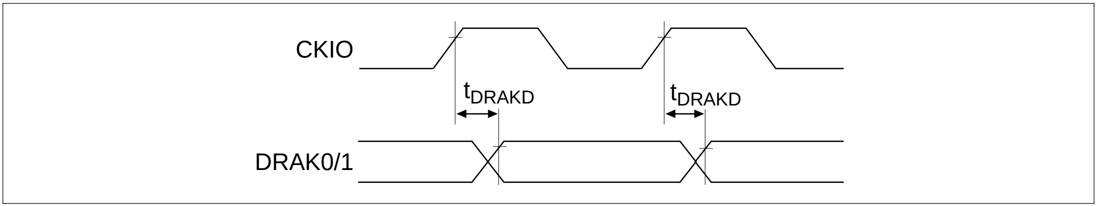


Figure 26.42 DRAK Output Timing

26.3.8 USB Module Signal Timing

Table 26.9 USB Module Signal Timing
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Item	Symbol	Min	Max	Unit	Figure
Frequency (48 MHz)	t_{FREQ}	47.9	48.1	MHz	26.43
Clock rise time	t_{R48}	—	2	ns	
Clock fall time	t_{F48}	—	2	ns	
Duty (t_{HIGH}/t_{LOW})	t_{DUTY}	90	110	%	

Note: When the USB is operated by supplying a clock to the UCLK pin from off-chip, and EXCPG is not used, the supplied clock must satisfy the above clock specifications.

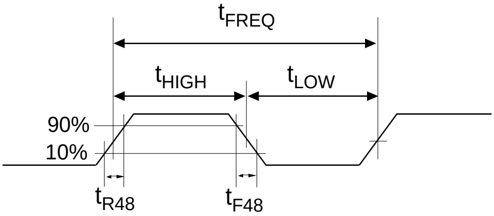


Figure 26.43 USB Clock Timing

Table 26.10 USB Module Pin Input/Output Timing
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Item	Symbol	Min	Max	Unit	Figure
Input data rise time	t_{RI}	—	4	ns	26.44
Input data fall time	t_{FI}	—	4	ns	
Rise/fall time matching (t_{RI}/t_{FI})	t_{RFMI}	90	110	%	
Output data rise time	t_{RO}	4	10	ns	
Output data fall time	t_{FO}	4	10	ns	
Rise/fall time matching (t_{RO}/t_{FO})	t_{RFMO}	90	110	%	

Note: The USB module in the SH7622 must be used together with a USB transceiver as a set. Transceivers that can be used are the Philips PDIUSBP11 Series or compatible models. See section 19, USB Function Module, for details of pin connections.

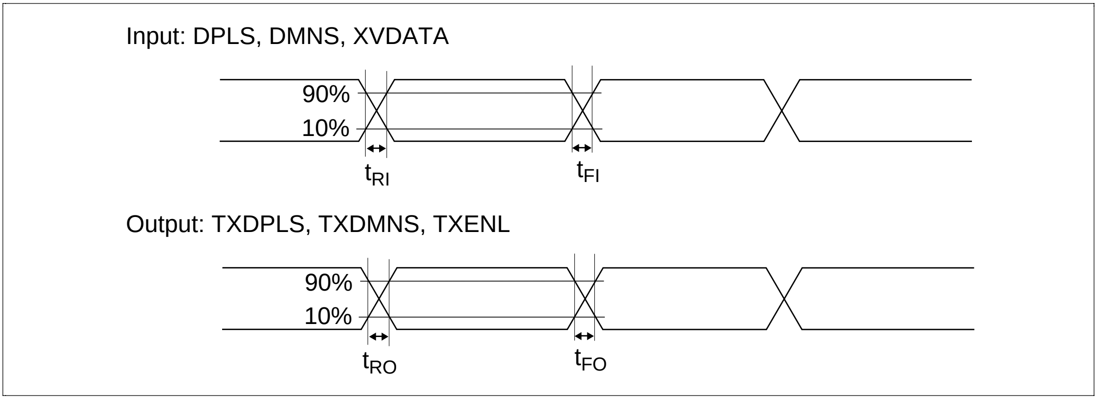


Figure 26.44 USB Module Pin Input/Output Timing

26.3.9 H-UDI-Related Pin Timing

Table 26.11 H-UDI-Related Pin Timing (VccQ = 3.3 ± 0.3 V, Vcc = 1.9/1.8 ± 0.15 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

Item	Symbol	Min	Max	Unit	Figure
TCK cycle time	t_{TCKcyc}	50	—	ns	Figure 26.45
TCK high pulse width	t_{TCKH}	12	—	ns	
TCK low pulse width	t_{TCKL}	12	—	ns	
TCK rise/fall time	t_{TCKf}	—	4	ns	
TRST setup time	t_{TRSTS}	12	—	ns	Figure 26.46
TRST hold time	t_{TRSTH}	50	—	t_{cyc}	
TDI setup time	t_{TDIS}	10	—	ns	Figure 26.47
TDI hold time	t_{TDIH}	10	—	ns	
TMS setup time	t_{TMSS}	10	—	ns	
TMS hold time	t_{TMSH}	10	—	ns	
TDO delay time	t_{TDOD}	—	16	ns	
$\overline{\text{ASEMD0}}$ setup time	t_{ASEMDH}	12	—	ns	Figure 26.48
$\overline{\text{ASEMD0}}$ hold time	t_{ASEMDS}	12	—	ns	

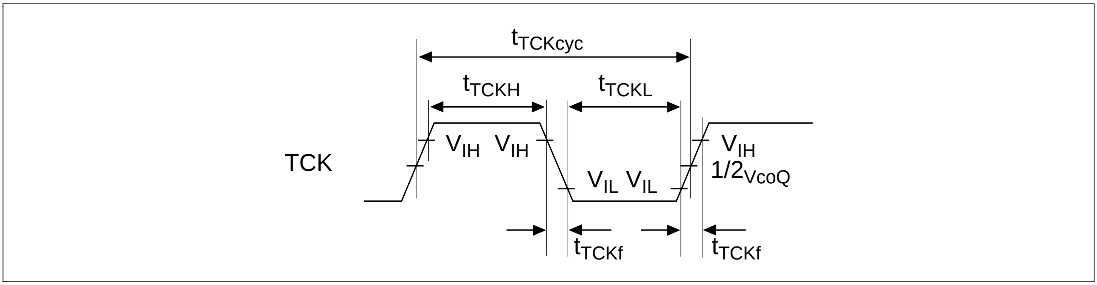


Figure 26.45 TCK Input Timing

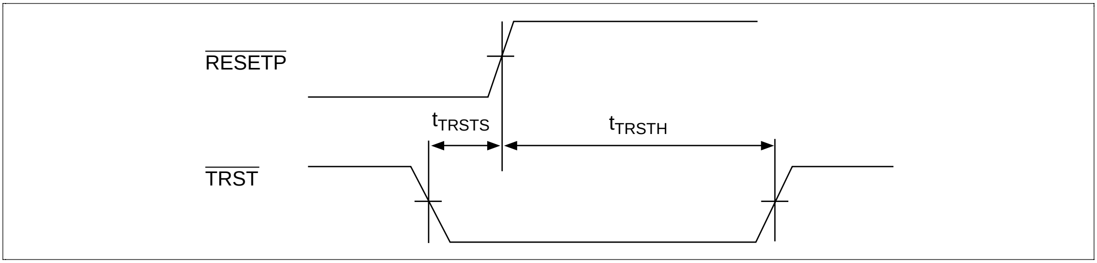


Figure 26.46 $\overline{\text{TRST}}$ Input Timing (Reset Hold)

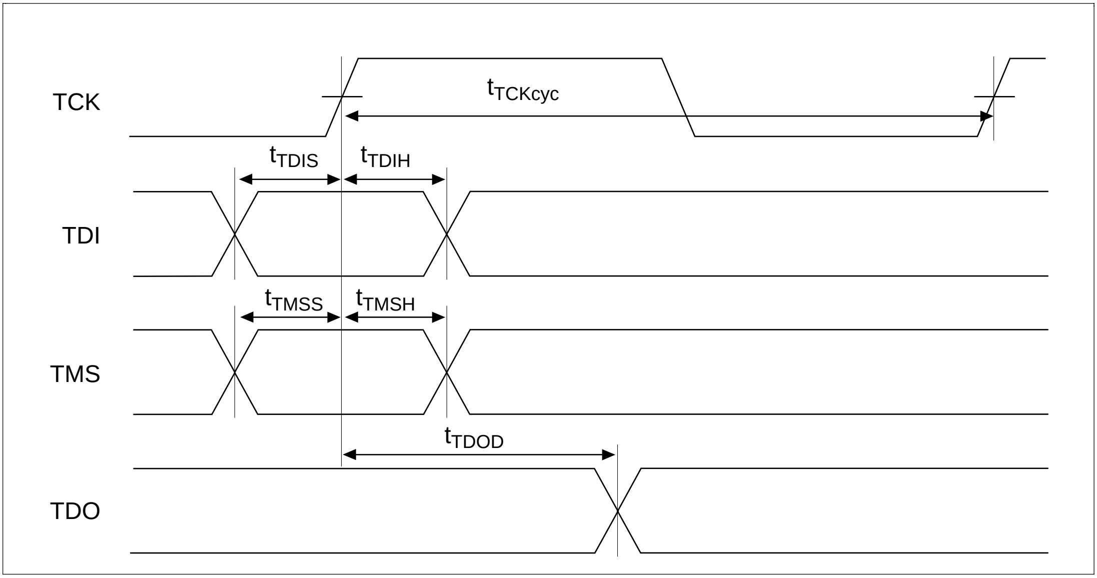


Figure 26.47 H-UDI Data Transfer Timing

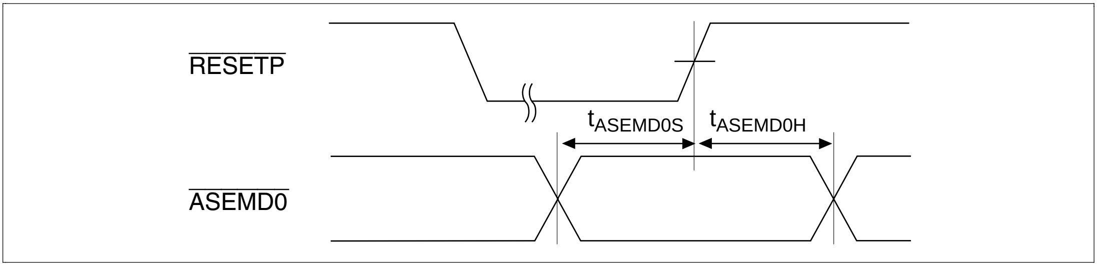


Figure 26.48 $\overline{\text{ASEMD0}}$ Input Timing

26.3.10 A/D Converter Timing

Table 26.12 A/D Converter Timing
(VccQ = 3.3 ± 0.3 V, Vcc = 1.9/1.8 ± 0.15 V, AVcc = 3.3 ± 0.3 V, Ta = –20 to 75°C)

Item		Symbol	Min	Typ	Max	Unit	Figure
External trigger input pulse width		t_{TRGW}	2	—	—	tcyc	26.49
External trigger input start delay time		t_{TRGS}	50	—	—	ns	
Input sampling time	(CKS = 0)	t_{SPL}	—	65	—	tcyc	26.50
	(CKS = 1)		—	32	—		
A/D conversion start delay time	(CKS = 0)	t_D	10	—	17	tcyc	
	(CKS = 1)		6	—	9		
A/D conversion time	(CKS = 0)	t_{CONV}	259	—	266	tcyc	
	(CKS = 1)		131	—	134		

tcyc: Pϕ cycle

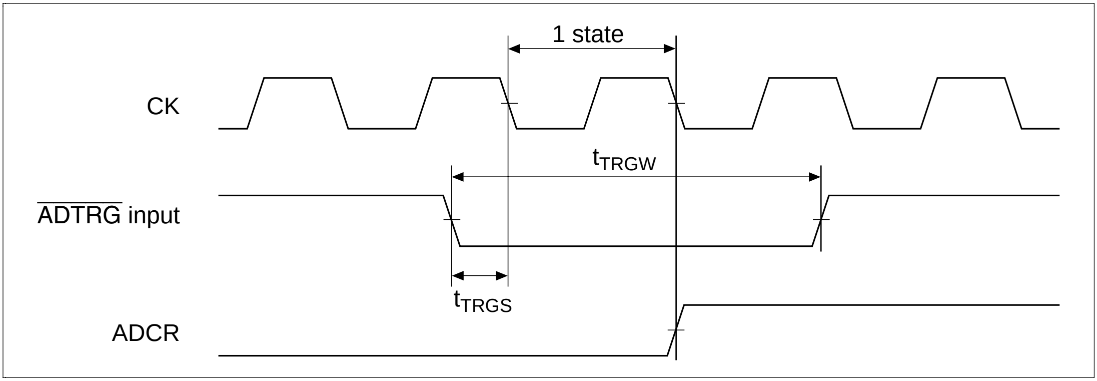
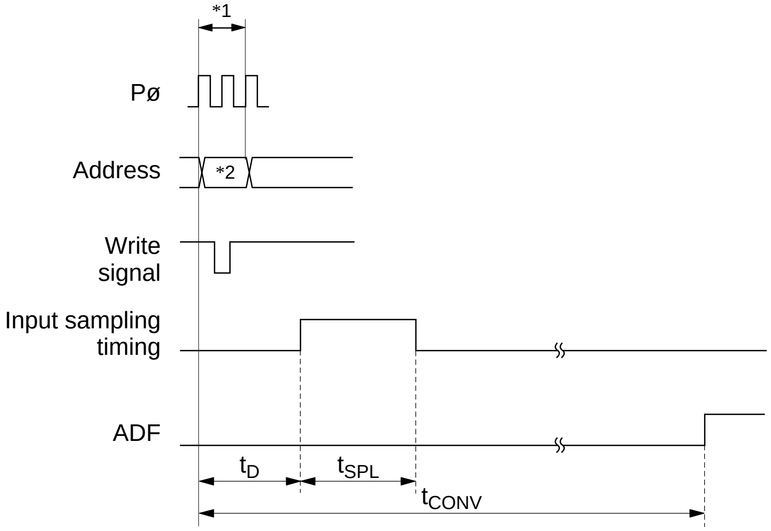


Figure 26.49 External Trigger Input Timing



t_D: A/D conversion start delay

t_{SPL}: Input sampling time

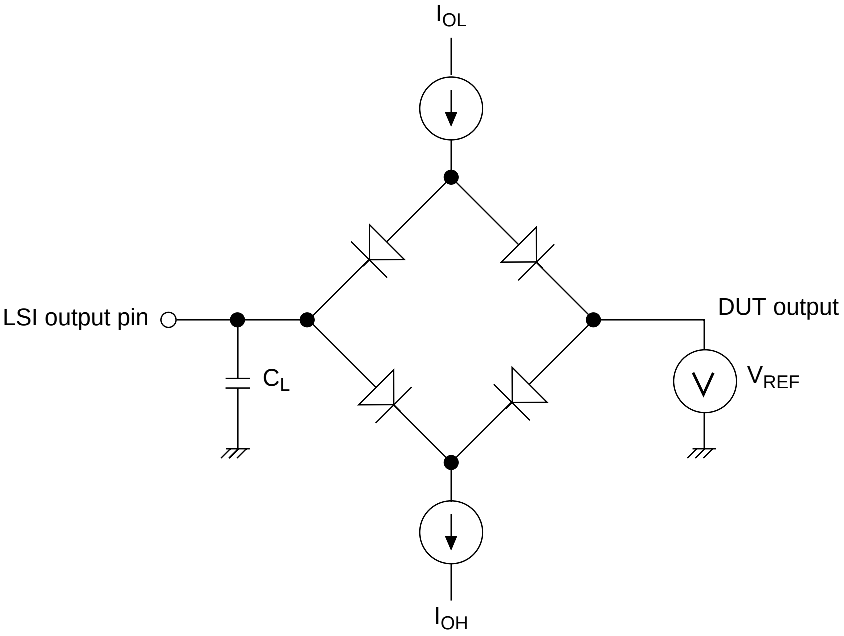
t_{CONV}: A/D conversion time

Notes: *1 ADCSR write cycle
*2 ADCSR address

Figure 26.50 A/D Conversion Timing

26.3.11 AC Characteristics Measurement Conditions

- I/O signal reference level: 1.5 V ($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V)
- Input pulse level: V_{SS} to 3.0 V (where $\overline{\text{RESETP}}$, $\overline{\text{RESETM}}$, NMI, $\overline{\text{IRQ5}}$–$\overline{\text{IRQ0}}$, CKIO, and MD4–MD0 are within V_{SS} to V_{CC})
- Input rise and fall times: 1 ns



Notes: 1. C_L is the total value that includes the capacitance of measurement instruments, etc., and is set as follows for each pin.
30 pF: CKIO, $\overline{\text{RAS}}$, $\overline{\text{CASxx}}$, $\overline{\text{CS0}}$, $\overline{\text{CS2}}$–$\overline{\text{CS6}}$, $\overline{\text{BACK}}$
50 pF: All other pins
2. I_{OL} and I_{OH} are the values shown in table 26.3.

Figure 26.51 Output Load Circuit

26.3.12 Delay Time Variation Due to Load Capacitance (Reference Values)

A graph (reference data) of the variation in delay time when a load capacitance greater than that stipulated (30 pF) is connected to the SH7622's pins is shown below. The graph shown in figure 26.52 should be taken into consideration in the design process if the stipulated capacitance is exceeded in connecting an external device.

If the connected load capacitance exceeds the range shown in figure 26.52, the graph will not be a straight line.

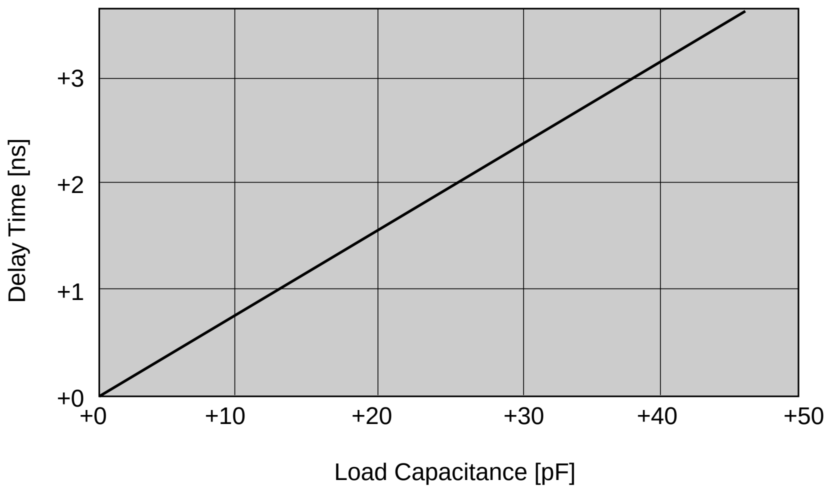


Figure 26.52 Load Capacitance vs. Delay Time

26.4 A/D Converter Characteristics

Table 26.13 lists the A/D converter characteristics.

Table 26.13 A/D Converter Characteristics
($V_{CC}Q$ = 3.0 to 3.6 V, V_{CC} = 1.75 to 2.05 V, AV_{CC} = 3.0 to 3.6 V, T_a = –20 to 75°C)

Item	Min	Typ	Max	Unit
Resolution	10	10	10	bits
Conversion time	—	—	8.9	µs
Analog input capacitance	—	—	20*	pF
Permissible signal-source (single-source) impedance	—	—	5*	kΩ
Nonlinearity error	—	—	±3.0*	LSB
Offset error	—	—	±2.0*	LSB
Full-scale error	—	—	±2.0*	LSB
Quantization error	—	—	±0.5*	LSB
Absolute accuracy	—	—	±4.0	LSB

Note: * Reference values

Appendix A On-Chip Peripheral Module Registers

A.1 Address List

		Bit Names								
Address	**Register Name**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Module**
H'A400 0200	SDIR	TI3	TI2	TI1	TI0	—	—	—	—	H-UDI
H'A400 0201		—	—	—	—	—	—	—	—	
H'A400 0202 to H'A400 001F	—	—	—	—	—	—	—	—	—	—
H'A400 0020	SAR0									DMAC
H'A400 0021										
H'A400 0022										
H'A400 0023										
H'A400 0024	DAR0									
H'A400 0025										
H'A400 0026										
H'A400 0027										
H'A400 0028	DMATCR0	—	—	—	—	—	—	—	—	
H'A400 0029										
H'A400 002A										
H'A400 002B										
H'A400 002C	CHCR0	—	—	—	—	—	—	—	—	
H'A400 002D		—	—	—	DI	RO	RL	AM	AL	
H'A400 002E		DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
H'A400 002F		—	DS	TM	TS1	TS0	IE	TE	DE	
H'A400 0030	SAR1									
H'A400 0031										
H'A400 0032										
H'A400 0033										
H'A400 0034	DAR1									
H'A400 0035										
H'A400 0036										
H'A400 0037										
H'A400 0038	DMATCR1	—	—	—	—	—	—	—	—	
H'A400 0039										
H'A400 003A										
H'A400 003B										

Address	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'A400 003C	CHCR1	—	—	—	—	—	—	—	—	DMAC
H'A400 003D		—	—	—	DI	RO	RL	AM	AL	
H'A400 003E		DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
H'A400 003F		—	DS	TM	TS1	TS0	IE	TE	DE	
H'A400 0040	SAR2									
H'A400 0041										
H'A400 0042										
H'A400 0043										
H'A400 0044	DAR2									
H'A400 0045										
H'A400 0046										
H'A400 0047										
H'A400 0048	DMATCR2	—	—	—	—	—	—	—	—	
H'A400 0049										
H'A400 004A										
H'A400 004B										
H'A400 004C	CHCR2	—	—	—	—	—	—	—	—	
H'A400 004D		—	—	—	DI	RO	RL	AM	AL	
H'A400 004E		DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
H'A400 004F		—	DS	TM	TS1	TS0	IE	TE	DE	
H'A400 0050	SAR3									
H'A400 0051										
H'A400 0052										
H'A400 0053										
H'A400 0054	DAR3									
H'A400 0055										
H'A400 0056										
H'A400 0057										
H'A400 0058	DMATCR3	—	—	—	—	—	—	—	—	
H'A400 0059										
H'A400 005A										
H'A400 005B										
H'A400 005C	CHCR3	—	—	—	—	—	—	—	—	
H'A400 005D		—	—	—	DI	RO	RL	AM	AL	
H'A400 005E		DM1	DM0	SM1	SM0	RS3	RS2	RS1	RS0	
H'A400 005F		—	DS	TM	TS1	TS0	IE	TE	DE	

Address	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'A400 0060	DMAOR	—	—	—	—	—	—	PR1	PR0	DMAC
H'A400 0061		—	—	—	—	—	AE	NMIF	DME	
H'A400 0062 to H'A400 006F	—	—	—	—	—	—	—	—	—	—
H'A400 0070	CMSTR0	—	—	—	—	—	—	—	—	CMT0
H'A400 0071		—	—	—	—	—	—	—	STR0	
H'A400 0072	CMCSR0	—	—	—	—	—	—	—	—	
H'A400 0073		CMF	—	—	—	—	—	CKS1	CKS0	
H'A400 0074	CMCNT0									
H'A400 0075										
H'A400 0076	CMCOR0									
H'A400 0077										
H'A400 0078 to H'A400 007F	—	—	—	—	—	—	—	—	—	—
H'A400 0080	ADDRAH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D converter
H'A400 0081	—	—	—	—	—	—	—	—	—	—
H'A400 0082	ADDRAL	AD1	AD0	—	—	—	—	—	—	A/D converter
H'A400 0083	—	—	—	—	—	—	—	—	—	—
H'A400 0084	ADDRBH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D converter
H'A400 0085	—	—	—	—	—	—	—	—	—	—
H'A400 0086	ADDRBL	AD1	AD0	—	—	—	—	—	—	A/D converter
H'A400 0087	—	—	—	—	—	—	—	—	—	—
H'A400 0088	ADDRCH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D converter
H'A400 0089	—	—	—	—	—	—	—	—	—	—
H'A400 008A	ADDRCL	AD1	AD0	—	—	—	—	—	—	A/D converter
H'A400 008B	—	—	—	—	—	—	—	—	—	—
H'A400 008C	ADDRDH	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	A/D converter
H'A400 008D	—	—	—	—	—	—	—	—	—	—
H'A400 008E	ADDRDL	AD1	AD0	—	—	—	—	—	—	A/D converter
H'A400 008F	—	—	—	—	—	—	—	—	—	—
H'A400 0090	ADCSR	ADF	ADIE	ADST	MULTI	CKS	—	CH1	CH0	A/D converter

Address	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'A400 0091	—	—	—	—	—	—	—	—	—	—
H'A400 0092	ADCR	TRGE1	TRGE0	SCN	RESVD1	RESVD2	—	—	—	A/D converter
H'A400 0093 to H'A400 009F	—	—	—	—	—	—	—	—	—	—
H'A400 0100	PACR	PA7MD1	PA7MD0	PA6MD1	PA6MD0	PA5MD1	PA5MD0	PA4MD1	PA4MD0	PFC
H'A400 0101		PA3MD1	PA3MD0	PA2MD1	PA2MD0	PA1MD1	PA1MD0	PA0MD1	PA0MD0	
H'A400 0102	PBCR	PB7MD1	PB7MD0	PB6MD1	PB6MD0	PB5MD1	PB5MD0	PB4MD1	PB4MD0	
H'A400 0103		PB3MD1	PB3MD0	PB2MD1	PB2MD0	PB1MD1	PB1MD0	PB0MD1	PB0MD0	
H'A400 0104	PCCR	PC7MD1	PC7MD0	PC6MD1	PC6MD0	PC5MD1	PC5MD0	PC4MD1	PC4MD0	
H'A400 0105		PC3MD1	PC3MD0	PC2MD1	PC2MD0	PC1MD1	PC1MD0	PC0MD1	PC0MD0	
H'A400 0106	PDCR	PD7MD1	PD7MD0	PD6MD1	PD6MD0	PD5MD1	PD5MD0	PD4MD1	PD4MD0	
H'A400 0107		PD3MD1	PD3MD0	PD2MD1	PD2MD0	PD1MD1	PD1MD0	PD0MD1	PD0MD0	
H'A400 0108	PECR	PE7MD1	PE7MD0	PE6MD1	PE6MD0	PE5MD1	PE5MD0	PE4MD1	PE4MD0	
H'A400 0109		PE3MD1	PE3MD0	PE2MD1	PE2MD0	PE1MD1	PE1MD0	PE0MD1	PE0MD0	
H'A400 010A	PFCR	PF7MD1	PF7MD0	PF6MD1	PF6MD0	PF5MD1	PF5MD0	PF4MD1	PF4MD0	
H'A400 010B		PF3MD1	PF3MD0	PF2MD1	PF2MD0	PF1MD1	PF1MD0	PF0MD1	PF0MD0	
H'A400 010C	PGCR	PG7MD1	PG7MD0	PG6MD1	PG6MD0	PG5MD1	PG5MD0	PG4MD1	PG4MD0	
H'A400 010D		PG3MD1	PG3MD0	PG2MD1	PG2MD0	PG1MD1	PG1MD0	PG0MD1	PG0MD0	
H'A400 010E	PHCR	PH7MD1	PH7MD0	PH6MD1	PH6MD0	PH5MD1	PH5MD0	PH4MD1	PH4MD0	
H'A400 010F		PH3MD1	PH3MD0	PH2MD1	PH2MD0	PH1MD1	PH1MD0	PH0MD1	PH0MD0	
H'A400 0110	PJCR	PJ7MD1	PJ7MD0	PJ6MD1	PJ6MD0	PJ5MD1	PJ5MD0	PJ4MD1	PJ4MD0	
H'A400 0111		PJ3MD1	PJ3MD0	PJ2MD1	PJ2MD0	PJ1MD1	PJ1MD0	PJ0MD1	PJ0MD0	
H'A400 0112	PKCR	PK7MD1	PK7MD0	PK6MD1	PK6MD0	PK5MD1	PK5MD0	PK4MD1	PK4MD0	
H'A400 0113		PK3MD1	PK3MD0	PK2MD1	PK2MD0	PK1MD1	PK1MD0	PK0MD1	PK0MD0	
H'A400 0114	PLCR	PL7MD1	PL7MD0	PL6MD1	PL6MD0	PL5MD1	PL5MD0	PL4MD1	PL4MD0	
H'A400 0115		PL3MD1	PL3MD0	PL2MD1	PL2MD0	PL1MD1	PL1MD0	PL0MD1	PL0MD0	
H'A400 0116	SCPCR	SCP7MD1	SCP7MD0	SCP6MD1	SCP6MD0	SCP5MD1	SCP5MD0	SCP4MD1	SCP4MD0	
H'A400 0117		SCP3MD1	SCP3MD0	SCP2MD1	SCP2MD0	SCP1MD1	SCP1MD0	SCP0MD1	SCP0MD0	
H'A400 0118 to H'A400 011F	—	—	—	—	—	—	—	—	—	—
H'A400 0120	PADR	PA7DT	PA6DT	PA5DT	PA4DT	PA3DT	PA2DT	PA1DT	PA0DT	I/O port
H'A400 0121	—	—	—	—	—	—	—	—	—	—
H'A400 0122	PBDR	PB7DT	PB6DT	PB5DT	PB4DT	PB3DT	PB2DT	PB1DT	PB0DT	I/O port
H'A400 0123	—	—	—	—	—	—	—	—	—	—
H'A400 0124	PCDR	PC7DT	PC6DT	PC5DT	PC4DT	PC3DT	PC2DT	PC1DT	PC0DT	I/O port
H'A400 0125	—	—	—	—	—	—	—	—	—	—
H'A400 0126	PDDR	PD7DT	PD6DT	PD5DT	PD4DT	PD3DT	PD2DT	PD1DT	PD0DT	I/O port

Address	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'A400 0127	—	—	—	—	—	—	—	—	—	—
H'A400 0128	PEDR	PE7DT	PE6DT	PE5DT	PE4DT	PE3DT	PE2DT	PE1DT	PE0DT	I/O port
H'A400 0129	—	—	—	—	—	—	—	—	—	—
H'A400 012A	PFDR	PF7DT	PF6DT	PF5DT	PF4DT	PF3DT	PF2DT	PF1DT	PF0DT	I/O port
H'A400 012B	—	—	—	—	—	—	—	—	—	—
H'A400 012C	PGDR	PG7DT	PG6DT	PG5DT	PG4DT	PG3DT	PG2DT	PG1DT	PG0DT	I/O port
H'A400 012D	—	—	—	—	—	—	—	—	—	—
H'A400 012E	PHDR	PH7DT	PH6DT	PH5DT	PH4DT	PH3DT	PH2DT	PH1DT	PH0DT	I/O port
H'A400 012F	—	—	—	—	—	—	—	—	—	—
H'A400 0130	PJDR	PJ7DT	PJ6DT	PJ5DT	PJ4DT	PJ3DT	PJ2DT	PJ1DT	PJ0DT	I/O port
H'A400 0131	—	—	—	—	—	—	—	—	—	—
H'A400 0132	PKDR	PK7DT	PK6DT	PK5DT	PK4DT	PK3DT	PK2DT	PK1DT	PK0DT	I/O port
H'A400 0133	—	—	—	—	—	—	—	—	—	—
H'A400 0134	PLDR	PL7DT	PL6DT	PL5DT	PL4DT	PL3DT	PL2DT	PL1DT	PL0DT	I/O port
H'A400 0135	—	—	—	—	—	—	—	—	—	—
H'A400 0136	SCPDR	SCP7DT	SCP6DT	SCP5DT	SCP4DT	SCP3DT	SCP2DT	SCP1DT	SCP0DT	I/O port
H'A400 0137 to H'A400 019F	—	—	—	—	—	—	—	—	—	—
H'A400 0200	SDIR	TI3	TI2	TI1	TI0	—	—	—	—	H-UDI
H'A400 0201		—	—	—	—	—	—	—	—	
H'A400 0202 to H'A400 089F	—	—	—	—	—	—	—	—	—	—
H'A400 0900	CHCRA0	MID	MID	MID	MID	MID	MID	RID	RID	DMAC
H'A400 0901		MID	MID	MID	MID	MID	MID	RID	RID	
H'A400 0902	CHCRA1	MID	MID	MID	MID	MID	MID	RID	RID	
H'A400 0903		MID	MID	MID	MID	MID	MID	RID	RID	
H'A400 0904 to H'A400 0A0F	—	—	—	—	—	—	—	—	—	—
H'A400 0A10	STBCR3	—	—	MSTPE	MSTPD	—	MSTPB	MSTPA	MSTP9	Power-down mode
H'A400 0A11 to H'A400 0A1F	—	—	—	—	—	—	—	—	—	—
H'A400 0A20	USBCLKCR	USSCS1	USSCS0	—	—	—	—	—	USDIVS0	EXCPG
H'A400 0A21 to H'A400 199F	—	—	—	—	—	—	—	—	—	—

Address	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'A400 2000	SCSMR0	—	—	—	—	—	—	CKS1	CKS0	SCIF0
H'A400 2001	—	—	—	—	—	—	—	—	—	—
H'A400 2002	SCBRR0									SCIF0
H'A400 2003	—	—	—	—	—	—	—	—	—	—
H'A400 2004	SCSCR0	TIE	RIE	TE	RE	—	—	CKE1	CKE0	SCIF0
H'A400 2005	—	—	—	—	—	—	—	—	—	—
H'A400 2006	SCLSR0	—	FST6	FST5	FST4	FST3	FST2	FST1	FST0	SCIF0
H'A400 2007		—	—	—	FSTE	—	—	—	ORER	
H'A400 2008	SCSSR0	—	—	—	—	—	—	—	—	
H'A400 2009		—	TEND	TDFST	—	—	—	RDFST	—	
H'A400 200A	SCRFDR0	—	—	—	—	—	—	—	R8	
H'A400 200B		R7	R6	R5	R4	R3	R2	R1	R0	
H'A400 200C	SCFCR0	—	—	—	—	—	TFRST	RFRST	—	SCIF0
H'A400 200D	—	—	—	—	—	—	—	—	—	—
H'A400 200E	—	—	—	—	—	—	—	—	—	
H'A400 200F	—	—	—	—	—	—	—	—	—	
H'A400 2010	SCFTDR0									SCIF0
H'A400 2011	—	—	—	—	—	—	—	—	—	—
H'A400 2012	—	—	—	—	—	—	—	—	—	
H'A400 2013	—	—	—	—	—	—	—	—	—	
H'A400 2014	SCFRDR0									SCIF0
H'A400 2015	—	—	—	—	—	—	—	—	—	—
H'A400 2016	—	—	—	—	—	—	—	—	—	—
H'A400 2017	—	—	—	—	—	—	—	—	—	
H'A400 2018	SCTFDR0	—	—	—	—	—	—	—	—	SCIF0
H'A400 2019		T7	T6	T5	T4	T3	T2	T1	T0	
H'A400 201A to H'A400 201F	—	—	—	—	—	—	—	—	—	—
H'A400 2020	SCSMR1	—	—	—	—	—	—	CKS1	CKS0	SCIF1
H'A400 2021	—	—	—	—	—	—	—	—	—	—
H'A400 2022	SCBRR1									SCIF1
H'A400 2023	—	—	—	—	—	—	—	—	—	—
H'A400 2024	SCSCR1	TIE	RIE	TE	RE	—	—	CKE1	CKE0	SCIF1
H'A400 2025	—	—	—	—	—	—	—	—	—	—
H'A400 2026	SCLSR1	—	FST6	FST5	FST4	FST3	FST2	FST1	FST0	SCIF1
H'A400 2027		—	—	—	FSTE	—	—	—	ORER	

Address	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'A400 2028	SCSSR1	—	—	—	—	—	—	—	—	SCIF1
H'A400 2029		—	TEND	TDFST	—	—	—	RDFST	—	
H'A400 202A	SCFCR1	—	—	—	—	—	TFRST	RFRST	—	
H'A400 202B	—	—	—	—	—	—	—	—	—	—
H'A400 202C	SCFDR1	T7	T6	T5	T4	T3	T2	T1	T0	SCIF1
H'A400 202D		R7	R6	R5	R4	R3	R2	R1	R0	
H'A400 202E	—	—	—	—	—	—	—	—	—	—
H'A400 202F	—	—	—	—	—	—	—	—	—	
H'A400 2030	SCFTDR1									SCIF1
H'A400 2031	—	—	—	—	—	—	—	—	—	—
H'A400 2032	—	—	—	—	—	—	—	—	—	
H'A400 2033	—	—	—	—	—	—	—	—	—	
H'A400 2034	SCFRDR1									SCIF1
H'A400 2035 to H'A400 203F	—	—	—	—	—	—	—	—	—	—
H'A400 2040	SCSMR2	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	—	CKS1	CKS0	SCIF2
H'A400 2041	—	—	—	—	—	—	—	—	—	—
H'A400 2042	SCBRR2									SCIF2
H'A400 2043	—	—	—	—	—	—	—	—	—	—
H'A400 2044	SCSCR2	TIE	RIE	TE	RE	—	—	CKE1	CKE0	SCIF2
H'A400 2045	—	—	—	—	—	—	—	—	—	—
H'A400 2046	SCFTDR2									SCIF2
H'A400 2047	—	—	—	—	—	—	—	—	—	—
H'A400 2048	SCSSR2	PER3	PER2	PER1	PER0	FER3	FER2	FER1	FER0	SCIF2
H'A400 2049		ER	TEND	TDFE	BRK	FER	PER	RDF	DR	
H'A400 204A	SCFRDR2									
H'A400 204B	—	—	—	—	—	—	—	—	—	—
H'A400 204C	SCFCR2	RTRG1	RTRG0	TTRG1	TTRG0	—	TFRST	RFRST	LOOP	SCIF2
H'A400 204D	—	—	—	—	—	—	—	—	—	—
H'A400 204E	SCFDR2	—	—	—	T4	T3	T2	T1	T0	SCIF2
H'A400 204F		—	—	—	R4	R3	R2	R1	R0	
H'A400 2050 to H'A400 206F	—	—	—	—	—	—	—	—	—	—

	Register	Bit Names								
Address	**Name**	**Bit 7**	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**	**Module**
H'A400 2070	CMSTR1	—	—	—	—	—	—	—	—	CMT1
H'A400 2071		—	—	—	—	—	—	—	STR	
H'A400 2072	CMCSR1	—	—	—	—	—	—	—	—	
H'A400 2073		CMF	—	CMR1	CMR0	—	—	CKS1	CKS0	
H'A400 2074	CMCNT1									
H'A400 2075										
H'A400 2076	CMCOR1									
H'A400 2077										
H'A400 2078 to H'A400 207F	—	—	—	—	—	—	—	—	—	—
H'A400 2080	IPRA									INTC
H'A400 2081										
H'A400 2082	IPRB									
H'A400 2083										
H'A400 2084	IPRC									
H'A400 2085										
H'A400 2086	IPRD									
H'A400 2087										
H'A400 2088	IPRE									
H'A400 2089										
H'A400 208A	—	—	—	—	—	—	—	—	—	
H'A400 208B										
H'A400 208C	IPRG									
H'A400 208D										
H'A400 208E	IPRH									
H'A400 208F										
H'A400 2090	ICR0	NMIL	—	—	—	—	—	—	NMIE	
H'A400 2091		—	—	—	—	—	—	IRQE	—	
H'A400 2092	ICR1	IRQ71S	IRQ70S	IRQ61S	IRQ60S	IRQ51S	IRQ50S	IRQ41S	IRQ40S	INTC
H'A400 2093		IRQ31S	IRQ30S	IRQ21S	IRQ20S	IRQ11S	IRQ10S	IRQ01S	IRQ00S	
H'A400 2094	IRR	—	—	—	—	—	—	—	—	
H'A400 2095		IRQ7R	IRQ6R	IRQ5R	IRQ4R	IRQ3R	IRQ2R	IRQ1R	IRQ0R	
H'A400 2096 to H'A400 799F	—	—	—	—	—	—	—	—	—	—
H'A400 8000	USBEPDR0I									USB

		Bit Names								
Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
H'A400 8001	—	—	—	—	—	—	—	—	—	—
H'A400 8002	—	—	—	—	—	—	—	—	—	
H'A400 8003	—	—	—	—	—	—	—	—	—	
H'A400 8004	USBEPDR0O									USB
H'A400 8005	—	—	—	—	—	—	—	—	—	—
H'A400 8006	—	—	—	—	—	—	—	—	—	
H'A400 8007	—	—	—	—	—	—	—	—	—	
H'A400 8008	USBEPDR0S									USB
H'A400 8009	—	—	—	—	—	—	—	—	—	—
H'A400 800A	—	—	—	—	—	—	—	—	—	
H'A400 800B	—	—	—	—	—	—	—	—	—	
H'A400 800C	USBEPDR1									USB
H'A400 800D	—	—	—	—	—	—	—	—	—	—
H'A400 800E	—	—	—	—	—	—	—	—	—	
H'A400 800F	—	—	—	—	—	—	—	—	—	
H'A400 8010	USBEPDR2									USB
H'A400 8011	—	—	—	—	—	—	—	—	—	—
H'A400 8012	—	—	—	—	—	—	—	—	—	
H'A400 8013	—	—	—	—	—	—	—	—	—	
H'A400 8014	USBEPDR3									USB
H'A400 8015	—	—	—	—	—	—	—	—	—	—
H'A400 8016	—	—	—	—	—	—	—	—	—	
H'A400 8017	—	—	—	—	—	—	—	—	—	
H'A400 8018	USBIFR0	BRST	EP1 FULL	EP2 TR	EP2 EMPTY	SETUP TS	EP0o TS	EP0i TR	EI0i TS	USB
H'A400 8019	—	—	—	—	—	—	—	—	—	—
H'A400 801A	USBIFR1	—	—	—	—	—	EP3 TR	EP3 TS	VBUS	USB
H'A400 801B	—	—	—	—	—	—	—	—	—	—
H'A400 801C	USBTRG	—	EP3 PKTE	EP1 RDFN	EP2 PKTE	—	PE0s RDFN	EP0o RDFN	EP0i PKTE	USB
H'A400 801D	—	—	—	—	—	—	—	—	—	—
H'A400 801E	USBFCLR	—	EP3 CLR	EP1 CLR	EP2 CLR	—	—	EP0o CLR	EP0i CLR	USB
H'A400 801F	—	—	—	—	—	—	—	—	—	—
H'A400 8020	USBEPSZ0O									USB
H'A400 8021	—	—	—	—	—	—	—	—	—	—
H'A400 8022	USBDASTS	—	—	EP3 DE	EP2 DE	—	—	—	EP0i DE	USB
H'A400 8023	—	—	—	—	—	—	—	—	—	—
H'A400 8024	USBEPSTL	—	—	—	—	EP3 STL	EP2 STL	EP1 STL	EP0 STL	USB

	Register	Bit Names								
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Module
H'A400 8025	—	—	—	—	—	—	—	—	—	—
H'A400 8026	USBIER0	BRST	EP1 FULL	EP2 TR	EP2 EMPTY	SETUP TS	EP0o TS	EP0i TR	EP0i TS	USB
H'A400 8027	—	—	—	—	—	—	—	—	—	—
H'A400 8028	USBIER1	—	—	—	—	—	EP3 TR	EP3 TS	VBUS	USB
H'A400 8029	—	—	—	—	—	—	—	—	—	—
H'A400 802A	USBEPSZ1									USB
H'A400 802B	—	—	—	—	—	—	—	—	—	—
H'A400 802C	USBDMAR									USB
H'A400 802D	—	—	—	—	—	—	—	—	—	—
H'A400 802E	USBISR0	BRST	EP1 FULL	EP2 TR	EP2 EMPTY	SETUP TS	EP0o TS	EP0i TR	EP0i TS	USB
H'A400 802F	—	—	—	—	—	—	—	—	—	—
H'A400 8030	USBISR1	—	—	—	—	—	EP3 TR	EP3 TS	VBUS	USB
H'A400 8031 to H'FFFF CFFF	—	—	—	—	—	—	—	—	—	—
H'FFFF D000 to H'FFFF DFFF	SDMR (area 2)									BSC
H'FFFF E000 to H'FFFF EFFF	SDMR (area 3)									
H'FFFF F000 to H'FFFF FE91	—	—	—	—	—	—	—	—	—	—
H'FFFF FE92	TSTR	—	—	—	—	—	STR2	STR1	STR0	TMU
H'FFFF FE93	—	—	—	—	—	—	—	—	—	—
H'FFFF FE94	TCOR0									TMU
H'FFFF FE95										
H'FFFF FE96										
H'FFFF FE97										
H'FFFF FE98	TCNT0									TMU
H'FFFF FE99										
H'FFFF FE9A										
H'FFFF FE9B										
H'FFFF FE9C	TCR0	—	—	—	—	—	—	—	UNF	
H'FFFF FE9D		—	—	UNIE	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	
H'FFFF FE9E	—	—	—	—	—	—	—	—	—	—
H'FFFF FE9F	—	—	—	—	—	—	—	—	—	

Address	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'FFFF FEA0	TCOR1									TMU
H'FFFF FEA1										
H'FFFF FEA2										
H'FFFF FEA3										
H'FFFF FEA4	TCNT1									
H'FFFF FEA5										
H'FFFF FEA6										
H'FFFF FEA7										
H'FFFF FEA8	TCR1	—	—	—	—	—	—	—	UNF	
H'FFFF FEA9		—	—	UNIE	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	
H'FFFF FEAA	—	—	—	—	—	—	—	—	—	—
H'FFFF FEAB	—	—	—	—	—	—	—	—	—	
H'FFFF FEAC	TCOR2									TMU
H'FFFF FEAD										
H'FFFF FEAE										
H'FFFF FEAF										
H'FFFF FEB0	TCNT2									
H'FFFF FEB1										
H'FFFF FEB2										
H'FFFF FEB3										
H'FFFF FEB4	TCR2	—	—	—	—	—	—	ICPF	UNF	
H'FFFF FEB5		ICPE1	ICPE0	UNIE	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	
H'FFFF FEB6	—	—	—	—	—	—	—	—	—	—
H'FFFF FEB7	—	—	—	—	—	—	—	—	—	
H'FFFF FEB8	TCPR2									TMU
H'FFFF FEB9										
H'FFFF FEBA										
H'FFFF FEBB										
H'FFFF FEBC to H'FFFF FF5F	—	—	—	—	—	—	—	—	—	—

Address	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'FFFF FF60	BCR1	—	—	HIZMEM	HIZCNT	—	A0BST1	A0BST0	A5BST1	BSC
H'FFFF FF61		A5BST0	A6BST1	A6BST0	—	DRAMTP1	DRAMTP0	—	—	
H'FFFF FF62	BCR2	—	—	A6SZ1	A6SZ0	A5SZ1	A5SZ0	A4SZ1	A4SZ0	
H'FFFF FF63		A3SZ1	A3SZ0	A2SZ1	A2SZ0	—	—	—	—	
H'FFFF FF64	WCR1	WAITSEL	—	A6IW1	A6IW0	A5IW1	A5IW0	A4IW1	A4IW0	
H'FFFF FF65		A3IW1	A3IW0	A2IW1	A2IW0	—	—	A0IW1	A0IW0	
H'FFFF FF66	WCR2	A6W2	A6W1	A6W0	A5W2	A5W1	A5W0	A4W2	A4W1	
H'FFFF FF67		A4W0	A3W1	A3W0	A2W1	A2W0	A0W2	A0W1	A0W0	
H'FFFF FF68	MCR	TPC1	TPC0	RCD1	RCD0	TRWL1	TRWL0	TRAS1	TRAS0	
H'FFFF FF69		RASD	—	AMX2	AMX1	AMX0	RFSH	RMODE	—	
H'FFFF FF6A to H'FFFF FF6D	—	—	—	—	—	—	—	—	—	—
H'FFFF FF6E	RTCSR	—	—	—	—	—	—	—	—	BSC
H'FFFF FF6F		CMF	CMIE	CKS2	CKS1	CKS0	OVF	OVIE	LMTS	
H'FFFF FF70	RTCNT									
H'FFFF FF71										
H'FFFF FF72	RTCOR									
H'FFFF FF73										
H'FFFF FF74	RFCR									
H'FFFF FF75										
H'FFFF FF76 to H'FFFF FF7F	—	—	—	—	—	—	—	—	—	—
H'FFFF FF80	FRQCR	STC2	IFC2	PFC2	—	—	—	—	—	CPG
H'FFFF FF81		—	—	STC1	STC0	IFC1	IFC0	PFC1	PFC0	
H'FFFF FF82	STBCR	STBY	—	—	—	—	MSTP2	—	—	Power-down mode
H'FFFF FF83	—	—	—	—	—	—	—	—	—	—
H'FFFF FF84	WTCNT									WDT
H'FFFF FF85	—	—	—	—	—	—	—	—	—	—
H'FFFF FF86	WTCSR	TME	WT/$\overline{\text{IT}}$	RSTS	WOVF	IOVF	CKS2	CKS1	CKS0	WDT
H'FFFF FF87	—	—	—	—	—	—	—	—	—	—
H'FFFF FF88	STBCR2	—	—	MSTP8	MSTP7	—	MSTP5	—	—	Power-down mode
H'FFFF FF89 to H'FFFF FF8F	—	—	—	—	—	—	—	—	—	—

Address	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'FFFF FF90	BDRB	BDB31	BDB30	BDB29	BDB28	BDB27	BDB26	BDB25	BDB24	UBC
H'FFFF FF91		BDB23	BDB22	BDB21	BDB20	BDB19	BDB18	BDB17	BDB16	
H'FFFF FF92		BDB15	BDB14	BDB13	BDB12	BDB11	BDB10	BDB9	BDB8	
H'FFFF FF93		BDB7	BDB6	BDB5	BDB4	BDB3	BDB2	BDB1	BDB0	
H'FFFF FF94	BDMRB	BDMB31	BDMB30	BDMB29	BDMB28	BDMB27	BDMB26	BDMB25	BDMB24	
H'FFFF FF95		BDMB23	BDMB22	BDMB21	BDMB20	BDMB19	BDMB18	BDMB17	BDMB16	
H'FFFF FF96		BDMB15	BDMB14	BDMB13	BDMB12	BDMB11	BDMB10	BDMB9	BDMB8	
H'FFFF FF97		BDMB7	BDMB6	BDMB5	BDMB4	BDMB3	BDMB2	BDMB1	BDMB0	
H'FFFF FF98	BRCR	—	—	—	—	—	—	—	—	
H'FFFF FF99		—	—	—	—	—	—	—	—	
H'FFFF FF9A		SCMFCA	SCMFCB	SCMFDA	SCMFDB	PCTE	PCBA	—	—	
H'FFFF FF9B		DBEB	PCBB	—	—	SEQ	—	—	ETBE	
H'FFFF FF9C	BETR	—	—	—	—					
H'FFFF FF9D										
H'FFFF FF9E	—	—	—	—	—	—	—	—	—	—
H'FFFF FF9F		—	—	—	—	—	—	—	—	
H'FFFF FFA0	BARB	BAB31	BAB30	BAB29	BAB28	BAB27	BAB26	BAB25	BAB24	UBC
H'FFFF FFA1		BAB23	BAB22	BAB21	BAB20	BAB19	BAB18	BAB17	BAB16	
H'FFFF FFA2		BAB15	BAB14	BAB13	BAB12	BAB11	BAB10	BAB9	BAB8	
H'FFFF FFA3		BAB7	BAB6	BAB5	BAB4	BAB3	BAB2	BAB1	BAB0	
H'FFFF FFA4	BAMRB	BAMB31	BAMB30	BAMB29	BAMB28	BAMB27	BAMB26	BAMB25	BAMB24	
H'FFFF FFA5		BAMB23	BAMB22	BAMB21	BAMB20	BAMB19	BAMB18	BAMB17	BAMB16	
H'FFFF FFA6		BAMB15	BAMB14	BAMB13	BAMB12	BAMB11	BAMB10	BAMB9	BAMB8	
H'FFFF FFA7		BAMB7	BAMB6	BAMB5	BAMB4	BAMB3	BAMB2	BAMB1	BAMB0	
H'FFFF FFA8	BBRB	—	—	—	—	—	—	XYE	XYS	UBC
H'FFFF FFA9		CDB1	CDB0	IDB1	IDB0	RWB1	RWB0	SZB1	SZB0	
H'FFFF FFAA	—	—	—	—	—	—	—	—	—	—
H'FFFF FFAB	—	—	—	—	—	—	—	—	—	
H'FFFF FFAC	BRSR	SVF	PID2	PID1	PID0	BSA27	BSA26	BSA25	BSA24	UBC
H'FFFF FFAD		BSA23	BSA22	BSA21	BSA20	BSA19	BSA18	BSA17	BSA16	
H'FFFF FFAE		BSA15	BSA14	BSA13	BSA12	BSA11	BSA10	BSA9	BSA8	
H'FFFF FFAF		BSA7	BSA6	BSA5	BSA4	BSA3	BSA2	BSA1	BSA0	
H'FFFF FFB0	BARA	BAA31	BAA30	BAA29	BAA28	BAA27	BAA26	BAA25	BAA24	
H'FFFF FFB1		BAA23	BAA22	BAA21	BAA20	BAA19	BAA18	BAA17	BAA16	
H'FFFF FFB2		BAA15	BAA14	BAA13	BAA12	BAA11	BAA10	BAA9	BAA8	
H'FFFF FFB3		BAA7	BAA6	BAA5	BAA4	BAA3	BAA2	BAA1	BAA0	

Address	Register Name	Bit Names								Module
		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
H'FFFF FFB4	BAMRA	BAMA31	BAMA30	BAMA29	BAMA28	BAMA27	BAMA26	BAMA25	BAMA24	UBC
H'FFFF FFB5		BAMA23	BAMA22	BAMA21	BAMA20	BAMA19	BAMA18	BAMA17	BAMA16	
H'FFFF FFB6		BAMA15	BAMA14	BAMA13	BAMA12	BAMA11	BAMA10	BAMA9	BAMA8	
H'FFFF FFB7		BAMA7	BAMA6	BAMA5	BAMA4	BAMA3	BAMA2	BAMA1	BAMA0	
H'FFFF FFB8	BBRA	—	—	—	—	—	—	—	—	
H'FFFF FFB9		CDA1	CDA0	IDA1	IDA0	RWA1	RWA0	SZA1	SZA0	
H'FFFF FFBA	—	—	—	—	—	—	—	—	—	—
H'FFFF FFBB		—	—	—	—	—	—	—	—	
H'FFFF FFBC	BRDR	DVF	—	—	—	BDA27	BDA26	BDA25	BDA24	UBC
H'FFFF FFBD		BDA23	BDA22	BDA21	BDA20	BDA19	BDA18	BDA17	BDA16	
H'FFFF FFBE		BDA15	BDA14	BDA13	BDA12	BDA11	BDA10	BDA9	BDA8	
H'FFFF FFBF		BDA7	BDA6	BDA5	BDA4	BDA3	BDA2	BDA1	BDA0	
H'FFFF FFC0 to H'FFFF FFEB	—	—	—	—	—	—	—	—	—	—
H'FFFF FFEC	CCR	—	—	—	—	—	—	—	—	Cache
H'FFFF FFED		—	—	—	—	—	—	—	—	
H'FFFF FFEE		—	—	—	—	—	—	—	—	
H'FFFF FFEF		—	—	—	—	CF	CB	WT	CE	

Appendix B Pin Functions

B.1 Pin States

Table B.1 shows pin states during resets, power-down states, and the bus-released states.

Table B.1 Pin States during Resets, Power-Down States, and Bus-Released State

		Reset		Power-Down	
Category	**Pin**	**Power-On Reset**	**Manual Reset**	**Standby**	**Bus Released**
Clock	EXTAL	I	I	I	I
	XTAL	O	O	O	O
	CKIO	O	O	OZ*[2]	OZ*[2]
	CAP1, CAP2	—	—	—	—
System control	$\overline{\text{RESETP}}$	I	I	I	I
	$\overline{\text{RESETM}}$	I	I	I	I
	$\overline{\text{BREQ}}$	I	I	I	I
	$\overline{\text{BACK}}$	O	O	O	L
	MD[4:0]	I	I	I	I
	STATUS[1:0]/PTJ[7:6]	O	O/P	O/P	O/P
Interrupt	NMI	I	I	I	I
	$\overline{\text{IRL}}$[3:0]/IRQ[3:0]/ PTH[3:0]	V*[1]	I/I/I	I/I/I	I/I/I
	IRQ4/ PTH[4]	V*[1]	I/I	I/I	I/I
	IRQ5/SCPT[7]	V*[1]	Z/I	I/I	I/I
	IRQ6/PTC7	V	I/K	I/K	I/K
	IRQ7/PTC6	V	I/K	I/K	I/K
	TCK/PTF4	I/V	I/K	I/K	I/K
	TDI/PTF5	I/V	I/K	I/K	I/K
	TMS/PTF6	I/V	I/K	I/K	I/K
	TRST/PTF7	I/V	I/K	I/K	I/K
	$\overline{\text{IRQOUT}}$	O	O	O	O

Table B.1 Pin States during Resets, Power-Down States, and Bus-Released State (cont)

		Reset		Power-Down	
Category	**Pin**	**Power-On Reset**	**Manual Reset**	**Standby**	**Bus Released**
Address bus	A[25:0]	Z	O	ZL*2	Z
Data bus	D[15:0]	Z	I	Z	Z
	D[23:16]/PTA[7:0]	Z	I/P	Z/K	Z/P
	D[31:24]/PTB[7:0]	Z	I/P	Z/K	Z/P
Bus control	$\overline{\text{CS0}}$	H	O	ZH*2	Z
	$\overline{\text{CS[2:4]}}$/PTK[0:2]	H	O/P	ZH*2/K	Z/P
	$\overline{\text{CS5}}$/PTK[3]	H	O/P	ZH*2/K	Z/P
	$\overline{\text{CS6}}$	H	O	ZH*2	Z
	$\overline{\text{BS}}$/PTK[4]	H	O/P	ZH*2/K	Z/P
	$\overline{\text{RAS3L}}$/PTJ[0]	H	O/P	ZO/K	ZO/P
	PTJ[1]	H	P	K	P
	$\overline{\text{RAS3U}}$/PTE[2]	Z	O/P	ZO/K	ZO/P
	PTE[1]	Z	P	K	P
	PTE[6]	Z	P	K	P
	PTE[3]	Z	P	K	P
	CASL/PTJ[2]	H	O/P	ZO/K	ZO/P
	CASU/PTJ[3]	H	O/P	ZO/K	ZO/P
	PTJ[4]	H	P	K	P
	PTJ[5]	H	P	K	P
	$\overline{\text{WE0}}$/DQMLL	H	O	ZH*2	Z
	$\overline{\text{WE1}}$/DQMLU/$\overline{\text{WE}}$	H	O	ZH*2	Z
	$\overline{\text{WE2}}$/DQMUL/PTK[6]	H	O/P	ZH*2/K	Z/P
	$\overline{\text{WE3}}$/DQMUU/PTK[7]	H	O/P	ZH*2/K	Z/P
	RD/$\overline{\text{WR}}$	H	O	ZH*2	Z
	$\overline{\text{RD}}$	H	O	ZH*2	Z
	CKE/PTK[5]	H	O/P	ZO/K	O/P
	$\overline{\text{WAIT}}$	Z	I	Z	Z

Table B.1 Pin States during Resets, Power-Down States, and Bus-Released State (cont)

		Reset		Power-Down	
Category	**Pin**	**Power-On Reset**	**Manual Reset**	**Standby**	**Bus Released**
DMAC	$\overline{\text{DREQ0}}$/PTD[4]	V	Z/I	Z	I
	DACK0/PTD[5]	Z	O/P	Z/K	O/P
	DRAK0/PTD[1]	V	O/P	ZH*^{2}/K	O/P
	$\overline{\text{DREQ1}}$/PTD[6]	V	Z/I	Z	I
	DACK1/PTD[7]	Z	O/P	Z/K	O/P
	DRAK1/PTD[0]	V	O/P	ZH*^{2}/K	O/P
Timer	TCLK/PTH[7]	V	Z/P	IO/P	IO/P
SCIF0	RxD0/SCPT[0]	Z	Z/I	Z	I/Z
	TxD0/SCPT[0]	Z	Z/O	Z/K	O/Z
	SCK0/SCPT[1]	V	Z/P	Z/K	IO/P
	SCPT[6]	V	I/P	I/K	I/P
SCIF1	RxD1/SCPT[2]	Z	Z/I	Z/K	I/Z
	TxD1/SCPT[2]	Z	Z/O	Z/K	O/Z
	SCK1/SCPT[3]	V	Z/P	Z/K	IO/P
SCIF2	RxD2/SCPT[4]	Z	Z/I	Z/K	I/Z
	TxD2/SCPT[4]	Z	Z/O	Z/K	O/Z
	SCK2/SCPT[5]	V	Z/P	Z/K	IO/P
	IRQ5/SCPT[7]	V*^{1}	Z/I	I/I	I/I
USB	DMNS/PTF3	V	I/P	I/K	I/P
	DPLS/PTF2	V	I/P	I/K	I/P
	TXDPLS/PTF1	V	O/P	O/K	O/P
	TXDMNS/PTF0	V	O/P	O/K	O/P
	XVDATA/PTC5	V	I/P	I/K	I/P
	TXENL/PTC4	V	O/P	O/K	O/P
	VBUS/PTD3	V	I/P	I/K	I/P
	SUSPND/PTD2	V	O/P	O/K	O/P
	UCLK/PTG4	V	I/P	I/K	I/I

Table B.1 Pin States during Resets, Power-Down States, and Bus-Released State (cont)

		Reset		Power-Down	
Category	**Pin**	**Power-On Reset**	**Manual Reset**	**Standby**	**Bus Released**
Ports (excluding previously mentioned pins)	PTC3	O	P	K	P
	PTC2	O	P	K	P
	PTC1	I	P	K	P
	PTC0	O	P	K	P
	AUDSYNC/PTE[7]	O/V	O/P	O/K	O/P
	PTE[5]	Z	P	K	P
	PTE[4]	Z	P	K	P
	TDO/PTE[0]	O/V	O/P	O/K	O/P
	PTG[7]	V	I/I	Z/K	I/Z
	AUDCK/PTH[6]	V	I/I	Z/K	I/I
	ADTRG/PTH[5]	V*[1]	I/I	I/K	I/I
	AUDATA[3:0]/PTG[3:0]	I/V port	I	I/K port	I/I
	ASEBRKAK/PTG[5]	O/V port	O/I port	O/K port	O/I port
	ASEMD0/PTG[6]	I(ASEMD)/V	I/I	Z/K	I/I
	PTL[7:4]	Z	P	Z	P
Analog	AN[3:0]/PTL[3:0]	Z	Z/I	Z	I

I: Input
O: Output
H: High-level output
L: Low-level output
Z: High impedance
P: Input or output depending on register setting
K: Input pin is high impedance, output pin holds the state
V: I/O buffer off, pullup MOS on

Notes: *1 Input Schmidt buffers and pullup MOS of IRQ[5:0] and ADTRG are on; other inputs are off.

*2 In the standby mode, Z or otherwise depending on register setting.

Appendix C Notes on Consecutive Execution of Multiply-Accumulate/Multiplication and DSP Instructions

Problem

When the execution of instructions is stalled by contention for a multiplier by multiplication/multiply-accumulate instructions, or contention for registers by the consecutive execution of DSP instructions, and the S bit (saturation operation bit) in the SR (status register) is changed immediately after a multiplication/multiply-accumulate instruction or DSP instruction, the order of instruction execution is reversed. That is, an instruction which causes the state of the S bit to change might actually be executed after the bit has been changed, so a false result for an operation might be indicated.

- Instructions which will be affected by the S bit change

  Multiply-accumulate instructions: MAC.W, MAC.L

  DSP instructions: ALU arithmetic instructions, fixed-point multiplication instructions, arithmetic shift instructions

Conditions for Occurrence

Examples where the problem may arise are given below.

1. In the case of a multiplication/multiply-accumulation instruction

   a. DMULU.L R4, R10 ← Applies to MUL.L, DMULS.L, DMULU.l, MAC.L.

   b. MAC.L @R5+, @R5+ ← Applies to MAC.W, MAC.L.
   Contention for multipliers occurs and conditions for stalling of instruction execution are satisfied.

   c. LDC R0, SR ← Saturation operation mode change

   A contention for multipliers occurs when the DMULU.L instruction of a and MAC.L instruction of b are executed, and the MAC.L instruction execution of b will be stalled.

   The S bit change caused by c is a pipelined operation, so it will be executed before the MAC.L instruction of a, and the order of executions of b and c will be reversed as a result, and the result of the MAC.L instruction will be a false result.

2. In the case of a DSP instruction
   a. PSHA #1, A1
   b. PINC X0, A0 MOVX.W Al, @R5 ← Contention for multipliers occurs and conditions for stalling of instruction execution are satisfied.
   c. LDC R0, SR ← Saturation operation mode change

   The result of the DSP operation is stored immediately after it is executed, so a contention for registers will occur between the PSHA instruction of a and MOVXL instruction of b, so the execution of the PINC instruction of b will be stalled.

   The S bit change caused by c is a pipelined operation, so it will be executed before the PINC instruction of b, and the order of execution of b and c will be reversed as a result, and the result of the PING instruction will be a false result.

Prevention Methods on Programs

To prevent the above limitations, be sure to follow either one of these three procedures.

1. Do not access the SR register immediately after a multiply-accumulate or DSP instruction.
2. Do not insert an NOP instruction immediately before an LDC Rn, SR instruction.
3. Be sure not to cause contention between multipliers or DSP registers (so that a stall does not occur).

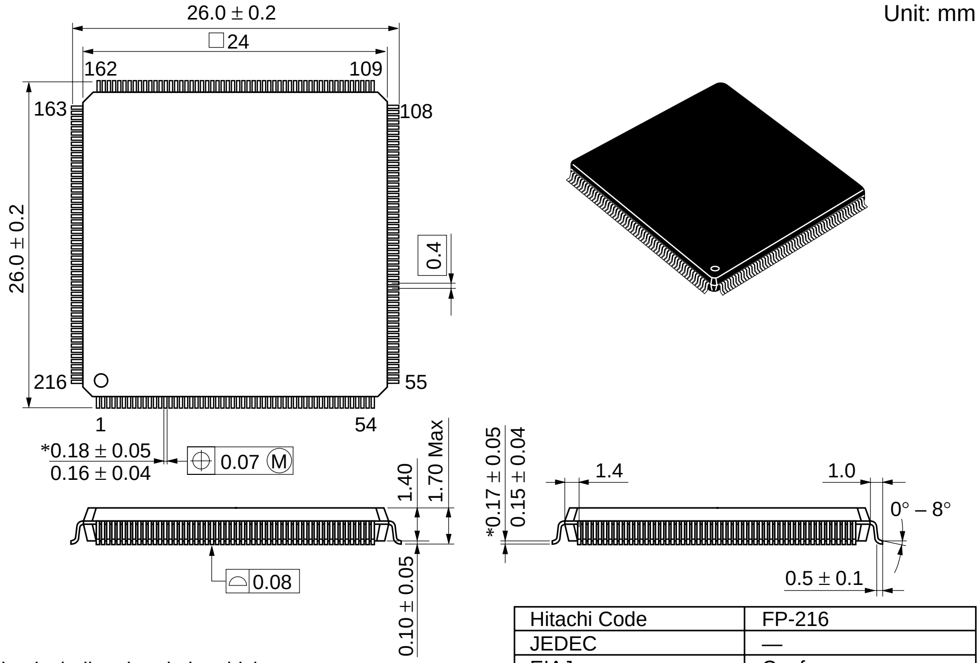
Appendix D Product Lineup

Table D.1 SH7622 Product Lineup

Product Name	Voltage	Operating Frequency	Marked Name	Package
SH7622	3.0 V	80 MHz	HD6417622FL80	216-pin plastic LQFP (FP-216)
			HD6417622BP80	208-pin TFBGA (TBP-208A)
			HD6417622F80	208-pin plastic QFP (FP-208C)
		100 MHz	HD6417622FL100	216-pin plastic LQFP (FP-216)
			HD6417622BP100	208-pin TFBGA (TBP-208A)
			HD6417622F100	208-pin plastic QFP (FP-208C)

Appendix E   Package Dimensions

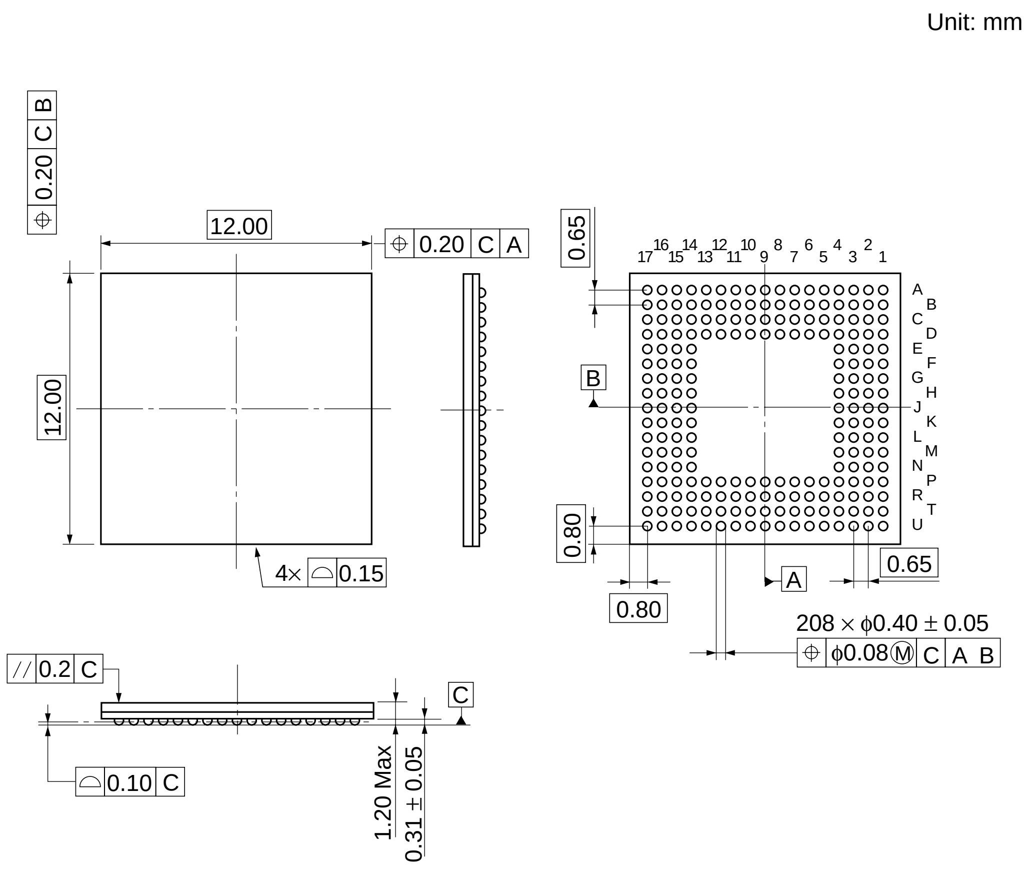
The SH7622 package dimensions are shown in figures E.1 (FP-216), E.2 (TBP-208A), and E.3 (FP-208C).



Hitachi Code	FP-216
JEDEC	—
EIAJ	Conforms
Weight (reference value)	—

$\frac{\text{*Dimension including the plating thickness}}{\text{Base material dimension}}$

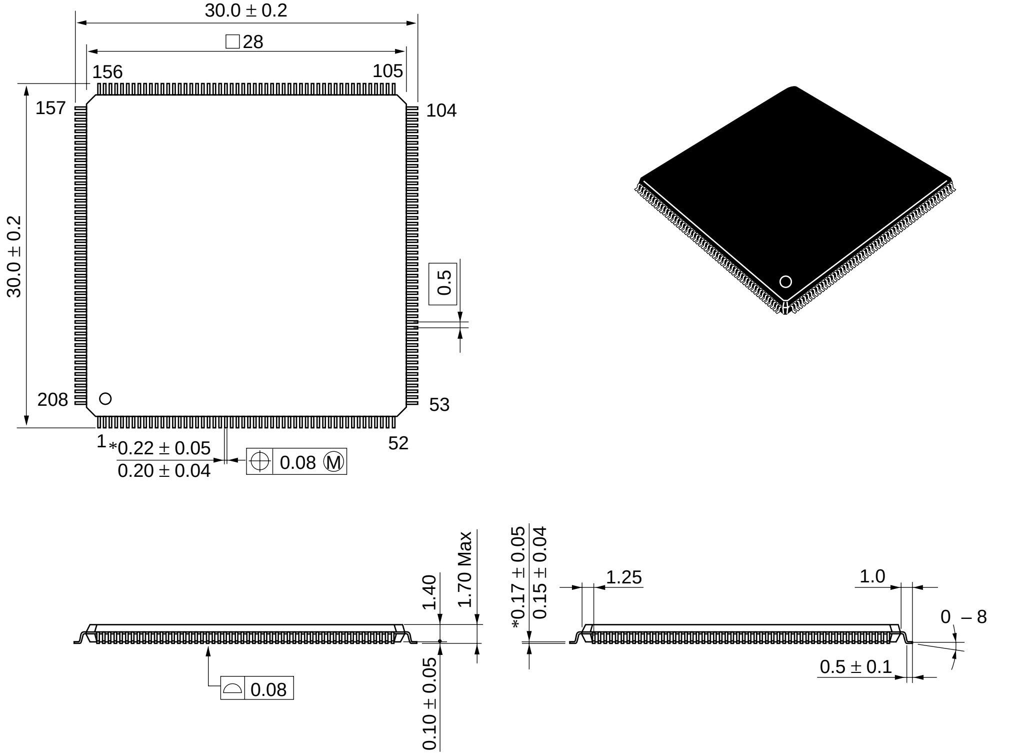
Figure E.1   Package Dimensions (FP-216)



Hitachi Code	TBP-208A
JEDEC	—
EIAJ	—
Weight (reference value)	0.25 g

Figure E.2 Package Dimensions (TBP-208A)

Unit: mm



*Dimension including the plating thickness
Base material dimension

Hitachi Code	FP-208C
JEDEC	—
EIAJ	Conforms
Weight (reference value)	2.7 g

Figure E.3 Package Dimensions (FP-208C)

SH7622 Hardware Manual

Publication Date: 1st Edition, January 2001
2nd Edition, September 2001
Published by: Customer Service Division
Semiconductor & Integrated Circuits
Hitachi, Ltd.
Edited by: Technical Documentation Group
Hitachi Kodaira Semiconductor Co., Ltd.