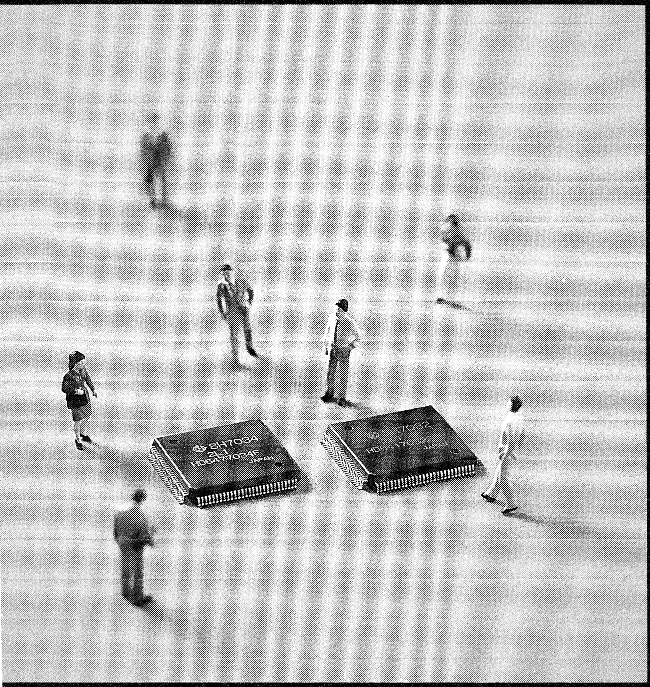
HITACHI

Hitachi Single-Chip RISC Microcomputer

SH7000Series

SH7032, SH7034, SH7020, SH7021



Hitachi semiconductor

When using this document, keep the following in mind:

1. This document may, wholly or partially, be subject to change without notice.
2. All rights are reserved: No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without Hitachi's permission.
3. Hitachi will not be held responsible for any damage to the user that may result from accidents or any other reasons during operation of the user's unit according to this document.
4. Circuitry and other examples described herein are meant merely to indicate the characteristics and performance of Hitachi's semiconductor products. Hitachi assumes no responsibility for any intellectual property claims or other problems that may result from applications based on the examples described herein.
5. No license is granted by implication or otherwise under any patents or other rights of any third party or Hitachi, Ltd.
6. MEDICAL APPLICATIONS: Hitachi's products are not authorized for use in MEDICAL APPLICATIONS without the written consent of the appropriate officer of Hitachi's sales company. Such use includes, but is not limited to, use in life support systems. Buyers of Hitachi's products are requested to notify the relevant Hitachi sales offices when planning to use the products in MEDICAL APPLICATIONS.

Introduction

The SH7000 (SH) series (SH7032, SH7034, SH7020, SH7021) is part of a new generation of reduced instruction set code (RISC : Reduced instruction set computer) microcomputers that integrate a RISC CPU and the peripheral functions required for system configuration onto a single chip to achieve high-performance operations processing. The SH series can operate in a power-down state, which is an essential feature for portable equipment.

The SH series CPUs have RISC instruction sets. Basic instructions can be executed in one clock cycle, which strikingly improves instruction execution speed. The SH microcomputers include peripheral functions such as large-capacity ROM (PROM or masked ROM), RAM, direct memory access controllers (DMAC), timers, serial communication interfaces (SCI), A/D converters (SH7032 and SH7034 only), interrupt controllers, and I/O ports. These onchip elements enable users to construct systems with the fewest possible components. External memory access support functions enable direct connection to SRAM and DRAM without the use of glue logic.

Support tools are available to increase the efficiency of developing application systems for the SH series. These include a real-time emulator (the E7000 series) that can be connected to workstations (EWS) and LANs, a compact evaluation board (A7 size) that can be combined with existing equipment, and other software tools such as an optimizing C compiler. These are all provided within an integrated programming environment so that the same user interface is used in all stages from program development through system debugging.

This document describes the main functions of the SH series. For detailed information about instructions, please refer to the *SH7000 Series Programming Manual* for the product in question. For more information on hardware functions and use, please refer to the *SH7000 Series Hardware Manual.*

Related Manuals

SH7032 and SH7034 Hardware:
SH7032, SH7034 Hardware Manual (Document No. ADE-602-062).
SH7020, SH7021 Hardware Manual (Document No. ADE-602-074).

SH7000 Series Instructions:
SH7000/SH7600 Series Programming Manual (Document No. ADE-602-063A).

For development support tools, contact your Hitachi sales office.

Note: The SH7032, SH7034, SH7020 and SH7021 differ as follows:

	ROM	RAM	Port		A/D Converter	Package
			I/Os	Inputs		
SH7032	—	8 kbytes	32	8	Yes	FP-112
SH7034	64 kbytes	4 kbytes	32	8	Yes	FP-112
SH7020	16 kbytes	1 kbytes	32	—	No	TFP-100
SH7021	32 kbytes	1 kbytes	32	—	No	TFP-100

Note: SH7034 has PROM or Masked ROM.
SH7020 and SH7021 have Masked ROM.

Contents

Section 1 SH7000 Series Features

The SH7000 series microcomputers are a new generation of reduced instruction set code (RISC) devices in which a Hitachi-original RISC CPU and peripheral functions required for system configuration are integrated in a single chip. Table 1.1 lists SH7000 series features.

Most instructions can be executed in one clock cycle, greatly improving instruction execution speed. The CPU also features 32-bit internal architecture for enhanced data-processing ability. This architecture enables high-performance systems to be constructed with advanced functionality at low cost, even in applications such as real-time control that require very high speeds.

The SH7000 series include peripheral functions such as large-capacity ROM, RAM, direct memory access controllers (DMAC), timers, serial communication interfaces (SCI), A/D converters, interrupt controllers, and I/O ports. External memory access support functions enable direct connection to SRAM and DRAM. These features can significantly reduce system cost.

The SH7000 Series is made up of the SH7032, SH7034, SH7020, and SH7021. The SH7034 is available in either masked ROM or ZTAT* versions, the latter with an on-chip PROM that can be programmed by the user. The ZTAT version can be programmed with a general-purpose PROM writer. The on-chip ROM for the SH7020 and SH7021 is masked ROM.

*ZTAT (Zero Turn Around Time) is trademark of Hitachi, Ltd.

Table 1.1 Features of the SH7000 Series

Feature	Description
CPU	• Original Hitachi architecture • 32-bit internal data paths • General-register machines: — Sixteen 32-bit general registers — Three 32-bit control registers — Four 32-bit system registers • RISC instruction set — Instruction length: 16-bit fixed length for improved code efficiency — Load-store architecture (basic arithmetic and logic operations are executed between registers) — Delayed unconditional branch instructions reduce pipeline disruption — Instruction set optimized for C language • Instruction execution time: basic instructions executed one instruction/cycle (50 ns per instruction at 20-MHz operation)

Table 1.1 Features of the SH7000 Series (cont)

Feature	Description
CPU (cont)	• Address space: 4 Gbytes available on the architecture • Onchip multiplier: multiplication operations (16 bits × 16 bits → 32 bits) executed in 1 to 3 cycles, and multiplication/accumulation operations (16 bits × 16 bits + 42 bits → 42 bits) executed in 2 to 3 cycles • 5-stage pipeline
Operating modes	• Operating modes: — Onchip ROMless mode — Onchip ROM mode • Processing states: — Power-on reset state — Manual reset state — Exception processing state — Program execution state — Power-down state — Bus-released state • Power-down states: — Sleep mode — Standby mode
Interrupt controller (INTC)	• 9 external interrupt pins (NMI, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ7}}$) • Internal interrupt sources — 31 [SH7032, SH7034] — 30 [SH7020, SH7021] • 16 programmable priority levels
User break controller (UBC)	• Generates an interrupt when the CPU or DMAC generates a bus cycle with specified conditions • Simplifies configuration of a self-debugger
Clock pulse generator (CPG)	• Onchip clock pulse generator (maximum operating frequency: 20 MHz): — 20-MHz pulses generated from a 20-MHz crystal using duty cycle correction — System clock output is controllable [SH7020, SH7021]

Large onchip memory		Memory		
	Product Name	**ROM**	**RAM**	**ROM type**
	SH7032	—	8 kbytes	—
	SH7034	64 kbytes	4 kbytes	PROM or masked ROM
	SH7020	16 kbytes	1 kbytes	Masked ROM
	SH7021	32 kbytes	1 kbytes	Masked ROM

Table 1.1 Features of the SH7000 Series (cont)

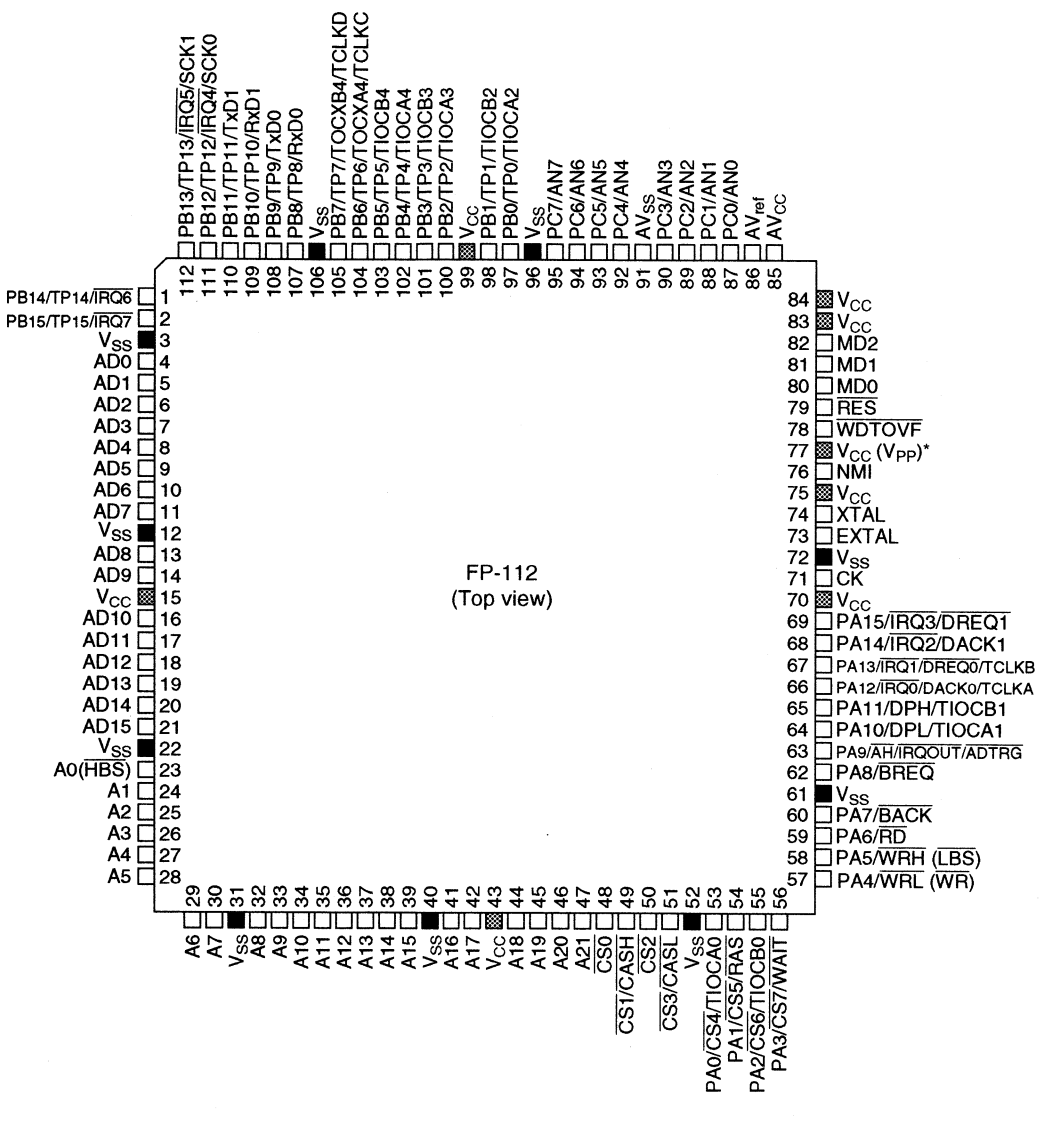
Feature	Description
Bus state controller (BSC)	• Supports external memory access — 16-bit external data bus • Address space divided into 8 areas with the following preset features: — Bus width (8 or 16 bits) — Number of wait cycles (variable) — Type of area (external memory space, DRAM space, etc.) • Simplifies connection to ROM, SRAM, DRAM, and peripheral I/O — When the DRAM area is accessed: - $\overline{\text{RAS}}$ and CAS signals for DRAM are output - Tp cycles can be generated to assure $\overline{\text{RAS}}$ precharge time - Address multiplexing is supported internally; DRAM can be connected directly — Chip select signals ($\overline{\text{CS0}}$ to $\overline{\text{CS7}}$) are output for each area • DRAM refresh function — Programmable refresh interval — Supports CAS-before-$\overline{\text{RAS}}$ refresh and self-refresh modes • DRAM burst access function — Supports high-speed access modes for DRAM • Wait cycles can be inserted by an external $\overline{\text{WAIT}}$ signal • One-stage write buffer improves the system performance • Data bus parity can be generated and checked
Direct memory access controller (DMAC) (4 channels)	• Permits DMA transfer between the following devices: — External memory, external I/O, onchip memory, and peripheral onchip modules (except DMAC) • DMA transfer can be requested from external pins, onchip SCI, onchip timers, and onchip A/D converter • Cycle-steal mode or burst mode • Channel priority level is selectable • Channels 0 and 1: dual or single address transfer mode is selectable; external request sources are supported. Channels 2 and 3: dual address transfer mode, internal request sources only
16-bit integrated-timer pulse unit (ITU)	• Ten types of waveforms can be output • Input pulse width and cycle can be measured • PWM mode: pulse output with 0 to 100% duty cycle (maximum resolution: 50 ns) • Complementary PWM mode: maximum 3 pairs of nonoverlapping PWM waveforms can be output • Phase counting mode: can count up or down according to the phase of an external two-phase clock

Table 1.1 Features of the SH7000 Series (cont)

Feature	Description
Programmable timing pattern controller (TPC)	• Maximum 16-bit output (4 bits × 4 channels) can be output • Non-overlap intervals can be established between pairs of waveforms • Timing source is selectable
Watchdog timer (WDT) (1 channel)	• Watchdog timer or interval timer can be selected • Count overflow can generate an internal reset, external signal, or interrupt • Power-on reset or manual reset can be selected as the internal reset
Serial communication interface (SCI) (2 channels)	• Mode (asynchronous or clocked synchronous) is selectable • Can transmit and receive simultaneously (full duplex) • Onchip baud rate generator in each channel • Multiprocessor communication function
A/D converter [SH7032, SH7034]	• 10 bits × 8 channels • Can be externally triggered • Variable reference voltage
I/O ports	• Total of 40 I/O lines: 32 input/output lines; 8 input-only lines [SH7032, SH7034] • Total of 32 I/O lines: 32 input/output lines [SH7020, SH7021] — Port A: 16 input/output lines (input or output can be selected for each bit) — Port B: 16 input/output lines (input or output can be selected for each bit) — Port C: 8 input lines [SH7032, SH7034]
Package	112-pin plastic QFP [SH7032, SH7034] 100-pin plastic shrink QFP [SH7020, SH7021]

Section 2 Pin Arrangement and Functions

Figure 2.1 shows SH7032 and SH7034 device pin arrangement (FP-112 package). Figure 2.2 shows SH7020 and SH7021 device pin arrangement (TFP-100). Tables 2.1 and 2.2 list pins by operating modes, and table 2.3 lists pin functions.



Note: V_{CC}: SH7032
V_{PP}: SH7034

Figure 2.1 Pin Arrangement (SH7032, SH7034)

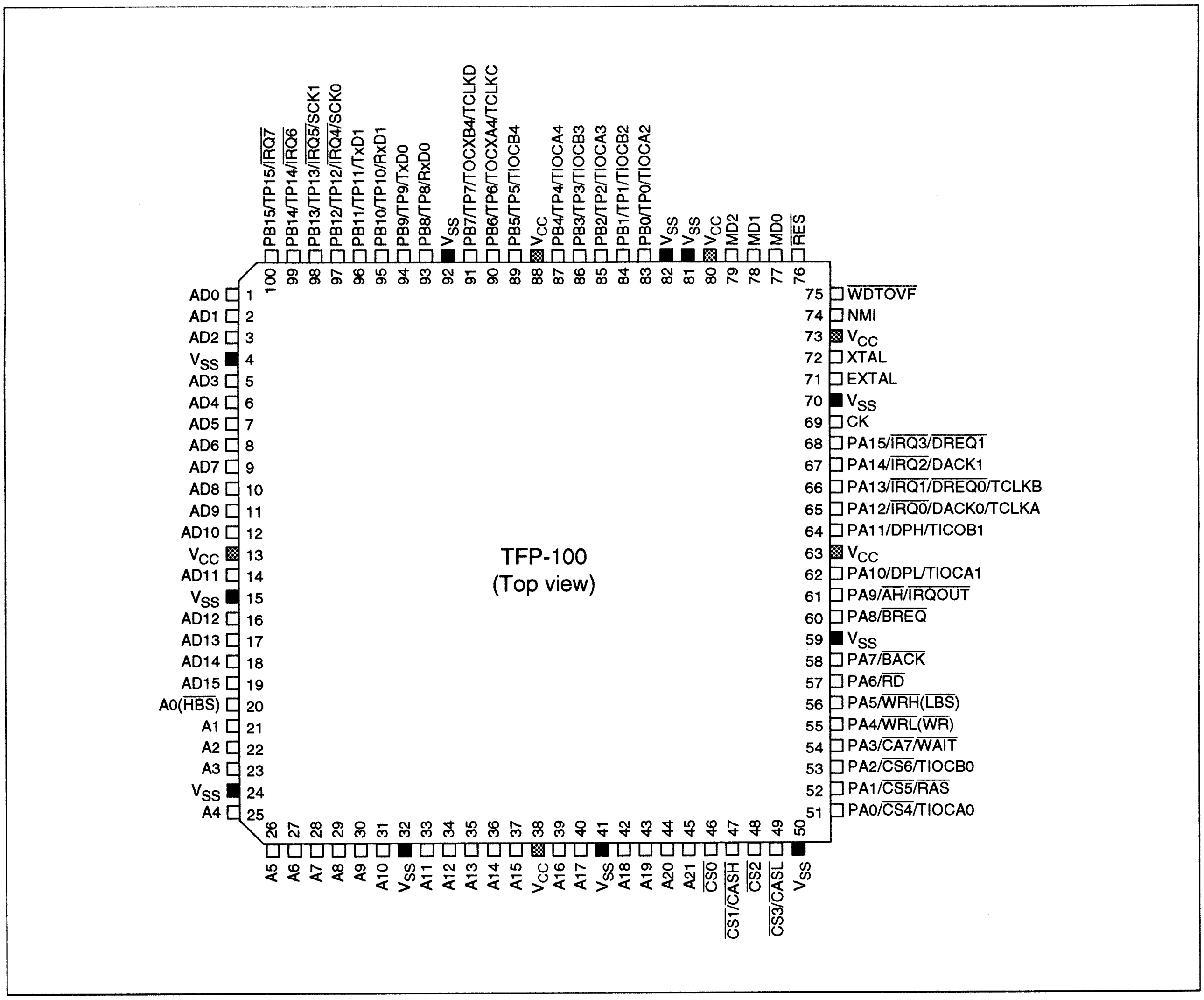


Figure 2.2 Pin Arrangement (SH7020, SH7021)

Table 2.1 SH7032 and SH7034 Pins Listed by Operating Modes

Pin No. (FP-112)	MCU Mode	PROM Mode (SH7034)
1	PB14/TP14/$\overline{\text{IRQ6}}$	NC
2	PB15/TP15/$\overline{\text{IRQ7}}$	NC
3	V_{SS}	V_{SS}
4	AD0	D0
5	AD1	D1
6	AD2	D2
7	AD3	D3
8	AD4	D4
9	AD5	D5
10	AD6	D6
11	AD7	D7
12	V_{SS}	V_{SS}
13	AD8	NC
14	AD9	NC
15	V_{CC}	V_{CC}
16	AD10	NC
17	AD11	NC
18	AD12	NC
19	AD13	NC
20	AD14	NC
21	AD15	NC
22	V_{SS}	V_{SS}
23	A0 ($\overline{\text{HBS}}$)	A0
24	A1	A1
25	A2	A2
26	A3	A3
27	A4	A4
28	A5	A5
29	A6	A6
30	A7	A7
31	V_{SS}	V_{SS}
32	A8	A8
33	A9	$\overline{\text{OE}}$
34	A10	A10
35	A11	A11
36	A12	A12

Table 2.1 SH7032 and SH7034 Pins Listed by Operating Modes (cont)

Pin No. (FP-112)	MCU Mode	PROM Mode (SH7034)
37	A13	A13
38	A14	A14
39	A15	A15
40	V_{SS}	V_{SS}
41	A16	A16
42	A17	V_{CC}
43	V_{CC}	V_{CC}
44	A18	V_{CC}
45	A19	NC
46	A20	NC
47	A21	NC
48	$\overline{\text{CS0}}$	NC
49	$\overline{\text{CS1}}$/$\overline{\text{CASH}}$	NC
50	$\overline{\text{CS2}}$	NC
51	$\overline{\text{CS3}}$/$\overline{\text{CASL}}$	NC
52	V_{SS}	V_{SS}
53	PA0/$\overline{\text{CS4}}$/TIOCA0	NC
54	PA1/$\overline{\text{CS5}}$/$\overline{\text{RAS}}$	NC
55	PA2/$\overline{\text{CS6}}$/TIOCB0	$\overline{\text{PGM}}$
56	PA3/$\overline{\text{CS7}}$/$\overline{\text{WAIT}}$	$\overline{\text{CE}}$
57	PA4/$\overline{\text{WRL}}$ ($\overline{\text{WR}}$)	NC
58	PA5/$\overline{\text{WRH}}$ ($\overline{\text{LBS}}$)	NC
59	PA6/$\overline{\text{RD}}$	NC
60	PA7/$\overline{\text{BACK}}$	NC
61	V_{SS}	V_{SS}
62	PA8/$\overline{\text{BREQ}}$	NC
63	PA9/$\overline{\text{AH}}$/$\overline{\text{IRQOUT}}$/$\overline{\text{ADTRG}}$	NC
64	PA10/DPL/TIOCA1	NC
65	PA11/DPH/TIOCB1	NC
66	PA12/$\overline{\text{IRQ0}}$/DACK0/TCLKA	NC
67	PA13/$\overline{\text{IRQ1}}$/$\overline{\text{DREQ0}}$/TCLKB	NC
68	PA14/$\overline{\text{IRQ2}}$/DACK1	NC
69	PA15/$\overline{\text{IRQ3}}$/$\overline{\text{DREQ1}}$	NC
70	V_{CC}	V_{CC}
71	CK	NC
72	V_{SS}	V_{SS}

Table 2.1 SH7032 and SH7034 Pins Listed by Operating Modes (cont)

Pin No. (FP-112)	MCU Mode	PROM Mode (SH7034)
73	EXTAL	NC
74	XTAL	NC
75	V_{CC}	V_{CC}
76	NMI	A9
77	V_{CC} (SH7032), V_{PP} (SH7034)	V_{PP}
78	$\overline{\text{WDTOVF}}$	NC
79	$\overline{\text{RES}}$	V_{SS}
80	MD0	V_{CC}
81	MD1	V_{CC}
82	MD2	V_{CC}
83	V_{CC}	V_{CC}
84	V_{CC}	V_{CC}
85	AV_{CC}	V_{CC}
86	AVref	V_{CC}
87	PC0/AN0	V_{SS}
88	PC1/AN1	V_{SS}
89	PC2/AN2	V_{SS}
90	PC3/AN3	V_{SS}
91	AV_{SS}	V_{SS}
92	PC4/AN4	V_{SS}
93	PC5/AN5	V_{SS}
94	PC6/AN6	V_{SS}
95	PC7/AN7	V_{SS}
96	V_{SS}	V_{SS}
97	PB0/TP0/TIOCA2	NC
98	PB1/TP1/TIOCB2	NC
99	V_{CC}	V_{CC}
100	PB2/TP2/TIOCA3	NC
101	PB3/TP3/TIOCB3	NC
102	PB4/TP4/TIOCA4	NC
103	PB5/TP5/TIOCB4	NC
104	PB6/TP6/TOCXA4/TCLKC	NC
105	PB7/TP7/TOCXB4/TCLKD	NC
106	V_{SS}	V_{SS}
107	PB8/TP8/RxD0	NC
108	PB9/TP9/TxD0	NC

Table 2.1 SH7032 and SH7034 Pins Listed by Operating Modes (cont)

Pin No. (FP-112)	MCU Mode	PROM Mode (SH7034)
109	PB10/TP10/RxD1	NC
110	PB11/TP11/TxD1	NC
111	PB12/TP12/$\overline{\text{IRQ4}}$/SCK0	NC
112	PB13/TP13/$\overline{\text{IRQ5}}$/SCK1	NC

Table 2.2 SH7020 and SH7021 Pins Listed by Operating Modes

Pin No. (TQFP-100)	MCU Mode
1	AD0
2	AD1
3	AD2
4	V_{SS}
5	AD3
6	AD4
7	AD5
8	AD6
9	AD7
10	AD8
11	AD9
12	AD10
13	V_{CC}
14	AD11
15	V_{SS}
16	AD12
17	AD13
18	AD14
19	AD15
20	A0 ($\overline{\text{HBS}}$)
21	A1
22	A2
23	A3
24	V_{SS}
25	A4
26	A5
27	A6
28	A7
29	A8

Table 2.2 SH7020 and SH7021 Pins Listed by Operating Modes (cont)

Pin No. (TQFP-100)	MCU Mode
30	A9
31	A10
32	V_{SS}
33	A11
34	A12
35	A13
36	A14
37	A15
38	V_{CC}
39	A16
40	A17
41	V_{SS}
42	A18
43	A19
44	A20
45	A21
46	$\overline{\text{CS0}}$
47	$\overline{\text{CS1}}/\overline{\text{CASH}}$
48	$\overline{\text{CS2}}$
49	$\overline{\text{CS3}}/\overline{\text{CASL}}$
50	V_{SS}
51	PA0/$\overline{\text{CS4}}$/TIOCA0
52	PA1/$\overline{\text{CS5}}/\overline{\text{RAS}}$
53	PA2/$\overline{\text{CS6}}$/TIOCB0
54	PA3/$\overline{\text{CS7}}/\overline{\text{WAIT}}$
55	PA4/$\overline{\text{WRL}}$ ($\overline{\text{WR}}$)
56	PA5/$\overline{\text{WRH}}$ ($\overline{\text{LBS}}$)
57	PA6/$\overline{\text{RD}}$
58	PA7/$\overline{\text{BACK}}$
59	V_{SS}
60	PA8/$\overline{\text{BREQ}}$
61	PA9/$\overline{\text{AH}}/\overline{\text{IRQOUT}}/\overline{\text{ADTRG}}$
62	PA10/DPL/TIOCA1
63	V_{CC}
64	PA11/DPH/TIOCB1
65	PA12/$\overline{\text{IRQ0}}$/DACK0/TCLKA

Table 2.2 SH7020 and SH7021 Pins Listed by Operating Modes (cont)

Pin No. (TQFP-100)	MCU Mode
66	PA13/$\overline{\text{IRQ1}}$/$\overline{\text{DREQ0}}$/TCLKB
67	PA14/$\overline{\text{IRQ2}}$/DACK1
68	PA15/$\overline{\text{IRQ3}}$/$\overline{\text{DREQ1}}$
69	CK
70	V_{SS}
71	EXTAL
72	XTAL
73	V_{CC}
74	NMI
75	$\overline{\text{WDTOVF}}$
76	$\overline{\text{RES}}$
77	MD0
78	MD1
79	MD2
80	V_{CC}
81	V_{SS}
82	V_{SS}
83	PB0/TP0/TIOCA2
84	PB1/TP1/TIOCB2
85	PB2/TP2/TIOCA3
86	PB3/TP3/TIOCB3
87	PB4/TP4/TIOCA4
88	V_{CC}
89	PB5/TP5/TIOCB4
90	PB6/TP6/TOCXA4/TCLKC
91	PB7/TP7/TOCXB4/TCLKD
92	V_{SS}
93	PB8/TP8/RxD0
94	PB9/TP9/TxD0
95	PB10/TP10/RxD1
96	PB11/TP11/TxD1
97	PB12/TP12/$\overline{\text{IRQ4}}$/SCK0
98	PB13/TP13/$\overline{\text{IRQ5}}$/SCK1
99	PB14/TP14/$\overline{\text{IRQ6}}$
100	PB15/TP15/$\overline{\text{IRQ7}}$

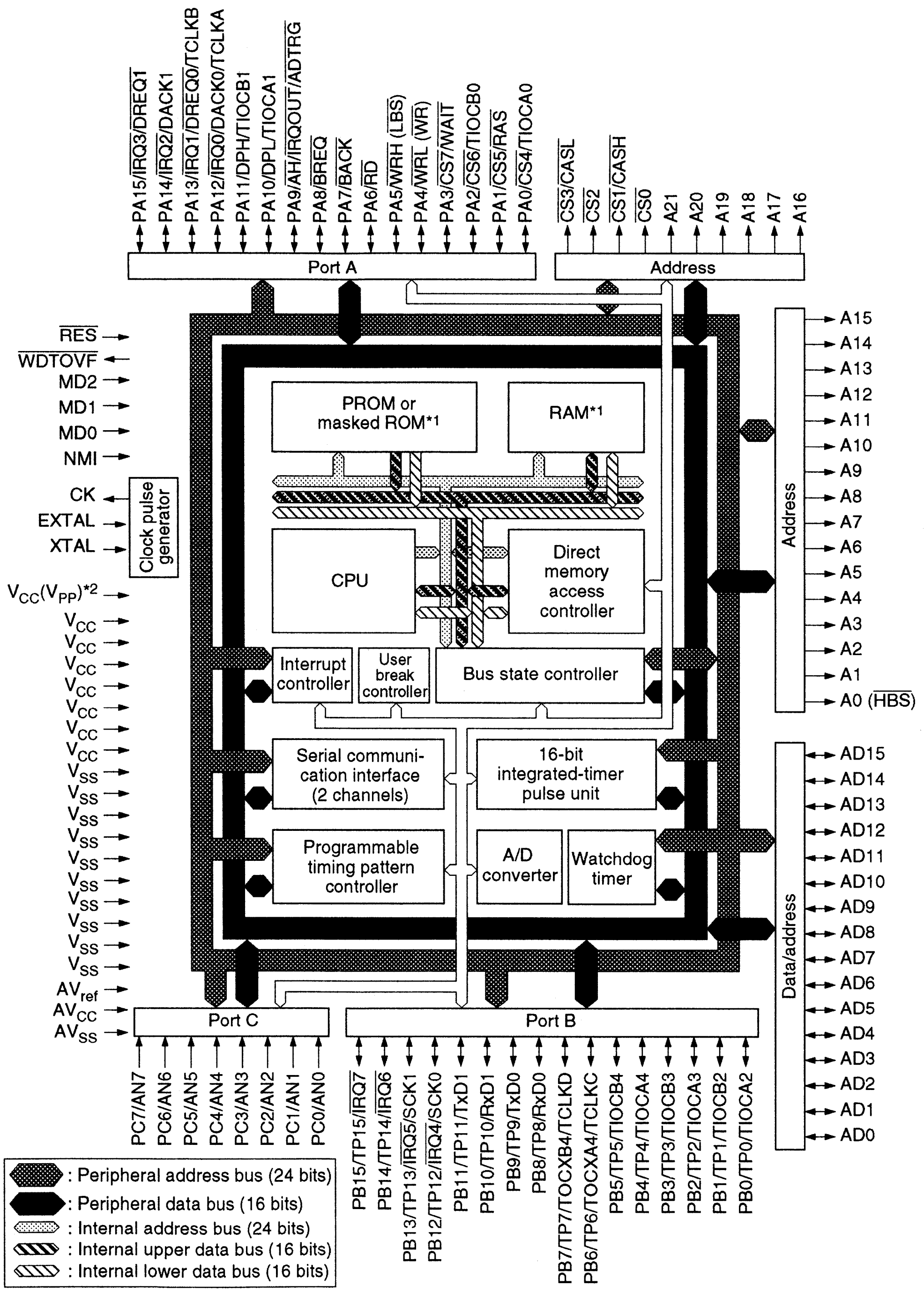
Table 2.3 Pin Functions

Type	Symbol	I/O	Name and Function
Power	V_{CC}	Input	Power
	V_{SS}	Input	Ground
	V_{PP}	Input	PROM programming power supply [SH7034]
Clock	EXTAL	Input	External clock
	XTAL	Input	Crystal
	CK	Output	System clock
System control	$\overline{\text{RES}}$	Input	Reset
	$\overline{\text{WDTOVF}}$	Output	Watchdog timer overflow
	$\overline{\text{BREQ}}$	Input	Bus right request
	$\overline{\text{BACK}}$	Output	Bus right request acknowledge
Operating mode control	MD2, MD1, MD0,	Input	Mode select
Interrupts	NMI	Input	Nonmaskable interrupt
	$\overline{\text{IRQ0}}$ to $\overline{\text{IRQ7}}$	Input	Interrupt requests 0 to 7
	$\overline{\text{IRQOUT}}$	Output	Interrupt request output
Address bus	A21 to A0	Output	Address bus
Data bus	AD15 to AD0	Input/output	Data bus
	DPH	Input/output	Upper data bus parity
	DPL	Input/output	Lower data bus parity
Bus control	$\overline{\text{WAIT}}$	Input	Wait
	$\overline{\text{RAS}}$	Output	Row address strobe
	$\overline{\text{CASH}}$	Output	Column address strobe high
	$\overline{\text{CASL}}$	Output	Column address strobe low
	$\overline{\text{RD}}$	Output	Read
	$\overline{\text{WRH}}$	Output	Upper write
	$\overline{\text{WRL}}$	Output	Lower write
	$\overline{\text{CS0}}$ to $\overline{\text{CS7}}$	Output	Chip selects 0 to 7
	$\overline{\text{AH}}$	Output	Address hold
	$\overline{\text{HBS}}$, $\overline{\text{LBS}}$	Output	Upper/lower byte strobe
	$\overline{\text{WR}}$	Output	Write

Table 2.3 Pin Functions (cont)

Type	Symbol	I/O	Name and Function
DMAC	$\overline{\text{DREQ0}}$, $\overline{\text{DREQ1}}$	Input	DMA transfer request (channels 0 and 1)
	DACK0, DACK1	Output	DMA transfer acknowledge (channels 0 and 1)
16-bit integrated-timer pulse unit (ITU)	TIOCA0 to TIOCA4, TIOCB0 to TIOCB4	Input/output	ITU input capture/output compares
	TOCXA4, TOCXB4	Output	ITU output compares (channel 4)
	TCLKA to TCLKD	Input	ITU timer clock inputs
Programmable timing pattern controller (TPC)	TP15 to TP0	Output	Timing pattern outputs
Serial communication interface (SCI)	TxD0, TxD1	Output	Transmit data (channels 0 and 1)
	RxD0, RxD1	Input	Receive data (channels 0 and 1)
	SCK0, SCK1	Input/output	Serial clock (channels 0 and 1)
A/D converter (SH7032, SH7034)	AN7 to AN0	Input	Analog input
	$\overline{\text{ADTRG}}$	Input	A/D convert trigger input
	AV_{ref}	Input	Analog reference power supply
	AV_{CC}	Input	Analog power supply
	AV_{SS}	Input	Analog ground
I/O ports	PA15 to PA0	Input/output	Port A
	PB15 to PB0	Input/output	Port B
	PC7 to PC0	Input	Port C [SH7032, SH7034]

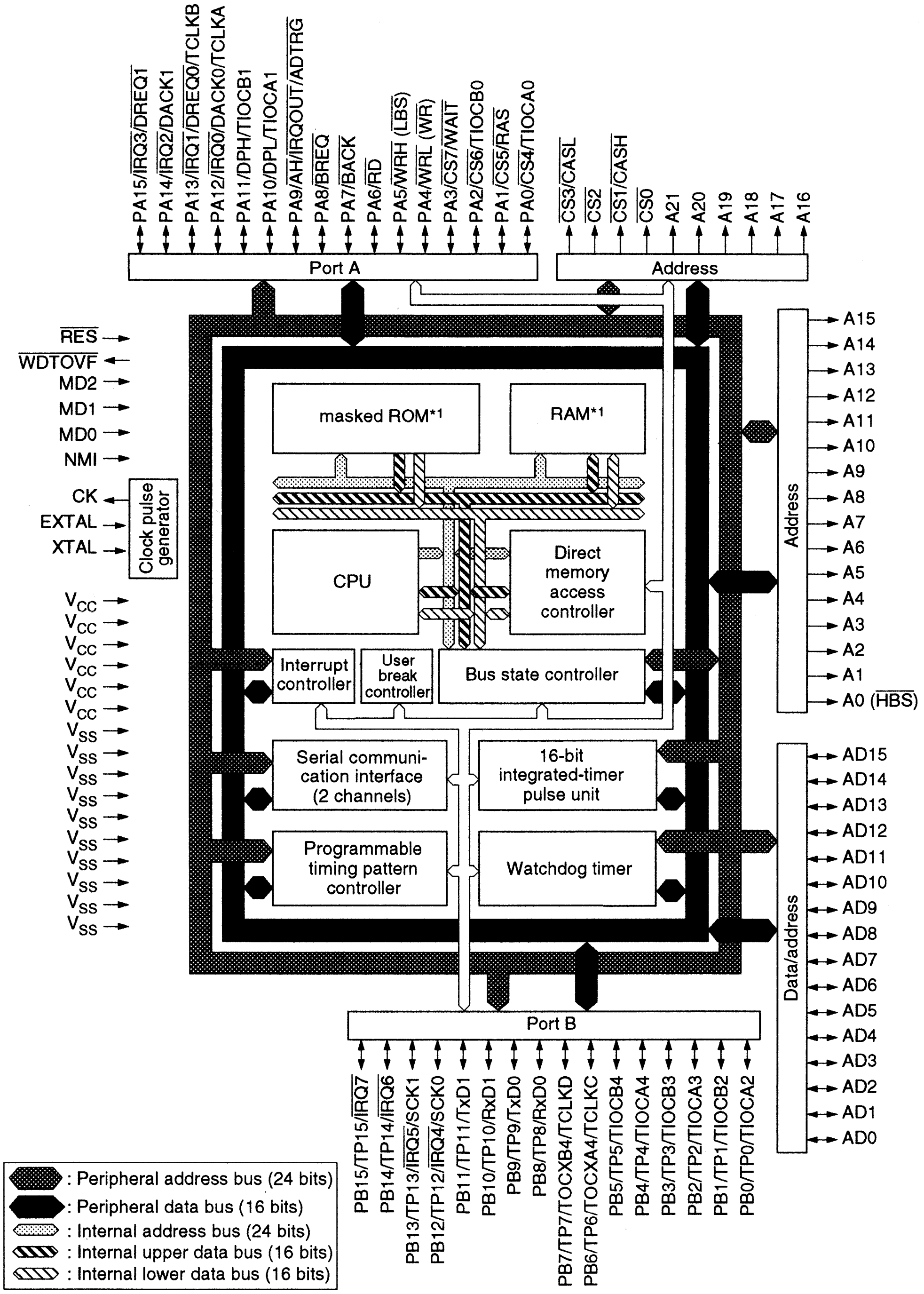
Section 3 Functional Block Diagram



Notes: 1. SH7032 (ROMless version): 8-kbyte RAM
SH7034 (onchip ROM version): 64-kbyte electrically programmable ROM or masked ROM, and 4-kbyte RAM

2. V_{CC}: SH7032
V_{PP}: SH7034

Figure 3.1 SH7032/34 Functional Block Diagram



Notes: SH7020: 16-kbyte masked ROM and 1-kbyte RAM
SH7021: 32-kbyte masked ROM and 1-kbyte RAM

Figure 3.2 SH7020/21 Functional Block Diagram

Section 4 CPU

4.1 Instruction Features

All instructions are RISC. Their features are as follows:

16-Bit Fixed Length: Every instruction is 16 bits long, making program coding substantially more efficient.

One Instruction/Cycle: Basic instructions can be executed in one cycle using the pipeline system. Instructions are executed in 50 ns at 20 MHz.

Data Length: Longword (32 bits) is the standard data length for all operations. Memory can be accessed in bytes, words, or longwords. Byte or word data accessed from memory is sign-extended and handled as longword data. Immediate data is sign-extended for arithmetic operations and zero-extended for logic operations. Immediate data is handled as longword data.

Load/Store Architecture: Basic operations are executed between registers. For operations that involve memory, data is loaded to the registers and executed (load-store architecture). Instructions such as AND that manipulate bits, however, are executed directly in memory.

Delayed Branch Instructions: Unconditional branch instructions are delayed. Pipeline disruption during branching is reduced by first executing the instruction that follows the branch instruction, and then branching. For example:

```
BRA     TRGET
ADD     R1,R0           ;Executes an ADD before branching to TRGET
```

Multiplication/Accumulation Operation: The 5-stage pipeline system and the onchip multiplier enable 16-bit × 16-bit → 32-bit multiplication operations to be executed in 1 to 3 cycles. 16-bit × 16-bit + 42-bit → 42-bit multiplication/accumulation operations can be executed in 2 to 3 cycles.

T Bit: T bit in the status register (SR) changes according to the result of the comparison, and in turn is the condition (true/false) that determines if the program will branch. The number of instructions that alter the T bit is kept to a minimum to improve the processing speed. Example:

```
ADD     #-1,R0          ;T bit is not changed by ADD
CMP/EQ  #0,R0           ;T bit is set when R0 = 0
BT      TRGET           ;The program branches to TRGET when T bit is set
                         (e.g., R0 = 0)
```

Immediate Data: Byte immediate data is located in the instruction code. Word or longword immediate data is not located in instruction codes but is stored in a memory table. The memory table is accessed by an immediate data transfer instruction (MOV) using the PC relative addressing mode with displacement. Example:

```
MOV      #H'12,R0          ;Byte immediate data transfer (transfer data =
                            H'12)
MOV.W    @(disp,PC),R0     ;Word immediate data transfer (transfer data =
                            H'1234)
.....................
.DATA.W H'1234
```

Absolute Address: When data is accessed by absolute address, the value already in the absolute address is placed in the memory table. By loading the immediate data when the instruction is executed, that value is transferred to the register and the data is accessed in the indirect register addressing mode.

16-Bit/32-Bit Displacement: When data is accessed by 16-bit or 32-bit displacement, the pre-existing displacement value is placed in the memory table. By loading the immediate data when the instruction is executed, that value is transferred to the register and the data is accessed in the indirect indexed register addressing mode.

4.2 Register Configuration

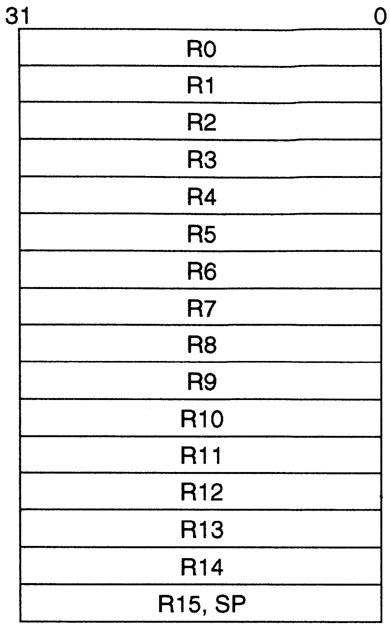
The register set consists of sixteen 32-bit general registers, three 32-bit control registers, and four 32-bit system registers. Table 4.1 lists register initial values.

Table 4.1 Initial Values of Registers

Classification	Register	Initial Value
General register	R0–R14	Undefined
	R15 (SP)	Value of the stack pointer in the vector address table
Control register	SR	Bits I0–I3 are 1111(H'F), reserved bits are 0, and other bits are undefined
	GBR	Undefined
	VBR	H'00000000
System register	MACH, MACL, PR	Undefined
	PC	Value of the program counter in the vector address table

4.2.1 General Registers (Rn)

General registers Rn consist of sixteen 32-bit registers R0–R15 (figure 4.1). General registers are used for data processing and address calculation. Register R0 functions also as an index register. For some instructions, R0 is a usable register. Register R15 functions as a stack pointer (SP) to save or recover status registers (SR) and program counter (PC) during exception processing.



R0 functions as an index register in the indirect indexed register addressing mode and indirect indexed GBR addressing mode. In some instructions, R0 functions as a source register or a destination register.

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

Figure 4.1 General Registers

4.2.2 Control Registers

Control registers consist of the 32-bit status register (SR), global base register (GBR), and vector base register (VBR). The status register indicates processing states. The global base register functions as a base address for the indirect GBR addressing mode to transfer data to the registers of peripheral onchip modules. The vector base register functions as the base address of the exception processing vector area including interrupts. Figure 4.2 shows control registers.

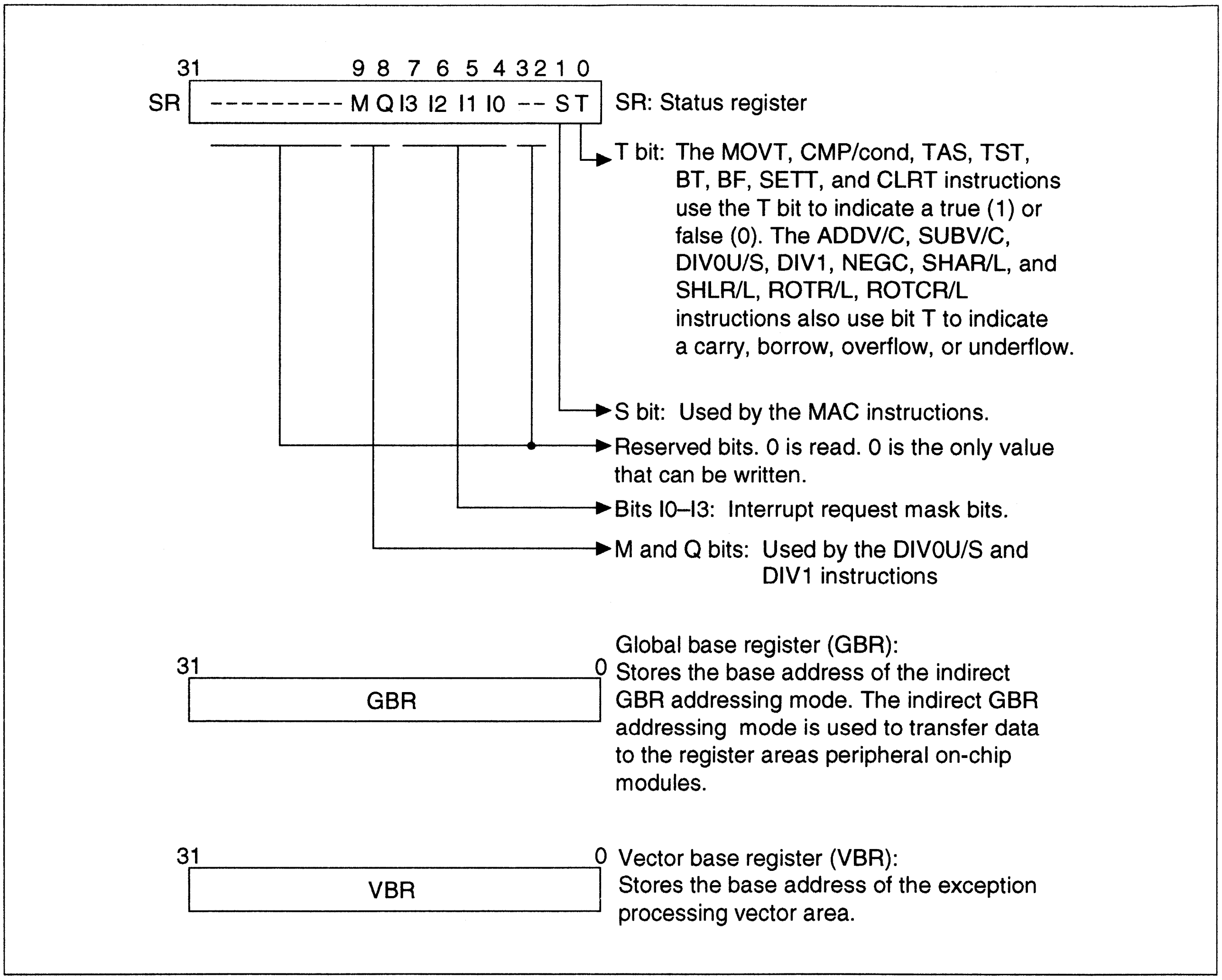
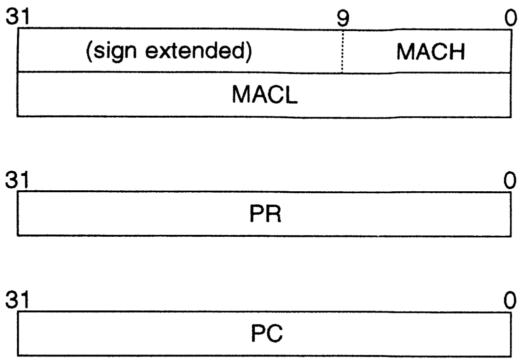

31
9 8 7 6 5 4 3 2 1 0
SR
--------- M Q I3 I2 I1 I0 -- S T
SR: Status register
T bit: The MOVT, CMP/cond, TAS, TST, BT, BF, SETT, and CLRT instructions use the T bit to indicate a true (1) or false (0). The ADDV/C, SUBV/C, DIV0U/S, DIV1, NEGC, SHAR/L, and SHLR/L, ROTR/L, ROTCR/L instructions also use bit T to indicate a carry, borrow, overflow, or underflow.
S bit: Used by the MAC instructions.
Reserved bits. 0 is read. 0 is the only value that can be written.
Bits I0–I3: Interrupt request mask bits.
M and Q bits: Used by the DIV0U/S and DIV1 instructions
31
0
GBR
Global base register (GBR): Stores the base address of the indirect GBR addressing mode. The indirect GBR addressing mode is used to transfer data to the register areas peripheral on-chip modules.
31
0
VBR
Vector base register (VBR): Stores the base address of the exception processing vector area.

Figure 4.2 Control Registers

4.2.3 System Registers

System registers (figure 4.3) consist of four 32-bit registers: multiply and accumulate registers high and low (MACH and MACL), procedure register (PR), and program counter (PC). The multiply and accumulate registers store the results of multiply and accumulate operations. The procedure register stores the return address from the subroutine procedure. The program counter stores program addresses to control the flow of the processing.



Multiply and accumulate registers high and low (MACH/L): Store the results of multiply and accumulate operations. MACH is sign-extended when read because only the lowest 10 bits are valid.

Procedure register (PR): Stores a return address from a subroutine procedure.

Program counter (PC): Indicates the fourth byte (second instruction) after the current instruction.

Figure 4.3 System Registers

4.3 Data Formats

4.3.1 Data Format in Registers

Register operands are always longwords (figure 4.4). When the memory operand is only a byte (8 bits) or a word (16 bits), it is sign-extended into a longword when stored into a register.

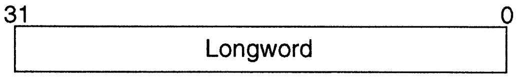


Figure 4.4 Longword Data Format

4.3.2 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 you try to access word data starting from an address other than 2n or longword data starting from an address other than 4n. In such cases, the data accessed cannot be guaranteed. The hardware stack area, which is referred to by the hardware stack pointer (SP, R15), uses only longword data starting from address 4n because this area stores the program counter and status register. Figure 4.5 shows memory data format.

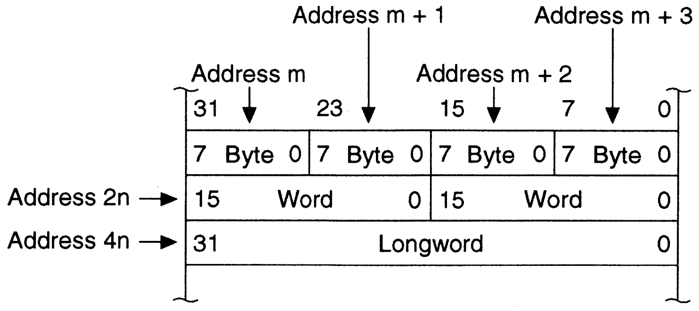


Figure 4.5 Memory Data Format

4.3.3 Immediate Data Format

Byte immediate data is located in instruction code. Immediate data accessed by the MOV, ADD, and CMP/EQ instructions is sign-extended and is handled in registers as longword data. Immediate data accessed by the TST, AND, OR, and XOR instructions is zero-extended and is handled as longword data. Consequently, AND instructions with immediate data always clear the upper 24 bits of the destination register.

Word or longword immediate data is not located in instruction code but rather is stored in a memory table. The memory table is accessed by a immediate data transfer instruction (MOV) using the PC relative addressing mode with displacement.

4.4 Addressing Modes

Addressing modes and effective address calculation are described in table 4.2.

Table 4.2 Addressing Modes and Effective Addresses Method

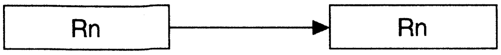
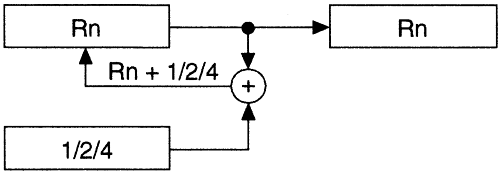
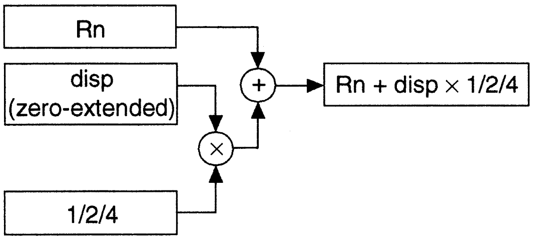
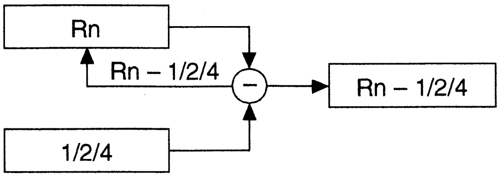
Addressing Mode	Mnemonic Expression	Effective Addresses Calculation	Equation
Direct register	Rn	The effective address is register Rn. (The operand is the contents of register Rn.)	—
Indirect register	@Rn	The effective address is the content of register Rn.  	Rn
Post-increment indirect register	@Rn+	The effective address is the content of register Rn. A constant is added to the content of Rn after the instruction is executed. 1 is added for a byte operation, 2 for a word operation, and 4 for a longword operation.  	Rn After the instruction is executed Byte: Rn + 1 → Rn Word: Rn + 2 → Rn Longword: Rn + 4 → Rn
Indirect register addressing with displacement	@(disp:4, Rn)	The effective address is Rn plus a 4-bit displacement (disp). disp is zero-extended, and remains the same for a byte operation, is doubled for a word operation, and is quadrupled for a longword operation.  Rn disp (zero-extended) + Rn + disp × 1/2/4 × 1/2/4	Byte: Rn + disp Word: Rn + disp × 2 Longword: Rn + disp × 4
Pre-decrement indirect register	@–Rn	The effective address is the value obtained by subtracting a constant from Rn. 1 is subtracted for a byte operation, 2 for a word operation, and 4 for a longword operation.  	Byte: Rn – 1 → Rn Word: Rn – 2 → Rn Longword: Rn – 4 → Rn Instruction executed with Rn after calculation

Table 4.2 Addressing Modes and Effective Addresses Method (cont)

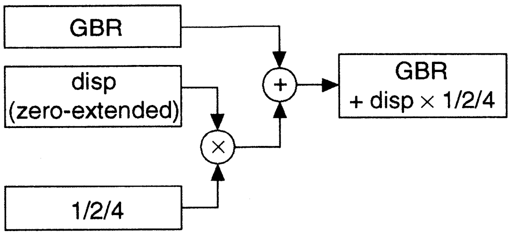
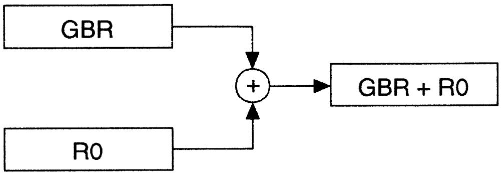
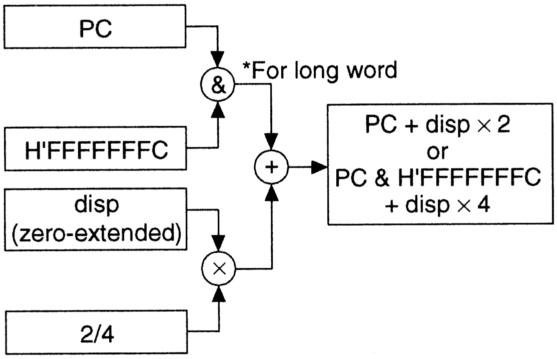
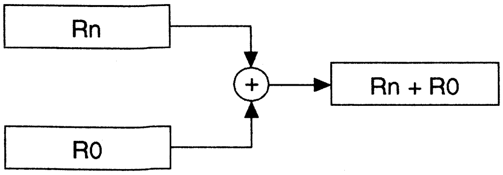
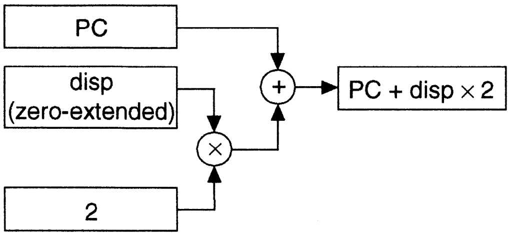
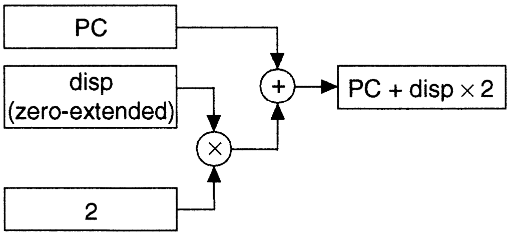
Addressing Mode	Mnemonic Expression	Effective Addresses Calculation	Equation
Indirect GBR addressing with displacement	@(disp:8, GBR)	The effective address is the GBR value plus an 8-bit displacement (disp). disp is zero-extended, and remains the same for a byte operation, is doubled for a word operation, and is quadrupled for a longword operation.  GBR disp (zero-extended) 1/2/4 × + GBR + disp × 1/2/4	Byte: GBR + disp Word: GBR + disp × 2 Longword: GBR + disp × 4
Indirect indexed GBR	@(R0, GBR)	The effective address is the GBR value plus the R0.  	GBR + R0
Indirect PC relative addressing with displacement	@(disp:8, PC)	The effective address is the PC value plus an 8-bit displacement (disp). disp is zero-extended, and remains the same for a byte operation, is doubled for a word operation, and is quadrupled for a longword operation. For a longword operation, the lowest 2 bits of the PC are masked.  PC & *For long word H'FFFFFFFC disp (zero-extended) 2/4 × + PC + disp × 2 or PC & H'FFFFFFFC + disp × 4	Word: PC + disp × 2 Longword: PC&H'FFFFFFFC + disp × 4

Table 4.2 Addressing Modes and Effective Addresses Method (cont)

Addressing Mode	Mnemonic Expression	Effective Addresses Calculation	Equation
Indirect indexed register	@(R0, Rn)	The effective address is the Rn value plus R0. 	Rn + R0
PC relative	disp:8	The effective address is the PC value sign-extended with an 8-bit displacement (disp), doubled, and added to the PC. 	PC + disp × 2
	disp:12	The effective address is the PC value sign-extended with a 12-bit displacement (disp), doubled, and added to the PC. 	PC + disp × 2
Immediate	#imm:8	The 8-bit immediate data (imm) for the TST, AND, OR, and XOR instructions are zero-extended.	—
	#imm:8	The 8-bit immediate data (imm) for the MOV, ADD, and CMP/EQ instructions are sign-extended.	—
	#imm:8	Immediate data (imm) for the TRAPA instruction is zero-extended and is quadrupled.	—

4.5 Instruction Set

Tables 4.3 and 4.5 through 4.10 list the SH7000 instruction set. Table 4.4 lists the format for instruction codes, operation, and execution states.

Table 4.3 Instruction Set by Classification

Classification	Operation Code	Function	Types
Data transfer	MOV	Data transfer, Immediate data transfer, Peripheral module data transfer, Structure data transfer	5
	MOVA	Effective address transfer	
	MOVT	T-bit transfer	
	SWAP	Swap of upper and lower bytes	
	XTRCT	Extraction of the middle of connected registers	
Arithmetic operations	ADD	Binary addition	17
	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	
	EXTS	Sign extension	
	EXTU	Zero extension	
	MAC	Multiplication and accumulation	
	MULS	Signed multiplication	
	MULU	Unsigned multiplication	
	NEG	Negation	
	NEGC	Negation with borrow	
	SUB	Binary subtraction	
	SUBC	Binary subtraction with borrow	
	SUBV	Binary subtraction with underflow check	
Logic operations	AND	Logical AND	6
	NOT	Bit inversion	
	OR	Logical OR	
	TAS	Memory test and bit set	
	TST	Logical AND and T bit set	
	XOR	Exclusive OR	

Table 4.3 Instruction Set by Classification (cont)

Classification	Operation Code	Function	Types
Shift instructions	ROTL	One-bit left rotation	10
	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 instructions	BF	Conditional branch (T = 0)	7
	BT	Conditional branch (T = 1)	
	BRA	Unconditional branch	
	BSR	Branch to subroutine procedure	
	JMP	Unconditional branch	
	JSR	Branch to subroutine procedure	
	RTS	Return from subroutine procedure	
System control	CLRT	T bit clear	11
	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:	56

Table 4.4 Instructions, Operations, and Execution State Format

Instruction	Instruction Code	Operation	Cycles	T bit
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*2	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-bit shift left >>n: n-bit shift right	Value when no wait states are inserted*1	Value of T bit after instruction is executed —: No change

Note: 1. Instruction execution cycles: The execution cycles shown 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.

2. Scaling (×1,×2,×4) is performed according to the instruction operand size. See "SH7000/SH7600 Series Programming Manual" for details.

Table 4.5 Data Transfer Instructions

Instruction		Instruction Code	Operation	Cycles	T bit
MOV	#imm, Rn	1110nnnniiiiiiii	#imm → Sign extension → Rn	1	—
MOV.W	@(disp, PC), Rn	1001nnnndddddddd	(disp×2 + PC) → Sign extension → Rn	1	—
MOV.L	@(disp, PC), Rn	1101nnnndddddddd	(disp×4 + PC) → Rn	1	—
MOV	Rm, Rn	0110nnnnmmmm0011	Rm → Rn	1	—
MOV.B	Rm, @Rn	0010nnnnmmmm0000	Rm → (Rn)	1	—
MOV.W	Rm, @Rn	0010nnnnmmmm0001	Rm → (Rn)	1	—
MOV.L	Rm, @Rn	0010nnnnmmmm0010	Rm → (Rn)	1	—
MOV.B	@Rm, Rn	0110nnnnmmmm0000	(Rm) → Sign extension → Rn	1	—
MOV.W	@Rm, Rn	0110nnnnmmmm0001	(Rm) → Sign extension → Rn	1	—
MOV.L	@Rm, Rn	0110nnnnmmmm0010	(Rm) → Rn	1	—
MOV.B	Rm, @-Rn	0010nnnnmmmm0100	Rn–1 → Rn, Rm → (Rn)	1	—
MOV.W	Rm, @-Rn	0010nnnnmmmm0101	Rn–2 → Rn, Rm → (Rn)	1	—
MOV.L	Rm, @-Rn	0010nnnnmmmm0110	Rn–4 → Rn, Rm → (Rn)	1	—
MOV.B	@Rm+, Rn	0110nnnnmmmm0100	(Rm) → Sign extension → Rn, Rm + 1 → Rm	1	—

Table 4.5 Data Transfer Instructions (cont)

Instruction	Instruction Code	Operation	Cycles	T bit
MOV.W @Rm+, Rn	0110nnnnmmmm0101	(Rm) → Sign extension → Rn, Rm + 2 → Rm	1	—
MOV.L @Rm+, Rn	0110nnnnmmmm0110	(Rm) → Rn, Rm + 4→ Rm	1	—
MOV.B R0, @(disp, Rn)	10000000nnnndddd	R0 → (disp + Rn)	1	—
MOV.W R0, @(disp, Rn)	10000001nnnndddd	R0 → (disp×2 + Rn)	1	—
MOV.L Rm, @(disp, Rn)	0001nnnnmmmmdddd	Rm → (disp×4 + Rn)	1	—
MOV.B @(disp, Rm), R0	10000100mmmmdddd	(disp + Rm) → Sign extension → R0	1	—
MOV.W @(disp, Rm), R0	10000101mmmmdddd	(disp×2 + Rm) → Sign extension → R0	1	—
MOV.L @(disp, Rm), Rn	0101nnnnmmmmdddd	(disp×4 + Rm) → Rn	1	—
MOV.B Rm, @(R0, Rn)	0000nnnnmmmm0100	Rm → (R0 + Rn)	1	—
MOV.W Rm, @(R0, Rn)	0000nnnnmmmm0101	Rm → (R0 + Rn)	1	—
MOV.L Rm, @(R0, Rn)	0000nnnnmmmm0110	Rm → (R0 + Rn)	1	—
MOV.B @(R0, Rm), Rn	0000nnnnmmmm1100	(R0 + Rm) → Sign extension → Rn	1	—
MOV.W @(R0, Rm), Rn	0000nnnnmmmm1101	(R0 + Rm) → Sign extension → Rn	1	—
MOV.L @(R0, Rm), Rn	0000nnnnmmmm1110	(R0 + Rm) → Rn	1	—
MOV.B R0, @(disp, GBR)	11000000dddddddd	R0 → (disp + GBR)	1	—
MOV.W R0, @(disp, GBR)	11000001dddddddd	R0 → (disp×2 + GBR)	1	—
MOV.L R0, @(disp, GBR)	11000010dddddddd	R0 → (disp×4 + GBR)	1	—
MOV.B @(disp, GBR), R0	11000100dddddddd	(disp + GBR) → Sign extension → R0	1	—
MOV.W @(disp, GBR), R0	11000101dddddddd	(disp×2 + GBR) → Sign extension → R0	1	—
MOV.L @(disp, GBR), R0	11000110dddddddd	(disp×4 + GBR) → R0	1	—
MOVA @(disp, PC), R0	11000111dddddddd	disp×4 + PC → R0	1	—

Table 4.6 Arithmetic Instructions

Instruction	Instruction Code	Operation	Cycles	T Bit
MOVT Rn	0000nnnn00101001	T → Rn	1	—
SWAP.B Rm, Rn	0110nnnnmmmm1000	Rm → Swap the bottom 2 bytes → REG	1	—
SWAP.W Rm, Rn	0110nnnnmmmm1001	Rm → Swap consecutive words → Rn	1	—
XTRCT Rm, Rn	0010nnnnmmmm1101	Center 32 bits of Rm and Rn → Rn	1	—
ADD Rm, Rn	0011nnnnmmmm1100	Rn + Rm → Rn	1	—
ADD #imm, Rn	0111nnnniiiiiiii	Rn + imm → Rn	1	—
ADDC Rm, Rn	0011nnnnmmmm1110	Rn + Rm + T → Rn, Carry → T	1	Carry
ADDV Rm, Rn	0011nnnnmmmm1111	Rn + Rm → Rn, Overflow → T	1	Overflow
CMP/EQ #imm, R0	10001000iiiiiiii	If R0 = imm, 1 → T	1	Comparison result
CMP/EQ Rm, Rn	0011nnnnmmmm0000	If Rn = Rm, 1 → T	1	
CMP/HS Rm, Rn	0011nnnnmmmm0010	If Rn ≥ Rm with unsigned data, 1 → T	1	
CMP/GE Rm, Rn	0011nnnnmmmm0011	If Rn ≥ Rm with signed data, 1 → T	1	
CMP/HI Rm, Rn	0011nnnnmmmm0110	If Rn > Rm with unsigned data,	1	
CMP/GT Rm, Rn	0011nnnnmmmm0111	If Rn > Rm with signed data, 1 → T	1	
CMP/PZ Rn	0100nnnn00010001	If Rn ≥ 0, 1 → T	1	
CMP/PL Rn	0100nnnn00010101	If Rn > 0, 1 → T	1	
CMP/STR Rm, Rn	0010nnnnmmmm1100	If Rn and Rm have an equivalent byte, 1 → T	1	
DIV1 Rm, Rn	0011nnnnmmmm0100	Single-step division (Rn/Rm)	1	Calculation result
DIV0S Rm, Rn	0010nnnnmmmm0111	MSB of Rn → Q, MSB of Rm → M, M ^ Q → T	1	
DIV0U	0000000000011001	0 → M/Q/T	1	0
EXTS.B Rm, Rn	0110nnnnmmmm1110	A byte in Rm is sign-extended → Rn	1	—
EXTS.W Rm, Rn	0110nnnnmmmm1111	A word in Rm is sign-extended → Rn	1	—
EXTU.B Rm, Rn	0110nnnnmmmm1100	A byte in Rm is zero-extended → Rn	1	—
EXTU.W Rm, Rn	0110nnnnmmmm1101	A word in Rm is zero-extended → Rn	1	—
MAC.W @Rm+, @Rn+	0100nnnnmmmm1111	Signed operation of (Rn) × (Rm) + MAC → MAC	3/(2)[*1]	—

Table 4.6 Arithmetic Instructions (cont)

Instruction		Instruction Code	Operation	Cycles	T Bit
MULS	Rm, Rn	0010nnnnmmmm1111	Signed operation of Rn × Rm → MAC	1(–3)*2	—
MULU	Rm, Rn	0010nnnnmmmm1110	Unsigned operation of Rn × Rm → MAC	1(–3)*2	—
NEG	Rm, Rn	0110nnnnmmmm1011	0–Rm → Rn	1	—
NEGC	Rm, Rn	0110nnnnmmmm1010	0 – Rm–T → Rn, Borrow → T	1	Borrow
SUB	Rm, Rn	0011nnnnmmmm1000	Rn – Rm → Rn	1	—
SUBC	Rm, Rn	0011nnnnmmmm1010	Rn – Rm–T → Rn, Borrow → T	1	Borrow
SUBV	Rm, Rn	0011nnnnmmmm1011	Rn – Rm → Rn, Underflow → T	1	Underflow

Notes: 1. Single instruction execution normally requires 3 cycles; when instructions are executed consecutively, instructions after the first require only 2 cycles.

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 4.7 Logic Operation Instructions

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

Table 4.8 Shift Instructions

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

Table 4.9 Branch Instructions

Instruction		Instruction Code	Operation	Cycles	T Bit
BF	label	10001011dddddddd	If T = 0, disp×2 + PC → PC; if T = 1, nop	3/1*	—
BT	label	10001001dddddddd	If T = 1, disp×2 + PC → PC; if T = 0, nop	3/1*	—
BRA	label	1010dddddddddddd	Delayed branch, disp×2 + PC → PC	2	—
BSR	label	1011dddddddddddd	Delayed branch, PC → PR, disp×2 + PC → PC	2	—
JMP	@Rn	0100nnnn00101011	Delayed branch, Rn → PC	2	—
JSR	@Rn	0100nnnn00001011	Delayed branch, PC → PR, Rn → PC	2	—
RTS		0000000000001011	Delayed branch, PR → PC	2	—

Note: The execution state is 3 cycles when program branches, and 1 cycle when program does not branch.

Table 4.10 System Control Instructions

Instruction		Instruction Code	Operation	Cycles	T Bit
CLRT		0000000000001000	0 → T	1	0
CLRMAC		0000000000101000	0 → MACH, MACL	1	—
LDC	Rm, SR	0100mmmm00001110	Rm → SR	1	LSB
LDC	Rm, GBR	0100mmmm00011110	Rm → GBR	1	—
LDC	Rm, VBR	0100mmmm00101110	Rm → VBR	1	—
LDC.L	@Rm+, SR	0100mmmm00000111	(Rm) → SR, Rm + 4 → Rm	3	LSB
LDC.L	@Rm+, GBR	0100mmmm00010111	(Rm) → GBR, Rm + 4 → Rm	3	—
LDC.L	@Rm+, VBR	0100mmmm00100111	(Rm) → VBR, Rm + 4 → Rm	3	—
LDS	Rm, MACH	0100mmmm00001010	Rm → MACH	1	—
LDS	Rm, MACL	0100mmmm00011010	Rm → MACL	1	—
LDS	Rm, PR	0100mmmm00101010	Rm → PR	1	—
LDS.L	@Rm+, MACH	0100mmmm00000110	(Rm) → MACH, Rm + 4 → Rm	1	—
LDS.L	@Rm+, MACL	0100mmmm00010110	(Rm) → MACL, Rm + 4 → Rm	1	—
LDS.L	@Rm+, PR	0100mmmm00100110	(Rm) → PR, Rm + 4 → Rm	1	—
NOP		0000000000001001	No operation	1	—
RTE		0000000000101011	Delayed branch, stack area → PC/SR	4	—
SETT		0000000000011000	1 → T	1	1
SLEEP		0000000000011011	Sleep	3*	—
STC	SR, Rn	0000nnnn00000010	SR → Rn	1	—
STC	GBR, Rn	0000nnnn00010010	GBR → Rn	1	—
STC	VBR, Rn	0000nnnn00100010	VBR → Rn	1	—
STC.L	SR, @-Rn	0100nnnn00000011	Rn – 4 → Rn, SR → (Rn)	2	—
STC.L	GBR, @-Rn	0100nnnn00010011	Rn – 4 → Rn, GBR → (Rn)	2	—
STC.L	VBR, @-Rn	0100nnnn00100011	Rn – 4 → Rn, VBR → (Rn)	2	—
STS	MACH, Rn	0000nnnn00001010	MACH → Rn	1	—
STS	MACL, Rn	0000nnnn00011010	MACL → Rn	1	—
STS	PR, Rn	0000nnnn00101010	PR → Rn	1	—
STS.L	MACH, @-Rn	0100nnnn00000010	Rn – 4 → Rn, MACH → (Rn)	1	—
STS.L	MACL, @-Rn	0100nnnn00010010	Rn – 4 → Rn, MACL → (Rn)	1	—
STS.L	PR, @-Rn	0100nnnn00100010	Rn – 4 → Rn, PR → (Rn)	1	—
TRAPA	#imm	11000011iiiiiiii	PC/SR → stack area, (imm×4+VBR) → PC	8	—

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

The above 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.

4.6 Operating Modes

The operating modes are set by the mode setting pins (MD2–MD0). Modes 0 and 1 disable the onchip ROM while mode 2 enables it. The PROM mode is for writing programming to the onchip PROM. Table 4.11 lists mode setting information.

Table 4.11 Setting Operating Modes

	Pin Settings					
Operating Mode	**MD2**	**MD1**	**MD0**	**Operation**	**Onchip ROM**	**Bus Width of Area 0**
Mode 0	0	0	0	MCU mode	Disabled	8 bits
Mode 1	0	0	1			16 bits
Mode 2*[1]	0	1	0		Enabled	—
Mode 3	0	1	1		Not usable (do not set)	
Mode 4	1	0	0			
Mode 5	1	0	1			
Mode 6	1	1	0			
Mode 7*[2]	1	1	1	PROM mode	—	—

Notes: 1. SH7034, SH7020, and SH7021 only.

2. SH7034 (EPROM version).

SH7034, SH7020 and SH7021 have onchip ROM. PROM is available only in the PROM version of the SH7034.

4.7 Processing States

The CPU has five processing states: reset, exception processing, bus right release, program execution, and power-down, shown in figure 4.6.

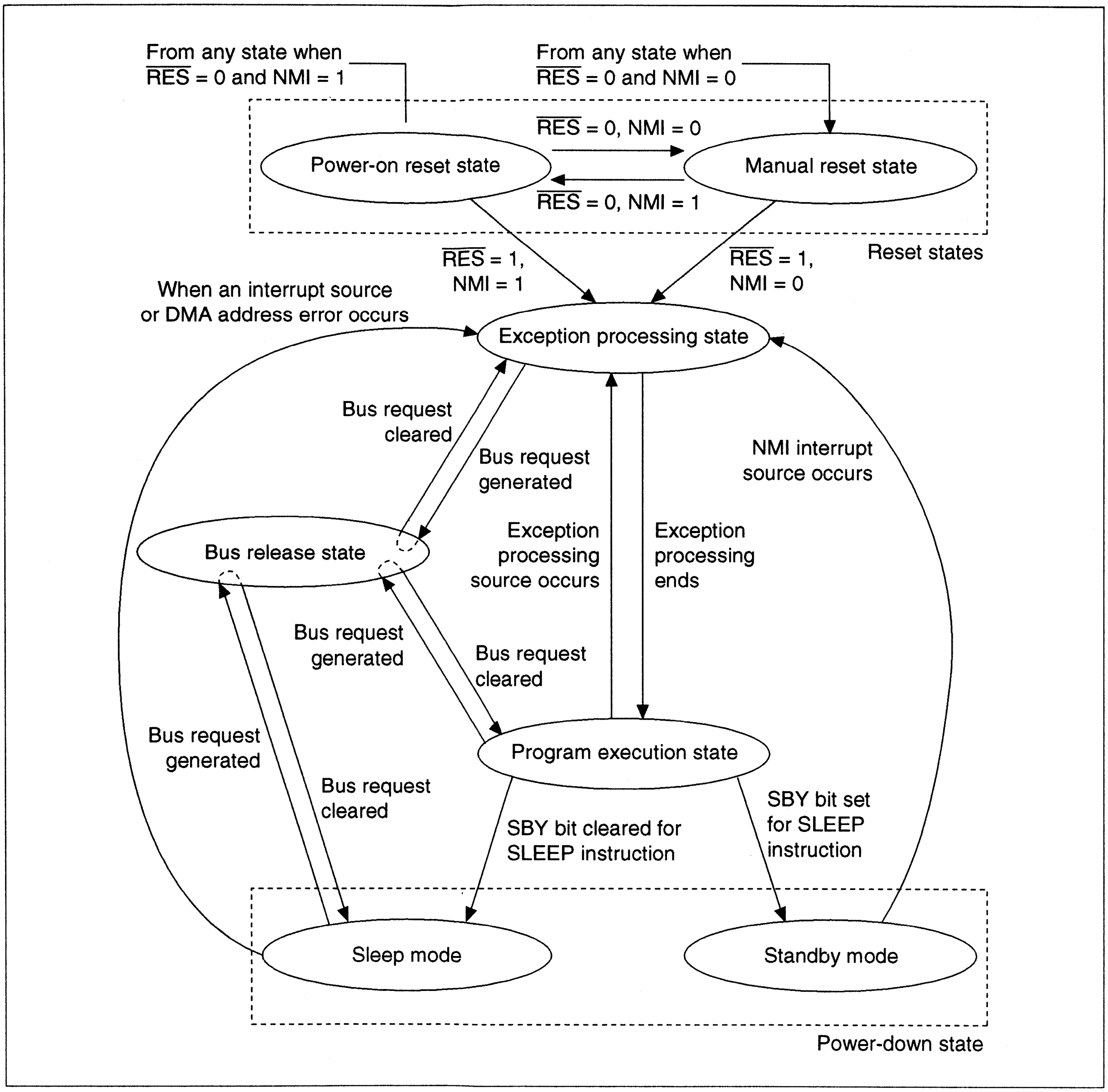

From any state when $\overline{\text{RES}}$ = 0 and NMI = 1
From any state when $\overline{\text{RES}}$ = 0 and NMI = 0
$\overline{\text{RES}}$ = 0, NMI = 0
Power-on reset state
Manual reset state
$\overline{\text{RES}}$ = 0, NMI = 1
Reset states
$\overline{\text{RES}}$ = 1, NMI = 1
$\overline{\text{RES}}$ = 1, NMI = 0
When an interrupt source or DMA address error occurs
Exception processing state
Bus request cleared
Bus request generated
NMI interrupt source occurs
Bus release state
Exception processing source occurs
Exception processing ends
Bus request generated
Bus request cleared
Bus request generated
Program execution state
Bus request cleared
SBY bit cleared for SLEEP instruction
SBY bit set for SLEEP instruction
Sleep mode
Standby mode
Power-down state

Figure 4.6 Transitions between Processing States

4.7.1 Reset State

CPU reset occurs when the $\overline{\text{RES}}$ pin level goes low. When the NMI pin is high, the result is a power-on reset; when it is low, a manual reset will occur.

In the power-on reset, all CPU internal states and onchip peripheral module registers are initialized. In a manual reset, all CPU internal states and onchip peripheral module registers, with the exception of the bus controller (BSC) and pin function controller (PFC), are initialized. In a manual reset, the BSC is not initialized, so the refresh operation will continue.

4.7.2 Exception Processing State

Exception processing is a transient state that occurs when the CPU's processing state flow is altered by exception processing sources such as resets or interrupts.

For a reset, the initial values of the program counter (PC) (execution start address) and stack pointer (SP) are fetched from the exception processing vector table and stored; the CPU then branches to the execution start address and execution of the program begins.

For an interrupt, the stack pointer (SP) is accessed and the program counter (PC) and status register (SR) are saved to the stack area. The exception service routine start address is fetched from the exception processing vector table; the CPU then branches to that address and the program starts executing, thereby entering the program execution state.

4.7.3 Program Execution State

In program execution state, the CPU sequentially executes the program.

4.7.4 Power-Down State

In power-down state, the CPU operation halts and power consumption declines. The SLEEP instruction places the CPU in the power-down state. This state has 2 modes: sleep mode and standby mode. Table 4.12 lists power down state parameters.

Table 4.12 Power-Down State

		State						
Mode	**Transition Conditions**	**Clock**	**CPU**	**Onchip Peripheral Modules**	**CPU Registers**	**On-chip RAM**	**I/O Port Pins**	**Canceling Method**
Sleep mode	Execute SLEEP instruction with SBY bit cleared in SBYCR	Run	Halt	Run	Held	Held	Held	1. Interrupt 2. DMA address error 3. Power-on reset 4. Manual reset
Standby mode	Execute SLEEP instruction with SBY bit set in SBYCR	Halt	Halt	Halt and initialize*	Held	Held	Held or high-Z* (selectable)	1. NMI interrupt 2. Power-on reset 3. Manual reset

Note: Differs depending on the peripheral module and pin.

4.7.5 Bus Right Release State

In bus right release state, the CPU releases rights to the bus to the device that has requested them.

4.8 Exception Processing

Exception processing occurs for resets, address errors, interrupts and instructions. When multiple exception processing requests occur simultaneously, they are accepted and processed according to a priority order. Tables 4.13 through 4.17 list exception processing information.

Table 4.13 Types of Exception Processing and Priority Orders

Priority	Type of Processing		Conditions to Start Exception Processing
High	Resets	Power on reset	When the NMI pin is high and the $\overline{\text{RES}}$ pin changes from low to high.
↑		Manual reset	When the NMI pin is low and the $\overline{\text{RES}}$ pin changes from low to high.
	Address errors	• CPU address error • DMA address error	When an address error occurs in the bus cycle of an instruction fetch or data read/write, it is detected when the instruction to be executed is decoded; exception processing then occurs instead of instruction execution. Exception processing starts only after all instructions executing at that time finish executing.
	Interrupts	• External NMI • User break • External IRQ • Onchip peripheral module	When an interrupt is requested, the request is detected when the instruction to be executed is decoded; exception processing then occurs instead of instruction execution. Exception processing starts only after all instructions executing at that time finish executing.
	Instructions	Trap instruction	Started by execution of a trap instruction (TRAPA).
		Ordinary illegal instruction	Starts when undefined code is decoded at a location other than a delay slot.*
↓ Low		Illegal slot instruction	Starts when undefined code at a location other than a delay slot* or an instruction that rewrites the PC is decoded.

Note: The instruction address location after the delay branch instruction is called the delay slot.

Table 4.14 Sources of Address Errors

Type	Generating Bus Master	Source
Instruction fetch	CPU	Instruction fetched from odd address
		Instruction fetched from onchip peripheral module register area
Data read/write	CPU or DMAC	Access of word data from outside word boundaries.
		Access of longword from outside longword boundaries.
		Access 8-bit register of onchip peripheral module register area with longword (no access error will occur if an 8-bit register is word accessed or a 16-bit register is longword accessed)

Table 4.15 Exception Sources Not Accepted

	Exception Source		
Location	Address Error	Interrupt	Remarks
Immediately after a delay branch instruction (delay slot)	X	X	Delay branch instructions: JMP, JSR, BRA, BSR, RTS, RTE
Immediately after interrupt-disabled instructions	O	X	Interrupt-disabled instructions: LDC, LDC.L, STC, STC.L, LDS, LDS.L, STS, STS.L

Note: O: Accepted

X: Not accepted (accepted after instructions that immediately follow a delay branch instruction are executed)

Table 4.16 Exception Processing Operation

Type of Exception Processing	Operation
Reset	1. Fetch initial value (execution start address) of program counter (PC) from exception processing vector table. 2. Fetch initial value of stack pointer (SP) from exception processing vector table. 3. Clear the vector register (VBR) to 0 and set the interrupt mask bits (I3–I0) in the status register (SR) to 1111 (binary). 4. Set the values fetched from the exception processing vector base tables to the PC and SP respectively and start program execution.
Address error, interrupt, instruction	1. Save SR to stack. 2. Save PC to stack. 3. Fetch the exception processing service routine start address from the exception processing vector table, jump to that address, and start executing the program. In this case, the jump is not a delay branch instruction.

Note: Interrupt exception processing is affected by the interrupt mask bits (I3–I0) of the status register. The interrupt is accepted when the values of the priority level set in them is greater than the interrupt mask bit values.

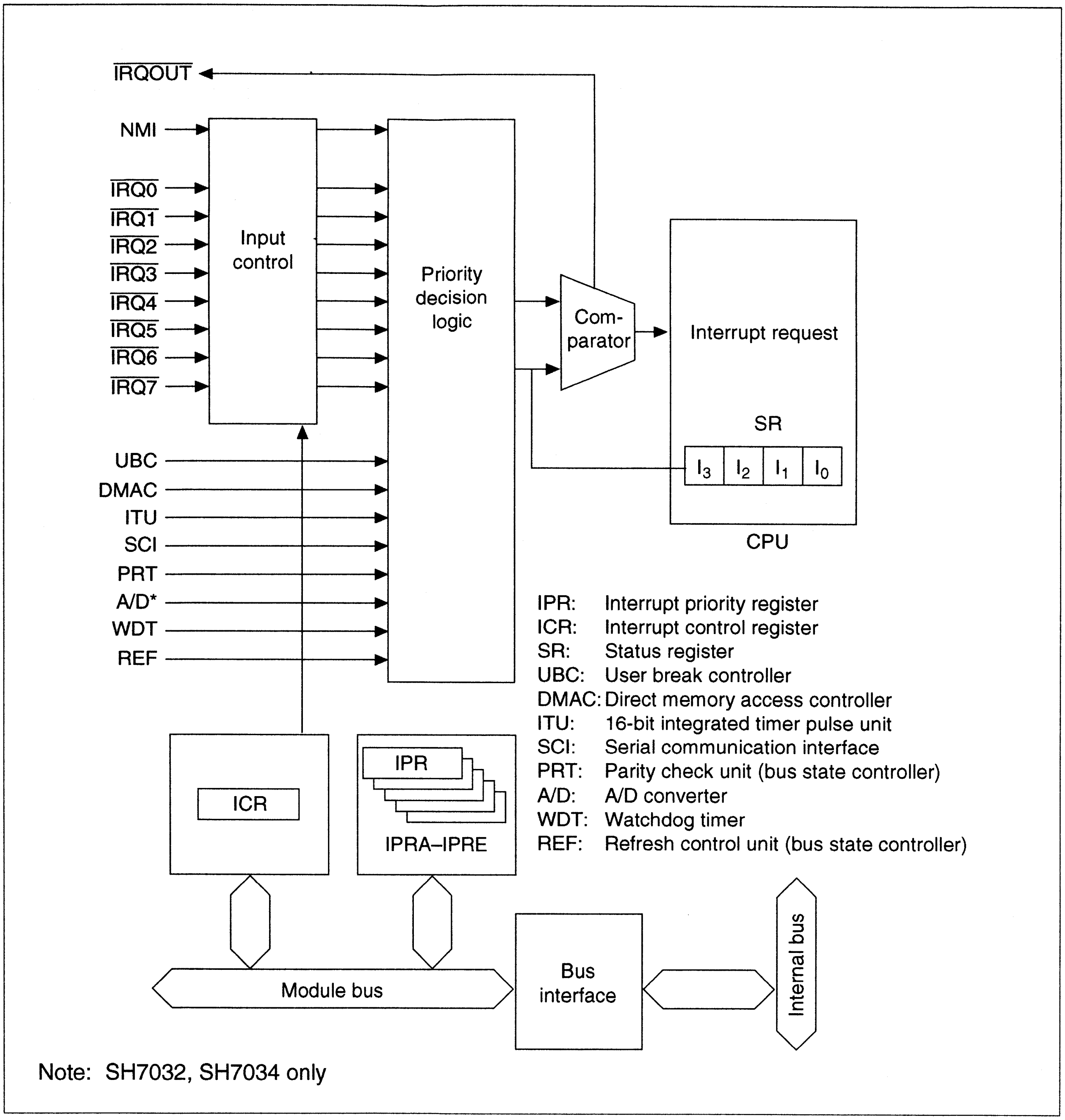
Table 4.17 Exception Processing Vector Table

Exception Processing Source		Vector Number	Vector Table Address Offset
Power on reset	PC	0	H'00000000–H'00000003
	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
(Reserved by system)		5	H'00000014–H'00000017
Illegal slot 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
(Reserved by system)		13–31	H'00000034–H'00000037 to H'0000007C–H'0000007F
Trap instruction (user vector)		32–63	H'00000080–H'00000083 to H'000000FC–H'000000FF
Interrupts	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
	Onchip peripheral module	72–255	H'00000120–H'00000123 to H'000003FC–H'000003FF

4.9 Interrupt Controller (INTC)

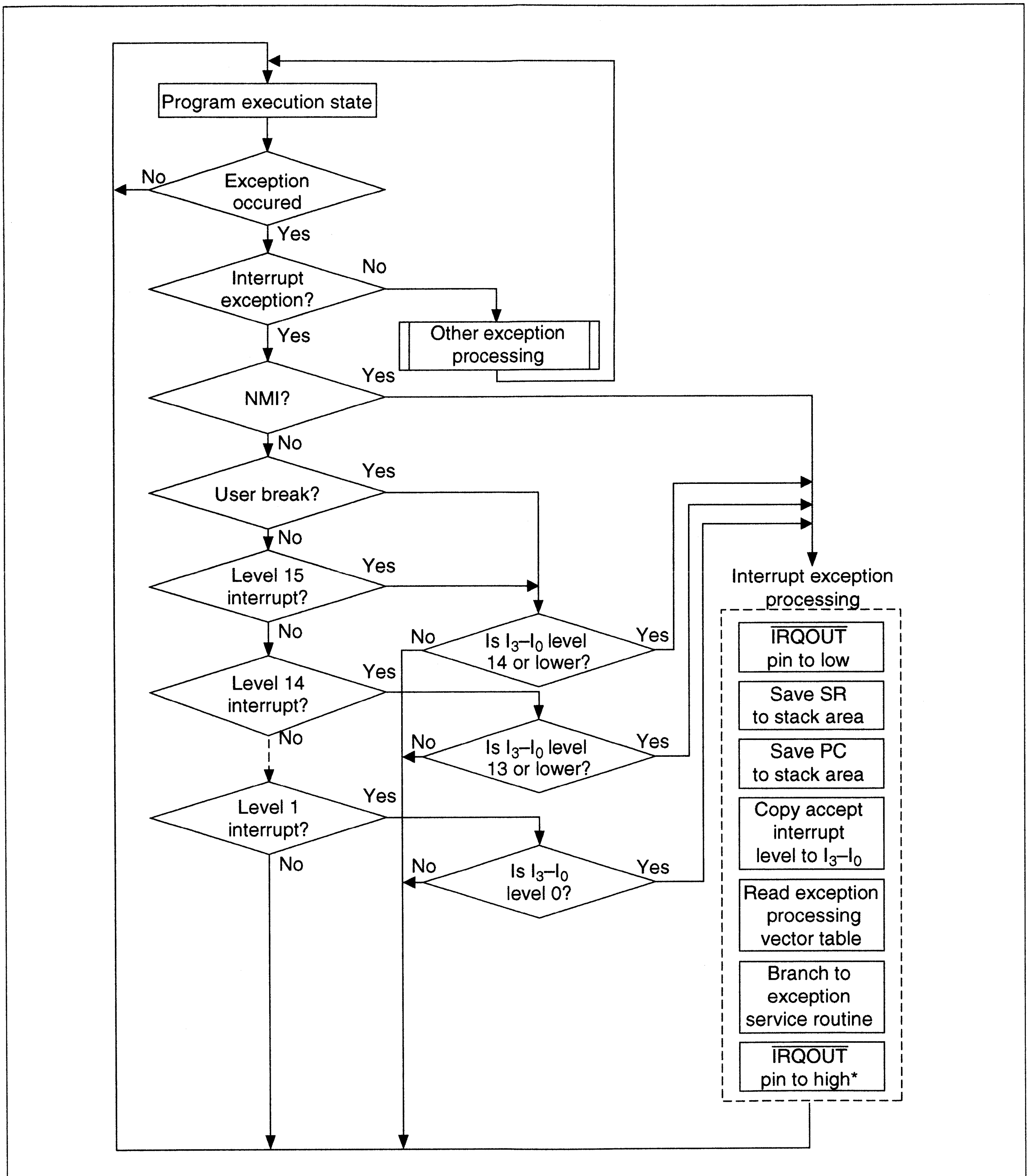
The interrupt controller (INTC) has registers that allow setting of priority levels for individual interrupt sources. These are used to control the priority order and process simultaneous interrupts. Figure 4.7 is a functional block diagram of the interrupt controller. Figure 4.8 shows interrupt execution processing flow. Table 4.18 lists interrupt sources and vector tables.

- The priority rankings of IRQ and onchip peripheral module interrupts can be selected in the range of 15–0 in the interrupt priority register (IPR). NMI is fixed at 16 and user break is fixed at 15.
- When multiple interrupt sources with the same priority occur at once, they are processed according to the default priority.
- NMI can be selected as either rising edge or falling edge triggered; IRQ input can be selected as low level or falling edge.
- The NMI level of external pins can be read.
- A signal can be output ($\overline{\text{IRQOUT}}$ pin) so that it can be externally determined when the interrupt controller has accepted an interrupt.


IRQOUT
NMI
IRQ0
IRQ1
IRQ2
IRQ3
IRQ4
IRQ5
IRQ6
IRQ7
Input control
Priority decision logic
Com-parator
Interrupt request
SR
I_3 I_2 I_1 I_0
CPU
UBC
DMAC
ITU
SCI
PRT
A/D*
WDT
REF
ICR
IPR
IPRA–IPRE
Module bus
Bus interface
Internal bus
IPR: Interrupt priority register
ICR: Interrupt control register
SR: Status register
UBC: User break controller
DMAC: Direct memory access controller
ITU: 16-bit integrated timer pulse unit
SCI: Serial communication interface
PRT: Parity check unit (bus state controller)
A/D: A/D converter
WDT: Watchdog timer
REF: Refresh control unit (bus state controller)

Note: SH7032, SH7034 only

Figure 4.7 Interrupt Controller Block Diagram



Note: IRQOUT stays low until the interrupt controller accepts another input.

Figure 4.8 Interrupt Exception Processing Flowchart

Table 4.18 Interrupt Sources Listed by Priority

Interrupt Source		Vector No.	Vector Table Address Offset	Interrupt Priority (Initial Value)	Corresponding IPR (Bit No.)	Priority within IPR Setting Unit
NMI		11	H'0000002C– H'0000002F	16	—	—
User break		12	H'00000030– H'00000033	15	—	—
IRQ0		64	H'00000100– H'00000103	0–15(0)	IPRA(15–12)	—
IRQ1		65	H'00000104– H'00000107	0–15(0)	IPRA(11–8)	—
IRQ2		66	H'00000108– H'0000010B	0–15(0)	IPRA(7–4)	—
IRQ3		67	H'0000010C– H'0000010F	0–15(0)	IPRA(3–0)	—
IRQ4		68	H'00000110– H'00000113	0–15(0)	IPRB(15–12)	—
IRQ5		69	H'00000114– H'00000117	0–15(0)	IPRB(11–8)	—
IRQ6		70	H'00000118– H'0000011B	0–15(0)	IPRB(7–4)	—
IRQ7		71	H'0000011C– H'0000011F	0–15(0)	IPRB(3–0)	—
DMAC0	DEI0	72	H'00000120– H'00000123	0–15(0)	IPRC(15–12)	3
	Reserved	73	H'00000124– H'00000127			2
DMAC1	DEI1	74	H'00000128– H'0000012B			1
	Reserved	75	H'0000012C– H'0000012F			0
DMAC2	DEI2	76	H'00000130– H'00000133	0–15(0)	IPRC(11–8)	3
	Reserved	77	H'00000134– H'00000137			2
DMAC3	DEI3	78	H'00000138– H'0000013B			1
	Reserved	79	H'0000013C– H'0000013F			0

Table 4.18 Interrupt Sources Listed by Priority (cont)

Interrupt Source		Vector No.	Vector Table Address Offset	Interrupt Priority (Initial Value)	Corresponding IPR (Bit No.)	Priority within IPR Setting Unit
ITU0	IMIA0	80	H'00000140–H'00000143	0–15(0)	IPRC(7–4)	3
	IMIB0	81	H'00000144–H'00000147			2
	OVI0	82	H'00000148–H'0000014B			1
	Reserved	83	H'0000014C–H'0000014F			0
ITU1	IMIA1	84	H'00000150–H'00000153	0–15(0)	IPRC(3–0)	3
	IMIB1	85	H'00000154–H'00000157			2
	OVI1	86	H'00000158–H'0000015B			1
	Reserved	87	H'0000015C–H'0000015F			0
ITU2	IMIA2	88	H'00000160–H'00000163	0–15(0)	IPRD(15–12)	3
	IMIB2	89	H'00000164–H'00000167			2
	OVI2	90	H'00000168–H'0000016B			1
	Reserved	91	H'0000016C–H'0000016F			0
ITU3	IMIA3	92	H'00000170–H'00000173	0–15(0)	IPRD(11–8)	3
	IMIB3	93	H'00000174–H'00000177			2
	OVI3	94	H'00000178–H'0000017B			1
	Reserved	95	H'0000017C–H'0000017F			0
ITU4	IMIA4	96	H'00000180–H'00000183	0–15(0)	IPRD(7-4)	3
	IMIB4	97	H'00000184–H'00000187			2
	OVI4	98	H'00000188–H'0000018B			1

Table 4.18 Interrupt Sources Listed by Priority (cont)

Interrupt Source		Vector No.	Vector Table Address Offset	Interrupt Priority (Initial Value)	Corresponding IPR (Bit No.)	Priority within IPR Setting Unit
ITU4 (cont)	Reserved	99	H'0000018C–H'0000018F	0–15(0)	IPRD(7–4)	0
SCI0	ERI0	100	H'00000190–H'00000193	0–15(0)	IPRD(3–0)	3
	RXI0	101	H'00000194–H'00000197			2
	TXI0	102	H'00000198–H'0000019B			1
	TEI0	103	H'0000019C–H'0000019F			0
SCI1	ERI1	104	H'000001A0–H'000001A3	0–15(0)	IPRE(15–12)	3
	RXI1	105	H'000001A4–H'000001A7			2
	TXI1	106	H'000001A8–H'000001AB			1
	TEI1	107	H'000001AC–H'000001AF			0
PRT	PEI	108	H'000001B0–H'000001B3	0–15(0)	IPRE(11–8)	3
A/D (Reserved in SH7020 and SH7021)	ADI	109	H'000001B4–H'000001B7			2
	Reserved	110	H'000001B8–H'000001BB			1
	Reserved	111	H'000001BC–H'000001BF			0
WDT	ITI	112	H'000001C0–H'000001C3	0–15(0)	IPRE(7-4)	3
REF	CMI	113	H'000001C4–H'000001C7			2
	Reserved	114	H'000001C8–H'000001CB			1
	Reserved	115	H'000001CC–H'000001CF			0
	Reserved	116 to 255	H'000001D0–H'000001D3 to H'000003FC–H'000003FF	—	—	—

4.10 User Break Controller (UBC)

The user break controller (UBC) requests user break interrupts of the CPU according to the contents of the bus cycle generated by the CPU or DMAC. This function can be used to construct a self-debugger to facilitate program debugging by users. Figure 4.9 is a functional block diagram of the UBC.

The following are set as a group for the conditions of the bus cycle under which the break will occur:

1. Address (bits can be masked individually with the break address mask register)
2. CPU cycle and/or DMA cycle
3. Instruction fetch and/or data access
4. Read and/or write
5. Operand size (byte, word, longword access)

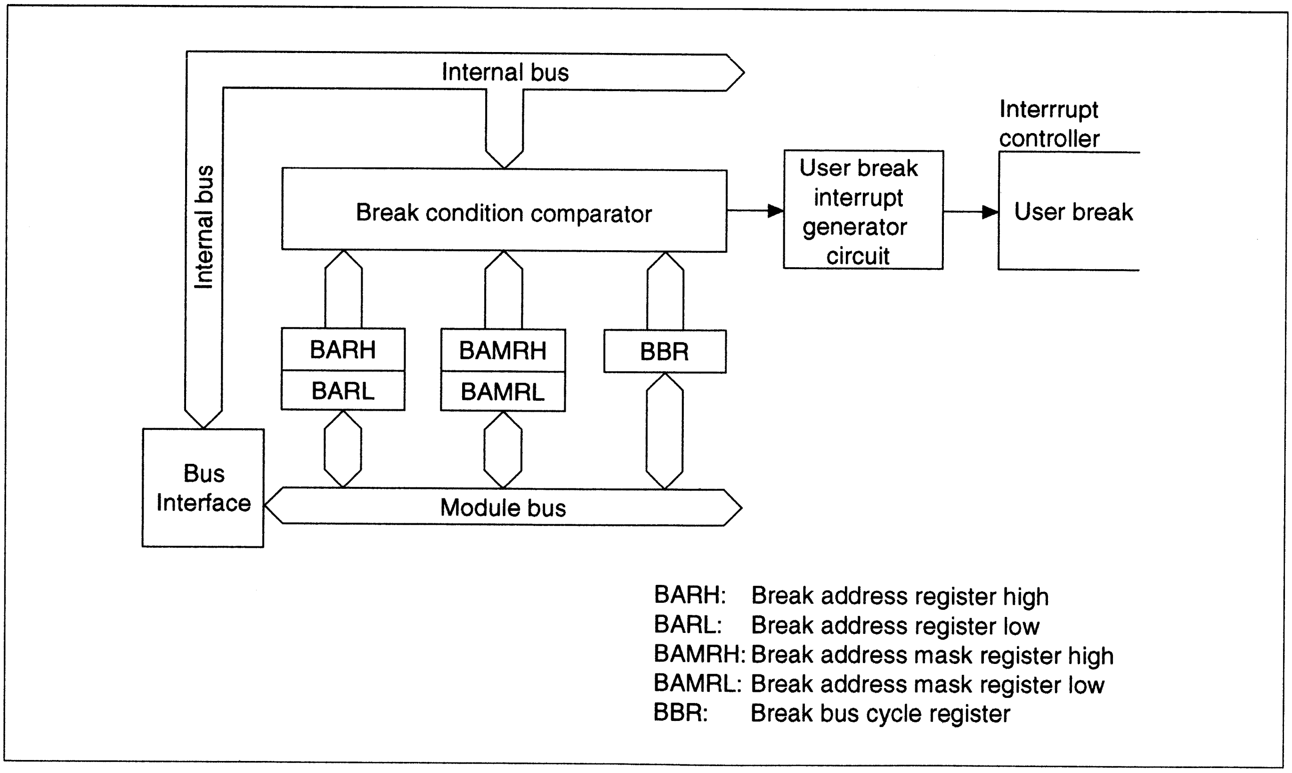

Internal bus
Internal bus
Interrrupt controller
User break interrupt generator circuit
Break condition comparator
User break
BARH
BARL
BAMRH
BAMRL
BBR
Bus Interface
Module bus
BARH: Break address register high
BARL: Break address register low
BAMRH: Break address mask register high
BAMRL: Break address mask register low
BBR: Break bus cycle register

Figure 4.9 User Break Controller Block Diagram

4.11 Clock Pulse Generator (CPG)

SH series microprocessors have built-in clock pulse generators (CPG) so they can run off a crystal oscillator or an external clock. The frequency used can be the same as the internal system clock (CK) frequency. When a 20-MHz crystal generator is used, for example, the LSI also runs at 20 MHz. Figure 4.10 is a functional block diagram of the CPG. Figure 4.11 shows crystal oscillator connections, and figure 4.12 shows external clock input.

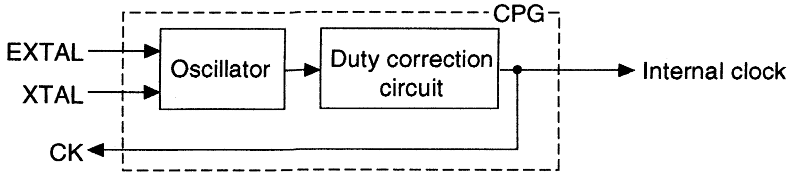


Figure 4.10 Clock Pulse Generator Block Diagram

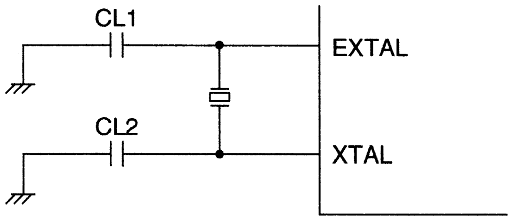


Figure 4.11 Connecting the Crystal Oscillator

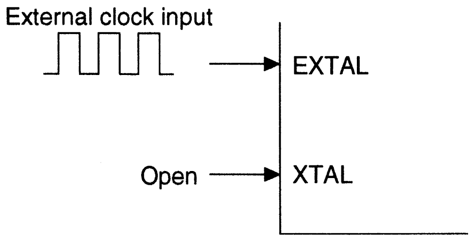


Figure 4.12 Inputting the External Clock

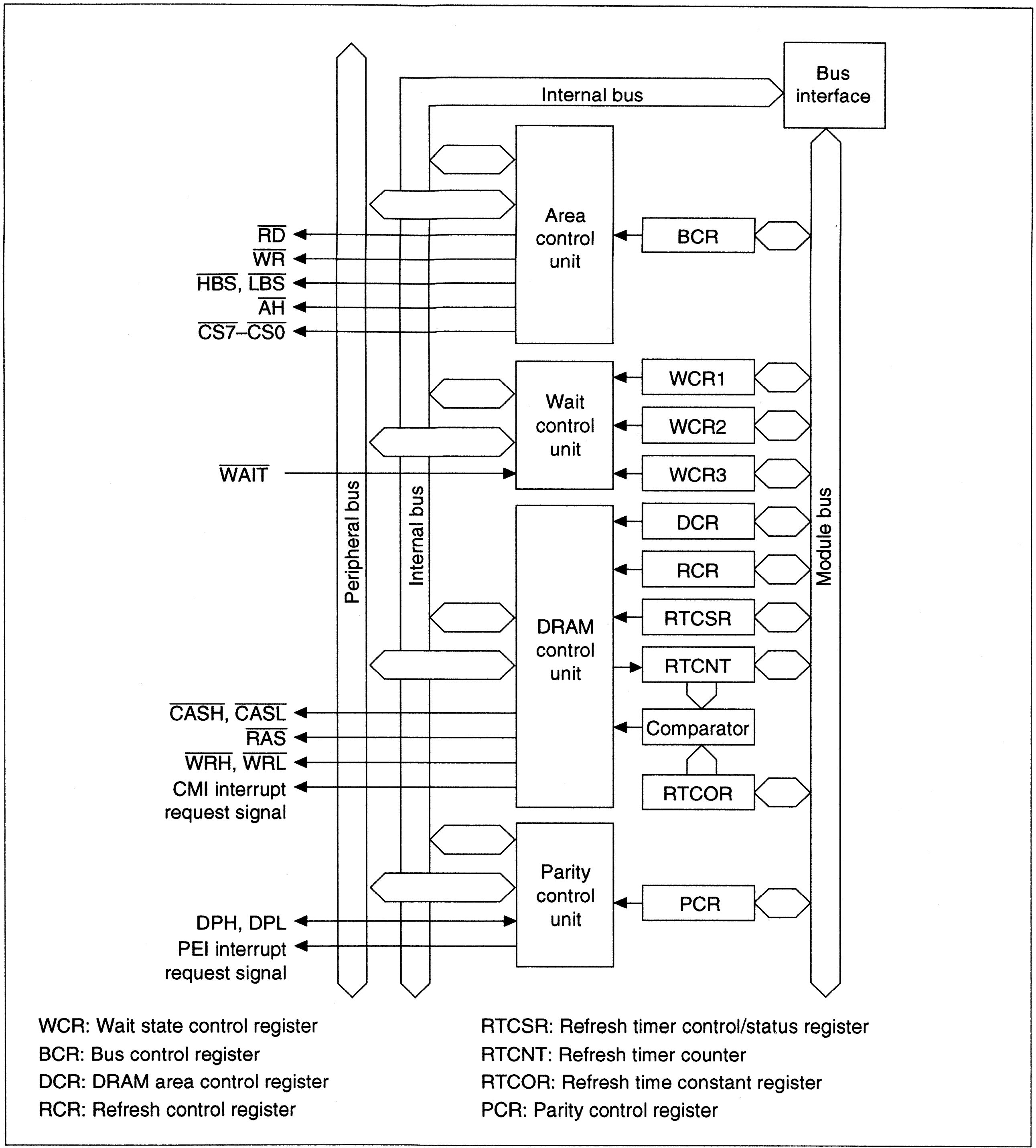
Section 5 Chip Peripheral Modules

5.1 Bus State Controller (BSC)

5.1.1 Address Space

The bus state controller (BSC) divides address space into eight areas and outputs control signals used to access the areas. BSC functions enable the LSI to link directly with DRAM, SRAM, and other kinds of memory, and to peripheral LSIs. Figure 5.1 is a block diagram of the BSC. Table 5.1 lists memory space division information. Figures 5.2–5.9 show address space for areas 0–7.

- Address space divided into 8 areas.
  - A maximum 4-Mbyte linear address space for each of 8 areas, 0–7 (area 1 can be up to 16 Mbytes linear space when set for DRAM). The space that can actually be used varies with the type of memory connected.
  - Bus width (8 bits or 16 bits) can be selected by access address.
  - Onchip ROM and RAM is accessed in 1 cycle (32-bit bus).
  - Wait states can be inserted for each area.
  - Control signal output and bus widths appropriate for each area.
- Direct interface to DRAM
  - Multiplexes row addresses/column addresses.
  - Two types of byte access signals.
  - Supports burst operation.
  - Supports CAS-before-RAS refresh and self-refresh.
- Access control for all peripheral LSIs
  - Address/data multiplex function.
- Parallel execution of external and internal accesses using write buffer
- Supports parity check and generation
  - Interrupt requests generated for parity error (PEI interrupt request signal).
- Refresh counter can be used as an interval timer
  - Interrupt request generated at compare match (CMI interrupt request signal).



WCR: Wait state control register
BCR: Bus control register
DCR: DRAM area control register
RCR: Refresh control register
RTCSR: Refresh timer control/status register
RTCNT: Refresh timer counter
RTCOR: Refresh time constant register
PCR: Parity control register

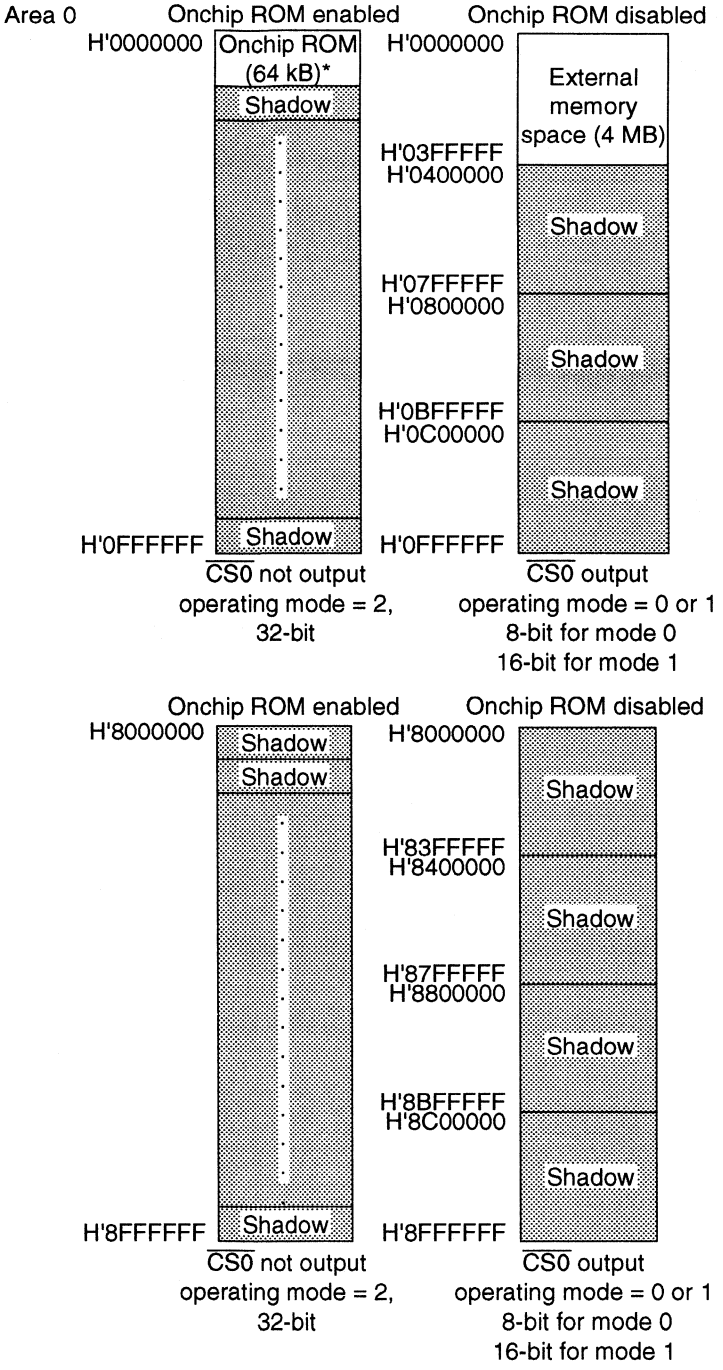
Figure 5.1 BSC Block Diagram

Table 5.1 Summary of Area Space Divisions

Area	Address	Assignable Memory	Capacity (Linear Space)	Access Size	CS Output
0	H'0000000–H'0FFFFFF	Onchip ROM*1	64/16/32 kbytes*1	32	—
		External memory*2	4 Mbytes	8/16*3	$\overline{\text{CS0}}$
	H'8000000–H'8FFFFFF	Onchip ROM*1	64/16/32 kbytes*1	32	—
		External memory*1	4 Mbytes	8/16*3	$\overline{\text{CS0}}$
1	H'1000000–H'1FFFFFF	External memory	4 Mbytes	8	$\overline{\text{CS1}}$
		DRAM*4	16 Mbytes	8	$\overline{\text{RAS}}$, CAS
	H'9000000–H'9FFFFFF	External memory	4 Mbytes	16	$\overline{\text{CS1}}$
		DRAM*4	16 Mbytes	16	$\overline{\text{RAS}}$, CAS
2	H'2000000–H'2FFFFFF	External memory	4 Mbytes	8	$\overline{\text{CS2}}$
	H'A000000–H'AFFFFFF	External memory	4 Mbytes	16	$\overline{\text{CS2}}$
3	H'3000000–H'3FFFFFF	External memory	4 Mbytes	8	$\overline{\text{CS3}}$
	H'B000000–H'BFFFFFF	External memory	4 Mbytes	16	$\overline{\text{CS3}}$
4	H'4000000–H'4FFFFFF	External memory	4 Mbytes	8	$\overline{\text{CS4}}$
	H'C000000–H'CFFFFFF	External memory	4 Mbytes	16	$\overline{\text{CS4}}$
5	H'5000000–H'5FFFFFF	Onchip peripheral module	512 bytes	8/16*3	—
	H'D000000–H'DFFFFFF	External memory	4 Mbytes	16	$\overline{\text{CS5}}$
6	H'6000000–H'6FFFFFF	External memory*7	4 Mbytes	8/16*6	$\overline{\text{CS6}}$
		Multiplexed I/O	4 Mbytes		
	H'E000000–H'EFFFFFF	External memory	4 Mbytes	16	$\overline{\text{CS6}}$
7	H'7000000–H'7FFFFFF	External memory	4 Mbytes	8	$\overline{\text{CS7}}$
	H'F000000–H'FFFFFFF	Onchip RAM	8 kbytes*8 4 kbytes*9 1 kbyte*10	32	—

Notes: 1. When onchip ROM of SH7034 (64 kbytes), SH7020 (16 kbytes) or SH7021 (32 kbytes) is enabled

2. When onchip ROM of SH7034, SH7020, or SH7021 is disabled or for SH7032
3. Select with MD0
4. Select with DRAME
5. Select with address bit A8
6. Select with address bit A14
7. Select with IOE
8. For SH7032
9. For SH7034
10. For SH7020, SH7021



Note: SH7032: None
SH7034: 64 kbytes
SH7020: 16 kbytes
SH7021: 32kbytes

Figure 5.2 Area 0 Address Space

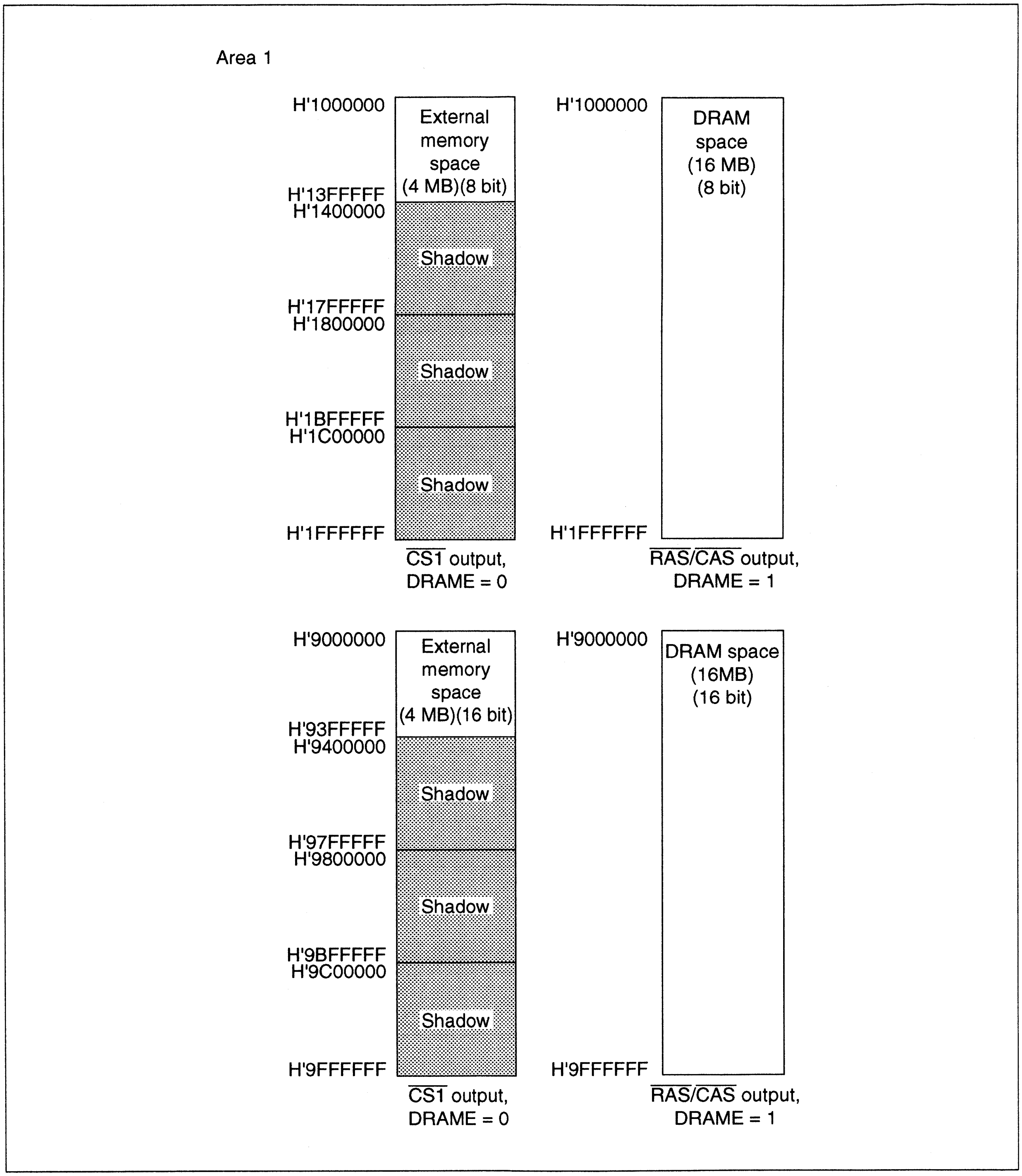


Figure 5.3 Area 1 Address Space

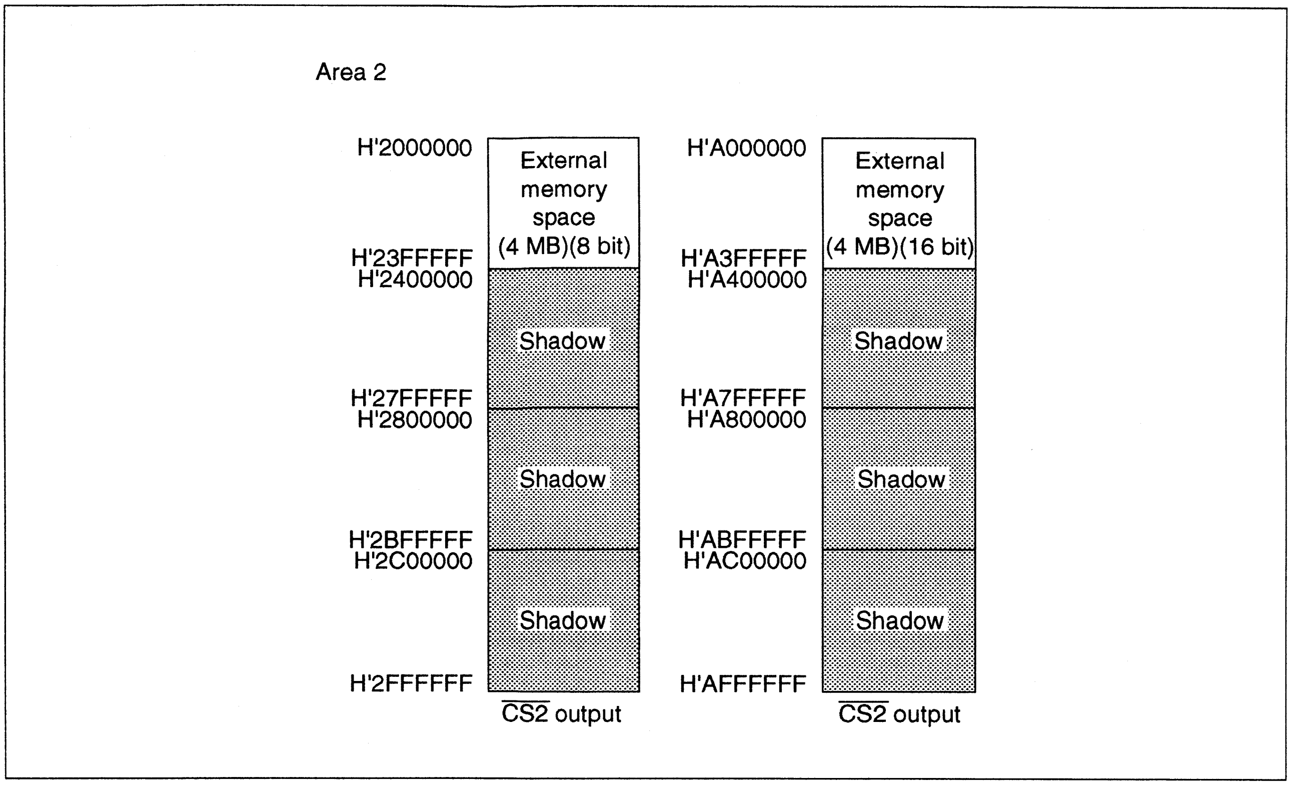


Figure 5.4 Area 2 Address Space

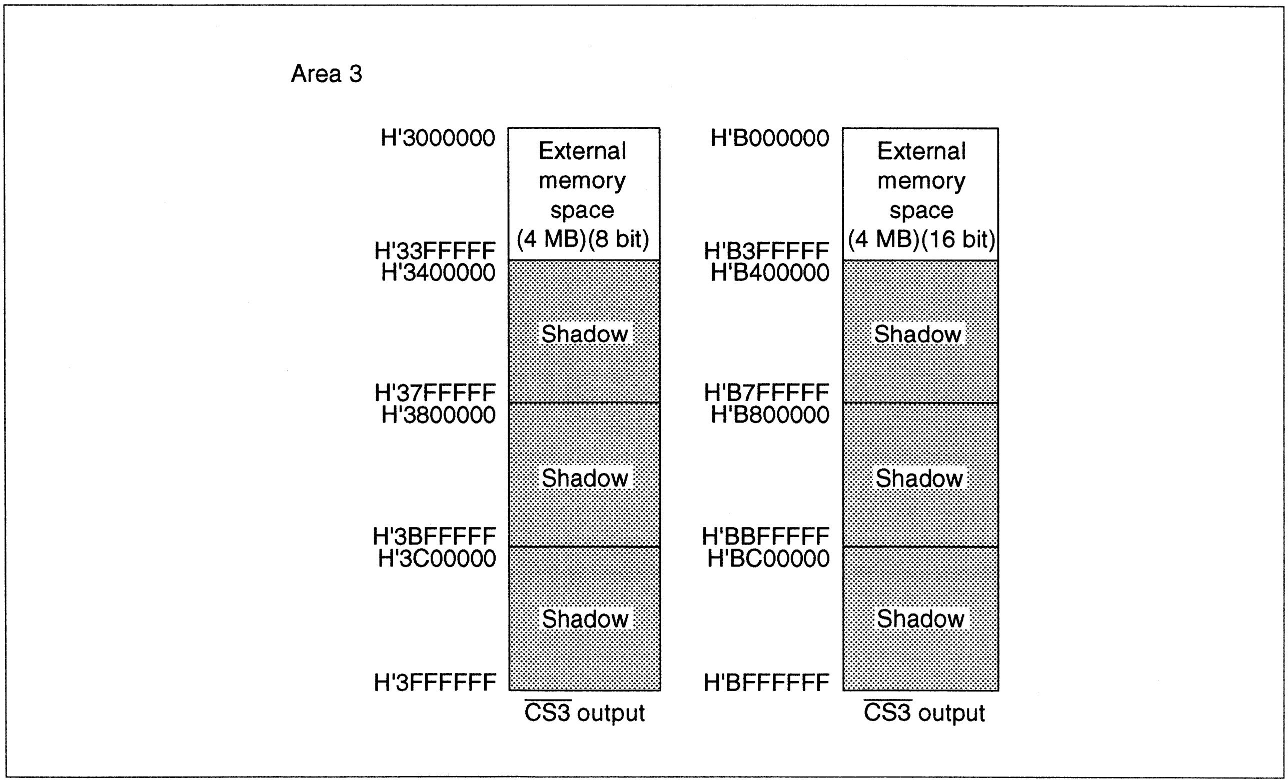


Figure 5.5 Area 3 Address Space

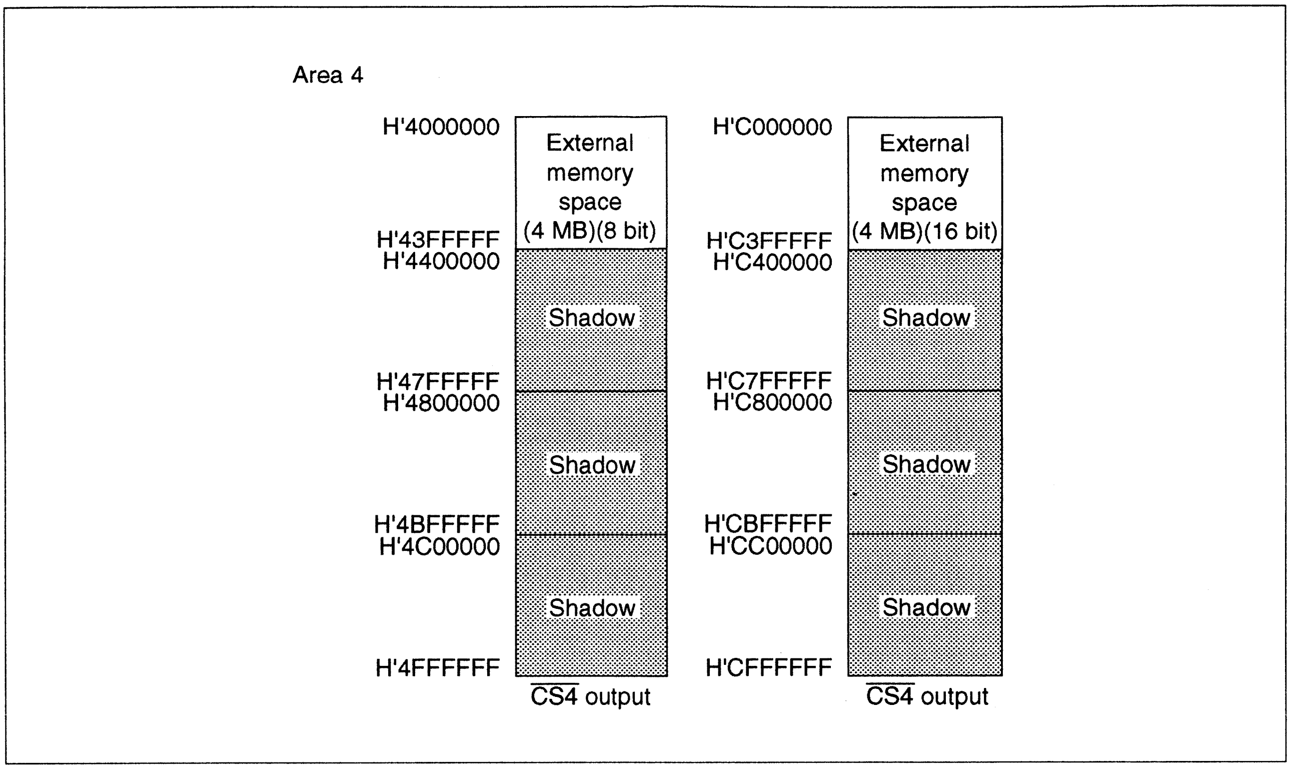


Figure 5.6 Area 4 Address Space

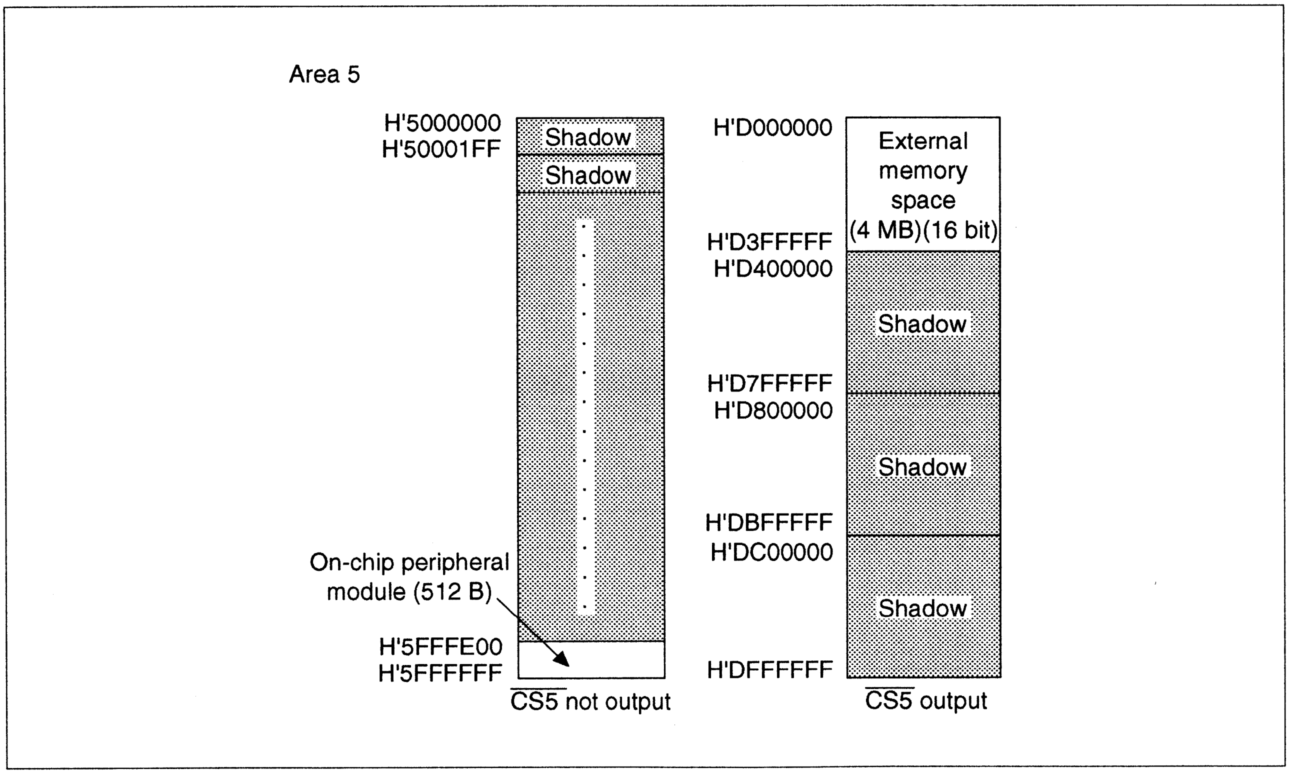


Figure 5.7 Area 5 Address Space

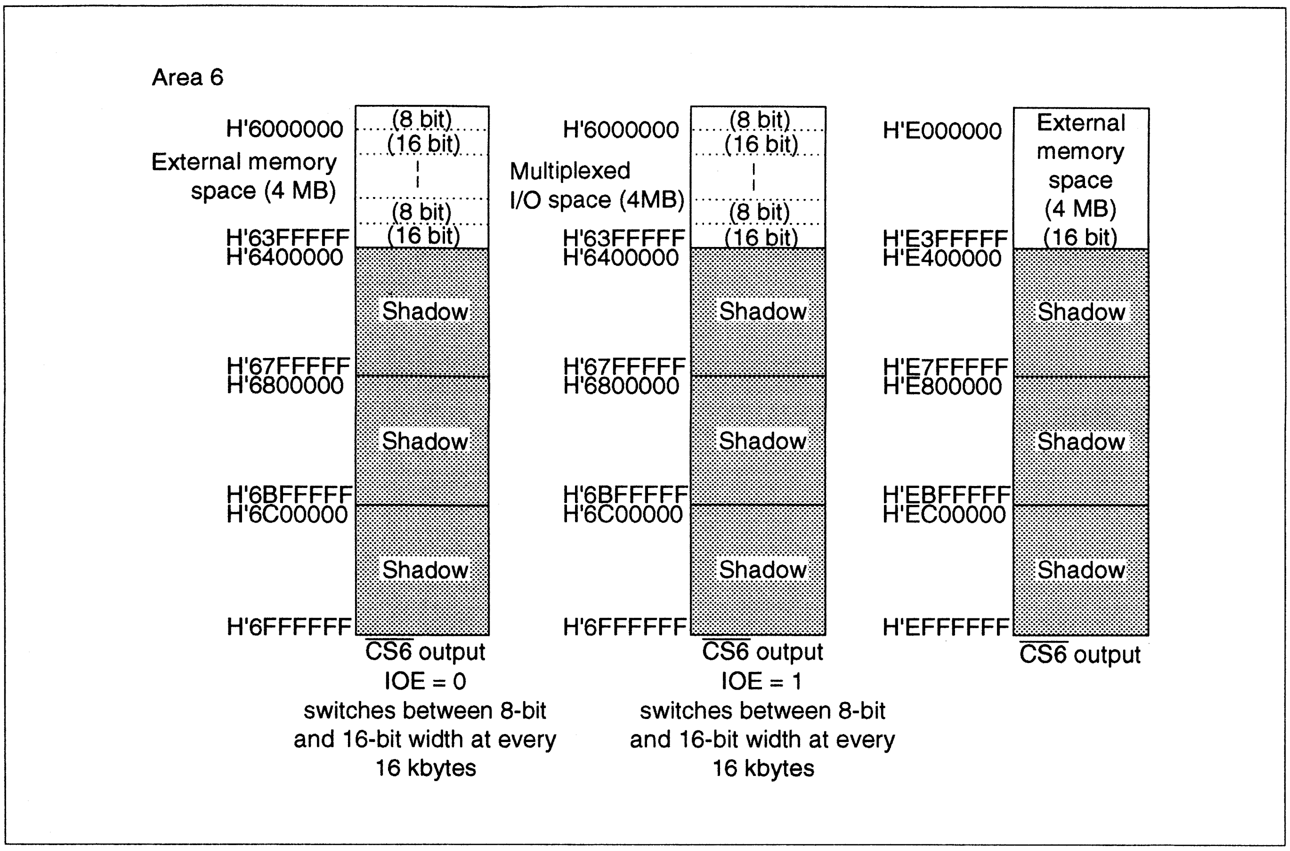


Figure 5.8 Area 6 Address Space

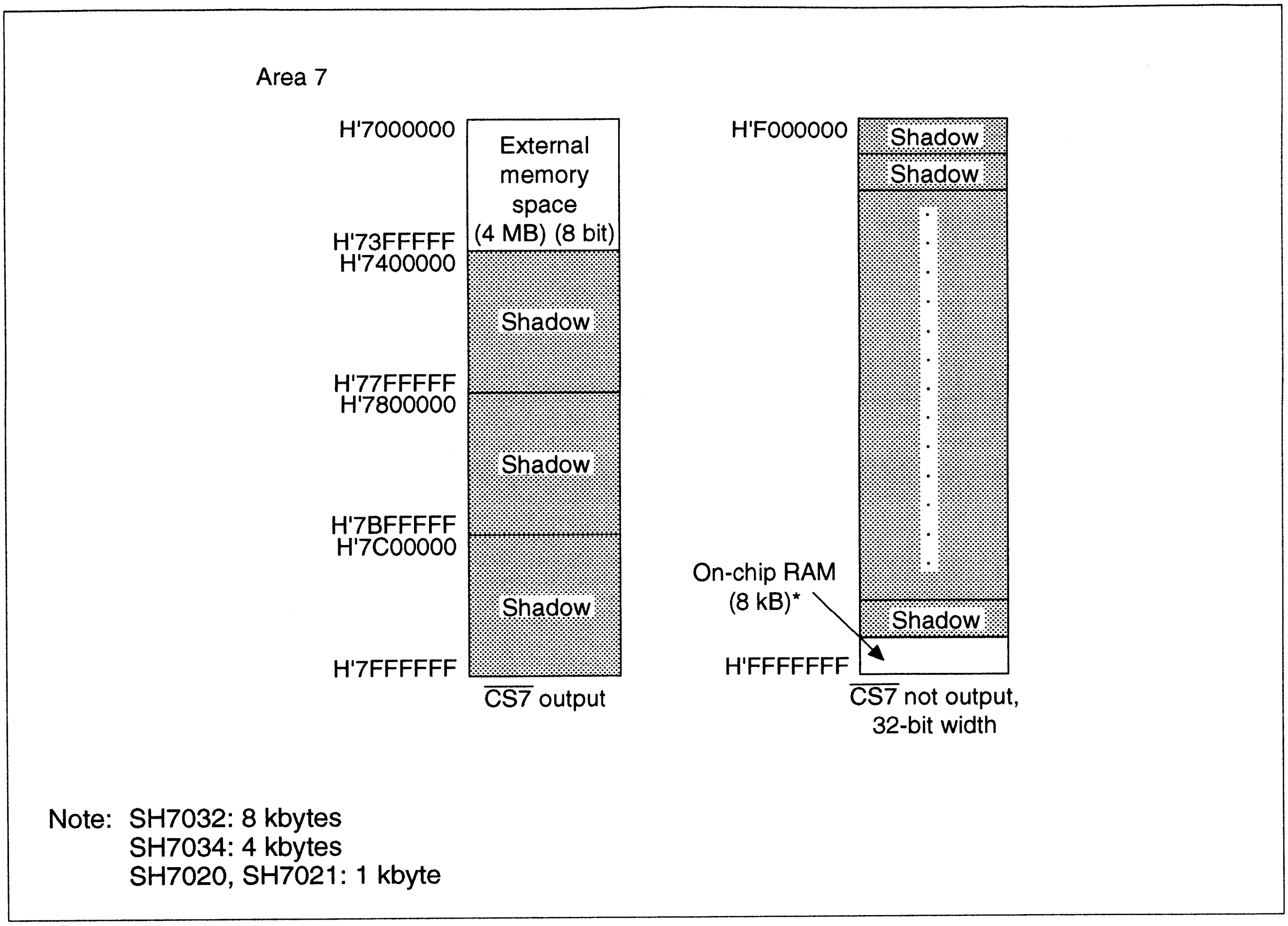


Figure 5.9 Area 7 Address Space

- DRAM interface
  - — Row address/column address multiplexing: Choose between 8, 9, and 10 bits.
  - — 2 types of byte access signals: Choose between dual $\overline{\text{CAS}}$ system and dual $\overline{\text{WE}}$ system when connected to ×16 DRAM.
  - — Burst operation: Supports high-speed page mode. Enables fast DRAM access when the same row address repeats.
  - — CAS-before-RAS refresh: Refresh interval is programmable.
  - — Self-refresh: Can do a self-refresh when the enable bit is set.

- Parity
  - — Parity can be generated or checked when DRAM area or area 2 is accessed.
  - — Odd/even selectable for parity polarity.

- Wait
  - — Pin wait insertion enabled when external memory space of areas 0–7 is accessed.
  - — A long wait of up to 4 cycles can be inserted in areas 0, 2, and 6 using software.

5.1.2 Typical Bus Cycle Examples

Figures 5.10–5.14 show cycle timing, memory connections, and parallel memory space write timing.

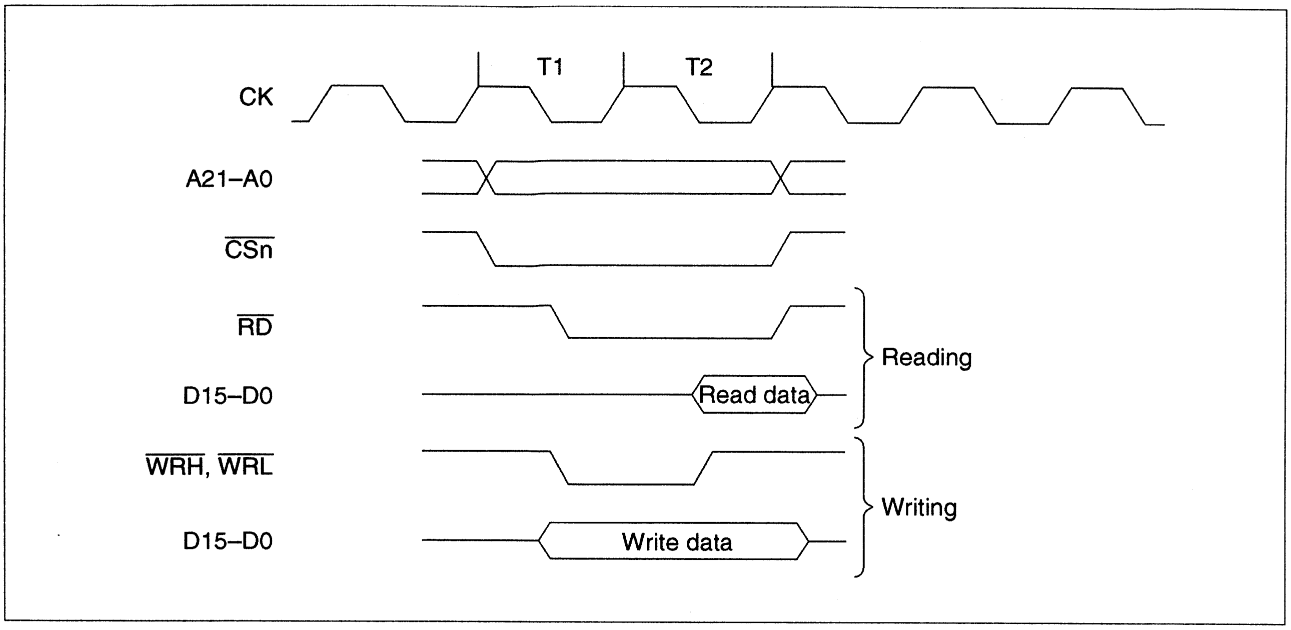


Figure 5.10 External Memory Space, Two-Cycle Access (SRAM, ROM)

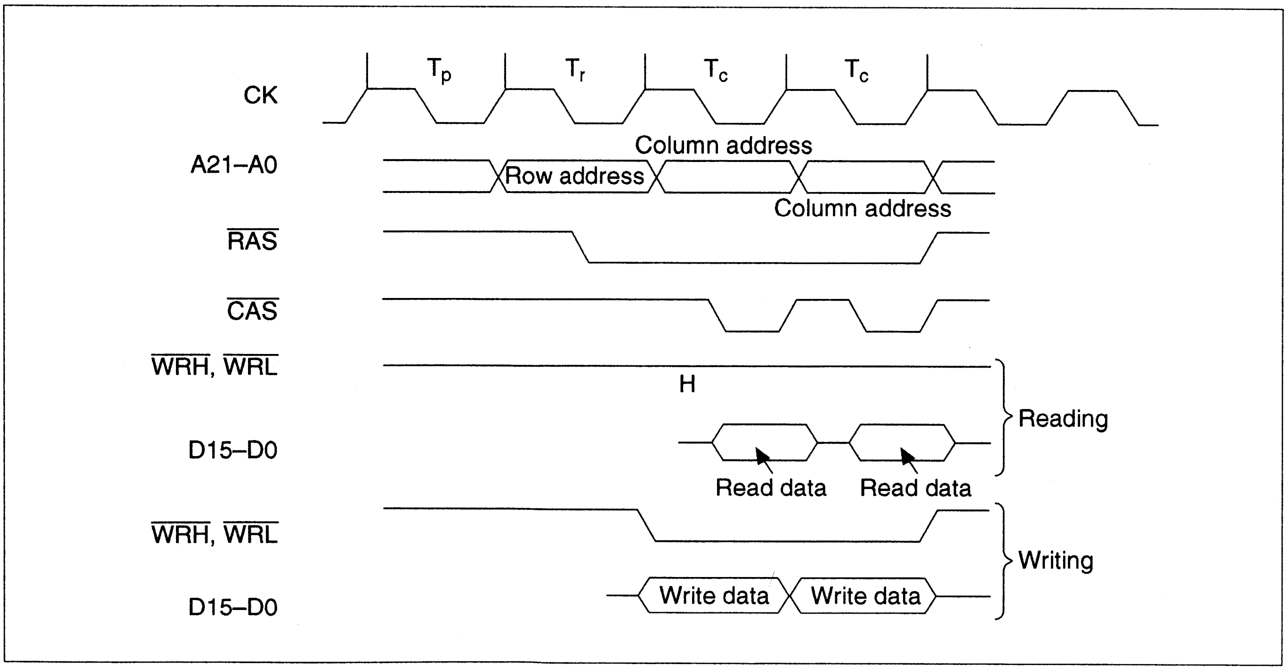


Figure 5.11 DRAM Space, One-Cycle Access, High-Speed Page Mode

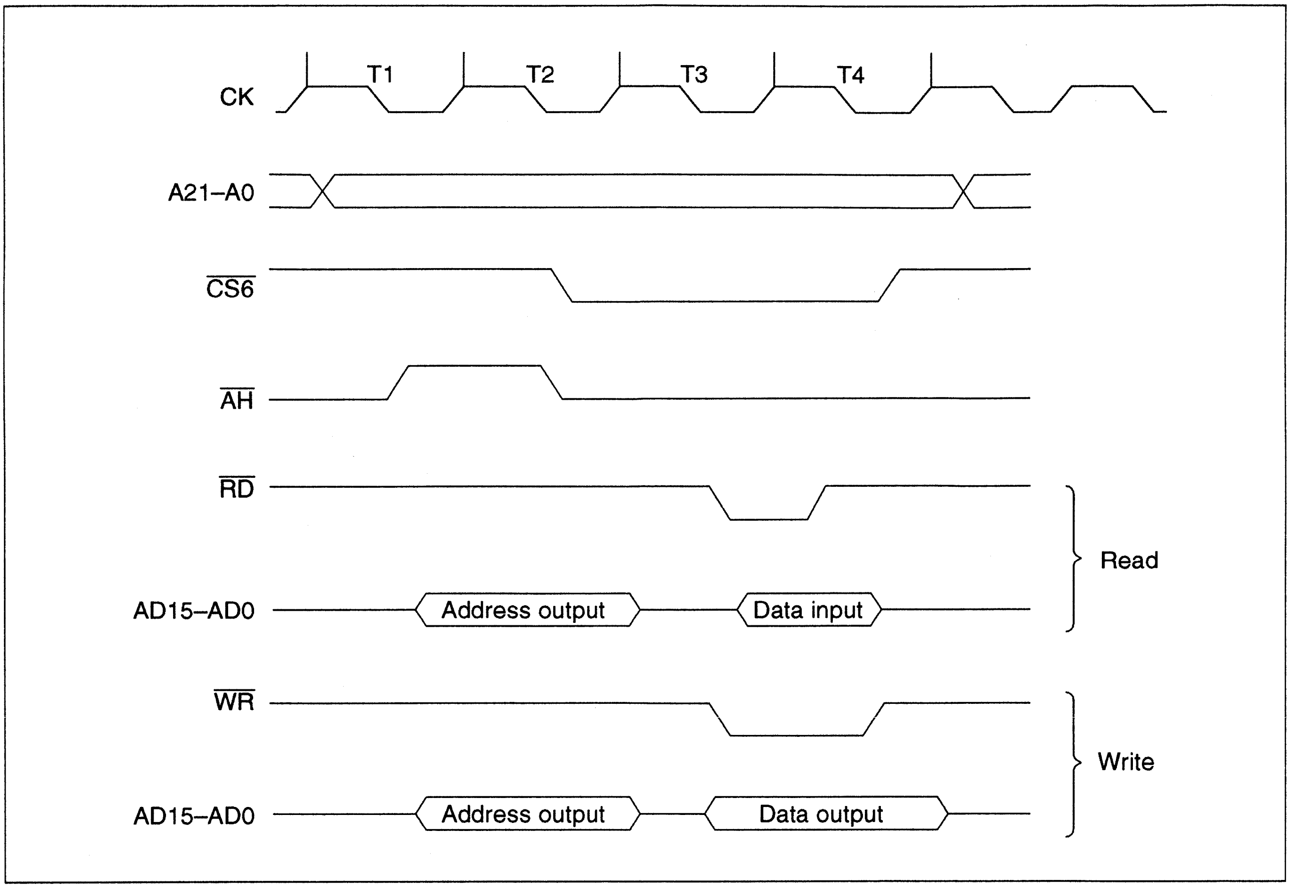


Figure 5.12 Multiplexed I/O Area Access

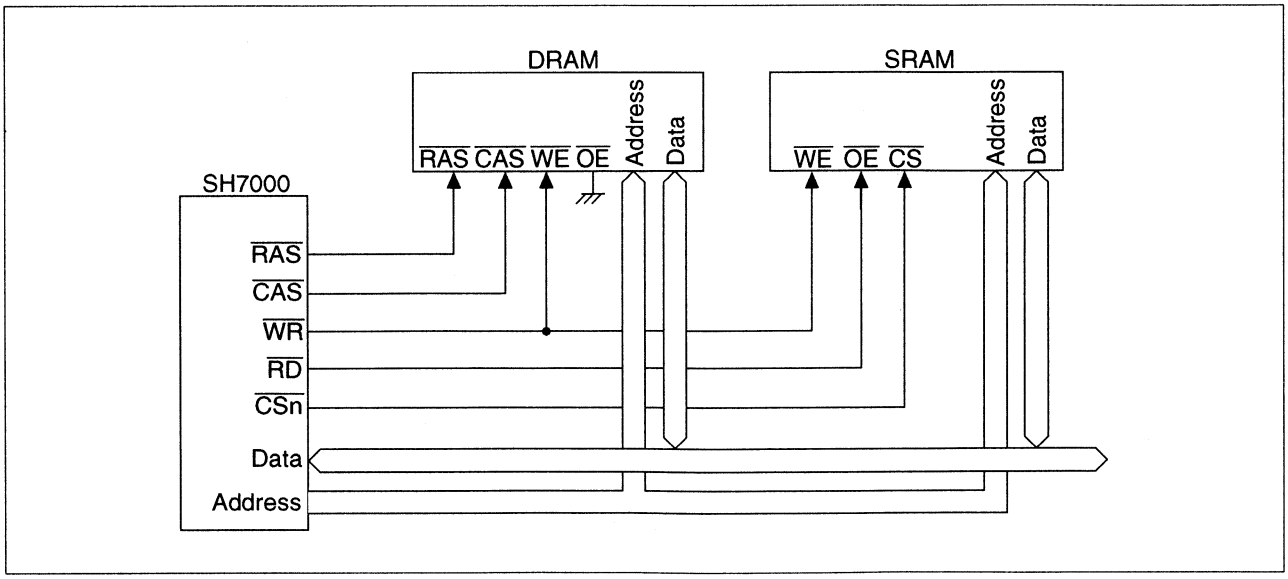
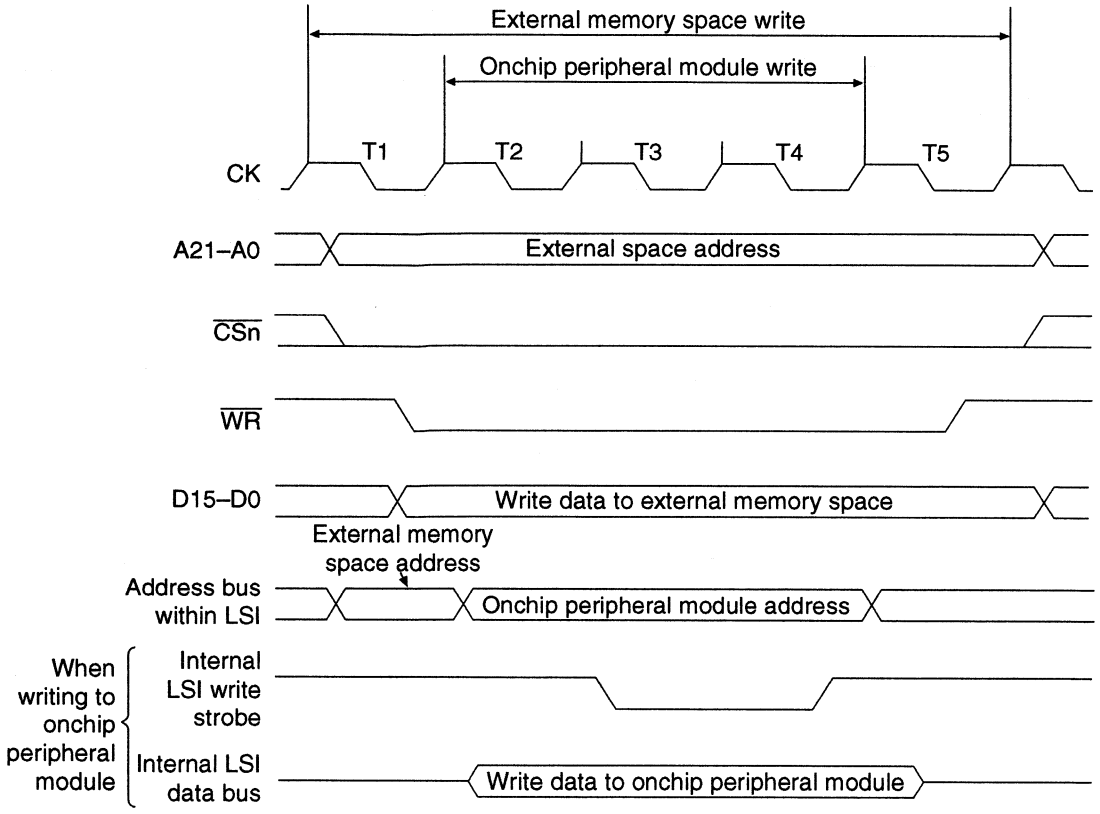


Figure 5.13 Typical Memory Connections


External memory space write
Onchip peripheral module write
T1
T2
T3
T4
T5
CK
A21–A0
External space address
$\overline{\text{CSn}}$
$\overline{\text{WR}}$
D15–D0
Write data to external memory space
External memory space address
Address bus within LSI
Onchip peripheral module address
When writing to onchip peripheral module
Internal LSI write strobe
Internal LSI data bus
Write data to onchip peripheral module

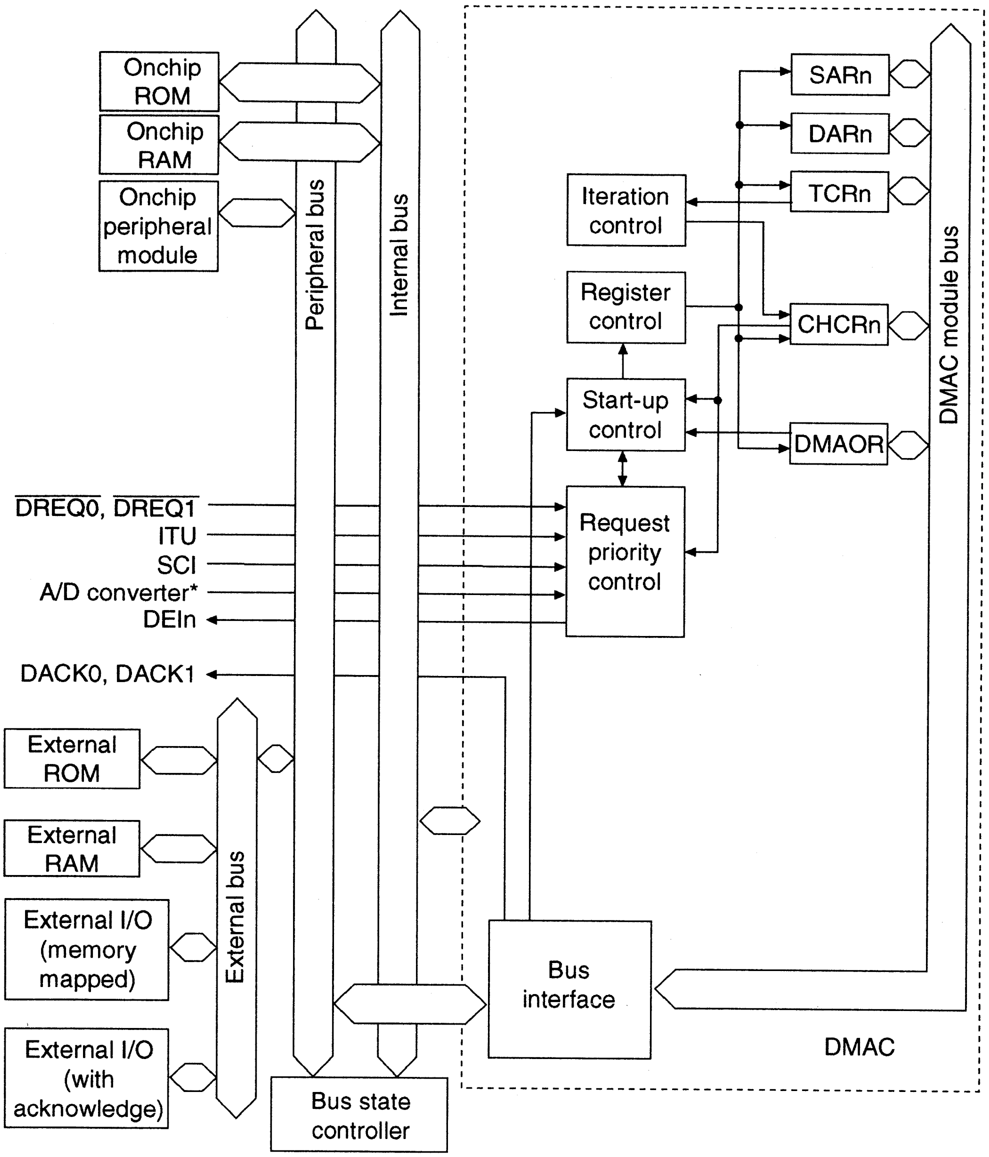
- Enables parallel operation of write to external space using write buffer and internal reads or writes.
- Enables parallel operation of onchip memory and onchip peripheral module accesses via the CPU with DMA signal transfers when cycle-steal mode DMA single transfers are used.

Figure 5.14 Example of Parallel Operation of External Memory Space Write and Onchip Peripheral Module Write

5.2 Direct Memory Access Controller (DMAC)

The SH7000 series microprocessors include four-channel direct memory access controllers (DMAC). DMACs can be used in place of the CPU to perform data transfers between external I/O devices, memory, and onchip peripheral modules. Figure 5.15 is a functional block diagram of the DMAC. DMAC has the following features:

- 4 channels
- Address space: 4 Gbytes on the architecture
- Selectable data transfer unit: 8 or 16 bits
- Maximum transfer count: 64 k (65536) transfers
- Single address mode transfers (channels 0 and 1) or dual address mode transfers: (channels 0–3)
- Transfer requests:
  - External request: two $\overline{\text{DREQ}}$ pins (channels 0 and 1 only), by edge or by level
  - Requests from onchip peripheral modules: SCI, A/D converter (SH7032 and SH7034 only), and ITU
  - Auto-request
- Eligible transfer devices (table 5.2):
  - Memory: Onchip ROM or RAM, external ROM, or RAM
  - Onchip peripheral modules
  - External I/O devices: Memory-mapped devices, or devices with acknowledge feature
- Bus modes: Cycle-steal mode or burst mode
- Channel priority levels: Fixed, round-robin, or external-pin round-robin
- Can request interrupt when data transfer ends



DMAOR:	DMA operation register
SARn:	DMA source address register
DARn:	DMA destination address register
TCRn:	DMA transfer count register
CHCRn:	DMA channel control register
DEIn:	Request to CPU for DMA transfer-end interrupt
n:	0–3

Note: SH7032 and SH7034 only

Figure 5.15 DMAC Block Diagram

Table 5.2 DMAC Transfers

Transfer Source/Destination	Address Mode	Bus Mode	Transfer Request
Memory* ↔ memory*	Dual	Burst/cycle steal	External/onchip peripheral/auto request
Memory* ↔ onchip peripheral module	Dual	Burst/cycle steal	External/onchip peripheral/auto request
Onchip peripheral module ↔ onchip peripheral module	Dual	Burst/cycle steal	External/onchip peripheral/auto request
Memory* ↔ memory-mapped external device	Dual	Burst/cycle steal	External/onchip peripheral/auto request
External memory ↔ external device with acknowledge	Single	Burst/cycle steal	External
Onchip peripheral module ↔ memory-mapped external device	Dual	Burst/cycle steal	External/onchip peripheral/auto request
Memory-mapped external device ↔ memory-mapped external device	Dual	Burst/cycle steal	External/onchip peripheral/auto request
Memory-mapped external device ↔ external device with acknowledge	Single	Burst/cycle steal	External

Note: Includes onchip memory

5.2.1 Address Modes

There are two address modes, dual, and single.

Dual Address Mode: Accesses of the transfer source and transfer destination are divided between two bus cycles and an address is output for each.

Single Address Mode: The external I/O device is accessed by DACK signal simultaneously with the addressing of external memory and the DMA transfer performed in one bus cycle.

5.2.2 Bus Mode Operation

There are 2 bus modes, cycle steal and burst.

Cycle Steal Mode: After one word of DMA transfer, the bus right is always released and the bus right is transferred to the other bus master.

Burst Mode: Once the bus is captured, the transfer continues until the transfer end conditions are satisfied. Transfer proceeds according to the $\overline{\text{DREQ}}$ pin state when the pin is sampled in the external request mode.

5.2.3 Transfer Request

There are three types of transfer requests: external requests, requests from onchip peripheral modules, and auto requests.

External Requests: Channel 0 is started up by the $\overline{\text{DREQ0}}$ pin and channel 1 by the $\overline{\text{DREQ1}}$ pin. The $\overline{\text{DREQ}}$ pin can be sampled by either falling edge or level.

Requests from Onchip Peripheral Modules:

- SCI0 receive data full interrupt transfer
- SCI0 transmit data empty interrupt transfer
- SCI1 receive data full interrupt transfer
- SCI1 transmit data empty interrupt transfer
- ITU timer 0 compare-match A/input capture A
- ITU timer 1 compare-match A/input capture A
- ITU timer 2 compare-match A/input capture A
- ITU timer 3 compare-match A/input capture A
- A/D converter A/D conversion interrupt request

These requests are automatically cleared when the DMAC is started up.

Auto Request: Transfer is started by setting the DE bit of the DMAC channel control register.

5.2.4 Channel Priority

When the DMAC receives simultaneous transfer requests on two or more channels, it selects a channel according to a predetermined priority order. There are two fixed priority modes, a round-robin mode, and an external-pin round-robin mode.

Fixed Priority Modes: In these modes, the priority levels among the channels remain fixed. Two priority orders can be selected:

1. Ch 0 > Ch 3 > Ch 2 > Ch 1
2. Ch 1 > Ch 3 > Ch 2 > Ch 0

Round-Robin Mode: Each time a word or byte is transferred on channel 1, the priority order is rotated. The channel on which the transfer has just finished rotates to the bottom of the priority order. When necessary, the priority order of channels other than the one that just finished the transfer can also be shifted to keep the relationship between the channels from changing. The initial priority order after a reset is channel 0 > channel 3 > channel 2 > channel 1. Figure 5.16 shows round robin transfer. Figure 5.17 is an example of round-robin transfer.

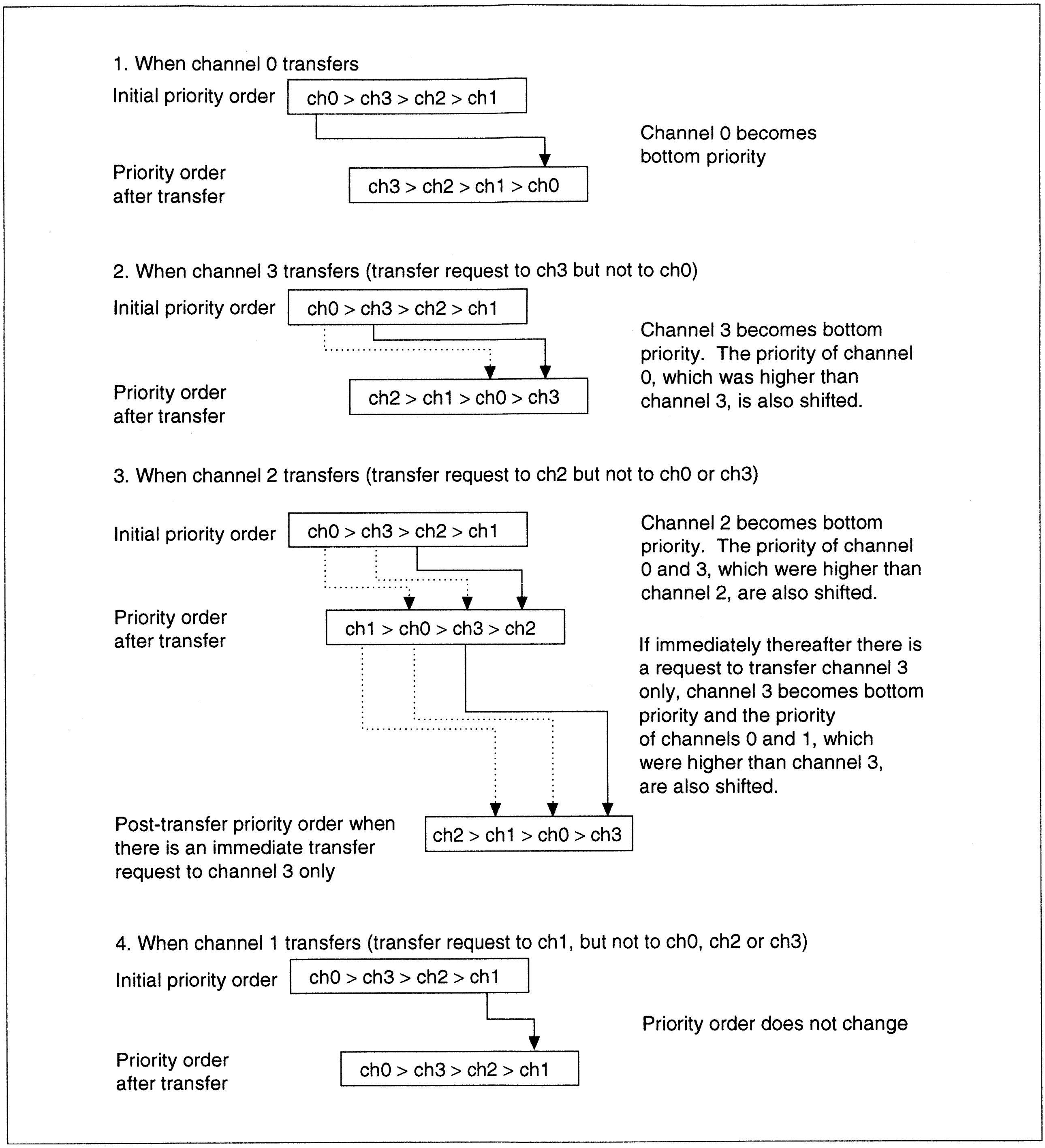

1. When channel 0 transfers
Initial priority order
ch0 > ch3 > ch2 > ch1
Channel 0 becomes bottom priority
Priority order after transfer
ch3 > ch2 > ch1 > ch0
2. When channel 3 transfers (transfer request to ch3 but not to ch0)
Initial priority order
ch0 > ch3 > ch2 > ch1
Channel 3 becomes bottom priority. The priority of channel 0, which was higher than channel 3, is also shifted.
Priority order after transfer
ch2 > ch1 > ch0 > ch3
3. When channel 2 transfers (transfer request to ch2 but not to ch0 or ch3)
Initial priority order
ch0 > ch3 > ch2 > ch1
Channel 2 becomes bottom priority. The priority of channel 0 and 3, which were higher than channel 2, are also shifted.
Priority order after transfer
ch1 > ch0 > ch3 > ch2
If immediately thereafter there is a request to transfer channel 3 only, channel 3 becomes bottom priority and the priority of channels 0 and 1, which were higher than channel 3, are also shifted.
Post-transfer priority order when there is an immediate transfer request to channel 3 only
ch2 > ch1 > ch0 > ch3
4. When channel 1 transfers (transfer request to ch1, but not to ch0, ch2 or ch3)
Initial priority order
ch0 > ch3 > ch2 > ch1
Priority order does not change
Priority order after transfer
ch0 > ch3 > ch2 > ch1

Figure 5.16 Round-Robin Mode Transfer

The following shows how the channel priority order changes when there are simultaneous transfer requests to channels 0 and 1 and then another request is made to channel 3 during the channel 0 transfer.

- Transfer requests occur simultaneously to channels 0 and 1.
- Channel 0 has higher priority, so the channel 0 transfer starts. (Channel 1 waits.)
- A transfer request to channel 3 occurs during the channel 0 transfer. (Channels 1 and 3 both wait.)
- When the channel 0 transfer ends, channel 0 goes to the bottom of the priority order.
- Channel 3 now has higher priority than channel 1, so the channel 3 transfer starts. (Channel 1 waits.)
- When the channel 3 transfer ends, channel 3 goes to the bottom of the priority order.
- The channel 1 transfer starts.
- When the channel 1 transfer ends, channel 1 goes to the bottom of the priority order. Channel 2 moves with channel 1, so its priority is lowered as well.

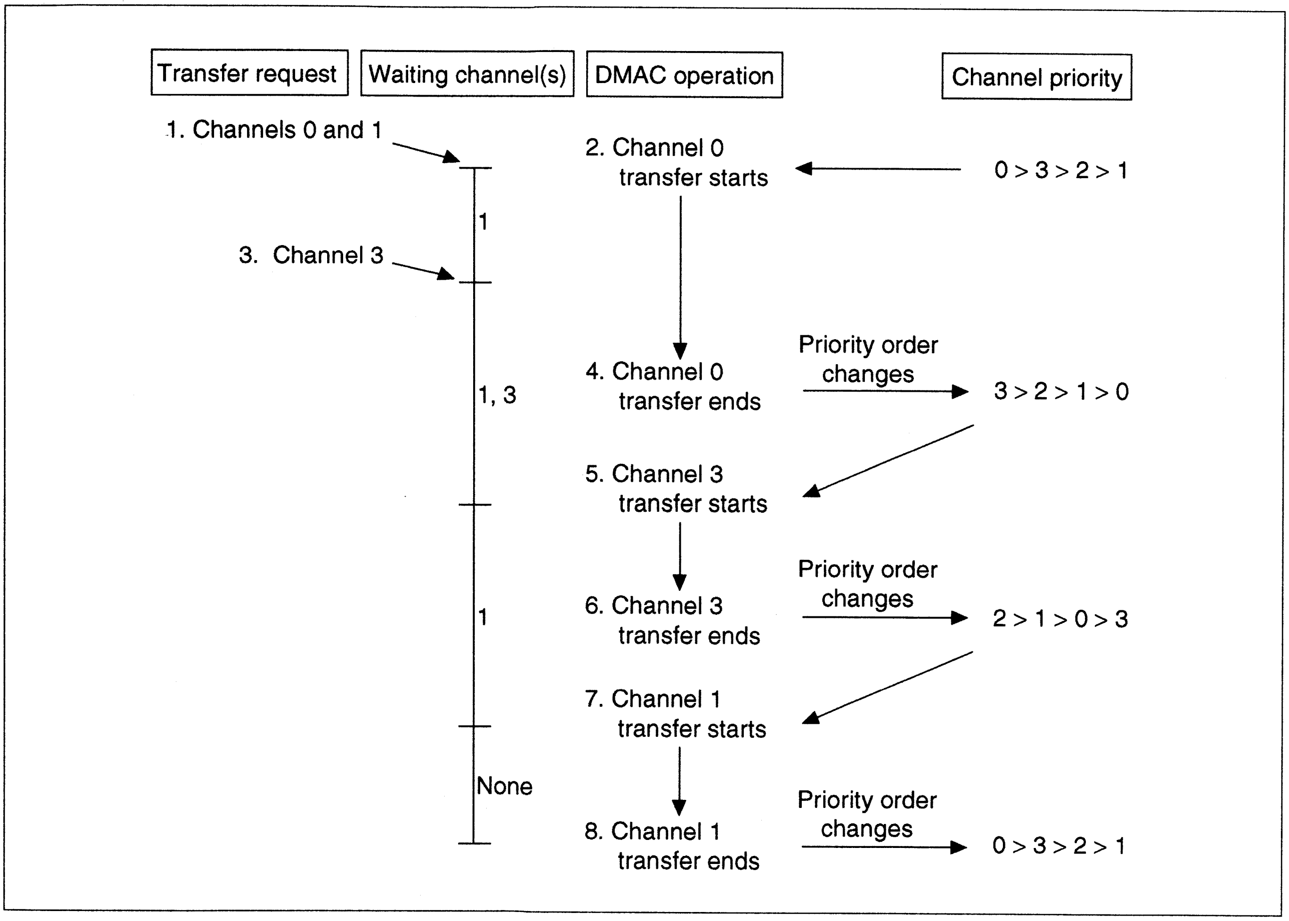


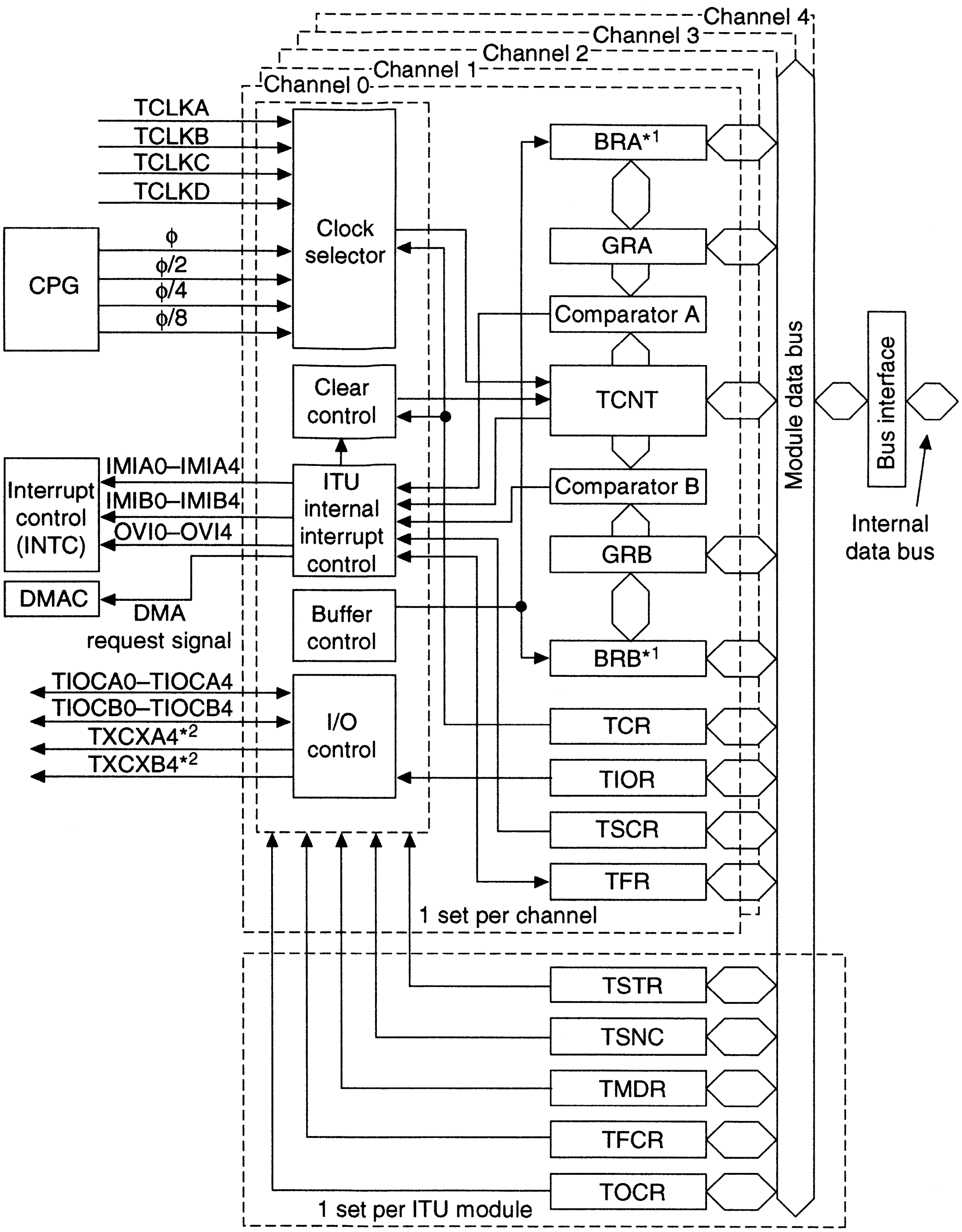
Figure 5.17 Example of Changes in Priority in Round-Robin Mode

External-Pin Round-Robin Mode: External-pin round-robin mode switches the priority levels of channel 0 and channel 1, which are the channels that can be used by the external request mode. Like the round-robin mode, the priority levels are changed after each (byte or word) transfer on channel 0 or channel 1 is completed. The priority levels of channels 2 and 3 do not change.

5.3 16-Bit Integrated Timer Pulse Units (ITU)

The 16-bit integrated-timer pulse unit (ITU) has five channels of 16-bit timers that come in three types that can output 10 systems of independent waveforms, process pulse I/O, output complementary PWM, measure pulse width, and process two-phase encoders. The 16-bit timers can also generate the output trigger for the programmable timing pattern controller (TPC) and to start the direct memory access controller (DMAC). Figure 5.18 is a functional block diagram of the ITU. Table 5.3 lists ITU functions. The ITU:

- Processes a maximum of 12 different pulse outputs and 10 different pulse inputs.
- Includes 10 general registers, which can be set to function independently as output compare or input capture.
- Selects from 8 counter input clock sources for all channels: ϕ, $\phi/2$, $\phi/4$, $\phi/8$, TCLKA, TCLKB, TCLKC, TCLKD.
- Compares match waveform output: 0 output/1 output/toggle output.
- Captures input (selectable) on rising edge, falling edge, or both rising and falling edges (input capture function).
- Clears counters by a compare match or input capture (counter clearing function).
- Writes to two or more timer counters (TCNT) simultaneously. Two or more timer counters can be simultaneously cleared by a compare match or input capture. Counter synchronization functions enable synchronized input/output for each register.
- Provides PWM output with any duty cycle (PWM mode). When combined with the counter synchronizing function, enables up to five-phase PWM output.
- Double-buffers input capture registers (buffer mode). Output compare registers can be updated automatically.
- Allows complementary PWM mode: three-phase PWM output is possible with non-overlapping positive and negative waveforms.
- Outputs three-phase PWM with positive and negative waveforms (reset-synchronized PWM mode).
- Counts two-phase encoder output automatically (phase counting mode).
- Allows 15 interrupt sources: 10 compare match/input capture interrupts (two sources per channel) and five overflow interrupts are vectored independently for a total of 15 sources. DMAC can be activated by compare match/input capture interrupts (4 sources).
- Allows high-speed access via the internal 16-bit bus.



Notes: 1. Channels 3, 4
2. Channel 4

TSTR: Timer start register
TSNC: Timer sync register
TMDR: Timer mode register
TFCR: Timer function control register
TOCR: Timer output control register
TCR: Timer control register
TIOR: Timer I/O control register

TIER: Timer interrupt enable register
TSR: Timer status register
TCNT: Timer counter
GRA: General register A
GRB: General register B
BRA: Buffer register A
BRB: Buffer register B

Figure 5.18 ITU Block Diagram

Table 5.3 Overview of ITU Functions

Category	Function	Description
Count	Count clock	Independently selectable with the TPCS bit of the TCR between ϕ, $\phi/2$, $\phi/4$, $\phi/8$, TCKLA, TCKLB, TCKLC, and TCKLD
	Counter clear	Set the CCLR bit of the TCR to clear the TCNT upon GRA/GRB register input capture or output compare.
Input/ output	Output compare/ input capture	Set for use as a compare register or as a input capture register using the IOA and IOB bits of TIOR. Select 0 output, 1 output, or toggle output (select 0 output or 1 output for channel 2)
Operating mode	Synchronized	Set the SYNC bit in TSNC for synchronized operation and do a synchronized preset/synchronized clear of the TCNT counters of multiple channels.
	PWM	Set the PWM bit of the TMDR for PWM mode and output a PWM waveform from the TIOCA pin. 1 is output for a compare match of the TCNT counter and GRA register and 0 for a compare match of TCNT and the GRB.
	Complementary PWM	Set channels 3 and 4 to complementary PWM mode with the CMD bits of TFCR to output a PWM waveform whose positive and negative phases do not overlap. TCNT3 and TCNT4 function as up/down counters. When TCNT3 matches GRA3, it starts decrementing; when TCNT4 underflows, it increments. TOCXA4 and TOCXB4 can be used for output to produce three-phase output. This is a function for channels 3 and 4 combined.
	Reset synchronized PWM	Set channels 3 and 4 to reset synchronized PWM mode with the CMD bits of TFCR to output positive and negative phase PWM waveforms. TCNT3 functions as an up counter; when it matches GRA3, it is cleared and begins counting cycles. TOCXA4 and TOCXB4 can be used for output to produce three-phase output. This is a function for channels 3 and 4 combined.
	Phase counting	Set the MDF bit on the TMDR for phase-counting mode and then use TCLKA as the A-phase input pin, TCLKB as the B-phase input pin, and increment or decrement at all edges. This function is for channel 2 only.

Table 5.3 Overview of ITU Functions (cont)

Category	Function	Description
Operating mode (cont)	Buffer	Set the BFA and BFB bits of the TMDR to the buffer mode. When GR is selected as an input capture register, the GR value is transferred to BR. When GR is selected as an output compare register, the BR value is transferred to GR. This function is for channels 3 and 4 only.
Interrupts and DMAC	Interrupts	There are 3 sources for each channel: compare match/input capture A and B and overflow. Select interrupt generation with the IMIEA, IMIEB and OVIE bits of TIER.
	DMAC startup	Set output compare match A or input capture B for channels 0, 1, 2, and 3 as sources in the RS bit of the CHCR of the DMAC module.

5.3.1 Interrupt Sources and Activating the DMAC

The 15 ITU interrupt sources (table 5.4) include sources that double as compare match/input capture interrupts that start up the DMAC and transfer data. The four input capture A/compare match interrupts of channels 0–3 can start the DMAC to transfer data.

Table 5.4 ITU Interrupt Sources and DMAC Startup

Channel	Interrupt Source	Description	DMAC Startup	Priority Order
0	IMIA0	Compare match or input capture A0	Yes	High
	IMIB0	Compare match or input capture B0	No	↑
	OVI0	Overflow 0	No	
1	IMIA1	Compare match or input capture A1	Yes	
	IMIB1	Compare match or input capture B1	No	
	OVI1	Overflow 1	No	
2	IMIA2	Compare match or input capture A2	Yes	
	IMIB2	Compare match or input capture B2	No	
	OVI2	Overflow 2	No	
3	IMIA3	Compare match or input capture A3	Yes	
	IMIB3	Compare match or input capture B3	No	
	OVI3	Overflow 3	No	
4	IMIA4	Compare match or input capture A4	No	
	IMIB4	Compare match or input capture B4	No	↓
	OVI4	Overflow 4	No	Low

5.3.2 Pulse Output

Pulse output is selectable for 0 output, 1 output, or toggle output (figure 5.19, table 5.5).

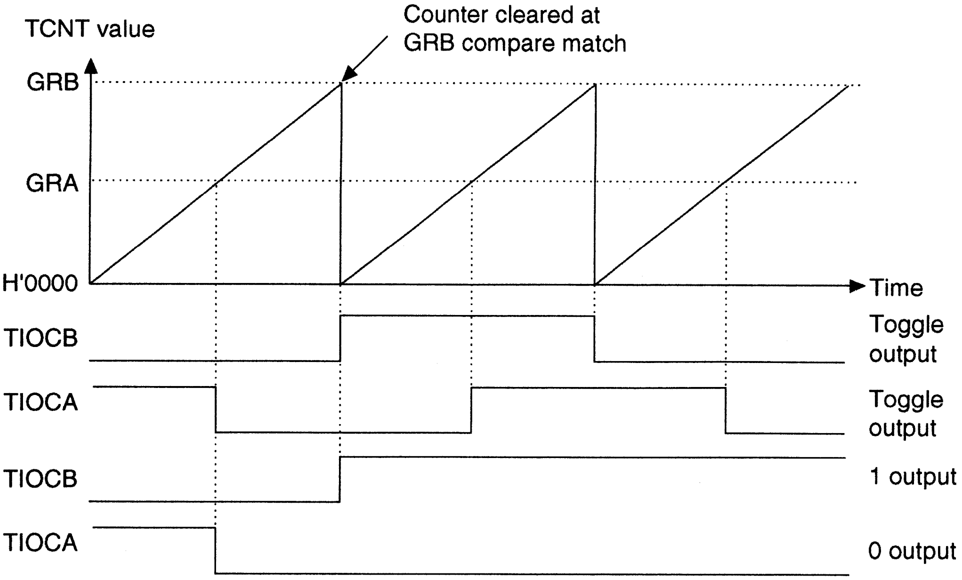


Figure 5.19 Example of Two-Phase Pulse Output

Table 5.5 Two-Phase Pulse Channel Outputs

	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4
0 Output/1 Output	Yes	Yes	Yes	Yes	Yes
Toggle Output	Yes	Yes	—	Yes	Yes

5.3.3 PWM Mode

The PWM mode is controlled using both the GRA and GRB in pairs. A single-phase PWM waveform whose cycle and duty are independently selectable can be output. By combining this mode with synchronized operation, a PWM waveform with up to five phases can be output (figure 5.20, table 5.6).

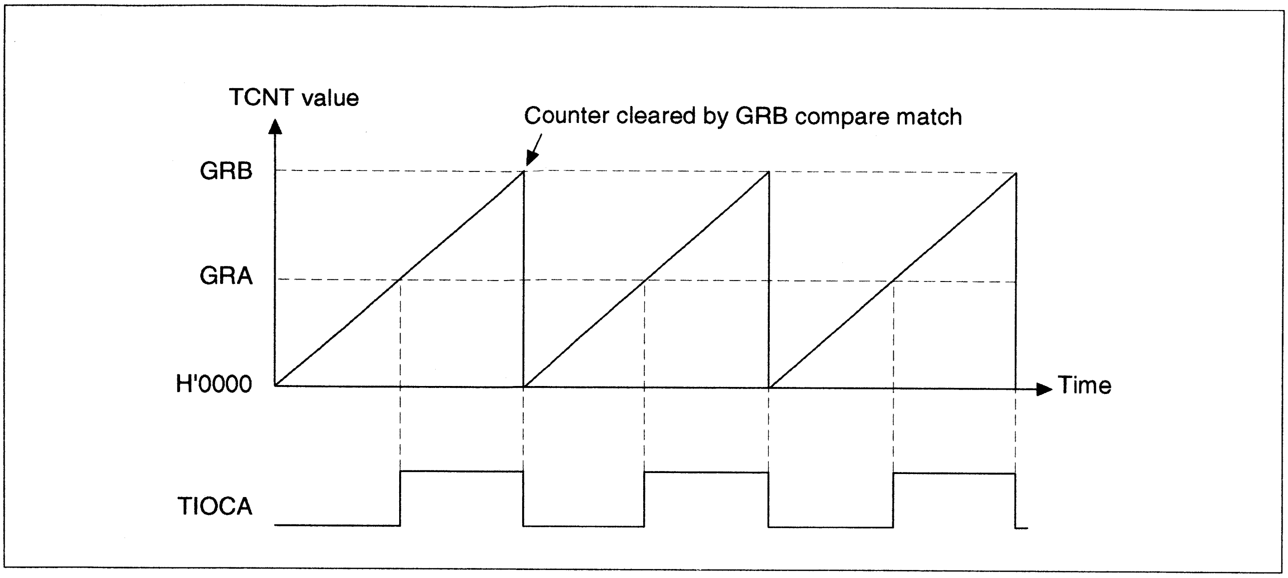


Figure 5.20 Example of Single-Phase PWM Output

Table 5.6 Single-Phase PWM Channel Output

	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4
PWM output	Yes	Yes	Yes	Yes	Yes
Output phases	Phase	Phase	Phase	Phase	Phase

5.3.4 Complementary PWM Mode Operation

Channels 3 and 4 are used in combination to output a three-phase PWM waveform in which the positive and negative phases do not overlap (figure 5.21).

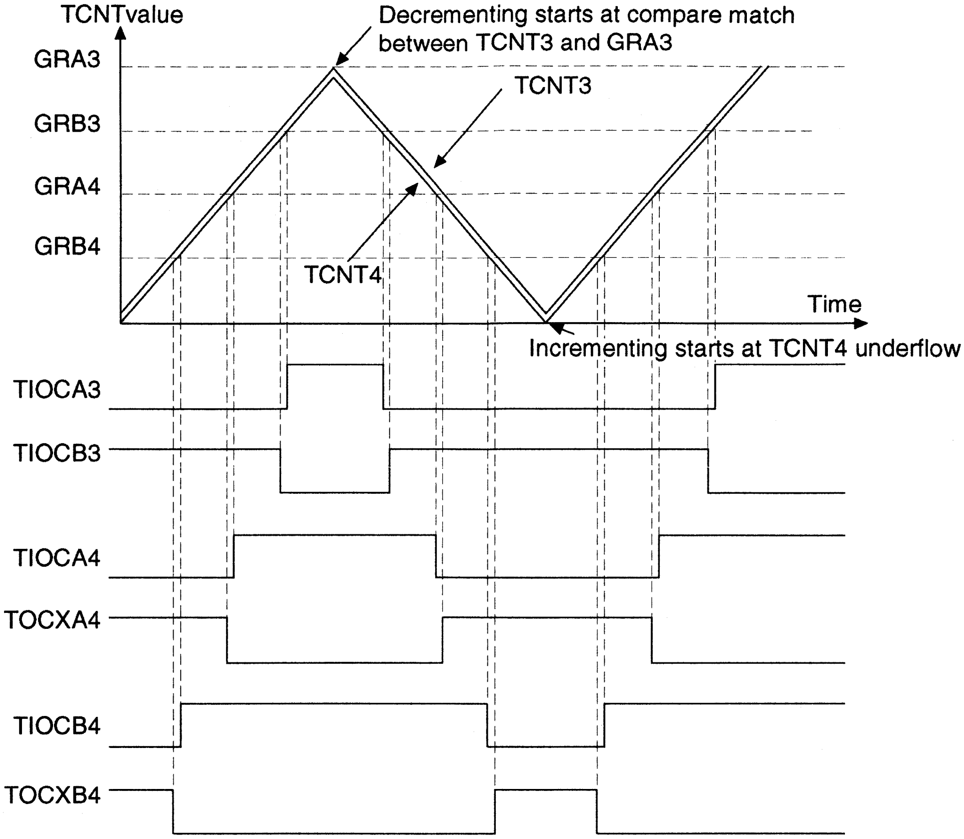

TCNTvalue
Decrementing starts at compare match between TCNT3 and GRA3
GRA3
TCNT3
GRB3
GRA4
GRB4
TCNT4
Time
Incrementing starts at TCNT4 underflow
TIOCA3
TIOCB3
TIOCA4
TOCXA4
TIOCB4
TOCXB4

Figure 5.21 Complementary PWM Mode

5.3.5 Reset-Synchronized PWM Mode Operation

Channels 3 and 4 are used in combination to output three-phase positive and negative phase PWM waveforms that share a common change point at one end (figure 5.22).

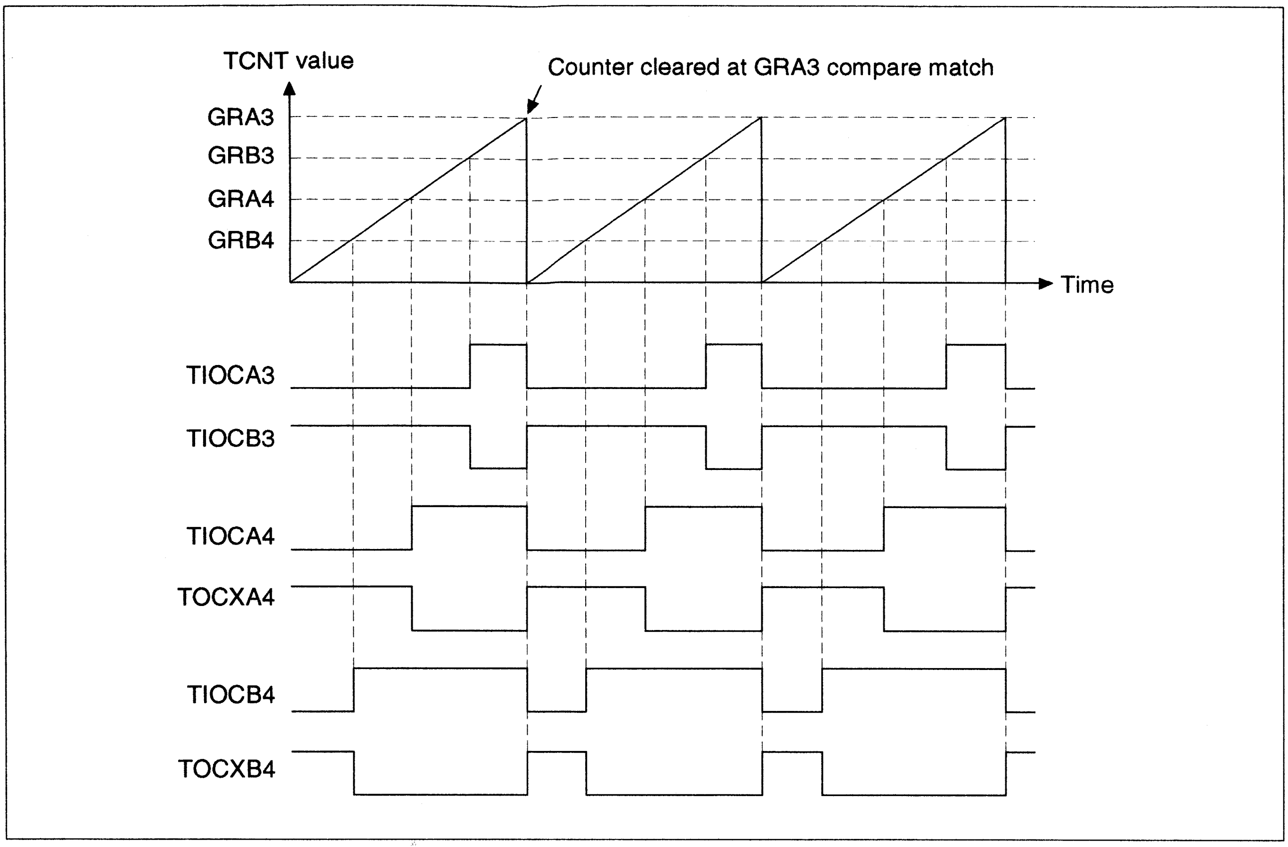


Figure 5.22 Reset-Synchronized PWM Operation

5.3.6 Pulse Measuring Function

Pulse cycles can be measured by inputting an external pulse to the input capture pin (figure 5.23). The input edge can be rising, falling, or both. The count edge can also be rising, falling, or both.

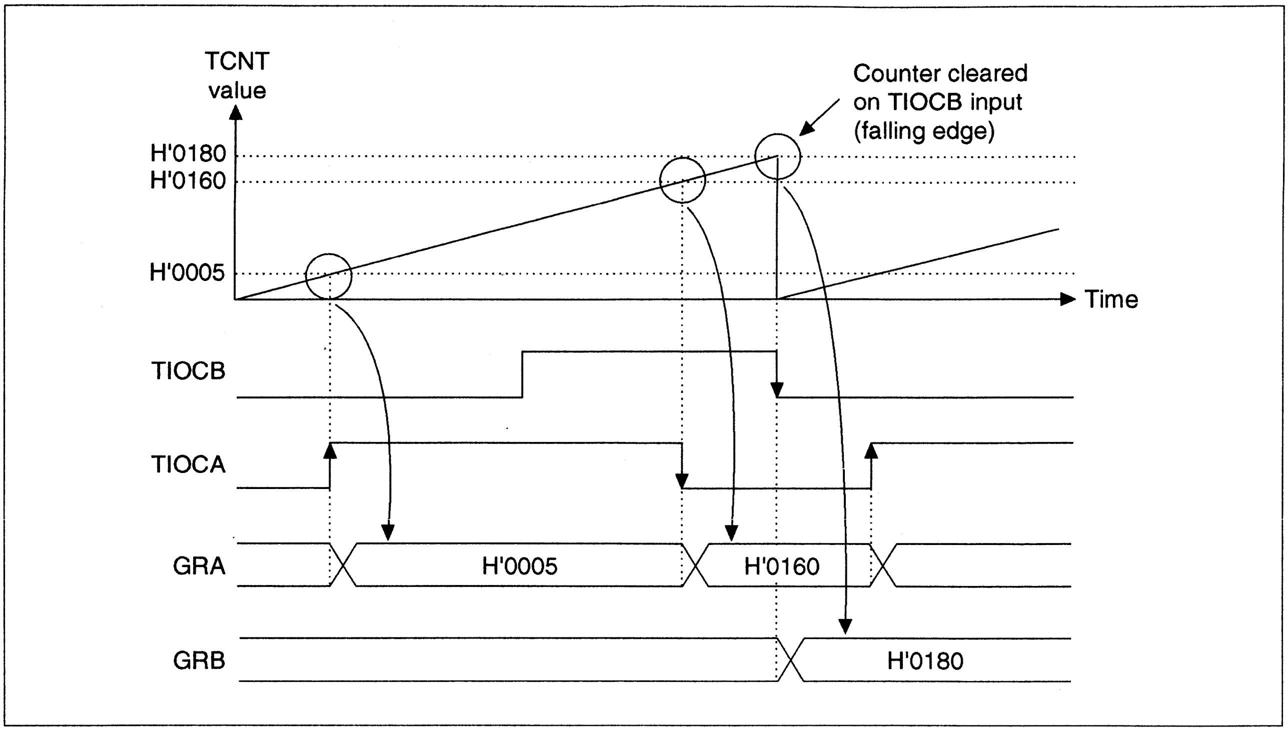
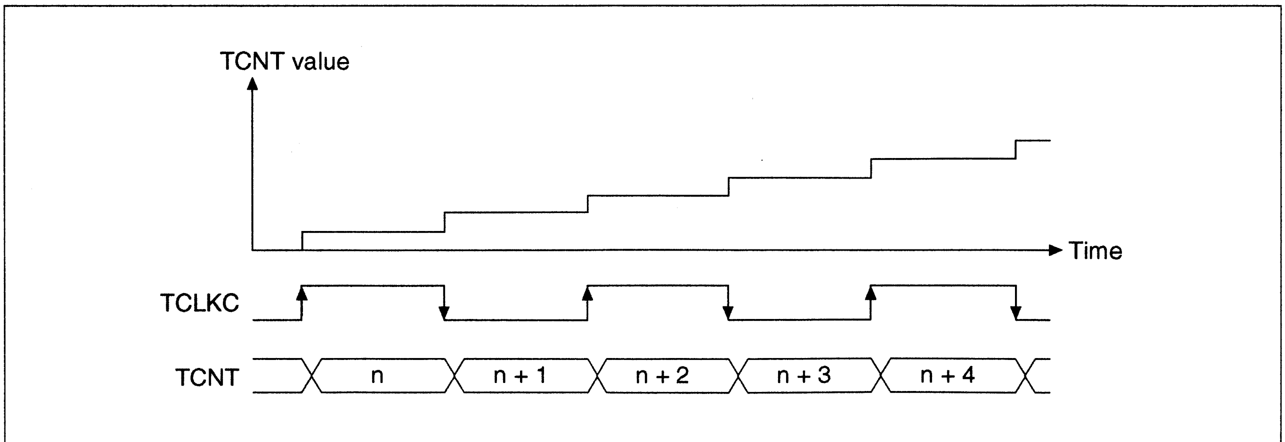


Figure 5.23 Example of Capture Input Mode Operation

5.3.7 Event Counting

The number of events can be counted by inputting an external event to TCLKA–TCLKD (figure 5.24).

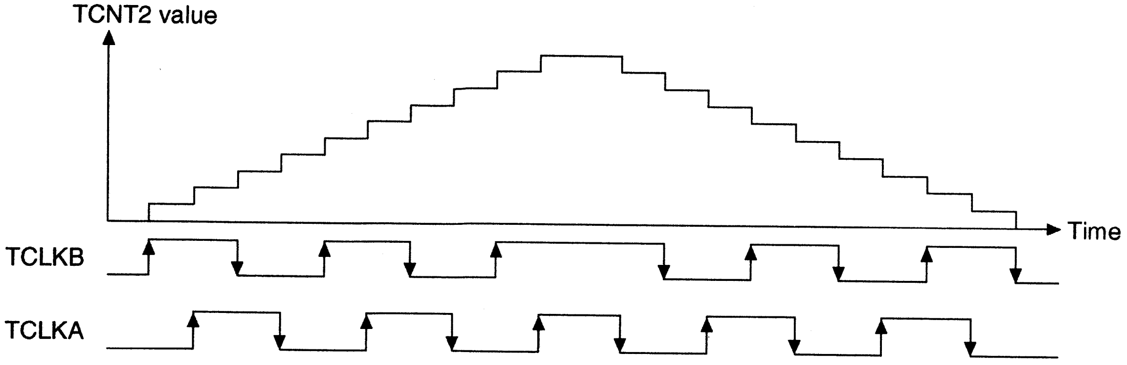


Note: In this example, the counter increments at either edge at the TCLKC pin.

Figure 5.24 Event Counting

5.3.8 Two-Phase Encoder Processing

Ascertains the phase of the motor's two-phase encoder output pulse input to TCLKA and TCLKB and increments/decrements the timer counter (figure 5.25). This function can be used on timer channel 2 only.



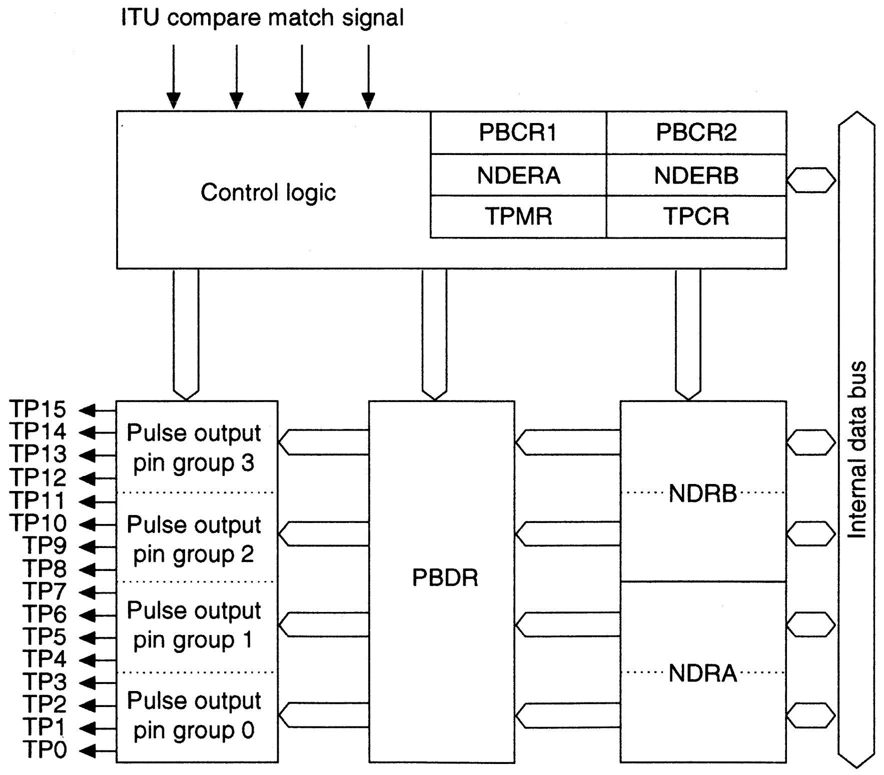
Count Direction	Increment				Decrement			
TCLKB	↑	High	↓	Low	↑	High	↓	Low
TCLKA	Low	↑	High	↓	High	↓	Low	↑

Figure 5.25 Encoder Processing

5.4 Programmable Timing Pattern Controller (TPC)

The programmable timing pattern controller (TPC) (figure 5.26) can provide pulse outputs using the 16-bit integrated-timer pulse unit (ITU) as a time base. The TPC pulse outputs are divided into 4-bit groups A, B, C, and D. These can operate simultaneously or independently. TPC features are:

- Maximum 16-bit data can be output. TPC output can be enabled on a bit-by-bit basis.
- Output trigger signals can be selected by group from the 4-channel compare-match signals of the 4 ITU channels.
- Output trigger signals can be selected in 4-bit groups to provide up to 4 different 4-bit outputs.
- A non-overlap interval can be set (figure 5.27).
- The compare-match signals of the ITU can activate the DMA controller and control the TPC.



TPMR: TPC output mode register
TPCR: TPC output control register
NDERB: next data enable register B
NDERA: next data enable register A
PBDR: port B data register

PBCR1: Port B control register 1
PBCR2: Port B control register 2
NDRB: next data register B
NDRA: next data register A

Figure 5.26 TPC Block Diagram

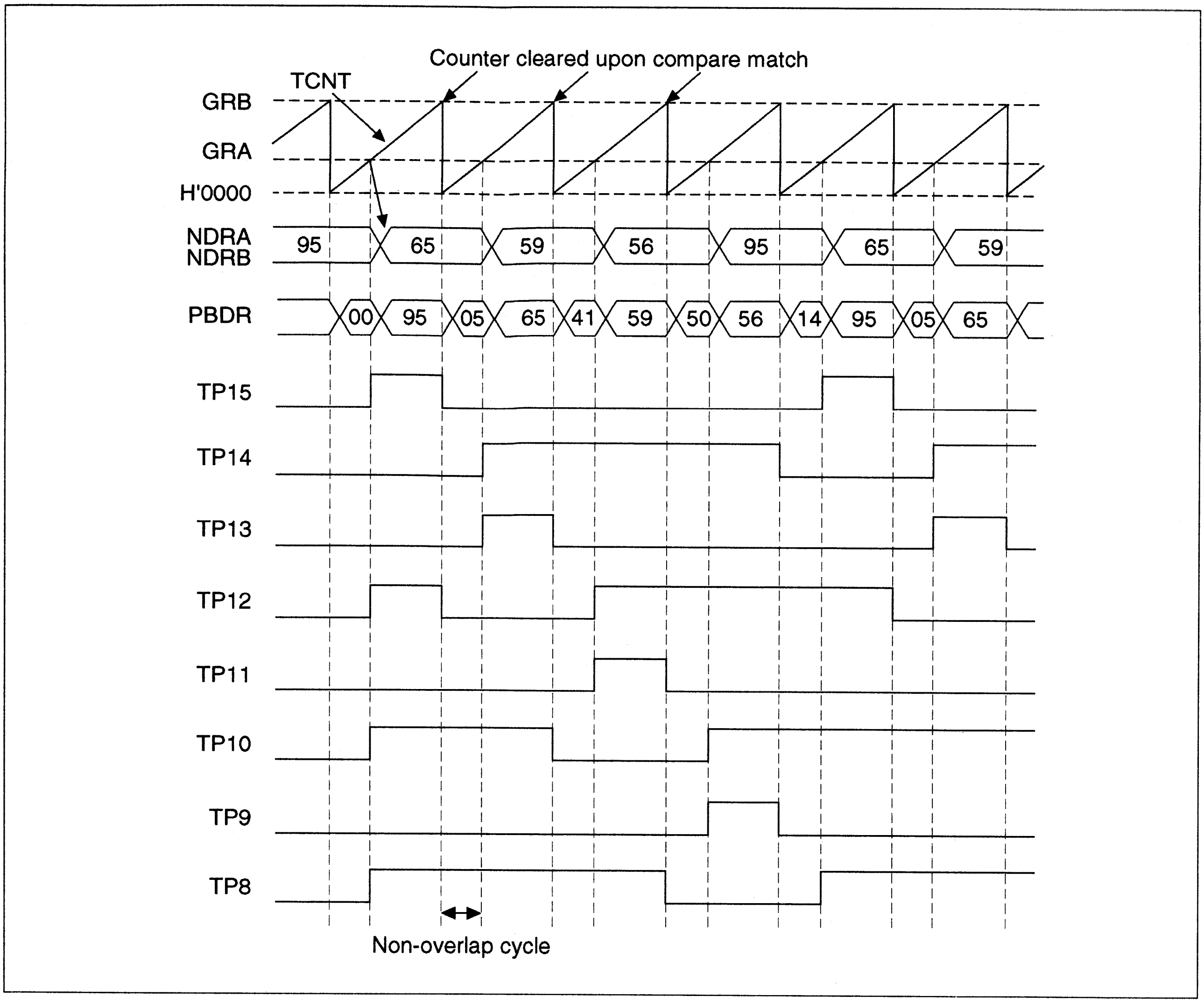
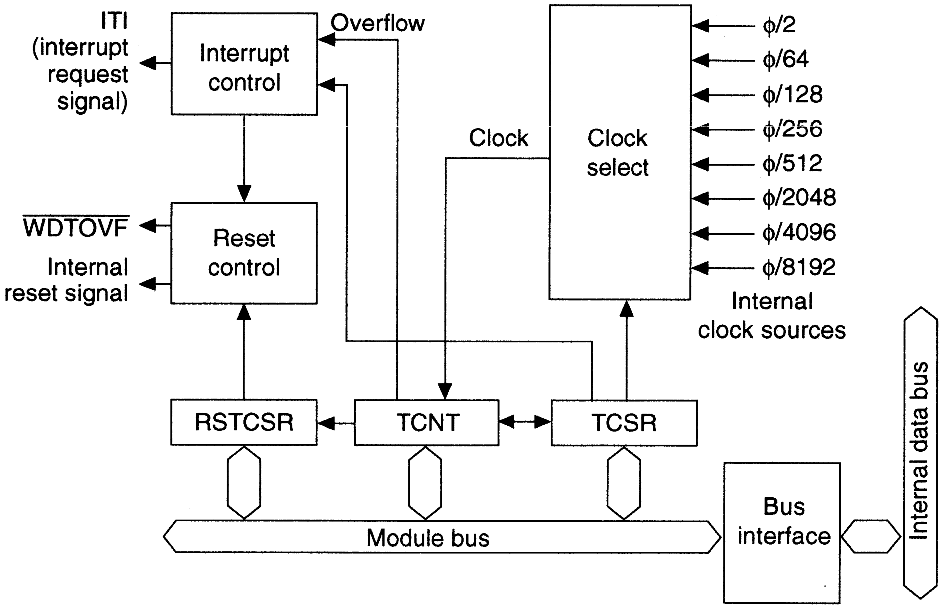


Figure 5.27 Example of Four-Phase Complementary Non-Overlap Pulse Output

5.5 Watchdog Timer (WDT)

The watchdog timer (WDT) (figure 5.28) monitors system operations. If a system becomes uncontrolled and the timer counter overflows without being rewritten correctly by the CPU within a prescribed length of time, the WDT can reset the system internally. WDT features are:

- Operates in watchdog timer mode (figure 5.29) or interval timer mode (figure 5.30).
- Outputs overflow signal $\overline{\text{WDTOVF}}$ when the counter overflows in the watchdog timer mode. The user can select power-on or manual for the internal chip reset.
- Generates an interval timer interrupt when the counter overflows in the interval timer mode.
- Selects from eight counter clock sources for input to the counter.
- Controls the system clock settling time when recovering from the standby mode.



TCSR: timer control/status register
TCNT: timer counter
RSTCSR: reset control/status register

Figure 5.28 Watchdog Timer Block Diagram

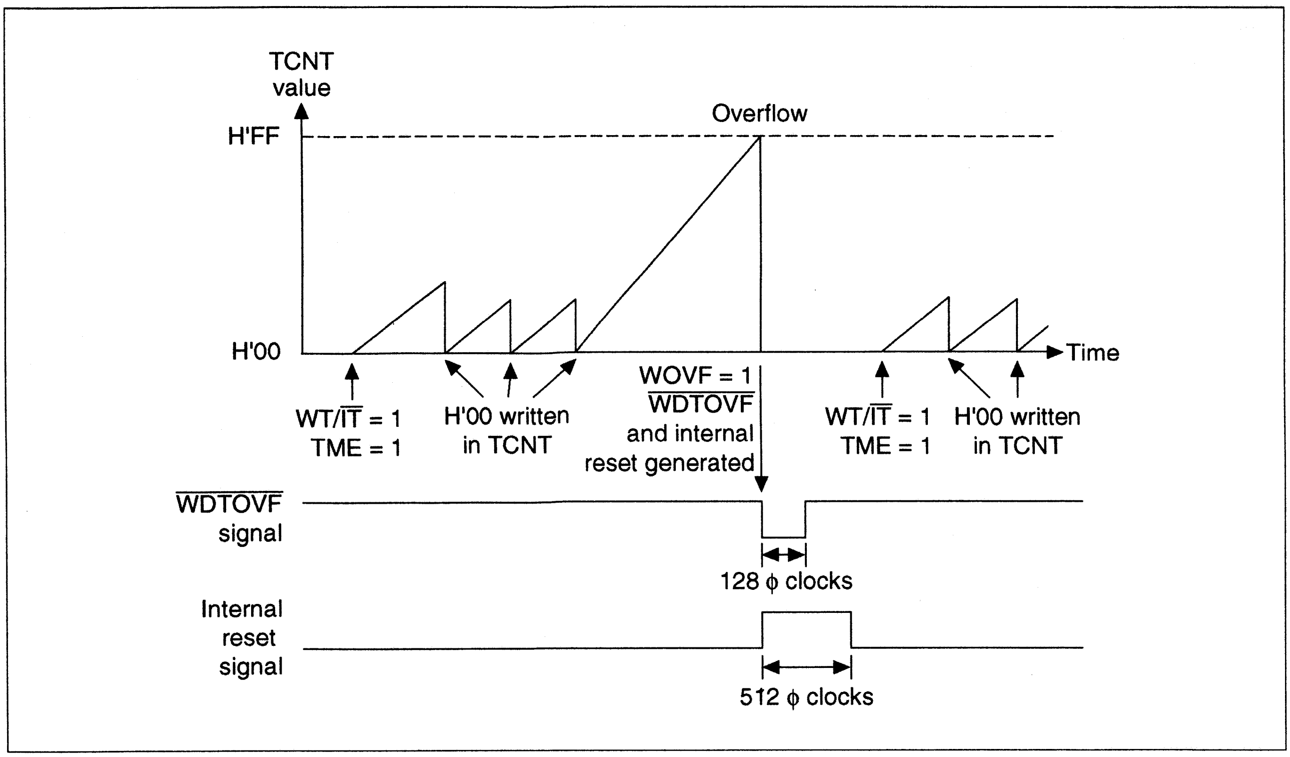


Figure 5.29 Operation in Watchdog Timer Mode

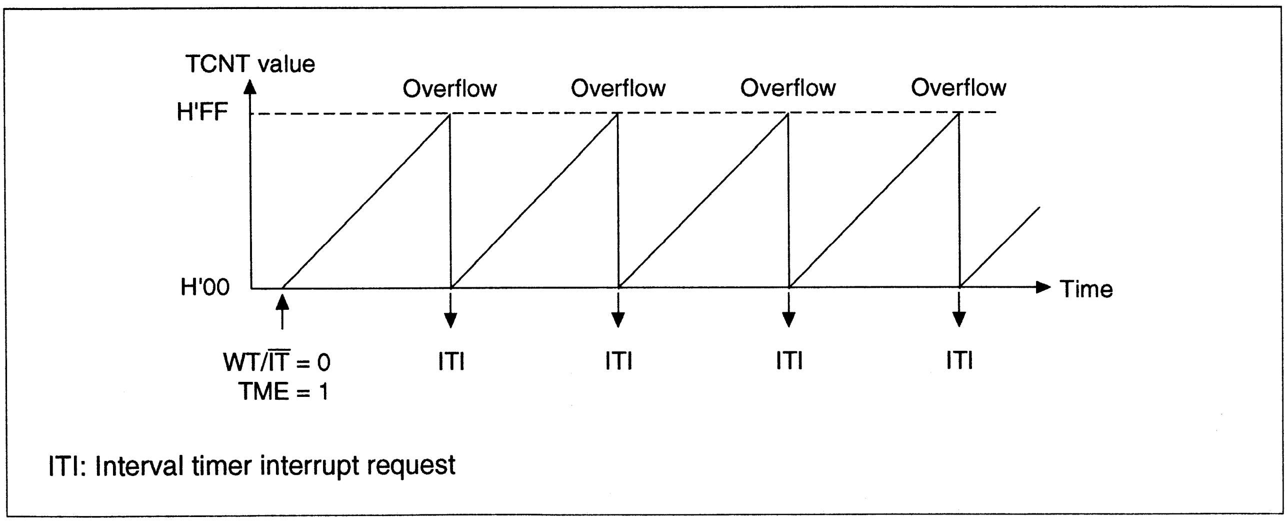


Figure 5.30 Operation in Interval Timer Mode

5.6 Serial Communications Interface (SCI)

The serial communication interface (SCI) has two built-in channels and supports both asynchronous and clocked synchronous serial communication. The SCI is an I/O module that performs serial communication with external devices in either the asynchronous or clocked synchronous modes. It also has a multiprocessor communication function for serial communication among two or more processors. Figure 5.31 is a functional block diagram of the SCI. SCI features are:

- Selection of asynchronous or clocked synchronous mode
- Full duplex communication
- Double buffering of data registers enables continuous data transfer is possible in both the transmit and receive directions.
- Built-in dedicated baud rate generator with selectable bit rates (any rate)
- Internal baud rate generator or external clock source (SCK pin)
- Detects parity, overrun, and framing errors
- 4 types of interrupts: Transmit-data-empty, transmit-end, receive-data-full, and receive-error interrupts are requested independently.
- Maximum transfer rates of 625 kbits/s (asynchronous) and 5 Mbits/s (clocked-synchronous)

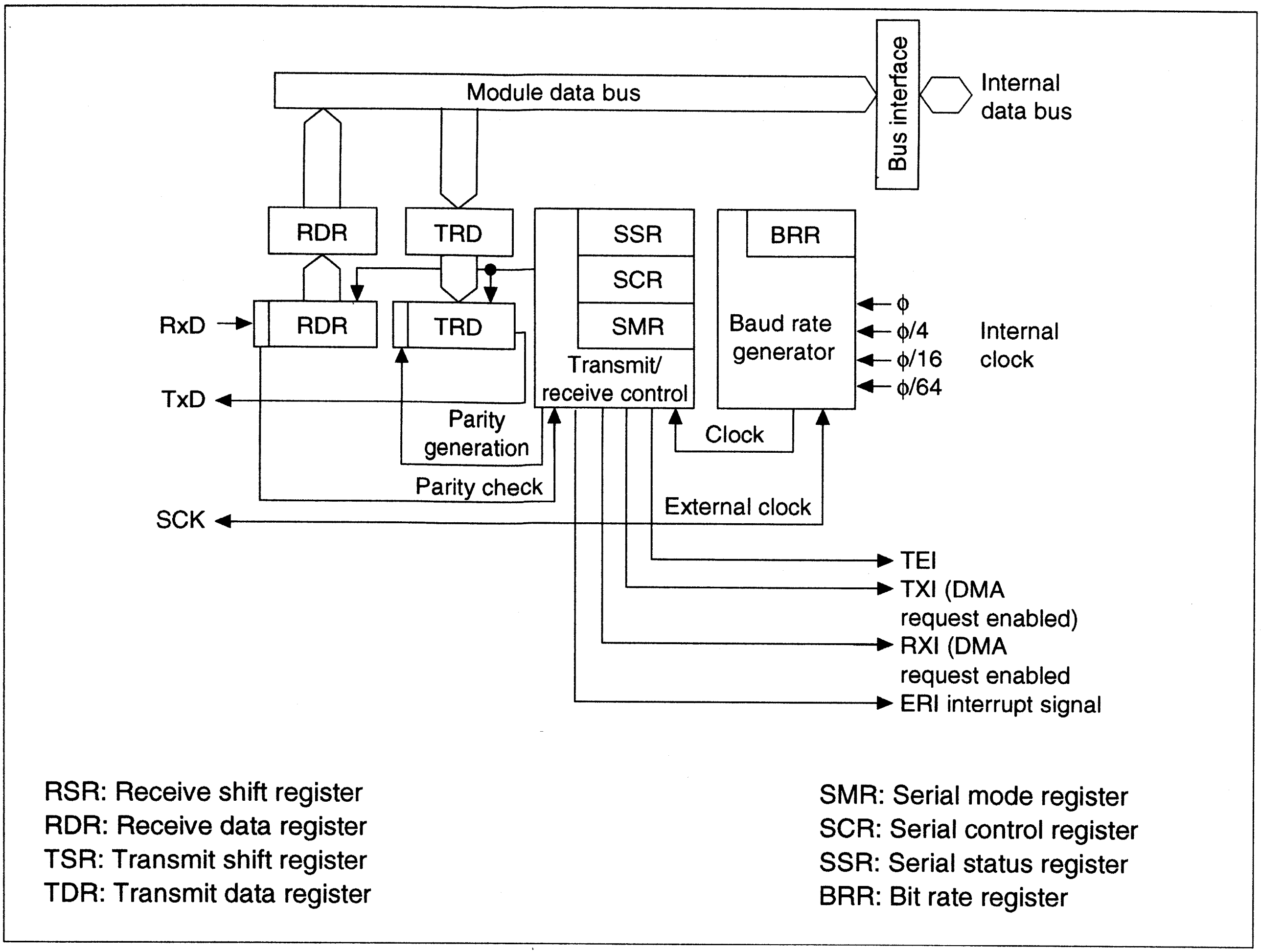
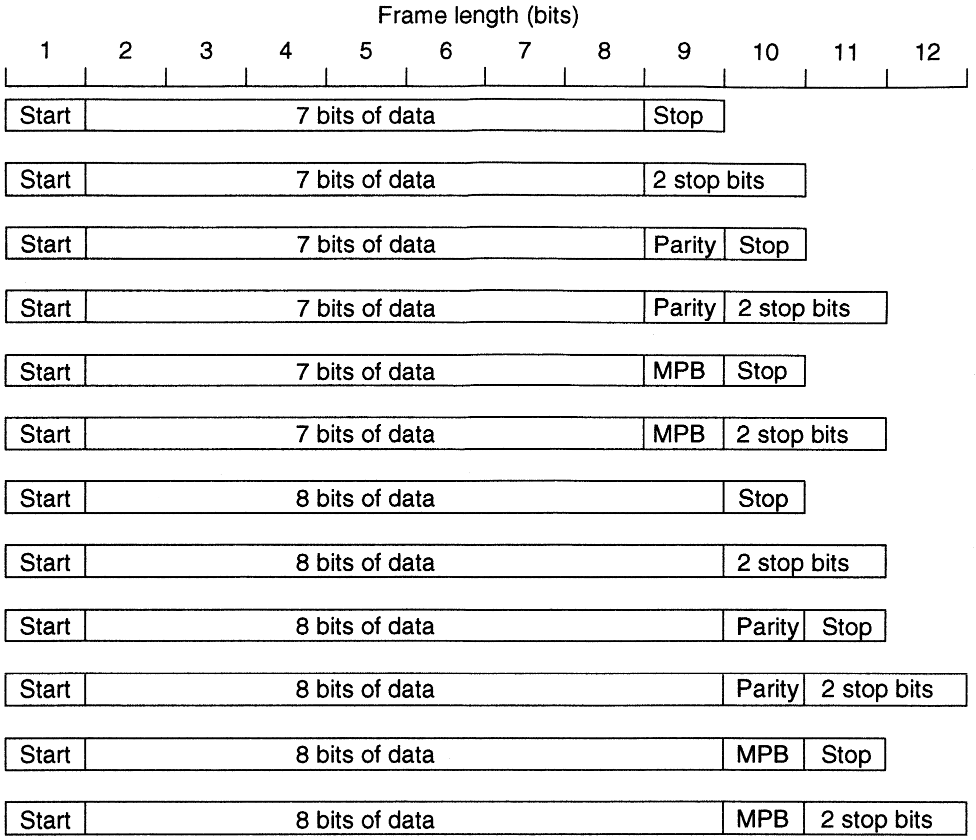


Figure 5.31 SCI Block Diagram

5.6.1 Asynchronous Mode

In the asynchronous mode, characters are synchronized individually for serial communication using start and stop bits.

- Twelve types of transfer formats (figure 5.32)
  - — Data length 7 or 8 bits
  - — Stop bit length 1 or 2 bits
  - — Parity and multiprocessor bits: Even parity, odd parity, multiprocessor bit, or nothing
- An internal baud rate generator or external clock source (SCK pin) can be selected as the clock source (table 5.8).


Frame length (bits)
1 2 3 4 5 6 7 8 9 10 11 12
Start | 7 bits of data | Stop
Start | 7 bits of data | 2 stop bits
Start | 7 bits of data | Parity | Stop
Start | 7 bits of data | Parity | 2 stop bits
Start | 7 bits of data | MPB | Stop
Start | 7 bits of data | MPB | 2 stop bits
Start | 8 bits of data | Stop
Start | 8 bits of data | 2 stop bits
Start | 8 bits of data | Parity | Stop
Start | 8 bits of data | Parity | 2 stop bits
Start | 8 bits of data | MPB | Stop
Start | 8 bits of data | MPB | 2 stop bits

MPB is the multiprocessor bit

Figure 5.32 Transfer Format Frame Length in Asynchronous Mode

5.6.2 Clocked Synchronous Mode

The clocked synchronous mode synchronizes communication with clock pulses for serial communication. It is best suited to continuous fast serial transmission (table 5.7).

- Data length is 8 bits/character (figure 5.33)
- Detects overrun errors
- Select either the onchip baud rate generator or the external clock that comes from the SCK pin as the clock source for the transmit/receive (tables 5.9, 5.10)
- LSB first system: Data is sent from the least significant bit
- Can communicate with LSIs that have the clocked synchronous mode, such as the H8/500, H8/300 and HD64180

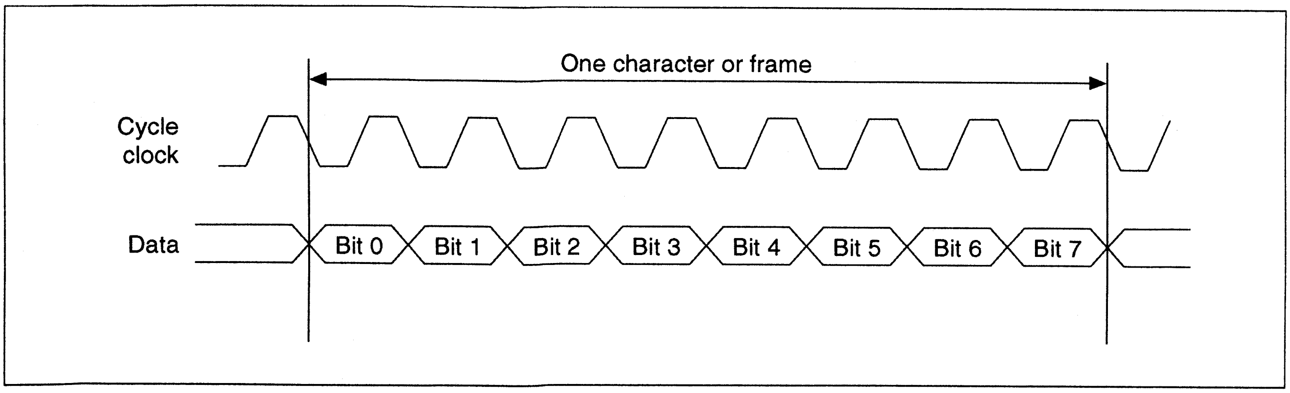


Figure 5.33 Example of Data Format in Clocked Synchronous Mode

Table 5.7 Example of Setting BRR for Baud Rate (Asynchronous Mode)

	f = 19.6608 MHz			f = 20 MHz		
Bit Rate (Bit/s)	**Baud Rate Generator Input Clock**	**BRR Setting**	**Error (%)**	**Baud Rate Generator Input Clock**	**BRR Setting**	**Error (%)**
110	φ/64	86	0.31348	φ/64	88	–0.249
150	φ/16	255	0	φ/64	64	0.1603
300	φ/16	127	0	φ/16	129	0.1603
600	φ/4	255	0	φ/16	64	0.1603
1200	φ/4	127	0	φ/4	129	0.1603
2400	φ	255	0	φ/4	64	0.1603
4800	φ	127	0	φ	129	0.1603
9600	φ	63	0	φ	64	0.1603
19200	φ	31	0	φ	32	–1.357
31250	φ	19	–1.696	φ	19	0
38400	φ	15	0	φ	15	1.7253

The BRR setting is calculated as follows:

$$N = [f/(64 \times 2^{2n-1} \times B)] \times 10^6 - 1$$

N: BRR setting for baud rate generator ($0 \leq N \leq 255$)
f: operating frequency (MHz)
B: bit rate (bit/s)
n: baud rate generator clock source (n = 0, 1, 2, 3)

Table 5.8 Clock Source Selection (Asynchronous Mode)

SCK1	SCK0	n	Clock Source
0	0	0	ϕ
0	1	1	$\phi/4$
1	0	2	$\phi/16$
1	1	3	$\phi/24$

Table 5.9 Examples of Bit Rates and BRR Settings in Synchronous Mode

	f (MHz):											
Bit Rate	2		4		8		10		16		20	
(bit/s)	n	N	n	N	n	N	n	N	n	N	n	N
110	3	70	—	—	—	—	—	—	—	—	—	—
250	2	124	2	249	3	124	—	—	3	249	—	—
500	1	249	2	124	2	249	—	—	3	124	—	—
1 k	1	124	1	249	2	124	—	—	2	249	—	—
2.5 k	0	199	1	99	1	199	1	249	2	99	2	124
5 k	0	99	0	199	1	99	1	124	1	199	1	249
10 k	0	49	0	99	0	199	0	249	1	99	1	124
25 k	0	19	0	39	0	79	0	99	0	159	0	199
50 k	0	9	0	19	0	39	0	49	0	79	0	99
100 k	0	4	0	9	0	19	0	24	0	39	0	49
250 k	0	1	0	3	0	7	0	9	0	15	0	19
500 k	0	0*	0	1	0	3	0	4	0	7	0	9
1 M			0	0*	0	1	—	—	0	3	0	4
2.5 M					—	—	0	0*	—	—	0	1
5 M									—	—	0	0*

Note: Continuous transmit/receive not possible

The BRR setting (table 5.9) is calculated as follows:

$$N = [f/(8 \times 2^{2n-1} \times B)] \times 10^6 - 1$$

B: bit rate (bit/s)

N: BRR setting for baud rate generator ($0 \leq N \leq 255$)

f: operating frequency (MHz)

n: baud rate generator clock source (n = 0, 1, 2, 3) (see table 5.10)

Table 5.10 Clock Source Selection (Synchronous Mode)

SCK1	SCK0	n	Clock Source
0	0	0	ϕ
0	1	1	ϕ/4
1	0	2	ϕ/16
1	1	3	ϕ/24

5.6.3 Multiprocessor Communication

The multiprocessor communication function enables several processors to share a single serial communication line by using a format with an additional multiprocessor bit (figure 5.34).

- The transmitting processor starts by sending the ID of the receiving processor with which it wants to communicate as data with the multiprocessor bit (MPB) set. Next the transmitting processor sends transmit data with the multiprocessor bit cleared.
- Receiving processors skip incoming data until they receive data with the multiprocessor bit set. When they receive data with the multiprocessor bit set, receiving processors compare the data with their IDs.
- The receiving processor with a matching ID continues to receive further incoming data. Processors with IDs not matching the received data skip further incoming data until they again receive data with the multiprocessor bit set.

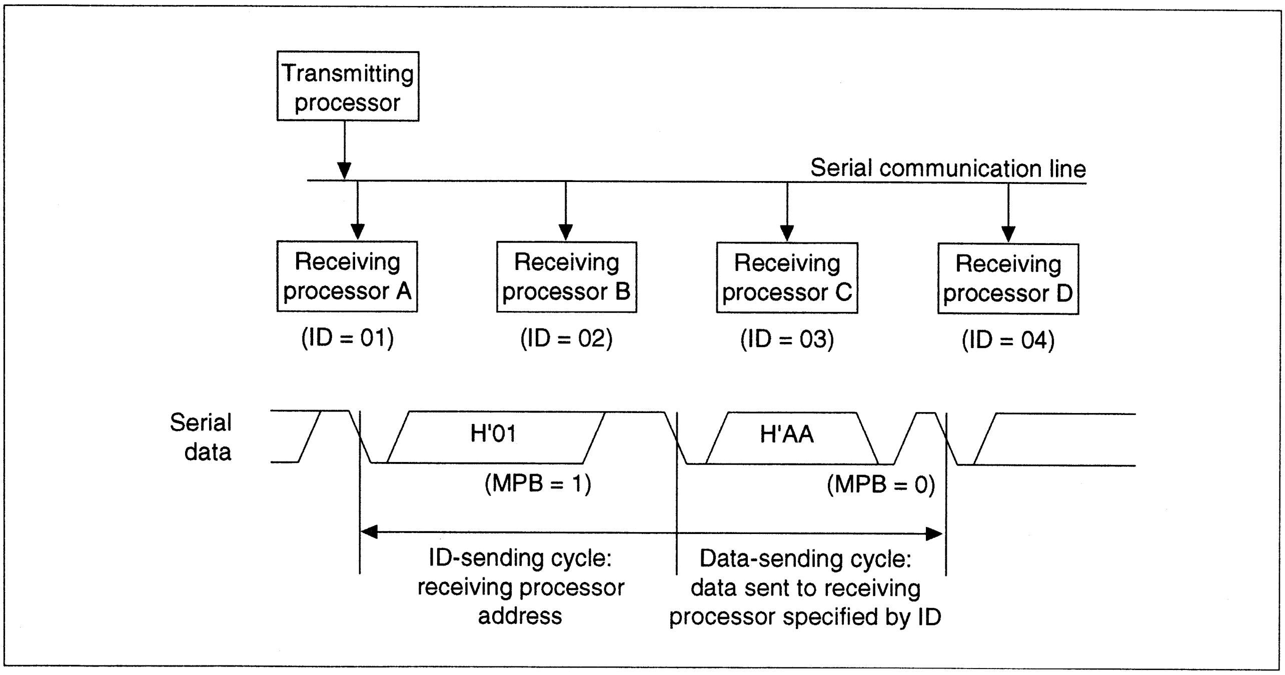


Figure 5.34 Example of Communication among Processors Using Multiprocessor Format (Sending Data H'AA to Receiving Processor A)

5.6.4 SCI Interrupt Sources and the DMAC

The SCI has four interrupt sources (table 5.11) in each channel: transmit-end (TEI), receive-error (ERI), receive-data-full (RXI), and transmit-data-empty (TXI).

The channel 0 RXI and TXI start up the DMAC channel for data to transfers. Using the receive data full interrupt to make the DMAC read the receive data and using the transmit data empty interrupt to make the DMAC write the next transmit data enables continuous transmit/receive without burdening the software.

Table 5.11 SCI Interrupt Sources

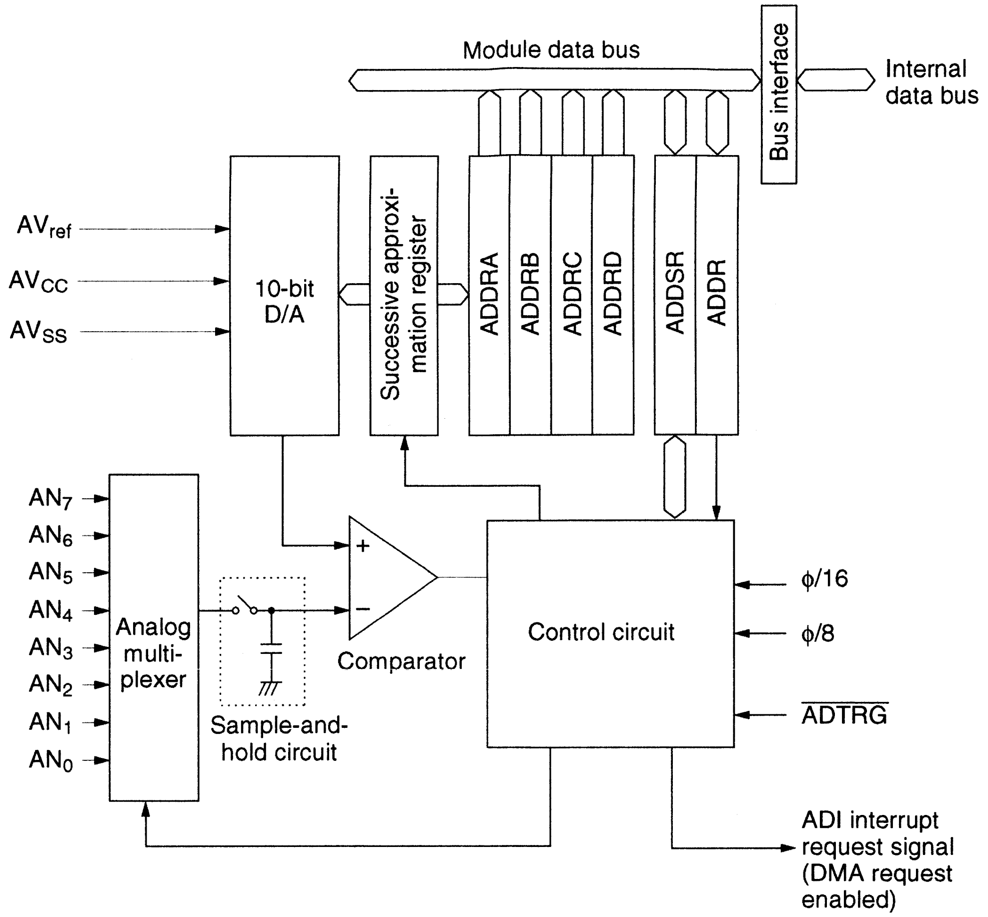
Interrupt Source	Description	DMAC Availability	Priority
ERI	Receive error (ORER, PER, or FER)	No	High
RXI	Receive data register full (RDRF)	Yes	↑
TXI	Transmit data register empty (TDRE)	Yes	↓
TEI	Transmit end (TEND)	No	Low

Note: The receive error may be an overrun error, a framing error or a parity error.

5.7 A/D Converter [SH7032, SH7034]

The SH microprocessor includes an analog-to-digital converter module with 10-bit resolution that can be programmed for input of up to 8 channels of analog signals (figure 5.35).

- 10-bit resolution
- 8 analog input channels
- Rapid conversion time: 6.7 µs per channel (at 20 MHz)
- Single mode or scan mode (selectable)
  - — Single mode: One-channel A/D conversion
  - — Scan mode: A/D conversion repeated on 1 to 4 channels
- Sample-and-hold circuit
- External reference voltage pin
- External trigger input can start A/D conversion.
- Four 16-bit data registers: A/D conversion results are transferred to and stored in the data registers corresponding to channels.
- A/D interrupt (ADI) request to the CPU can be generated at the end of each A/D conversion cycle (DMA request enabled).



ADCR:	A/D control register
ADCSR:	A/D control/status register
A DDRA:	A/D data register A: ADDRA H/ADDRA L
A DDRB:	A/D data register B: ADDRB H/ADDRB L
A DDRC:	A/D data register C: ADDRC H/ADDRC L
A DDRD:	A/D data register D: ADDRD H/ADDRD L

Figure 5.35 A/D Converter Block Diagram

A/D conversion is performed by the successive approximation method with 10-bit resolution. The analog input of the 8 channels is selected with the channel selection bits (CH2–CH0) of the ADCSR (table 5.12).

Table 5.12 A/D Conversion

			Selected Channels	
Bit 2 , CH2	**Bit 1, CH1**	**Bit 0, CH0**	**Single Mode**	**Scan Mode**
0	0	0	AN0	
		1	AN1	AN0 and AN1
	1	0	AN2	AN0 to AN2
		1	AN3	AN0 to AN3
1	0	0	AN4	
		1	AN5	AN4 and AN5
	1	0	AN6	AN4 to AN6
		1	AN7	AN4 to AN7

Single Mode: In the single mode, A/D conversion is controlled by the CPU and performed on a single channel. A/D conversion starts when the ADST bit of the A/D control/status register (ADCSR) is set. When the conversion is completed, the conversion end flag (ADF bit) is set. If the interrupt enable bit (ADIE) of the ADCSR is also set, an A/D conversion interrupt (ADI) is generated, and the results of the A/D conversion processed by the interrupt service routine.

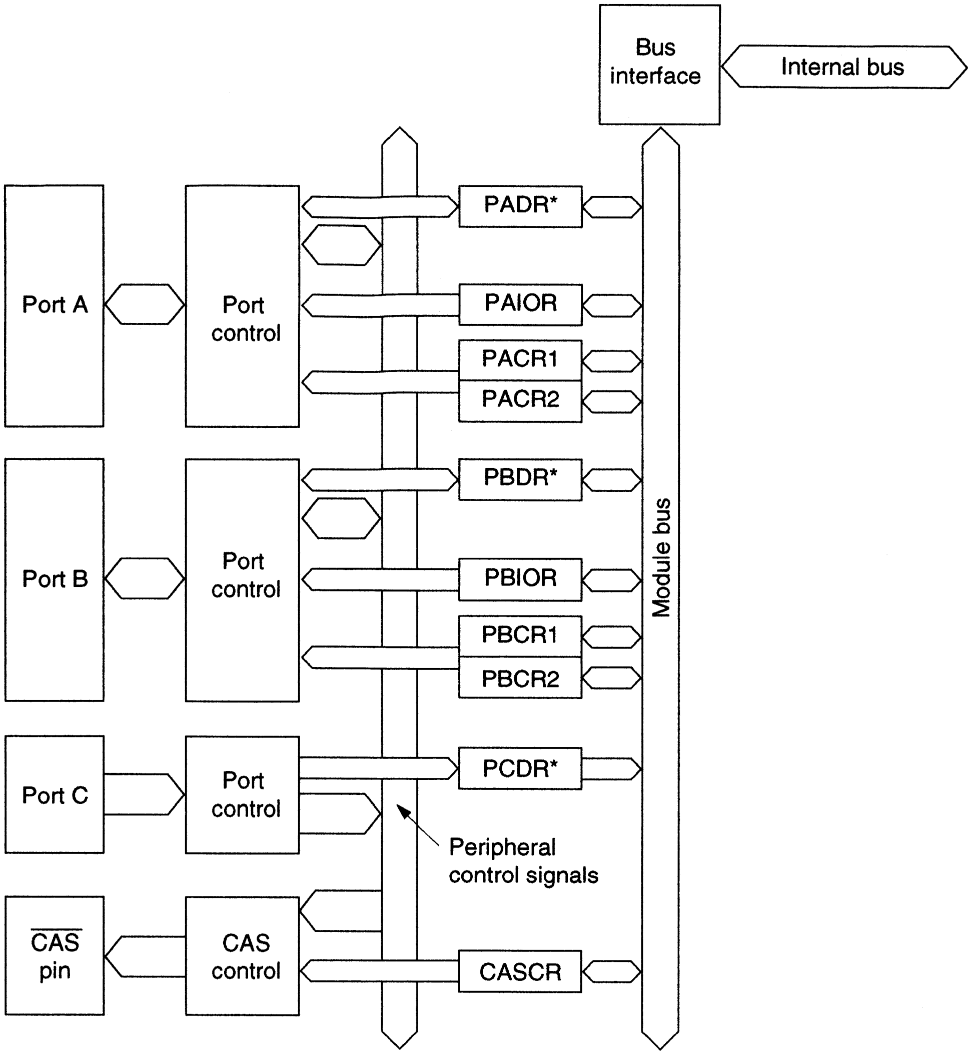
Scan Mode: The scan mode can be used to monitor analog inputs on 1 to 4 channels. When the ADST bit is set, A/D conversion starts with the first channel. If the scan group includes more than one channel, the conversion of the second channel begins as soon as the conversion of the first channel ends. If there is only one channel, the first channel begins converting again. Conversion of the selected channels continues cyclically until the ADST bit is cleared. The conversion results are stored in the data registers corresponding to the selected channels (ADDRA–ADDRD).

The ADST bit can be set not only through software, but using an external trigger signal ($\overline{\text{ADTRG}}$) as well.

Note: The SH7032 and SH7034 have A/D converters. The SH7020 and SH7021 do not.

5.8 Pin Function Controller (PFC)

The pin function controller (PFC) is composed of registers for selecting the functions of the multiplexed pins of port A, port B, port C, and the column address strobe pin and for selecting their direction (input/output) (figure 5.36). The pin function and input/output direction can be selected for each pin individually. Tables 5.13–5.16 list pin functions.



Note: I/O register

PAIOR:	Port A I/O register
PACR1:	Port A control register 1
PACR2:	Port A control register 2
PBIOR:	Port B I/O register
PBCR1:	Port B control register 1
PBCR2:	Port B control register 2
CASCR:	Column address strobe pin control register
PADR:	Port A data register
PBDR:	Port B data register
PCDR:	Port C data register

Note: The SH7032 and SH7034 have port C and the PCDR. The SH7020 and SH7021 do not.

Figure 5.36 Pin Function Controller Block Diagram

Table 5.13 Port A Pin Functions

Mode Bit = 00 (Port A Input Pin)	Mode Bit = 01	Mode Bit = 10	Mode Bit = 11
PA15 I/O (port)	$\overline{\text{IRQ3}}$ input (INTC)	—	$\overline{\text{DREQ1}}$ input (DMAC)
PA14 I/O (port)	$\overline{\text{IRQ2}}$ input (INTC)	—	DACK1 output (DMAC)
PA13 I/O (port)	$\overline{\text{IRQ1}}$ input (INTC)	TCLKB input (ITU)	$\overline{\text{DREQ0}}$ input (DMAC)
PA12 I/O (port)	$\overline{\text{IRQ0}}$ input (INTC)	TCLKA input (ITU)	DACK0 output (DMAC)
PA11 I/O (port)	DPH I/O (BSC)	TIOCB1 I/O (ITU)	—
PA10 I/O (port)	DPL I/O (BSC)	TIOCA1 I/O (ITU)	—
PA9 I/O (port)	$\overline{\text{AH}}$ output (BSC)	$\overline{\text{ADTRG}}$ input (A/D)	$\overline{\text{IRQOUT}}$ output (INTC)
PA8 I/O (port)	—	—	$\overline{\text{BREQ}}$ input (system)
PA7 I/O (port)	—	—	$\overline{\text{BACK}}$ output (system)
PA6 I/O (port)	—	—	$\overline{\text{RD}}$ output (BSC)
PA5 I/O (port)	—	—	$\overline{\text{WRH}}$ output (BSC) ($\overline{\text{LBS}}$ output (BSC))
PA4 I/O (port)	—	—	$\overline{\text{WRL}}$ output (BSC) ($\overline{\text{WR}}$ output (BSC))
PA3 I/O (port)	$\overline{\text{CS7}}$ output (BSC)	$\overline{\text{WAIT}}$ input (BSC)	—
PA2 I/O (port)	$\overline{\text{CS6}}$ output (BSC)	TIOCB0 I/O (ITU)	—
PA1 I/O (port)	$\overline{\text{CS5}}$ output (BSC)	$\overline{\text{RAS}}$ output (BSC)	—
PA0 I/O (port)	$\overline{\text{CS4}}$ output (BSC)	TIOCA0 I/O (ITU)	—

Table 5.14 Port B Pin Functions

Mode Bit = 00 (Port B Input Pin)	Mode Bit = 01	Mode Bit = 10	Mode Bit = 11
PB15 I/O (port)	$\overline{\text{IRQ7}}$ input (INTC)	—	TP15 output (TPC)
PB14 I/O (port)	$\overline{\text{IRQ6}}$ input (INTC)	—	TP14 output (TPC)
PB13 I/O (port)	IRQ5 input (INTC)	SCK1 I/O (SCI)	TP13 output (TPC)
PB12 I/O (port)	IRQ4 input (INTC)	SCK0 I/O (SCI)	TP12 output (TPC)
PB11 I/O (port)	—	TxD1 output (SCI)	TP11 output (TPC)
PB10 I/O (port)	—	RxD1 input (SCI)	TP10 output (TPC)
PB9 I/O (port)	—	TxD0 output (SCI)	TP9 output (TPC)
PB8 I/O (port)	—	RxD0 input (SCI)	TP8 output (TPC)
PB7 I/O (port)	TCLKD input (ITU)	TOCXB4 output (ITU)	TP7 output (TPC)
PB6 I/O (port)	TCLKC input (ITU)	TOCXA4 output (ITU)	TP6 output (TPC)
PB5 I/O (port)	—	TIOCB4 I/O (ITU)	TP5 output (TPC)
PB4 I/O (port)	—	TIOCA4 I/O (ITU)	TP4 output (TPC)
PB3 I/O (port)	—	TIOCB3 I/O (ITU)	TP3 output (TPC)
PB2 I/O (port)	—	TIOCA3 I/O (ITU)	TP2 output (TPC)
PB1 I/O (port)	—	TIOCB2 I/O (ITU)	TP1 output (TPC)
PB0 I/O (port)	—	TIOCA2 I/O (ITU)	TP0 output (TPC)

Table 5.15 Port C Pin Functions [SH7032, SH7034]

Port C Input Pin	A/D Converter Analog Input Pin (Channel Selected by A/D Converter)
PC7 input (port)	AN7 input (A/D)
PC6 input (port)	AN6 input (A/D)
PC5 input (port)	AN5 input (A/D)
PC4 input (port)	AN4 input (A/D)
PC3 input (port)	AN3 input (A/D)
PC2 input (port)	AN2 input (A/D)
PC1 input (port)	AN1 input (A/D)
PC0 input (port)	AN0 input (A/D)

Table 5.16 Column Address Strobe Pin Functions

Mode Bit = 01	Mode Bit = 10
$\overline{\text{CS1}}$ output (BSC)	$\overline{\text{CASH}}$ output (BSC)
$\overline{\text{CS3}}$ output (BSC)	$\overline{\text{CASL}}$ output (BSC)

5.9 I/O Ports

There are three ports, A, B, and C. Ports A and B are 16-bit I/O ports, while port C is an 8-bit input port. The pins of the ports are all multiplexed for use as general-purpose I/Os (or general-purpose inputs in the case of port C) or for other functions. (Use the pin function controller (PFC) to select the function of multiplexed pins.) Ports A, B, and C each have one data register for storing pin data.

Table 5.17 lists the data read to or written to the port data registers.

Table 5.17 Port Register Data

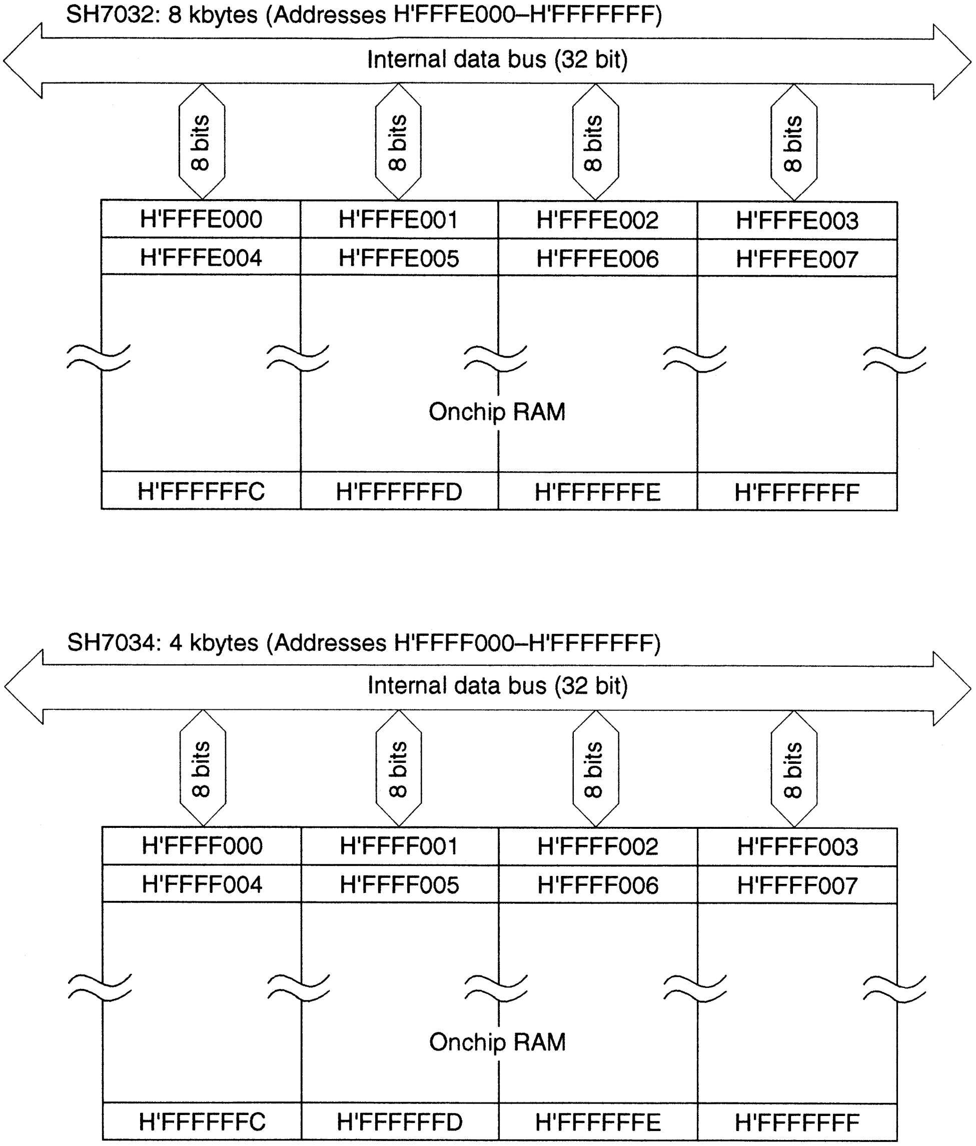
PAIOR, PBIOR*1	Data when Port Data Registers A and B Are Read	Data when Port Data Registers A and B Are Written	Data when Port Data Register C*2 is Read	Data when Port Data Register C*2 is Written
0	Pin value	Does not affect on pin status.	Pin value	Ignored (does not affect pin)
1	Port data register value	When the pin is set as a PFC output port, the value of the port data register is output by pin. Other settings do not affect pin status.		

Notes: 1. I/O registers of ports A and B (see section 5.7, Pin Function Controller)

2. The SH7032 and SH7034 have the port data register C (PCDR) and Port C; the SH7020 and SH7021 do not.

5.10 RAM

The SH series microprocessors have high-speed static onchip RAM. The RAM is linked to the CPU by a 32-bit data bus. The CPU can access data in the onchip RAM in byte, word, or longword units. The RAM can always be accessed in one state, making the RAM ideal for high-speed data transfers. The onchip RAM is allocated the addresses shown in figures 5.37 and 5.38. The onchip RAM space (addresses H'F000000 to H'FFFFFFF) is divided in 8-kbyte shadows on the SH7032 microprocessor and 4-kbyte shadows on the SH7034. For the SH7020 and SH7021, it is divided into 1-kbyte shadows. See table 5.18. All shadows can be accessed.



Note: The figures show the shadow address at the upper part of the RAM space.

Figure 5.37 RAM Block Diagram [SH7032, SH7034]

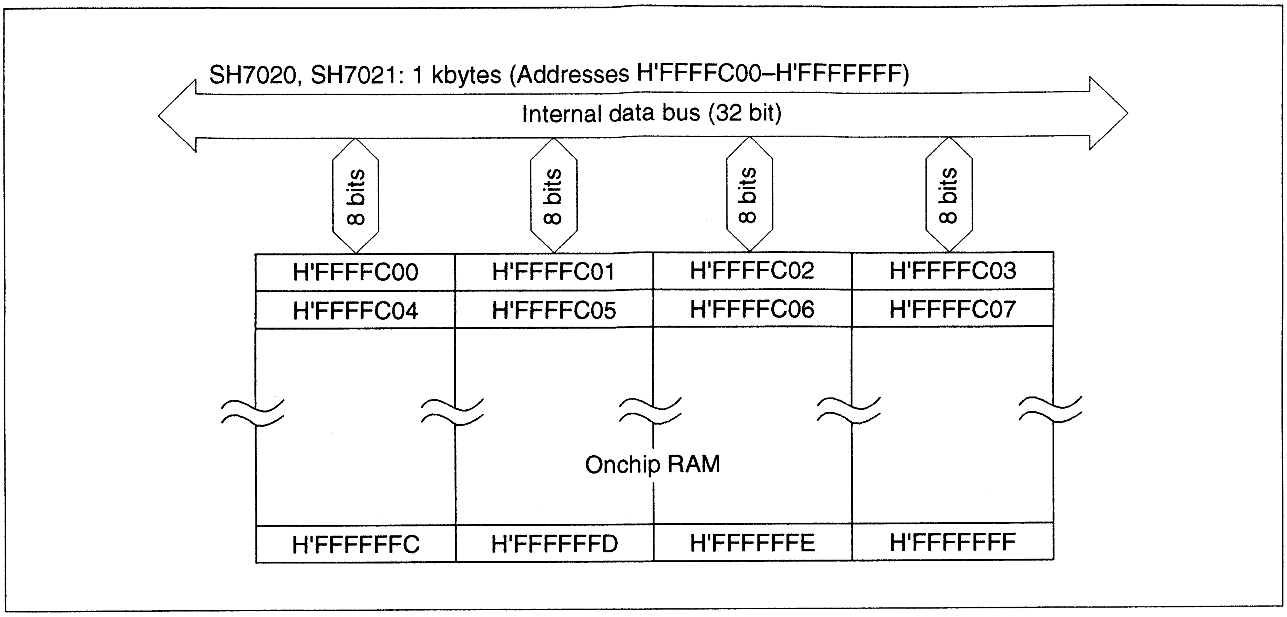


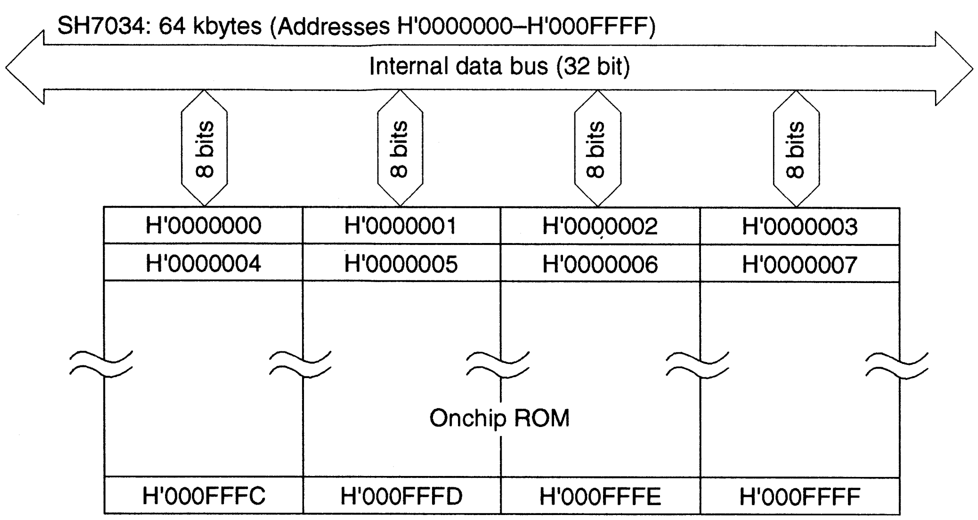
Figure 5.38 RAM Block Diagram [SH7020, SH7021]

Table 5.18 RAM Capacity, Addresses and Shadows

Product Name	Capacity	Addresses	Shadows (example)
SH7032	8 kbytes	H'FFFE000–H'FFFFFFF	H'F000000–H'F001FFF
SH7034	4 kbytes	H'FFFF000–H'FFFFFFF	H'F000000–H'F000FFF
SH7020, SH7021	1 kbyte	H'FFFFC00–H'FFFFFFF	H'F000000–H'F0003FF

5.11 ROM (PROM, Masked ROM)

The SH7034 has an onchip 64 kbyte PROM or masked ROM. The SH7020 has 16 kbytes of onchip masked ROM; the SH7021 has 32 kbytes of onchip masked ROM. It is connected to the CPU by a 32-bit bus, enabling high-speed access in one state. The SH7034 onchip ROM is allocated addresses H'0000000–H'000FFFFF. Since onchip ROM space is divided into shadows in 64-kbyte units, all shadows can be accessed (figure 5.39). The SH7020 is allocated addresses H'0000000–H'0003FFF and is divided into shadows in 32 kbyte units.


SH7034: 64 kbytes (Addresses H'0000000–H'000FFFF)
Internal data bus (32 bit)
8 bits
8 bits
8 bits
8 bits
Onchip ROM

H'0000000	H'0000001	H'0000002	H'0000003
H'0000004	H'0000005	H'0000006	H'0000007
H'000FFFC	H'000FFFD	H'000FFFE	H'000FFFF

Note: The figure shows shadow addresses at the lower part of the onchip ROM space

Figure 5.39 PROM Block Diagram

5.11.1 PROM Programming [SH7034]

Placing an SH7034 that has an onchip PROM in the PROM mode stops its MCU functions, allowing data to be written directly onto the onchip PROM. This allows the onchip PROM to be programmed in exactly the same way as HN27C101 EPROMs (Vpp = 12.5 V). Using a 112-pin to 32-pin adapter socket allows a commercial PROM to be used. The address range is H'00000–H'0FFFF. The data for the H'10000–H'1FFFF address area of the PROM writer should all be H'FF (figure 5.40).

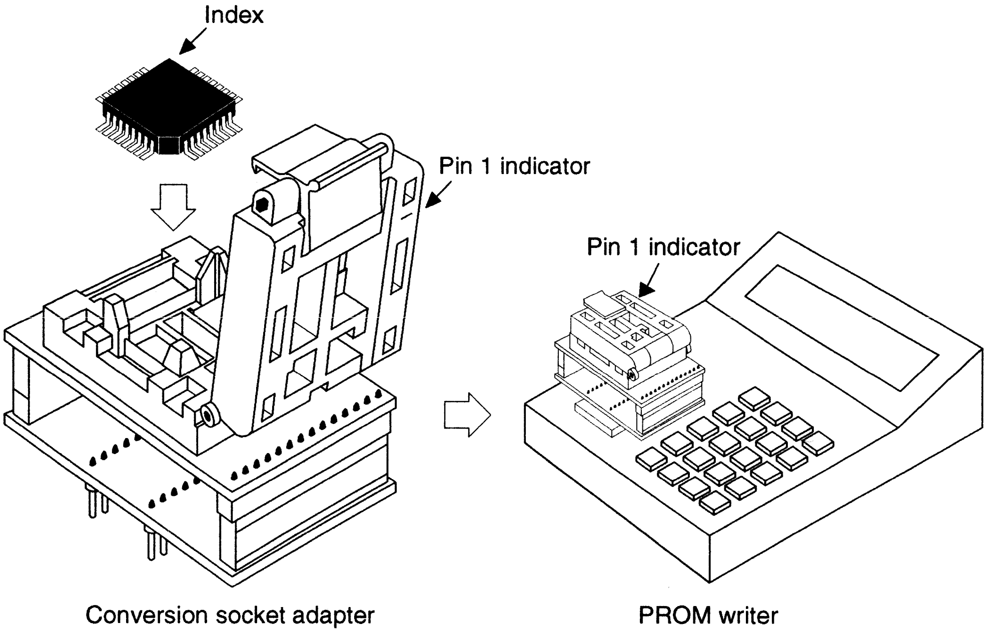
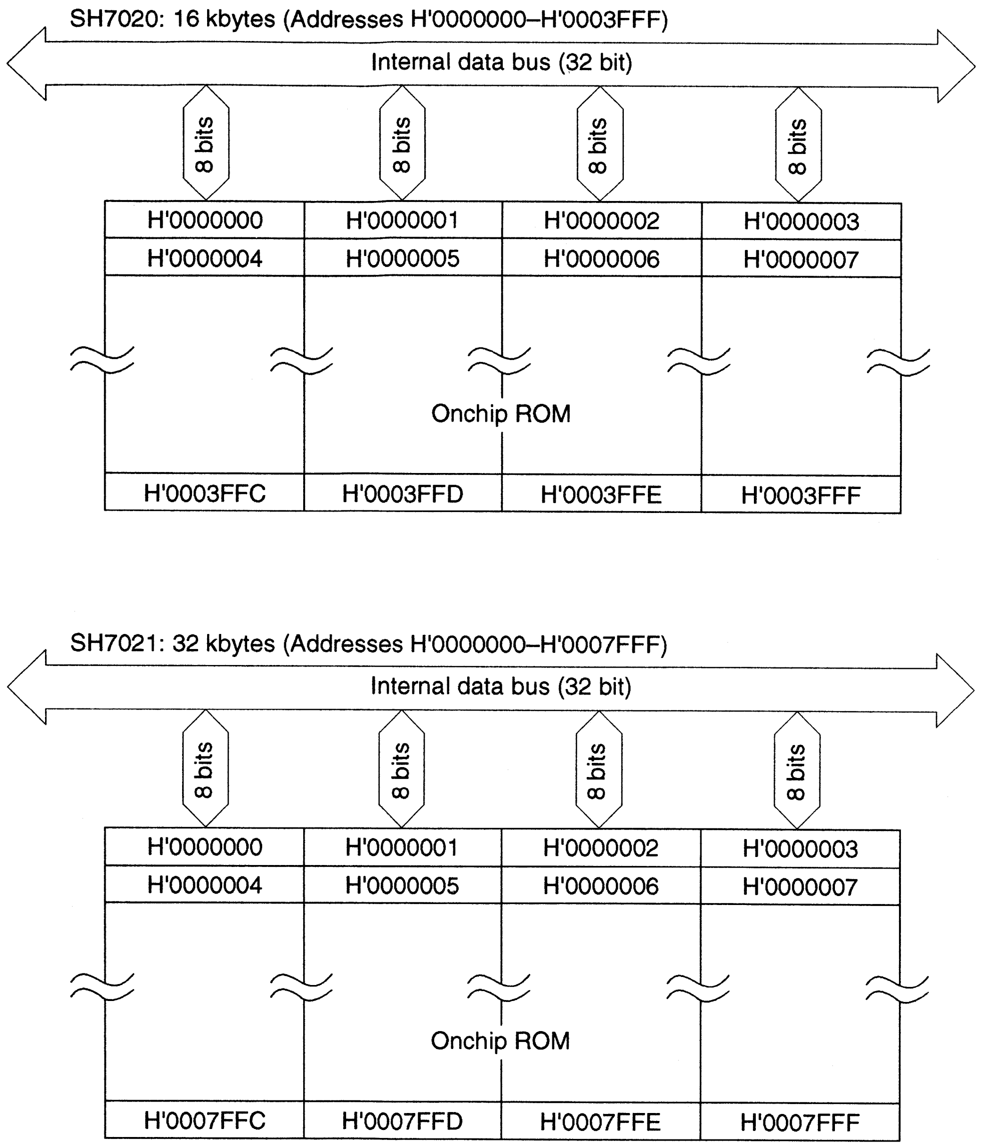


Figure 5.40 PROM Programming

5.11.2 ROM

Figure 5.41 shows the configuration of the masked ROM (table 5.19).



Note: The figures show the shadow address at the lower part of the ROM space.

Figure 5.41 Masked ROM Block Diagram

Table 5.19 ROM Capacity, Addresses and Shadows

Product Name	Capacity	Type	Addresses	Shadows (Example)	
SH7032	—	—	—	—	—
SH7034	64 kbytes	PROM or masked ROM	H'0000000–H'000FFFF	H'0FF0000–H'0FFFFFF	H'8000000–H'800FFFF
SH7020	16 kbyte	Masked ROM	H'0000000–H'0003FFF	H'0FFC000–H'0FFFFFF	H'8000000–H'8003FFF
SH7021	32 kbyte	Masked ROM	H'0000000–H'0007FFF	H'0FF8000–H'0FFFFFF	H'8000000–H'8007FFF

Section 6 Support Tools

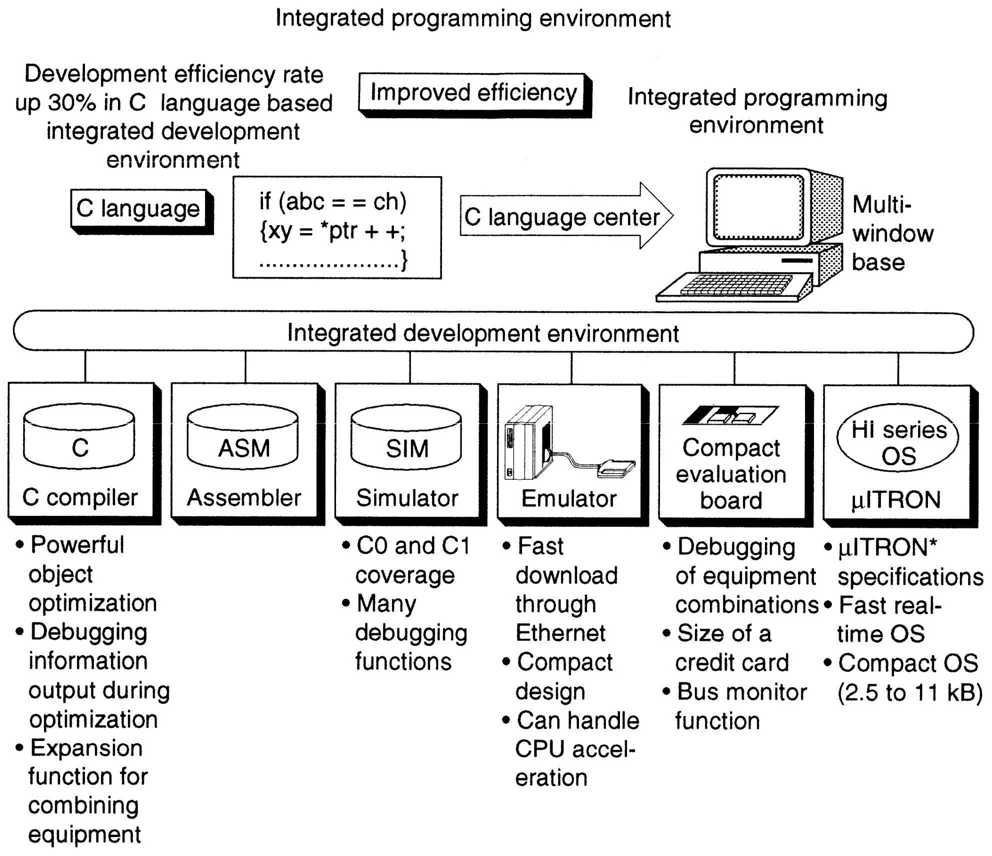
To improve the efficiency of development, the SH series has the following support tools:

- Software
  - Integrated programming environment
  - C compiler for the SH series
  - Assembler for the SH series
  - Linkage editor (including librarian and object converter)
  - Simulator/debugger for SH series
  - Real-time OS for SH series
- Hardware
  - SH series real-time emulator E7000
  - SH series compact evaluation board

6.1 Software

6.1.1 Integrated Programming Environment

Figure 6.1 shows components and features of the integrated programming environment.


Integrated programming environment
Development efficiency rate up 30% in C language based integrated development environment
Improved efficiency
Integrated programming environment
C language
if (abc = = ch)
{xy = *ptr + +;
.....................}
C language center
Multi-window base
Integrated development environment
C
C compiler
ASM
Assembler
SIM
Simulator
Emulator
Compact evaluation board
HI series OS
μITRON
• Powerful object optimization
• Debugging information output during optimization
• Expansion function for combining equipment
• C0 and C1 coverage
• Many debugging functions
• Fast download through Ethernet
• Compact design
• Can handle CPU acceleration
• Debugging of equipment combinations
• Size of a credit card
• Bus monitor function
• μITRON* specifications
• Fast real-time OS
• Compact OS (2.5 to 11 kB)

Note: μITRON is an abbreviation of "Micro Industrial TRON." ITRON is an abbreviation of "Industrial TRON." TRON is an abbreviation of "The Real Time Operating System Nucleus." This product was developed under the direction of Mr. Ken Sakamura of the Computer Science Section of the Tokyo University Physics Department and is based on TRON specifications.

Figure 6.1 Integrated Programming Environment

- The same system of operations and menus can be used for all tasks from editing through debugging
- Support for multiple windows: All information from program editing to debugging can be simultaneously displayed during development
- Simplified user operations: Operation has been simplified by adding an automatic make function after program editing, an auto editor start-up for when assembler/compiler errors occur, and an automatic loading function that works with the simulator or emulator during program construction
- Full-fledged help function

6.1.2 C Compiler

- Language specification is based on the C language standard proposal of the American National Standard Institute (ANSI)
- The optimization utility enables the object program size to be decreased for faster execution (ratio to assembler 1:2)

6.1.3 Assembler

- Common assembler language specifications with H series
- Objects are output in the SYSROF format

6.1.4 Linkage Editor (Including Librarian and Object Converter)

- The linkage editor links and relocates object programs output by the assembler and C compiler (SYSROF format) and creates load modules (SYSROF format)
- The librarian records user programs in the library
- The object converter converts the load module files in the SYSROF format output by the linkage editor into S-format files

6.1.5 SH Series Simulator Debugger

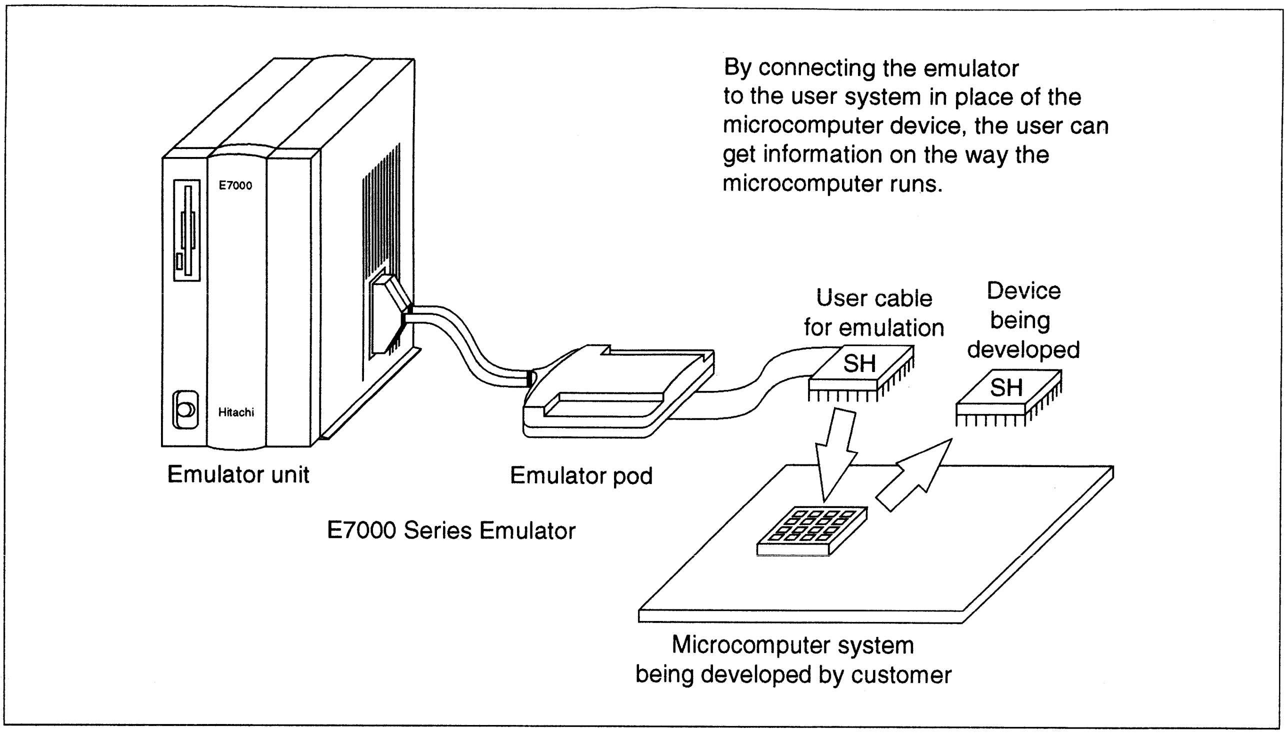
- Debugs even without the target system (run on host computer)
- Supports C source level debugging function
- Traces breaks related to registers
- Debugs programs in the initial development stage using the stab and function call utilities
- Runs simulations with relocatable objects

6.1.6 Real-Time OS for SH Series (HI-SH7)

- Easy to shift OS and application programs for H series microprocessors
- Fast OS processes task start-up system calls in 16 µs
- Supports a stack-sharing function between tasks

6.2 Hardware

6.2.1 Real-time emulator E7000



By connecting the emulator to the user system in place of the microcomputer device, the user can get information on the way the microcomputer runs.

Figure 6.2 E7000 Emulator

- Hardware break function (4 breakpoints)
  - — Break setting conditions: Address bus (bit mask and range specifications can be set), data bus (enables bit mask), $\overline{\text{RD/WR}}$ signal, external interrupt (NMI, $\overline{\text{IRQ0–IRQ7}}$), specify the number of times the condition is matched (maximum 4095), and delay count (maximum 32k cycle).
  - — Sequential breakpoints: When 2 to 4 set points are passed through in the specified order, program execution stops.
- PC break functions (2 Mbyte space, 1 Mbyte boundary, 255 breakpoints)
  - — Break setting conditions: Program counter, the specified number of condition matches

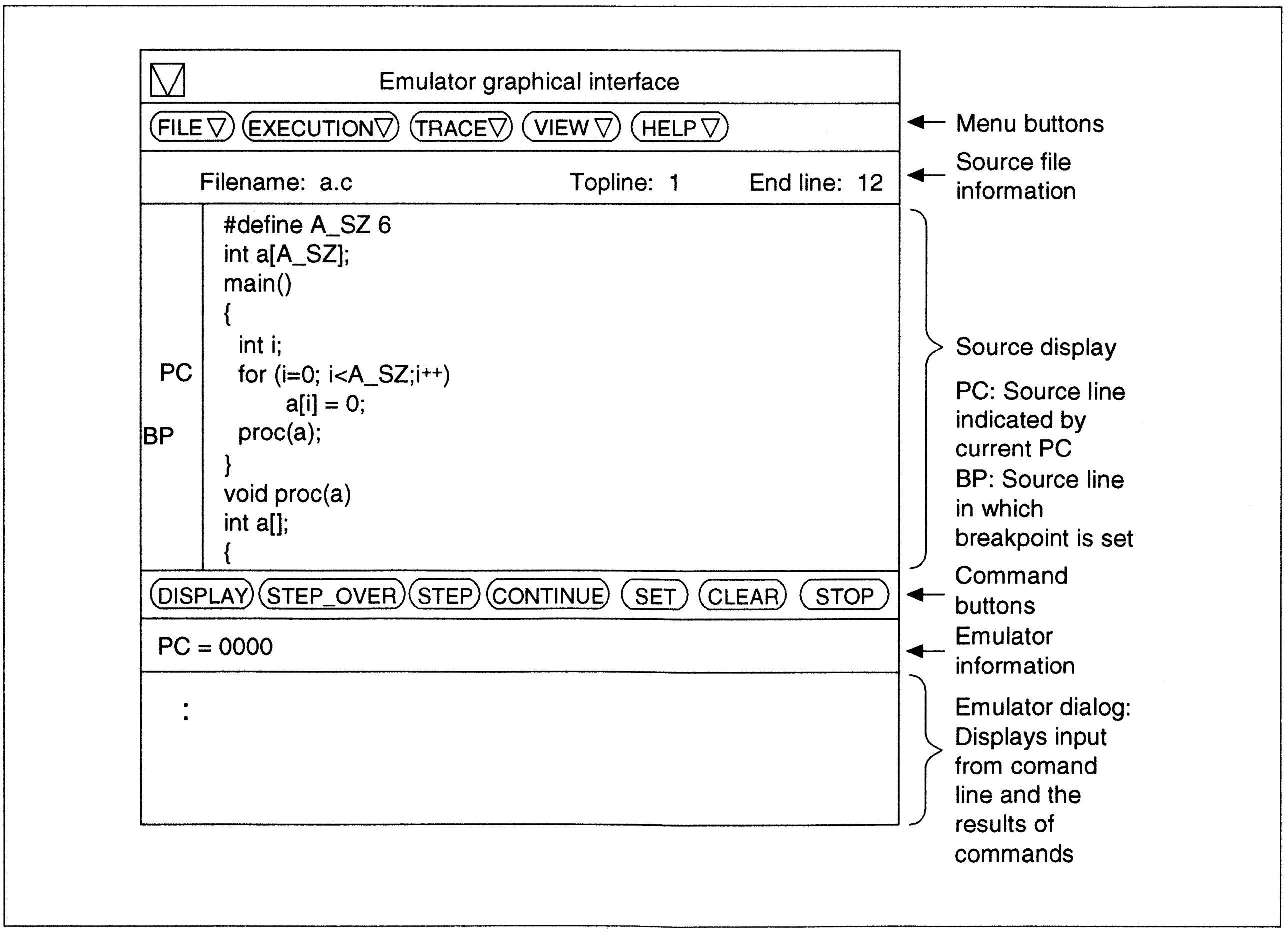
- Real-time trace function
  - Trace information: Address bus, data bus, external interrupt (NMI, $\overline{\text{IRQ0}}$ ~ $\overline{\text{IRQ7}}$), external probe signal, control signal
  - Trace information pick-up: Free trace (pick up all trace data while executing), subroutine trace (specify trace pick-up function) range specification (specify trace pick-up address range), stop trace (stop picking up trace when conditions are satisfied)
  - Trace data search, search for data in trace buffer under specified conditions
  - Set trigger conditions (when trigger conditions are satisfied, condition match output is output to the trigger pin)

- C0 coverage function (2 Mbyte space, 1 Mbyte boundary): Displays proportion of region transited during program execution.
- Performance analysis: Measures program execution time and frequency.
- Parallel mode: Enables memory command, trace display command input without halting operation when the program is running.

- E7000 Graphical Interface Software

Figure 6.3 shows the graphical user interface. The E7000:

- Debugs source-level code
- Displays the location on the source program where to halt execution (PC)
- Displays the contents of the variables selected by the source program
- Sets the breakpoints in the lines selected by the source program
- Executes code in source program line units (steps)
- Displays real-time trace data
  — Displays the trace information used in the source program
  — Displays the address bus, data bus, interrupt signal, external probe signal, etc. as waveforms
- Allows multi-windowing
  — Continuous display of base frame and subframe. For example, the register subframe can be displayed and then the contents of the register displayed updated when the step is executed.
- Simplifies operation, using the command subframe



Filename: a.c Topline: 1 End line: 12

```
#define A_SZ 6
int a[A_SZ];
main()
{
  int i;
  for (i=0; i<A_SZ;i++)
        a[i] = 0;
  proc(a);
}
void proc(a)
int a[];
{
```

PC = 0000

:

Menu buttons

Source file information

Source display

PC: Source line indicated by current PC

BP: Source line in which breakpoint is set

Command buttons

Emulator information

Emulator dialog: Displays input from comand line and the results of commands

Figure 6.3 Base Frame of Graphical Interface (Window at Startup)

6.2.2 SH Series Compact Evaluation Board

Figure 6.4 shows the SH Series Compact Evaluation Board. The evaluation board:

- Enables real-time emulation (Max 20 MHz)
- Allows onchip ROM area to be set in emulation RAM
- Maps firmware independently of user space, so all memory space can be released to the user
- Includes an interface for terminals (RS-232C I/F). Supports data transfers to and from host
- Connects to user equipment (using a special QFP adapter)

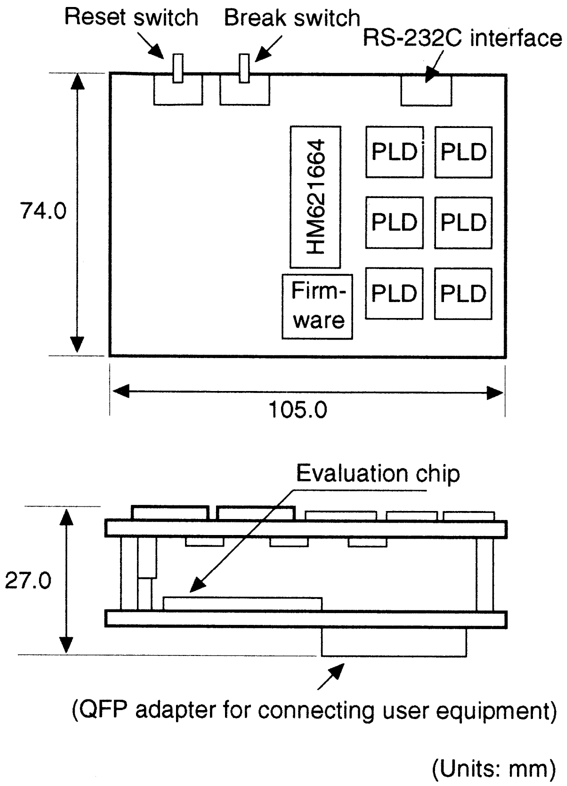


Figure 6.4 SH Series Evaluation Board